Scale-Aware Graph Neural Network for Few-Shot Semantic Segmentation

Guo-Sen Xie[1], Jie Liu[1], Huan Xiong[1,3*], Ling Shao[2]
[1]Mohamed bin Zayed University of AI, UAE [2]Inception Institute of Artificial Intelligence, UAE
[3]Harbin Institute of Technology, China

Abstract

Few-shot semantic segmentation (FSS) aims to segment unseen class objects given very few densely-annotated support images from the same class. Existing FSS methods find the query object by using support prototypes or by directly relying on heuristic multi-scale feature fusion. However, they fail to fully leverage the high-order appearance relationships between multi-scale features among the support-query image pairs, thus leading to an inaccurate localization of the query objects. To tackle the above challenge, we propose an end-to-end scale-aware graph neural network (SAGNN) by reasoning the cross-scale relations among the support-query images for FSS. Specifically, a scale-aware graph is first built by taking support-induced multi-scale query features as nodes and, meanwhile, each edge is modeled as the pairwise interaction of its connected nodes. By progressive message passing over this graph, SAGNN is capable of capturing cross-scale relations and overcoming object variations (e.g., appearance, scale and location), and can thus learn more precise node embeddings. This in turn enables it to predict more accurate foreground objects. Moreover, to make full use of the location relations across scales for the query image, a novel self-node collaboration mechanism is proposed to enrich the current node, which endows SAGNN the ability of perceiving different resolutions of the same objects. Extensive experiments on PASCAL-5^i and COCO-20^i show that SAGNN achieves state-of-the-art results.

1. Introduction

Deep convolutional neural networks [14, 29, 40] have advanced the development of many downstream vision tasks, such as semantic segmentation [1, 3, 21, 25, 49], which is a dense prediction task. To achieve an effective segmentation model, e.g. Deeplab [3], large amounts of pixel-wise image annotations are required for model training, which are costly and time consuming to acquire. Weakly supervised learning [23, 43] can alleviate the annotation costs to some extent; however, the model performance drops significantly under this scenario. Moreover, both fully and weakly supervised models suffer poor generalization to unseen domains with only a few densely-annotated training images available. As such, few-shot semantic segmentation (FSS) [27] is proposed for dealing with the unseen object segmentation and data annotation issue.

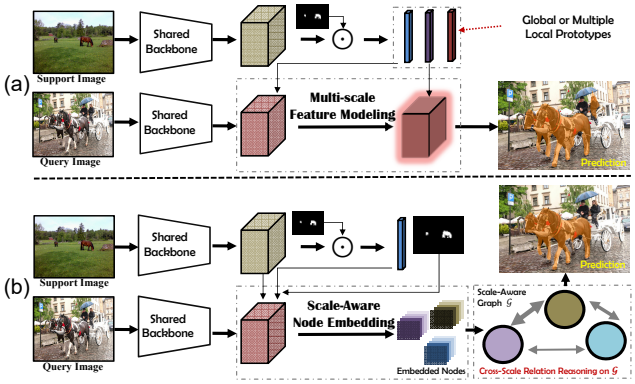

Figure 1. **Comparisons of SAGNN and existing FSS models. (a) Existing methods** usually first generate a global or multiple local prototypes by masked averaging/clustering over the foreground of the support image. Then these support prototypes are applied to query features (and/or their multi-scale counterparts) for the later prediction of the query mask. However, this paradigm segments the *human* region to *horse*. This happens because it is difficult to fully capture the object variations (e.g., the scale, appearance, and spatial location of *horse* in the support and query images are different from each other) by such limited prototypes. **(b) Our SAGNN** can well estimate the foreground *horse* in the query image. It benefits from our scale-aware node embedding and cross-scale relation reasoning on the scale-aware graph. Zoom in for details.

FSS aims to segment the foreground object of an unseen class in a query image by merely utilizing a few (one) annotated support images (image), where both the support/query images share a common unseen class. Typically, a base training set with annotations, which has different object categories from the unseen class set, can assist the learning of the FSS model. Inspired by few-shot classification and meta-learning [30], the segmentor can be trained by

*Corresponding author

episode-based meta-training – i.e., each episode (sub-task) is sampled from the base training set and consists of support and query images – to mimic the testing scheme on the unseen classes. During each sub-task, support images are used as guidance to segment the query image. The difference between the ground-truth mask and the predicted query mask can thus supervise the model training.

Specifically, most of the existing FSS models [5,7,22,28, 34] adopt a two-stream (support and query) metric-based network. The global [48] or multiple local [10, 20, 41] prototypes (Fig. 1(a)) for a class are first calculated from the support branch, based on the ground-truth mask of the support image. Then, these prototypes are applied to the query feature map (e.g., by cosine similarity) from the query branch, thus leading to a support-induced matching score map, which serves as an important clue for encoding the query mask. At first, since the above process is a few-to-many matching, it is difficult to fully capture the object scale/appearance/location variations by such limited prototypes (the *human* region is mis-segmented as *horse* in Fig. 1(a)). In addition, in the query branch, multi-scale features [19, 31], e.g., ASPP [4], have been widely-used for accurate foreground estimates. However, the high-order appearance relationships among them have still not been completely leveraged for better inferring query objects. Moreover, some works [33, 42, 46] propose a pixel-to-pixel matching paradigm among support/query images based on non-local attention variants [36], which still suffers from the object scale variations. For instance, if the *horse* region in the support image is very small, most locations of the segmented query *horse* will correspond to the same location from the support *horse* region.

To address the above challenges, we propose an end-to-end scale-aware graph neural network (SAGNN) (§3.3) to explicitly and comprehensively reason the high-order appearance relationships across the multi-scale support-query features for FSS. As in Fig. 1(b), since FSS is a structural prediction task, SAGNN models the object variations among support-query images in a structure-to-structure manner. Here, *structure* means 1) each node in SAGNN is a whole feature map (the embedded nodes in Fig. 1(b)) corresponding to a specific scale; and 2) the node features are updated in a holistic manner by aggregating neighborhood node features (cross-scale relation reasoning in Fig. 1(b)). In this way, SAGNN can preserve and transfer more structural context from the seen to the unseen domains. Specifically, we first build a scale-aware graph with each node representing a support-induced multi-scale query feature. Meanwhile, the edge in the graph is the pairwise interaction of its connected nodes. By message passing to encourage favorable information exchange among support-query pairs over this graph, SAGNN can capture higher-order cross-scale relationships and overcome harmful object variations (e.g., scale, appearance, and spatial location variations for the *horse* object in Fig. 1), and thus lead to a precise localization of query objects in a structural way. Furthermore, we propose a novel self-node collaboration mechanism (Fig. 3) for enriching the current node features during feature aggregation, which can make full use of the location relations across scales for the query image. To sum up, our main contributions are:

- We propose a scale-aware graph neural network (SAGNN) which explicitly reasons the cross-scale relationships among support-query images in a structure-to-structure manner, for tackling the FSS task. To the best of our knowledge, this is the first work to do this in the FSS domain.
- We propose a novel self-node collaboration mechanism for enriching the current node features during feature aggregation, which can bring to SAGNN a significant performance gain.
- We set new state-of-the-art results on two FSS benchmarks. Our SAGNN solves FSS from the perspective of structural modeling, which sheds light for future research in this field.

