UNIVERSITY OF LIVERPOOL

Learning Semantic Segmentation with Weak and Few-shot Supervision

A thesis submitted in accordance with the requirements of the
University of Liverpool
for the degree of Doctor in Philosophy
by

Bingfeng Zhang

Department of Electrical Engineering and Electronics
School of Electrical Engineering and Electronics and
Computer Science
University of Liverpool

May, 2022

Abstract

Semantic segmentation, aiming to make dense pixel-level classification, is a core problem in computer vision. Requiring sufficient and accurate pixel-level annotated data during training, semantic segmentation has witnessed great progress with recent advances in deep neural network. However, such pixel-level annotation is time-consuming and highly relies on human effort, and segmentation performance dramatically drops on unseen classes or the annotated data is not sufficient.

In order to overcome the mentioned drawbacks, many researchers focus on learning semantic segmentation with weak and few-shot supervision, *i.e.*, weakly supervised semantic segmentation and few-shot segmentation. Specifically, weakly supervised semantic segmentation aims to make pixel-level classification with weak annotations (*e.g.*, bounding-box, scribble and image level) as supervision while few-shot segmentation attempts to segment unseen object classes with a few annotated samples. In this thesis, we mainly focus on image label supervised semantic segmentation, bounding-box supervised semantic segmentation, scribble supervised semantic segmentation and few-shot segmentation.

For weakly supervised semantic segmentation with image level annotation, current approaches mainly adopt a two-step solution, which generates pseudo pixel masks first that are then fed into a separate semantic segmentation network. However, these two-step solutions usually employ many bells and whistles in producing high-quality pseudo masks, making this kind of methods complicated and inelegant. We harness the image-level labels to produce reliable pixel-level annotations and design a fully end-to-end network to learn to predict segmentation maps. Concretely, we firstly leverage an image classification branch to generate class activation maps for the annotated categories, which are further pruned into tiny reliable object/background regions. Such reliable regions are then directly served as ground-truth labels for the segmentation branch, where both global information and local information sub-branch are used to generate accurate pixel-level prediction. Furthermore, a new joint loss is proposed that considers both shallow and high-level features.

For weakly supervised semantic segmentation with bounding-box level annotation,

most existing approaches rely on deep convolution neural network (CNN) to generate pseudo labels by initial seeds propagation. However, CNN-based approaches only aggregate local features, ignoring long-distance information. We proposed a graph neural network (GNN)-based architecture that takes full advantage of both local and long-distance information. We firstly transfer the weak supervision to initial labels, which are then formed into semantic graphs based on our newly proposed affinity Convolutional Neural Network. Then the built graphs are input to our graph neural network (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, 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. More importantly, our approach can be readily applied to bounding box supervised instance segmentation task or other weakly supervised semantic segmentation tasks, showing great potential to become a unified framework for weakly supervised semantic segmentation.

For weakly supervised semantic segmentation with scribble level annotation, the regularized loss has been proven to be an effective solution for this task. However, most existing regularized losses only leverage static shallow features (color, spatial information) to compute the regularized kernel, which limits its final performance since such static shallow features fail to describe pair-wise pixel relationship in complicated cases. We propose a new regularized loss which utilizes both shallow and deep features that are dynamically updated in order to aggregate sufficient information to represent the relationship of different pixels. Moreover, in order to provide accurate deep features, we adopt vision transformer as the backbone and design a feature consistency head to train the pairwise feature relationship. Unlike most approaches that adopt multi-stage training strategy with many bells and whistles, our approach can be directly trained in an end-to-end manner, in which the feature consistency head and our regularized loss can benefit from each other.

For few-shot segmentation, most existing approaches use masked Global Average Pooling (GAP) to encode an annotated support image to a feature vector to facilitate query image segmentation. However, this pipeline unavoidably loses some discriminative information due to the average operation. We propose a simple but effective self-guided learning approach, where the lost critical information is mined. Specifically, through making an initial prediction for the annotated support image, the covered and uncovered foreground regions are encoded to the primary and auxiliary support vectors using masked

GAP, respectively. By aggregating both primary and auxiliary support vectors, better segmentation performances are obtained on query images. Enlightened by our self-guided module for 1-shot segmentation, we propose a cross-guided module for multiple shot segmentation, where the final mask is fused using predictions from multiple annotated samples with high-quality support vectors contributing more and vice versa. This module improves the final prediction in the inference stage without re-training.

Key Words: semantic segmentation, weakly supervised, few-shot annotation, single stage, graph neural network, regularized loss, end-to-end, prototype, attention mechanism, global average pooling

Acknowledgement

Firstly, I would like to express my sincere gratitude to my advisor Prof. Jimin Xiao for the continuous support of my Ph.D study and related research, for his patience, motivation, and immense knowledge. His guidance helped me in all the time of research and writing of papers including this thesis. I could not have imagined having a better advisor and mentor for my Ph.D study. I would like to thank Prof. Eng Gee Lim, Prof. Kaizhu Huang, Prof. Shan Luo and Prof. Yunchao Wei, especially my IPAP advisors Prof. Ming Xu and Prof. Al-Nuaimy Waleed, for their insightful comments and encouragement, but also for the hard questions which incented me to widen my research from various perspectives.

I would like to express my sincere thanks to Suzhou Dinnar Automation Technology and National Nature Science Foundation of China, for their generous funding support to my research activities.

My sincere thanks also go to my labmates: Dr. Yanchun Xie, Dr. Samer Jammal, Mr. Dingyuan Zheng, Mr. Mingjie Sun, Mr. Xiaoyang Wang, Mr. Xinqiao Zhao, Mr. Jian wang, Mr. Junkun Peng, Ms. Hui Li, Ms. Siyue Yu. Without their precious support it would not be possible to conduct this research. I treasure the days being with all my friends in our Lab MMT408 and MMT410.

I thank my fellow labmates for the stimulating discussions, for the sleepless nights we were working together before deadlines, and for all the fun we have had in the last four years.

Last but not the least, I would like to thank my parents and wife for supporting me spiritually throughout writing this thesis. Without their support and love during the past few years, I could not finish this thesis.

Contents

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction

1.1 Background

Semantic segmentation has been applying widely in real scenario such as industrial defeat detection, auto automatic drive and medical image analysis. With recent advances in deep neural network especially Fully Convolutional Network (FCN) [4], semantic segmentation has been making great progress.

Previous approaches mainly focus on designing more powerful module to generate accurate prediction based on FCN architecture. For example, Deeplab-v2 [5] proposed an ASPP module which utilized dilated convolution to increase respective filed while Deeplab-v3+ [6] introduced an encoder-decoder structure to up-sample its prediction. PSPNet [7] designed a pyramid pooling module in an FCN architecture to generate more refined object details. SegSort [8] proposed a clustering method to segment objects. Tree-FCN [9] designed a learnable tree filter to utilize the structural property to model long-range dependencies.

Requiring sufficient and accurate pixel-level annotated data, previous state-of-the-art semantic segmentation approaches can produce satisfying segmentation masks. However, the current framework still has two main drawbacks: 1) In order to produce satisfying segmentation predictions, these approaches heavily rely on massive annotated data, which is time-consuming and highly relies on human effort. More importantly, in some real-world scenarios, generating high quality annotation is difficult. For example, it is hard to label medical images for normal data annotators. 2) Most current approaches can only remain high level performance on trained categories, once encountering unseen classes or insufficient annotated data, their performances will drops dramatically. While in most cases, it is not guaranteed that all classes are trained with massive annotated samples, even in some special cases such as defeat detection, some defeat annotations can only be generated with a few samples, which means that using such samples cannot effectively train a fully-supervised semantic segmentation model.

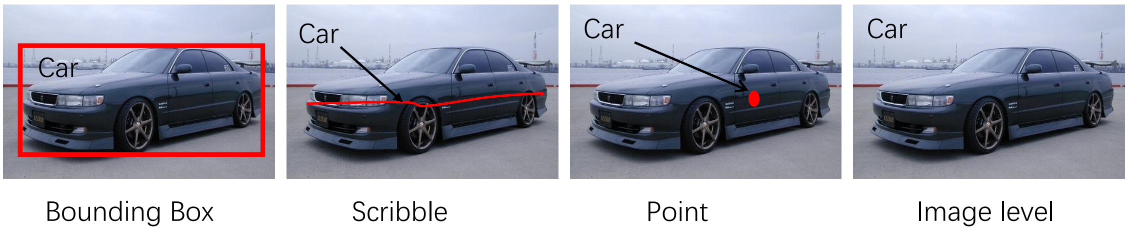

Fig. 1.1: Details of different weak supervisions. Scribble, bounding box and point labels are stronger supervision signals compared to the image-level class label since both class and localization information are provided. Whereas image-level labels only provide image class tags with the lowest annotation cost.

To tackle the aforementioned limitations, many researchers focus on two main challenging sub-tasks: weakly supervised semantic segmentation and few-shot segmentation. Specifically, weakly supervised semantic segmentation is to make high quality semantic segmentation with weak supervision, while few-shot segmentation concentrates on to segment on untrained categories.

For weakly supervised semantic segmentation, according to the level of provided weak annotations, the weak supervision can be divided into scribble level [10, 11, 12], bounding box level [13, 14, 15, 16], point level [17] and image level [18, 19, 20, 21]. In this thesis, we focus on **image level**, **bounding-box level** and **scribble** supervision. The details of different weak supervision can be found in Fig. 1.1

For few-shot segmentation, most approaches [22, 23, 24, 2, 25, 26] adopt a Siamese Convolutional Neural Network to encode both support and query images. In order to apply the information from support images, they mainly use masked Global Average Pooling (GAP) [27] or other strengthened methods [28] to extract all foreground [24, 2, 29] or background [24] as one feature vector, which is used as a prototype to compute cosine distance [30] or make dense comparison [2] on query images.

Both weakly supervised semantic segmentation and few-shot segmentation are the effective solutions to solve the case that lacking of the pixel-level labels to train the fully-supervised model. In the real scenario, it is common that the model cannot generate massive pixel-level annotation or the distribution of the annotation for different classes will be different, although weakly supervised semantic segmentation and few-shot segmentation adopt different detailed techniques, they are proposed to deal with the shortage of the supervision information.

1.2 Weakly Supervised Semantic Segmentation

1.2.1 Overview of 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 [10, 11, 12], bounding box label [13, 14, 15], point label [17] and image-level class label [18, 19, 20]. 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 image-level supervision, as shown in Fig. 1.2, class activation map (CAM) [27] is the common strategy that used as seeds to get pseudo labels. Such initial object seeds or regions are converted to generate pseudo labels to train a semantic segmentation model. For example, Wei *et al.* [31] proposed to erase iteratively the discriminative areas computed by a classification network so that more seed regions can be mined which are then combined with a saliency map to generate the pseudo pixel-level label. Wei *et al.* [32] also proved that dilated convolution can increase the receptive filed and improve the weakly segmentation network performance. Besides, Wang *et al.* [33] trained a region network and a pixel network to make predictions from image level to region level, and then to pixel level gradually. Also, this method takes saliency map as extra supervision. Moverover, Ahn and Suha [18] designed an affinity network to compute the relationship between different image pixels and exploited this network to get the pseudo object labels for segmentation model training. Huang *et al.* [34] deployed a traditional algorithm named seed growing to iteratively expand the seed regions.

For the bounding box supervision, SDI [14] used the segmentation proposal by combining MCG [35] with GrabCut [36] to generate the pseudo labels. Song *et al.* proposed a box-driven method [15], using box-driven class-wise masking and filling rate guided adaptive loss to generate pseudo labels. Box2Seg [37] attempt to design a segmentation network which is suitable to utilize the noisy labels as supervision.

For the scribble supervision, ScribbleSup [11] proposed to utilize super pixel [38] to expand initial annotation and design a loss function to use the expanded supervision. Tang *et.al.* [10, 12] proposed Normalized Cut loss and Kernel Cut loss to directly use initial labels as supervision. However, both Normalized Cut and Kernel Cut need multi-round training. Gated CRF loss [39] improves the efficiency of Kernel Cut loss through adding a gate operation. However, only relying static shallow feature cannot build accurate relationship for different pixels. SPML [40] used SegSort [8] as the backbone and

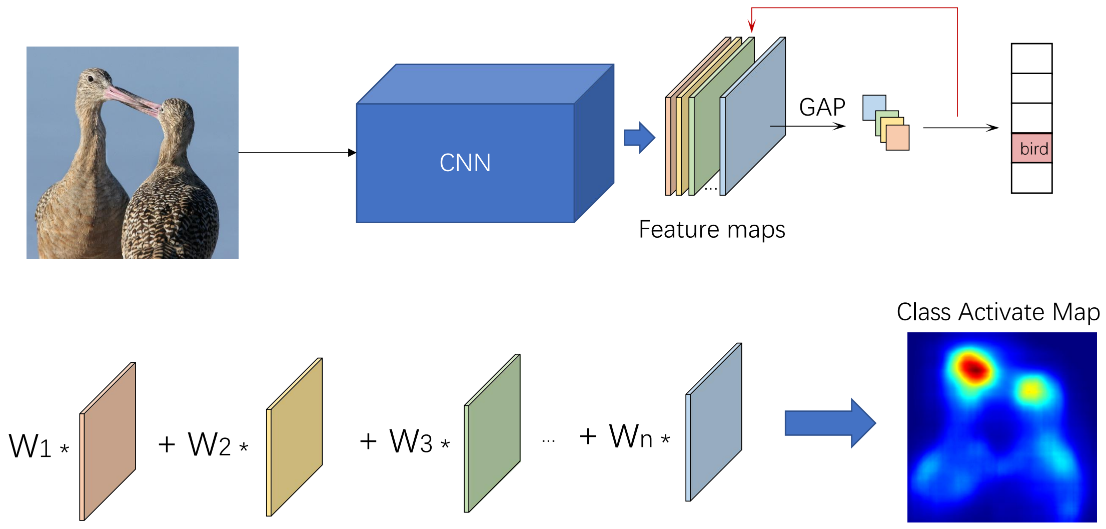

Fig. 1.2: Details of class activation map (CAM). CAM can locate some object regions after training a classification network with image level supervision. GAP: global average pooling. $W_1, W_2, ..., W_n$ are the weights of the final classifier for “bird”.

HED contour detector [41] as extra supervision. BPG [42] designed an iterative strategy to produce the fine-grained feature maps, which also applied contour detector [43, 41] to provide boundary supervision.

1.2.2 End-to-end Approach for Image-level Supervision

To learn semantic segmentation models using image-level labels as supervision, many existing approaches can be categorized as one-step approaches and two-step approaches. One-step approaches [44] often establish an end-to-end framework, which augments multi-instance learning with other constrained strategies for optimization. This family of methods is elegant and easy to implement. However, one significant drawback of these approaches is that the segmentation accuracy is far behind their fully supervised counterparts. To achieve better segmentation performance, many researchers alternatively propose to leverage two-step approaches [31, 34]. This family of approaches usually aim to take bottom-up [45] or top-down [46, 27] strategies to firstly generate high-quality pseudo pixel-level masks with image-level labels as supervision. These pseudo masks then act as ground-truth and are fed into the off-the-shelf fully convolutional networks such as FCN [4] and Deeplab [5, 47] to train the semantic segmentation models. Current state-of-the-arts are mainly two-step approaches, with segmentation performance approaching that of their fully supervised counterparts. However, to produce high-quality pseudo masks, these approaches often employ many bells and whistles, such as introducing additional object/background cues from object proposals [48] or saliency maps [49] in an off-line

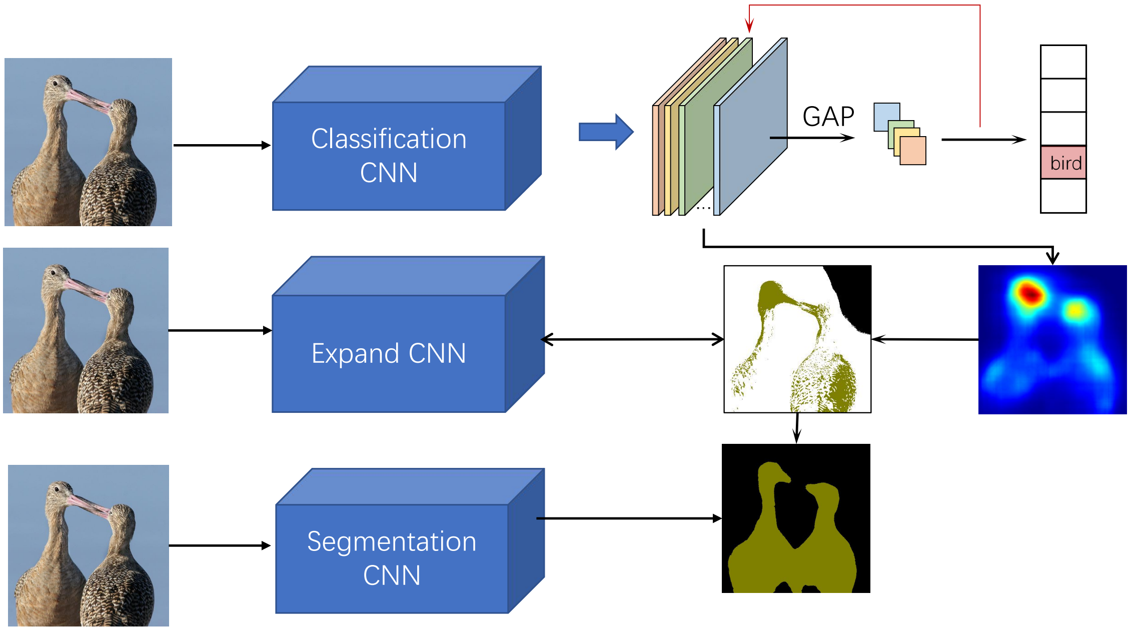

Fig. 1.3: The common framework of two steps approaches. The current two-step solution usually adopts several separate CNNs for the image label supervised semantic segmentation, which are usually very complicated and hard to be re-implemented, limiting their application to research areas such as object localization and video object tracking.

manner. Therefore, the two-step approaches are usually very complicated and hard to be re-implemented, limiting their application to research areas such as object localization and video object tracking. One common two-step framework, adopting in recent approaches [34, 21, 50, 51, 52, 53, 54] can be found in Fig. 1.3, which contains three individual networks for solving this task.

We present a simple yet effective one-step approach, which can be easily trained in an end-to-end manner. It achieves competitive segmentation performance compared with two-step approaches. Our approach named Reliable Region Mining (RRM) includes two branches: one to produce pseudo pixel-level masks using image-level annotations, and the other to produce the semantic segmentation results. In contrast to the previous two-step methods [18, 55, 56, 57] that prefer to mine dense and integral object regions, our RRM only leverages those reliable object/background regions that are usually tiny but with high response scores on the class activation maps. We find these regions can be further pruned into more reliable ones by augmenting an additional Conditional Random Field (CRF) operation, which are then employed as supervision for the parallel semantic segmentation branch. We design two parallel sub-branches for the segmentation branch: one extracts local information using the regular convolution layer, the other extracts global information with our proposed Re-weighting Feature-Attention Module (R-FAM). More importantly, with limited pixels as supervision, we design a new joint training loss, including a pixel-wise cross-entropy loss, a regularized loss named dense energy loss and a Batch-based Class Distance loss (BCD loss) to optimize the training process. We introduce the dense energy loss to use the shallow features such as RGB color and spatial information, and BCD loss to make the high-level semantic features more discriminative for different classes.

Our one-step RRM achieves 65.4% and 65.3% of mIoU scores on the Pascal VOC *val* and *test* sets, respectively. These results achieve the state-of-the-art performance, and are even competitive compared with those two-step state-of-the-arts, which usually adopt complex bells and whistles to produce pseudo masks. We believe that our proposed RRM offers a new insight to the one-step solution for weakly supervised semantic segmentation. Furthermore, in order to show the effectiveness of our method, we also extend our method to a two-step framework and get a new state-of-the-art performance with 69.3% and 69.2% on the Pascal VOC *val* and *test* sets.

1.2.3 Graph-based Approach for Box-level Supervision

For utilizing bounding-box as supervision, Most previous practices [13, 14, 15, 44] use object proposals [58, 59] to provide some seed labels as supervision. These methods follow a common pipeline of employing object proposals [58, 59] and CRF [60] 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 [1] attempts to use Graph Convolutional Network (GCN) [61] for the bounding-box supervised semantic segmentation . They convert images to unweighted graphs by grouping pixels in a superpixel to a graph node [38]. 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 [1] 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 [1] 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

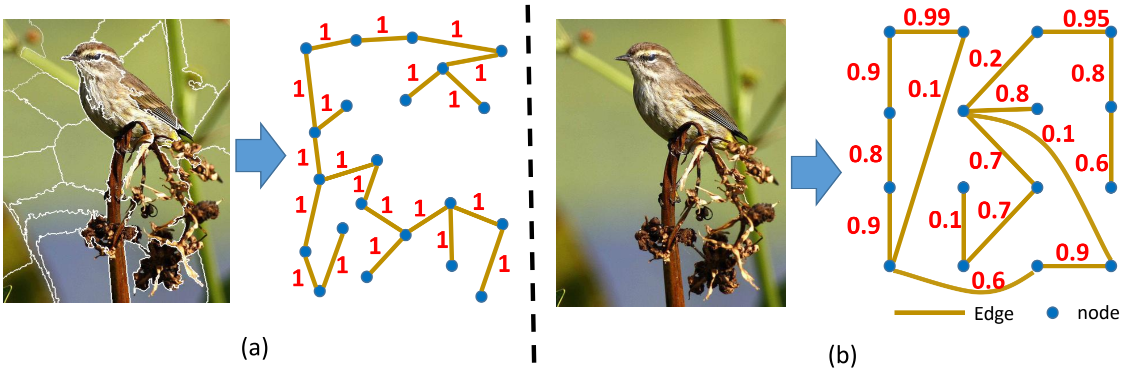

Fig. 1.4: The difference between our built graph and that of previous approach [1]. (a) Superpixel based approach [1]. (b) Our approach. The numbers along the edges indicate the edge values, soft edge allows any edge weights between 0 and 1.

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 [1] only considers the initial edges for feature propagation. Thus, if the strong 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^{2}GNN) 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. 1.4 shows the difference between our built graph and that of the previous approach [1]. It can be seen that the previous approach only considers locally connected nodes, and they build an unweighted graph based on superpixel [38], while we consider both local and long distance edges, and the built weighted graph views one pixel as one node.

Secondly, 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 pair-wise 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^{2}GNN. 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^{2}GNN, 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^{2}GNN, 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^{2}GNN 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.

1.2.4 Dynamic Feature Regularized Loss for Scribble Supervision

For scribble supervised semantic segmentation, most recent state-of-the-art approaches can be divided into two main categories: pseudo-label based approaches [1, 62] and loss function based approaches [10, 12, 40, 42]. Pseudo-label based approaches focus on generating more pseudo labels through expanding the initial annotations so that the segmentation model receives more completed pixel-level labels as supervision. But such approaches usually need multi-stage training process with many bells and whistles. For example, in A^{2}GNN [62], three different models are used for this task. Loss function based approaches concentrate on directly utilizing limited labels to train the segmentation model with well-designed loss functions. However, some approaches [40, 42] rely on extra dataset [43, 41] to provide edges or boundaries information as supervision, while some loss function based approaches [10, 12] still need multi-round training procedures. Although Gated CRF loss [39] can be directly trained in an end-to-end manner, its performance is limited as it solely relies on static shallow feature (color and spatial information), which fails to capture accurate pair-wise pixel relationship. For example, the shallow features are similar for a pixel pair which belongs to different objects with similar color and close spatial positions (*e.g.*, a white dog close to a white cat). In this case, the shallow features cannot accurately describe the semantic relationship of different pixels. Using such information to compute the regularized loss enforces the network to be optimized towards an inaccurate direction. More importantly, since shallow features are static, such process can not be corrected in the whole training period. Therefore, it is important to introduce more comprehensive representations for the regularized loss.

We propose a new Dynamic Feature Regularized (DFR) loss function in the semantic segmentation head to overcome the aforementioned drawbacks. Our DFR loss makes full use of both static shallow feature and dynamic deep feature, which provides more sufficient information to describe the semantic similarity of different pixels. However, pixel features from the same semantic category may not be sufficient similar, so we design a feature consistency head to enforce this goal. Our feature consistency head utilizes the highly confident prediction from our semantic segmentation head as supervision. It closes up feature distance for pixels from the same semantic category and widens feature distance for pixels from different categories.

Our semantic segmentation head and feature consistency head are directly coupled as they enhance each other mutually. On one hand, deep feature from our feature consistency head provides a third dimension of input for the regularized loss of the semantic segmentation head, so as to produce accurate semantic prediction. On the other hand, accurate semantic prediction provides more reliable supervision for the feature consistency head, empowering it to build more discriminative features. As a result, compared to solely relying on static shallow feature to compute regularized kernel, the interaction between the two heads allows the deep feature to dynamically change, which also enables deep feature level self-correction and mitigates the negative influence of the inaccurate shallow feature.

Meanwhile, in order to keep high computational efficiency for our loss functions, a local window is used to restrict the loss computing region. Thus, in order to provide more comprehensive information, we adopt vision transformer [63, 64] as our backbone since such model can extract global feature representations.

Our approach can be directly trained in an end-to-end manner and it does not rely on any extra dataset to provide supervision. Without applying any post-processing method such as dense CRF [60] to refine the results, our approach significantly outperforms the previous state-of-the-art approaches, with an mIoU increase of more than 6%.

1.3 Few-shot Segmentation

1.3.1 Overview of Few-Shot Segmentation

For few-shot segmentation, compared to fully supervised semantic segmentation [47, 65, 66, 67] which can solely segment the same classes in the training set, the objective of few-shot segmentation is to utilize one or a few annotated samples to segment new classes. The data for few-shot segmentation is divided into two sets: support set and query set. This task requires to segment images from the query set given one or several annotated images from the support set. Thus, the key challenge of this task is how to leverage the information from the support set.

Most previous approaches adopt a metric learning strategy [68, 69, 70, 71, 72] for

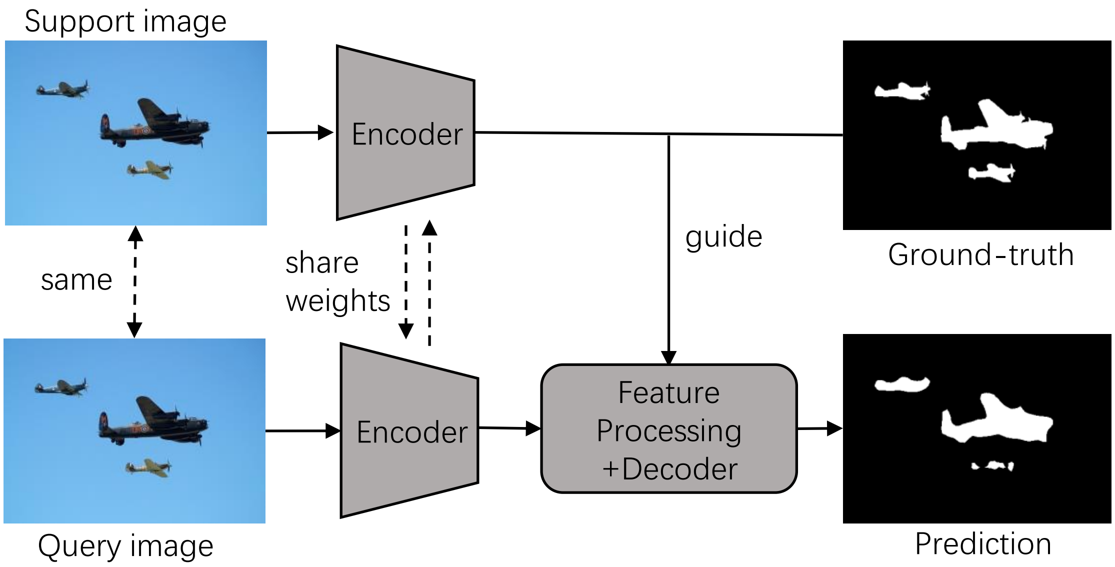

Fig. 1.5: Motivation of our approach for few-shot segmentation. Even using the same image as both support and query input, previous approaches cannot generate accurate segmentation under the guide of its ground-truth mask.

few-shot segmentation. For example, In PL [22], a two-branch prototypical network was proposed to segment objects using metric learning. SG-One [30] proposed to compute a cosine similarity between the generated single support vector and query feature maps to guide the segmentation process. CANet [2] designed a dense comparison module to make comparisons between the support vector and query feature maps. PANet [24] introduced a module to use the predicted query mask to segment the support images, where it still relied on the generated support vector. FWB [28] tried to enhance the feature representation of generated support vector using feature weighting while CRNet [29] focused on utilizing co-occurrent features from both query and support images to improve the prediction, and it still used a support vector to guide the final prediction. PPNet [23] tried to generate prototypes for different parts as support information. PFENet [3] designed a multi-scale module as decoder to utilize the generated single support vector.

1.3.2 Self-guided and Cross-guided Learning

As mentioned before, Using a support feature vector extracted from the support image does facilitate the query image segmentation, but it does not carry sufficient information. Fig. 1.5 shows an extreme example where the support image and query image are exactly the same. However, even the existing best performing approaches fail to accurately segment the query image. We argue that when we use masked GAP or other methods [28] to encode a support image to a feature vector, it is unavoidable to lose some useful information due to the average operation. Using such a feature vector to guide the segmentation cannot make a precise prediction for pixels which need the lost information as support. Furthermore, for the multiple shot case such as 5-shot segmentation, the common practice is to use the average of predictions from 5 individual support images as the final prediction [30] or the average of 5 support vectors as the final support vector [24]. However, the quality of different support images is different, using an average operation forces all support images to share the same contribution.

We propose a simple yet effective Self-Guided and Cross-Guided Learning approach (SCL) to overcome the above mentioned drawbacks. Specifically, we design a Self-Guided Module (SGM) to extract comprehensive support information from the support set. Through making an initial prediction for the annotated support image with the initial prototype, the covered and uncovered foreground regions are encoded to the primary and auxiliary support vectors using masked GAP, respectively. By aggregating both primary and auxiliary support vectors, better segmentation performances are obtained on query images.

Enlightened by our proposed SGM, we propose a Cross-Guided Module (CGM) for multiple shot segmentation, where we can evaluate prediction quality from each support image using other annotated support images, such that the high-quality support image will contribute more in the final fusion, and vice versa. Compared to other complicated approaches such as the attention mechanism [2, 73], our CGM does not need to re-train the model, and directly applying it during inference can improve the final performance. Extensive experiments show that our approach achieves new state-of-the-art performances on PASCAL-5^i and COCO-20^i datasets.

1.4 Overview of This Thesis

1.4.1 Main Contributions

The major contributions of the research reported in this thesis are summarized as follows:

- We design an elegant and efficient end-to-end network for weakly supervised semantic segmentation. Relying on tiny reliable pixel-level pseudo labels, our network can be trained in a one-stage manner given image-level labels, without bells and whistles. For achieving this, We firstly propose two new loss functions for utilizing the reliable labels, including a new dense energy loss and a batch-based class distance (BCD) loss. The former relies on shallow features, whilst the latter focuses on distinguishing high-level semantic features for different classes. Besides, We design a new attention module (R-FAM) to extract comprehensive global information. By using a re-weighting technique, our R-FAM can suppress dominant or noisy attention values. Thus our semantic segmentation branch can aggregate sufficient global information. Our end-to-end approach achieves competitive performance compared to other two-step approaches on PASCAL VOC 2012 dataset. By extending our network to a two-step solution, our approach achieves a new state-of-the-art performance

- 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. Specifically, 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. Moreover, We propose a new GNN, A^{2}GNN, 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 Bounding-box Supervised Semantic Segmentation as well as Bounding-box Supervised Instance Segmentation on PASCAL VOC 2012 and COCO. 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.

- We propose a new dynamic feature regularized loss for weakly supervised semantic segmentation. Our regularized loss combines both static shallow and dynamic deep features for the regularized kernel, which can better represent the pair-wise pixel relationship. Meanwhile, we design a new feature consistency head to produce consistent features for pixels of same semantic category, enabling to build more accurate pair-wise pixel relationship. Meanwhile, we introduce vision transformer to strengthen the feature representation. To the best of our knowledge, this is the first work that uses transformer architecture for this task. Our approach achieves state-of-the-art performances on PASCAL VOC 2012 (*val*: 82.8%, *test*: 82.9%) and PASCAL CONTEXT (*val*: 52.9%), outperforming other approaches by a large margin (more than 6% and 12% mIoU increases on PASCAL VOC 2012 and PASCAL CONTEXT, respectively).

