Part-aware Prototype Network for Few-shot Semantic Segmentation

Yongfei Liu[1⋆], Xiangyi Zhang[1⋆], Songyang Zhang[1], and Xuming He[1,2]

[1]School of Information Science and Technology, ShanghaiTech University
[2]Shanghai Engineering Research Center of Intelligent Vision and Imaging
{liuyf3, zhangxy9, zhangsy1, hexm}@shanghaitech.edu.cn

Abstract. Few-shot semantic segmentation aims to learn to segment new object classes with only a few annotated examples, which has a wide range of real-world applications. Most existing methods either focus on the restrictive setting of one-way few-shot segmentation or suffer from incomplete coverage of object regions. In this paper, we propose a novel few-shot semantic segmentation framework based on the prototype representation. Our key idea is to decompose the holistic class representation into a set of part-aware prototypes, capable of capturing diverse and fine-grained object features. In addition, we propose to leverage unlabeled data to enrich our part-aware prototypes, resulting in better modeling of intra-class variations of semantic objects. We develop a novel graph neural network model to generate and enhance the proposed part-aware prototypes based on labeled and unlabeled images. Extensive experimental evaluations on two benchmarks show that our method outperforms the prior art with a sizable margin. [1]

1 Introduction

Semantic segmentation is a core task in modern computer vision with many potential applications ranging from autonomous navigation [4] to medical image understanding [6]. A particular challenge in deploying segmentation algorithms in real-world applications is to adapt to novel object classes efficiently in dynamic environments. Despite the remarkable success achieved by deep convolutional networks in semantic segmentation [19, 2, 5, 38], a notorious disadvantage of those supervised approaches is that they typically require thousands of pixel-wise labeled images, which are very costly to obtain. While much effort has been made to alleviate such burden on data annotation, such as weak supervision [15], most of them still rely on collecting large-sized datasets.

A promising strategy, inspired by human visual recognition [25], is to enable the algorithm to learn to segment a new object class with only a few annotated examples. Such a learning task, termed as *few-shot semantic segmentation*, has

⋆ Both authors contributed equally to the work. This work was supported by Shanghai NSF Grant (No. 18ZR1425100)

[1] Code is avilable at: https://github.com/Xiangyi1996/PPNet-PyTorch

attracted much attention recently [22, 3, 33, 21]. Most of those initial attempts adopt the metric-based meta-learning framework [32], in which they first match learned features from support and query images, and then decode the matching scores into final segmentation.

However, the existing matching-based methods often suffer from several drawbacks due to the challenging nature of semantic segmentation. First, some prior works [37, 36, 35] solely focus on the task of one-way few-shot segmentation. Their approaches employ dense pair-wise feature matching and specific decoding networks to generate segmentation, and hence it is non-trivial or computationally expensive to generalize to the multi-way setting. Second, other prototype-based methods [7, 27, 33] typically use a holistic representation for each semantic class, which is difficult to cope with diverse appearance in objects with different parts, poses or subcategories. More importantly, all those methods represent a semantic class based on a small support set, which is restrictive for capturing rich and fine-grained feature variations required for segmentation tasks.

In this work, we propose a novel prototype-based few-shot learning framework of semantic segmentation to tackle the aforementioned limitations. Our main idea is to enrich the prototype representations of semantic classes in two directions. First, we decompose the commonly used holistic prototype representation into a small set of part-aware prototypes, which is capable of capturing diverse and fine-grained object features and yields better spatial coverage in semantic object regions. Moreover, inspired by the prior work in image classification [24, 1], we incorporate a set of unlabeled images into our support set so that our part-aware prototypes can be learned from both labeled and unlabeled data source. This enables us to go beyond the restricted small support set and to better model the intra-class variation in object features. We refer to this new problem setting as semi-supervised few-shot semantic segmentation. Based on our new prototypes, we also design a simple and yet flexible matching strategy, which can be applied to either one-way or multi-way setting.

Specifically, we develop a deep neural network for the task of few-shot semantic segmentation, which consists of three main modules: an embedding network, a prototypes generation network and a part-aware mask generation network. Given a few-shot segmentation task, our embedding network module first computes a 2D conv feature map for each image. Taking as input all the feature maps, the prototype generation module extracts a set of part-aware representations of semantic classes from both labeled and unlabeled support images. To achieve this, we first cluster object features into a set of prototype candidates and then use a graph attention network to refine those prototypes using all the support data. Finally, the part-aware mask generation network fuses the score maps generated by matching the part-aware prototypes to the query image and predicts the semantic segments. We train our deep network using the meta-learning strategy with an augmented loss [34] that exploits the original semantic classes for efficient network learning.

We conduct extensive experiments evaluation on the PASCAL-5^i[3, 37] and COCO-20^i dataset[33, 20] to validate our few-shot semantic segmentation strat-

egy. The results show that our part-aware prototype learning outperforms the state of the art with a large margin. We also include the detailed ablation study in order to provide a better understanding of our method.

The main contribution of this work can be summarized as the following:

- We develop a flexible prototype-based method for few-shot semantic segmentation, achieving superior performances in one-way and multi-way setting.
- We propose a part-aware prototype representation for semantic classes, capable of encoding fine-grained object features for better segmentation.
- To better capture the intra-class variation, we leverage unlabeled data for semi-supervised prototype learning with a graph attention network.

2 Related Work

2.1 Few-shot Classification

Few-shot learning aims to learn a new concept representation from only a few annotated examples. Most of existing works can be categorized into metric-learning based [34, 28, 32], optimization-learning based [23, 8], and graph neural network [9, 18] based methods. Our work is inspired by the metric-learning based methods. In particular, Oriol et al. [32] propose to encode an input into an embedded feature and to perform a weighted nearest neighbor matching for classification. The prototypical network [28] aims to learn a metric space in which an input is classified according to its distance to class prototypes. Our work is in line with the prototypical network, but we adopt this idea in more challenging segmentation tasks, enjoying a simple design and yet high performance.