2. Related Work

Graph Neural Networks (GNNs). GNNs [11, 12, 26] are useful tools for modeling the structural relationships among various kinds of input data, e.g., they have been used for video understanding [35,37], biology structure prediction [11], graph-structured data classification [32], object detection [45], and zero-shot learning [38, 39], among others. The key components constituting a GNN are its nodes, edges, and the parameterized message passing functions among them, which are usually jointly updated in an end-to-end manner. Recently, the graph convolutional network (GCN) [8, 16] is proposed to endow CNNs with relation modeling power. The GCN, usually with its edge fixed as a similarity scalar, can be seen as a special case of GNN. In contrast, GNNs are more flexible to model various kinds of data, and both nodes and edges should be updated alternatively by the message passing procedure. Our SAGNN is a general GNN, with its own merits: 1) The nodes of SAGNN are cross-scale feature maps, thus the structural context (beneficial for FSS) is preserved automatically. 2) We make full use of the location relations across scales while aggregating the neighboring node features.

Semantic Segmentation. Semantic segmentation [21] is a dense prediction task, which assigns a class label to each pixel of a given image. Since the succesful application to the semantic segmentation of fully convolutional neural networks [21], various network designs have been made in this field, such as UNet [25], Deeplab [3], SegNet [1], and PSPNet [49]. In addition, dilated convolutions [44] have been proposed to enlarge the receptive field while

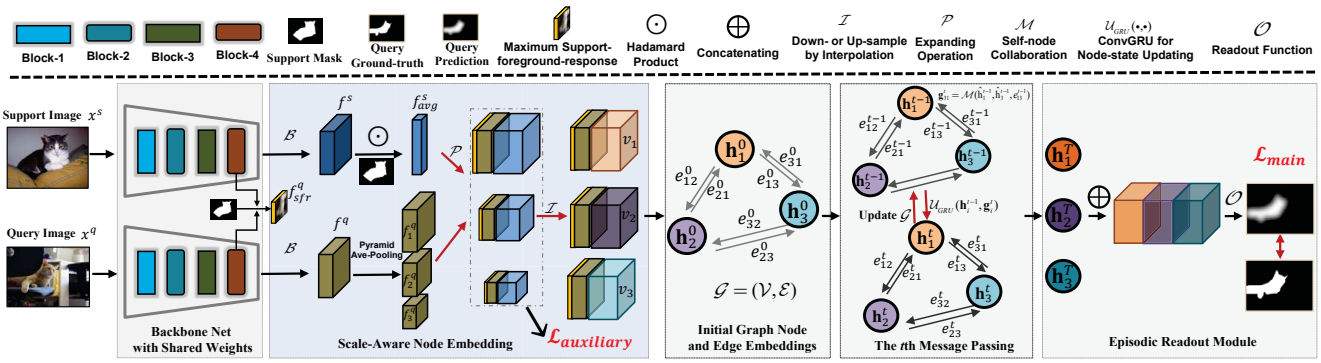

Figure 2. **SAGNN architecture (three nodes are used).** SAGNN (§3.3) takes an episode, i.e., the support-query image pair (x^s, x^q), as input. Based on the mid- and high-level backbone features, the support-induced multi-scale features are derived by the *Scale-Aware Node Embedding*. Then a scale-aware graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is built and initialized. Next, SAGNN conducts T times of *Message Passing* over $\mathcal{G}$ to capture high-order cross-scale relationships among these nodes, thus leading to updated node representations. Finally a concatenation and readout function are used to segment the query image. $\mathcal{L}_{\text{main}}$ and $\mathcal{L}_{\text{auxiliary}}$ (§3.4) are binary cross-entropy losses for SAGNN training.

maintaining the spatial resolution, which has become a widely-used strategy in modern semantic segmentation systems. Other representative techniques include the encoder-decoder structure [25], skip connection [3] and ASPP [3]. However, the fully supervised segmentation systems cannot generalize to the unseen class segmentation scenario. In this paper, SAGNN aims to solve the unseen class segmentation problem.

Few-Shot Semantic Segmentation. Almost all existing FSS models exploit a two-branch architecture design to conduct meta-training on a base training set, followed by meta-testing on a disjoint test set. OSLSM [27] is the pioneer work for FSS, consisting of a conditional branch and a segmentation branch. The conditional branch is used to generate classifier weights for segmenting the query image. Later on, single or multiple prototype-based methods are proposed under this two-branch paradigm, representative ones including PL [7], coFCN [24], A-MCG [15], SG-One [48], PANet [34], FWB [22], CANet [47], SimPropNet [10], PPN [20] and PMM [41]. The core of these methods lies in constructing meaningful prototypes with different strategies, which, however, undertake a few-to-many matching among the support-guided prototypes and query features. It is thus difficult to well segment query images with limited prototypes. Some works [19, 33, 42, 46] explore pixel-to-pixel matching based on dual-image attention, which still fail to preserve the structural informations.

Recently, PFENet [31] claims two findings about FSS that CANet and BriNet [42] have also validated, i.e., *during meta-training of FSS, the backbone weights should be fixed and both the mid- and high-level features should be used*. By doing so, the generalization of the models can be preserved as much as possible. As such, in this paper, we also adopt these strategies for building our SAGNN models. In the literatures, PGNet [46] and PFENet are two methods that are most related to SAGNN. They both use multi-scale features for FSS, however, their designs are heuristic (e.g., the top-down design in PFENet) and lack the ability to model the high-order relationships. In contrast, SAGNN is the first work in FSS field to model object variations among support-query images in a structural manner.

3. Method

3.1. Task Definition

As mentioned in §1, like most FSS methods, we utilize episode-based meta-learning [30] to perform model training. We denote the base training set and the test set as $\mathcal{D}_{\text{tr}}$ and $\mathcal{D}_{\text{ts}}$, respectively, and their label sets are disjoint from each other. To train SAGNN, we randomly sample multiple episodes from $\mathcal{D}_{\text{tr}}$ for meta-training. Each episode consists of a support set $\mathcal{S}$ and a query set $\mathcal{Q}$. After completing the SAGNN training, we sample episodes from $\mathcal{D}_{\text{ts}}$ for testing. For the K-shot FFS task, suppose $\mathcal{S} = \{(x_i^s, m_i^s)\}_{i=1}^K$, which contains K image-mask pairs, i.e., x_i^s and m_i^s are the ith support image and its ground-truth (GT) binary mask for a specific class c, respectively. Meanwhile, let $\mathcal{Q} = \{(x^q, m^q)\}$, where, x^q and m^q are the query image and its GT binary mask sampled from the same class c. Each episode $(\mathcal{S}, \mathcal{Q})$ focuses on a different class c and tries to produce a prediction mask $\hat{m}^q$ for the query image, by taking $\mathcal{S}$ and x^q as inputs. In this way, the binary cross-entropy loss between $\hat{m}^q$ and m^q, i.e., $\text{BCE}(\hat{m}^q, m^q)$, is further used to supervise the meta-training. Finally, given the trained model, we sample multiple episodes, $\{(\mathcal{S}_i^{\text{ts}}, \mathcal{Q}_i^{\text{ts}})\}_{i=1}^{N_{\text{ts}}}$, from $\mathcal{D}_{\text{ts}}$ for meta-testing.

