Weakly-supervised Object Representation Learning for Few-shot Semantic Segmentation

Xiaowen Ying, Xin Li and Mooi Choo Chuah
Lehigh University
xiy517@lehigh.edu, xincoder@gmail.com, chuah@cse.lehigh.edu

Abstract

Training a semantic segmentation model requires large densely-annotated image datasets that are costly to obtain. Once the training is done, it is also difficult to add new object categories to such segmentation models. In this paper, we tackle the few-shot semantic segmentation problem, which aims to perform image segmentation task on unseen object categories merely based on one or a few support example(s). The key to solving this few-shot segmentation problem lies in effectively utilizing object information from support examples to separate target objects from the background in a query image. While existing methods typically generate object-level representations by averaging local features in support images, we demonstrate that such object representations are typically noisy and less distinguishing. To solve this problem, we design an object representation generator (ORG) module which can effectively aggregate local object features from support image(s) and produce better object-level representation. The ORG module can be embedded into the network and trained end-to-end in a weakly-supervised fashion without extra human annotation. We incorporate this design into a modified encoder-decoder network to present a powerful and efficient framework for few-shot semantic segmentation. Experimental results on the Pascal-VOC and MS-COCO datasets show that our approach achieves better performance compared to existing methods under both one-shot and five-shot settings.

1. Introduction

Semantic Segmentation is one of the fundamental tasks in Computer Vision. Given an input image, the algorithm is asked to assign a class label to each pixel. It is therefore also called dense prediction or pixel-wise classification. In recent years, the performance of semantic segmentation has been greatly improved [21] [2] [33] [9] by the success of deep Convolutional Neural Networks (CNNs) and the availability of large-scale datasets. However, data-labeling is costly, and pixel-wise annotations required for training semantic segmentation models are several times costlier than image-wise labeling. Besides, once the training is done, the model is limited to perform segmentation only on seen classes during training.

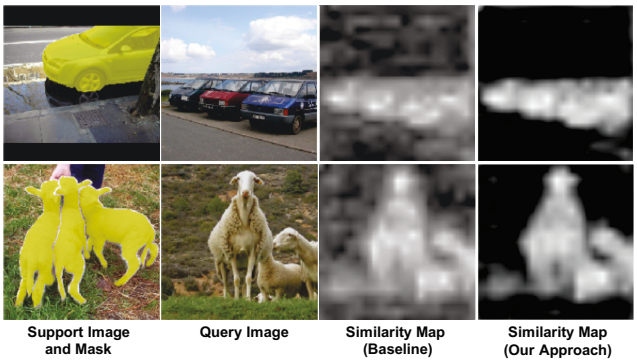

Figure 1. Illustration of the similarity maps produced by using different object representation approaches. The **first column** shows a support image with selected objects (**yellow mask**). The **second column** shows a query image which contains objects in target categories. **The third column** shows the pixel-wise similarity matching scores based on the object representation generated by the traditional Mask Average Pooling method. **The fourth column** shows the same similarity map but using the object representation generated by our approach. Better object representations are more distinguishable and hence can better differentiate target objects from backgrounds.

While learning-based systems have become the mainstream in many application scenarios such as face authentication [5] [19] [29], video surveillance systems [26] [11] [12] and autonomous driving [4] [14] [13], the performance of such systems heavily rely on huge amounts of training data. On the contrary, human-beings can quickly learn from a few examples. To conquer this challenge, the concept of few-shot learning was proposed by the community and quickly become a trending topic. The goal of few-shot learning is to develop learning schemes that help machine learning models to learn in data-constrained scenar-

ios. While much literature has studied the problem of few-shot classification [25] [23] [24], only a few works focused on the few-shot segmentation problem, which we believe is also worth studying due to the greater difficulties of obtaining pixel-wise annotations.

For few-shot segmentation, existing methods typically formulate this task as a feature matching problem. A common framework consists of a support branch and a query branch. Both branches first apply a convolutional neural network to extract feature maps from their corresponding input images. Then, the Masked Average Pooling (MAP) operation is applied to the support feature map to generate an object-level representation by pooling the local features over the foreground area specified by the support mask. Finally, this object representation is used to locate target objects in the query image, typically achieved by pixel-wise similarity comparison between query local features and the object representation.

An obvious drawback in the above pipeline is that the object representation produced by the MAP operation might not be able to represent the object well. For example, a car is composed of many parts, and local features for wheels may appear differently than those local features for windows. Simply pooling over the foreground features may result in a noisy and non-discriminating representation, which further increases the difficulties to locate target objects in the query image. To solve this problem, we propose an Object Representation Generator (ORG) module which learns to produce higher-level and better-quality object representation. The ORG module can be integrated into the network and trained end-to-end in a weakly-supervised fashion. Our qualitative and quantitative results show that object representations generated using our approach are more distinguishable and less noisy. Figure 1 illustrates some example similarity maps obtained using the traditional MAP operation vs. our ORG module.