- We observe that it is unavoidable to lose some useful critical information using the average operation to obtain the support vector for few-shot segmentation. To mitigate this issue, we propose a self-guided mechanism to mine more comprehensive support information by reinforcing such easily lost information, thus accurate segmentation mask can be predicted for query images. Meanwhile, We also propose a cross-guided module to fuse multiple predictions from different support images for the multiple shot segmentation task. Without re-training the model, it can be directly used during inference to improve the final performance. Our approach can be applied to different baselines to improve their performance directly. Using our approach achieves new state-of-the-art performances on PASCAL-$5^i$and COCO-20^i datasets.

1.4.2 Brief Summary of the Remaining Chapters

In this chapter, the final summary of this thesis will be presented, followed by the future work for the research in relevant domains.

Chapter 2: In this chapter, we will introduce our proposed single-stage framework for weakly supervised semantic segmentation with image-level annotation. Our approach includes two branches: one to produce pseudo pixel-level masks using image-level annotations, and the other to produce the semantic segmentation results. Besides, We also propose two new loss functions for utilizing the reliable labels, including a new dense energy loss and a batch-based class distance (BCD) loss. The former relies on shallow features, whilst the latter focuses on distinguishing high-level semantic features for different classes. We design a new attention module (R-FAM) to extract comprehensive global information. By using a re-weighting technique, our R-FAM can suppress dominant or noisy attention values. Thus our semantic segmentation branch can aggregate sufficient global information. All of them will be introduced in this chapter and extensive experiments will be evaluated to show the effectiveness of our proposed method in this chapter. Finally, a short conclusion will be given to make a summary of this chapter. This chapter mainly comes from our two papers: the first one is “Reliability does matter: An End-to-end Weakly Supervised Semantic Segmentation Approach” [74]. The other is “End-to-End Weakly Supervised Semantic Segmentation with Reliable Region Mining” [75].

Chapter 3: In this chapter, we will show a new framework that effectively combines the advantage of CNN and GNN for weakly supervised semantic segmentation with bounding-box annotation. Specifically, We will firstly show how to convert the weakly supervision to initial seed labels, then the designed affinity CNN network will be illustrated, which aims to convert a given image to an irregular graph, where the graph node features and the node edges are generated simultaneously. Moreover, We will show the details of our proposed new GNN, A^{2}GNN, where we design a new GNN layer that can effectively mitigate inaccurate feature propagation through information aggregation based on edge weights and node attention. After that, We will explain the proposed new loss function (MP loss), which aims to mine extra reliable labels using the bounding box constraint and remove existing label noise by consistency-checking. Finally, a larger number of experimental results will be shown and there will be a conclusion to make a short summary. This chapter mainly contains our work "Affinity Attention Graph Neural Network for Weakly Supervised Semantic Segmentation" [62].

Chapter 4: In this chapter, we will firstly introduce the whole framework for weakly supervised semantic segmentation with scribble annotation, including a vision transformer as the backbone, a semantic segmentation head and a feature consistency head. Then we will introduce the details of these two heads, Specifically, the semantic segmentation head utilizes two loss functions: partial cross-entropy loss and our proposed dynamic feature regularized loss. Partial cross-entropy loss uses the scribble-annotation as supervision while our proposed dynamic feature regularized loss applies the original image information and feature map from the feature consistency head to produce regularized kernel. The feature consistency head also introduces two loss functions: feature distance loss and feature regularized loss. Feature distance loss uses the predicted highly confident pseudo labels from the semantic segmentation head as supervision. Feature regularized loss solely uses the shallow feature as the kernel to compute feature distance. In this chapter, the main work is from our work: "Dynamic Feature Regularized Loss for Weakly Supervised Semantic Segmentation" [76].

Chapter 5: In this chapter, we will firstly show our observation that it is unavoidable to lose some useful critical information using the average operation to obtain the support vector for few-shot segmentation. Then, we will introduce our proposed approach in order to mitigate this issue through mining more comprehensive support information. After that, we will show how to fuse multiple predictions from different support images for the multiple shot segmentation task. Finally, there will be some experiments to evaluate the effectiveness of our approach and a conclusion section to make a short summary. This chapter includes our previous work "Self-Guided and Cross-Guided Learning for Few-Shot Segmentation", published in CVPR 2021 [77].

Chapter 6: In this chapter, we will make a brief summary relating to the aforementioned research works, and based on this, I will attempt to give a discussion about the possible directions/works in the future, including exploring the probability of applying the vision transformer for weakly supervised semantic segmentation and utilizing the background information for few-shot segmentation.

For each of these chapters mentioned above, we have tried to make them self-contained. Therefore, some of the crucial contents, demonstrations, model definitions and illustrations might be reiterated in the following chapters when necessary.

Chapter 2

End-to-end Approach for Image label Supervised Semantic Segmentation

2.1 Motivation

As mentioned in Sect. 1.2.2, most recent approaches [34, 21, 50, 51, 52, 53, 54] leverage two-step approaches. These approaches usually aim to firstly generate high-quality pseudo pixel-level masks based on class activation map [27] with one or two individual CNNs are employed. After that, the generated pseudo labels are act as ground-truth and are fed into the off-the-shelf fully convolutional networks such as FCN [4] and Deeplab [5, 47] to train the semantic segmentation models. However, to produce high-quality pseudo masks, these approaches often employ many bells and whistles, such as introducing additional object/background cues from object proposals [48] or saliency maps [49] in an off-line manner. Therefore, the two-step approaches are usually very complicated and hard to be re-implemented, limiting their application to research areas such as object localization and video object tracking.

In order to overcome the drawbacks of the two-step solution, we propose a end-to-end framework named Reliable Region Mining (RRM) for this task. To achieve this, our framework mainly includes two parallel branches: one is to online mine reliable pseudo pixel-level masks using image-level annotations, and the other to produce the semantic segmentation results using the mined pixel-level masks. In the following parts, we will give the details of our RRM, including the architectures of our dual branches and the loss functions.

This chapter mainly comes from our two papers: the first one is “Reliability does matter: An End-to-end Weakly Supervised Semantic Segmentation Approach” [74]. The other is “End-to-End Weakly Supervised Semantic Segmentation with Reliable Region Mining” [75].

2.2 Proposed method

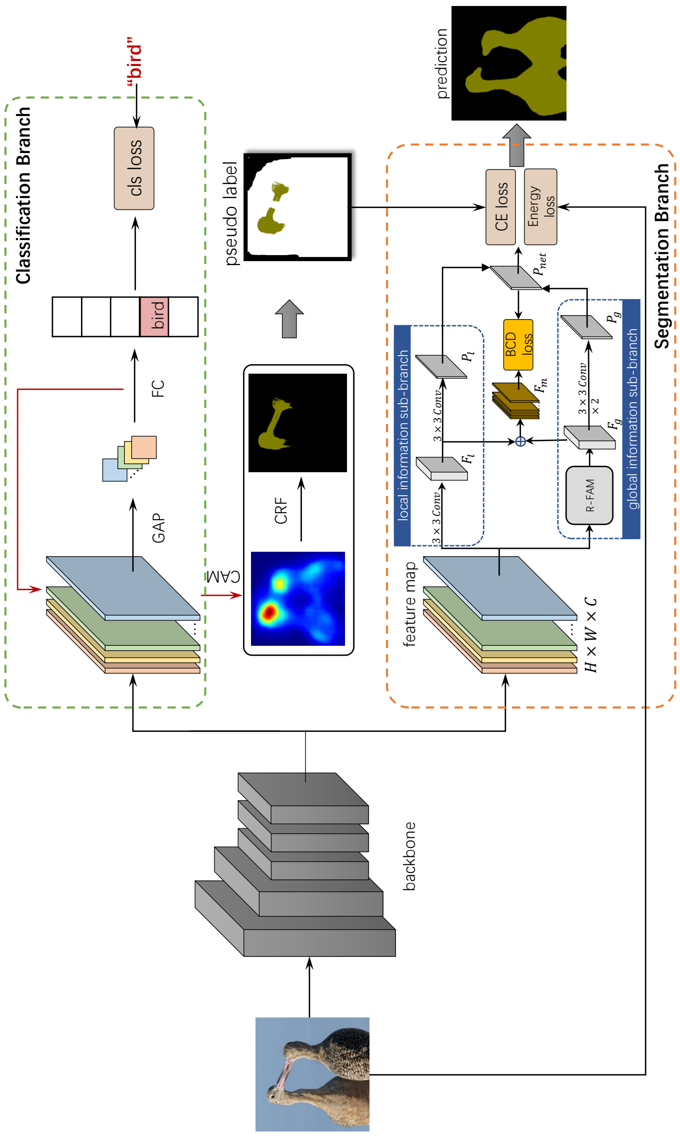

Fig. 2.1: The framework of our proposed RRM network. First of all, original regions are calculated through the classification branch, then the pseudo pixel-level masks are generated. Finally, the pseudo labels are applied as supervision to train the semantic segmentation branch. In the segmentation branch, two parallel sub-branches are used to extract local and global information, respectively. The whole RRM is jointly optimized end-to-end via a standard back-propagation algorithm during training.

2.2.1 Overview

Our proposed RRM can be divided into two parallel branches including a classification branch and a semantic segmentation branch. Both branches share the same backbone network, and during training, both of them update the whole network at the same time. The overall framework of our method is illustrated in Fig. 2.1. The algorithm flow is illustrated in Algorithm 1.

- The classification branch is used to generate reliable pixel-level annotations. Original CAMs will be processed to generate tiny reliable regions. The final remained reliable regions are regarded as labeled regions, while the other regions as unlabeled. These labels are used as supervision information for the semantic segmentation branch for training.

- The semantic segmentation branch is used to predict pixel-level labels. We designed two parallel sub-branches, one is named local information sub-branch, which is used to extract local features using regular convolutional layers. The other one is named global information sub-branch, which is used to extract global features using our newly designed R-FAM.
- The overall loss function of our RRM is: $\mathcal{L} = \mathcal{L}_{class} + \mathcal{L}_{joint\_seg}$, where $\mathcal{L}_{class}$ represents a conventional classification softmax loss [18, 57], while $\mathcal{L}_{joint\_seg}$ is a newly introduced joint loss for the segmentation branch, including a pixel-wise cross-entropy loss, a newly designed dense energy loss and a novel batch-based class distance (BCD) loss. The cross-entropy loss mainly considers labeled pixels, the dense energy loss takes into account all pixels by making full use of RGB color and pixel positions, and the BCD loss is used to make the high-level semantic features more discriminative for different classes.

2.2.2 Classification Branch: Generating Labels for Reliable Regions

High-quality pixel-level annotation has a direct impact on our final semantic segmentation performance as it is the only ground-truth in the training processing. Original CAMs can highlight the most discriminative regions of an object, but they still contain some non-object areas, which are the mislabeled pixels. Therefore, after getting the original CAM regions, post-processing such as dense CRF [60] is needed. We followed this basic idea and do further process for generating the reliable labels.

We compute the initial CAMs of the training dataset following [27]. In our network, Global Average Pooling (GAP) is applied to the last convolution layer, the output of which is classified with a fully-connected layer. Finally, the fully-connected layer weights are used on the last convolution layer to obtain the heatmap for each class. Besides, inspired by the fact that dilated convolution can increase the respective field [32], we add dilated convolution into the last three layers. Details of our network settings are reported in Section 2.3.

Mathematically, given an image *I*, the CAM of class c M_{ocam}^{c} is:

$$M_{ocam}^{c} = \mathrm{RS}(\sum_{ch=1}^{D} \omega_{ch}^{c} \cdot F_{ch}), (c \in C_{fg}), \tag{2.1}$$

where D is the channel dimension of F_{ch}, $C_{fg} = \{c_1, c_2, ..., c_N\}$ includes all foreground classes, ω_{ch}^{c} denotes the weights of the fully-connected layer for class c, and F_{ch} is the

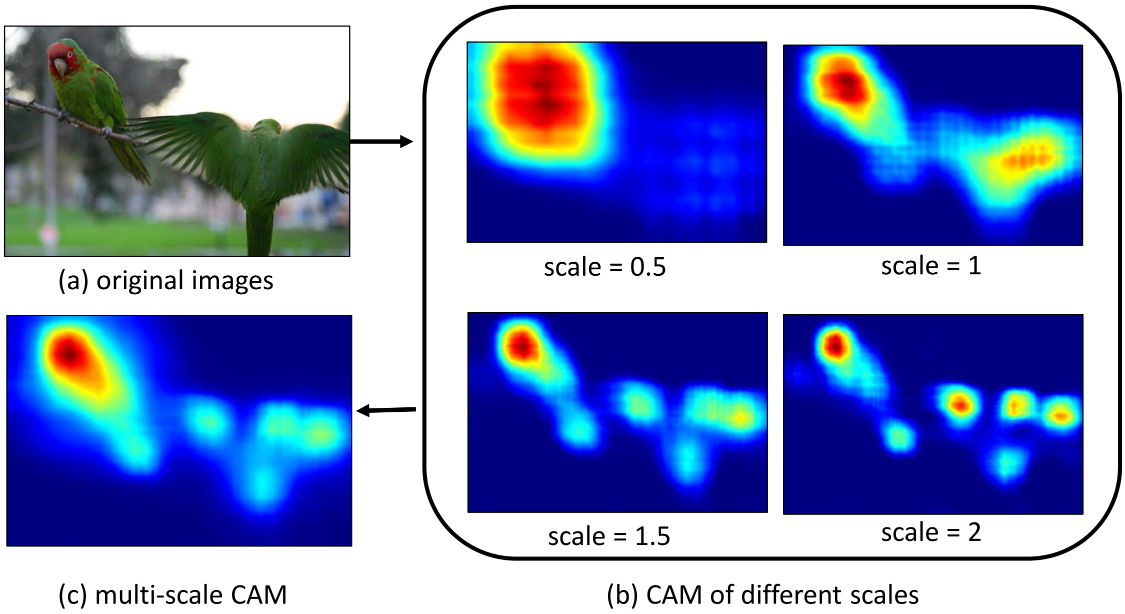

Fig. 2.2: An example of computing multi-scale CAM.

feature maps from the last convolution layer of the backbone. RS$(\cdot)$ is an operation to resize the input to the same width and height as I.

Using multi-scale of original images is beneficial for generating a stable CAM. Given I and it is scaled by a factor s_i, $s_i \in \{s_0, s_1, ..., s_n\}$, the multi-scale CAM for I is detonated as:

$$M^c_{cam} = \sum_{i=0}^{n} (M^c_{ocam}(s_i)/(n+1)), \tag{2.2}$$

where $M^c_{ocam}(s_i)$ is the CAM of class c for the scaled image I with a factor s_i. Fig. 2.2 shows that compared to original CAM (scale=1), the multi-scale CAM provides more accurate object localization.

The CAM scores are normalized, so that we can get the classification probabilities for each pixel in I,

$$P^c_{fg} = M^c_{cam}/max(M^c_{cam}), (c \in C_{fg}), \tag{2.3}$$

where $max(M^c_{cam})$ is the maximum value in the CAM of class c_j.

The background score is calculated using a similar way as in [18]:

$$P_{bg}(i) = (1 - \max_{c \in C_{fg}} (p^c_{fg}(i))^{\gamma}, \gamma > 1. \tag{2.4}$$

where i is the pixel position index, γ is the decay rate which helps to suppress background labels. The overall probability map, namely P_{fg_bg}, is obtained by concatenating foreground and background probabilities P_{fg} and P_{bg}.

After that, we use the dense CRF [60] as post-processing to remove some mislabeled pixels, and the CRF pixel label map is:

$$I_{crf} = \text{CRF}(I, [P_{fg}, P_{bg}]). \tag{2.5}$$

The selected reliable CAM label is:

$$I_{cam}(i) = \begin{cases} \underset{c \in C}{\text{argmax}}(P^c_{fg\_bg}(i)), & \text{if } \max_{c \in C}(P^c_{fg\_bg}(i)) > \alpha \\ 255, & else \end{cases} \tag{2.6}$$

where $C = \{c_0, c_1, ..., c_N\}$ includes all classes of objects and the background (c_0). 255 means the class label is not decided yet.

The final pixel label input to the semantic segmentation branch is:

$$I_{final}(i) = \begin{cases} I_{cam}(i), & \text{if } I_{cam}(i) = I_{crf}(i) \\ 255, & else \end{cases} \tag{2.7}$$

In Eq. (2.6), $\max_{c \in C}(P^c_{fg-bg}(i)) > \alpha$ selects the highly confident regions. In Eq. (2.7), $I_{crf}(i) = I_{cam}(i)$ considers the CRF constraints. Taking this strategy, highly reliable regions as well as their labels can be obtained. The regions which are detonated as 255 in Eq. (2.7) are regarded as unreliable regions.

Fig. 2.3 shows an example of our approach. It is observed that the original CAM labels (as shown in Fig. 2.3 (c)) contain most foreground labels but introduce a number of background pixels as foreground. The CRF labels (Fig. 2.3 (d)) can get accurate boundary of some parts but at the same time, many foreground pixels are regarded as background. In other words, the CAM label can provide reliable background pixels and CRF label can provide reliable foreground pixels. Combining the CAM label and CRF label map using our method, some unreliable pixel-level labels are removed while the reliable regions are still remained, especially obvious at the object boundaries (a clear difference can be seen in the object from Fig. 2.3 (e) and (f)).

2.2.3 Semantic Segmentation Branch: Making Predictions

The reliable pixel-level annotations obtained above are then used as labels for our semantic segmentation branch. Different from the other methods which train their semantic segmentation network with the integral pseudo labels independently, our segmentation branch, which shares the same backbone network with the classification branch, using the provided pixel-level annotation to make prediction. In semantic segmentation branch, there are two parallel sub-branches: one includes two regular convolutional layers, which

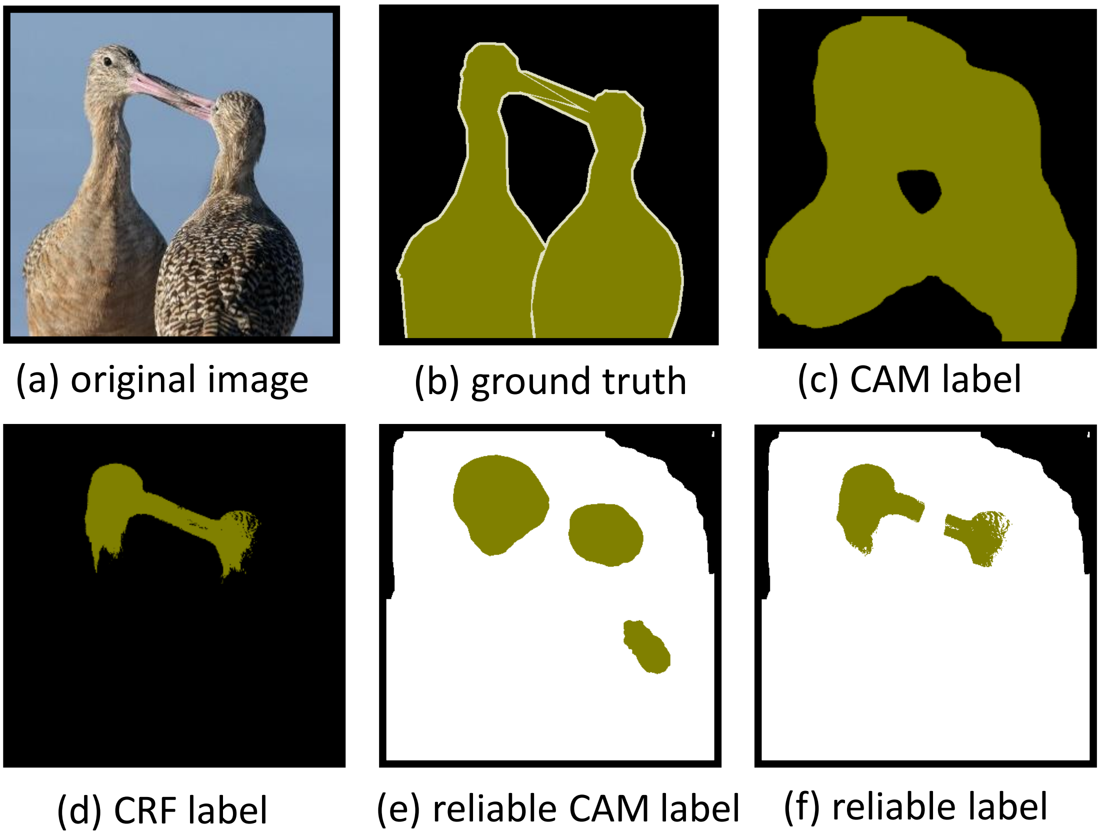

Fig. 2.3: An example of generating reliable pixel labels. (c) only the corresponding class labels of P_{fg_bg} are considered. (d) the CRF pixel label map. *i.e.*, Eq. (2.5), (e) and (f) are generating through Eq. (2.6) and Eq. (2.7), respectively. The white pixels in (e) and (f) are the unreliable regions.

directly aggregate local information to get the mask score map P_l; the other one includes a newly designed R-FAM to aggregate the global information, followed by two convolutional layers to get the mask score map P_g. The final predicted probability map P_{net} is generated after computing the mean value of P_l and P_g and passing it to a softmax layer.

In order to improve the final performance, we designed a new joint loss function for the segmentation branch. In the following part, we will firstly introduce our R-FAM, and then we will give the details of our joint loss function.

Re-weighting Feature Attention Module

Since the classification branch can only provide a limited number of reliable pixel-level pseudo labels, and at the same time the provided pixel-level labels usually focus on the discriminative parts, regular convolution layer cannot make accurate prediction as it can only aggregate local information. For example, in Fig. 2.3, it can be seen that the provided reliable labels only focus on the bird heads, regular convolutional layer cannot

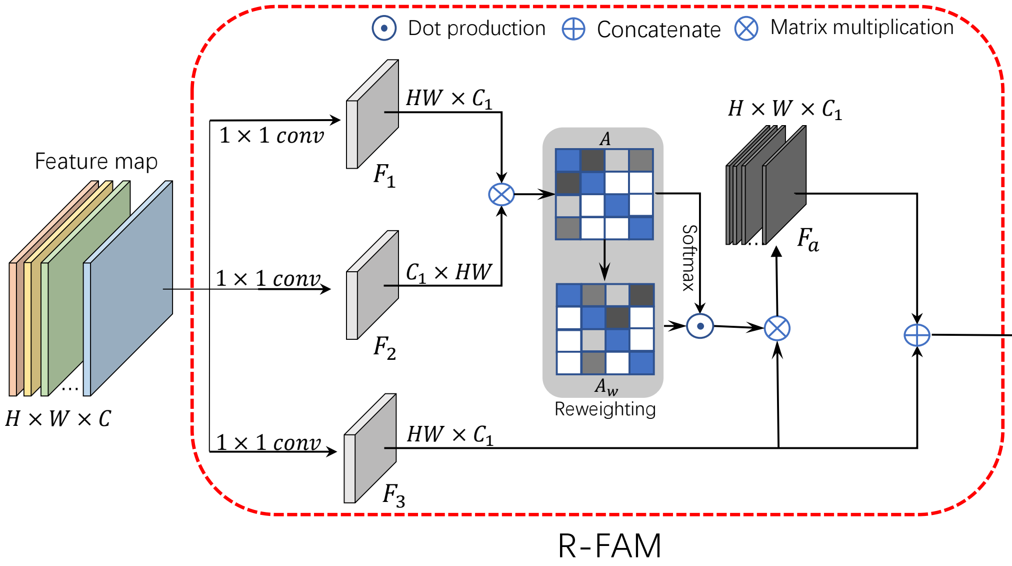

Fig. 2.4: The architecture of our proposed Re-weighting Feature Attention Module (R-FAM). H and W represent height and width of feature map, respectively. C, C_1 and C_g are the channel number of feature maps and $\mathrm{C}_\mathrm{g} = 2 \cdot \mathrm{C}_1$. Note that we set batch size as 1 to simplify description.

extract accurate comprehensive features when the current pixel is far away from those labels. Therefore, it is necessary to introduce global information in order to utilize the limited pixel-level pseudo labels. In this chapter, we use a self-attention mechanism to extract accurate global features. However, the regular attention operation [78] cannot work effectively in this case since it still produces high response to the similar local parts without aggregating sufficient global information. To this end, in order to overcome the drawbacks of the previous attention operations, in this task, we design a Re-weighting Feature Attention Module (R-FAM) to aggregate accurate global information, as shown in Fig. 2.4. Our R-FAM can suppress the influence the original high-level attention response and reduce the influence of the low-level attention response. At the same time, it encourages more middle-level attention to produce higher response.

In this section, we set batch size as 1 to simplify the description. Given an image, after passing through the backbone, suppose the feature map is $F \in \mathbb{R}^{H\times W\times C}$, where H, W and C represent height, width and number of channels of feature maps, respectively. Then three parallel 1×1 convolutional layers are used to reduce the dimension of F, the three generated feature maps are reshaped as $F_1 \in \mathbb{R}^{HW\times C_1}$, $F_2 \in \mathbb{R}^{HW\times C_1}$ and $F_3 \in \mathbb{R}^{HW\times C_1}$, respectively. In order to extract global information, we firstly compute an affinity matrix A:

$$A = F_1 F_2^T, \tag{2.8}$$

where $A \in \mathbb{R}^{HW\times HW}$ and the i-th row ($1 \leqslant i \leqslant HW$) in A indicates the relationship between the ith pixel and all pixels in the feature map.

Then we sort attention values for each pixel in descending order and generate the corresponding index. Mathematically, for pixel k, the index of pixel m in A after sorting is represented as $e_{m|k}$, then the corresponding re-weighting coefficient $A_w(m|k)$ is defined as:

$$A_w(m|k) = \begin{cases} \frac{e(m|k)}{N_{k+}}, & \text{if } A(m|k) > 0 \\ \sqrt{\frac{|N_k+1-e(m|k)|}{N_{k+}}}, & \text{else} \end{cases}, \tag{2.9}$$

where $A(m|k)$ is the attention value of pixel m in A for pixel k (the mth value in the kth row). N_k is the number of all attention in A for pixel k, *i.e.*, $N_k = HW$. N_{k^+} is the number of all positive attention in A for pixel k:

$$N_{k^+} = \sum_{m=1}^{HW} \left[A(m|k) > 0\right], \tag{2.10}$$

where $[\cdot]$ is Iverson bracket, which equals to 1 if the inside condition is true, otherwise equals to 0.

The re-weighting coefficient matrix A_w will produce small coefficients for both high and low-level attention values, which suppress the influence of both cases. At the same time, it will produce large coefficients for those middle-level attention values, making the whole attention be more democratic.

We then can generate the affinity feature map F_a.

$$F_a = \text{RS}((\text{softmax}(A) \odot A_w)F_3), \tag{2.11}$$

where $F_a \in \mathbb{R}^{H \times W \times C_1}$. $\text{softmax}(\cdot)$ denotes the softmax layer. $\odot$ means element-wise multiplication. Each pixel aggregates information from the whole feature map, and the final modified global feature map is defined as:

$$F_g = \text{concat}([F_a, \text{RS}(F_3)]), \tag{2.12}$$

where $F_g \in \mathbb{R}^{H \times W \times 2C_1}$. Then we can use regular convolutional layer to generate the predicted score map P_g. The final probability map is the output of a softmax layer, which takes the average of two score maps from two sub-branches as input:

$$P_{net} = softmax(\frac{P_l + P_g}{2}), \tag{2.13}$$

where P_l is the predicted score from the local information sub-branch.

```
Algorithm 1: Algorithm flow of our proposed RRM.
Input:
Images $I$
image-level class labels $C_{fg}$
Output:
The trained end-to-end network, $Net$

while iteration is true do
    Use the classification Network branch to get the original CAMs;
    Get the multi-scale CAMs with Eq. (2.2) for each class;
    Use Eq. (2.3) and Eq. (2.4) to get foreground probability $P_{fg}$ and background
      probability $P_{bg}$;
    Get the overall CAM probability map $P_{fg\_bg}$ by combining $P_{bg}$ and $P_{fg}$;
    Calculate reliable CAM label $I_{cam}$ and CRF label $I_{crf}$;
    Get the reliable regions and label $I_{final}$ from $I_{cam}$ and $I_{crf}$ using Eq. (2.6) and
      Eq. (2.7);
    Produce predictions using segmentation branch and update the whole
      network using loss function $\mathcal{L} = \mathcal{L}_{class} + \mathcal{L}_{ce} + \mathcal{L}_{energy} + \mathcal{L}_{BCD}$;
end
```

2.2.4 Loss Functions

Loss Function of the Classification Branch

In the classification branch, we adopt the same loss function used in [18, 57], which is a multi-label soft margin loss:

$$\begin{aligned}\mathcal{L}_{class}(\hat{y}, y) = -\frac{1}{C_{\text{fg}}}\sum_{c=1}^{C_{\text{fg}}}(y_c\text{log}(\frac{1}{1+e^{-\hat{y}_c}}) \\ + (1-y_c)\text{log}(1-\frac{1}{1+e^{-\hat{y}_c}}))\end{aligned}, \tag{2.14}$$

where y is the image-level annotation, and $\hat{y}$ is the output of GAP layer.

Loss Functions of the Segmentation Branch

In order to adapt to the tiny reliable labels, we design a new joint loss function $\mathcal{L}_{joint-seg}$, including three parts: the first one is a cross-entropy loss $\mathcal{L}_{ce}$, focusing on utilizing the labeled data; the second one is the dense energy loss $\mathcal{L}_{energy}$, utilizing the shallow features such as RGB and spatial information; the third one is a BCD loss $\mathcal{L}_{BCD}$ for considering the high-level semantic features. As a result, the joint loss is:

$$\mathcal{L}_{joint-seg} = \mathcal{L}_{ce} + \lambda_1\mathcal{L}_{energy} + \mathcal{L}_{BCD}. \tag{2.15}$$

Cross-Entropy Loss In our approach, cross-entropy loss is based on the provided pseudo labels from our classification branch,

$$\mathcal{L}_{ce} = -\sum_{c \in C, i \in \Phi} B_c(i) log(P^c_{net}(i)), \tag{2.16}$$

where $B_c(i)$ is a binary indicator, which equals to 1 if the label of pixel i is c and otherwise 0; Φ denotes the labeled regions, $\Phi = \{i | I_{final}(i) \neq 255\}$; $P^c_{net}(i)$ is the output probability of being class c from the trained network.

Dense Energy Loss So far, all labeled pixels have been used for training with cross-entropy loss, but there are a large number of unlabeled pixels. In order to make predictions for those unlabeled regions, we design a new shallow loss named dense energy loss considering both RGB colors and spatial positions.

Given an image I, we firstly define the energy formulation between pixel i and j based on [79]:

$$E(i,j) = \sum_{\substack{c_a, c_b \in C \\ c_a \neq c_b}} G(i,j) P^{c_a}_{net}(i) P^{c_b}_{net}(j). \tag{2.17}$$

In Eq. (2.17), both c_a and c_b are the class labels, $P^{c_a}_{net}(i)$ and $P^{c_b}_{net}(j)$ are the softmax output of our segmentation branch at pixel i and j, respectively. $G(i,j)$ is a Gaussian kernel bandwidth filter:

$$G(i,j) = \frac{1}{W} exp(-\frac{\|I_s(i) - I_s(j)\|^2}{2\sigma_d^2} - \frac{\|I_{rgb}(i) - I_{rgb}(j)\|^2}{2\sigma_r^2}), \tag{2.18}$$

where $\frac{1}{W}$ is the normalized constant, $I_s(\cdot)$ is the pixel spatial position while $I_{rgb}(\cdot)$ is the RGB color of the corresponding pixel. σ_d and σ_r are hyper parameters which control the scale of Gaussian kernels. Eq. (2.17) can be simplified using Potts model [12]:

$$\begin{aligned} E(i,j) &= \sum_{\substack{c_a, c_b \in C \\ c_a \neq c_b}} G(i,j) P^{c_a}_{net}(i) P^{c_b}_{net}(j) \\ &= G(i,j) \sum_{c \in C} P^c_{net}(i)(1 - P^c_{net}(j)). \end{aligned} \tag{2.19}$$

Finally, our dense energy loss $\mathcal{L}_{energy}$ can be written as:

$$\mathcal{L}_{energy} = \sum_{i=0}^{N} \sum_{\substack{j=0 \\ j \neq i}}^{N} S(i) E(i,j). \tag{2.20}$$

In Eq. (2.20), N is the pixel number of the given image I. Considering the fact that cross-entropy loss is designed for supervised learning with label information 100%

accurate, but in this task, all pixel labels are not 100% reliable, which means that using cross-entropy loss might introduce some errors. Thus, our dense energy loss is applied to mitigate this problem. Based on this idea, we design a soft filter $S(i)$ for pixel i:

$$S(i) = \begin{cases} 1 - \max_{c \in C}(P^c_{net}(i)), & i \in \Phi \\ 1, & else \end{cases} \tag{2.21}$$

Batch-based Class Distance Loss Dense energy loss attempts to utilize shallow features. In fact, the network also provides high-level features. Thus, we designed a new BCD loss to use the high-level features. The BCD loss is based on the following three motivations. Firstly, the feature embedding for different classes should be distinct, if the feature distance between different classes can be increased in the embedding space, they are more distinguishable. Secondly, as observed in [80, 81], features extracted in lower CNN layer are more related to the low-level cues such as edge and color, while in higher layer features are more related with semantic meaning. Thus, the high-level semantic features are considered for our BCD feature. Thirdly, directly computing feature distance for all pixels requires high computational cost, we only consider all images in a batch for efficiency.