There have been several recent attempts aiming to improve the few-shot learning by incorporating a set of unlabeled data, referred to as semi-supervised few-shot learning [24, 16, 1]. Ren et al. [24] first try to leverage unlabeled data to refine the prototypes by Soft K-means. Ayyad et al.[1] introduced a consistency loss both in local and global for utilizing unlabeled data effectively. These methods are initially proposed for solving semi-supervised problems in few-shot classification regime and hence it is non-trivial to extend them to few-shot segmentation directly. We are the first to leverage unlabeled data in the challenging few-shot segmentation task for capturing the large intra-class variations.

2.2 Few-shot Semantic Segmentation

Few-shot semantic segmentation aims to segment semantic objects in an image with only a few annotated examples, and attracted much attention recently. The existing works can be largely grouped into two types: parametric matching-based methods [37, 36, 35, 20, 3] and prototype-based methods [27, 33]. A recent exception, MetaSegNet [29], adopts the optimization-based few-shot learning strategy and formulates few-shot segmentation as a pixel classification problem.

In the parametric-matching based methods, Shaban et al. [3] first develop a weight imprinting mechanism to generate the classification weight for few-shot

segmentation. Zhang et al. [36] propose to concatenate the holistic objects representation with query features in each spatial position and introduce a dense comparison module to estimate their prediction.The subsequent method, proposed by Zhang et al. [35], attends to foreground features for each query feature with a graph attention mechanism. These methods however mainly focus on the restrictive one-way few-shot setting and it is computationally expensive to generalize them to the multi-way setting.

Prototype-based methods conduct pixel-wise matching on query images with holistic prototypes of semantic classes. Wang et al. [33] propose to learn class-specific prototype representation by introducing the prototypes alignment regularization between support and query images. Siam et al. [27] adopt a novel multi-resolution prototype imprinting scheme for few-shot segmentation. All these prototype-based methods are limited by their holistic representations. To tackle this issue, we propose to decompose object representation into a small set of part-level features for modeling diverse object features at a fine-grained level.

2.3 Graph Neural Networks

Our work is related to learning deep networks on graph-structured data. The Graph Neural Networks are first proposed in [10, 26] which learn a feature representation via a recurrent message passing process. Graph convolutional networks are a natural generalization of convolutional neural networks to non-Euclidean graphs. Kipf et al. [14] introduce learning polynomials of the graph laplacian instead of computing eigenvectors to alleviate the computational bottleneck, and validated its effectiveness on semi-supervised learning. Velic et al. [31] incorporate the attention mechanism into the graph neural network to augment node representation with their contextual information. Garcia et al. [9] firstly introduce the graph neural network into the few-shot image classification. By contrast, our work employ graph neural network to learn a set of prototypes for the task of semantic segmentation.

3 Problem Setting

We consider the problem of few-shot semantic segmentation, which aims to learn to segment semantic objects from only a few annotated training images per class. To this end, we adopt a meta-learning strategy [32, 3] that builds a meta learner $\mathcal{M}$ to solve a family of few-shot semantic segmentation tasks $\mathcal{T} = \{T\}$ sampled from an underlying task distribution P_T.

Formally, each few-shot segmentation task T (also called an episode) is composed of a set of support data $\mathcal{S}$ with ground-truth masks and a set of query images $\mathcal{Q}$. In our semi-supervised few-shot semantic segmentation setting, the support data $\mathcal{S} = \{\mathcal{S}^l, \mathcal{S}^u\}$ where $\mathcal{S}^l$ and $\mathcal{S}^u$ are the annotated image-label pairs and unlabeled images, respectively. More specifically, for the C-way K-shot setting, the annotated support data consists of K image-label pairs from each class, denoted as $\mathcal{S}^l = \{(\mathbf{I}^l_{c,k}, \mathbf{Y}^l_{c,k})\}^{c\in\mathcal{C}_T}_{k=1,\dots,K}$, where $\{\mathbf{Y}^l_{c,k}\}$ are pixel-wise annotations,

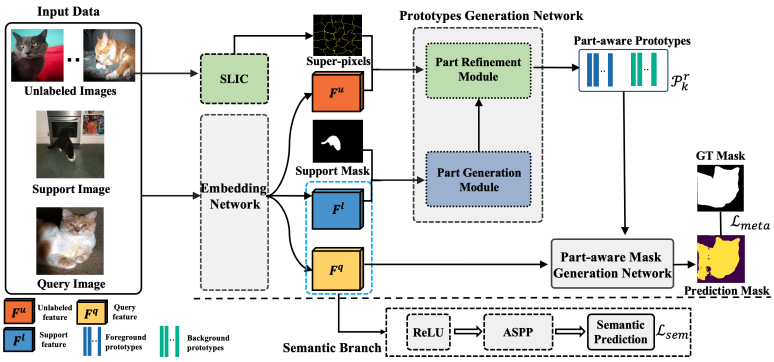

Fig. 1. Model Overview: For each task T, **Embedding Network** first aims to prepare convolutional feature maps of support, unlabeled and query images. **Prototypes Generation Network** then generates a set of part-aware prototypes by taking support and unlabeled image features as input. It consists of two submodules: **Part Generation Module** and **Part Refinement Module** (see below for details). Finally, the **Part-aware Mask Generation Network** performs segmentation on query features based on a set of part-aware prototypes. In addition, **Semantic Branch** generates mask predictions over the global semantic class space $\mathcal{C}^{tr}$.

$\mathcal{C}_T$ is the subset of class sets for the task T and $|\mathcal{C}_T| = C$. The unlabeled support images $\mathcal{S}^u = \{\mathbf{I}_i^u\}_{i=1}^{N_u}$ are randomly sampled from the semantic class set $\mathcal{C}$ with their class labels removed during training and inference. Similarly, the query set $\mathcal{Q} = \{(\mathbf{I}_j^q, \mathbf{Y}_j^q)\}_{j=1}^{N_q}$, contains N_q images from the class set $\mathcal{C}_T$ whose ground-truth annotations $\{\mathbf{Y}_j^q\}$ are provided during training but *unknown* in test.