We incorporate this design into an encoder-decoder network with several additional modifications to create a powerful and efficient framework for end-to-end few-shot semantic segmentation. Our framework consists of an Attention Branch and a Segmentation Branch. The Attention Branch combines the two-branch pipeline and our ORG module to produce an attention map that will be provided to the Segmentation Branch to further refine and generate the final predictions. More details of the framework are introduced in Section 4.

We evaluate our approach on PASCAL-5^i [22] and COCO-20^i [17] benchmarks under both one-shot and five-shot settings. Experimental results show that our scheme performs better than existing methods on both benchmarks which demonstrates the effectiveness of our approach.

Our contributions can be summarized as follows:

1. We propose a novel Object Representation Generator module that effectively aggregates local features and produces better object-level representations for few-shot semantic segmentation.

2. We design an effective training scheme for the ORG module to enable end-to-end training with the network in a weakly-supervised fashion.

3. We incorporate the proposed object representation approach into a modified encoder-decoder network to present a powerful and efficient end-to-end few-shot semantic segmentation network.

4. Experimental results show that our model achieves better performance compared to existing state-of-the-art methods on both PASCAL-5^i and COCO-20^i benchmarks.

2. Related Work

In this section, we review prior works related to semantic segmentation, few-shot learning, and few-shot semantic segmentation.

Semantic Segmentation. Semantic segmentation is the task of assigning a class label to each pixel. FCN [16] was the first deep learning framework for semantic segmentation. This framework replaces the fully connected layers in a typical classification model with convolution layers to support pixel-wise classification. The following works were more or less based on this framework. UNet [21] proposed to use a symmetric encoder-decoder network and found that skip-connections can significantly improve the quality of segmentation. Chen et al. [1] proposed to use atrous (dilated) convolution to maintain a large feature map while increasing the receptive field. Zhao et al. [33] proposed a pyramid pooling module that consists of several parallel convolution branches with different receptive fields to capture multi-scale information. All of the above methods require large-amount of densely-annotated images for supervision, and it is difficult to add new categories to a trained model.

Few-shot Learning. Few-shot learning is a newly emerging topic that aims to quickly learn a new task from limited labeled training examples. Within the scope of computer vision, few-shot learning for image classification is the most intensively studied task [7] [25] [23] [24], while little attention has been paid to other tasks such as object detection and semantic segmentation. Although semantic segmentation can be seen as a dense classification problem, it is not trivial to adapt existing few-shot classification methods to this problem due to the highly unbalanced data points and the existence of local relationships between neighboring pixels.

Few-shot Semantic Segmentation. OSLSM [22] was the first work on few-shot segmentation. Their method

directly predicts the weight of the dense-classifier based on support images. They also created a dataset, namely Pascal-5^i for few-shot segmentation which has become the most used benchmark for evaluating few-shot segmentation methods. Subsequent works on few-shot semantic segmentation are typically based on a two-branch comparison framework, which can be seen as an extension of metric-learning methods in few-shot image classification. PANet [27] proposed a prototype alignment regularization term to better exploit the available information from support images. CANet [31] designed a dense comparison module to implicitly learn the relationship between local features. They also developed an iterative refinement module to refine the raw prediction results. Nguyen et al. [17] proposed to include a relevance term that re-weights the support features to produce more discriminative features. They also proposed an approach to improve the quality of support feature via back-propagation during inference. Yang et al. [28] proposed a new local-transformation module that directly computes the similarity score between every single pair of local features in support and query image.

Comparisons with recent similar works. Our pipeline is similar to [28] in the sense that they also first predict an attention map as raw prediction and use a deep decoder to generate the final results. However, our approach is essentially different from their method in the way of producing this similarity attention map. The main difference is that their method does not produce object-level representations, instead, they first calculate the affinity matrix from local feature pairs between support and query images, and use such pair-wise similarity matrix to locate the target object in the query image. However, we argue that local features matching are less effective in the few-shot segmentation setting, in which the query and support images are not from the same video and typically look very different. In contrast, our approach focuses on first generating a better object-level representation, then using this high-quality object representation to find the target object in the query image.

The framework proposed by [17] uses the traditional MAP operation to generate object-level representation. However, their framework includes a guided inference procedure to refine the object representation via back-propagation during inference, which is similar to the motivation of our ORG module training scheme. However, this approach may result in the object representation over-fitted to a particular support example and hence does not generalize well, especially when the target object looks very different in support and query image. In contrast, the object representation produced by our ORG module generalizes better as it learns to produce better object representation by training end-to-end with the network on the entire dataset, which is also evidenced by our experimental results. Besides, their guided inference procedure requires extra gradient calculations and back-propagation during inference, while our method can directly generate prediction through a single pass.

3. Task Setting

The task setting of few-shot semantic segmentation is extended from few-shot classification, in which we aim to train a model that can quickly adapt to new tasks with few examples.

