Generalized Few-Shot Semantic Segmentation: All You Need is Fine-Tuning

Josh Myers-Dean*  Yinan Zhao+  Brian Price†  Scott Cohen†  Danna Gurari*,†

*University of Colorado, Boulder  †University of Texas at Austin  ‡Adobe Research
{josh.myers-dean, danna.gurari}@colorado.edu, yinanzhao@utexas.edu, {bprice, scohen}@adobe.com

Abstract

Generalized few-shot semantic segmentation was introduced to move beyond only evaluating few-shot segmentation models on novel classes to include testing their ability to remember base classes. While all approaches currently are based on meta-learning, they perform poorly and saturate in learning after observing only a few shots. We propose the first fine-tuning solution, and demonstrate that it addresses the saturation problem while achieving state-of-art results on two datasets, PASCAL-5i and COCO-20i. We also show it outperforms existing methods whether fine-tuning multiple final layers or only the final layer. Finally, we present a triplet loss regularization that shows how to redistribute the balance of performance between novel and base categories so that there is a smaller gap between them.

1. Introduction

Few-shot semantic segmentation is the task of learning to segment select categories in an image when given only a limited number of annotated examples for each category. Generally, such methods first train on base classes for which there is an abundance of labeled data and then try to generalize to novel classes for which only few annotated examples are available. A limitation of most models is that they are only evaluated with respect to their performance on the novel classes [1, 42, 49]. In other words, existing work largely ignores whether the model retains knowledge of how to analyze the base categories. To combat this limitation, generalized few-shot semantic segmentation was introduced as an extension of few-shot semantic segmentation in 2020, such that the performance of models are evaluated on both the base and novel categories.

All generalized few-shot segmentation approaches (and many few-shot segmentation approaches) use a meta-learning based approach. Inspired by the recent success of fine-tuning for other few-shot learning tasks [27, 43], we sought to examine whether fine-tuning methods could be successful for generalized few-shot semantic segmentation. We introduce a simple fine-tuning approach and demonstrate that, without any bells and whistles, it achieves large improvements over prior work overall as well as with respect to the novel and base classes on two datasets for many few-shot learning scenarios (i.e., 1 to 100 shots). An overview of our proposed approach is shown in Figure 1. Furthermore, in the spirit of generalized learning, to achieve strong performance on both base and novel classes we demonstrate how to incorporate contrastive learning (i.e., a triplet loss) to achieve a smaller gap between the performance of base and novel classes. Finally, we analyze how performance changes as we freeze different number of layers in the classifier of our network.

To our knowledge, our work is the first to introduce a fine-tuning approach for generalized few-shot segmentation. We offer this work as promising evidence that fine-tuning is a better alternative to the status quo of meta-learning.
learning based approaches. We hope future researchers will persist with the goal of more generalizable models based on a holistic evaluation procedure that achieves a better balance of performance across base and novel categories.

2. Related Work

Semantic Segmentation. Since FCN [24] was introduced in 2015, deep convolutional neural networks have been the dominant solution for semantic segmentation. While a variety of architectures and features have been introduced to improve the FCN framework [4, 5, 10, 11, 20, 35, 45, 46, 51, 52], a typical underlying assumption is that a large number of densely annotated images is available for training. Unlike such works, we focus on the few-shot scenario where there are limited annotations for some of the object categories.

Few-Shot Semantic Segmentation. Few-shot learning was introduced for semantic segmentation in 2017 [31]. Since, many approaches were proposed that employ a range of techniques including prototypical networks [8, 23, 33, 42, 44], metric learning [12, 15, 22, 28, 30, 38, 41, 48–50] and imprinting weights [32]. The status quo is to evaluate the performance of such methods only on the novel classes that have few training examples. Yet, often a model is also needed to predict the base classes, for which there are many training examples. An undesired outcome of the status quo is that existing methods perform poorly on base classes [37]. Our experiments demonstrate that our new approach outperforms existing methods [39, 42, 49] across all categories by preserving knowledge of base classes when learning to recognize novel classes.