The meta learner $\mathcal{M}$ aims to learn a functional mapping from the support set $\mathcal{S}$ and a query image $\mathbf{I}^q$ to its segmentation $\mathbf{Y}^q$ for all the tasks. To achieve this, we construct a training set of segmentation tasks $\mathcal{D}^{tr} = \{(\mathcal{S}_n, \mathcal{Q}_n)\}_{n=1}^{|\mathcal{D}^{tr}|}$ with a class set $\mathcal{C}^{tr}$, and train the meta learner episodically on the tasks in $\mathcal{D}^{tr}$. After the meta-training, the model $\mathcal{M}$ encodes the knowledge on how to perform segmentation on different semantic classes across tasks. We finally evaluate the learned model in a test set of tasks $\mathcal{D}^{te} = \{(\mathcal{S}_m, \mathcal{Q}_m)\}_{m=1}^{|\mathcal{D}^{te}|}$ whose class set $\mathcal{C}^{te}$ is non-overlapped with $\mathcal{C}^{tr}$.

4 Our Approach

In this work, we adopt a prototype-based few-shot learning framework to build a meta learner $\mathcal{M}$ for semantic segmentation. The main idea of our method is to capture the intra-class variation and fine-grained features of semantic classes by a new prototype representation. Specifically, we propose to decompose the commonly-used holistic representations of support objects into a set of part-aware prototypes for each class, and additionally utilize unlabeled data to enrich their representations.

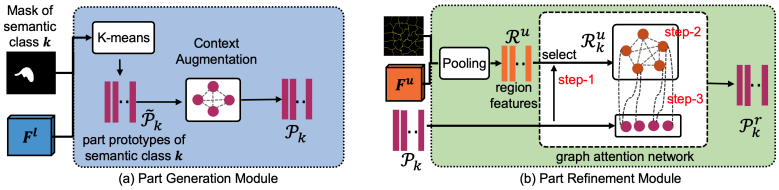

Fig. 2. Part Generation Module aims to generate the initial part-aware prototypes on support images and further incorporate with their global context of the same semantic class. **Part Refinement Module** further improves part-aware prototypes representation with unlabeled images features by a graph attention network.

To this end, we develop a deep graph network, as our meta learner, to encode such a new representation and to segment the query images. Our network consists of three main networks: an *embedding network* that computes convolutional feature maps for the images within a task (in Sec. 4.1); a *prototypes generation network* that extracts a set of part-aware prototypes from the labeled and unlabeled support images (in Sec. 4.2); and *a part-aware mask generation network* that generates the final semantic segmentation of the query images (in Sec. 4.3).

To train our meta model, we adopt a hybrid loss and introduce an auxiliary *semantic branch* that exploits the original semantic classes for efficient learning (in Sec. 4.4). We refer to our deep model as the **Part-aware Prototype Network (PPNet)**. An overview of our framework is illustrated in Fig.1 and we will introduce the model details in the remaining of this section.

4.1 Embedding Network

Given a task (or episode), the first module of our PPNet is an embedding network that extracts the convolutional feature maps of all images in the task. Following prior work [36, 35], we adopt ResNet [12] as our embedding network, and introduce the dilated convolution to enlarge the receptive field and preserve more spatial details. Formally, we denote the embedding network as f_{em}, and compute the feature maps of images in a task T as $\mathbf{F} = f_{\text{em}}(\mathbf{I})$, $\forall \mathbf{I} \in \mathcal{S} \cup \mathcal{Q}$. Here $\mathbf{F} \in \mathbb{R}^{H_f \times W_f \times n_{ch}}$, n_{ch} is the number of feature channels, and (H_f, W_f) is the height and width of the feature map. We also resize the annotation mask $\mathbf{Y}$ into the same spatial size as the feature map, denoted as $\mathbf{M} \in \mathbb{R}^{H_f \times W_f}$.

In the C-way K-shot setting, we reshape and group all the image features in the labeled support set $\mathcal{S}^l$ into $C+1$ subsets: $\mathcal{F}^l = \{\mathcal{F}^l_k, k = 0, 1, \cdots, C\}$, where 0 indicates background class and $\mathcal{F}^l_k$ contains all the features $\mathbf{f} \in \mathbb{R}^{n_{ch}}$ annotated with semantic class k. Similarly, we denote all the features in the unlabeled support set $\mathcal{S}^u$ as $\mathcal{F}^u$.

4.2 Prototypes Generation Network

Our second module, the prototypes generation network, aims to generate a set of discriminative part-aware prototypes for each class. For notation clarity, here

we focus on a single semantic class k. The module takes the image feature set $\mathcal{F}_k^l$ and $\mathcal{F}^u$ as input, and outputs the set of prototypes $\mathcal{P}_k$.

To this end, we introduce a graph neural network defined on the features, which computes the prototypes in two steps according to the different nature of the labeled and unlabeled support sets. Specifically, the network first extracts part-aware prototypes directly from the labeled support data $\mathcal{F}_k^l$ and then refines their representation by making use of the unlabeled data $\mathcal{F}^u$. As a result, the prototypes generation network consists of two submodules: a *Part Generation Module* and a *Part Refinement Module*, which are described in detail as following.

Part Generation with Labeled Data We first introduce the part generation module, which builds a set of part-aware prototypes from the labeled support set in order to capture fine-grained part-level variation in object regions.

Specifically, we denote the number of prototypes per class as N_p and the prototype set $\mathcal{P}_k = \{\mathbf{p}_i\}_{i=1}^{N_p}$, $\mathbf{p}_i \in \mathbb{R}^{n_{ch}}$. To define our prototypes, we first compute a data partition $\mathcal{G} = \{G_1, G_2, \cdots, G_{N_p}\}$ on the feature set $\mathcal{F}_k^l$ using the K-means clustering and then generate an initial set of prototypes $\tilde{\mathcal{P}}_k = \{\tilde{\mathbf{p}}_i\}_{i=1}^{N_p}$ with an average pooling layer as follows,

$$\tilde{\mathbf{p}}_i = \frac{1}{|G_i|} \sum_{j \in G_i} \mathbf{f}_j, \quad \mathbf{f}_j \in \mathcal{F}_k^l \tag{1}$$

We further incorporate a global context of the semantic class into the part-aware prototypes by augmenting each initial prototype with a context vector, which is estimated from other prototypes in the same class based on the attention mechanism [30]:

$$\mathbf{p}_i = \tilde{\mathbf{p}}_i + \lambda_p \sum_{j=1 \wedge j \neq i}^{N_p} \mu_{ij} \tilde{\mathbf{p}}_j, \quad \mu_{ij} = \frac{d(\tilde{\mathbf{p}}_i, \tilde{\mathbf{p}}_j)}{\sum_{j \neq i} d(\tilde{\mathbf{p}}_i, \tilde{\mathbf{p}}_j)} \tag{2}$$

where λ_p is a scaling parameter and μ_{ij} is the attention weight calculated with a similarity metric d, e.g., cosine similarity.