In this task, we are given a densely annotated training set $\mathcal{D}_{train}$ which consists of objects in base categories C_{train}. We are asked to train a model based on the training set and evaluate it on a testing set $\mathcal{D}_{test}$ which consists of objects in novel categories C_{test}, i.e. $\mathcal{C}_{train} \cap \mathcal{C}_{test} = \emptyset$. The testing set D_{test} is specifically constructed in an episodic form — for a K-shot learning task, each episode $e_i = \{(S_i, Q_i)\}$ consists of a support set $S_i = \{(x_s^k, y_s^k), k \in [1...K]\}_i$ and a query set $Q_i = \{(x_q, y_q)\}_i$, where x_s^k and y_s^k are the k^{th} support image and its corresponding object mask, respectively. x_q and y_q are the query image and the ground truth, respectively. During each testing episode, the model is asked to perform segmentation on x_q based on the object information in x_s^k specified by y_s^k.

4. Proposed Method

Our overall framework consists of an Attention Branch and a Segmentation Branch as illustrated in Figure 2. The Attention Branch is designed by combining our ORG module and an existing two-branch pipeline, which is described in Section 4.1. The Segmentation Branch is a modified encoder-decoder network which is described in Section 4.2. The detail of ORG module and its training scheme is described in Section 4.3.

4.1. Attention Branch

The Attention Branch is designed by incorporating our ORG module with an existing two-branch pipeline. We first use a backbone CNN to extract feature maps F_s and F_q from both support and query images, respectively. Since prior study on few-shot segmentation [31] has demonstrated that middle-level features generalize better to unseen classes, we directly utilize their conclusion and use the first three blocks in ResNet as our backbone CNN. Similar to prior works, we use ImageNet pretrained weights to initialize the backbone and do not update it during training. F_s and the down-scaled support mask are then fed into the proposed Object Representation Generator (ORG) module (described in Section 4.3) to generate an object representation vector V_s that represents the object feature.

To further generate the attention map on a query image, a common approach is to calculate the similarity between V_s and every local feature in F_q. However, since the back-

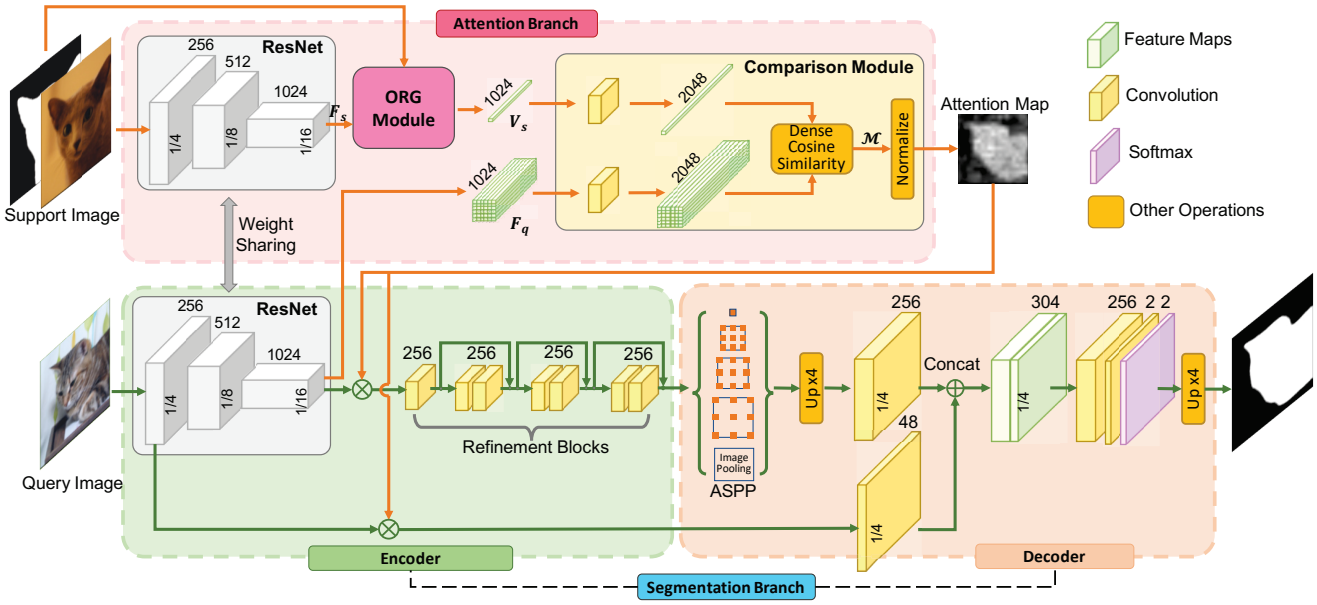

Figure 2. The proposed model architecture.

bone was pretrained on ImageNet and not updated during training, it may not be appropriate to directly use the extracted feature to calculate the distance/similarity without further transformation. Therefore, we map V_s and F_q to a higher-dimensional space (from 1024 to 2048) before calculating the cosine similarity (illustrated in the Comparison Module in Figure 2). This mapping is approximated with a single layer neural network, which is implemented as a point-wise convolution layer. The dense cosine similarity operation can be efficiently implemented via matrix multiplication, hence it does not incur much computation overhead. Our final attention map A is then produced by normalizing the similarity map $\mathcal{M}$ into the range $[0, 1]$.