As most FFS model [31,41,42,46], the one-way scenario is our focus in this paper, i.e., the category number (way) in each episode is one in our model. In the following, for simplicity, we consider the 1-shot setting (i.e., K=1 in $\mathcal{S}$) to illustrate our SAGNN.

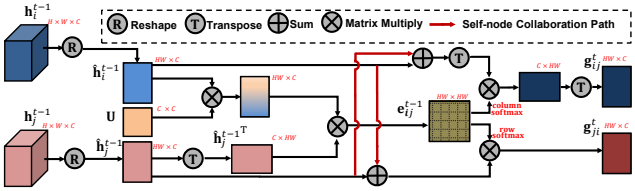

Figure 3. The edge embedding and message generation process with self-node collaboration among two node embeddings.

3.2. Overview

SAGNN (Fig. 2) [26] improves the mainstream two-branch FSS models by introducing cross-scale relation reasoning in a structure-to-structure manner. Given an episode $(\mathcal{S}, \mathcal{Q})$, SAGNN is parameterized as support-induced multi-scale query nodes, edges, and the differentiable message passing functions among them. As such, under the 1-shot setting, SAGNN builds a scale-aware graph, $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, based on the support image-mask pair (x^s, m^s) $(\in \mathcal{S})$ and the query image x^q. Here, $\mathcal{V} = \{v_i\}_{i=1}^{|\mathcal{V}|}$ is the node set with $|\mathcal{V}|$ as the number of scales (or graph size/node number), and each v_i can be initialized as a fused feature map (node embedding) $\mathbf{h}_i^0$ under the ith scale, which is thus a structural representation with spatial layout. To fully exploit the cross-scale relations among these nodes, $\mathcal{G}$ is modelled as a fully-connected graph. Each edge $e_{ij} \in \mathcal{E}$ connecting v_i and v_j denotes a directed relation of $v_i \rightarrow v_j$. Further, SAGNN conducts T progressive message passing steps over $\mathcal{G}$ to capture the high-order cross-scale relationships among each episode. This leads to an updated node representation for each node, termed as $\{\mathbf{h}_i^T\}_{i=1}^{|\mathcal{V}|}$. Next, a readout module is used to fuse these node features and get the query prediction mask $\hat{m}^q$. In §3.3, each component of SAGNN is described in detail.

3.3. Scale-Aware Graph Neural Network

Scale-Aware Node Embedding. As discussed in §2, following the findings by [31, 42, 47], we fix the backbone weights, use mid- and high-level features for node feature extraction, and adopt the maximum support-foreground-response mask [31]. Specifically, for ResNet with layers divided into four groups (*block*1-4), the spatial size of feature maps after *block*2 is the same by dilated convolution [3]. As in [31,47], we concatenate the feature maps after *block*2 and 3, and compress them to a fixed channel dimension C using a 1×1 convolution. Denoting the above process as $\mathcal{B}$, given the support/query image x^s/x^q, we can obtain their convolutional feature maps under the largest scale:

$$\mathbf{f}^s = \mathcal{B}(x^s) \in \mathbb{R}^{H\times W\times C}, \quad \mathbf{f}^q = \mathcal{B}(x^q) \in \mathbb{R}^{H\times W\times C}, \tag{1}$$

where H,W,C are their height, width, and channel dimension, respectively, and $H \times W$ is our adopted largest resolution. To obtain the multi-scale node embeddings (Fig. 2), we first conduct pyramid average-pooling [49] on $\mathbf{f}^q$ to generate $|\mathcal{V}|$ new-scaled features $\mathcal{F}^q = \{\mathbf{f}_i^q\}_{i=1}^{|\mathcal{G}|}$ with $\mathbf{f}_i^q \in \mathbb{R}^{H_i\times W_i\times C}$ and $\mathbf{f}^q \in \mathcal{F}^q$. We suppose $\mathbf{f}_1^q = \mathbf{f}^q$ (i.e., H_1=H, W_1=W), and H_i>H_j, W_i>W_j when i<j.

In addition, the widely-used global masked average pooling vector w.r.t. (x^s, m^s) is calculated as follows:

$$f_{avg}^s = \text{avg\_pool}(\mathbf{f}^s \odot \mathcal{R}(\bar{m}^s)) \in \mathbb{R}^{1\times1\times C}, \tag{2}$$

where $\bar{m}^s$ has a size of $W \times H$ by down-sampling m^s, $\mathcal{R}$ reshapes $\bar{m}^s$ to be the same shape as $\mathbf{f}^s$, and $\odot$ denotes element-wise multiplication. Motivated by [31], we further use the high-level features $\mathbf{f}_h^s$ and $\mathbf{f}_h^q$ (also having $H \times W$ spatial size due to the dilated convolution) w.r.t. x^s,x^q after *block*4, to get the maximum support-foreground-response (sfr) mask f_{sfr}^q $(\in \mathbb{R}^{H\times W})$. The mask value $f_{sfr}^q(x, y)$ in location (x,y) is obtained by first using $\mathbf{f}_h^s \odot \mathcal{R}(\bar{m}^s)$ to achieve a cosine similaritiy vector with $\mathbf{f}_h^q(x, y)$, followed by finding the maximum response in it. To this end, based on $\mathcal{F}^q$, f_{avg}^s and f_{sfr}^q, we construct the node embedding $\mathbf{h}_i^0 \in \mathbb{R}^{H\times W\times C}$ w.r.t. the ith scale as follows:

$$\mathbf{h}_i^0 = \mathcal{I}_{H\times W}(\mathcal{C}(\mathbf{f}_i^q \oplus \mathcal{P}_{H_i\times W_i}(f_{avg}^s) \oplus \mathcal{I}_{H_i\times W_i}(f_{sfr}^q))), \tag{3}$$

where $\mathcal{P}_{x\times y}$ expands the input vector to a spatial size $x \times y$, $\mathcal{I}_{x\times y}$ down- or up-samples the input to a spatial size $x \times y$ by interpolation, $\mathcal{C}$ consists of one 1×1 and two 3×3 convolutions for channel compression and feature fusion, and $\oplus$ is the concatenating operation.