Part Refinement with Unlabeled Data: The second submodule, the part refinement module, aims to capture the intra-class variation of each semantic class by enriching the prototypes on additional unlabeled support images. However, exploiting the unlabeled data is challenging due to the fact that the set of unannotated image features $\mathcal{F}^u$ is much more noisy and in general has a much larger volume than the labeled set $\mathcal{F}_k^l$.

We tackle the above two problems by a grouping and pruning process, which yields a smaller and more relevant set of features $\mathcal{R}_k^u$ for class k. Based on $\mathcal{R}_k^u$, we then design a graph attention network to smooth the unlabeled features and to refine the part-aware prototypes by aggregating those features. Concretely, our refinement process includes the following three steps:

Step-1: *Relevant feature generation.* We first compute a region-level feature representation of unlabeled images by utilizing the idea of superpixel generation. Concretely, we apply SLIC [13] to all the unlabeled images and generate a set of groupings on $\mathcal{F}^u$. Denoting the groupings as $\mathcal{R} = \{R_1, R_2, \cdots, R_{N_r}\}$, we use the average pooling to produce a pool of region-level features $\mathcal{R}^u = \{\mathbf{r}_i\}_{i=1}^{N_r}$. We then select a set of relevant features for class k as follows:

$$\mathcal{R}_k^u = \{\mathbf{r}_j : \mathbf{r}_j \in \mathcal{R}^u \wedge \exists \mathbf{p}_i \in \mathcal{P}_k, d(\mathbf{p}_i, \mathbf{r}_j) > \sigma\} \tag{3}$$

where $d(\cdot, \cdot)$ is a cosine similarity function between prototype and feature, and σ is a threshold that determines the relevance of the features.

Step-2: *Unlabeled feature augmentation.* With the selected unlabeled features, the second step aims to enhance those region-level representations by incorporating contextual information in the unlabeled feature set. This allows us to encode both local and global cues of a semantic class.

Specifically, we build a fully-connected graph on the feature set $\mathcal{R}_k^u$ and use the following message passing function to compute the update $\tilde{\mathcal{R}}_k^u = \{\tilde{\mathbf{r}}_i\}_{i=1}^{|\tilde{\mathcal{R}}_k^u|}$:

$$\tilde{\mathbf{r}}_i = \mathbf{r}_i + h\left(\frac{1}{Z_i^u}\sum_{j=1 \wedge j \neq i}^{|\mathcal{R}_k^u|} d(\mathbf{r}_i, \mathbf{r}_j)\mathbf{W}\mathbf{r}_j\right) \tag{4}$$

where $\tilde{\mathbf{r}}_i$ represents the updated representation at node i, h is an element-wise activate function (e.g., ReLU). d is a similarity function encoding the relations between two feature vectors $\mathbf{r}_i$ and $\mathbf{r}_j$, and Z_i^u is a normalization factor for node i. $\mathbf{W} \in \mathbb{R}^{n_{ch} \times n_{ch}}$ is the weight matrix defining a linear mapping to encode the message from node j.

Step-3: *Part-aware prototype refinement.* Given the augmented unlabeled features, we refine the original part-aware prototypes with an attention strategy similar to the labeled one. We use the part-aware prototypes $\mathcal{P}_k$ as attention query to choose similar unlabeled features in $\tilde{\mathcal{R}}_k^u$ and aggregate them into $\mathcal{P}_k$:

$$\mathbf{p}_i^r = \mathbf{p}_i + \lambda_r \sum_{j=1}^{|\tilde{\mathcal{R}}_k^u|} \phi_{ij}\tilde{\mathbf{r}}_j, \quad \phi_{ij} = \frac{d(\mathbf{p}_i, \tilde{\mathbf{r}}_j)}{\sum_j d(\mathbf{p}_i, \tilde{\mathbf{r}}_j)} \tag{5}$$

where λ_r is a scaling parameter and ϕ_{ij} is the attention weight. The final refined prototype set for class k is denoted as $\mathcal{P}_k^r = \{\mathbf{p}_1^r, \mathbf{p}_2^r, \cdots, \mathbf{p}_{N_p}^r\}$.

4.3 Part-aware Mask Generation Network

Given the part-aware prototypes $\{\mathcal{P}_k^r\}_{k=0}^C$ of every semantic class and background in each task, we introduce a simple and yet flexible matching strategy to generate the semantic mask prediction on a query image $\mathbf{I}^q$. We denote its conv feature map as $\mathbf{F}^q$ and its feature column at location (m, n) as $\mathbf{f}_{m,n}^q$.

Specifically, we first generate a similarity score map for each part-aware prototype performing the *prototype-pixel* matching as follows,

$$\mathbf{S}_{k,j}(m, n) = d(\mathbf{f}_{m,n}^q, \mathbf{p}_j^r), \quad \mathbf{p}_j^r \in \mathcal{P}_k^r, \quad \mathbf{S}_{k,j} \in \mathbb{R}^{H_f \times W_f} \tag{6}$$

where d is the cosine similarity function and $\mathbf{S}_{k,j}(m,n)$ is the score at location (m,n). We then fuse together all the score maps from the class k by max-pooling and generate the output segmentation scores by concatenating score maps from all the classes:

$$\mathbf{S}_k^q = \text{MaxPool}(\{\mathbf{S}_{k,j}\}_{j=1}^{N_p}), \quad \hat{\mathbf{Y}}^q = \bigoplus \{\mathbf{S}_k^q\}_{k=0}^{C} \tag{7}$$

where $\bigoplus$ indicates concatenation. To generate the final segmentation, we upsample the score output $\hat{\mathbf{Y}}^q$ by bilinear interpolation and choose the class label with the highest score at each pixel location.