4.2. Segmentation Branch

Our Segmentation Branch is designed based on the encoder-decoder structure from DeepLabV3+ [3] with several modifications. In the Encoder part, we first multiply the attention map generated by the Attention Branch with the query feature map F_q extracted by the backbone CNN. Considering that the attention map is generated by a point-wise cosine similarity operation and hence is lacking context information, we add several Refinement Blocks after the feature extractor to refine the feature maps. Each Refinement Block consists of two convolution layers with a residual connection that does not change the spatial size of the feature maps but can incorporate context information into each local feature from neighboring pixels.

In the Decoder part, the Atrous Spatial Pyramid Pooling (ASPP) module is first applied to the high-level feature to capture multi-scale information. The resulting feature map is then bilinearly upsampled by a factor of 4 and then concatenated with the low-level feature via an attentive skip-connection to recover object details. The attentive-skip connection provides low-level information (also multiplied by the attention map) to the decoder which helps to recover details in the final prediction. Eventually, the concatenated features are fed into several convolution layers to produce the final prediction.

4.3. Object Representation Generator

We proposed an Object Representation Generator (ORG) module that learns to generate better object-level representation. This module is designed to be jointly trained with the entire network in a weakly-supervised fashion without extra human annotations.

Architecture. As illustrated in Figure 3(a), the ORG module consists of several consecutive fully-convolutional blocks with large kernel-size to quickly reduce the spatial dimension and eventually produces a feature vector representing the target object. Considering that the number of channels of the input feature maps F_s is very large (1024 in our case), directly applying convolution layers will result in too many learnable parameters (e.g. $1024 \times 5 \times 5 \times 1024$ parameters for a single 5×5 convolution layer without the bias term). Having too many parameters will increase the size of the model and make this module more difficult to train. To solve this problem, we use a design similar to the bottleneck block in ResNet to construct our convolution block. A convolution block in ORG consists of 3 consecutive layers — a 1×1 convolution layer to reduce the number of channels of the feature map; a convolution layer with a large kernel to

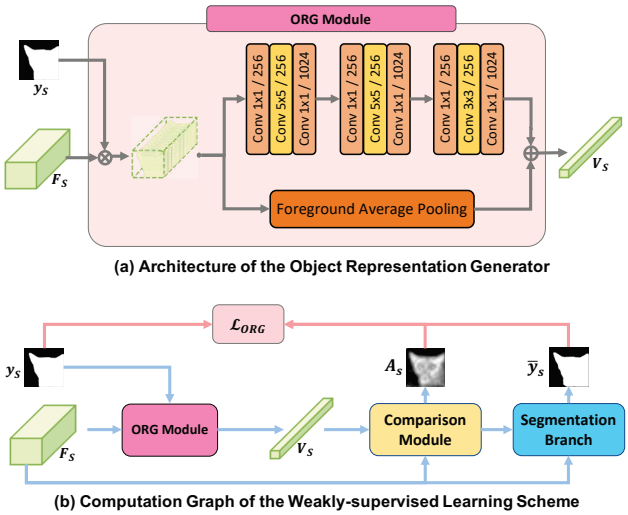

Figure 3. The architecture of the ORG module and the proposed weakly-supervised learning scheme.

reduce the spatial dimension; and another 1×1 convolution layer to increase the number of channels to its original size.

Instead of directly learning the object representation without any prior knowledge, we also add a parallel branch which does the foreground average pooling operation and fuse the results from both branches. Note that this parallel branch does not involve trainable parameters and can be seen as a regularization term. We found that adding this parallel branch greatly reduces the training difficulty and produces more stable results.

Weakly-supervised Object Representation Learning. The major difficulty of learning object representation is that we cannot provide explicit supervision to the output of the ORG module during training, as the quality of the generated object representation cannot be directly evaluated. Without any constraint, the training may not converge or could produce unexpected results. We solve this problem by proposing a weakly-supervised training scheme for the ORG module. Figure 3(b) illustrates the computation graph of the proposed training scheme. During training, we first feed the support feature map F_s and support object mask y_s to the ORG module to produce an object representation V_s. We then feed V_s and F_s to the Comparison Module to generate a similarity attention map A_s for the support image itself. The generated A_s is further passed to the subsequent Segmentation Branch to produce a final segmentation result $\bar{y}_s$ for the current support image. Finally, the training process is supervised by the following loss function $\mathcal{L}_{ORG}$:

$$\mathcal{L}_{ORG} = BCE(A_s, y_s) + CE(\bar{y}_s, y_s) \quad (1)$$

Intuitively, $\mathcal{L}_{ORG}$ encourages the ORG module to keep improving the quality of object representation, such that it can be used to generate a better segmentation result for the support image itself. While we cannot directly evaluate the quality of the object representation, this training scheme enables the implicit optimization of the ORG module in a weakly-supervised fashion. It is worth noting that our Comparison Module also has trainable parameters, and they are optimized together with the ORG module during this process. This gives our Comparison Module the ability to perform feature matching in a higher-level feature space, which further improves the ability of the ORG module to produce higher-level and more expressive object representations.