Mathematically, before passing to the final convolution layer, the feature map in the local information sub-branch is $F_l \in \mathbb{R}^{B\times H\times W\times C_l}$ and the one in the global information sub-branch (after R-FAM) is $F_g \in \mathbb{R}^{B\times H\times W\times C_g}$, where B is the batch size. The final high-level feature F_m is the concatenation of F_l and F_g,

$$F_m = \text{concat}([F_l, F_g]). \tag{2.22}$$

As previously defined in Eq. (2.13), the final output of the segmentation branch is P_{net}, then the predicted mask is:

$$M_p = \underset{c \in C}{\text{argmax}}(P^c_{net}), \tag{2.23}$$

where $M_p \in \mathbb{R}^{B\times H\times W}$, the predicted foreground class set in M_p is defined as C_p, which is a subset of C_{fg} and $C_p = \left\{c^1_p, c^2_p, ..., c^K_p\right\}$. For an arbitrarily predicted class c^i_p from C_p, we can always get that $c^i_p \in C_{\text{fg}}$.

Then we can use the feature map F_m and the predicted mask M_p to compute a loss function, so that features for different classes can be pulled apart in the embedding space. Directly calculating distance among all different pixels is the most intuitive solution, but it needs high computing cost. Therefore, we rely on class centers to improve the computing efficiency:

$$F_{ctr}^{c_p^i} = \frac{1}{N_{c_p^i}} \sum_{j=1}^{\text{BHW}} [M_p(j) = c_p^i] F_m(j), c_p^i \in C_p. \tag{2.24}$$

$$F_{ctr}^{c_0}(b) = \frac{1}{N_{c_0}^b} \sum_{j=1}^{\text{HW}} [M_p^b(j) = c_0] F_m^b(j), b \in \{1, ..., B\}. \tag{2.25}$$

In Eq. (2.24) and Eq. (2.25), $F_{ctr}^{c_p^i}$ is the feature center for foreground class c_p^i in the given batch while $F_{ctr}^{c_0}(b)$ is the feature center for background of the b-th image in the batch, $[\cdot]$ is z Iverson bracket. $N_{c_p^i}$ is the pixel number for class c_p^i in the batch and $N_{c_0}^b$ is the pixel number for background in the b-th image of the batch. In Eq. (2.25), M_p^b and F_m^b denote the b-th predicted mask and its corresponding feature map in the batch, respectively. Note that for foreground class center, all pixels belonging to the same class in the batch are used, while for the background center, for each image we calculate one center. This is because the semantic feature should be similar for the same foreground class, but it could be different for the background in different images. Then the BCD loss is defined as follows:

$$\begin{aligned}
\mathcal{L}_{BCD} = & \underbrace{\frac{1}{N_{\text{ff}}} \sum_{i=1}^{K} \sum_{j=1}^{K} [c_p^i \neq c_p^j](1 + cos(F_{ctr}^{c_p^i}, F_{ctr}^{c_p^j}))}_{\mathcal{L}_{BCD}(F_{ctr}^{fg}, F_{ctr}^{fg})} \\
& + \underbrace{\frac{1}{N_{\text{fb}}} \sum_{i=1}^{K} \sum_{b=1}^{B} (1 + cos(F_{ctr}^{c_p^i}, F_{ctr}^{c_0}(b)))}_{\mathcal{L}_{BCD}(F_{ctr}^{fg}, F_{ctr}^{bg})} \\
& + \underbrace{\frac{1}{N_{\text{fg}}} \sum_{i=1}^{K} \sum_{j=1}^{\text{BHW}} [M_p(j) = c_p^i](1 - cos(F_m(j), F_{ctr}^{c_p^i}))}_{\mathcal{L}_{BCD}(F_m, F_{ctr}^{fg})} \\
& + \underbrace{\frac{1}{N_{\text{bg}}} \sum_{b=1}^{B} \sum_{j=1}^{\text{HW}} [M_p^b(j) = c_0](1 - cos(F_m^b(j), F_{ctr}^{c_0}(b))),}_{\mathcal{L}_{BCD}(F_m, F_{ctr}^{bg})}
\end{aligned} \tag{2.26}$$

where $cos(\cdot)$ means the cosine similarity operator. $\mathcal{L}_{BCD}(F_{ctr}^{fg}, F_{ctr}^{fg})$ aims to pull apart the feature centers of all foreground classes, while $\mathcal{L}_{BCD}(F_{ctr}^{fg}, F_{ctr}^{bg})$ try to pull apart the foreground class centers and the background class centers. $\mathcal{L}_{BCD}(F_m, F_{ctr}^{fg})$ and $\mathcal{L}_{BCD}(F_m, F_{ctr}^{bg})$ pull all features close to theirs corresponding centers. N_{ff}, N_{fb}, N_{fg} and N_{bg} are the number of pairs involved in distance computing for the four scenarios.

Through closing up the distance between features and their corresponding center and pulling apart the distance between different feature centers, the high-level features are more discriminative for different classes. More importantly, compared to calculating distance for all pixels directly, $\mathcal{L}_{BCD}$ is more efficient.

2.3 Experiments

2.3.1 Dataset and Implementation Details

Dataset. Our RRM mdoel is trained and validated on PASCAL VOC 2012 [82] as well as its augmented data, including $10,582$ images for training, $1,449$ images for validating and $1,456$ images for testing. The Mean Intersection over Union (mIoU) is considered as the evaluation criterion.

Implementation Details. The backbone network is a ResNet model with 38 convolution layers [83]. We remove all the fully connected layers of the original network and engage dilated convolution for the last three ResNet blocks (a ResNet block is a set of residual units with the same output size), the dilated rate is 2 for the last third layer, and 4 for the last 2 layers. For the semantic segmentation branch, we add two dilation convolution layers of the same configuration for the local information in the segmentation branch after the backbone [83], with kernel size 3, dilated rate 12, and padding size 12. The channel size is set to 512 and 21, respectively. For R-FAM, the channel size of three 1×1 convolution layers is 256, then we added two convolution layers, with kernel size 3, dilated rate 1, and padding size 1. The number of channels are set as 256 and 21, respectively. The cross-entropy loss is computed for background and foreground individually. σ_d and σ_r in our dense energy loss are set as 100 and 15, respectively.

The training learning rate is $1e$-3 with weight decay being $5e$-4. The training images are resized with a ratio randomly sampled from $(0.7, 1.3)$, and they are randomly flipped. Finally, they are normalized and randomly cropped to size 321×321. Batch size is set to 8, and the maximum iteration is 40K.

To generate reliable regions, the scale ratio in Eq. (2.2) is set to $\{0.5, 1, 1.5, 2\}$, γ in Eq. (2.5) is set to 4 for P_{fg_bg}. The CRF parameters in Eq. (2.5) follow the setting in [18]. In Eq. (2.6), α is chosen with 40% pixels selected as labeled pixels for each class. During validating and testing, dense CRF is applied as a post-processing method, and the parameters are set as the default values given in [34]. The weighting parameter λ_1 in Eq. (2.15) is set as 1. During training, both two branches update the backbone network. During testing, only the segmentation branch is used to produce the predictions.

In order to show the effectiveness and scalability of our idea, we also extend our method to a two-step framework. We firstly used our network to produce the pseudo masks for the training dataset. After that, we train and evaluate based on the fully-supervised segmentation Deeplab-v2 (ResNet-101 is used as backbone) [47] with those generated pixel labels. All parameters follows the default setting in [47].

2.3.2 Analysis of Our Approach

Table 2.1: Performance on PASCAL VOC 2012 *val* set based on different mined region. Ratio means the proportion of reliable regions mined by our method to the whole pixels. "CE loss" means only cross-entropy loss was used for our segmentation branch. "Joint loss" means that our joint loss function is used for the segmentation branch.

Ratio	mIoU (%)						
	0.1	0.2	0.3	0.4	0.6	0.8	1.0
CE loss	52.2	52.3	52.7	51.9	52.9	54.2	55.4
Joint loss	53.0	63.7	64.7	**65.4**	63.4	62.8	62.9

Our RRM has several important aspects: using the tiny reliable pseudo masks for supervision, a new segmentation branch and a new joint loss function for end-to-end training. Ablation studies are conducted to illustrate their individual and joint effectiveness, with results reported in Table 2.1, Table 2.2 and Table 2.3.

Table 2.2: Analysis of the provided pseudo label from classification branch on performance using PASCAL VOC 2012 *val* dataset. "CAM" refers to the case that class activate maps are directly used as pseudo masks, and Ours-RRM refer to the case that mined reliable regions are used as pseudo masks. "CE loss" means only cross-entropy loss was used for our segmentation branch and "Joint loss" means that cross-entropy loss, dense energy loss and BCD loss are applied. Both CAM and ours-RRM use top 40% pixels according to Table 2.1.

	CE loss (mIoU) (%)	Joint loss (mIoU) (%)
CAM	48.3	58.5
Ours-RRM	52.0	**65.4**

We firstly validate the influence of different pseudo mask size by changing α. Table 2.1 reports the results. A smaller pseudo mask size means that less regions are selected but all selected regions are more reliable, while a larger size means that more pixels are labeled with less reliability. Table 2.1 demonstrates that 20%-60% labeled pixels lead to the best performance. If there are too few labeled pixels, satisfactory performance cannot be obtained since the segmentation network cannot get enough labels for learning. On the other hand, too many labeled pixels means more incorrect labels are used, which are noise for the training processing.

Table 2.3: Analysis of different loss functions and components in our segmentation branch with performance on PASCAL VOC 2012 *val* dataset. Ours-RRM (local) means that only local information sub-branch is used to produce pseudo masks. Ours-RRM (global) means that only global information sub-branch is used to produce pseudo masks. Ours-RRM (full) is that we used the whole segmentation branch. “CE loss” means only cross-entropy loss was used for our segmentation branch and “CE+DE loss” means our dense energy loss was combined with cross-entropy loss was used. “CE+BCD loss” means that cross-entropy loss and BCD loss are used. “Joint loss” means that besides cross-entropy loss, dense energy loss and BCD loss are used.

	mIoU (%)			
	CE loss	CE+DE loss	CE+BCD loss	Joint loss
Ours-RRM (local)	48.5	62.6	49.9	63.6
Ours-RRM (global)	51.3	61.6	51.9	62.2
Ours-RRM (full)	52.0	64.8	52.9	**65.4**

Table 2.2 shows the effectiveness of provided pseudo label. First of all, the results obtained using original CAM regions and the mined reliable regions with RRM are compared. It is observed that the pseudo label generated by RRM outperforms original CAM labels. If we remove the joint loss from our segmentation branch (see the performance of “CE loss”), it also shows that the reliable pseudo labels generated by RRM improves the segmentation performance.

In Table 2.3, we firstly make an ablation study for analyzing the loss function of our segmentation branch. The comparison between *Ours-RRM (full)* with CE loss and *Ours-RRM (full)* with the joint loss illustrates the effectiveness of the introduced joint loss. Without the joint loss, the mIoU obtained with RRM with CE loss gets lower. This is because the mined reliable regions with RRM cannot provide sufficient labels for segmentation model training when only considering cross-entropy loss. Compared to the performance of cross-entropy loss, both dense energy loss and BCD loss improve the performance, with 12.8% and 0.9% improvement, respectively. After adopting the joint loss, segmentation performance improves with a big margin from 52.0% to 65.4%, which is a 13.4% increase. Besides, from *Ours-RRM (local)* and *Ours-RRM (global)*, and it can also be found that no matter what the structure of our segmentation branch is, the introduced joint loss can improve the final performance .

In addition, Table 2.3 shows the influence of different components in our network.

Table 2.4: Analysis of different components in our approach for the training seed and performance. Conf.: select confident seeds as pseudo labels. Loc.: local information sub-branch. Glo.: global information sub-branch. Speed: training speed (time costing per iteration)

CAM	CRF	Conf.	Loc.	Glo.	DE	BCD	Speed	mIoU
✓			✓				1.26s	46.1
✓		✓	✓				1.40s	47.3
✓	✓	✓	✓				4.59s	48.5
✓	✓	✓	✓	✓			4.68s	52.0
✓	✓	✓	✓		✓	✓	4.77s	63.6
✓		✓	✓	✓	✓	✓	1.53s	58.3
✓	✓	✓	✓	✓	✓		4.82s	64.8
✓	✓	✓	✓	✓	✓	✓	4.86s	**65.4**

It can be seen that when the local information sub-branch or global information sub-branch are used separately to predict the mask, their performance are 63.6% and 62.2% (with joint loss), respectively. Using both of them for predicting, the performance of "*Ours-RRM (full)*" is improved to 65.4%, with 1.8% and 3.2% improvement, respectively. From the all results of "CE loss", "CE+DE loss" and "CE+BCD loss", it is observed that combining the local information sub-branch or global information sub-branch leads to better performance than using them individually.

In Table. 2.4, we show the influence of different components in our approach for the training speed and the performance. It can be seen that with all our proposed components, it has the highest performance but with the low training speed. Using CRF [60] to refine the CAM label requires high computing cost as it is a dense and global pixel-level operation. Note that only global information sub-branch is used during inference (the classification branch and CRF is not used during inference), so the proposed components only has no influence on the inference speed (about 0.15s per image on Nvidia 2080Ti).

In Table. 2.5, we compare our R-FAM with the regular attention module [78], "FAM" means that we do not generate the re-weighting coefficient matrix A_w and we directly use the original attention matrix A in Eq. (2.11). It can be seen that using our re-weighting module generates a higher performance than the regular attention module, obtaining mIoU increases of 0.9% and 0.7% for cross-entropy loss and our joint loss, respectively. Be-

Table 2.5: Analysis of our proposed R-FAM. FAM means that there is no A_w in Eq. (2.11). FAM is also can be seen as the non-local module in [78].

FAM	R-FAM	CE loss	DE+BCD loss	mIoU (%)
✓		✓		51.1
	✓	✓		52.0
✓		✓	✓	64.7
	✓	✓	✓	**65.4**

sides, we also conduct ablation study on the influence of the shared backbone. Specifically, we divide our end-to-end framework into two individual networks: the classification network and the segmentation network. Then we train them separately. Finally, we find that sharing a backbone significantly improves the performance. Without a shared backbone, mIoU on PASCAL VOC val set is 61.7%, while with a shared backbone, the performance is 65.4%, which is a 3.7% mIoU increase. We think the classification branch provides extra pre-train to the segmentation branch. The feature map from the shared backbone has to provide more discriminative information to distinguish each class in the classification branch, which also benefits the segmentation branch to conduct pixel-level classification.

2.3.3 Comparisons with Previous Approaches

In order to show the effectiveness and scalability of our idea, we also extend our method to a two-step framework. The difference is that for our one-step method, *i.e.*, *Ours-RRM (one-step)*, we produce the predictions through our segmentation branch directly. Whereas for our two-step method, we firstly used our *Ours-RRM (one-step)* network to produce the pseudo masks for the training dataset. Following that, we train and evaluate Deeplab-v2 (ResNet-101 is used as the backbone) [47] with those generated pixel labels, called *Ours-RRM-ResNet (two-step)*. The final results can be found in Table 2.6. It is observed that among existing methods solely using image-level label without extra data, most approaches apply multiple different DNNs with many bells and whistles, while we get equivalent results with only one end-to-end network (*Ours-RRM (one-step)*) and our two-step approach (*ours-RRM-ResNet (two-step)*) outperforms them significantly. More importantly, it should be noticed that AffinityNet [18], SSDD [56] and SEAM [57] all used ResNet-38 [83] as baseline, which is more powerful than ResNet-101 [55], and even in this case *ours-RRM-ResNet (two-step)* still outperforms them with a big margin.

To the best of our knowledge, the previous state-of-the-art, RIB [54], achieves the mIoU score of 68.3% and 68.6% on PASCAL VOC *val* and *test* set, but it uses at least three individual networks separately during both training and testing, making both of them being complicated and time-costing. *Ours-RRM-ResNet (two-step)* gives a better performance with mIoU scores of 69.3% and 69.2% on PASCAL VOC *val* and *test* set, which represents 1.0% and 0.6% improvement. Note that we do not use extra data or information.

Furthermore, compared to all our baselines, the performance of our one-step and two-step solution are boosted to 65.4% and 69.3% on *val* set, achieving new state-of-the-art performances for one-step and two-step image-level label weakly supervised semantic segmentation, respectively. It is also interesting to find that our two-step solution even performs better than fully supervised semantic segmentation model DeepLab [5] on PASCAL VOC 2012 *val* set.

Table 2.6: Comparison with the state-of-the-art approaches on PASCAL VOC 2012 *val* and *test* dataset. Sup.-supervision information, GT-ground truth, F-full supervision, L-image-level class label.

Method	Baseline	Sup.	Extra Data	End-to-end	val (mIoU) (%)	test (mIoU) (%)
Deeplab-v2 [47]	ResNet-101	F	-	-	76.8	79.7
Model A1 [83]	ResNet-38	F	-	-	80.8	82.5
DSRG (CVPR'18) [34]	ResNet-101	L	MSRA-B [84]	×	61.4	63.2
FickleNet (CVPR'19) [55]	ResNet-101	L	MSRA-B [84]	×	64.9	65.3
zhang *et.al* (ECCV'20) [85]	ResNet-101	L	MSRA-B [84]	×	66.6	66.7
EME (ECCV'20) [86]	ResNet-101	L	MSRA-B [84]	×	67.2	66.1
MCIS (ECCV'20) [87]	ResNet-101	L	MSRA-B [84]	×	66.2	66.9
ICD (CVPR'20) [50]	ResNet-101	L	MSRA-B [84]	×	67.8	68.0
ILLD (TPAMI'20) [51]	ResNet-101	L	24K ImageNet [51]	×	67.8	68.3
ISISU (PR'21) [88]	ResNet-101	L	MSRA-B [84]	×	62.5	62.7
OAA (TPAMI'21) [89]	ResNet-101	L	MSRA-B [84]	×	66.1	67.2
EPS (CVPR'21) [52]	ResNet-101	L	MSRA-B [84]	×	67.0	67.3
Group-wise (AAAI'21) [90]	ResNet-101	L	MSRA-B [84]	×	**68.2**	**68.5**
EM-Adapt (ICCV'15) [44]	VGG-16	L	-	✓	38.2	39.6
AffinityNet (CVPR'18) [18]	ResNet-38	L	-	×	61.7	63.7
IRN (CVPR'19) [91]	ResNet-50	L	-	×	63.5	64.8
SSDD (ICCV'19) [56]	ResNet-38	L	-	×	64.9	65.5
SEAM (CVPR'20) [57]	ResNet-38	L	-	×	64.5	65.7
ICD (CVPR'20) [50]	ResNet-101	L	-	×	64.1	64.3
SubCat (CVPR'20) [92]	ResNet-101	L	-	×	66.1	65.9
BES (ECCV'20) [93]	ResNet-101	L	-	×	65.7	66.6
CONTA (NeurIPS'20) [94]	ResNet-101	L	-	×	66.1	66.7
ECS-Net (ICCV'21) [95]	ResNet-38	L	-	×	66.6	67.6
A^2GNN (TPAMI'21) [62]	ResNet-101	L	-	×	66.8	67.4
AdvCAM (CVPR'21) [53]	ResNet-101	L	-	×	68.1	68.0
CGNet (ICCV'21) [96]	ResNet-101	L	-	×	**68.4**	68.2
RIB (NeurIPS'21) [54]	ResNet-101	L	-	×	68.3	**68.6**
Ours-RRM (one-step)	ResNet-38	L	-	✓	**65.4**	**65.3**
Ours-RRM-ResNet (two-step)	ResNet-101	L	-	×	**69.3**	**69.2**

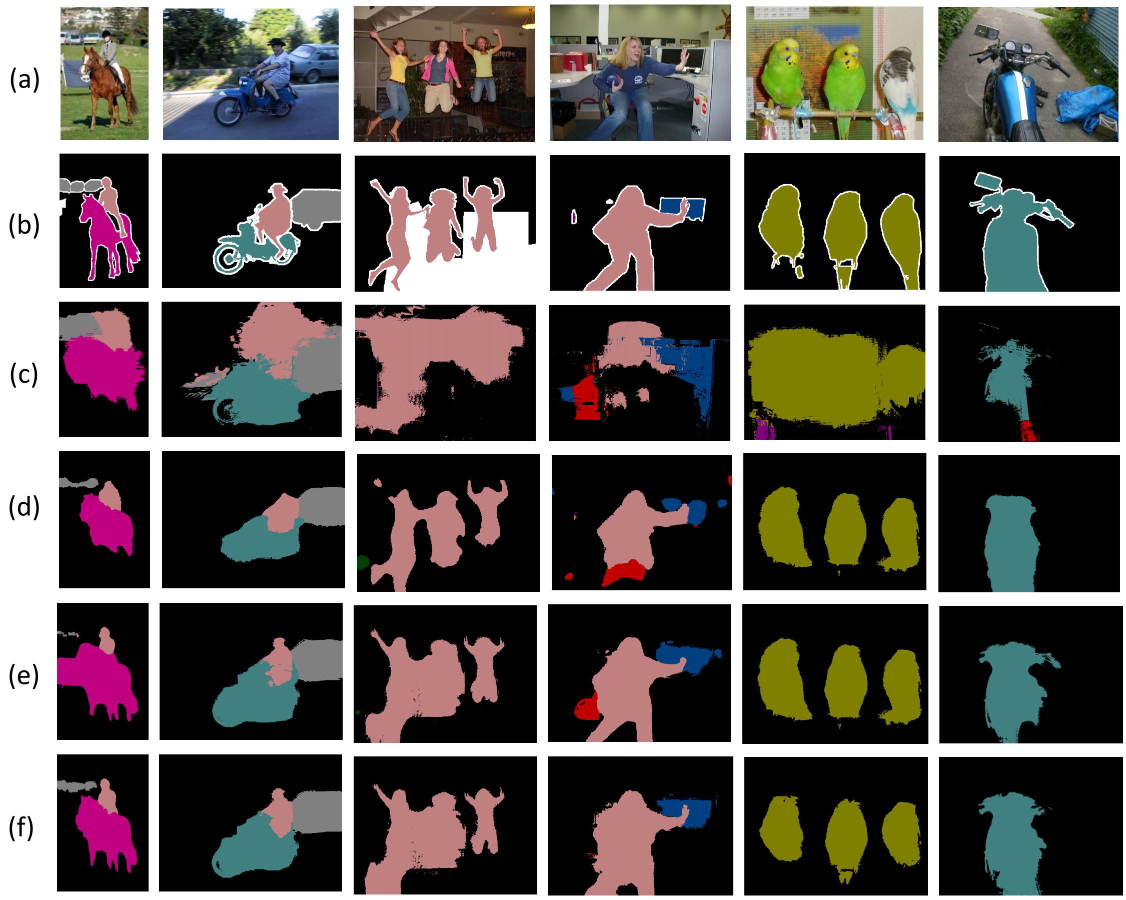

Fig. 2.5: Qualitative segmentation results on PASCAL VOC 2012 *val* set. (a) Original images. (b) Ground-truth. (c) EM-Adapt results [44]. (d) *The baseline* results [97]. (e) *Ours-RRM (one-step)* results. (f) *Ours-RRM-ResNet (two-step)* results.

In Figure 2.5, we report some subjective semantic segmentation results of ours methods, which are compared with EM-Adapt [44], the state-of-the-art end-to-end network. *Ours-RRM (one-step)* obtains much better segmentation results on both large and small objects, with much accurate boundaries. We also show some results of our two-step approaches, and it can be seen that among our three methods, *ours-RRM-ResNet (two-step)* obtains the best performance duo to the powerful network architecture.

2.4 Discussion

Both one-step and two-step solutions have their own strengths and weaknesses. For the one-step solution, the most advantage is the simpler architecture with less training procedure compared to the two-step solution. But at the same time, its performance is still limited. For the two-step solution, the main advantage is that it can divide the task into several sub-tasks, which means some advanced techniques can be applied flexibly and rapidly. Besides, it can achieve better performance than the one-step solution. However, the longer training processes with more manual parameters make it harder to be used and implemented.

2.5 Conclusion

We proposed the Reliable Region Mining model, an end-to-end network for image-level weakly supervised semantic segmentation. We revisited drawbacks of the state-of-the-art methods, which adopt the two-step approach. We proposed a one-step approach through mining tiny reliable regions and used them as ground-truth labels directly for our segmentation branch training. With limited pixels as supervision, we designed a dense energy loss and a batch-based class distance loss, which consider shallow features (RGB colors and spatial information) and high-level feature, respectively. The two new losses cooperate with the pixel-wise cross-entropy loss to optimize the training process. Furthermore, we design a new feature attention module to extract global information, which also proves to be effective for the final prediction. Based on our one-step RRM, we extended a two-step method. Both our one-step and two-step approaches achieve the state-of-the-art performance. More importantly, our RRM offers a different perspective from the traditional two-step solutions. We believe that the proposed one-step approach could further boost research in this direction.

Chapter 3

Graph based Framework for Bounding-Box Supervised Semantic Segmentation

3.1 Motivation

As mentioned in Sect. 1.2.3, for utilizing bounding-box as supervision, there are two kinds main solutions: CNN-based approaches and GNN-based approaches. CNN-based approaches [13, 14, 15, 44] mainly use object proposals [58, 59] 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. Graph-based approaches are proposed to use the confident but a limited number of pixels to generate the pseudo labels. Compared to previous approaches, graph-based learning can directly build the relationship among different pixels, enabling to suppress the negative impact of the label noise. However, current graph-based approaches such as GraphNet [1] mainly has two drawbacks: (1) they built an unweighted graph as input which cannot accurately provide edge information since it treats all edges equally. (2) They used standard GCN [61], which will lead to incorrect feature aggregation as input nodes and edges are not 100% accurate. Thus, if the strong correlations among pixels from different semantics can be effectively alleviated, a better propagation model can be acquired to generate more accurate pseudo object masks.

We design an Affinity Attention Graph Neural Network (A^{2}GNN) to address the above mentioned issues. Firstly, in order to produce accurate 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. Secondly, we design a new GNN layer to produce accurate pseudo labels. Our GNN layer considers both the attention mechanism and the edge weights to make 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. Finally, considering that we only use a limited number of confident seed labels as supervision, which is insufficient for the network optimization, we introduce a multi-point (MP) loss and a consistency-checking mechanism to augment the training of A^{2}GNN. Our MP loss adopts an online update mechanism to provide extra supervision from bounding box information. Meanwhile, it also attempts to close up the feature distance of the same semantic objects, making the pixels of the same object distinguishable from others. The proposed consistency-checking mechanism attempt to remove the noisy labels from the selected seed labels, by comparing them with the labels used in the MP loss. In the following parts, we will introduce the details of our A^{2}GNN.

This chapter mainly contains our work "Affinity Attention Graph Neural Network for Weakly Supervised Semantic Segmentation", which is published in TPAMI [62].

3.2 Generate Pixel-level Seed Label

The common practice to initialize weakly supervised task is to generate pixel-level seed labels from weak supervision [1, 57, 50]. For the bounding-box supervised semantic segmentation 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 [27, 57, 18, 21]. To generate pixel-level labels from bounding box supervision, Grab-cut [36] 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 [57], 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:

$$M_I = \text{Net}_{\text{SEAM}}(I), \tag{3.1}$$

where M_I is the generated seed labels. $\text{Net}_{\text{SEAM}}(\cdot)$ is the classification CNN used in SEAM [57].

For the Bound-box Supervised Semantic Segmentation (bounding-box supervised semantic segmentation) 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

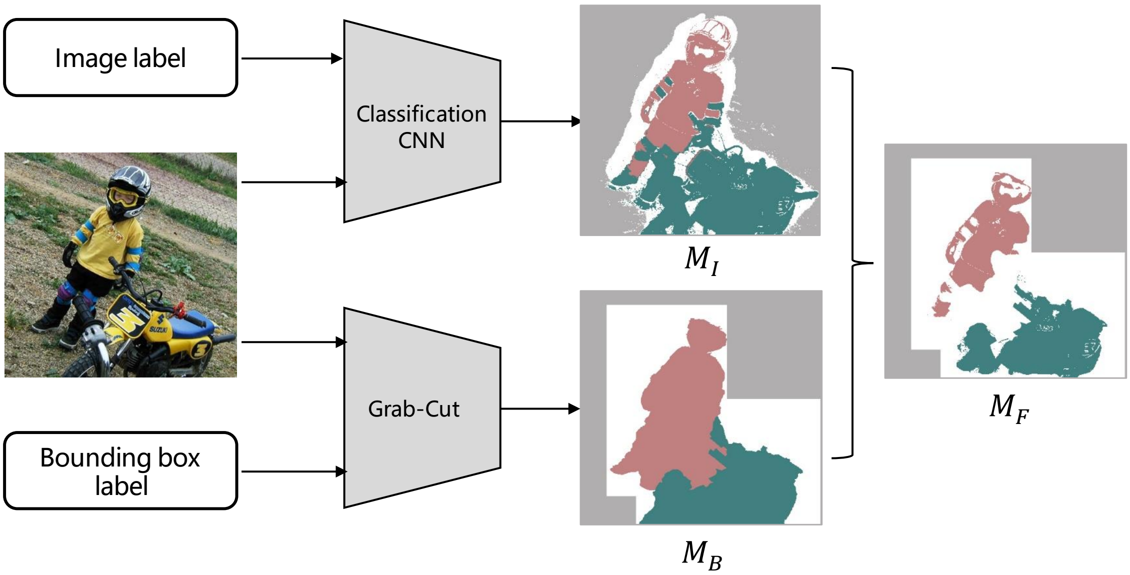

Fig. 3.1: An example of generating pixel-level seed labels. Given an image with its label, we firstly generate M_I from image-level label using a classification CNN and SEAM [57] method. Meanwhile, bounding box label is transferred to pixel-level label M_B using Grab-cut. Finally, M_I and M_B are integrated together to get the pixel-level seed label M_F. Each color represents one class and "white" means the pixel label is unknown.

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 [36] to generate the seed labels from bounding box supervision, the seed labels for each bounding box are defined as:

$$M_{B_k}(i) = \begin{cases} \text{Grab}(i), & \text{if } i \in B_k \text{ and Grab}(i) \neq c_0 \\ 255, & \text{else} \end{cases}, \tag{3.2}$$

where Grab$(\cdot)$ is the Grab-cut operator and 255 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:

$$M_B(i) = \begin{cases} c_0, & \text{if } i \notin B \\ M_{B_k}(i), & \text{if } i \in B \end{cases}. \tag{3.3}$$

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

$$M_F(i) = \begin{cases} M_B(i), & \text{if } i \notin B \\ M_I(i), & \text{if } i \in B \text{ and } M_B(i) = M_I(i) \\ M_{B_k}(i), & \text{if } i \in B_k \text{ and } L_{B_k} \notin S(M_{I\text{-}B_k}) \\ 255, & \text{else} \end{cases}, \tag{3.4}$$

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. 3.1, 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.

3.3 The Proposed A^{2}GNN

3.3.1 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^{2}GNN 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 [61] or AGNN [98] 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. 3.2, 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 3.2. Then the pixel-level seed labels with high confidence will be selected as confident seed labels (section 3.3.2).

(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^{2}GNN 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.

The while algorithm can be found in algorithm 2.