4.4 Model Training with Semantic Regularization

To estimate the parameters of proposed model, we train our PPNet in the meta-learning framework. Specifically, we adopt the standard cross-entropy loss to train our entire network on all the tasks in the training set $\mathcal{D}^{tr}$. Inspired by [33], we compute the cross-entropy loss on both support and query images. The loss for each task can be written as:

$$\mathcal{L}_{meta} = \mathcal{L}_{CE}(\hat{\mathbf{Y}}^q, \mathbf{Y}^q) + \mathcal{L}_{CE}(\hat{\mathbf{Y}}^l, \mathbf{Y}^l) \tag{8}$$

where $\mathcal{L}_{CE}$ is the cross-entropy function, $\mathbf{Y}^l, \hat{\mathbf{Y}}^l$ are the ground-truth and prediction mask for support image. We note that while our full model is not strictly differentiable w.r.t the embedding network thanks to the prototype clustering and candidate region selection, we are able to compute an approximate gradient by fixing the clustering and selection outcomes. This approximation works well empirically largely due to a well pre-trained embedding network.

In order to learn better visual representation for few-shot segmentation, we introduce another **semantic branch** [34] for computing a semantic loss defined on the global semantic class space C^{tr} (in contrast to C classes in individual tasks). To achieve this, we augment the network with an Atrous Spatial Pyramid Pooling module (ASPP) decoder to predict mask scores $\hat{\mathbf{Y}}_{sem}^q, \hat{\mathbf{Y}}_{sem}^l$ of support and query image respectively in the global class space C^{tr}, and compute the semantic loss as below,

$$\mathcal{L}_{sem} = \mathcal{L}_{CE}(\hat{\mathbf{Y}}_{sem}^q, \mathbf{Y}_{sem}^q) + \mathcal{L}_{CE}(\hat{\mathbf{Y}}_{sem}^l, \mathbf{Y}_{sem}^l) \tag{9}$$

Here $\mathbf{Y}_{sem}^q, \mathbf{Y}_{sem}^l$ are ground-truth masks defined over shared class space C^{tr}. The overall training loss for each task is:

$$\mathcal{L}_{full} = \mathcal{L}_{meta} + \beta \mathcal{L}_{sem} \tag{10}$$

where β is hyper-parameter to balance the weight of task loss and semantic loss.

5 Experiments

We evaluate our method on the task of few-shot semantic segmentation by conducting a set of experiments on two datasets, including PASCAL-5^i [3] and

COCO-20^i [33, 20]. In each dataset, we compare our approach with the state-of-the-art methods in terms of prediction accuracy.

Below we first introduce the implementation details and experimental configuration in Sec. 5.1. Then we present our experimental analysis on PASCAL-5^i dataset in Sec. 5.2, followed by our results on the COCO-20^i dataset in Sec. 5.3. We report comparisons of quantitative results and analysis on each dataset. Finally, we conduct a series of ablation studies to evaluate the importance of each component of the model in Sec. 5.4.

5.1 Experimental Configuration

Network Details: We adopt ResNet [12], initialized with weights pre-trained on ILSVRC [25], as feature extractor to compute the convolutional feature maps. In last two res-blocks, the strides of max-pooling are set as 1 and dilated convolutions are taken with dilation rate 2, 4 respectively. The last ReLU operation is removed for generating the prototypes. The input images are resized into a fixed resolution [417,417] and horizontal random flipping is used for data augmentation. For the part-aware prototypes network, the typical hyper-parameter of the parts is $N_p = 5$. In the part refinement module, we first generate N_r=100 candidate regions on unlabeled data, and select the relevant regions for each semantic class by setting similarity threshold σ as 0. In addition, λ_p in Eq. 2 and λ_r in Eq. 5 are set to 0.8 and 0.2 respectively, which control the proportion of parts and unlabeled information passed.

Training Setting: For the meta-training phase, the model is trained with the SGD optimizer, initial learning rate 5e-4, weight decay 1e-4 and momentum 0.9. We train 24k iterations in total, and decay the learning rate 10 times in 10k, 20k iteration respectively. The weight β of semantic loss $\mathcal{L}_{sem}$ is set as 0.5. At the testing phase, we average the mean-IoU of 5-runs [33] with different random seeds in each fold with each run containing 1000 tasks.

Baseline and Evaluation Metrics: We adopt ResNet-50 [12] as feature extractor in PANet [33] to be our baseline model, denoted as PANet*. Following previous works [3, 37, 22, 33, 35], mean-IoU and binary-IoU are adopted for model evaluation. Mean-IoU measures the averaged Intersection-over-Union (IoU) of all the classes. Binary-IoU[2]is calculated by treating all object classes as the foreground and averaging the IoU of foreground and background. In our experiments, we mainly focus on mean-IoU metrics for evaluation since it reflects the model generalization ability more precisely.

5.2 Experiments on PASCAL-5^i

Dataset: The PASCAL-5^i is introduced in [3, 37], which is created from PASCAL VOC 2012 dataset with SBD [11] augmentation. Specifically, the 20 classes

[2] We report Binary-IoU in supplementary material for a clear comparison with the previous works.

Table 1. Mean-IoU of 1-way on PASCAL-5^i. $*$ denotes the results implemented by ourselves. MS denotes the model evaluated with multi-scale inputs.[35, 36]. Red numbers denote the averaged mean-IoU over 4 folds.