Edge Embedding. $\mathcal{G}$ is a fully-connected graph without self-connections. An edge $e_{ij} \in \mathcal{E}$ indicates a directed connection of node $v_i \rightarrow v_j$. Since we have achieved the initial node embeddings (or representations) $\mathbf{h}_i^0$ (Eq. (3)) w.r.t. v_i for a given episode, for the tth GNN reasoning, we utilize the edge embedding $\mathbf{e}_{ij}^t$ to represent the relation of $v_i \rightarrow v_j$ (Fig. 2). As node embeddings are structural feature maps, we use the dual-image attention variant [36,42] to grasp the relation of $v_i \rightarrow v_j$ and vice versa:

$$\mathbf{e}_{ij}^t = \hat{\mathbf{h}}_i^t \mathbf{U} \hat{\mathbf{h}}_j^{t\top} \in \mathbb{R}^{HW\times HW}, \tag{4}$$

where $\hat{\mathbf{h}}_i^t \in \mathbb{R}^{WH\times C}$ $(\hat{\mathbf{h}}_j^t \in \mathbb{R}^{WH\times C})$ is reshaped by $\mathbf{h}_i^t \in \mathbb{R}^{W\times H\times C}$ $(\mathbf{h}_j^t \in \mathbb{R}^{W\times H\times C})$, and $\mathbf{U} \in \mathbb{R}^{\mathbf{C}\times\mathbf{C}}$ is a learnable matrix during all the episodes (Fig. 3). Similarly, we leverage $\mathbf{e}_{ji}^t$=$\mathbf{e}_{ij}^{t\top}$=$\hat{\mathbf{h}}_j^t \mathbf{U}^\top \hat{\mathbf{h}}_i^{t\top}$ to represent the relation of $v_j \rightarrow v_i$. $\mathbf{e}_{ij}^t$ reflects the long-range similarity between node features with different resolutions (scales).

Message Passing with Self-Node Collaboration. In a standard GNN, in iteration t, the message $\mathbf{g}_{ji}^t$ passed from v_j to v_i is defined as (Fig. 3):

$$\mathbf{g}_{ji}^t = \mathcal{M}(\hat{\mathbf{h}}_j^{t-1}, \mathbf{e}_{ij}^{t-1}) = \text{softmax}(\mathbf{e}_{ij}^{t-1})\hat{\mathbf{h}}_j^{t-1} \in \mathbb{R}^{HW\times C}, \tag{5}$$

where $\text{softmax}(\cdot)$ is a row-wise softmax normalization. Under SAGNN, taking the kth row of $\mathbf{e}_{ij}^{t-1}$ as an example, it represents the similarity of location k in node feature

i with all locations in node feature j. Since different node features (i.e., $\hat{\mathbf{h}}_i^{t-1}$ and $\hat{\mathbf{h}}_j^{t-1}$, with the same spatial size) represent different-resolution features for the same query image (Eq. (3)), $\text{softmax}(\mathbf{e}_{ij}^{t-1})\hat{\mathbf{h}}_i^{t-1} \in \mathbb{R}^{HW\times C}$ becomes meaningful, and each of its rows is thus a weighted summation of each location of $\hat{\mathbf{h}}_i^{t-1}$ and weights in columns of $\text{softmax}(\mathbf{e}_{ij}^{t-1})$. Notably, $\hat{\mathbf{h}}_i^{t-1}$ itself contains the cross-resolution location information. As such, to enable SAGNN to perceive different resolutions of the same object, we improve Eq. (5) by including a self-node collaboration:

$$\begin{aligned}\mathbf{g}_{ji}^t =&\mathcal{M}(\hat{\mathbf{h}}_i^{t-1}, \hat{\mathbf{h}}_j^{t-1}, \mathbf{e}_{ij}^{t-1}) \\ =&\text{softmax}(\mathbf{e}_{ij}^{t-1})(\hat{\mathbf{h}}_j^{t-1} + \hat{\mathbf{h}}_i^{t-1}) \in \mathbb{R}^{HW\times C}.\end{aligned} \tag{6}$$

Experimental evaluation shows performance gains by our simple self-node collaboration (Fig. 3 and Table 6). Finally, the total messages to v_i from all neighboring nodes are:

$$\mathbf{g}_i^t = \mathcal{F}_{\text{reshape}}(\sum_{v_j\in\mathcal{V}(i)} \mathbf{g}_{ji}^t) \in \mathbb{R}^{H\times W\times C}, \tag{7}$$

where $\mathcal{F}_{\text{reshape}}(\cdot)$ reshapes the input size to $H \times W \times C$.

Node-State Updating. In iteration t, after acquiring the self-node collaborated message $\mathbf{g}_i^t$ (Eq. (7)) and the node feature in the (t-1)th iteration $\mathbf{h}_i^{t-1}$, we utilize ConvGRU [2] to update the state of the node v_i:

$$\mathbf{h}_i^t = \mathcal{U}_{\text{GRU}}(\mathbf{h}_i^{t-1}, \mathbf{g}_i^t) \in \mathbb{R}^{H\times W\times C}. \tag{8}$$

SAGNN is the first work using ConvGRU to update the node-state of the whole feature maps in the FSS field.

Episodic Readout Module. As discussed in §3.2, we achieve the updated node representations $\{\mathbf{h}_i^T\}_{i=1}^{|\mathcal{V}|}$, after T times of message passing. To fuse them, we conduct the following sequential operations to readout the predicted segmentation mask $\hat{m}^q$ for the query x^q under the current episode.

$$\hat{m}^q = \mathcal{F}_{\text{cls}}(\mathcal{F}_{\text{ASPP}}(\mathcal{F}_{\text{conv}}(\mathbf{h}_1^T \oplus \mathbf{h}_2^T \oplus \cdots \oplus \mathbf{h}_{|\mathcal{V}|}^T))) \in \mathbb{R}^{h\times w\times 2}, \tag{9}$$

where $\mathcal{F}_{\text{conv}}$, $\mathcal{F}_{\text{ASPP}}$ and $\mathcal{F}_{\text{cls}}$ together constitute the fusion operations; details of their implementations are provided in §4.2. In Eq. (9), h and w are the height and width of the ground-truth mask m^q, and $\mathcal{O}(\cdot)$=$\mathcal{F}_{\text{cls}}(\mathcal{F}_{\text{ASPP}}(\mathcal{F}_{\text{conv}}(\cdot)))$ is dubbed our readout function.

Our SAGNN undergoes an episode-based meta-training process; we therefore call this module an episodic readout module. The components (Eq. (3)-(9)) in SAGNN are all differentiable and their weights are shared during meta-training. As such, SAGNN can be trained in an end-to-end manner and can transfer structural knowledge from the seen to unseen domain (meta-testing).