5. Experiments

5.1. Implementation Details

Episodic Training. To train our model to segment unseen classes based on a few support examples, we use the episodic training paradigm to mimic the testing protocol during training similar to prior works. For a K-shot learning task, each training episode is constructed by sampling 1) a query $q = (x_q, y_q)$ where x_q is the query image and y_q is its ground truth binary mask; and 2) a support set $S = \{(x_s^k, y_s^k), k \in [1...K]\}$ where x_s^k is the kth support image and y_s^k is its ground truth binary mask.

Loss Functions. We employ the widely used pixel-wise cross-entropy loss as our main loss function for the final prediction. Since the intermediate attention map A_q can be seen as a raw prediction, an auxiliary loss is also applied to A_q which is defined as the binary cross-entropy between the attention map and the ground truth. The final loss term is the weighted sum of the main loss, auxiliary attention loss, and the aforementioned $\mathcal{L}_{ORG}$, with the weight of the main loss term being 1 and the remaining terms being 0.8.

Extension to K-shots setting. When there is more than one support example during inference, we simply average the attention map generated from all support examples and feed this averaged attention map to the segmentation branch to produce the final result. While some recent approaches [31] [17] employ specially designed K-shot mechanisms to boost the performance under K-shots setting, we show that our results are still better than other approaches even with this simple averaging strategy. We plan to explore the potential of incorporating a better K-shot mechanism to further improves our K-shots results in our future works.

Evaluation Metrics. There are two different evaluation metrics used in prior works: Mean IoU and FB-IoU. The Mean IoU is calculated by averaging the per-class Intersection-over-Union (IoU), while the FB-IoU ignores the object categories and simply computes the mean of the foreground IoU and background IoU. In this task, Mean IoU is generally considered a better metric because of two reasons: 1) FB-IoU may be biased to some categories with more examples; 2) FB-IoU cannot properly measure the scenario when foreground objects are very small, e.g., if the

Dataset	Test Categories
Pascal-5^0	Aeroplane, Bicycle, Bird, Boat, Bottle
Pascal-5^1	Bus, Car, Cat, Chair, Cow
Pascal-5^2	Dining Table, Dog, Horse, Motorbike, Person
Pascal-5^3	Potted Plant, Sheep, Sofa, Train, TV/Monitor
COCO-20^0	Person, Airplane, Boat, Park Meter, Dog, Elephant, Backpack, Suitcase, Sports Ball, Skateboard, Wine Glass, Spoon, Sandwich, Hot Dog, Chair, Dining Table, Mouse, Microwave, Fridge, Scissors
COCO-20^1	Bicycle, Bus, Traffic Light, Bench, Horse, Bear, Umbrella, Frisbee, Kite, Surfboard, Cup, Bowl, Orange, Pizza, Couch, Toilet, Remote, Oven, Book, Teddy
COCO-20^2	Car, Train, Fire Hydrant, Bird, Sheep, Zebra, Handbag, Skis, Baseball Bat, Tennis Racket, Fork, Banana, Broccoli, Donut, Potted Plant, TV, Keyboard, Toaster, Clock, Hairdrier
COCO-20^3	Motorcycle, Truck, Stop Sign, Cat, Cow, Giraffe, Tie, Snowboard, Baseball Glove, Bottle, Knife, Apple, Carrot, Cake, Bed, Laptop, Cellphone, Sink, Vase, Toothbrush

Table 1. Summary of testing object categories used in each fold for both benchmarks.

model mistakenly predicts every pixel as background, the FB-IoU is still very high. Therefore, in this paper, we use Mean IoU as our evaluation metric as is done in most of the prior works.

To study the impact of different backbone, we provide results with both ResNet50 and ResNet101 as our backbone CNN. We follow the experiment setting in [31] to resize the input images to 321×321 and additionally experiment with 513×513 to study the impact of different input sizes. When the input size is set to 513×513, the kernel size of the last convolution block in the ORG module is changed to 6×6 to reduce the spatial dimension to 1. The dilation rates in Atrous Spatial Pyramid Pooling (ASPP) are set to [1, 6, 12, 18]. We preprocess the input using the same normalization as in the ImageNet dataset. Data augmentation techniques including random scale crop, random horizontal flip, and random gaussian blur are also applied during training.

We implement our method using the PyTorch [18] library. Our network is trained end-to-end on an Nvidia GTX 1080Ti GPU using the episodic training scheme. We use the AMSGrad [20] variant of Adam [10] optimizer to train the model for 200 epochs with batch size of 48. The initial learning rate is set to 1×10^{-5}, and the weight decay is set to 1×10^{-6}.

5.2. Dataset

We evaluate our scheme on two benchmarks for few-shot segmentation — PASCAL-5^i and COCO-20^i.

PASCAL-5^i was first proposed in the OSLSM paper [22]. It is created based on the PASCAL-VOC 2012 [6] and extra annotations from SDS [8]. In PASCAL-5^i, 20 categories in the original PASCAL-VOC dataset are evenly divided into 4 splits for 4-fold cross-validation. Each fold consists of 1 split for testing and the other 3 splits for training.