Methods	MS	Backbone	1-shot					5-shot					#params
			fold-1	fold-2	fold-3	fold-4	mean	fold-1	fold-2	fold-3	fold-4	mean	
OSLSM [3]	x	VGG16	33.60	55.30	40.90	33.50	40.80	35.90	58.10	42.70	39.10	43.90	272.6M
co-FCN [22]	x	VGG16	31.67	50.60	44.90	32.40	41.10	37.50	50.00	44.10	33.90	41.40	34.20M
SG-one [37]	x	VGG16	40.20	58.40	48.40	38.40	46.30	41.90	58.60	48.60	39.40	47.10	19.00M
AMP [27]	x	VGG16	36.80	51.60	46.90	36.00	42.80	44.60	58.00	53.30	42.10	49.50	15.8M
PANet [33]	x	VGG16	42.30	58.00	51.10	41.20	48.10	51.80	64.60	59.80	46.50	55.70	14.7M
PANet* [33]	x	RN50	44.03	57.52	50.84	44.03	49.10	55.31	67.22	61.28	53.21	59.26	23.5M
PGNet* [35]	x	RN50	53.10	63.60	47.60	47.70	53.00	56.30	66.10	48.00	53.20	55.90	32.5M
FWB[20]	x	RN101	51.30	64.49	56.71	**52.24**	**56.19**	54.84	67.38	62.16	55.30	59.92	43.0M
CANet [36]	✓	RN50	52.50	65.90	51.30	51.90	55.40	55.50	67.80	51.90	53.20	57.10	36.35M
PGNet [35]	✓	RN50	**56.00**	**66.90**	50.60	50.40	56.00	57.70	68.70	52.90	54.60	58.50	32.5M
Ours(w/o $\mathcal{S}^u$)	x	RN50	47.83	58.75	53.80	45.63	51.50	58.39	67.83	64.88	56.73	61.96	23.5M
our	x	RN50	48.58	60.58	55.71	46.47	52.84	**58.85**	**68.28**	**66.77**	**57.98**	**62.97**	31.5M
Ours	x	RN101	52.71	62.82	**57.38**	47.74	55.16	**60.25**	**70.00**	**69.41**	**60.72**	**65.10**	50.5M

Table 2. Mean-IoU of 2-way on PSACAL-5^i. $*$ denotes our implementation. Red numbers denote the averaged mean-IoU over 4 folds.

Methods	Backbone	1-shot					5-shot				
		fold-1	fold-2	fold-3	fold-4	mean	fold-1	fold-2	fold-3	fold-4	mean
MetaSegNet [29]	RN9	-	-	-	-	-	41.9	41.75	46.31	43.63	43.30
PANet[33]	VGG16	-	-	-	-	45.1	-	-	-	-	53.10
PANet*[33]	RN50	42.82	56.28	48.72	45.53	48.33	54.65	64.80	57.61	54.94	58.00
Ours(**w/o** $\mathcal{S}^u$)	RN50	45.63	58.00	51.65	45.69	50.24	55.34	66.38	63.79	56.85	60.59
Ours	RN50	**47.36**	**58.34**	**52.71**	**48.18**	**51.65**	**55.54**	**67.26**	**64.36**	**58.02**	**61.30**

in PASCAL VOC is split into 4-folds evenly, each containing 5 categories. Models are trained on three folds and evaluated on the rest using cross-validation.

Quantitative Results: We compare the performance of our PPNet with the previous state-of-the-art methods. The detail results of 1-way setting are reported in Tab.1. With the ResNet-50 as the embedding network, our model achieves **61.96**% mean-IoU in 5-shot setting, which outperforms the state-of-the-art method with a sizable margin of **2.71**%. The performance can be further improved to **52.84**%(1-shot) and **62.97**%(5-shot) by refining the part prototypes with the unlabeled data. Compared with the PANet [33], our method achieves the considerable improvement in both 1-shot and 5-shot setting, which demonstrates the part-aware prototypes can provide a superior representation than the holistic prototype. Moreover, our method achieves the state-of-the-art performance at **65.10**% in 5-shot setting relied on the ResNet-101 backbone[3].

To investigate the effectiveness of our method on multi-way setting, a 2-way experiment is conducted and the results are reported in Tab.2. Our method can outperform the previous works both in with/without unlabeled data with a large margin. Our model can achieve **51.65**% and **61.30**% for 1-shot and 5-shot setting, which has **3.32**% and **3.30**% performance gain compared with

[3] We note that our 1-shot performance is affected by the limited representation power of the prototypes learned from a single support image while prior methods [35, 36] employ a complex Convnet decoder to exploit additional spatial smoothness prior.

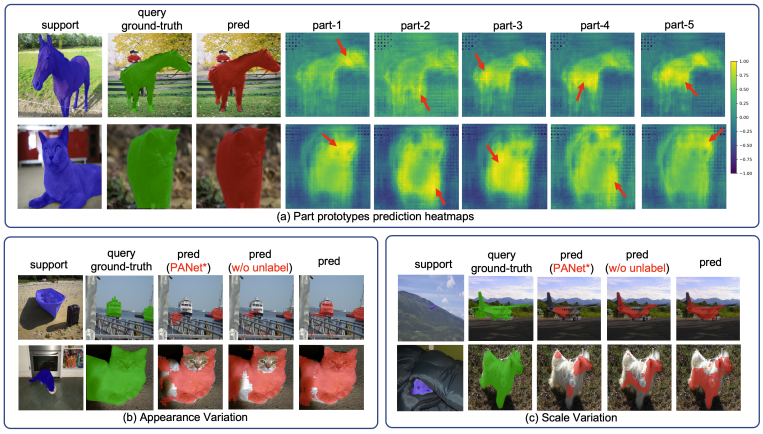

Fig. 3. Qualitative Visualization of 1-way 1-shot setting on PASCAL-5^i. (a) demonstrates the part-aware prototypes response heatmaps. The bright region denotes a high similarity between prototypes and query images. (b) and (c) show the capabilities of our model in coping with appearance and scale variation by utilizing unlabeled data. Red masks denote prediction results of our models. Blue and green masks denote the ground-truth of support and query images. See suppl. for more visualization results.

PANet*, respectively. The quantitative results indicate our PPNet can achieve the state-of-the-art in a more challenging setting.

Visualization Analysis: To better understand our part-aware prototype framework, we visualize the responding region of our part prototypes and the prediction results in Fig.3. The response heatmaps are presented in the column 4-8 of the (a). For example, given a support image(horse or cat), the part prototypes are corresponding to different body parts, which is capable of modeling one semantic class at a fine-grained level.

Moreover, our model can cope with the large appearance and scale variation between support and query images, which is illustrated in (b) and (c). Compared with the PANet*, our method can enhance the modeling capability with the part prototypes, and has a significant improvement on the segmentation prediction by utilizing the unlabeled images to better model the intra-class variations.