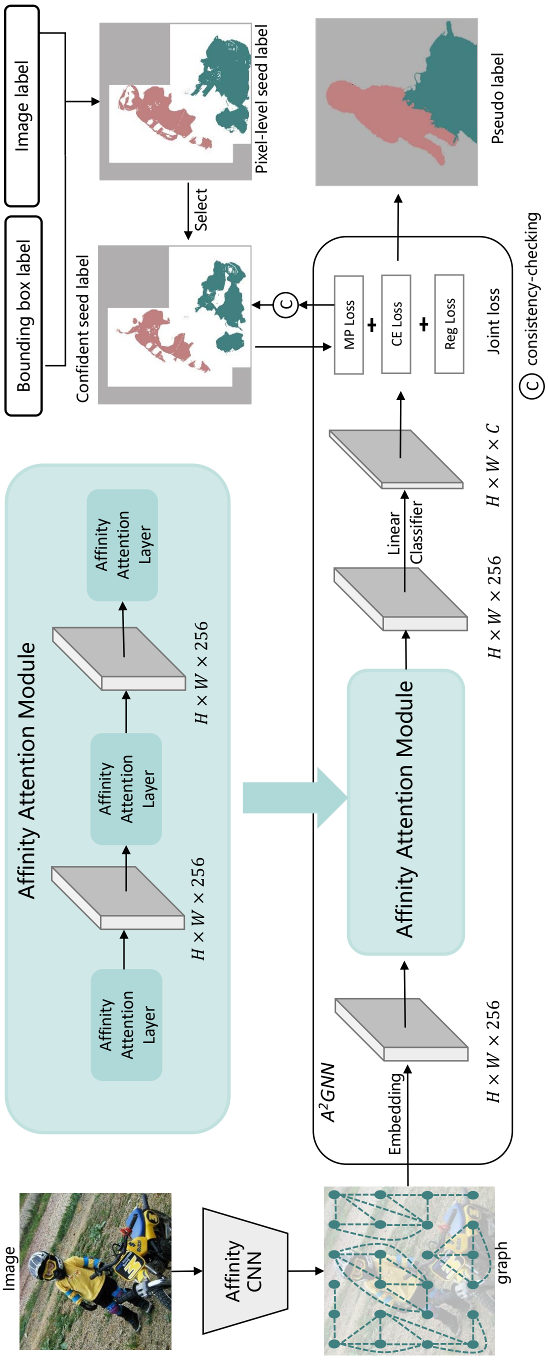

Fig. 3.2: The framework of our proposed A^{2}GNN. Firstly, we generate pixel-level seed labels using the bounding box and the image-level label. Then our affinity CNN is used to convert images to graphs. Meanwhile, we select confident labels from pixel-level seed labels as the node labels (The node labels 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.

After that, a FCN model such as Deeplab [99, 6] for bounding-box supervised semantic segmentation or MaskR-CNN [100] for bounding-box supervised instance segmentation is trained using above pixel-level pseudo labels as supervision.

In the following section, we will firstly introduce how to provide useful supervision, and then we will give an explanation about how to build a graph from the image (section 3.3.3). Finally, we will introduce A^{2}GNN, including its affinity attention layer (section 3.3.4) and its loss function (section 3.3.5).

Algorithm 2: Algorithm flow of our proposed A^{2}GNN for bounding-box supervised semantic segmentation.

```
Input:
Images
image level class labels
bounding-box labels
Output:
Pixel-level pseudo labels

while iteration is true do
    Train a classification network using the image level class labels;
end
Generate the confident seed labels using the image level class labels and
  bounding-box labels following section 3.2 and section 3.3.2;

while iteration is true do
    Convert the confident seed labels to corresponding node labels using Eq. (3.6)
      and Eq. (3.7);
    Train the affinity CNN following section 3.3.3;
end
Convert the image to graph using Eq. (3.14) and Eq. (3.15);

while iteration is true do
    Train A$^2$GNN using the convert graph as input and the confident seed labels
      as supervision following section 3.3.4 and section 3.3.5;
end
Generate the final pseudo labels using the trained A$^2$GNN.
```

3.3.2 Confident Seed Label Selection

An intuitive solution is to use the pixel-level seed label M_F obtained from Eq. (3.4) 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 chapter, 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 [97] from the pixel label M_I in Eq. (3.1). Then the selected seed labels are defined as:

$$M_g(i) = \begin{cases} M_F(i), & \text{if } M_F(i) = M_B(i) \text{ or } M_F(i) = M_I'(i) \\ 255, & \text{else} \end{cases}, \tag{3.5}$$

where 255 means that the label is unknown. M_B and M_F are obtained from Eq. (3.3) and Eq. (3.4), respectively. Fig. 3.2 (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 bounding-box supervised semantic segmentation 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. 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 3.3.5 since they rely on the output of our A^{2}GNN.

3.3.3 Graph Construction

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. Firstly, instead of regrading one superpixel as a node, it views a pixel as one node which introduces less noise. Secondly, the affinity CNN uses node affinity labels as training supervision, which ensures to generate suitable node features for this specific task, while previous *GraphNet* [1] uses classification supervision for training. Thirdly, compared to the short distance unweighted graph (edges are only represented as 0 and 1) built in *GraphNet* [1], 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 [18, 21, 101, 102] that use all noisy labels in Eq. (3.4) as supervision, our affinity CNN only uses the confident seed labels as defined in Eq. (3.5) 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. (3.5):

$$L_A(i,j) = \begin{cases} 1, & (i,j) \in R_{pair} \text{ and } M'_g(i) = M'_g(j) \\ 0, & (i,j) \in R_{pair} \text{ and } M'_g(i) \neq M'_g(j) \\ 255, & else \end{cases}, \tag{3.6}$$

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:

$$\begin{aligned} R_{pair} = \{(i,j) | M'_g(i) \neq 255 \text{ and } M'_g(j) \neq 255 \\ \text{and } ||Pos(i) - Pos(j)||_2 \leqslant r\}, \end{aligned} \tag{3.7}$$

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 [18], L1 distance is applied to compute the relationship of the two pixels i and j in F_{A}:

$$D(i,j) = exp(-\frac{\|F_A(i) - F_A(j)\|}{d_A}), \tag{3.8}$$

where d_A is the channel dimension of feature map F_A.

The training loss of affinity CNN is defined as:

$$\mathcal{L}_{\text{Aff}} = \mathcal{L}_{\text{Ac}} + \lambda \mathcal{L}_{\text{Ar}}. \tag{3.9}$$

In Eq. (3.9), $\mathcal{L}_{\text{Ac}}$ is a cross-entropy loss which focuses on using the annotated affinity labels as supervision:

$$\begin{aligned} \mathcal{L}_{\text{Ac}} = & -\frac{1}{|A^+|} \sum_{(i,j) \in A^+} L_A(i,j) log(D(i,j)) \\ & -\frac{1}{|A^-|} \sum_{(i,j) \in A^-} (1 - L_A(i,j)) log(1 - D(i,j)), \end{aligned} \tag{3.10}$$

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

$$R_{Ar} = \{(i,j) | \, \|Pos(i) - Pos(j)\|_2 \leqslant r\}. \tag{3.11}$$

Then the affinity regularized loss is defined as:

$$\mathcal{L}_{Ar} = \sum_{i=1}^{\text{HW}} \sum_{(i,j) \in R_{Ar}} G(i,j) \frac{\|(F_A(i) - F_A(j))\|}{d_A}, \tag{3.12}$$

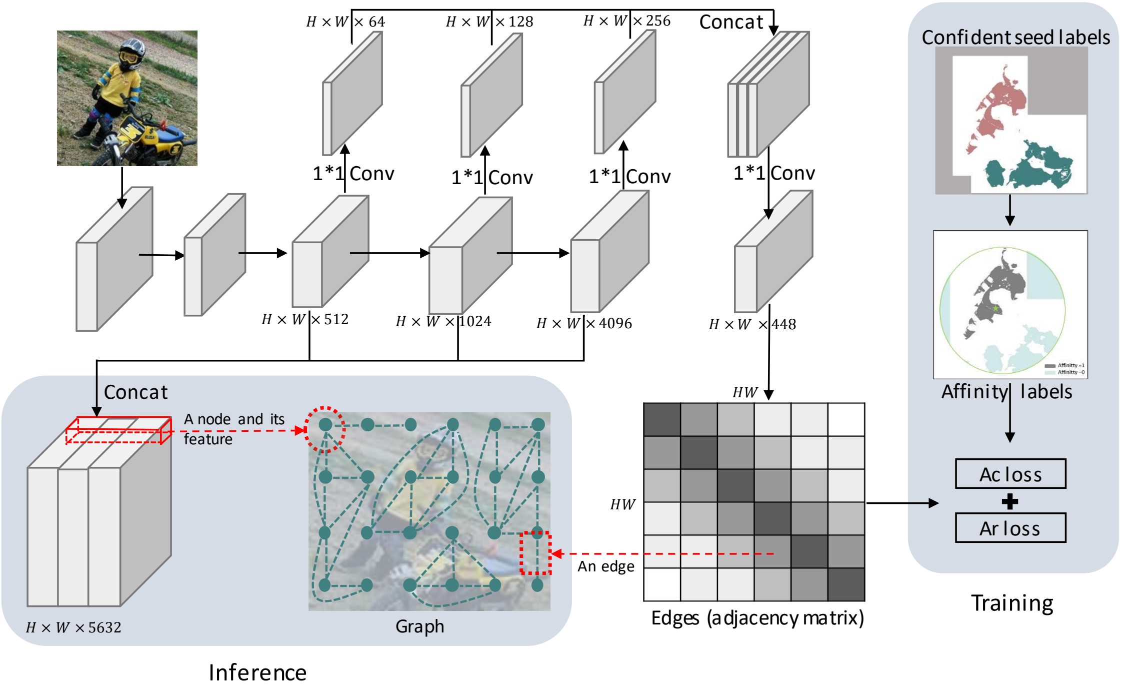

Fig. 3.3: Converting an image to a graph using our affinity CNN. During inference, the given image will be converted to a graph, in which a node is a pixel in the concatenated feature maps from the last three blocks and its feature is the corresponding pixel feature. The weight of graph edges is defined 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.

where $G(\cdot, \cdot)$ is a Gaussian bandwidth filter [10], which utilizes the color and spatial information:

$$G(i, j) = exp(-\frac{\|Pos(i)-Pos(j)\|_2}{2\sigma_{xy}^2} - \frac{\|Cor(i)-Cor(j)\|_2}{2\sigma_{rgb}^2}) \cdot [i \neq j], \quad (3.13)$$

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.

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. 3.3. Specifically, the node v_i and its feature x_i corresponds to *i*th 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:

$$E_{ij} = \begin{cases} D(i,j), & \text{if } D(i,j) > \sigma \\ 0, & \text{else} \end{cases}, \tag{3.14}$$

where i and j are pixels in the feature map, and $D(i,j)$ is obtained from Eq. (3.8). 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:

$$x_{i,j} = x_{i,j} / \sum_{j=1}^{D_g} (x_{i,j}), \tag{3.15}$$

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

3.3.4 Affinity Attention Layer

Effective GNN architectures have been studied in existing works [61, 98], where most of them are designed based on the assumption that the graph node and edge information is 100% accurate. However, in the bounding-box supervised semantic segmentation task, it is not the case. We propose a new GNN layer with attention mechanism to mitigate this issue. As shown in Fig. 3.2, in the proposed A^{2}GNN, 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:

$$H^1 = ReLU(XW^0), \tag{3.16}$$

where X is the feature matrix defined in section 3.3.3 and W^0 is the parameter set of the embedding layer. Then we design several affinity attention layers to leverage the edge weights:

$$H^{l+1} = P^l H^l, \tag{3.17}$$

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:

$$\begin{aligned} P^l(i,j) &= \text{softmax}(w^l \cos(H^l(i), H^l(j)) + \beta[\cos(H^l(i), H^l(j)) > 0]E_{ij}) \\ &= \frac{\exp\left\{w^l \cos(H^l(i), H^l(j)) + \beta[\cos(H^l(i), H^l(j)) > 0]E_{ij}\right\}}{\sum\limits_{v_j \in S(i)} \exp\left\{w^l \cos(H^l(i), H^l(j)) + \beta[\cos(H^l(i), H^l(j)) > 0]E_{ij}\right\}}, \end{aligned} \tag{3.18}$$

where $l \in \{1, 2, ..., L\}$ is the layer index (L is set as 3 in our model) of A^{2}GNN 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. (3.14). β is a weighting factor. The final output is:

$$O = \text{softmax}(H^{L+1}W^{L+1}), \tag{3.19}$$

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

Fig. 3.4 shows the flowchart of our affinity attention layer. Compared to GCN layer [61] and AGNN layer [103], our affinity attention layer makes full advantage of node similarity and edge weighting information.

3.3.5 Training of A^2GNN

As described in section 3.3.2, 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^{2}GNN. Specifically, we design a new joint loss function, including a cross-entropy loss, a regularized shallow loss [12] and a multi-point (MP) loss:

$$\mathcal{L}_G = \mathcal{L}_{ce} + \mathcal{L}_{mp} + \lambda_1 \mathcal{L}_{reg}, \tag{3.20}$$

where $\mathcal{L}_{ce}$ is the cross-entropy loss to use the labeled nodes M_g generated in section 3.3.2. $\mathcal{L}_{reg}$ 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.

Cross-Entropy Loss

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

$$\mathcal{L}_{ce} = -\frac{1}{|V^l|} \sum_{\substack{c_i \in C \\ v_j \in V^l}} [c_i = M_g(v_j)] log(O^{c_i}(v_j)), \tag{3.21}$$

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.

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

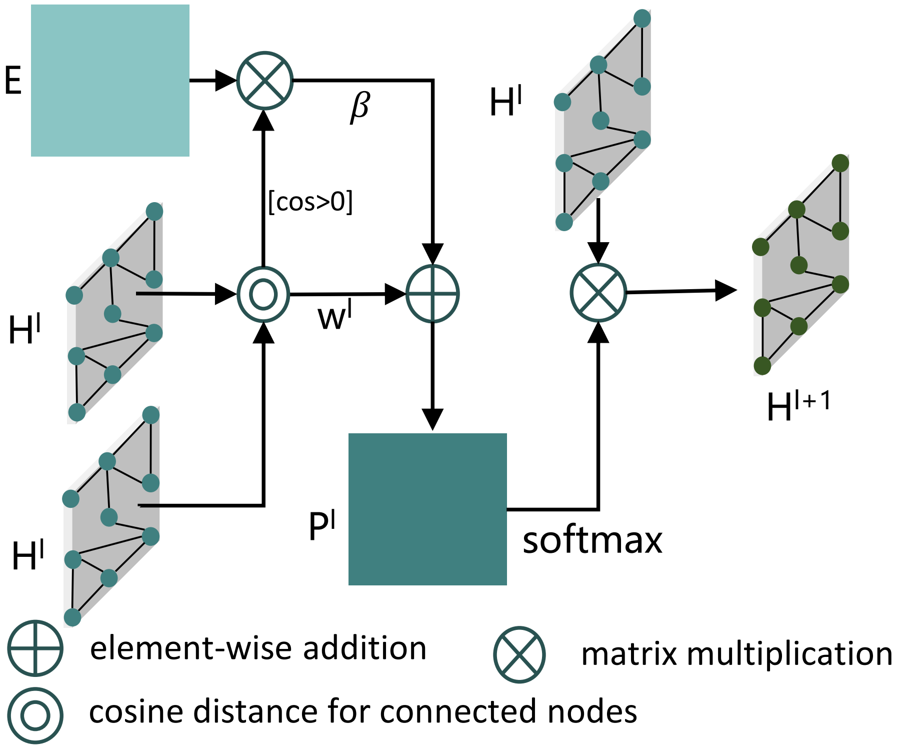

Fig. 3.4: Our proposed affinity attention layer. E is the adjacent matrix which provides soft edges information. H^l is the input feature of the layer and H^{l+1} is the output feature. P^l is the computed affinity attention matrix. w^l is the learning parameter and β is the weighting factor. With the attention mechanism and the soft edges, it can ensure accurate feature propagation.

position).

$$\mathcal{L}_{reg} = \sum_{\substack{c_i \in C \\ v_a \in V}} \sum_{\substack{c_j \in C \\ v_b \in V}} G(v_a, v_b) O^{c_i}(v_a) O^{c_j}(v_b). \tag{3.22}$$

In Eq. (3.22), 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. (3.13).

Multi-Point Loss

Inspired by [16], 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 3.3.2 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 3.2, 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:

$$i_{\max}^{l_m} = \text{index}(\max_{i \in B_j^{l_m}}(O^{B_j}(i))), \tag{3.23}$$

where l_m means the *m*th row/column, $\max\limits_{i \in B_j^{l_m}}(O^{B_j}(i))$ 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. $i_{\max}^{l_m}$ is the index of the selected node in the *m*th row/column.

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

$$K_j = \left\{ i_{\max}^{l_1}, i_{\max}^{l_2}, i_{\max}^{l_3}, ..., i_{\max}^{l_{(w+h)}} \right\}, \tag{3.24}$$

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:

$$K = \{K_1, K_2, ..., K_M\}, \tag{3.25}$$

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

$$\begin{aligned} \mathcal{L}_{mp} = &-\frac{1}{N_p} \sum_{K_j \in K} \sum_{k_i \in K_j} log(O^{B_j}(k_i)) + \\ &\frac{1}{N_f} \sum_{K_j \in K} \sum_{\substack{k_m \in K_j \\ k_n \in K_j}} [k_m \neq k_n](d(H(k_m), H(k_n))). \end{aligned} \tag{3.26}$$

In Eq. (3.26), $d(\cdot,\cdot)$ 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. (3.24)) 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.

Consistency-Checking

As mentioned in section 3.3.2, 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 firstly generate a prototype using the feature of all additional labels inside the box:

$$H_P^{K_j} = \frac{1}{N_{K_j}} \sum_{k_i \in K_j} H(k_i), \tag{3.27}$$

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. (3.5) 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:

$$M_g^{K_j}(i) = \begin{cases} M_g^{K_j}(i), & \text{if } d(H^{K_j}(i), H_P^{K_j}) > 0 \\ & \text{and } M_g^{K_j}(i) = L_{B_j} \\ 255, & \text{else} \end{cases}, \tag{3.28}$$

where $M_g^{K_j}$ is the selected confident label map of the jth bounding box (section 3.3.2), 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.

3.4 Implement Details

To generate pixel-level seed label from image-level label, we use the same classification network as SEAM [57], which is a ResNet-38 [83]. All the parameters are kept the same as in [57].

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. 3.3, 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

Table 3.1: Comparison with other approaches on PASCAL VOC 2012 *val* and *test* sets for bounding-box supervised semantic segmentation. 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 [5]	-	-	F	62.3	67.6	-	70.3
Deeplab-Vgg [99]	TPAMI'18	-	F	68.8	71.5	-	72.6
Deeplab-Resnet101 [99]	TPAMI'18	-	F	75.6	76.8	-	79.7
PSPNet [104]	CVPR'17	-	F	79.2	-	82.6	-
ScribbleSup [11]	CVPR'16	Deeplab-Vgg	S	-	63.1	-	-
RAWKS [105]	CVPR'17	Deeplab-V1	S	-	61.4	-	-
Regularized Loss [12]	ECCV'18	Deeplab-Resnet101	S	73.0	75.0	-	-
Box-Sup [13]	CVPR'15	Deeplab-V1	B	-	62.0	-	64.6
WSSL [44]	CVPR'15	Deeplab-V1	B	-	60.6	-	62.2
GraphNet [1]	ACMM'18	Deeplab-Resnet101	B	61.3	65.6	-	-
SDI [14]	CVPR'17	Deeplab-Resnet101	B	-	69.4	-	-
BCM [15]	CVPR'19	Deeplab-Resnet101	B	-	70.2	-	-
Lin *et al.* [106]	ECCV'18	PSPNet	B	-	74.3	-	-
Box2Seg [37]	ECCV'20	UperNet [107]	B	74.9	76.4	-	-
Box2Seg-CEloss [37]	ECCV'20	UperNet [107]	B	72.7	-	-	-
A^{2}GNN (ours)	-	Deeplab-Resnet101	B	**72.2**	**73.8**	**72.8**	**74.4**
A^{2}GNN (ours)	-	PSPNet	B	**74.4**	**75.6**	**73.9**	**74.9**
A^{2}GNN (ours)	-	Tree-FCN [9]	B	**75.1**	**76.5**	**74.5**	**75.2**

final feature map F_A. Following [18], we set $r = 5$ for both training and inference. λ in Eq. (3.9) is set to 3 and $\sigma_{xy} = 6$, $\sigma_{rgb} = 0.1$.

Our A^{2}GNN has five layers as mentioned in section 3.3.4, the output channel number for the first layer and three affinity attention layers are 256. λ_1 in Eq. (3.20) are set as 0.01. In $\mathcal{L}_{reg}$, we adopt the same parameters with Eq. (3.13). We use Adam as optimizer [108] 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 [60] is used as the post-processing method during inference. The unary potential of CRF uses the final output probability O in Eq. (3.19) while pair-wise potential corresponds to the color and spatial position of different nodes. All CRF parameters are the same as [18, 97]. Note that for the bounding-box supervised instance segmentation 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 bounding-box supervised semantic segmentation task, we take the Deeplab-Resnet101 [99], PSPNet [104] and Tree-FCN [9] as our fully supervised semantic segmentation models for fair comparison. For the bounding-box supervised instance segmentation task, MaskR-CNN [100] is taken as the final instance segmentation model and we use Resnet-101 as the backbone. Following the same post-processing with [16], we use CRF [60] 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^{2}GNN takes about 16 hours.

3.5 Experiment

3.5.1 Datasets

We evaluate our method on PASCAL VOC 2012 [109] and COCO [110] dataset. For PASCAL VOC 2012, the augmented data SBD [111] 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) [112] 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 [113], mAP, mAP$^r_{0.5}$, mAP$^r_{0.75}$, mAP$_s$, mAP$_m$ and mAP$_l$ are reported.

3.5.2 Comparison with State-of-the-Art

Weakly supervised semantic segmentation: In Table 3.1, we compare the performance between our method and other state-of-the-art approaches for bounding-box supervised semantic segmentation. For using deeplab as the segmentation model, it can be seen that

Table 3.2: Comparison with other approaches on PASCAL VOC 2012 *val* dataset for bounding-box supervised instance segmentation.

Method	Pub.	Sup.	$\text{mAP}^r_{0.5}$	$\text{mAP}^r_{0.7}$	$\text{mAP}^r_{0.75}$
SDS [112]	ECCV'14	F	49.7	25.3	-
MaskR-CNN [100]	ICCV'17	F	67.9	52.5	44.9
PRM [114]	CVPR'18	I	26.8	-	9.0
IRN [21]	CVPR'19	I	46.7	23.5	-
SDI [14]	CVPR'17	B	44.8	-	16.3
BBTP [16]	NeurIPS'19	B	58.9	30.4	21.6
A^{2}GNN (ours)	-	B	**59.1**	**35.5**	**27.4**

our approach obtains 96.1% of our upper-bound with pixel-level supervision (Deeplab-Resnet101 [99] with CRF). Compared to the other approaches, our approach gives a new state-of-the-art performance. Specifically, our approach with deeplab-resnet101 [9] outperforms Box-Sup [13], WSSL [44] by big margins, approximately 11.8% and 13.2%, respectively. Besides, compared to *GraphNet* [1], 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 [14], which uses MCG [58] and BSDS [59] as extra pixel-level supervision. When using PSPNet [104] as the segmentation model, our approach obtains 74.4% mIoU without CRF as post-processing, which is even higher than the results in [106] with CRF. Finally, our method with Tree-FCN [9] outperforms the state-of-the-art Box2Seg [37] 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 3.2, we compare our approach to other state-of-the-art approaches on bounding-box supervised instance segmentation. 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 [14], 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 [16], 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

Table 3.3: Comparison with other approaches on COCO test-dev dataset for weakly supervised instance segmentation. E: extra dataset [115] with instance-level annotation. S^{4}Net: salient instance segmentation model [116].

Method	Pub.	Sup.	mAP	mAP$^r_{0.5}$	mAP$^r_{0.75}$	AP$_s$	AP$_m$	AP$_l$
MNC [117]	CVPR'16	F	24.6	44.3	24.8	4.7	25.9	43.6
Mask-RCNN [100]	ICCV'17	F	37.1	60.0	39.6	35.3	35.3	35.3
Fan *et.al.* [113]	ECCV'18	I+E+S^{4}Net	13.7	25.5	13.5	0.7	15.7	26.1
LIID [51]	PAMI'20	I+E+S^{4}Net	16.0	27.1	16.5	3.5	15.9	27.7
A^{2}GNN (ours)	-	B	**20.9**	**43.9**	**17.8**	**8.3**	**20.1**	**31.8**

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 [112].

In Fig. 3.5, 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 [14][1] for the bounding-box supervised semantic segmentation task and BBTP [16][2] for the bounding-box supervised instance segmentation task. It can be seen that compared to other approaches, our approach produces better segmentation masks covering object details.

In Table 3.3, 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 [51], with an increase of 16.8% on mAP$^r_{50}$. Furthermore, our approach even performs competitive with fully-supervised approach MNC [117], which also indicates the effectiveness of our approach.

In Fig 3.6, we show some qualitative results of our A^{2}GNN on COCO *val* set for bounding box supervised instance segmentation. It can be seen that our approach can remain segmentation details for both large and small objects.

[1] we use a re-implement code from: github.com/johnnylu305

[2] github.com/chengchunhsu/WSIS_BBTP

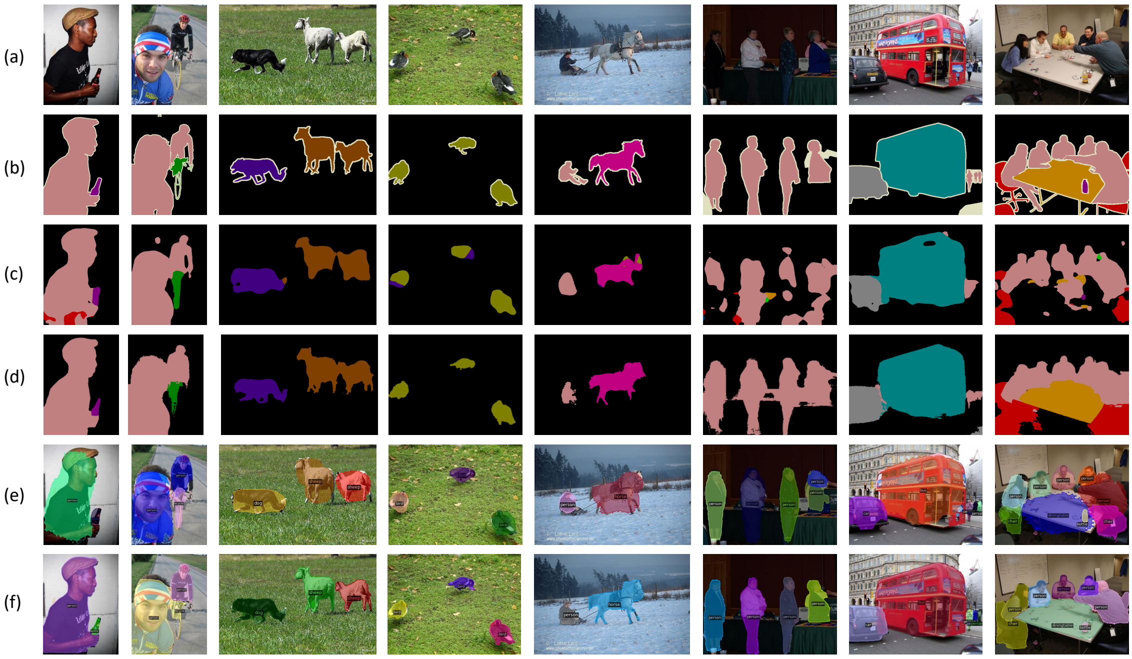

Fig. 3.5: Qualitative results of our A^{2}GNN and other state-of-the-art approaches on PASCAL VOC 2012 *val* dataset. (a) Original image. (b) Ground truth of semantic segmentation. (c) SDI [14] for bounding-box supervised semantic segmentation. (d) Our results for bounding-box supervised semantic segmentation. (e) BBTP [16] for bounding-box supervised instance segmentation. (f) Our results for bounding-box supervised instance segmentation.

3.5.3 Ablation Studies

Since the pseudo labels for bounding-box supervised instance segmentation are generated from the bounding-box supervised semantic segmentation task, in this section, we will conduct ablation studies only on the bounding-box supervised semantic segmentation task. We simply evaluate the pseudo label mIoU on the *training* set, without touching the *val* and *test* set.

In Fig 3.7, we make a comparison between our A^{2}GNN and others for bounding-box supervised semantic segmentation. It can be seen that our A^{2}GNN performs much better than other GNNs, with an improvement of 1.9% mIoU over AGNN [98] when only using the cross-entropy loss, and the full A^{2}GNN outperforms AGNN [98] by a large margin (6.9%).

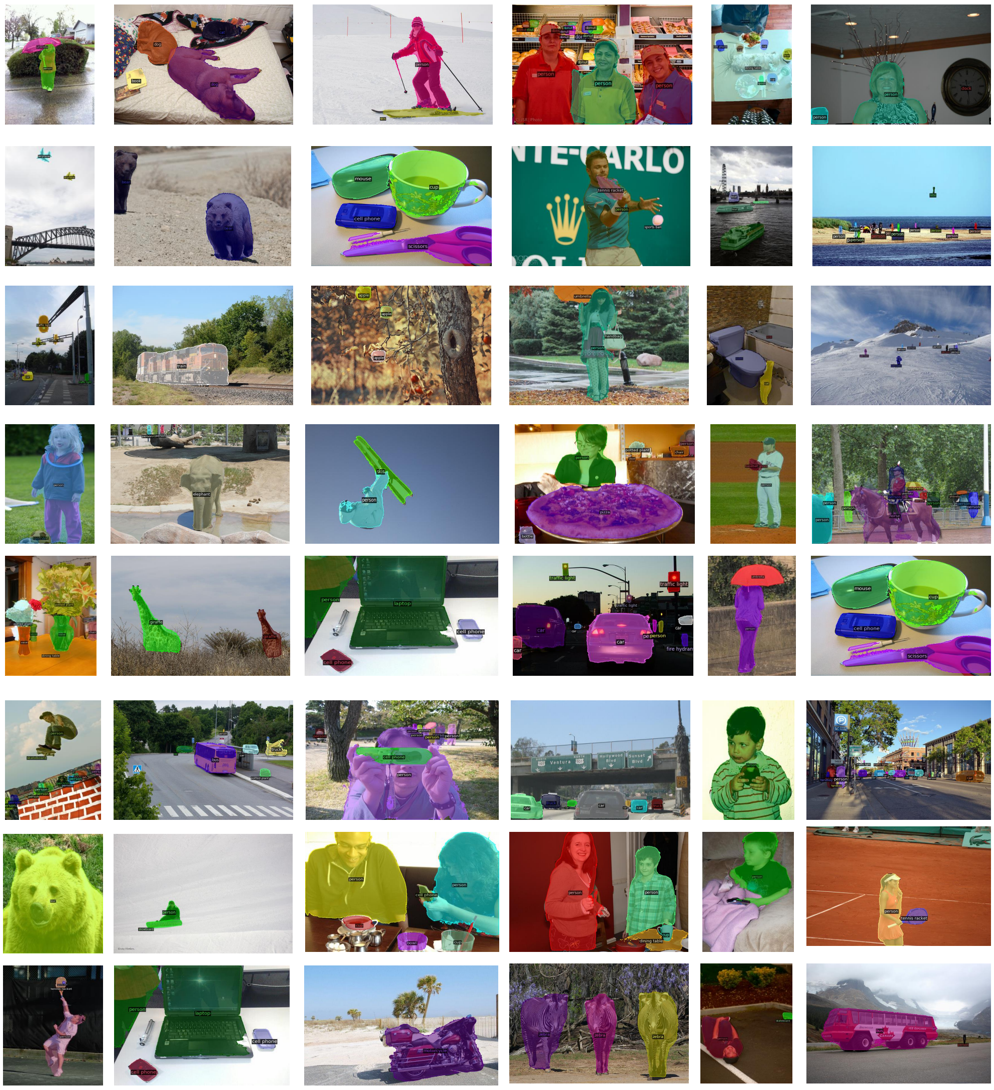

Fig. 3.6: Qualitative results of our A^{2}GNN on COCO *val* set for bounding-box supervised instance segmentation.

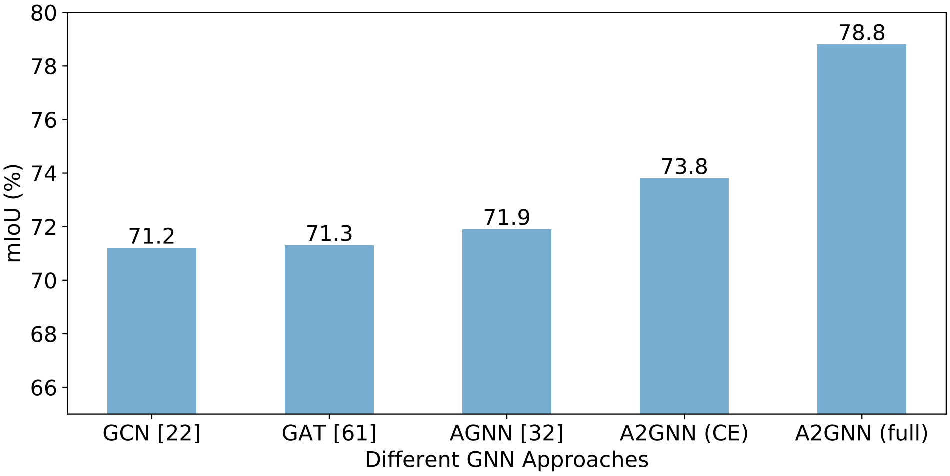

Fig. 3.7: Comparison between our A^{2}GNN and other GNNs (GCN [61], GAT [118], AGNN [98]) on Pascal VOC 2012 *training* set. “CE” means only cross-entropy loss is used.