During testing, prior works typically use 1000 randomly sampled support-query pairs to evaluate the model. However, we found that the randomly generated testing set has varying levels of segmentation difficulties since some examples are easier to segment than others. To make fair comparisons, we evaluate our model on the same testing list as used by OSLSM [22] who first proposed the benchmark. This is done by running their code with the same random seed they used and export the pair list produced by their code.

COCO-20^i is a new benchmark that was recently proposed by Nguyen et al. [17]. It is created using a similar setting as PASCAL-5^i but with images and annotations from MS-COCO 2014 [15] dataset. In this benchmark, 80 categories in the original MS-COCO 2014 dataset are evenly divided into 4 splits for 4-fold cross-validation. Each fold consists of 20 categories for testing and the remaining 60 categories for training. This benchmark also uses 1000 support-query pairs for testing in each split.

COCO-20^i is considered more challenging not only because it has more categories but also due to the noisy annotations in the MS-COCO dataset which increases the learning difficulties.

We summarize the test object categories used in each fold for both benchmarks in Table 1 for references.

5.3. Experiments on PASCAL-5^i

While it has been a consensus that deeper backbones and larger input sizes can generally improve the performance of a semantic segmentation model, we realized that many prior works ignored the impact of these factors when making comparisons. Thus, we manually gather information from prior works and list relevant information in our comparisons for reference.

We compare our results with state-of-the-art methods under both 1-shot and 5-shots settings in Table 2. We can see that our approach consistently achieves better performance compared to all existing methods in terms of the mean over 4-folds cross-validation which demonstrates the effectiveness of our approach.

The results of LT [28] in Table 2 is reproduced by ourselves and different from the number reported in their paper. This is because the implementation of Mean IoU in their source code is different from the common definition (we have verified the issue with the authors). Therefore, we retrain their model and report the 1-shot results in Table 2 for reference. If we directly evaluate our best model using their implementation of Mean IoU, our 1-shot performance using ResNet-50 backbone is 57.2% which is still comparable with the number reported in their paper (57.0%).

CANet [31] and PGNet [30] report their results using

				1-Shot					5-Shots				
Index	Method	Backbone	Input Size	Fold 0	Fold 1	Fold 2	Fold 3	**Mean**	Fold 0	Fold 1	Fold 2	Fold 3	**Mean**
1	OSLSM [22]	VGG-16	224×224	33.6	55.3	40.9	33.5	40.8	35.9	58.1	42.7	39.1	43.9
2	SG-One [32]	VGG-16	–	40.2	58.4	48.4	38.4	46.3	41.9	58.6	48.6	39.4	47.1
3	PANet [27]	VGG-16	417×417	42.3	58.0	51.1	41.2	48.1	51.8	64.6	59.8	46.5	55.7
4	FWB [17]	VGG-16	512×512	47.0	59.6	52.6	48.3	51.9	50.9	62.9	56.5	50.1	55.1
5	CANet [31]	ResNet50	321×321	49.7	65.0	49.8	51.5	54.0	53.7	66.6	51.5	51.8	55.9
6	LT † [28]	ResNet50	320×320	50.2	65.4	**54.9**	49.4	55.0	–	–	–	–	–
7	**Ours**	ResNet50	321×321	**52.6**	**65.8**	54.7	**52.1**	**56.3**	**57.2**	**67.8**	**57.5**	**56.2**	**59.7**
8	CANet (MS) [31]	ResNet50	321×321	52.5	65.9	51.3	51.9	55.4	55.5	67.8	51.9	53.2	57.1
9	PGNet (MS) [30]	ResNet50	–	**56.0**	**66.9**	50.6	50.4	56.0	57.7	**68.7**	52.9	54.6	58.5
10	**Ours (MS)**	ResNet50	321×321	53.2	66.2	**54.7**	**53.4**	**56.9**	**58.0**	68.0	**57.7**	**57.6**	**60.3**
11	FWB [17]	ResNet101	512×512	51.3	64.5	**56.7**	52.2	56.2	54.8	67.4	**62.2**	55.3	59.9
12	**Ours**	ResNet101	321×321	55.4	67.6	53.4	51.5	57.0	58.7	69.7	55.8	56.6	60.2
13	**Ours**	ResNet101	513×513	**55.7**	**68.5**	54.7	**53.2**	**58.0**	**60.8**	**70.6**	57.0	**57.5**	**61.5**

Table 2. Experimental results on Pascal-5^i dataset under Mean IoU metric. LT †: results are reproduced by ourselves due to the evaluation issue in the original paper (see section 5.3 for more details).