5.3 Experiments on COCO-20^i

Dataset: COCO-20^i [33, 20] is another more challenging benchmark built from MSCOCO [17]. Similar to PASCAl-5^i, MS-COCO dataset is divided into 4-folds with 20 categories in each fold. There are two splits of MSCOCO: we refer the data partition in [33] as *split-A* while the split in [20] as *split-B*. We mainly focus on *split-A* and also report the performance on *split-B*. Models are trained on three folds and evaluated on the rest with a cross-validation strategy.

Table 3. Mean-IoU results of 1-way on COCO-20^i split-A. Red numbers denote the averaged mean-IoU over 4 folds. * is our implementation

Methods	Split	Backbone	1-shot					5-shot				
			fold-1	fold-2	fold-3	fold-4	mean	fold-1	fold-2	fold-3	fold-4	mean
PANet [33]	A	VGG16	28.70	21.20	19.10	14.80	20.90	39.43	28.30	28.20	22.70	29.70
PANet* [33]	A	RN50	31.50	22.58	21.50	16.20	22.95	45.85	29.15	30.59	29.59	33.80
Ours(**w/o** $\mathcal{S}^u$)	A	RN50	34.53	25.44	24.33	18.57	25.71	48.30	30.90	35.65	30.20	36.24
Ours	A	RN50	**36.48**	**26.53**	**25.99**	**19.65**	**27.16**	**48.88**	**31.36**	**36.02**	**30.64**	**36.73**
FWB [20]	B	RN101	16.98	17.78	20.96	**28.85**	21.19	19.13	21.46	23.39	30.08	23.05
Ours	B	RN50	**28.09**	**30.84**	**29.49**	27.70	**29.03**	**38.97**	**40.81**	**37.07**	**37.28**	**38.53**

Table 4. Ablation Studies of 1-way 1-shot on COCO-20^i split-A in every fold. Red numbers denote the averaged mean-IoU over 4 folds.

Model	PAP	SEM	UD	1-shot				
				fold-1	fold-2	fold-3	fold-4	mean
Baseline (PANet*)	-	-	-	31.50	22.58	21.50	16.20	22.95
	✓(w/o context)	-	-	32.05	23.09	21.33	16.94	23.35
	✓	-	-	34.45	24.37	23.46	17.79	25.02
	✓	✓	-	34.53	25.44	24.33	18.57	25.71
	✓	✓	✓(w/o *step-2*)	36.02	24.45	25.82	19.07	26.34
Ours	✓	✓	✓	**36.48**	**26.53**	**25.99**	**19.65**	**27.16**

Quantitative Results: We report the performance of our method on this more challenging benchmark in Tab.3. Compared with the recent works [33, 20], our method can achieve the state-of-the-art performance in different splits that use the same type of embedding networks by a sizable margin. Compared with the baseline PANet*, the same performance improvement trends are shown in both setting. In split-B [20], our model is superior to FWB [20] nearly in every fold, except for fold-4 in 1-shot, achieving 29.03% in 1-shot and 38.53% in 5-shot.

5.4 Ablation Study

In this subsection, we conduct several experiments to evaluate the effectiveness of our model components on COCO-20^i *split*-A 1-way 1-shot setting.

Part-aware Prototypes (PAP): As in Tab. 4, by decomposing the holistic object representation [33, 37, 36] into a small set of part-level representations, the averaged mean-IoU is improved from 22.95% to **23.35%**. We further demonstrate the effectiveness of the global context used for augmenting part prototype(in Eq.2). The performance can achieve continuous improvement to 25.02%, which suggests that global semantic is important for part-level representations.

Semantic Branch (SEM): We also conduct experiments to validate the semantic branch [34]. It is evident that the semantic branch is able to improve the convergence and the final performance significantly, which indicates that the full PPNet exploits the semantic information efficiently.

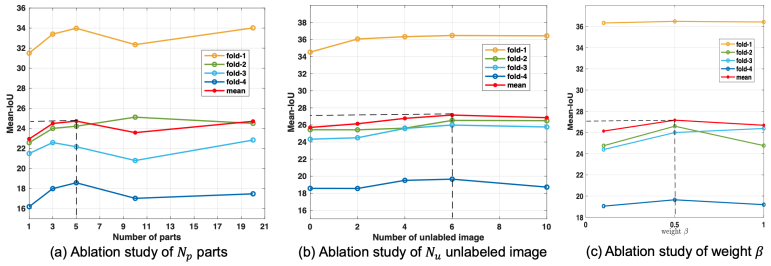

Fig. 4. Ablation studies of N_p parts, N_u unlabeled data and weight β for semantic loss on COCO-20^i split-A 1-way 1-shot.

Unlabel Data (UD): We also investigate the graph attention network for the exploitation of the unlabeled data. As discussed in the method, we propose to utilize the graph attention network to refine the part prototypes. We compare the performance of the full PPNet with the PPNet without the GNN module used in *step-2*. The performance demonstrates the effectiveness of the GNN, and our full PPNet can achieve 27.16% in terms of averaged mean-IoU over 4 folds.

Hyper-parameters N_p, N_u and β: We conduct several ablation studies to explore the influence of the hyper-parameters of our PPNet. We first investigate the part number N_p on 'baseline+PAP' model and plot the performance curve in Fig.4(a). In our experiments, the highest performance is achieved when N_p is 5 and 20 (red line) over 4 folds, and we set N_p=5 for computation efficiency. In our semi-supervised few-shot segmentation task, we also investigate the influence of the unlabeled image number N_u. In Fig.4(b), we can achieve the highest averaged mean-IoU over 4 folds (red line) with our full PPNet when N_u=6. In addition, we also investigate the weight β for semantic loss $\mathcal{L}_{sem}$ in our final model during training stage. As shown in Fig.4(c), the optimal value is β=0.5.

6 Conclusion

In this work, we presented a flexible prototype-based method for few-shot semantic segmentation. Our approach is able to capture diverse appearances of each semantic class. To achieve this, we proposed a part-aware prototypes representation to encode the fine-grained object features. In addition, we leveraged unlabeled data to capture the intra-class variations of the prototypes, where we introduce the first framework of semi-supervised few-shot semantic segmentation. We developed a novel graph neural network model to generate and enhance the part-aware prototypes based on support images with and without pixel-wise annotations. We evaluated our method on several few-shot segmentation benchmarks, in which our approach outperforms the prior works with a large margin, achieving the state-of-the-art performance.