3.4. Training Loss for SAGNN

Our SAGNN is trained in a meta-learning paradigm, and each episode is equivalent to a *data sample* in general learning algorithms. §3.3 shows the whole process of SAGNN training over a single episode, under a 1-shot setting. In practice, we train SAGNN in a batch mode by sampling a batch of N_e episodes ($\{(\mathcal{S}_i, \mathcal{Q}_i)\}_{i=1}^{N_e}$). Suppose $\mathcal{Q}_i = (x_i^q, m_i^q)$, based on Eq. (9). We can get the predicted mask, denoted as $\hat{m}_i^q \in \mathbb{R}^{h\times w\times 2}$, for each query x_i^q. To this end, we leverage the binary cross entropy (BCE) loss (over all spatial locations) among $\hat{m}_i^q$ and m_i^q to update all parameters in SAGNN. This loss is denoted as our main loss:

$$\mathcal{L}_{\text{main}} = \frac{1}{N_e}\sum_{i=1}^{N_e} \text{BCE}(\hat{m}_i^q, m_i^q). \tag{10}$$

In addition, motivated by the deeply-supervised nets and PFENet [17,31], after obtaining the initial node embeddings without the $\mathcal{I}_{H\times W}$ operation (see Eq. (3)) for $|\mathcal{V}|$ scales – denoted as $\{\tilde{\mathbf{h}}_i^0\}_{i=0}^{|\mathcal{V}|}$ – we use $|\mathcal{V}|$ encoding branches to predict the segmentation masks for each of these scales. These $|\mathcal{V}|$ encoding branches are implemented as one 3×3 and one 1×1 convolutions. As such, under the N_e episodes, we obtain another auxiliary loss:

$$\mathcal{L}_{\text{auxiliary}} = \frac{1}{N_e \times |\mathcal{V}|}\sum_{i=1}^{N_e}\sum_{k=1}^{|\mathcal{V}|} \text{BCE}(\mathcal{F}_{\text{enc}}^k(\tilde{\mathbf{h}}_k^0), m_i^q), \tag{11}$$

where $\mathcal{F}_{\text{enc}}^k(\cdot)$ is the encoding function of the kth scale features. Our final training loss for SAGNN is:

$$\mathcal{L} = \mathcal{L}_{\text{main}} + \alpha\mathcal{L}_{\text{auxiliary}}, \tag{12}$$

where α is set to 1.0 in all the experiments. More implementation details of SAGNN are in §4.2.

4. Experiments

4.1. Settings

Datasets. We use two standard FFS datasets, i.e., PASCAL-5^i [27] and COCO-20^i [22], to evaluate SAGNN. PASCAL-5^i is built from PASCAL VOC 2012 [9] with additional annotations from SDS [13], meanwhile, COCO-20^i is created from MSCOCO [18].

For the two datasets used, the cross-validation results are reported by evenly dividing the data into different folds – i.e., four folds for both of them – according to the categories. We use the same data splits as [27] and [22] on PASCAL-5^i and COCO-20^i, respectively. As such, for each cross-validation on PASCAL-5^i and COCO-20^i, 15 and 60 object classes serve as the training data with the remaining (5 and 20 object classes) used as the test data. For meta-testing under each cross-validation for the two datasets, we randomly sample 1,000 episodes (support-query pairs) from the test set and evaluate their metrics.

Metrics. Like other methods [27,34,35], two evaluation metrics, i.e., mean-IoU and FB-IoU, are adopted in this paper. mean-IoU is calculated by averaging the intersection-over-unions (IoUs) over different foreground classes in the

Methods	Backbone	mean-IoU (1-shot)					FB-IoU	mean-IoU (5-shot)					FB-IoU
		Fold-0	Fold-1	Fold-2	Fold-3	Mean	(1-shot)	Fold-0	Fold-1	Fold-2	Fold-3	Mean	(5-shot)
OSLSM (BMVC'17) [27]	VGG-16	33.6	55.3	40.9	33.5	40.8	61.3	35.9	58.1	42.7	39.1	43.9	61.5
co-FCN (ICLRW'18) [24]	VGG-16	31.7	50.6	44.9	32.4	41.1	60.1	37.5	50.0	44.1	33.9	41.4	60.2
AMP (ICCV'19) [28]	VGG-16	41.9	50.2	46.7	34.7	43.4	62.2	41.8	55.5	50.3	39.9	46.9	63.8
SG-One (TCYB'19) [48]	VGG-16	40.2	58.4	48.4	38.4	46.3	63.1	41.9	58.6	48.6	39.4	47.1	65.9
PANet (ICCV'19) [34]	VGG-16	42.3	58.0	51.1	41.2	48.1	66.5	51.8	64.6	59.8	46.5	55.7	70.7
CANet (CVPR'19) [47]	ResNet-50	52.5	65.9	51.3	51.9	55.4	66.2	55.5	67.8	51.9	53.2	57.1	69.6
PGNet (ICCV'19) [46]	ResNet-50	56.0	66.9	50.6	50.4	56.0	69.9	57.7	68.7	52.9	54.6	58.5	70.5
FWB (ICCV'19) [22]	ResNet-101	51.3	64.5	56.7	52.2	56.2	-	54.8	67.4	62.2	55.3	59.9	-
PMMs (ECCV'20) [41]	ResNet-50	52.0	67.5	51.5	49.8	55.2	-	55.0	68.2	52.9	51.1	56.8	-
PPNet (ECCV'20) [20]	ResNet-50	47.8	58.8	53.8	45.6	51.5	-	58.4	67.8	**64.9**	56.7	62.0	-
DAN (ECCV'20) [33]	ResNet-101	54.7	68.6	**57.8**	51.6	58.2	71.9	57.9	69.0	60.1	54.9	60.5	72.3
PFENet (TPAMI'20) [31]	ResNet-50	61.7	69.5	55.4	56.3	60.8	73.3	63.1	**70.7**	55.8	57.9	61.9	73.9
BriNet (BMVC'20) [42]	ResNet-50	56.5	67.2	51.6	53.0	57.1	-	-	-	-	-	-	-
SimPropNet (IJCAI'20) [10]	ResNet-50	54.9	67.3	54.5	52.0	57.2	73.0	57.2	68.5	58.4	56.1	60.0	72.9
Baseline	ResNet-50	62.1	68.2	55.3	53.8	59.9	71.7	63.3	68.7	55.1	55.3	60.6	71.8
SAGNN	ResNet-50	**64.7**	**69.6**	57.0	**57.2**	**62.1**	73.2	**64.9**	70.0	57.0	59.3	**62.8**	73.3

Table 1. Results under 1-shot and 5-shot settings on PASCAL-5^i. mean-IoU under each fold, and the averaged mean-IoU (termed as Mean) and averaged FB-IoU under four folds are reported. Baseline results are obtained by removing the graph reasoning in SAGNN. The best mean-IoUs are marked in **bold**. FB-IoU is biased towards and benefits from the background class to achieve a good number, as such it is a reference indicator (See also *italic text* in §4.3). In addition, some of the leading methods, e.g., FWB, do not even report their FB-IoUs.