Table 3.4: Evaluation for different modules in our approach. *RW*:random walk [57]. *H*: affinity attention layer. *C.C.*: consistency-checking.

Baseline	affinity CNN		*RW*	A^{2}GNN				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**

In Table 3.4, we explore the influence of different modules in our approaches to generate pseudo labels. *Baseline* means that we use SEAM [57] to generate the foreground seed labels and then use bounding box supervision to generate the background. *RW* means that we follow SEAM [57] 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.

Table 3.5: Evaluation for the loss functions of our A^{2}GNN. *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**

In Table 3.5, we study the effectiveness of our joint loss function. It can be seen that compared to A^{2}GNN 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%.

Table 3.6: 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**

In Table 3.6, we study different ways to build our graph. *Superpixel (S.P.)* means that we adopt [1] to produce graph nodes and their features. *Distance (Dis.)* means that we build the graph edge using L_1 distance of feature map [1]. 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 can build a more accurate graph than other approaches.

Table 3.8: 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**

Table 3.7: Evaluation of the affinity attention layer in A^{2}GNN.

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

In Table 3.7, 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^{2}GNN, which indicates the effectiveness of our designed GNN layer.

In Table 3.8, 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^{2}GNN. Specifically, we make a comparison of the results when using M_F (in Eq. (3.4)) and M_g (in Eq. (3.5)) as supervision for our A^{2}GNN, 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.

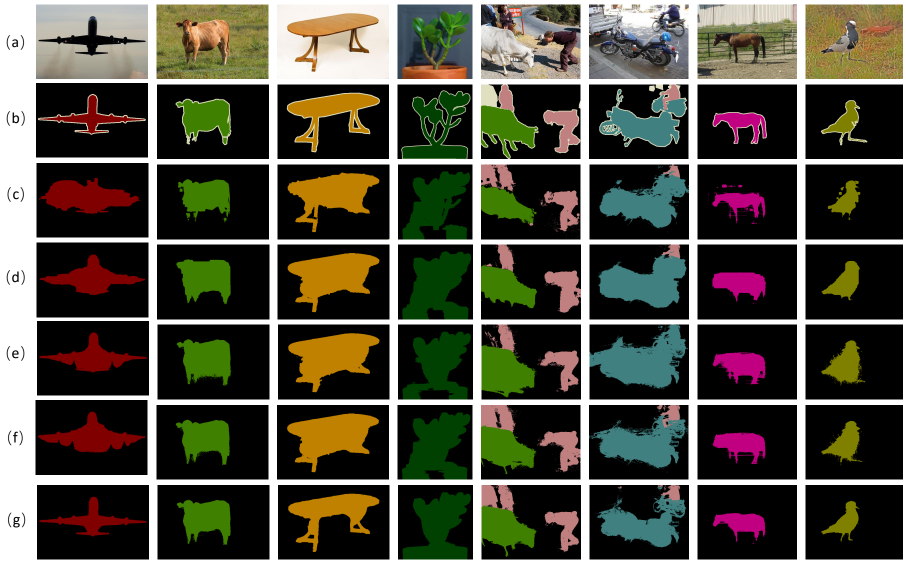

Fig. 3.8: Qualitative Results of the generated pseudo labels on PASCAL VOC 2012 *training* set for bounding box supervised semantic segmentation. (a) original images. (b) ground-truth. (c) A^{2}GNN ($\mathcal{L}_{ce}$). (d) A^{2}GNN ($\mathcal{L}_{ce}$+$\mathcal{L}_{reg}$). (e) A^{2}GNN ($\mathcal{L}_{ce}$+$\mathcal{L}_{reg}$+$\mathcal{L}_{mp}$ (without feature operator)). (f) A^{2}GNN ($\mathcal{L}_{ce}$+$\mathcal{L}_{reg}$+$\mathcal{L}_{mp}$). (g) A^{2}GNN ($\mathcal{L}_{ce}$+$\mathcal{L}_{reg}$+$\mathcal{L}_{mp}$+*c.c.*).

In Fig 3.8, we report some qualitative comparison of our proposed A^{2}GNN on the PASCAL VOC 2012 *training* dataset for bounding box supervised semantic segmentation. "Without feature operator" means we do not introduce the feature distance operator in our MP loss. "*C.C*" means the consistency-checking mechanism. From (e) and (f), it can be found that using feature operator can retain more object details, and without the feature operator in the MP loss, some foreground pixels are misclassified. And it can be found in (g) that using consistency-checking mechanism significantly removes many noisy labels.

3.6 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 3.3.5) and the consistency-checking (section 3.3.5) 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 bounding-box supervised semantic segmentation 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.

Algorithm 3: Algorithm flow of our proposed A^{2}GNN for other weakly supervised semantic segmentation.

```
Input:
Images
Weak labels (scribble, point and image level class labels)
Output:
Pixel-level pseudo labels

while iteration is true do
    Train a classification network using the image level class labels;
end
Generate the confident seed labels using the weak labels following section 3.6.1;

while iteration is true do
    Convert the confident seed labels to corresponding node labels using Eq. (3.6)
      and Eq. (3.7);
    Train the affinity CNN following section 3.3.3;
end
Convert the image to graph using Eq. (3.14) and Eq. (3.15);

while iteration is true do
    Train A$^2$GNN using the convert graph as input and the confident seed labels
      as supervision, using Eq. (3.21) and Eq.( 3.22);
end
Generate the final pseudo labels using the trained A$^2$GNN.
```

3.6.1 Pixel-level Seed Label Generation

As mentioned in section 3.2, 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 to pixel-level seed labels.

Image-level supervision: We directly use M_I' defined in Eq. (3.5) to train our affinity CNN and use it as M_g to train our A^{2}GNN. 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 [38] 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^{2}GNN, 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^{2}GNN, we use a superpixel method [38] 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^{2}GNN, we use the same setting with our bounding box task.

The whole algorithm can be found in algorithm 3.

Table 3.9: Performance comparison in mIoU (%) for evaluating the pseudo labels on the PASCAL VOC training data set.

Method	Pub.	Sup.	mIoU (%)
PSA [18]	CVPR'18	I	58.4
ICD [50]	CVPR'20	I	62.2
SubCat [92]	CVPR'20	I	63.4
SEAM [57]	CVPR'20	I	63.6
A^{2}GNN (ours)	-	I	**65.3**
A^{2}GNN (ours)	-	I+E	**66.5**
Box2Seg [37]	ECCV'20	B	73.6*
A^{2}GNN (ours)	-	B	**78.8**

* Reproduce by ourself.

3.6.2 Experimental Evaluations

In Table 3.9, 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 [57], our approach obtains 1.7% mIoU improvement. We also compare the quality of the pseudo labels between our approach and Box2Seg [37]. It can be seen that our method outperforms Box2Seg [37] by a large margin, with 5.2% mIoU improvement.

In Fig. 3.9, 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.

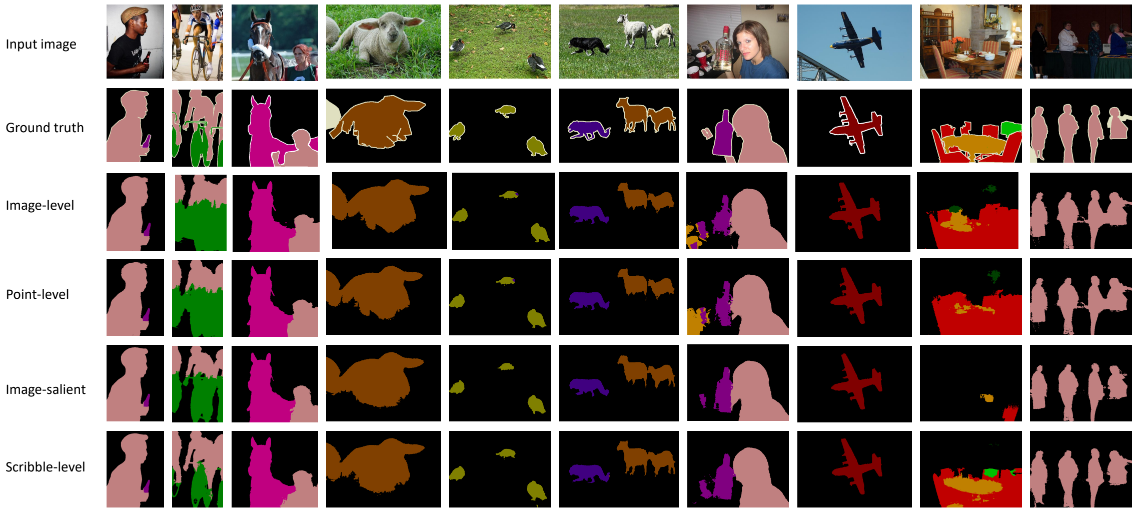

Fig. 3.9: Qualitative results of our A^2GNN on PASCAL VOC 2012 *val* dataset. We show the results from different levels of supervision signals (3rd – 6th rows). Stronger supervision signals (*e.g.*, scribble) produce more accurate results than weaker signals (*e.g.*, point, image-level label).

In Table 3.10 and 3.11, we compare the performance between our method and other state-of-the-art weakly supervised semantic segmentation approaches.

Table 3.10: Comparison with other state-of-the-arts on PASCAL VOC 2012 *val* and *test* datasets for scribble and ponit level supervision. Sup.: Segmentation model. F: fully supervised. S: scribble. P: point. "highlight" means the best performance for a specific task.

Method	Pub.	Seg.	Sup.	*val*	*test*
(1) Deeplab-V1 [5]	-	-	F	67.6	70.3
(2) Deeplab-Vgg [99]	TPAMI'18	-	F	71.5	72.6
(3) Deeplab-Resnet [99]	TPAMI'18	-	F	76.8	79.7
(4) Tree-FCN [9]	NeurIPS'19	-	F	80.9	-
RAWKS [105]	CVPR'17	(1)	S	61.4	-
ScribbleSup [11]	CVPR'16	(2)	S	63.1	-
GraphNet [1]	ACMM'18	(3)	S	73.0	-
Regularized loss [12]	ECCV'18	(3)	S	75.0	-
A^{2}GNN (ours)	-	(3)	S	74.3	74.0
A^{2}GNN (ours)	-	(4)	S	**76.2**	**76.1**
What's the point [17]	ECCV'16	(2)	P	43.4	43.6
Regularized loss [12]	ECCV'18	(3)	P	57.0	-
A^{2}GNN(ours)	-	(3)	P	**66.8**	**67.7**

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 [17] and [12], our method increases 23.4% and 9.8% in mIoU on PASCAL VOC 2012 *val* dataset, respectively.

For the image-level supervision task, our A^{2}GNN achieves mIoU of 66.8% and 67.4% on *val* and *test* set, respectively. It should be noticed that PSA [18], SEAM [57] and CONTA [94] apply Wider ResNet-38 [83] as segmentation model, which has a higher upper-bound than Deeplab-Resnet101 [99]. Using Deeplab-Resnet101 [99] as the segmentation, Subcat [92] 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 [99], being 87.0% of our upper-bound (76.8% mIoU score with Deeplab-Resnet [99]) on *val* set.

Table 3.11: Comparison with other state-of-the-arts on PASCAL VOC 2012 *val* and *test* datasets for image level supervision. Sup.: Segmentation model. F: fully supervised. 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 [5]	-	-	F	67.6	70.3
(2) Deeplab-Resnet [99]	TPAMI’18	-	F	76.8	79.7
(3) WiderResnet38 [83]	PR’19	-	F	80.8	82.5
AE-PSL [31]	CVPR’17	(1)	I+E	55.0	55.7
DSRG [34]	CVPR’18	(2)	I+E	61.4	63.2
FickleNet [55]	CVPR’19	(2)	I+E	64.9	65.3
Zhang *et.al* [85]	ECCV’20	(2)	I+E	66.6	66.7
ICD [50]	CVPR’20	(2)	I+E	67.8	68.0
EME [86]	ECCV’20	(2)	I+E	67.2	66.7
MCIS [87]	ECCV’20	(2)	I+E	66.2	66.9
ILLD [51]	TPAMI’20	(2)	I+E	66.5	67.5
ILLD [51]	TPAMI’20	(2)†	I+E	**69.4**	**70.4**
A^{2}GNN(ours)	-	(2)	I+E	68.3	68.7
A^{2}GNN(ours)	-	(2)†	I+E	69.0	69.6
PSA [18]	CVPR’18	(3)	I	61.7	63.7
SEAM [57]	CVPR’20	(3)	I	64.5	65.7
ICD [50]	CVPR’20	(2)	I	64.1	64.3
BES [93]	ECCV’20	(2)	I	65.7	66.6
SubCat [92]	CVPR’20	(2)	I	66.1	65.9
CONTA [94]	NeurIPS’20	(3)	I	66.1	66.7
A^{2}GNN(ours)	-	(3)	I	**66.8**	**67.4**

† means using Res2Net [119] as the backbone.

Besides, for the image-level supervision, some approaches [34, 55, 50, 51] used salient model with extra pixel-level salient dataset [45] or instance pixel-level salient dataset [115] to generate more accurate pseudo labels. Follow these approaches, we also use saliency models. Specifically, we use the saliency approach [120] following ICD [50] to produce the initial seed labels, and then use our approach to produce the final pseudo labels. It can be seen from Table 3.11 that our approach outperforms other approaches (using ResNet101 as the backbone). Following ILLD [51], we also evaluate our approach using Res2Net [119] 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.,* [51], 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.

3.7 Conclusion

We have proposed a new system, A^{2}GNN, 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^{2}GNN can produce more accurate pseudo labels.

Chapter 4

Dynamic Feature Regularized Loss for Scribble Supervised Semantic Segmentation

4.1 Motivation

As mentioned in Sect. 1.2.4, for scribble supervised semantic segmentation, most recent approaches can be divided into two main categories: pseudo-label based approaches [1, 62] and loss function based approaches [10, 12, 40, 42]. Pseudo-label based approaches focus on generating accurate pseudo labels through expanding the initial annotations. But such approaches usually need multi-stage training process with many bells and whistles. Loss function based approaches concentrate on directly utilizing limited labels to train the segmentation model with well-designed loss functions. However, some approaches [40, 42] rely on extra dataset [43, 41] to provide edges or boundaries information as supervision, while some loss function based approaches [10, 12] still need multi-round training procedures.

In order to overcome the aforementioned drawbacks, we propose a new end-to-end network where a new Dynamic Feature Regularized (DFR) loss function is introduced to provide more sufficient information to describe the semantic similarity of different pixels. Specifically, our network has two branches: one is the semantic segmentation head and the other one is the feature consistency head. The semantic segmentation head aims to make semantic segmentation with our proposed DFR loss, also it will provide reliable supervision for the feature consistency head. While the feature consistency head aims to ensure that the pixels which have the same semantic category can share the similar features, at the same time, it will provide accurate feature relationship for the DFR loss in the semantic segmentation head. Next, we will introduce the details of our framework.

In this chapter, the main work is from our work: “Dynamic Feature Regularized Loss

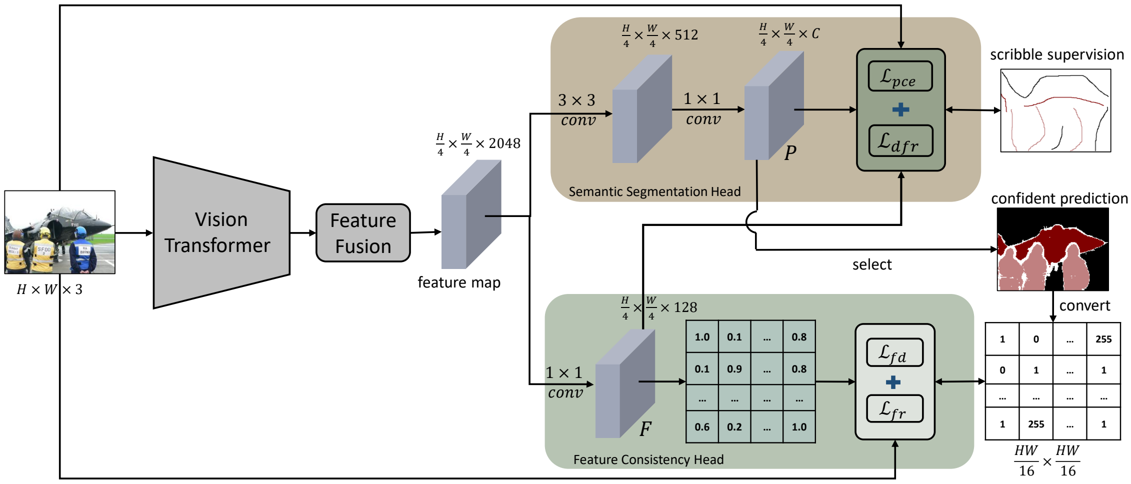

Fig. 4.1: The framework of our proposed approach. Firstly, an image is input to the vision transformer to generate its feature maps, then the feature maps from all blocks are fused to generate a shared feature map, which is input to both the semantic segmentation head and the feature consistency head. The semantic segmentation head is used to make semantic prediction and provide highly confident regions as pseudo labels for the feature consistency head. Meanwhile, the feature consistency head is used to produce consistent features for pixels with the same semantic category, which are in turn used in the regularized loss of the semantic segmentation head. Note that both the semantic segmentation head and feature consistency head are used during training while only the semantic segmentation head is used during inference.

for Weakly Supervised Semantic Segmentation" [76].

4.2 Methodology

4.2.1 Overview

Fig. 4.1 shows the overall framework of our approach. Firstly, we use vision transformer as the backbone to generate the feature maps. Then the feature maps are input to the feature fusion module to generate the shared feature map for the semantic segmentation head and feature consistency head. The semantic segmentation head is to make semantic prediction and provide supervision for the feature consistency head. The feature consistency head enforces feature consistency for pixels with the same semantic category based on the supervision from the semantic segmentation head, which in turn provides reliable dynamic feature for the semantic segmentation head.

The semantic segmentation head utilizes two loss functions: partial cross-entropy loss and our proposed dynamic feature regularized loss. Partial cross-entropy loss uses the scribble-annotation as supervision while our proposed dynamic feature regularized loss applies the original image information and feature map from the feature consistency head to produce regularized kernel.

The feature consistency head also introduces two loss functions: feature distance loss and feature regularized loss. Feature distance loss uses the predicted highly confident pseudo labels from the semantic segmentation head as supervision. Feature regularized loss solely uses the shallow feature as the kernel to compute feature distance.

The whole framework is trained in an end-to-end manner, the loss function is defined as:

$$\mathcal{L} = \underbrace{\mathcal{L}_{\text{pce}} + \lambda_1 \mathcal{L}_{\text{dfr}}}_{\text{semantic head}} + \lambda_2 (\underbrace{\mathcal{L}_{\text{fd}} + \mathcal{L}_{\text{fr}}}_{\text{feature head}}), \tag{4.1}$$

where λ_1 and λ_2 are loss weights. $\mathcal{L}_{\text{pce}}$ and $\mathcal{L}_{\text{dfr}}$ are the loss functions for the semantic segmentation head. $\mathcal{L}_{\text{pce}}$ is the partial cross-entropy loss, which uses the scribble annotation as supervision. $\mathcal{L}_{\text{dfr}}$ is our proposed regularized loss. Both of them will be introduced in Sect. 4.2.2. $\mathcal{L}_{\text{fd}}$ and $\mathcal{L}_{\text{fr}}$ are the loss functions for the feature consistency head, $\mathcal{L}_{\text{fd}}$ is the feature distance loss, which uses the prediction of the semantic segmentation head as supervision. $\mathcal{L}_{\text{fr}}$ is the feature regularized loss. Both $\mathcal{L}_{\text{fd}}$ and $\mathcal{L}_{\text{fr}}$ will be introduced in Sect. 4.2.3.

4.2.2 Semantic Segmentation Head

The semantic segmentation head is to make semantic prediction, which includes several convolution layers to produce the final probability map P, a partial cross-entropy loss to utilize the scribble annotation and our proposed dynamic feature regularized loss to restrict the prediction of the whole map.

Specifically, the partial cross-entropy loss is:

$$\mathcal{L}_{\text{pce}} = -\frac{1}{N_s} \sum_{i=1}^{hw} [M_s(i) \neq 255] log(P^t(i)), \tag{4.2}$$

where $P^t(i)$ is the probability of pixel i to be classified to the ground truth class. N_s is the annotated pixel number. h and w correspond to the height and width of the feature map, respectively. M_s is the provided scribble annotation and 255 means that there is no annotation. $[\cdot]$ is the Iverson bracket operation, which equals to 1 if the inside condition is true, otherwise it equals to 0.

For scribble annotation, the main limitation is that very few pixel-level labels are provided, *e.g.*, 3% pixels are annotated in PASCAL VOC 2012 dataset [12]. In this case, using partial cross-entropy loss is not enough. Therefore, we design a new DFR loss to impose restriction on the prediction of the model. Our intuition is that for two different pixels i and j, if their features are highly similar, the probability for them to belong to the same category is high.

In order to impose the above restriction and keep high computing efficiency, for two different pixels i and j, we only compute the loss when both of them locate within a local window:

$$R = \left\{ (i,j) \middle| |i_x - j_x| \leqslant r \text{ and } |i_y - j_y| \leqslant r \right\}, \tag{4.3}$$

where R is the effective pixel pair set. i_x and j_x represent the x-coordinate, i_y and j_y represent the y-coordinate. r is the window size.

Then our proposed DFR loss is:

$$\mathcal{L}_{\text{dfr}} = \frac{1}{hw} \sum_{i=1}^{hw} \sum_{(i,j) \in R} [i \neq j] \, \varphi(i,j), \tag{4.4}$$

where $\varphi(i,j)$ is the loss for pixels i and j, which follows the definition:

$$\begin{aligned} \varphi(i,j) &= \sum_{c \in C} \sum_{c' \in C} [c \neq c'] \, K(i,j) P^c(i) P^{c'}(j) \\ &= K(i,j) \sum_{c \in C} P^c(i) \left(1 - P^c(j)\right) \\ &= K(i,j) \left(1 - \sum_{c \in C} P^c(i) P^c(j)\right), \end{aligned} \tag{4.5}$$

where C is the class set, *i.e.*, $C = \{c_1, c_2, ..., c_N\}$. $P^c(i)$ and $P^{c'}(j)$ are the probabilities for pixels i and j to be classified to class c and c', respectively, which are provided by the network (after softmax layer). $K(i,j)$ is the regularized kernel, which is defined as a Gaussian kernel:

$$K(i,j) = exp\left(-\frac{\|S_i - S_j\|^2}{2\sigma_1^2} - \frac{\|I_i - I_j\|^2}{2\sigma_2^2} - \frac{\|F_i - F_j\|^2}{2\sigma_3^2}\right), \tag{4.6}$$

where $||\cdot||^2$ is the L2 distance. S_i and S_j correspond to the pixel positions for pixel i and j. I_i and I_j are the RGB information of pixel i and j. F_i and F_j are the deep features of pixels i and j from the feature consistency head.

The previous regularized loss functions [10, 12, 39] only adopt the position and RGB information to compute the kernel. However, both position and RGB information in Eq. (4.6) are static, once the two types of features fail to correctly describe the true relationship of a pixel pair, the network will be optimized towards an inaccurate direction, and such a problem cannot be addressed during the whole training period.

Different from the previous approaches, we introduce the dynamic deep feature, which is provided by the feature consistency head (as described in Sect. 4.2.3), to compute the regularized kernel. Note that when using deep features to compute the regularized kernel, they are regarded as non-gradient values. Through introducing dynamic feature to compute regularized kernel, on one hand, more comprehensive representation for pixel relationship is provided. On the other hand, dynamic features allow the network to correct its previous results. The remaining task is how to guarantee deep features accurately representing the relationship of different pixels, which is addressed in Sect. 4.2.3.

4.2.3 Feature Consistency Head

In order to provide correct relationship for deep features of different pixels, we design a feature consistency head. Our motivation is that for two pixels i and j, if they belong to the same class, their features should have high similarity. If they belong to different classes, the similarity of their features should be low.

Based on above analysis, we need to provide supervision for the feature relationship. We select the predicted labels with highly confident scores from the semantic segmentation head as supervision:

$$M(i) = \begin{cases} \underset{c\in C}{\text{argmax}}(P^c(i)), & \max_{c\in C}(P^c(i)) > \gamma \\ 255, & \text{else} \end{cases}, \tag{4.7}$$

where $M(i)$ is the semantic label for pixel i, 255 means that it is not annotated to any class. $P^c(i)$ is the predicted probability for class c.

Then the supervision is converted to the pair-wise pixel relationship. Following the operation in Sect. 4.2.2, we use the same local window to restrict the computing region. Considering that some pixels are not annotated, so the effective pixel pairs are:

$$\begin{aligned} R_A = \{(i,j) | M(i) \neq 255 \text{ and } M(j) \neq 255 \\ \text{and } \ (i,j) \in R\}, \end{aligned} \tag{4.8}$$

where R_A is the effective pixel pair set. R is the set defined in Eq. (4.3). After that, the supervision M is converted to the pair-wise pixel relationship label:

$$A(i,j) = \begin{cases} 1, & (i,j) \in R_A \text{ and } M(i) = M(j) \\ 0, & (i,j) \in R_A \text{ and } M(i) \neq M(j) \\ 255, & else \end{cases} \cdot \tag{4.9}$$

Eq. (4.9) indicates that when pixels i and j belong to the same class, they have strong relationship (set as 1). If they belong to different classes, they should have weak relationship (set as 0). 255 means the pixel pair is ignored. In order to utilize such supervision, we compute the feature distance as the feature relationship for the pixel pair in R_A:

$$D(i,j) = exp\left(-\frac{||F_i - F_j||}{d}\right), \tag{4.10}$$

where $||\cdot||$ is L1 distance. d is the channel dimension of the feature map. Both F_i and F_j are the final features of pixels i and j from the feature consistency head.

Finally, the feature distance loss is:

$$\begin{aligned}\mathcal{L}_{\text{fd}} = &-\frac{1}{|A^+_{\text{bg}}|}\sum_{(i,j)\in A^+_{\text{bg}}} A(i,j)log(D(i,j)) \\ &-\frac{1}{|A^+_{\text{fg}}|}\sum_{(i,j)\in A^+_{\text{fg}}} A(i,j)log(D(i,j)) \\ &-\frac{2}{|A^-|}\sum_{(i,j)\in A^-} (1-A(i,j))log(1-D(i,j)),\end{aligned} \tag{4.11}$$

where A^+_{bg} is the pixel pair set that $A(i,j) = 1$ and the label of i and j is background. A^+_{fg} is the pixel pair set that $A(i,j) = 1$ and the label of i and j is foreground. A^- corresponds to the pixel pair set that $A(i,j) = 0$. $|\cdot|$ indicates the number of elements in a set.

Following the same strategy in our semantic segmentation head, we also introduce feature regularized loss since the supervision only provide limited annotations. The feature regularized loss is defined as:

$$\mathcal{L}_{\text{fr}} = \frac{1}{hw}\sum_{i=1}^{hw}\sum_{(i,j)\in R_A}[i \neq j]K_{\text{f}}(i,j)\left(\frac{||F_i - F_j||}{d}\right), \tag{4.12}$$

where $K_{\text{f}}(i,j)$ have the similar formation with Eq. (4.6):

$$K_{\text{f}}(i,j) = exp\left(-\frac{\|S_i - S_j\|^2}{2\sigma_1^2} - \frac{\|I_i - I_j\|^2}{2\sigma_2^2}\right). \tag{4.13}$$

From Sect. 4.2.2 and Sect. 4.2.3, it can be found that both the semantic segmentation and feature consistency heads receive the online updated information. Specifically, the semantic segmentation head receives the dynamically updated feature from the feature consistency head, while the feature consistency head receives the updated supervision from the semantic segmentation head. On one hand, better supervision enables the feature consistency head to provide more accurate feature relationship. On the other hand, more accurate feature relationship facilitates to produce better semantic segmentation. Thus, we argue that with such an interaction mechanism two heads benefit from each other and the final performance is boosted accordingly.

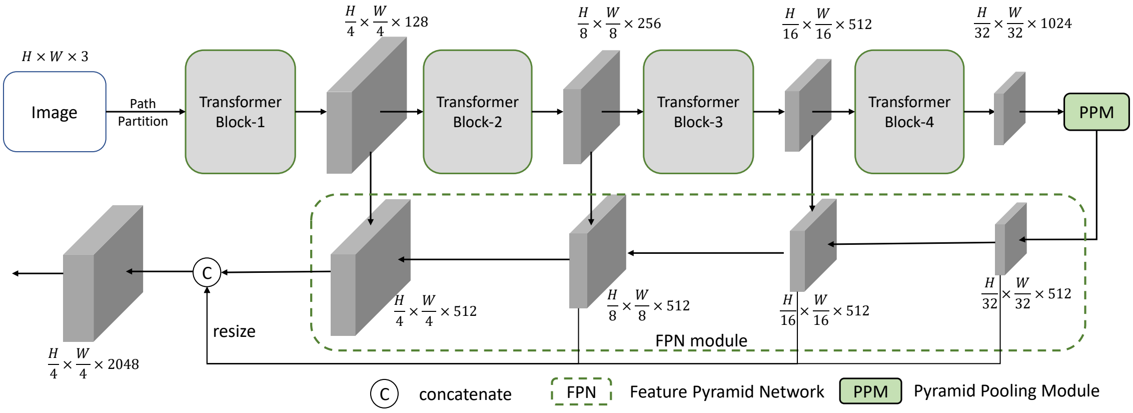

Fig. 4.2: Details of the backbone and the feature fusion process. To fuse features from all stages, we use the same architecture with Swin-transformer [64] and UperNet [107], both of which use Pyramid Pooling Module (PPM) [7] and Feature Pyramid Network (PPN) [121] to fuse feature maps.

4.3 Experiment

4.3.1 Datasets and Evaluation Metric

We evaluate our method on PASCAL VOC 2012 [109] and PASCAL CONTEXT [122] dataset. For PASCAL VOC 2012 dataset, following the previous approaches [10, 12, 39, 40] in weakly supervised semantic segmentation, the augmented data SBD [111] is also used and the whole dataset contains 10,582 images for training, 1,449 images for validating and 1,456 images for testing with 20 foreground classes. For PASCAL CONTEXT dataset, it includes 4,998 images for training and 5,105 images for validating with 59 foreground categories. For the scribble annotation, we also follow the previous approaches [10, 12, 39, 40] to use the supervision provided by ScribbleSup [11]. Mean Intersection over Union (mIoU) is adopted as the evaluation metric.