References

1. Ayyad, A., Navab, N., Elhoseiny, M., Albarqouni, S.: Semi-supervised few-shot learning with local and global consistency. arXiv preprint arXiv (2019)
2. Badrinarayanan, V., Kendall, A., Cipolla, R.: Segnet: A deep convolutional encoder-decoder architecture for image segmentation. IEEE Transactions on Pattern Analysis and Machine Intelligence (2017)
3. Boots, Z.L.I.E.B., Shaban, A., Bansal, S.: One-shot learning for semantic segmentation. British Machine Vision Conference(BMVC) (2017)
4. Brabandere, B.D., Neven, D., Gool, L.V.: Semantic instance segmentation for autonomous driving. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPR) (2017)
5. Chen, L.C., Papandreou, G., Schroff, F., Adam, H.: Rethinking atrous convolution for semantic image segmentation. arXiv preprint arXiv (2017)
6. Chung, Y.A., Weng, W.H.: Learning deep representations of medical images using siamese cnns with application to content-based image retrieval. In: NIPS Machine Learning for Health Workshop(NIPS workshop) (2017)
7. Dong, N., Xing, E.: Few-shot semantic segmentation with prototype learning. In: British Machine Vision Conference(BMVC) (2018)
8. Finn, C., Abbeel, P., Levine, S.: Model-agnostic meta-learning for fast adaptation of deep networks. In: Proceedings of the 34th International Conference on Machine Learning(ICML) (2017)
9. Garcia, V., Bruna, J.: Few-shot learning with graph neural networks. arXiv preprint arXiv (2017)
10. Gori, M., Monfardini, G., Scarselli, F.: A new model for learning in graph domains. In: Proceedings. 2005 IEEE International Joint Conference on Neural Networks. IEEE (2005)
11. Hariharan, B., Arbeláez, P., Bourdev, L., Maji, S., Malik, J.: Semantic contours from inverse detectors. In: 2011 International Conference on Computer Vision(ICCV) (2011)
12. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition(CVPR) (2016)
13. Kim, A.: Fast slic. https://github.com/Algy/fast-slic
14. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907 (2016)
15. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. In: International Conference on Learning Representations (ICLR) (2017)
16. Li, X., Sun, Q., Liu, Y., Zhou, Q., Zheng, S., Chua, T.S., Schiele, B.: Learning to self-train for semi-supervised few-shot classification. In: Advances in Neural Information Processing Systems(NIPS) (2019)
17. Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft coco: Common objects in context. In: European conference on computer vision(ECCV) (2014)
18. Liu, Y., Lee, J., Park, M., Kim, S., Yang, E., Hwang, S.J., Yang, Y.: Learning to propagate labels: Transductive propagation network for few-shot learning. arXiv preprint arXiv (2018)
19. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition(CVPR) (2015)

20. Nguyen, K., Todorovic, S.: Feature weighting and boosting for few-shot segmentation. In: Proceedings of the IEEE International Conference on Computer Vision(ICCV) (2019)
21. Rakelly, K., Shelhamer, E., Darrell, T., Efros, A.A., Levine, S.: Few-shot segmentation propagation with guided networks. arXiv preprint (2018)
22. Rakelly, K., Shelhamer, E., Darrell, T., Efros, A., Levine, S.: Conditional networks for few-shot semantic segmentation (2018)
23. Ravi, S., Larochelle, H.: Optimization as a model for few-shot learning (2016)
24. Ren, M., Triantafillou, E., Ravi, S., Snell, J., Swersky, K., Tenenbaum, J.B., Larochelle, H., Zemel, R.S.: Meta-learning for semi-supervised few-shot classification. arXiv preprint arXiv (2018)
25. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., et al.: Imagenet large scale visual recognition challenge. International journal of computer vision (2015)
26. Scarselli, F., Gori, M., Tsoi, A.C., Hagenbuchner, M., Monfardini, G.: The graph neural network model. IEEE Transactions on Neural Networks (2008)
27. Siam, M., Oreshkin, B.: Adaptive masked weight imprinting for few-shot segmentation. arXiv preprint arXiv (2019)
28. Snell, J., Swersky, K., Zemel, R.: Prototypical networks for few-shot learning. In: Advances in neural information processing systems(NIPS) (2017)
29. Tian, P., Wu, Z., Qi, L., Wang, L., Shi, Y., Gao, Y.: Differentiable meta-learning model for few-shot semantic segmentation. arXiv preprint arXiv (2019)
30. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. In: Advances in neural information processing systems(NIPS) (2017)
31. Veličković, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., Bengio, Y.: Graph attention networks. arXiv preprint arXiv:1710.10903 (2017)
32. Vinyals, O., Blundell, C., Lillicrap, T., Wierstra, D., et al.: Matching networks for one shot learning. In: Advances in neural information processing systems(NIPS) (2016)
33. Wang, K., Liew, J.H., Zou, Y., Zhou, D., Feng, J.: Panet: Few-shot image semantic segmentation with prototype alignment. arXiv preprint arXiv (2019)
34. Yan, S., Zhang, S., He, X., et al.: A dual attention network with semantic embedding for few-shot learning. In: Proceedings of the AAAI Conference on Artificial Intelligence(AAAI) (2019)
35. Zhang, C., Lin, G., Liu, F., Guo, J., Wu, Q., Yao, R.: Pyramid graph networks with connection attentions for region-based one-shot semantic segmentation. In: Proceedings of the IEEE International Conference on Computer Vision(ICCV) (2019)
36. Zhang, C., Lin, G., Liu, F., Yao, R., Shen, C.: 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(CVPR) (2019)
37. Zhang, X., Wei, Y., Yang, Y., Huang, T.: Sg-one: Similarity guidance network for one-shot semantic segmentation. arXiv preprint arXiv (2018)
38. Zhao, H., Shi, J., Qi, X., Wang, X., Jia, J.: Pyramid scene parsing network. In: the IEEE Conference on Computer Vision and Pattern Recognition(CVPR) (2017)