			1-Shot					5-Shots				
Method	Backbone	Input Size	Fold 0	Fold 1	Fold 2	Fold 3	Mean	Fold 0	Fold 1	Fold 2	Fold 3	Mean
FWB [17]	ResNet101	512×512	17.0	18.0	21.0	**28.9**	21.2	19.1	21.5	24.0	30.1	23.7
Ours	ResNet101	513×513	**25.7**	**27.1**	**28.5**	25.6	**26.7**	**28.3**	**31.9**	**35.5**	**31.2**	**31.7**

Table 3. Experimental results on COCO-20^i dataset under Mean IoU metric.

multi-scale testing in their original papers while none of the other approaches on Table 2 uses multi-scale testing. To fairly compare with these two approaches, we also report our performance with multi-scale testing according to their configurations, marked with (MS) in the table. Remarkably, our performance without multi-scale testing is even better than their performances with multi-scale testing (comparing results 8, 9, and 10 in Table 2).

While using ResNet101 as the backbone, our performance with smaller input resolution is better than existing methods with larger input resolution (comparing results 11 and 12 in Table 2). It is worth mentioning that FBW [17] uses a more complicated K-shots scheme to boost up their 5-shots results while we employ a simple aforementioned average strategy and achieves 61.5% Mean IoU (result 13). For reference, they also report their 5-shots performance using average strategy in their paper which is 57.8%.

5.4. Experiments on COCO-20^i

Since COCO-20^i benchmark is a relatively new benchmark, at this point, only FBW [17] has available results on this benchmark for us to compare. Thus, we compare the performance of our approach with FBW [17] under both one-shot and five-shot settings in Table 3. We can see that our method achieves significant improvements on this dataset under both settings. Compared to the original FWB paper, we improved the Mean IoU by 25.9% and 33.7% under one-shot and five-shot settings, respectively, which further demonstrates the effectiveness of our method on a more challenging dataset. It is worth highlighting that our one-shot performance is even better than the five-shot performance of FWB.

5.5. Qualitative Comparison

We compare the qualitative results of our method with two state-of-the-art methods that have source code accessible — CANet [31] and LT [28]. Both of their models are reproduced by ourselves using their source code, denoted as CANet* and LT*, respectively. We use these three models to generate segmentation results on images from PASCAL-5^i dataset and visualize example results in Figure 4. We can see that our model consistently generates more accurate and stable results compared to existing methods. The last column in Figure 4 shows an interesting case where the query image does not include target objects. Our model can successfully handle this case while two other methods still produce some false-positive masks. This shows that the object representation generated by our approach is more discriminative so that our model is more confident to reject irrelevant objects in the background.

5.6. Ablation Study

In this section, we analyze the contributions of our key components to the final performance. We implement several variants of our model with some components removed and compare the results in Table 4. In Table 4, MAP denotes a variant of our model where we replace the ORG module with the traditional Mask Average Pooling operation; ORG

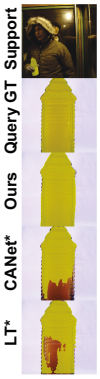

Figure 4. Qualitative comparison with state-of-the-art methods. Best view in color. Zoom in for details.

Variant	1-shot (%)	5-shots (%)
MAP (Baseline)	54.0	56.1
ORG	54.9 (+0.9)	56.7 (+0.6)
ORG + WST	**56.3 (+2.3)**	**59.7 (+3.6)**

Table 4. Ablation study of key components on Pascal-5^i dataset.

denotes a variant of our model where the ORG module is used but without the weakly-supervised training scheme; ORG + WST is our final design where the ORG module is trained with the weakly-supervised training scheme.

Our ablation study shows the effectiveness of the proposed weakly-supervised object learning approach. We can see that it is crucial to use the weakly-supervised training scheme when training the ORG module as it implicitly helps the ORG module learn a better object representation.

Intuitively, a better object representation is more discriminating and hence can be used to better separate target objects from the background. To further validate this idea, we visualize the attention map A_q generated by two different object representation methods in Figure 5. As we can see from the visualization, the attention maps generated using our approach are more accurate and have less background noise. By helping the model focuses on the correct region and filtered out the background noises, the ORG module eventually helps the segmentation branch to produce better segmentation results.

5.7. Execution Time

We report the inference time of our approach on a single NVIDIA GTX 1080 Ti GPU with different configurations in Table 5. The inference time of FWB [17] is copied from their paper which was also tested on the same GPU when K=10. We can see that using a deeper backbone will significantly increase the inference time, while increasing the input size or the number of support examples only adds small computational overhead. This demonstrates that our approach scales well to cases with more support examples.

In summary, our approach is able to run at 42 FPS for

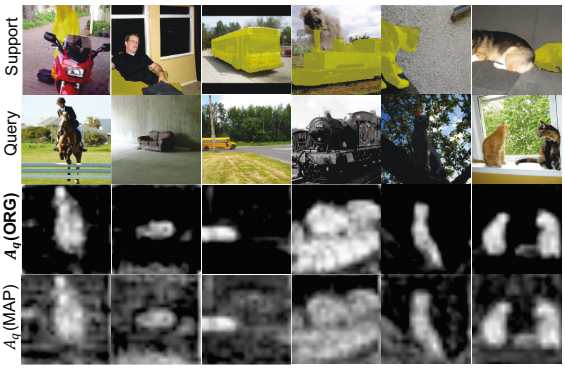

Figure 5. Attention maps generated using the object representation produced by the proposed ORG module vs. by MAP operation. Zoom in for details.