4.3.2 Implementation Details

Our approach mainly includes three network modules: the backbone, the semantic segmentation head and the feature consistency head. For the backbone, we choose Swin-Transformer-Base [64] (with UperNet head [107] to fuse the features from 4 stages). The details can be found in Fig. 4.2. After passing the backbone and the feature fusion stage, a fused feature map with a dimension of 2048 is generated. For the semantic segmentation head, we also use the same setting as in Swin-Transformer-Base [64], which uses the scene head in [107]. For the feature consistency head, we utilize a 1×1 convolutional layer followed by a ReLU function to produce the final feature, and the dimension d of the feature in this head is set as 128. In Eq. (4.1), λ_1, λ_2 are set as 1×10^{-2} and 1×10^{-3},

Table 4.1: Comparison with other state-of-the-art on PASCAL VOC 2012 dataset. Pub.: Publication. Sup.: Supervision. F: Fully-supervised. B: bounding-box level supervision. I: Image-level supervision. S: scribble-level. "ss" means single scale inference. "ms" means multi-scale inference. Multi-scale inference is used without explicit indication. "S.S." means single stage training

Method	Pub.	Sup.	Backbone	S.S.	CRF	mIoU(%)	
						val	*test*
(1) Deeplab-v2 [99]	TPAMI'18	F	vgg16	✓	✓	71.5	72.6
(2) DeepLab-v2 [99]	TPAMI'18	F	resnet101	✓	✓	76.8	79.7
(3) Deeplab-v3+ [65]	ECCV'18	F	resnet18	✓	-	76.7	-
(4) SegSort [8]	ICCV'19	F	resnet101	✓	-	77.3	-
(5) Tree-FCN [9]	NeurIPS'19	F	resnet101	✓	-	82.3	-
(6) Swin-Base (ss) [64]*	-	F	transformer	✓	-	82.9	82.9
(7) Swin-Base (ms) [64]*	-	F	transformer	✓	-	84.6	84.4
ScribbleSup [11]	CVPR'16	S	(1)	-	✓	63.1	-
RAWKS [105]	CVPR'17	S	resnet101	✓	✓	61.4	-
NormalizedCut [10]	CVPR'18	S	(2)	-	✓	74.5	-
GraphNet [1]	ACMM'18	S	(2)	-	✓	73.0	-
KernelCut+CRF [12]	ECCV'18	S	(2)	-	✓	75.0	-
GatedCRF [39]	NeurIPS'19	S	(3)	✓	-	75.5	-
BPG+CRF [42]	IJCAI'19	S	(2)	✓	✓	76.0	-
SPML+CRF [40]	ICLR'21	S	(4)	-	✓	76.1	-
A^2GNN [62]	TPAMI'21	S	(5)	-	✓	76.2	76.1
DFR-ours (ss)	-	S	(6)	✓	-	81.5	82.1
DFR-ours (ms)	-	S	(7)	✓	-	**82.8**	**82.9**

* Reproduced by ourselves.

respectively. The window size for Eq. (4.3) and Eq. (4.8) is set as 5. γ in Eq. (4.7) is 0.98. σ_1 and σ_2 are shared for Eq. (4.6) and Eq. (4.13). σ_1, σ_2 and σ_3 are set as 6, 0.5 and 50, respectively. Note that the RGB is normalized before inputting to the network to compute the kernel.

We use the weights pretrained on ImageNet-22K [123] to initialize the model of Swin-Transformer-Base [64]. AdamW [108] is used as the optimizer with an initial learning rate of 3×10^{-5} and weight decay of 0.01. Models are trained on 8 Nvidia Tesla V100 GPUs with batch size of 16 for 40K iterations. During training, we adopt the default settings in mmseg [124], including random flipping, random rescaling (range is $[0.5, 2.0]$) and random photometric distortion. The input size is 512×512. During inference, the feature consistency head is not used and multi-scale strategy is used with resolution ratios of $\{0.5, 0.75, 1.0, 1.25, 1.5, 1.75\}$. Other settings follow that in Swin-Transformer-Base [64].

4.3.3 Comparison with State-of-the-Art

In Table 4.1, we compare our approach with other approaches on PASCAL VOC 2012 dataset. It can be seen that our approach significantly outperforms other approaches. Specifically, A^2GNN [62] achieves 76.2% mIoU with dense CRF [60] as post-processing, while we achieve 82.8% mIoU without using CRF, which brings 6.6% mIoU gain. Note that A^2GNN [62] is a multi-stages method which uses more than three individual networks during training, while our approach is a single-stage method. Besides, for the single-stage method, BPG [42] achieves the best performance, but it used extra dataset (HED contour detector [43], pretrained on BSDS500 dataset [41]) to provide edge supervision. We do not rely on any extra dataset and outperform it by 9.6% mIoU without CRF (82.8% *v.s.* 73.2%). SPML used the same extra dataset as BPG [42] with multi-round training process, and we also significantly outperform it (82.8% *v.s.* 76.1%). More importantly, our approach reaches 98.3% of the upper-bound performance (the fully-supervised case for the single scale setting), showing its effectiveness for this task. It can also be found that using multi-scale strategy brings 1.3% mIoU increase. For the *test* set, our approach outperforms A^2GNN with a clear gain of 6.8%. Generally, without using any extra-dataset and post-processing, our approach outperforms other approaches by a large margin through single-stage training.

In Table 4.2, we report the per-class results on PASCAL VOC 2012 *val* set. It can be seen that our approach generates new state-of-the-art performances for each class. Note that we do not use dense CRF while the other reported approaches use dense CRF as post-processing.

In Fig. 4.3, segmentation performance comparisons with different scribble lengths

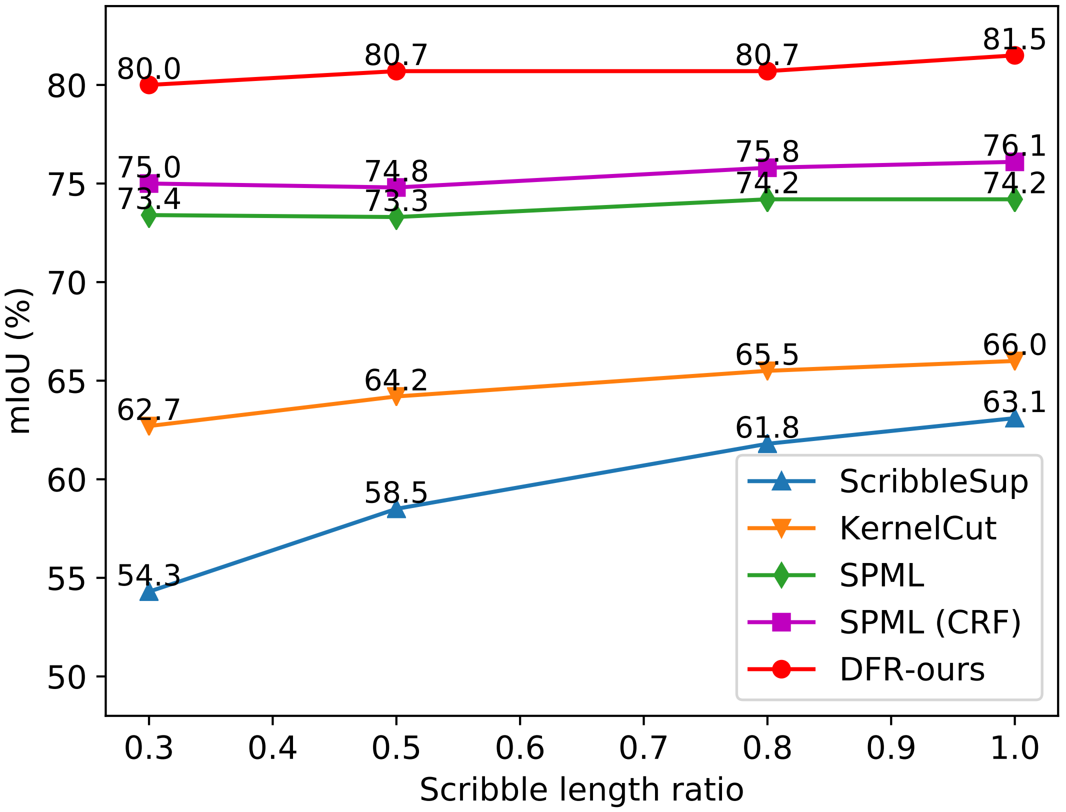

Fig. 4.3: Comparison with other state-of-the-art approaches on PASCAL VOC 2012 *val* set for different scribble lengths. "ration 1.0" means that we use the original length provided by the dataset. A smaller ratio value means that we use fewer annotated pixels as supervision.

are reported. Our approach consistently outperforms other approaches using different scribble lengths. Even only provided with 30% of original scribble length, our approach still obtains an mIoU of 80.0%.

In Table 4.3, we compare our approach with others on the PASCAL CONTEXT dataset, it can be seen that our approach also achieves a new state-of-the-art performance, with an mIoU gain of 12.7%.

In Fig. 4.4, we show qualitative comparisons between our approach and the previous state-of-the-art approaches. It can be seen that our approach keeps more details with

Table 4.2: Per-class comparison between our approach and others on PASCAL VOC 2012 *val* set.

Method	bkg	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv	Mean
KernelCut [12]	-	86.2	37.3	85.5	69.4	77.8	91.7	85.1	91.2	38.8	85.1	55.5	85.6	85.8	81.7	84.1	61.4	84.3	43.1	81.4	74.2	75.0
BPG [42]	93.4	84.8	38.4	84.6	65.5	78.8	91.4	85.9	89.5	41.0	87.3	58.3	84.1	85.2	83.7	83.6	64.9	88.3	46.0	86.3	73.9	76.0
SPML [40]	-	89.0	38.4	86.0	72.6	77.9	90.0	83.9	91.0	40.0	88.3	57.7	87.7	82.8	79.1	86.5	57.1	87.4	50.5	81.2	76.9	76.1
DFR-ours (ss)	**95.0**	**90.8**	**39.0**	**89.8**	**76.4**	**82.9**	**93.8**	**87.3**	**94.9**	**49.4**	**92.7**	**66.2**	**90.9**	**89.9**	**86.8**	**87.8**	**71.8**	**90.4**	**64.0**	**92.4**	**79.4**	**81.5**

Table 4.3: Comparison with other state-of-the-art on PASCAL CONTEXT dataset.

Method	Pub.	Sup.	CRF	mIoU (%)
ScribbleSup [11]	CVPR'16	S	✓	36.1
RAWKS [105]	CVPR'17	S	✓	37.4
GraphNet [1]	ACMM'18	S	-	33.9
GraphNet+CRF [1]	ACMM'18	S	✓	40.2
DFR-ours (ss)	-	S	-	50.9
DFR-ours (ms)	-	S	-	**52.9**

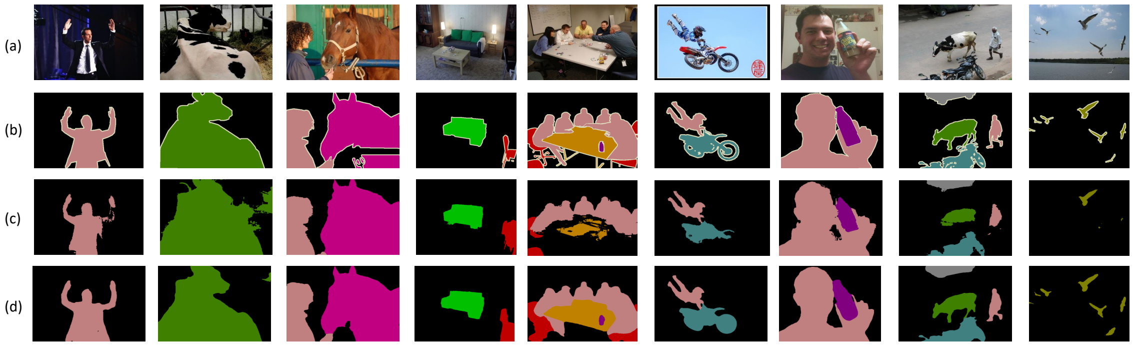

Fig. 4.4: Qualitative comparison between our method and other state-of-the-art approaches on PASCAL VOC 2012 *val* dataset. (a) Original image (b) Ground-truth (c) Results of A^2GNN [62] with dense CRF [60] as post-processing (d) Our results.

refined boundaries. Even for complicated cases, our approach still obtains accurate segmentation results.

4.3.4 Ablation Studies

In this section, we conduct our ablation studies on PASCAL VOC 2012 *val* dataset, and we use the single scale results.

Table. 4.6 reports the results on applying different elements to compute the regularized kernel. By adding the deep feature, the final performance increases to 81.5%, being 0.7% higher than only using static information (80.8%), which proves the effectiveness of the dynamic deep feature. It can also be found that RGB is an essential element, without it, the performance drops rapidly (from 80.8% to 72.8%). Nevertheless, it can also be found that adopting deep feature over spatial position can also improve the performance, with an mIoU increase of 2.1%. Note that when deep feature is not used in $\mathcal{L}_{\text{dfr}}$ (Eq. (4.6)), we

Table 4.4: Ablation study about the influence of the selected share feature for both semantic segmentation head and feature consistency head on PASCAL VOC 2012 *val* dataset. "Block" is shown in Fig. 4.2.

Feature				mIoU (%)
Block-1	Block-2	Block-3	Block-4	
✓				81.1
	✓			81.1
		✓		81.3
			✓	81.3
✓	✓			81.2
		✓	✓	80.9
✓	✓	✓		80.8
	✓	✓	✓	80.5
✓	✓	✓	✓	**81.5**

simply remove the full feature consistency head.

In Table 4.5, we evaluate the influence of the loss functions. It can be seen that without using any loss of the feature consistency head, our proposed regularized loss brings 12.1% mIoU increase (81.0% *v.s.* 68.9%). Using our feature consistency head further improves the final performance, with 0.5% mIoU growth. Besides, it can also be found that two loss functions of the feature consistency head are both useful to improve the final performance.

In Table. 4.7, we explore the influence of different supervision for the feature consistency head. It can be found that if only the ground truth scribble annotations are used as supervision, the performance is limited (only 80.6%) since the ground truth can only provide limited annotations (about 3% pixels are labeled), thus a local window will receive very few negative labels, which is insufficient for the feature distance loss $\mathcal{L}_{\text{fd}}$. Besides, using the confident prediction (M defined in Eq. (4.7)) from the semantic segmentation head performs better than using both ground truth and M, with an mIoU gain of 0.4% . This is because when the ground truth and M are merged, it is unavoidable to introduce some incorrect pixel relationship. Specifically, there are some noisy labels in M, while the labels in ground truth are all correct, thus it will lead to incorrect negative pixel pairs as supervision, which is harmful for training.

Table 4.5: Ablation study about the influence of the loss functions on PASCAL VOC 2012 *val* dataset.

Semantic Head		Feature Head		mIoU (%)
$\mathcal{L}_{\text{ce}}$	$\mathcal{L}_{\text{dfr}}$	$\mathcal{L}_{\text{fd}}$	$\mathcal{L}_{\text{fr}}$	
✓				68.9
✓	✓			81.0
✓	✓	✓		81.1
✓	✓	✓	✓	**81.5**

Table 4.6: Ablation study about the influence of the shallow feature and deep feature for our regularized loss (Eq. (4.6)) on PASCAL VOC 2012 *val* dataset. “XY” is the spatial position. “RGB” is the color information. “Feature” is the dynamic feature from the feature consistency head.

Kernel			mIoU (%)
XY	RGB	Feature	
✓			72.8
✓	✓		80.8
✓		✓	74.9
✓	✓	✓	**81.5**

It is interesting to notice that even the feature consistency head is not used, directly using deep feature in $\mathcal{L}_{\text{dfr}}$ can improve the performance (81.0% in Table. 4.5 *v.s.* 80.8% in Table. 4.6), which also proves the positive influence of the introduced deep feature.

Table. 4.4 shows the influence of the selected feature, which is a shared feature for both the semantic segmentation head and the feature consistency head. It can be seen that the obtained performance using the feature from each block individually is sightly limited. Finally, using all features together generates the best performance. Considering that the feature map from lower block contains more low-level information and the feature map from the higher block contains more high-level information, using all of these features can supply more comprehensive representations to build accurate relationship for different pixels.

Table 4.7: The influence of the supervision for the feature consistency head on PASCAL VOC 2012 *val* dataset. "GT" means the provided scribble annotation. "M" is our selected confident labels from the semantic segmentation head, defined in Eq. (4.7).

Supervision		mIoU (%)
GT	M	
✓		80.6
	✓	**81.5**
✓	✓	81.1

4.4 Conclusion

In this chapter, we have proposed a dynamic feature regularized loss for weakly supervised semantic segmentation with scribble annotation. Our regularized loss makes full use of the static shallow feature and dynamic deep feature to build the regularized kernel, which is more accurate to describe relationship of different pixels. Meanwhile, in order to provide more powerful deep features, we introduce vision transformer as the backbone and design a feature consistency head to restrict the pair-wise pixel relationship under the supervision of the prediction from the semantic segmentation head. We found that both our regularized loss and the feature consistency head can benefit from each other and lead to a better performance. Extensive experiments show that our approach achieves new state-of-the-art performances with large margins. In the future, we plan to apply our approach on other weakly supervised semantic segmentation tasks.

Chapter 5

Self-guided and Cross-guided Learning for Few-shot Segmentation

5.1 Motivation

As mentioned in Sect. 1.3, most approaches used masked GAP [27] or some more advanced methods such as FWB [28] to fuse all foreground or background features as a single vector, which unavoidably loses some useful information, using such a feature vector to guide the segmentation cannot make a precise prediction for pixels which need the lost information as support. Furthermore, for the multiple shot case, the common practice is to use the average of predictions from multiple individual support images as the final prediction [30] or the average of multiple support vectors as the final support vector [24]. However, the quality of different support images is different, using an average operation forces all support images to share the same contribution.

In order to overcome the aforementioned drawbacks, we propose a simple yet effective Self-Guided and Cross-Guided Learning approach (SCL). Specifically, we design a Self-Guided Module (SGM) to extract comprehensive support information from the support set. Through making an initial prediction for the annotated support image with the initial prototype, the covered and uncovered foreground regions are encoded to the primary and auxiliary support vectors using masked GAP, respectively. By aggregating both primary and auxiliary support vectors, better segmentation performances are obtained on query images.

Enlightened by our proposed SGM, we propose a Cross-Guided Module (CGM) for multiple shot segmentation, where we can evaluate prediction quality from each support image using other annotated support images, such that the high-quality support image will contribute more in the final fusion, and vice versa. Compared to other complicated approaches such as the attention mechanism [2, 73], our CGM does not need to re-train the model, and directly applying it during inference can improve the final performance.

In the following parts, we will introduce our approach in details.

This chapter includes our previous work "Self-Guided and Cross-Guided Learning for Few-Shot Segmentation", published in CVPR 2021 [77].

5.2 Problem Setting

The purpose of few-shot segmentation is to learn a segmentation model which can segment unseen objects provided with a few annotated images of the same class. We need to train a segmentation model on a dataset D_{train} and evaluate on a dataset D_{test}. Suppose the classes set in D_{train} is C_{train} and the classes set in D_{test} is C_{test}, there is no overlap between training set and test set, *i.e.*, $C_{\text{train}} \cap C_{\text{test}} = \varnothing$.

Following the previous definition in [125], episodes are applied to both training set D_{train} and test set D_{test} to set a K-shot segmentation task. Each episode is composed of a support set S and a query set Q for a specific class c. For one episode, the support set contains K images and their masks, *i.e.*, $S = \{(I_s^i, M_s^i)\}_{i=1}^{K}$, where I_s^i represents the *i*th image and M_s^i indicates its binary mask for the class c. A query set contains N images and their binary masks for the class c, *i.e.*, $Q = \left\{(I_q^i, M_q^i)\right\}_{i=1}^{N}$, where M_q^i is only used for training. For clear description, we use S_{train} and Q_{train} to represent the training support set and query set, while S_{test} and Q_{test} for the test set. A model is learned using the training support set S_{train} and query set Q_{train}. Then the model is evaluated on D_{test} using the test support set S_{test} and query set Q_{test}.

5.3 Methodology

5.3.1 Proposed Method

Fig. 5.1 shows our framework for 1-shot segmentation, which can be divided into the following steps:

1) Both support and query images are input to the same encoder to generate their feature maps. After that, an initial support vector is generated using masked GAP from all foreground pixels of the support image.

2) With the supervision of the support image mask, our SGM produces two new feature vectors including the primary and auxiliary support vectors, using the initial support vector and support feature map as input.

3) In this step, the primary and auxiliary support vectors are concatenated with the query feature map to guide the segmentation of query images. Through a query Feature

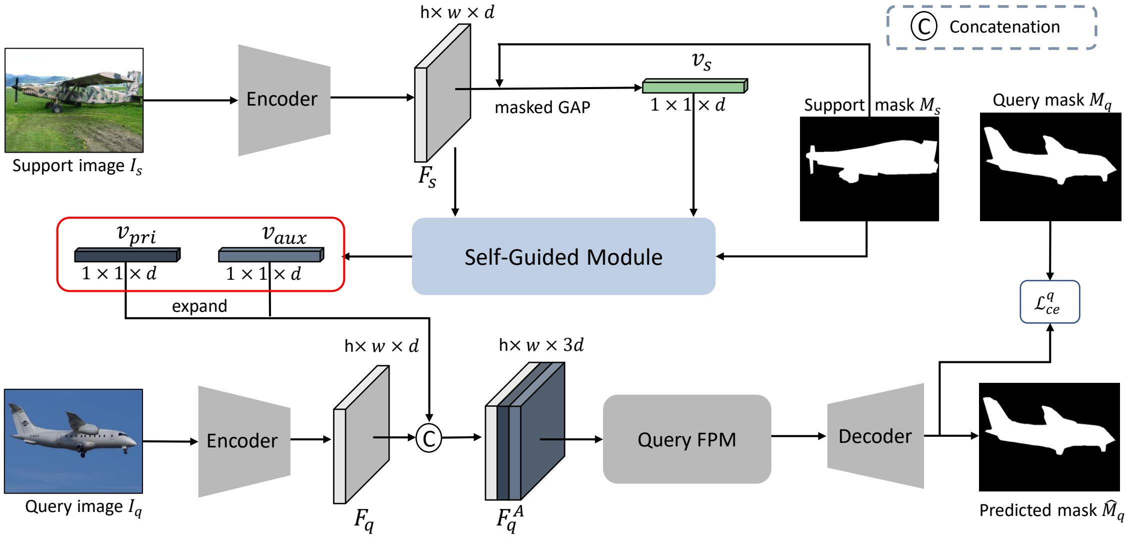

Fig. 5.1: The framework of our SCL approach for 1-shot segmentation. We firstly use an encoder to generate feature maps F_s and F_q from a support image and a query image, respectively. Then masked GAP is used to generate the initial support vector v_s. After that, our proposed self-guided module (SGM) takes v_s and F_s as input and output two new support vectors v_{pri} and v_{aux}, which are then used as the support information to segment the query image. Encoders for support and query images share the same weights.

Processing Module (FPM) and a decoder, the segmentation mask for the query image is generated. Note that all encoders and decoders are shared.

5.3.2 Self-Guided Learning on Support Set

Self-Guided module (SGM) is proposed to provide comprehensive support information to segment the query image. The details of our SGM can be found in Fig. 5.2.

Suppose the support image is I_s, after passing through the encoder, its feature maps is F_s. Then we use masked GAP to generate the initial support vector following previous approaches [2, 30, 126]:

$$v_s = \frac{\sum\limits_{i=1}^{hw} F_s(i) \cdot [M_s(i) = 1]}{\sum\limits_{i=1}^{hw} [M_s(i) = 1]}, \tag{5.1}$$

where i is the index of the spatial position. h and w are the height and width of the feature map, respectively. $[\cdot]$ is Iverson bracket, which equals to 1 if the inside condition is true, otherwise equals to 0. M_s is a binary mask and $M_s(i) = 1$ indicates the ith pixel belongs to class c. Note that M_s needs to be downsampled to the same height and width as F_s.

Both F_s and v_s are input to our proposed self-guided module (SGM). The initial feature vector v_s is firstly duplicated and expanded to the same size with F_s following [3, 2],

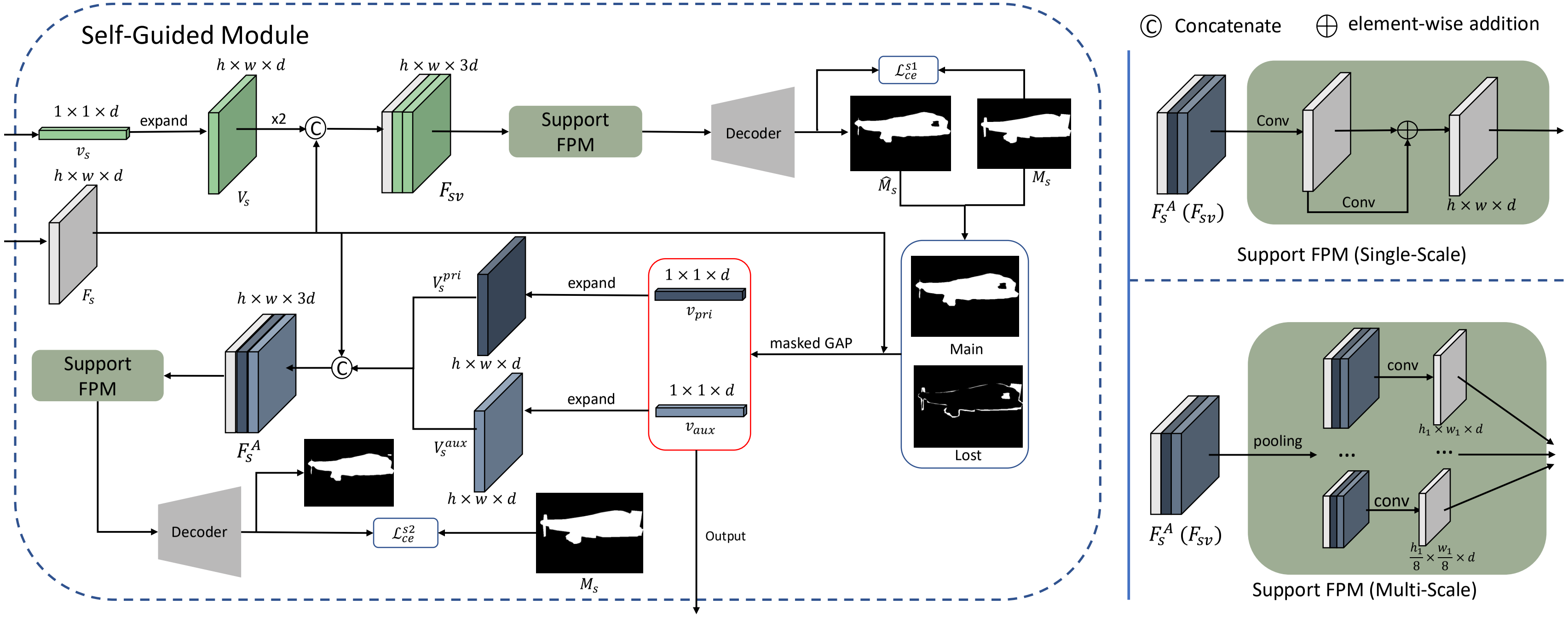

Fig. 5.2: The details of our proposed SGM. Our SGM uses the feature map F_s and the support vector v_s of the support image as input, and produces two new support vectors v_{pri} and v_{aux}. In order to provide high-quality support vectors, the support image mask is used as supervision. We provide two kinds of support Feature Processing Modules (FPM) to adapt to different decoders. All support FPMs share the same weights and all decoders are shared with the decoder in Fig 5.1.

represented as V_s, which is then concatenated with F_s to generate a new feature map:

$$F_{sv} = \mathit{Concat}([F_s, V_s, V_s]), \tag{5.2}$$

where $\mathit{Concat}(\cdot)$ is the concatenation operator.

Then, the probability map for the support image is generated after passing through the support FPM and the decoder:

$$P_{s1} = \mathit{softmax}(\mathcal{D}(\mathit{FPM}_s(F_{sv}))), \tag{5.3}$$

where P_{s1} is the predicted probability map, *i.e.*, $P_{s1} \in \mathbb{R}^{h \times w \times 2}$. $\mathcal{D}(\cdot)$ means the decoder and details can be found in Sec. 5.4.1. *softmax* is the softmax layer. $\mathit{FPM}_s(\cdot)$ is the support FPM, as shown in Fig. 5.2. According to the requirements of different decoders, we design two kinds of support FPMs: one for providing single-scale input to the decoder [2, 25] and the other one for providing multi-scale input to the decoder [3]. Note that we use a residual block in single-scale support FPM while only use convolution operator in multi-scale support FPM. This is for keeping consistent with the architecture of the corresponding query FPM. For example, in single-scale approach such as [2, 25], they adopted the residual block in query FPM while in multi-scale approach [3], only convolution operator is adopted in the query FPM.

Then the predicted mask is generated from P_{s1}:

$$\hat{M}_s = \mathrm{argmax}(P_{s1}), \tag{5.4}$$

where $\hat{M}_s$ is a binary mask, in which element 0 is the background and 1 is the indicator for being class c.

Using the predicted mask $\hat{M}_s$ and the ground-truth mask M_s, we can generate the primary support vector v_{pri} and the auxiliary support vector v_{aux}:

$$v_{pri} = \frac{\sum_{i=1}^{hw} F_s(i) \cdot [M_s(i) = 1] \cdot [\hat{M}_s(i) = 1]}{\sum_{i=1}^{hw} [M_s(i) = 1] \cdot [\hat{M}_s(i) = 1]}, \tag{5.5}$$

$$v_{aux} = \frac{\sum_{i=1}^{hw} F_s(i) \cdot [M_s(i) = 1] \cdot [\hat{M}_s(i) \neq 1]}{\sum_{i=1}^{hw} [M_s(i) = 1] \cdot [\hat{M}_s(i) \neq 1]}. \tag{5.6}$$

In Eq. (5.5), $[M_s(i) = 1] \cdot [\hat{M}_s(i) = 1]$ indicates the correctly predicted foreground mask using the initial support vector v_s as support. In Eq. (5.6), $[M_s(i) = 1] \cdot [\hat{M}_s(i) \neq 1]$ indicates the missing foreground mask. From Eq. (5.5) and Eq. (5.6), it can be found that

v_{pri} keeps the main support information as it focuses on aggregating correctly predicted information, v_{aux} focuses on collecting the lost critical information which cannot be predicted using v_s. Fig. 5.3 shows more examples about the masks to produce v_{pri} and v_{aux}. It can be seen that v_{pri} ignores some useful information unavoidably while v_{aux} collect all the lost information in v_{pri}.

In order to guarantee v_{pri} can collect most information from the support feature map, a cross-entropy loss is used on P_{s1} predicted in Eq. (5.3) :

$$\mathcal{L}_{ce}^{s1} = -\frac{1}{hw}\sum_{i=1}^{hw}\sum_{c_j \in \{0,1\}}[M_s(i) = c_j]log(P_{s1}^{c_j}(i)), \tag{5.7}$$

where 0 is the background class and 1 is the indicator for a specific foreground class c. $P_{s1}^{c_j}(i)$ denotes the predicted probability belonging to class c_j for pixel i.

Then we duplicate and expand v_{pri} and v_{aux} to the same height and width with F_s, represented as V_{pri}^s and V_{aux}^s, respectively. Following previous process, F_s, V_s^{pri} and V_s^{aux} are concatenated to generate a new feature map F_s^A:

$$F_s^A = \mathit{Concat}(\big[F_s, V_s^{pri}, V_s^{aux}\big]). \tag{5.8}$$

After that, the predicted probability map P_{s2} is generated based on the new feature map F_s^A:

$$P_{s2} = \mathit{softmax}(\mathcal{D}(\mathit{FPM}_s(F_s^A))). \tag{5.9}$$

Similar with Eq. (5.7), we use a cross-entropy loss to ensure aggregating v_{pri} and v_{aux} together can produce accurate segmentation mask on the support image:

$$\mathcal{L}_{ce}^{s2} = -\frac{1}{hw}\sum_{i=1}^{hw}\sum_{c_j \in \{0,1\}}[M_s(i) = c_j]log(P_{s2}^{c_j}(i)). \tag{5.10}$$

We only use foreground pixels to produce support vectors since background is more complicated than the foreground. Therefore, we cannot guarantee the support vector from background is far away from that of the foreground.

5.3.3 Training on Query Set

Using our proposed SGM, we generate the primary support vector v_{pri} and auxiliary support vector v_{aux}, where v_{pri} contains the primary information of support image and v_{aux} collects the lost information in v_{pri}.