Methods	Backbone	mean-IoU (1-shot)					FB-IoU	mean-IoU (5-shot)					FB-IoU
		Fold-0	Fold-1	Fold-2	Fold-3	Mean	(1-shot)	Fold-0	Fold-1	Fold-2	Fold-3	Mean	(5-shot)
PANet (ICCV'19) [34]	VGG-16	-	-	-	-	20.9	59.2	-	-	-	-	29.7	63.5
FWB (ICCV'19) [22]	VGG-16	18.4	16.7	19.6	25.4	20.0	-	20.9	19.2	21.9	28.4	22.6	-
PFENet (TPAMI'20) [31]	VGG-16	33.4	36.0	34.1	32.8	34.1	60.0	35.9	40.7	38.1	36.1	37.7	61.6
FWB (ICCV'19) [22]	ResNet-101	17.0	18.0	21.0	28.9	21.2	-	19.1	21.5	23.9	30.1	23.7	-
PMMs (ECCV'20) [41]	ResNet-101	29.3	34.8	27.1	27.3	29.6	-	33.0	40.6	30.1	33.3	34.3	-
DAN (ECCV'20) [33]	ResNet-101	-	-	-	-	24.4	62.3	-	-	-	-	29.6	63.9
PPNet (ECCV'20) [20]	ResNet-50	28.1	30.8	29.5	27.7	29.0	-	39.0	40.8	37.1	37.3	38.5	-
BriNet (BMVC'20) [42]	ResNet-50	32.9	36.2	37.4	30.9	34.4	-	-	-	-	-	-	-
PFENet (TPAMI'20) [31]	ResNet-101	34.3	33.0	32.3	30.1	32.4	58.6	38.5	38.6	38.2	34.3	37.4	61.9
Baseline	VGG-16	32.1	36.1	35.2	32.3	33.9	60.1	35.0	40.1	37.1	36.5	37.2	61.8
SAGNN	VGG-16	**35.0**	**40.5**	**37.6**	**36.0**	**37.3**	61.2	**37.2**	**45.2**	**40.4**	**40.0**	**40.7**	63.1
Baseline	ResNet-101	32.0	36.3	35.0	33.0	34.1	60.2	36.8	40.5	36.8	37.2	37.8	62.4
SAGNN	ResNet-101	**36.1**	**41.0**	**38.2**	**33.5**	**37.2**	60.9	**40.9**	**48.3**	**42.6**	**38.9**	**42.7**	63.4

Table 2. Results under 1-shot and 5-shot settings on COCO-20^i. mean-IoU under each fold, and the averaged mean-IoU (termed as Mean) and averaged FB-IoU under four folds are reported. Baseline results are obtained by removing the graph reasoning in SAGNN. The best mean-IoUs are marked in **bold**. As in Table 1, FB-IoU serves as a reference indicator as well (See also *italic text* in §4.3).

test set. The mean-IoU of each fold and averaged mean-IoU of four folds are reported during cross-validations. FB-IoU is the foreground/background IoU, which takes all object classes in the test set as a single foreground class and the IoUs of foreground/background are averaged to obtain the FB-IoU. The mean-IoU is the key evaluation criterion for FFS, since the performance bias of scarce classes can be alleviated by considering the differences of all classes. As such, we solely report the averaged FB-IoU over four folds.

4.2. Implementation Details and Parameters

We use ResNet-50 [14] (VGG-16 [29] and ResNet-101) to conduct our main body experiments on PASCAL-5^i (COCO-20^i). Ablation experiments for different backbones on PASCAL-5^i are carried out and shown in Table 3. As in [10, 31, 42, 47], all of the backbones used are initialized by the ImageNet [6] pre-trained models and are fixed during SAGNN training. The reasons for doing so can be found in §3.3 w.r.t. the *scale-aware node embedding*. The channel dimension C (Eq. (1)) is fixed as 256 in all of the experiments. For ResNet, dilated convolutions are used to make the feature maps after *block*2 have a size of about 1/8 of the input. As such, by taking images with size 473×473 as input for ResNet-50, we can get the largest feature map with spatial size 60×60, i.e., $H = W = 60$ in Eq. (1). In fact, the resolutions (scales) used in $\mathcal{F}^q$ on PASCAL-5^i and COCO-20^i are [60×60, 15×15, 8×8] and [40×40, 20×20, 10×10], respectively. This also means that the node number $|\mathcal{V}| = 3$. Ablation studies show that three scales have better results than scales larger than three. Furthermore, the 1×1 and 3×3 convolutions in $\mathcal{C}$ (Eq. (3)) all have 256 channel outputs. The readout functions in Eq. (9) are specified as: $\mathcal{F}_{\text{conv}}$: two 1×1 convolutions all with 256 output channels, followed by two 3×3 convolutions all with 256 channel and residual connections, finally another two 3×3 convolutions

Figure 4. Segmentation results on unseen classes under 1-shot setting. The left panel is from PASCAL-5^i, and right panel from COCO-20^i. Each column indicates one test episode and its predictions. From top to bottom, each row represents support images and their ground-truth (GT) masks (green), query images and their GT masks (yellow), baseline predictions (red), SAGNN predictions (red), respectively (§4.5).

with 256 channels and residual connections. $\mathcal{F}_{\text{ASPP}}$: atrous spatial pyramid pooling with dilation rates 6, 12, 18 of 3×3 convolution and a 1×1 convolution without dilation. $\mathcal{F}_{\text{cls}}$: a 3×3 convolution with 256 channels, and a 1×1 convolution with 2 channels for predicting foreground/background.

We implement SAGNN using Pytorch on a Tesla V100 GPU. SAGNN is trained in an episode-based meta-learning fashion; the batch size N_e in Eq. (11) for PASCAL-5^i and COCO-20^i is 4 and 8, respectively. The SGD optimizer is used with a learning rate of 0.0025 (for 100 epochs) and 0.005 (for 50 epochs) on PASCAL-5^i and COCO-20^i, respectively. In addition, momentum and weight decay are 0.9 and 0.0001, and 'poly' strategy is also used. We try different times (T) of message passing, and find that the results with $T = 1$ and $T = 2$ are all promising.

For the K-shot setting (K>1), as in [22, 42, 47], we leverage a feature-level early fusion strategy [24] by averaging the support image features to get a single fused support feature. Then, the subsequent operations are the same as the 1-shot case. Table 4 shows the comparisons to other late fusion methods.

4.3. Comparisons with State-of-the-Arts

We compare SAGNN (details are depicted in §3.3 and §4.2) with all the leading FFS methods under the same evaluation metrics on the two datasets used. Notably, unlike [46, 47] that use multi-scale testing, we use a single scale as input during meta-testing.

PASCAL-5^i. The mean-IoU and FB-IoU under 1-shot and 5-shot are shown in Table 1. We conclude that **(i)** SAGNN consistently outperforms the compared methods under the averaged mean-IoU (on four folds), although it is only slightly better than [31] on fold1. **(ii)** SAGNN significantly outperforms its vallina baseline method (e.g., the averaged mean-IoU is 62.1 versus 59.9), which is implemented with the same multi-scale nodes as SAGNN, yet without the graph reasoning among them. This further demonstrates that the proposed scale-aware message passing model intrinsically captures the high-order relationships of the cross-scale features and can advance the FSS task greatly. **(iii)** *Since most foreground classes only occupy a small spatial region of the whole image, the FB-IoU is biased towards and benefits from the background class to achieve a good number, which makes it not convincing to evaluate the performances. However, we also show the averaged FB-IoU on four folds, and the numbers are competitive* (Table 1 and 2).