Method	Backbone	Input Size	1-shot	5-shots	10-shots
Ours	R50	321×321	23.71	23.72	25.63
Ours	R101	321×321	38.69	39.83	42.08
Ours	R101	513×513	38.75	39.02	43.34
FWB [17]	R101	512×512	–	–	360.00

Table 5. Inference time per example in milliseconds (ms). Tested on a single NVIDIA GTX 1080 Ti GPU.

one-shot segmentation when using ResNet50 as the backbone and 321×321 as the input resolution. Even when using ResNet101 as the backbone and 513×513 as the input resolution for 10-shots segmentation, our approach can still run at 23 FPS.

6. Conclusion

In this paper, we present a semantic segmentation framework that is capable of segmenting unseen object categories merely based on one or a few support examples. Our motivation is to design an approach that can produce better object-level representations from support examples so that it can be used to better differentiate the target object from the background. To achieve this goal, we propose an object representation generator module that learns to generate high-quality object representation in a weakly-supervised fashion without extra human annotations. This core mechanism is incorporated into a modified encoder-decoder network to create a powerful and efficient end-to-end few-shot segmentation framework. We conduct comprehensive experiments and analyses to demonstrate the effectiveness of our design. In particular, our approach achieves superior performance on two few-shot segmentation benchmarks, outperforming all existing works. Finally, our speed evaluation shows that our method is efficient and scalable.

Acknowledgement. This work was partially supported by National Science Foundation Grant CPS 1931867, and a Qualcomm gift. We also thank Nvidia for a GPU gift to our laboratory.

References

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

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

[3] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *Proceedings of the European conference on computer vision (ECCV)*, pages 801–818, 2018.

[4] Felipe Codevilla, Eder Santana, Antonio M López, and Adrien Gaidon. Exploring the limitations of behavior cloning for autonomous driving. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 9329–9338, 2019.

[5] Jiankang Deng, Jia Guo, Niannan Xue, and Stefanos Zafeiriou. Arcface: Additive angular margin loss for deep face recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4690–4699, 2019.

[6] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results. http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html.

[7] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1126–1135. JMLR. org, 2017.

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

[9] Seunghoon Hong, Junhyuk Oh, Honglak Lee, and Bohyung Han. Learning transferrable knowledge for semantic segmentation with deep convolutional neural network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3204–3212, 2016.

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

[11] Xin Li and Mooi Choo Chuah. Sbgar: Semantics based group activity recognition. In *Proceedings of the IEEE international conference on computer vision*, pages 2876–2885, 2017.

[12] Xin Li and Mooi Choo Chuah. Rehar: Robust and efficient human activity recognition. In *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 362–371. IEEE, 2018.

[13] Xin Li, Xiaowen Ying, and Mooi Choo Chuah. Grip++: Enhanced graph-based interaction-aware trajectory prediction for autonomous driving.

[14] Xin Li, Xiaowen Ying, and Mooi Choo Chuah. Grip: Graph-based interaction-aware trajectory prediction. In *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, pages 3960–3966. IEEE, 2019.

[15] 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 *European conference on computer vision*, pages 740–755. Springer, 2014.

[16] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440, 2015.

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

[18] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, pages 8024–8035, 2019.

[19] Keyurkumar Patel, Hu Han, and Anil K Jain. Secure face unlock: Spoof detection on smartphones. *IEEE transactions on information forensics and security*, 11(10):2268–2283, 2016.

[20] Sashank J Reddi, Satyen Kale, and Sanjiv Kumar. On the convergence of adam and beyond. *arXiv preprint arXiv:1904.09237*, 2019.

[21] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

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

[23] Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. In *Advances in Neural Information Processing Systems*, pages 4077–4087, 2017.

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

[25] Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Daan Wierstra, et al. Matching networks for one shot learning. In *Advances in neural information processing systems*, pages 3630–3638, 2016.

[26] Boyue Wang, Yongli Hu, Junbin Gao, Yanfeng Sun, and Baocai Yin. Laplacian lrr on product grassmann manifolds for human activity clustering in multicamera video surveillance. *IEEE Transactions on Circuits and Systems for Video Technology*, 27(3):554–566, 2016.

[27] Kaixin Wang, Jun Hao Liew, Yingtian Zou, Daquan Zhou, and Jiashi Feng. Panet: Few-shot image semantic segmentation with prototype alignment. In *Proceedings of the IEEE*

International Conference on Computer Vision, pages 9197–9206, 2019.

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

[29] Xiaowen Ying, Xin Li, and Mooi Choo Chuah. Liveface: A multi-task cnn for fast face-authentication. In *2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 955–960. IEEE, 2018.

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

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

[32] Xiaolin Zhang, Yunchao Wei, Yi Yang, and Thomas Huang. Sg-one: Similarity guidance network for one-shot semantic segmentation. *arXiv preprint arXiv:1810.09091*, 2018.

[33] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. Pyramid scene parsing network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2881–2890, 2017.