Using the same encoder with I_s, we also generate the query feature map F_q, then v_{pri} and v_{aux} are duplicated and expanded to the same height and width as F_q, both of which are then concatenated with F_q to generate a new feature map:

$$F_q^A = \mathit{Concat}(\big[F_q, V_q^{pri}, V_q^{aux}\big]), \tag{5.11}$$

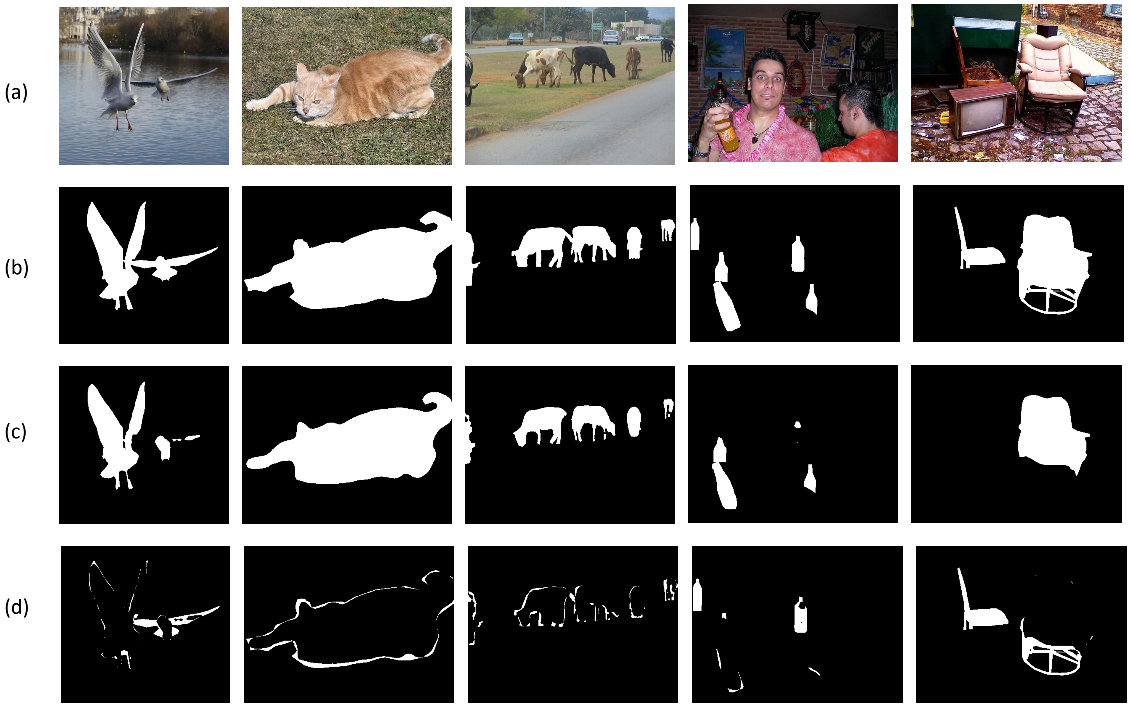

Fig. 5.3: Visualization of the masks for generating v_{pri} and v_{aux}. (a) original images. (b) ground-truth (masks for generating v_s). (c) masks for generating v_{pri}. (d) masks for generating v_{aux}. In most cases, v_{pri} aggregates the main information of the support image and v_{aux} mainly collects edge information. In some special cases (the last two columns), v_{pri} loses some body information and v_{aux} encodes all the lost information.

where F_q is the feature map of query image I_q, which is generated using the same encoder with the support image I_s. V_q^{pri} and V_q^{aux} correspond to expanded results of v_{pri} and v_{aux}, respectively.

Then F_q^A is input to a query FPM followed by a decoder to obtain the final prediction:

$$P_q = softmax(\mathcal{D}(FPM_q(F_q^A))), \tag{5.12}$$

where $FPM_q(\cdot)$ is the query FPM. P_q is the predicted probability map. (More details about the query FPM and decoder can be found in Sec. 5.4.1.)

We use a cross-entropy loss to supervise the segmentation of the query image:

$$\mathcal{L}_{ce}^{q} = -\frac{1}{hw}\sum_{i=1}^{hw}\sum_{c_j\in\{0,1\}}[M_q(i) = c_j]log(P_q^{c_j}(i)), \tag{5.13}$$

where $P_q^{c_j}(i)$ denotes the predicted probability belonging to class c_j for pixel i.

The overall training loss is defined as:

$$\mathcal{L} = \mathcal{L}_{ce}^{s1} + \mathcal{L}_{ce}^{s2} + \mathcal{L}_{ce}^{q}, \tag{5.14}$$

where $\mathcal{L}_{ce}^{s1}$, $\mathcal{L}_{ce}^{s2}$ are the loss functions defined by Eq.(5.7) and Eq.(5.10) in Sec. 5.3.2.

5.3.4 Cross-Guided Multiple Shot Learning

Enlightened by our SGM for 1-shot segmentation, we extend it to Cross-Guided Module (CGM) for the K-shot ($K > 1$) segmentation task. Among the K support images, each annotated support image can guide the query image segmentation individually. Based on this principle, we design our CGM where the final mask is fused using predictions from multiple annotated samples with high-quality support images contributing more and vice versa.

For K-shot segmentation task, there are K support images in one episode, *i.e.*, the support set $S = \{(I_s^1, M_s^1), (I_s^2, M_s^2), ..., (I_s^K, M_s^K)\}$. For the *k*th support image I_s^k, we can firstly use it as the support image and all K support images as query images to input to our proposed 1-shot segmentation model $\mathcal{G}$. The predicted mask for the *i*th support image I_s^i is:

$$\hat{M}_s^{i|k} = \text{argmax}(\mathcal{G}(I_s^i|I_s^k)), \tag{5.15}$$

where $\hat{M}_s^{i|k}$ is the predicted mask of I_s^i under the support of I_s^k. $\mathcal{G}(I_s^i|I_s^k)$ outputs the predicted score map of I_s^i using I_s^k as the support image and I_s^i as the query image.

The ground-truth mask M_s^i for image I_s^i is available. Thus, we can evaluate the confident score of I_s^k based on the IOU between the predicted masks and their ground-truth masks:

$$U_s^k = \frac{1}{K}\sum_{i=1}^{K} \text{IOU}(\hat{M}_s^{i|k}, M_s^i), \tag{5.16}$$

where $\text{IOU}(\cdot,\cdot)$ is used to compute the intersection over union score. Then the final predicted score map for an given query image I_q is:

$$\hat{P}_q = \textit{softmax}(\frac{1}{K}\sum_{k=1}^{K} U_s^k \mathcal{G}(I_q|I_s^k)). \tag{5.17}$$

A support image with a larger U_s^k makes more contribution to the final prediction, and the generated support vector is more likely to provide sufficient information to segment query images, and vice versa.

Using CGM does not need to re-train a new model, and we can directly use the segmentation model from 1-shot task to make predictions. Thus, CGM can improve the performance during inference without re-training.

5.4 Experiments

5.4.1 Implementation Details

Our SCL approach can be easily integrated into many existing few-shot segmentation approaches, and the effectiveness of our approach is evaluated using two baselines: CANet [2]

Table 5.1: Comparison with other state-of-the-arts using mIoU (%) as evaluation metric on Pascal-5^i for 1-shot and 5-shot segmentation. “P.” means Pascal. “*ours-SCL* (CANet)” and “*ours-SCL* (PFENet)” means CANet [2] and PFENet [3] are applied as baselines, respectively.

Method	Backbone	1-shot					5-shot				
		P.-5^0	P.-5^1	P.-5^2	P.-5^3	Mean	P.-5^0	P.-5^1	P.-5^2	P.-5^3	Mean
OSLSM (BMVC’17) [125]	vgg16	33.6	55.3	40.9	33.5	40.8	35.9	58.1	42.7	39.1	44.0
SG-One [30]	vgg16	40.2	58.4	48.4	38.4	46.3	41.9	58.6	48.6	39.4	47.1
PANet (ICCV’19) [24]	vgg16	42.3	58.0	51.1	41.2	48.1	51.8	64.6	59.8	46.5	55.7
PGNet (ICCV’19) [73]	resnet50	56.0	66.9	50.6	50.4	56.0	57.7	68.7	52.9	54.6	58.5
CRNet (CVPR’20) [29]	resnet50	-	-	-	-	55.7	-	-	-	-	58.8
RPMMs (ECCV’20) [127]	resnet50	55.2	65.9	52.6	50.7	56.3	56.3	67.3	54.5	51.0	57.3
FWB (ICCV’19) [28]	resnet101	51.3	64.5	56.7	52.2	56.2	54.8	67.4	62.2	55.3	59.9
PPNet*(ECCV’20) [23]	resnet50	47.8	58.8	53.8	45.6	51.5	58.4	67.8	**64.9**	56.7	62.0
DAN (ECCV’20) [128]	resnet101	54.7	68.6	**57.8**	51.6	58.2	57.9	69.0	60.1	54.9	60.5
CANet (CVPR’19) [2]	resnet50	52.5	65.9	51.3	51.9	55.4	55.5	67.8	51.9	53.2	57.1
PFENet (TPAMI’20) [3]	resnet50	61.7	69.5	55.4	56.3	60.8	63.1	70.7	55.8	57.9	61.9
ours-SCL (CANet)	resnet50	56.8	67.3	53.5	52.5	57.5	59.5	68.5	54.9	53.7	59.2
ours-SCL (PFENet)	resnet50	**63.0**	**70.0**	56.5	**57.7**	**61.8**	**64.5**	**70.9**	57.3	**58.7**	**62.9**

* We report the performance without extra unlabeled support data.

and PFENet [3], both of which use masked GAP to generate one support vector for a support image. All decoders in our SGM share the same weights with the decoder in the baseline.

We use single-scale support FPM in our SGM when using CANet [2] as the baseline since its decoder adopted single-scale architecture. Besides, the query FPM in CANet [2] used the probability map $P_{q(t-1)}$ from the previous iteration in the cache to refine the prediction. Fig. 5.4 shows details of the query FPM and decoder in CANet [2].

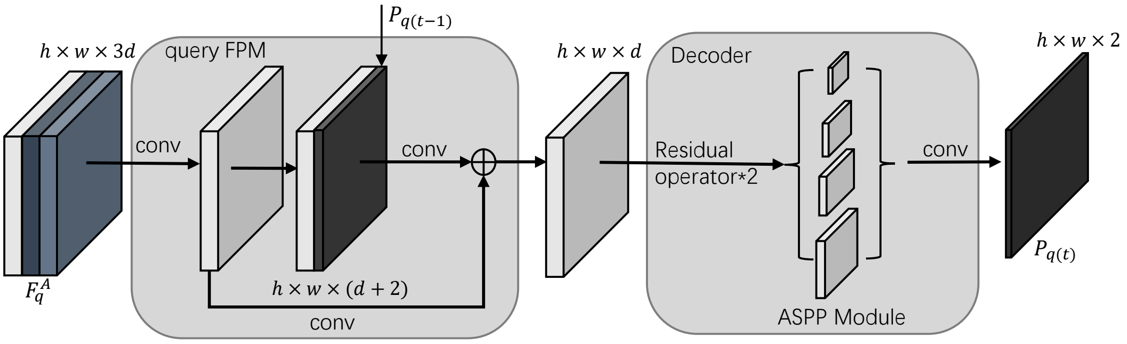

Fig. 5.4: Architecture of the query FPM and decoder in CANet [2]. CANet used the predicted probability map $P_{q(t-1)}$ from the previous iteration in its query FPM, and its decoder adopts single-scale residual layers following an ASPP module [6].

We use multi-scale support FPM in our SGM when using PFENet [3] as the baseline since its decoder adopted a multi-scale architecture. Additionally, the query FPM in PFENet [3] used a prior mask from the pre-trained model on ImageNet [129] as extra support, as shown in Fig. 5.5. Note that none of $P_{q(t-1)}$ or the prior mask is used in the support FPM in our SGM.

All training settings are the same as that in CANet [2] or PFENet [3]. The channel size d in Fig. 5.1 and Fig. 5.2 is set to 256. The batch size is 4 with 200 epochs used. The learning rate is 2.5×10^{-4} and weight decay is 5×10^{-4} if CANet [2] is the baseline. The learning rate is 2.5×10^{-3} and weight decay is 1×10^{-4} if PFENet [3] is the baseline.

During inference for the 1-shot task, we follow the same settings as in CANet [2] or PFENet [3]. For 5-shot segmentation, we directly use the segmentation model trained on 1-shot task. Following [24], we average the results from 5 runs with different random seeds as the final performance. All experiments are run on Nvidia RTX 2080Ti.

5.4.2 Dataset and Evaluation Metric

We evaluate our approach on PASCAL-5^i and COCO-20^i dataset. PASCAL-5^i is proposed in OSLSM [125], which is built based on PASCAL VOC 2012 [82] and SBD

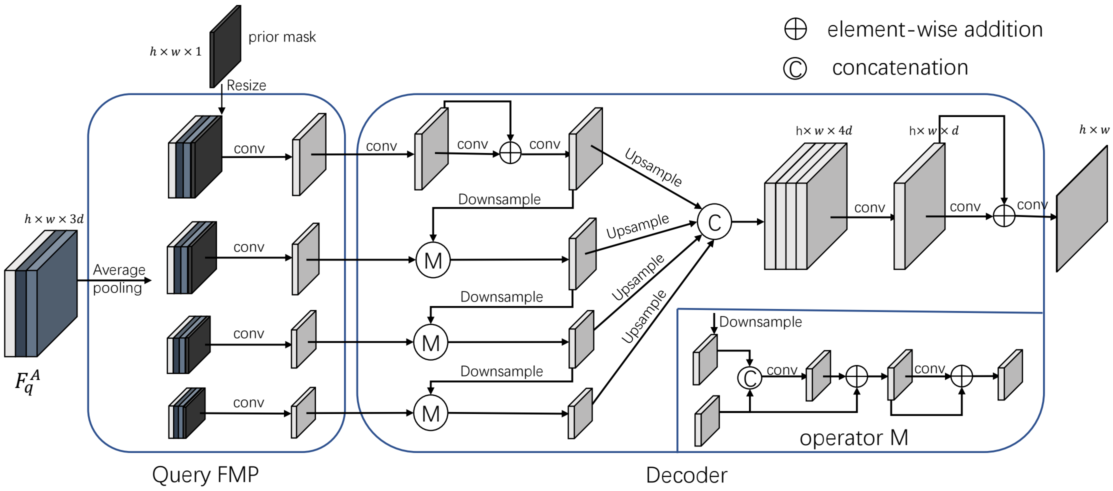

Fig. 5.5: Architecture of the multi-scale query Feature Processing Module (FPM) and decoder in PFENet [3]. PFENet [3] used a prior mask which is generated from the pre-trained model on ImageNet [129] in its query FPM. The height (the width shares the same size) of the feature map after the average pooling is set as $\{60, 30, 15, 8\}$.

dataset [112]. COCO-20^i is proposed in FWB [28], which is built based on MS-COCO [110] dataset.

In PASCAL-5^i, 20 classes are divided into 4 splits, in which 3 splits for training and 1 for evaluation. During evaluation, 1000 support-query pairs are randomly sampled from the evaluation set. For more details, please refer to OSLSM [125]. In COCO-20^i , the only difference with PASCAL-5^i is that it divides 80 classes to 4 splits. For more details, please refer to FWB [28]. For PASCAL-5^i, we evaluate our approach using both CANet [2] and PFENet [3] as baselines. For COCO-20^i, we evaluate our approach based on PFENet [3].

Following [24], mean intersection-over-union (mIoU) and foreground-background intersection-over-union (FB-IoU) are used as evaluation metrics.

5.4.3 Comparisons with State-of-the-art

In Table 5.1, we compare our approach with other state-of-the-art approaches on PASCAL-5^i. It can be seen that our approach achieves new state-of-the-art performances on both 1-shot and 5-shot tasks. Additionally, our approach significantly improves the performances of two baselines on 1-shot segmentation task, with mIoU increases of 2.1% and 1.0% for CANet [2] and PFENet [3], respectively. For the 5-shot segmentation task, our approach achieves 59.2% and 62.9% mIoU using CANet [2] and PFENet [3], respectively, both of which are direct improvement without re-training the model.

In Table 5.2 and Table 5.3, we compare our approach with others on the COCO-20^i dataset. Our approach outperforms other approaches by a large margin, with mIoU gain

Table 5.2: Comparison with other state-of-the-arts using mIoU (%) as evaluation metric on COCO-20^i for 1-shot. "C." means COCO-20. "*ours-SCL* (PFENet)" means PFENet [3] is applied as the baseline.

Method	Backbone	1-shot				
		C.^{0}	C.^{1}	C.^{2}	C.^{3}	Mean
FWB (ICCV'19) [28]	resnet101	19.9	18.0	21.0	28.9	21.2
PPNet (ECCV'20) [23]	resnet50	28.1	30.8	29.5	27.7	29.0
DAN (ECCV'20) [128]	resnet101	-	-	-	-	24.4
PFENet (TPAMI'20) [3]	resnet101	34.3	33.0	32.3	30.1	32.4
ours-SCL (PFENet)	resnet101	**36.4**	**38.6**	**37.5**	**35.4**	**37.0**

of 4.6% and 1.4% for 1-shot and 5-shot tasks, respectively.

Table 5.4 shows FB-IoU results between ours and other state-of-the-art methods on COCO-20^i. It can be seen that using our approach achieves new state-of-the-art performance. Compared to the baseline, our approach obtain a large gain of 3.3% and 1.8% FB-IoU for 1-shot and 5-shot segmentation, respectively.

In Table 5.5, we make a comparison between ours and other approaches on COCO (2017)-20^i. The difference between COCO (2017)-20^i and COCO-20^i is that COCO-20^i is built based on MS-COCO 2014 [110] while COCO (2017)-20^i is built based on MS-COCO 2017. It can be seen that compared to the state-of-the-art method RPMMs [127], our approach obtains a mIoU gain of 1.9% and 0.3% for 1-shot and 5-shot segmentation, respectively. Note that RPMMs [127] also adopted CANet [2] as its baseline.

In Table 5.6, we show the influence of using the multi-scale method during inference, it can be seen that using our method can improve the performance with or without using multi-scale. Besides, it can also be found that the improvement is more obvious for single scale inference.

In Fig. 5.6, we report more qualitative results using CANet [2] as the baseline on Pascal-5^i. It can be seen that our approach produces integral segmentation masks covering object details.

In Fig. 5.7, we report some qualitative results generated by our approach using PFENet [3] as the baseline. It can be seen that our approach produces integral segmentation masks covering object details. More experimental and qualitative results can be found in our supplement material.

In Fig. 5.8, we report the 5-shot results using PFENet [3] as the baseline on Pascal-5^i.

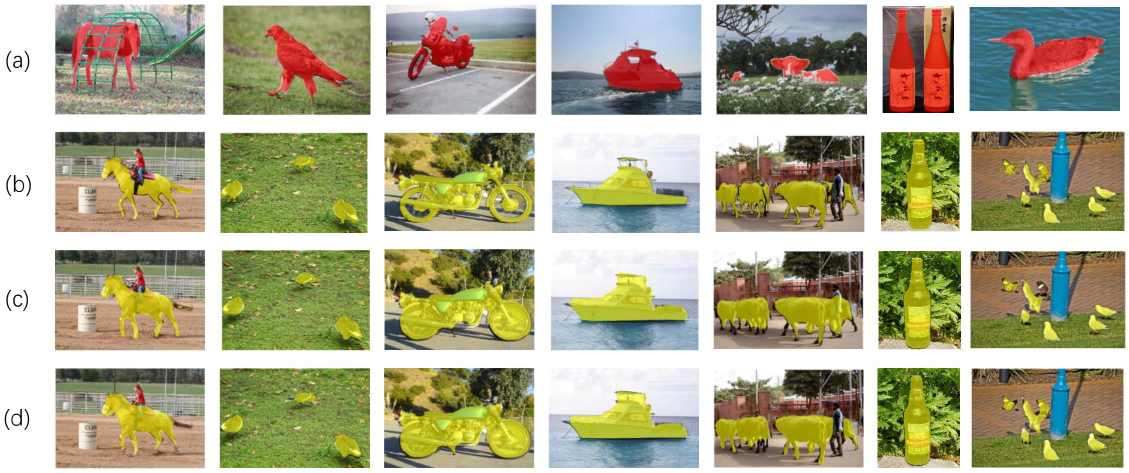

Fig. 5.6: Qualitative results of our proposed approach using CANet [2] as the baseline on Pascal-5^i. (a) Support images for the 1-shot task and their masks. (b) Query images and their ground-truth. (c) *ours-SCL* (CANet) 1-shot results. (d) *ours-SCL* (CANet) 5-shot results.

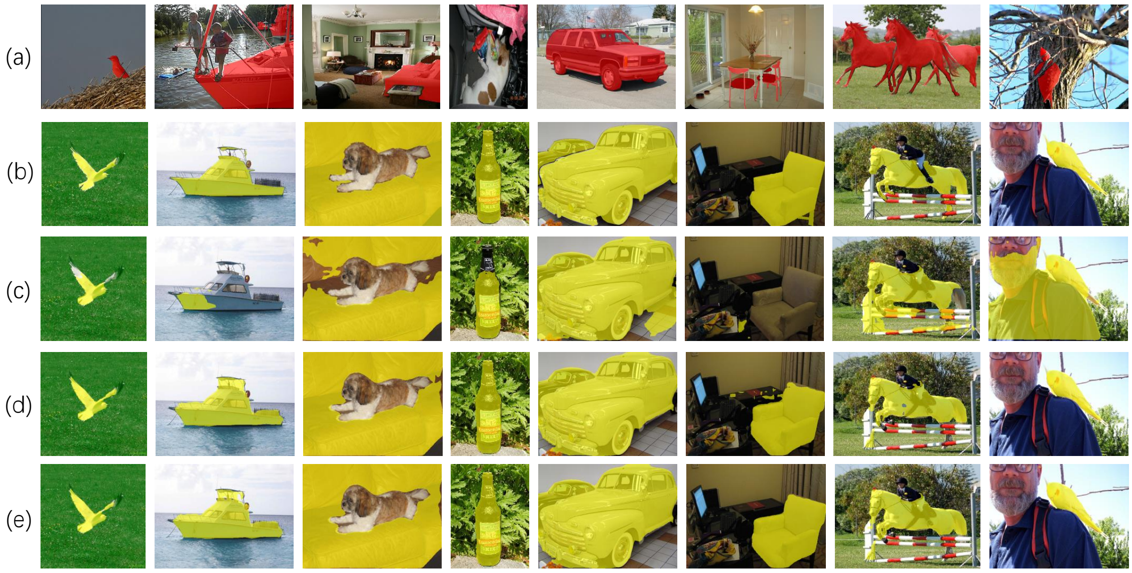

Fig. 5.7: Qualitative results of our approach on Pascal-5^i. (a) Support images for the 1-shot task and their masks. (b) Query images and their ground-truth. (c) PFENet [3] 1-shot results. (d) *Ours-SCL* (PFENet) 1-shot results. (e) *Ours-SCL* (PFENet) 5-shot results.

Table 5.3: Comparison with other state-of-the-arts using mIoU (%) as evaluation metric on COCO-20^i 5-shot segmentation. "C." means COCO-20. "*ours-SCL* (PFENet)" means PFENet [3] is applied as the baseline.

Method	Backbone	5-shot				
		C.^{0}	C.^{1}	C.^{2}	C.^{3}	Mean
FWB (ICCV'19) [28]	resnet101	19.1	21.5	23.9	30.1	23.7
PPNet (ECCV'20) [23]	resnet50	**39.0**	**40.8**	37.1	37.3	38.5
DAN (ECCV'20) [128]	resnet101	-	-	-	-	29.6
PFENet (TPAMI'20) [3]	resnet101	38.5	38.6	38.2	34.3	37.4
ours-SCL (PFENet)	resnet101	38.9	40.5	**41.5**	**38.7**	**39.9**

Table 5.4: Comparison with other state-of-the-art methods using FB-IoU (%) on COCO-20^i for 1-shot and 5-shot segmentation.

Method	Backbone	FB-IoU (%)	
		1-shot	5-shot
PANet (ICCV'19) [24]	vgg16	59.2	63.5
A-MCG (AAAI'19) [68]	resnet101	52.0	54.7
PFENet (TPAMI'20) [3]	resnet101	58.6	61.9
ours-SCL (PFENet)	resnet101	**61.9**	**63.7**

It can be seen that our approach produces integral segmentation masks covering object details.

In Fig. 5.9, we report more qualitative results using PFENet [3] as the baseline on COCO-20^i. It can be seen that our approach retains integral object details for both large and small objects.

5.4.4 Ablation Study

In this section, we conduct ablation studies on PASCAL-5^i using CANet [2] as the baseline and all results are average mIoU across 4 splits.

We firstly conduct an ablation study to show the influence of our proposed SGM and CGM in Table 5.7. For 1-shot, compared with the baseline, using SGM improves the

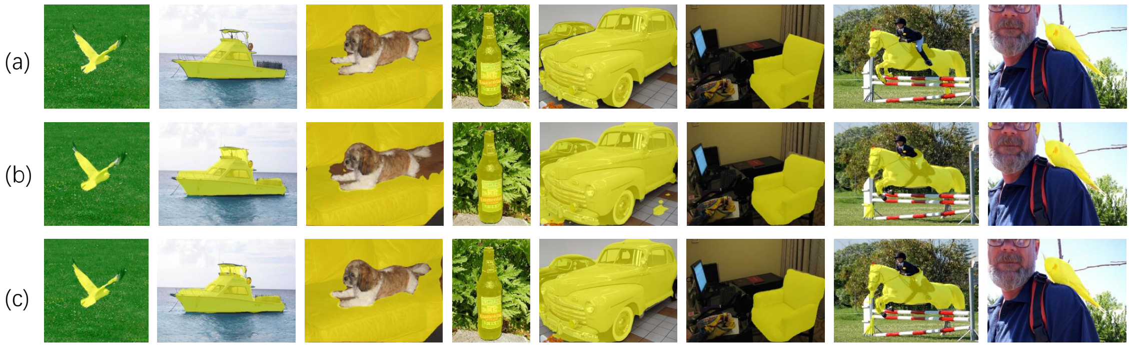

Fig. 5.8: Qualitative results of our proposed approach using PFENet [3] as the baseline on Pascal-5^i. (a) Query images and their ground-truth. (b) PFENet [3] 5-shot segmentation. (c) *ours-SCL* (PFENet) 5-shot results.

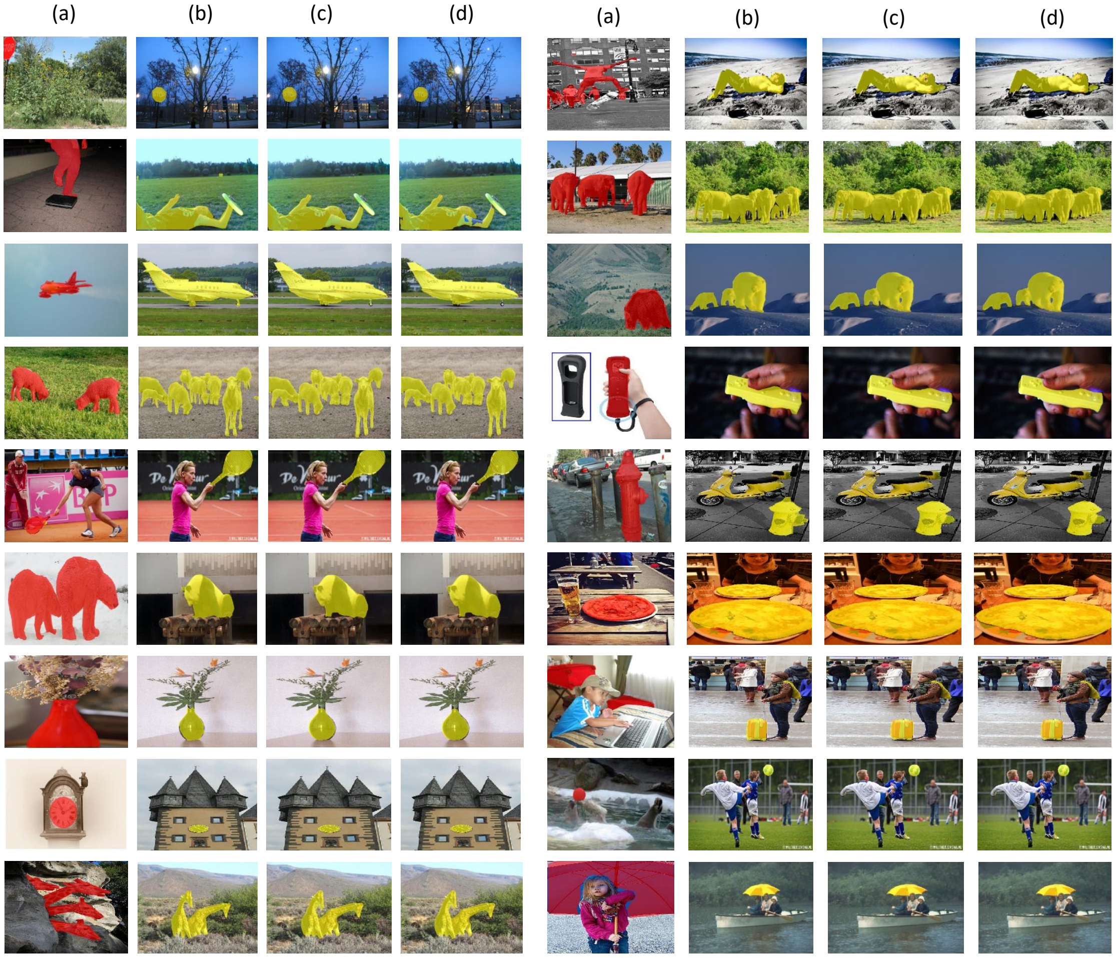

Fig. 5.9: Qualitative results of our proposed approach using PFENet [3] as the baseline on COCO-20^i. (a) Support images for the 1-shot task and their masks. (b) Query images and their ground-truth. (c) *ours-SCL* (PFENet) 1-shot results. (d) *ours-SCL* (PFENet) 5-shot results.

Table 5.5: Comparison with other state-of-the-art methods using mIoU (%) on COCO (2017)-20^i for 1-shot and 5-shot segmentation.

Method	Backbone	mIoU (%)	
		1-shot	5-shot
CANet (CVPR'19) [2]	resnet50	-	-
RPMMs (ECCV'20) [127]	resnet50	30.6	35.5
ours-SCL (CANet)	resnet50	**32.5**	**35.8**

Table 5.6: Comparison with the baseline (CANet [2]) about multi-scale inference on Pascal-5^i. MS: multi-scale inference.

Method	MS	mIoU (%)	
		1-shot	5-shot
CANet (CVPR'19) [2]	-	54.0	55.8
CANet (CVPR'19) [2]	✓	55.4	57.1
ours-SCL (CANet)	-	56.3	58.2
ours-SCL (CANet)	✓	**57.5**	**59.2**

performance by a large margin, being 2.1% and 4.1% for mIoU and FB-IoU, respectively. For 5-shot, using both SGM and CGM together obtains a 59.2% mIoU score, which is 3.3% higher compared to the baseline with the average method. Compared with the average method, our CGM directly increases the mIoU score by 0.5% when SGM is adopted. It is worth to notice that our CGM does not need to re-train the model and the gain is obtained in the inference stage.

Table 5.9: Ablation study of loss functions in the SGM on PASCAL-5^i for 1-shot segmentation. $\mathcal{L}_{ce}^{s1}$ means the loss function in Eq. (5.7). $\mathcal{L}_{ce}^{s2}$ means the loss function in Eq. (5.10).

$\mathcal{L}_{ce}^{s1}$	$\mathcal{L}_{ce}^{s2}$	mIoU (%)	FB-IoU (%)
✓		55.6	67.3
	✓	56.8	69.6
✓	✓	**57.5**	**70.3**

Table 5.7: Ablation study of our proposed SGM and CGM on PASCAL-5^i for both 1-shot and 5-shot segmentation. "Avg." means we use the average score of predictions from multiple support images. "base." means the baseline, which only uses the initial support vector without $\mathcal{L}_{ce}^{s1}$.

shot	base.	SGM	Avg.	CGM	mIoU	FB-IoU
1	✓		-	-	55.4	66.2
1	✓	✓	-	-	**57.5**	**70.3**
5	✓		✓		55.9	66.7
5	✓			✓	56.9	69.7
5	✓	✓	✓		58.7	70.3
5	✓	✓		✓	**59.2**	**70.7**

Table 5.8 shows the influence of the support vectors on the proposed SGM for 1-shot segmentation. If only v_s is adopted, the mIoU and FB-IoU scores are 55.6% and 67.3% respectively. Using SGM (with both v_{pri} and v_{aux}) achieves 57.5% and 70.3% on mIoU and FB-IoU, with a significant gain of 1.9% and 3.0% on mIoU and FB-IoU, respectively. Besides, It can also be seen that when using v_{pri} and v_{aux} individually, it only achieves 56.6% and 51.4% on mIoU, both of which are much lower than using them jointly. Solely using v_{aux} even performs worse than the baseline (only using v_s). Furthermore, we also evaluate the performance when using all support vectors (v_s, v_{pri} and v_{aux}) together, it can be seen that it does not improve the results, which also proves that v_{pri} and v_{aux} already provide sufficient information as support, demonstrating the effectiveness of our SGM. Note that when using all support vectors, channels of F_q^A should be increased to $4d$.

Table 5.9 studies the influence of loss functions $\mathcal{L}_{ce}^{s1}$ and $\mathcal{L}_{ce}^{s2}$ in SGM. Using both $\mathcal{L}_{ce}^{s1}$ and $\mathcal{L}_{ce}^{s2}$ significantly outperforms the baseline. If only $\mathcal{L}_{ce}^{s1}$ is adopted without $\mathcal{L}_{ce}^{s2}$, the obtained mIoU score is 55.6%, being 1.9% lower than using both loss functions together. This is because $\mathcal{L}_{ce}^{s2}$ provides one more step of training by treating the support image as query image, where both support vectors v_{pri} and v_{aux} are deployed. Similarly, if only $\mathcal{L}_{ce}^{s2}$ is adopted without $\mathcal{L}_{ce}^{s1}$, the obtained performance is also lower than using both loss functions together. This is because using $\mathcal{L}_{ce}^{s1}$ can ensure primary support vector v_{pri} focus on extracting the main information while v_{aux} focus on the lost information. Without $\mathcal{L}_{ce}^{s1}$, the roles of v_{pri} and v_{aux} get mixed and vague.