COCO-20^i. Similarly, we illustrate the mean-IoU and FB-IoU in Table 2, from which it can be seen that SAGNN achieves state-of-the-art results under both 1-shot and 5-shot settings by a clear margin. For instance, with the same protocols and the same VGG-16 backbone, taking the averaged mean-IoU as the metric, we have **(1)** SAGNN outperforms PFENet [31] by 3.2 and 3.0 on 1-shot and 5-shot respectively. **(2)** SAGNN outperforms FWB [22] by even 17.3 and 18.1 on 1-shot and 5-shot, respectively. As such, the cross-scale structural relation modeling in SAGNN indeed captures some inner benefits for boosting FFS performance, and we hope our model can shed light for future research in FFS.

4.4. Ablation Study

PASCAL-5^i is used to perform the following ablation studies, and unless specified, we mainly report averaged

Backbone	mean-IoU	
	1-shot	5-shot
VGG-16	58.4	59.3
ResNet-50	**62.1**	**62.8**
ResNet-101	60.8	61.5

Table 3. Effects of backbones.

5-shot testing	mean-IoU	FB-IoU
1-shot baseline	62.1	73.2
Feature-Avg	**62.8**	**73.3**
Mask-Avg	62.5	72.4
Mask-OR	61.8	72.0

Table 4. Feature fusion under 5-shot setting.

Setting		mean-IoU	
$\lvert\mathcal{V}\rvert$	T	1-shot	5-shot
1	1	n.a.	n.a.
2	1	61.0	61.7
3	1	**62.1**	**62.8**
4	1	61.0	61.9
3	1	62.1	62.8
3	2	**62.4**	**62.9**
3	3	62.1	62.5

Table 5. Effects of $\lvert\mathcal{V}\rvert$ and T.

Models	mean-IoU	
	1-shot	5-shot
SAGNN *w* SC	61.2	62.7
SAGNN *w* OI	61.1	62.4
Full SAGNN	**62.1**	**62.8**

Table 6. Effects of self-node collaboration.

mean-IoU over 4 folds under 1-shot and 5-shot settings.

Effects of Node Number $|\mathcal{V}|$. The node number $|\mathcal{V}|$ is the key parameter in SAGNN, which controls the tradeoff between the accuracy and training cost. We vary $|\mathcal{V}|$ from 1 to 4 to observe the performances of SAGNN. In Table 5, $|\mathcal{V}|$=3 performs best, which indicates that SAGNN equipped with 3 nodes (scales) is desirable to model the cross-scale relations. As such, we set $|\mathcal{V}|$=3 in all of our experiments.

Effects of Self-Node Collaboration (SC). Since Eq. (5) essentially captures opposite-node information (OI), we thus denote the SAGNN model variants using the *message* formulas in Eq. (5) and Eq. (6) as SAGNN *w* OI and the full SAGNN, respectively. Another variant solely using SC is also considered, i.e., $\mathbf{g}_{ji}^t = \mathrm{softmax}(\mathbf{e}_{ij}^{t-1})\hat{\mathbf{h}}_i^{t-1}$, termed as SAGNN *w* SC. Results in Table 6 show that a sizable gains are achieved by our simple SC.

Message Passing Iteration Number T. We take the values of T from $1, 2, 3$ to investigate the performances of our full SAGNN model. We conclude from Table 5 that T=2 and T=1 achieve the overall best and second best result. However, the performance becomes slightly worse when T=3. We take $T = 1$ to conduct our main experiments.

Coefficient α of $\mathcal{L}_{\text{auxiliary}}$. We set $\alpha = 1.0$ (Eq. (12)) in all experiments. We further vary its values from $\{0.0, 0.5, 1.0\}$ to observe the mean-IoU under each fold and the averaged mean-IoU of four folds, under 1- and 5-shot settings (Fig. 5). As in Fig. 5, the performances are robust w.r.t. different α and better results are achieved when α=1.0.

Effects of Backbones. To evaluate the effects of different backbones on the full SAGNN model, VGG-16, ResNet-50, and ResNet-101 are used to conduct experiments under the same settings. As shown in Table 3, ResNet-50 yields a superior result.

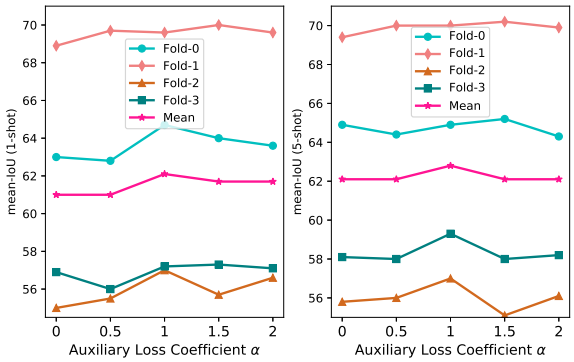

Figure 5. mean-IoUs under different auxiliary loss coefficients α.

Feature Fusion under 5-shot Setting. As stated in § 4.2, SAGNN averages five support image features for fusion (Feature Fusion), which is an early fusion strategy. Two late fusion methods, i.e., OR fusion on masks [27] and average fusion on masks [47], are compared in Table 4. For SAGNN, feature fusion performs best, and is thus adopted to derive the 5-shot results.

4.5. Qualitative Results

We take testing episodes from PASCAL-5^i and COCO-20^i to visualize the segmentation results, under a 1-shot setting. Fig. 4 depicts the qualitative comparisons of SAGNN with the *baseline without graph reasoning* (i.e., T=0 in Eq. (9)) yet still using the same scales as SAGNN. By contrast, besides the intrinsic nature for graph-based model to capture the appearance relationships, SAGNN also has several other merits: (1) It can overcome object scale variations by mutual passing of scale *messages*, e.g., the *sofa* and *cake* objects are well segmented. (2) It can alleviate the cluttered background issue thanks to the support-induced query node construction strategy (see §3.3), thus leading to accurate estimates for *bird* and *clock*. (3) It can mitigate object location variations using self-node collaboration (Eq. (6)), e.g., *flower pot* and *motorbike*.

5. Conclusion

In this paper, a scale-aware graph neural network (SAGNN) is proposed to tackle the challenging and important few-shot semantic segmentation (FSS) task. Specifically, a scale-aware graph is first built upon the support-induced query nodes, followed by graph reasoning on these nodes. A novel self-node collaboration mechanism is also proposed to enrich the current node features during feature aggregation. SAGNN sets new state-of-the-arts on both PASCAL-5^i and COCO-20^i.

Acknowledgments This work was supported by the National Natural Science Foundation of China (Nos. 61702163).

References

[1] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. In *TPAMI*, 2017. 1, 2

[2] Nicolas Ballas, Li Yao, Chris Pal, and Aaron Courville. Delving deeper into convolutional networks for learning video representations. In *arXiv:1511.06432*, 2015. 5

[3] 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. In *TPAMI*, 2017. 1, 2, 3, 4

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

[5] Tao Chen, Guo-Sen Xie, Yazhou Yao, Qiong Wang, Fumin Shen, Zhenmin Tang, and Jian Zhang. Semantically meaningful class prototype learning for one-shot image segmentation. *TMM*, 2021. 2

[6] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009. 6

[7] Nanqing Dong and Eric P Xing. Few-shot semantic segmentation with prototype learning. In *BMVC*, 2018. 2, 3

[8] David K Duvenaud, Dougal Maclaurin, Jorge Iparraguirre, Rafael Bombarell, Timothy Hirzel, Alán Aspuru-Guzik, and Ryan P Adams. Convolutional networks on graphs for learning molecular fingerprints. In *NeurIPS*, 2015. 2

[9] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. In *IJCV*, 2010. 5

[10] Siddhartha Gairola, Mayur Hemani, Ayush Chopra, and Balaji Krishnamurthy. Simpropnet: Improved similarity propagation for few-shot image segmentation. In *IJCAI*, 2020. 2, 3, 6

[11] Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. Neural message passing for quantum chemistry. In *arXiv:1704.01212*, 2017. 2

[12] Marco Gori, Gabriele Monfardini, and Franco Scarselli. A new model for learning in graph domains. In *IJCNN*, 2005. 2

[13] Bharath Hariharan, Pablo Arbeláez, Lubomir Bourdev, Subhransu Maji, and Jitendra Malik. Semantic contours from inverse detectors. In *ICCV*, 2011. 5

[14] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 1, 6

[15] 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 *AAAI*, 2019. 3

[16] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *arXiv preprint arXiv:1609.02907*, 2016. 2

[17] Chen-Yu Lee, Saining Xie, Patrick Gallagher, Zhengyou Zhang, and Zhuowen Tu. Deeply-supervised nets. In *Artificial intelligence and statistics*, pages 562–570, 2015. 5

[18] 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 *ECCV*, 2014. 5

[19] Weide Liu, Chi Zhang, Guosheng Lin, and Fayao Liu. Crnet: Cross-reference networks for few-shot segmentation. In *CVPR*, 2020. 2, 3

[20] Yongfei Liu, Xiangyi Zhang, Songyang Zhang, and Xuming He. Part-aware prototype network for few-shot semantic segmentation. In *ECCV*, 2020. 2, 3, 6

[21] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *CVPR*, 2015. 1, 2

[22] Khoi Nguyen and Sinisa Todorovic. Feature weighting and boosting for few-shot segmentation. In *CVPR*, 2019. 2, 3, 5, 6, 7

[23] George Papandreou, Liang-Chieh Chen, Kevin P Murphy, and Alan L Yuille. Weakly-and semi-supervised learning of a deep convolutional network for semantic image segmentation. In *ICCV*, 2015. 1

[24] Kate Rakelly, Evan Shelhamer, Trevor Darrell, Alyosha Efros, and Sergey Levine. Conditional networks for few-shot semantic segmentation. In *ICLR Workshop*, 2018. 3, 6, 7

[25] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *MICCAI*, 2015. 1, 2, 3

[26] Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. The graph neural network model. In *TNNLS*, 2008. 2, 4

[27] Amirreza Shaban, Shray Bansal, Zhen Liu, Irfan Essa, and Byron Boots. One-shot learning for semantic segmentation. In *BMVC*, 2017. 1, 3, 5, 6, 8

[28] Mennatullah Siam, Boris N Oreshkin, and Martin Jagersand. Amp: Adaptive masked proxies for few-shot segmentation. In *ICCV*, 2019. 2, 6

[29] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2014. 1, 6

[30] Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. In *NeurIPS*, 2017. 1, 3

[31] Zhuotao Tian, Hengshuang Zhao, Michelle Shu, Zhicheng Yang, Ruiyu Li, and Jiaya Jia. Prior guided feature enrichment network for few-shot segmentation. In *TPAMI*, 2020. 2, 3, 4, 5, 6, 7

[32] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. In *arXiv:1710.10903*, 2017. 2

[33] Haochen Wang, Xudong Zhang, Yutao Hu, Yandan Yang, Xianbin Cao, and Xiantong Zhen. Few-shot semantic segmentation with democratic attention networks. In *ECCV*, 2020. 2, 3, 6

[34] Kaixin Wang, Jun Hao Liew, Yingtian Zou, Daquan Zhou, and Jiashi Feng. Panet: Few-shot image semantic segmentation with prototype alignment. In *ICCV*, 2019. 2, 3, 5, 6

[35] Wenguan Wang, Xiankai Lu, Jianbing Shen, David J Crandall, and Ling Shao. Zero-shot video object segmentation via attentive graph neural networks. In *CVPR*, 2019. 2, 5

[36] Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *CVPR*, 2018. 2, 4

[37] Xiaolong Wang and Abhinav Gupta. Videos as space-time region graphs. In *ECCV*, 2018. 2

[38] Guo-Sen Xie, Li Liu, Xiaobo Jin, Fan Zhu, Zheng Zhang, Jie Qin, Yazhou Yao, and Ling Shao. Attentive region embedding network for zero-shot learning. In *CVPR*, 2019. 2

[39] Guo-Sen Xie, Li Liu, Fan Zhu, Fang Zhao, Zheng Zhang, Yazhou Yao, Jie Qin, and Ling Shao. Region graph embedding network for zero-shot learning. In *ECCV*, 2020. 2

[40] Guo-Sen Xie, Xu-Yao Zhang, Shuicheng Yan, and Cheng-Lin Liu. Sde: A novel selective, discriminative and equalizing feature representation for visual recognition. *IJCV*, 2017. 1

[41] Boyu Yang, Chang Liu, Bohao Li, Jianbin Jiao, and Qixiang Ye. Prototype mixture models for few-shot semantic segmentation. In *ECCV*, 2020. 2, 3, 6

[42] Xianghui Yang, Bairun Wang, Kaige Chen, Xinchi Zhou, Shuai Yi, Wanli Ouyang, and Luping Zhou. Brinet: Towards bridging the intra-class and inter-class gaps in one-shot segmentation. In *BMVC*, 2020. 2, 3, 4, 6, 7

[43] Yazhou Yao, Tao Chen, Guo-Sen Xie, Chuanyi Zhang, Fumin Shen, Qi Wu, Zhenmin Tang, and Jian Zhang. Non-salient region object mining for weakly supervised semantic segmentation. In *CVPR*, 2021. 1

[44] Fisher Yu and Vladlen Koltun. Multi-scale context aggregation by dilated convolutions. In *arXiv:1511.07122*, 2015. 2

[45] Qiang Zhai, Xin Li, Fan Yang, Chenglizhao Chen, Hong Cheng, and Deng-Ping Fan. Mutual graph learning for camouflaged object detection. In *CVPR*, 2021. 2

[46] 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 *CVPR*, 2019. 2, 3, 6, 7

[47] 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 *CVPR*, 2019. 3, 4, 6, 7, 8

[48] Xiaolin Zhang, Yunchao Wei, Yi Yang, and Thomas S Huang. Sg-one: Similarity guidance network for one-shot semantic segmentation. In *TCYB*, 2020. 2, 3, 6

[49] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. Pyramid scene parsing network. In *CVPR*, 2017. 1, 2, 4