Table 5.8: Ablation study of the support vectors in our proposed SGM on PASCAL-5^i for 1-shot segmentation. v_s, v_{pri} and v_{aux} are initial, primary and auxiliary feature vectors generated by our SGM, respectively. Note that $\mathcal{L}_{ce}^{s1}$ is used for v_s.

v_s	v_{pri}	v_{aux}	mIoU (%)	FB-IoU (%)
✓			55.6	67.3
	✓		56.6	69.5
		✓	51.4	65.2
✓	✓	✓	57.1	69.9
	✓	✓	**57.5**	**70.3**

5.5 Conclusion

We propose a self-guided learning approach for few-shot segmentation. Our approach enables to extract comprehensive support information using our proposed self-guided module. Besides, in order to improve the drawbacks of average fusion for multiple support images, we propose a new cross-guided module to make highly quality support images contribute more in the final prediction, and vice versa. Extensive experiments show the effectiveness of our proposed modules. In the future, we will try to use the background information as extra support to improve our approach.

Chapter 6

Conclusions

In this chapter, the final summary of this thesis will be presented, followed by some future works in relevant research directions.

6.1 Summary

Semantic segmentation has been boosting a lot with the rapidly development of deep neural network, to improve the limitation that current fully supervised semantic segmentation heavily rely on massive accurate pixel-level annotation and increase the generalization ability of the model, in this thesis, we focus on weakly supervised semantic segmentation and few-shot segmentation, in which the weakly supervised semantic segmentation concentrates on providing accurate pixel-level prediction utilizing weak annotation as supervision, while few-shot aims to segment unseen categories during inference given a few support samples.

For weakly supervised semantic segmentation with image label as annotation, we proposed the Reliable Region Mining model, an end-to-end network for image-level weakly supervised semantic segmentation. We revisited drawbacks of the state-of-the-art methods, which adopt the two-step approach. We proposed a one-step approach through mining tiny reliable regions and used them as ground-truth labels directly for our segmentation branch training. With limited pixels as supervision, we designed a dense energy loss and a batch-based class distance loss, which consider shallow features (RGB colors and spatial information) and high-level feature, respectively. The two new losses cooperate with the pixel-wise cross-entropy loss to optimize the training process. Furthermore, we design a new feature attention module to extract global information, which also proves to be effective for the final prediction. Based on our one-step approach, we extended a two-step method. Both our one-step and two-step approaches achieve the state-of-the-art performance. More importantly, our approach offers a different perspective from the traditional two-step solutions. We believe that the proposed one-step approach could further boost research in this direction.

For weakly supervised semantic segmentation with bounding-box as annotation, we have proposed a new system, Affinity Attention Graph Neural Network, 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. Extensive experiments show the great potential that our approach can be regarded as an unified framework to handle different weak supervisions.

For weakly supervised semantic segmentation with scribble as annotation, we have proposed a dynamic feature regularized loss for weakly supervised semantic segmentation with scribble annotation. Our regularized loss makes full use of the static shallow feature and dynamic deep feature to build the regularized kernel, which is more accurate to describe relationship of different pixels. Meanwhile, in order to provide more powerful deep features, we introduce vision transformer as the backbone and design a feature consistency head to restrict the pair-wise pixel relationship under the supervision of the prediction from the semantic segmentation head. We found that both our regularized loss and the feature consistency head can benefit from each other and lead to a better performance. Extensive experiments show that our approach achieves new state-of-the-art performances with large margins. In the future, we plan to apply our approach on other weakly supervised semantic segmentation tasks.

For few-shot segmentation, we firstly observe that it is unavoidable to lose some useful critical information using the average operation to obtain the support vector. Then we propose a self-guided learning approach to mitigate this issue. Our approach enables to extract comprehensive support information using our proposed self-guided module. Besides, in order to improve the drawbacks of average fusion for multiple support images, we propose a new cross-guided module to make highly quality support images contribute more in the final prediction, and vice versa. Extensive experiments show the effectiveness of our proposed modules.

6.2 Future works

At the time of concluding this manuscript, several exciting perspectives can be proposed to further continue the work done in this thesis.

Vision transformer architecture for weakly supervised semantic segmentation. Recently, inspired by the success of vision transformer [63] for image classification, some new vision transformer architectures [64, 130] were introduced for fully supervised semantic segmentation, which led to clear performance improvement. However, there is few researches attempting to design or apply such architecture to weakly supervised semantic segmentation. More importantly, since one of the most advantages for vision transformer is that it can build the global relationship for each pixel and produce the attention matrix in each attention layer, we need to study how to utilize these information for weakly supervised semantic segmentation. Besides, the difference between CNN and vision transformer should also be analyzed. Finally, designing a end-to-end architecture based on vision transformer is also import research area.

Utilizing background information for few-hot segmentation. Current State-of-the-art approaches mainly encode the foreground information as prototypes. In the future, we will attempt to encode background information to support the segmentation. There are two main challenges: the first one is that background usually is complicated, encoding it as single prototype is not suitable. One possible solution is to encode the background information as several prototypes. The second challenge is that some background contains the classes which is used during inference, once such information is encoded as background, it will do harmful for the model. So it is important that only the real background information should be encoded. To achieve, we will try to use prior mask based on pre-trained model or probability model.

Appendix: A list of Publications

Here is the list of my research publications during my Ph.D. study:

1. Bingfeng Zhang, Jimin Xiao and Yao Zhao. “Dynamic Feature Regularized Loss for Weakly Supervised Semantic Segmentation”, arXiv preprint arXiv:2108.01296 (Under Review) [76].

2. Bingfeng Zhang, Jimin Xiao, Yunchao Wei, Kaizhu Huang, Shan Luo and Yao Zhao. “End-to-End Weakly Supervised Semantic Segmentation with Reliable Region Mining.”, Under Review [75].

3. Bingfeng Zhang, Jimin Xiao, Jianbo Jiao, Yunchao Wei, and Yao Zhao. “Affinity Attention Graph Neural Network for Weakly Supervised Semantic Segmentation.” IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI), 2021. [62].

4. Bingfeng Zhang, Jimin Xiao, and Terry Qin. “Self-Guided and Cross-Guided Learning for Few-Shot Segmentation.” In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2021. [77].

5. Bingfeng Zhang, Jimin Xiao, Yunchao Wei, Mingjie Sun, and Kaizhu Huang. “Reliability does matter: An End-to-end Weakly Supervised Semantic Segmentation Approach.” In Proceedings of the AAAI Conference on Artificial Intelligenc, 2020. [74].

6. Siyue Yu, Bingfeng Zhang, Jimin Xiao, and Eng Gee Lim. “Structure-consistent weakly supervised salient object detection with local saliency coherence.” In Proceedings of the AAAI Conference on Artificial Intelligence. 2021. [131].

7. Yu, Siyue, Jimin Xiao, Bingfeng Zhang, Eng Gee Lim, and Yao Zhao. “Fast pixel-matching for video object segmentation.” Signal Processing: Image Communication, 2021. [132].

8. Mingjie Sun, Jimin Xiao, Eng Gee Lim, Bingfeng Zhang, and Yao Zhao. “Fast template matching and update for video object tracking and segmentation.” In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020. [133].

Reference

[1] Mengyang Pu, Yaping Huang, Qingji Guan, and Qi Zou. Graphnet: Learning image pseudo annotations for weakly-supervised semantic segmentation. In *Proceedings of the 26th ACM International Conference on Multimedia*, pages 483–491, 2018.

[2] Chi Zhang, Guosheng Lin, Fayao Liu, Rui Yao, and Chunhua Shen. Canet: Class-agnostic segmentation networks with iterative refinement and attentive few-shot learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5217–5226, 2019.

[3] Zhuotao Tian, Hengshuang Zhao, Michelle Shu, Zhicheng Yang, Ruiyu Li, and Jiaya Jia. Prior guided feature enrichment network for few-shot segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.

[4] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3431–3440, 2015.

[5] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Semantic image segmentation with deep convolutional nets and fully connected crfs. *arXiv preprint arXiv:1412.7062*, 2014.

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

[7] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. Pyramid scene parsing network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2881–2890, 2017.

[8] Jyh-Jing Hwang, Stella X Yu, Jianbo Shi, Maxwell D Collins, Tien-Ju Yang, Xiao Zhang, and Liang-Chieh Chen. Segsort: Segmentation by discriminative sorting

of segments. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 7334–7344, 2019.

[9] Lin Song, Yanwei Li, Zeming Li, Gang Yu, Hongbin Sun, Jian Sun, and Nanning Zheng. Learnable tree filter for structure-preserving feature transform. In *Advances in Neural Information Processing Systems*, pages 1711–1721, 2019.

[10] Meng Tang, Abdelaziz Djelouah, Federico Perazzi, Yuri Boykov, and Christopher Schroers. Normalized cut loss for weakly-supervised cnn segmentation. In *CVPR*, pages 1818–1827, 2018.

[11] Di Lin, Jifeng Dai, Jiaya Jia, Kaiming He, and Jian Sun. Scribblesup: Scribble-supervised convolutional networks for semantic segmentation. In *CVPR*, pages 3159–3167, 2016.

[12] Meng Tang, Federico Perazzi, Abdelaziz Djelouah, Ismail Ben Ayed, Christopher Schroers, and Yuri Boykov. On regularized losses for weakly-supervised cnn segmentation. In *ECCV*, pages 507–522, 2018.

[13] Jifeng Dai, Kaiming He, and Jian Sun. Boxsup: Exploiting bounding boxes to supervise convolutional networks for semantic segmentation. In *CVPR*, pages 1635–1643, 2015.

[14] Anna Khoreva, Rodrigo Benenson, Jan Hosang, Matthias Hein, and Bernt Schiele. Simple does it: Weakly supervised instance and semantic segmentation. In *CVPR*, pages 876–885, 2017.

[15] Chunfeng Song, Yan Huang, Wanli Ouyang, and Liang Wang. Box-driven class-wise region masking and filling rate guided loss for weakly supervised semantic segmentation. *arXiv preprint arXiv:1904.11693*, 2019.

[16] Cheng-Chun Hsu, Kuang-Jui Hsu, Chung-Chi Tsai, Yen-Yu Lin, and Yung-Yu Chuang. Weakly supervised instance segmentation using the bounding box tightness prior. In *Advances in Neural Information Processing Systems*, pages 6582–6593, 2019.

[17] Amy Bearman, Olga Russakovsky, Vittorio Ferrari, and Li Fei-Fei. What's the point: Semantic segmentation with point supervision. In *ECCV*, pages 549–565. Springer, 2016.

[18] Jiwoon Ahn and Suha Kwak. Learning pixel-level semantic affinity with image-level supervision for weakly supervised semantic segmentation. In *CVPR*, pages 4981–4990, 2018.

[19] Yunchao Wei, Xiaodan Liang, Yunpeng Chen, Xiaohui Shen, Ming-Ming Cheng, Jiashi Feng, Yao Zhao, and Shuicheng Yan. Stc: A simple to complex framework for weakly-supervised semantic segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(11):2314–2320, 2016.

[20] Qibin Hou, PengTao Jiang, Yunchao Wei, and Ming-Ming Cheng. Self-erasing network for integral object attention. In *NIPS*, pages 549–559, 2018.

[21] Jiwoon Ahn, Sunghyun Cho, and Suha Kwak. Weakly supervised learning of instance segmentation with inter-pixel relations. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2209–2218, 2019.

[22] Nanqing Dong and Eric P Xing. Few-shot semantic segmentation with prototype learning. In *Proceedings of the British Machine Vision Conference*, volume 3, 2018.

[23] Yongfei Liu, Xiangyi Zhang, Songyang Zhang, and Xuming He. Part-aware prototype network for few-shot semantic segmentation. *arXiv preprint arXiv:2007.06309*, 2020.

[24] Kaixin Wang, Jun Hao Liew, Yingtian Zou, Daquan Zhou, and Jiashi Feng. Panet: Few-shot image semantic segmentation with prototype alignment. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 9197–9206, 2019.

[25] Yuwei Yang, Fanman Meng, Hongliang Li, Qingbo Wu, Xiaolong Xu, and Shuai Chen. A new local transformation module for few-shot segmentation. In *International Conference on Multimedia Modeling*, pages 76–87, 2020.

[26] Pinzhuo Tian, Zhangkai Wu, Lei Qi, Lei Wang, Yinghuan Shi, and Yang Gao. Differentiable meta-learning model for few-shot semantic segmentation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 12087–12094, 2020.

[27] Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. Learning deep features for discriminative localization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2921–2929, 2016.

[28] Khoi Nguyen and Sinisa Todorovic. Feature weighting and boosting for few-shot segmentation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 622–631, 2019.

[29] Weide Liu, Chi Zhang, Guosheng Lin, and Fayao Liu. Crnet: Cross-reference networks for few-shot segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4165–4173, 2020.

[30] Xiaolin Zhang, Yunchao Wei, Yi Yang, and Thomas S Huang. Sg-one: Similarity guidance network for one-shot semantic segmentation. *IEEE Transactions on Cybernetics*, 2020.

[31] Yunchao Wei, Jiashi Feng, Xiaodan Liang, Ming-Ming Cheng, Yao Zhao, and Shuicheng Yan. Object region mining with adversarial erasing: A simple classification to semantic segmentation approach. In *CVPR*, pages 1568–1576, 2017.

[32] Yunchao Wei, Huaxin Xiao, Honghui Shi, Zequn Jie, Jiashi Feng, and Thomas S Huang. Revisiting dilated convolution: A simple approach for weakly-and semi-supervised semantic segmentation. In *CVPR*, pages 7268–7277, 2018.

[33] Xiang Wang, Shaodi You, Xi Li, and Huimin Ma. Weakly-supervised semantic segmentation by iteratively mining common object features. In *CVPR*, pages 1354–1362, 2018.

[34] Zilong Huang, Xinggang Wang, Jiasi Wang, Wenyu Liu, and Jingdong Wang. Weakly-supervised semantic segmentation network with deep seeded region growing. In *CVPR*, pages 7014–7023, 2018.

[35] Jordi Pont-Tuset, Pablo Arbelaez, Jonathan T Barron, Ferran Marques, and Jitendra Malik. Multiscale combinatorial grouping for image segmentation and object proposal generation. *IEEE transactions on Pattern Analysis and Machine Intelligence*, 39(1):128–140, 2016.

[36] Carsten Rother, Vladimir Kolmogorov, and Andrew Blake. Grabcut: Interactive foreground extraction using iterated graph cuts. In *ACM Transactions on Graphics (TOG)*, volume 23, pages 309–314, 2004.

[37] Viveka Kulharia, Siddhartha Chandra, Amit Agrawal, Philip Torr, and Ambrish Tyagi. Box2seg: Attention weighted loss and discriminative feature learning for weakly supervised segmentation. In *Proceedings of the European Conference on Computer Vision*, pages 290–308. Springer, 2020.

[38] Radhakrishna Achanta, Appu Shaji, Kevin Smith, Aurelien Lucchi, Pascal Fua, and Sabine Süsstrunk. Slic superpixels compared to state-of-the-art superpixel methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(11):2274–2282, 2012.

[39] Anton Obukhov, Stamatios Georgoulis, Dengxin Dai, and Luc Van Gool. Gated crf loss for weakly supervised semantic image segmentation. *arXiv preprint arXiv:1906.04651*, 2019.

[40] Tsung-Wei Ke, Jyh-Jing Hwang, and Stella X Yu. Universal weakly supervised segmentation by pixel-to-segment contrastive learning. *arXiv preprint arXiv:2105.00957*, 2021.

[41] Pablo Arbelaez, Michael Maire, Charless Fowlkes, and Jitendra Malik. Contour detection and hierarchical image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2010.

[42] Bin Wang, Guojun Qi, Sheng Tang, Tianzhu Zhang, Yunchao Wei, Linghui Li, and Yongdong Zhang. Boundary perception guidance: a scribble-supervised semantic segmentation approach. In *International Joint Conference on Artificial Intelligence*, 2019.

[43] Saining Xie and Zhuowen Tu. Holistically-nested edge detection. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1395–1403, 2015.

[44] G Papandreou, L-Ch Chen, K Murphy, and AL Yuille. Weakly-and semi-supervised learning of a dcnn for semantic image segmentation. *http://arxiv.org/abs/1502*, 2734, 2015.

[45] Qibin Hou, Ming-Ming Cheng, Xiaowei Hu, Ali Borji, Zhuowen Tu, and Philip HS Torr. Deeply supervised salient object detection with short connections. In *CVPR*, pages 3203–3212, 2017.

[46] Jianming Zhang, Sarah Adel Bargal, Zhe Lin, Jonathan Brandt, Xiaohui Shen, and Stan Sclaroff. Top-down neural attention by excitation backprop. *IJCV*, 126(10):1084–1102, 2018.

[47] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(4):834–848, 2018.

[48] Pedro O Pinheiro and Ronan Collobert. From image-level to pixel-level labeling with convolutional networks. In *CVPR*, pages 1713–1721, 2015.

[49] Huaizu Jiang, Jingdong Wang, Zejian Yuan, Yang Wu, Nanning Zheng, and Shipeng Li. Salient object detection: A discriminative regional feature integration approach. In *CVPR*, pages 2083–2090, 2013.

[50] Junsong Fan, Zhaoxiang Zhang, Chunfeng Song, and Tieniu Tan. Learning integral objects with intra-class discriminator for weakly-supervised semantic segmentation. In *CVPR*, pages 4283–4292, 2020.

[51] Yun Liu, Yu-Huan Wu, Peisong Wen, Yujun Shi, Yu Qiu, and Ming-Ming Cheng. Leveraging instance-, image- and dataset-level information for weakly supervised instance segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.

[52] Seungho Lee, Minhyun Lee, Jongwuk Lee, and Hyunjung Shim. Railroad is not a train: Saliency as pseudo-pixel supervision for weakly supervised semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5495–5505, 2021.

[53] Jungbeom Lee, Eunji Kim, and Sungroh Yoon. Anti-adversarially manipulated attributions for weakly and semi-supervised semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4071–4080, 2021.

[54] Jungbeom Lee, Jooyoung Choi, Jisoo Mok, and Sungroh Yoon. Reducing information bottleneck for weakly supervised semantic segmentation. *arXiv preprint arXiv:2110.06530*, 2021.

[55] Jungbeom Lee, Eunji Kim, Sungmin Lee, Jangho Lee, and Sungroh Yoon. Ficklenet: Weakly and semi-supervised semantic image segmentation using stochastic inference. *arXiv preprint arXiv:1902.10421*, 2019.

[56] Wataru Shimoda and Keiji Yanai. Self-supervised difference detection for weakly-supervised semantic segmentation. In *ICCV*, pages 5208–5217, 2019.

[57] Yude Wang, Jie Zhang, Meina Kan, Shiguang Shan, and Xilin Chen. Self-supervised equivariant attention mechanism for weakly supervised semantic segmentation. *arXiv preprint arXiv:2004.04581*, 2020.

[58] Pablo Arbeláez, Jordi Pont-Tuset, Jonathan T Barron, Ferran Marques, and Jitendra Malik. Multiscale combinatorial grouping. In *CVPR*, pages 328–335, 2014.

[59] David Martin, Charless Fowlkes, Doron Tal, and Jitendra 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.*, volume 2, pages 416–423. IEEE, 2001.

[60] Philipp Krähenbühl and Vladlen Koltun. Parameter learning and convergent inference for dense random fields. In *International Conference on Machine Learning*, pages 513–521, 2013.

[61] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

[62] Bingfeng Zhang, Jimin Xiao, Jianbo Jiao, Yunchao Wei, and Yao Zhao. Affinity attention graph neural network for weakly supervised semantic segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.

[63] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[64] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. *arXiv preprint arXiv:2103.14030*, 2021.

[65] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *Proceedings of the European Conference on Computer Vision*, pages 801–818, 2018.

[66] Zilong Huang, Xinggang Wang, Lichao Huang, Chang Huang, Yunchao Wei, and Wenyu Liu. Ccnet: Criss-cross attention for semantic segmentation. In *Proceedings of the IEEE International Conference on Computer Vision*, 2019.

[67] Xiangtai Li, Xia Li, Li Zhang, Guangliang Cheng, Jianping Shi, Zhouchen Lin, Shaohua Tan, and Yunhai Tong. Improving semantic segmentation via decoupled body and edge supervision. *arXiv preprint arXiv:2007.10035*, 2020.

[68] Tao Hu, Pengwan Yang, Chiliang Zhang, Gang Yu, Yadong Mu, and Cees GM Snoek. Attention-based multi-context guiding for few-shot semantic segmentation.

In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 8441–8448, 2019.

[69] Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Daan Wierstra, et al. Matching networks for one shot learning. In *Advances in Neural Information Processing Systems*, pages 3630–3638, 2016.

[70] Flood Sung, Yongxin Yang, Li Zhang, Tao Xiang, Philip HS Torr, and Timothy M Hospedales. Learning to compare: Relation network for few-shot learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1199–1208, 2018.

[71] Reza Azad, Abdur R Fayjie, Claude Kauffman, Ismail Ben Ayed, Marco Pedersoli, and Jose Dolz. On the texture bias for few-shot cnn segmentation. *arXiv preprint arXiv:2003.04052*, 2020.

[72] Shuo Lei, Xuchao Zhang, Jianfeng He, Fanglan Chen, and Chang-Tien Lu. Few-shot semantic segmentation augmented with image-level weak annotations. *arXiv preprint arXiv:2007.01496*, 2020.

[73] Chi Zhang, Guosheng Lin, Fayao Liu, Jiushuang Guo, Qingyao Wu, and Rui Yao. Pyramid graph networks with connection attentions for region-based one-shot semantic segmentation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 9587–9595, 2019.

[74] Bingfeng Zhang, Jimin Xiao, Yunchao Wei, Mingjie Sun, and Kaizhu Huang. Reliability does matter: An end-to-end weakly supervised semantic segmentation approach. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 12765–12772, 2020.

[75] Bingfeng Zhang, Jimin Xiao, Yunchao Wei, Shan Luo, Kaizhu Huang, and Zhao Yao. End-to-end weakly supervised semantic segmentation with reliable region mining. *Pattern Recognition*, 2022.

[76] Bingfeng Zhang, Jimin Xiao, and Yao Zhao. Dynamic feature regularized loss for weakly supervised semantic segmentation. *arXiv preprint arXiv:2108.01296*, 2021.

[77] Bingfeng Zhang, Jimin Xiao, and Terry Qin. Self-guided and cross-guided learning for few-shot segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8312–8321, 2021.

[78] Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7794–7803, 2018.

[79] Thomas Joy, Alban Desmaison, Thalaiyasingam Ajanthan, Rudy Bunel, Mathieu Salzmann, Pushmeet Kohli, Philip HS Torr, and M Pawan Kumar. Efficient relaxations for dense crfs with sparse higher-order potentials. *SIAM Journal on Imaging Sciences*, 12(1):287–318, 2019.

[80] Jason Yosinski, Jeff Clune, Anh Nguyen, Thomas Fuchs, and Hod Lipson. Understanding neural networks through deep visualization. *arXiv preprint arXiv:1506.06579*, 2015.

[81] Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *ECCV*, pages 818–833. Springer, 2014.

[82] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *International Journal of Computer Vision*, 88(2):303–338, 2010.

[83] Zifeng Wu, Chunhua Shen, and Anton Van Den Hengel. Wider or deeper: Revisiting the resnet model for visual recognition. *Pattern Recognition*, 90:119–133, 2019.

[84] Tie Liu, Zejian Yuan, Jian Sun, Jingdong Wang, Nanning Zheng, Xiaoou Tang, and Heung-Yeung Shum. Learning to detect a salient object. *IEEE Transactions on PAMI*, 33(2):353–367, 2010.

[85] Tianyi Zhang, Guosheng Lin, Weide Liu, Jianfei Cai, and Alex Kot. Splitting vs. merging: Mining object regions with discrepancy and intersection loss for weakly supervised semantic segmentation. In *ECCV*, 2020.

[86] Junsong Fan, Zhaoxiang Zhang, and Tieniu Tan. Employing multi-estimations for weakly-supervised semantic segmentation. In *ECCV*. Springer, 2020.

[87] Guolei Sun, Wenguan Wang, Jifeng Dai, and Luc Van Gool. Mining cross-image semantics for weakly supervised semantic segmentation. In *ECCV*, pages 347–365. Springer, 2020.

[88] Wenfeng Luo, Meng Yang, and Weishi Zheng. Weakly-supervised semantic segmentation with saliency and incremental supervision updating. *Pattern Recognition*, 115:107858, 2021.

[89] Peng-Tao Jiang, Ling-Hao Han, Qibin Hou, Ming-Ming Cheng, and Yunchao Wei. Online attention accumulation for weakly supervised semantic segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.

[90] Xueyi Li, Tianfei Zhou, Jianwu Li, Yi Zhou, and Zhaoxiang Zhang. Group-wise semantic mining for weakly supervised semantic segmentation. *arXiv preprint arXiv:2012.05007*, 2020.

[91] Jiwoon Ahn, Sunghyun Cho, and Suha Kwak. Weakly supervised learning of instance segmentation with inter-pixel relations. In *CVPR*, June 2019.

[92] Yu-Ting Chang, Qiaosong Wang, Wei-Chih Hung, Robinson Piramuthu, Yi-Hsuan Tsai, and Ming-Hsuan Yang. Weakly-supervised semantic segmentation via subcategory exploration. In *CVPR*, pages 8991–9000, 2020.

[93] Liyi Chen, Weiwei Wu, Chenchen Fu, Xiao Han, and Yuntao Zhang. Weakly supervised semantic segmentation with boundary exploration. In *ECCV*, pages 347–362. Springer, 2020.

[94] Dong Zhang, Hanwang Zhang, Jinhui Tang, Xiansheng Hua, and Qianru Sun. Causal intervention for weakly-supervised semantic segmentation. *arXiv preprint arXiv:2009.12547*, 2020.

[95] Kunyang Sun, Haoqing Shi, Zhengming Zhang, and Yongming Huang. Ecs-net: Improving weakly supervised semantic segmentation by using connections between class activation maps. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7283–7292, 2021.

[96] Hyeokjun Kweon, Sung-Hoon Yoon, Hyeonseong Kim, Daehee Park, and Kuk-Jin Yoon. Unlocking the potential of ordinary classifier: Class-specific adversarial erasing framework for weakly supervised semantic segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6994–7003, 2021.

[97] Bingfeng Zhang, Jimin Xiao, Yunchao Wei, Mingjie Sun, and Kaizhu Huang. Reliability does matter: an end-to-end weakly supervised semantic segmentation approach. *arXiv preprint arXiv:1911.08039*, 2019.

[98] Zhijiang Guo, Yan Zhang, and Wei Lu. Attention guided graph convolutional networks for relation extraction. *arXiv preprint arXiv:1906.07510*, 2019.

[99] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(4):834–848, 2017.

[100] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2961–2969, 2017.

[101] Chunfeng Song and Jun Xiao. Cian: Cross-image affinity net for weakly supervised semantic segmentation. 2020.

[102] Boxi Wu, Shuai Zhao, Wenqing Chu, Zheng Yang, and Deng Cai. Improving semantic segmentation via dilated affinity. *arXiv preprint arXiv:1907.07011*, 2019.

[103] Kiran K Thekumparampil, Chong Wang, Sewoong Oh, and Li-Jia Li. Attention-based graph neural network for semi-supervised learning. *arXiv preprint arXiv:1803.03735*, 2018.

[104] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. Pyramid scene parsing network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[105] Paul Vernaza and Manmohan Chandraker. Learning random-walk label propagation for weakly-supervised semantic segmentation. In *CVPR*, pages 7158–7166, 2017.

[106] Qizhu Li, Anurag Arnab, and Philip HS Torr. Weakly-and semi-supervised panoptic segmentation. In *Proceedings of the European Conference on Computer Vision*, pages 102–118, 2018.

[107] Tete Xiao, Yingcheng Liu, Bolei Zhou, Yuning Jiang, and Jian Sun. Unified perceptual parsing for scene understanding. In *Proceedings of the European Conference on Computer Vision*, pages 418–434, 2018.

[108] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

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

[110] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *Proceedings of the European Conference on Computer Vision*, pages 740–755, 2014.

[111] 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*, pages 991–998, 2011.

[112] Bharath Hariharan, Pablo Arbeláez, Ross Girshick, and Jitendra Malik. Simultaneous detection and segmentation. In *Proceedings of the European Conference on Computer Vision*, pages 297–312, 2014.

[113] Ruochen Fan, Qibin Hou, Ming-Ming Cheng, Gang Yu, Ralph R Martin, and Shi-Min Hu. Associating inter-image salient instances for weakly supervised semantic segmentation. In *Proceedings of the European Conference on Computer Vision*, pages 367–383, 2018.

[114] Yanzhao Zhou, Yi Zhu, Qixiang Ye, Qiang Qiu, and Jianbin Jiao. Weakly supervised instance segmentation using class peak response. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3791–3800, 2018.

[115] Guanbin Li, Yuan Xie, Liang Lin, and Yizhou Yu. Instance-level salient object segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2386–2395, 2017.

[116] Ruochen Fan, Ming-Ming Cheng, Qibin Hou, Tai-Jiang Mu, Jingdong Wang, and Shi-Min Hu. S4net: Single stage salient-instance segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6103–6112, 2019.

[117] Jifeng Dai, Kaiming He, and Jian Sun. Instance-aware semantic segmentation via multi-task network cascades. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3150–3158, 2016.

[118] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. *arXiv preprint arXiv:1710.10903*, 2017.

[119] Shang-Hua Gao, Ming-Ming Cheng, Kai Zhao, Xin-Yu Zhang, Ming-Hsuan Yang, and Philip Torr. Res2net: A new multi-scale backbone architecture. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.

[120] Jiang-Jiang Liu, Qibin Hou, Ming-Ming Cheng, Jiashi Feng, and Jianmin Jiang. A simple pooling-based design for real-time salient object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3917–3926, 2019.

[121] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2117–2125, 2017.

[122] Roozbeh Mottaghi, Xianjie Chen, Xiaobai Liu, Nam-Gyu Cho, Seong-Whan Lee, Sanja Fidler, Raquel Urtasun, and Alan Yuille. The role of context for object detection and semantic segmentation in the wild. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014.

[123] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, pages 248–255. Ieee, 2009.

[124] MMSegmentation Contributors. Mmsegmentation: Openmmlab semantic segmentation toolbox and benchmark, 2020.

[125] Amirreza Shaban, Shray Bansal, Zhen Liu, Irfan Essa, and Byron Boots. One-shot learning for semantic segmentation. *arXiv preprint arXiv:1709.03410*, 2017.

[126] Kai Zhu, Wei Zhai, Zheng-Jun Zha, and Yang Cao. Self-supervised tuning for few-shot segmentation. *arXiv preprint arXiv:2004.05538*, 2020.

[127] Boyu Yang, Chang Liu, Bohao Li, Jianbin Jiao, and Qixiang Ye. Prototype mixture models for few-shot semantic segmentation. *arXiv preprint arXiv:2008.03898*, 2020.

[128] Haochen Wang, Xudong Zhang, Yutao Hu, Yandan Yang, Xianbin Cao, and Xiantong Zhen. Few-shot semantic segmentation with democratic attention networks. In *Proceedings of the European Conference on Computer Vision*, 2020.

[129] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015.

[130] Wenhai Wang, Enze Xie, Xiang Li, Deng-Ping Fan, Kaitao Song, Ding Liang, Tong Lu, Ping Luo, and Ling Shao. Pyramid vision transformer: A versatile backbone for dense prediction without convolutions. *arXiv preprint arXiv:2102.12122*, 2021.

[131] Siyue Yu, Bingfeng Zhang, Jimin Xiao, and Eng Gee Lim. Structure-consistent weakly supervised salient object detection with local saliency coherence. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*. AAAI Palo Alto, CA, USA, 2021.

[132] Siyue Yu, Jimin Xiao, Bingfeng Zhang, Eng Gee Lim, and Yao Zhao. Fast pixel-matching for video object segmentation. *Signal Processing: Image Communication*, 98:116373, 2021.

[133] Mingjie Sun, Jimin Xiao, Eng Gee Lim, Bingfeng Zhang, and Yao Zhao. Fast template matching and update for video object tracking and segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10791–10799, 2020.