Generalized Few-Shot Semantic Segmentation. Generalized few-shot semantic segmentation, the phrase used to denote methods which are evaluated on both novel and base classes, was proposed in 2020 [37]. While prior work adopted a meta-learning approach [37], we instead introduce a fine-tuning approach. We will show in our experiments that our new approach outperforms the baseline [37] by a significant margin across two datasets regardless of the amount of training data (i.e., 1 to 100 shots), both overall and individually on base and novel categories.

Fine-tuning for Few-Shot Learning. Fine-tuning methods have outperformed meta-learning methods by a large margin in few-shot learning for object detection [43] and image classification [7, 36], achieving state-of-the-art performance. Our experiments reinforce the advantage of fine-tuning by showing that our fine-tuning methods achieve state-of-the-art results for few-shot semantic segmentation across two datasets. Extending findings of prior work, we also demonstrate that a smaller gap can be achieved between the performance of base and novel classes by incorporating contrastive learning (i.e., a triplet loss).

Contrastive Learning. Contrastive learning relies on both positive and negative examples of a class during training. This type of learning has shown to be a successful auxiliary task independently for the few-shot paradigm [3, 6, 16, 17, 19, 21, 25, 29, 34, 47] and for segmentation tasks [14, 40]. Inspired by prior work, we offer the first examination of using contrastive learning for few-shot semantic segmentation with a fine-tuning approach. We show that augmenting triplet loss to our approach results in a better balance of performance between the novel and base categories while achieving comparable performance to a baseline method lacking this additional loss.

3. Method

We now introduce our few-shot semantic segmentation method. We will first introduce the problem definition in Section 3.1, then describe our two-stage fine-tuning approach in Section 3.2, and finally describe how we augment triplet loss to our approach in Section 3.3. We will publicly share all code upon publication to ensure reproducibility.

3.1. Problem Definition

Let \( D_{\text{train}} \) and \( D_{\text{test}} \) denote the training and testing image set of a semantic segmentation dataset respectively. In \( D_{\text{train}} \), let \( C_b \) denote a set of base classes that have many annotated examples and \( C_n \) represent a set of novel classes that have only a few annotated examples. For each image \( I \in D_{\text{test}} \), our goal is to produce a label \( c_{i,j} \in C_b \cup C_n \) for each 2D location \((i,j)\) of image \( I \).

3.2. Vanilla Fine-tuning Approach

Architecture: An overview of our architecture is shown in Figure 1. It consists of two components: a backbone and classifier. To enable fair comparison with the existing generalized few-shot segmentation approach, we design the backbone and classifier using the same architectural elements employed in that meta learning approach [37]: PSPNet [51] with a backbone of ResNet-50 [13].

For our backbone, we use ResNet-50 [13] (up to stage 4). More generally it is an efficient, compact, and popular backbone for many computer vision applications.

For our classifier, we use all the layers of PSPNet after stage 4 of ResNet-50. This classifier consists of a pyramid pooling module [51] followed by a convolution layer (with 512 filters), BatchNorm layer, ReLU activation function, and a final convolution layer (with the number of filters set to match the number of classes to be predicted).

Finally, the output feature is bilinearly upsampled to match the spatial dimensions of the input.

Training: Our training strategy is to separate the backbone learning and classifier learning into two stages: a base
training stage and a few-shot fine-tuning stage. The aim is to first teach the backbone a feature representation that can generalize to a broader range of classes than observed during base training and then teach the classifier to use this feature representation to segment the broader range of classes when observing only a few examples per class.

**Stage I: Base Training.** During base training, we train both the backbone and classifier on base classes for which there is an abundance of annotated examples. Following prior work [51], we train with the following loss function:

\[ L = L_{\text{main}} + \lambda_{\text{aux}} L_{\text{aux}} \]  

(1)

where \( L_{\text{main}} \) is the cross entropy loss for the final semantic segmentation output, and \( L_{\text{aux}} \) is an auxiliary cross entropy loss for another additional classifier applied inside the backbone. The auxiliary loss helps optimize the learning process of deep networks, as reported in PSPNet [51].

As in PSPNet [51], we set \( \lambda_{\text{aux}} = 0.4 \) in our experiments.

For our training implementation, we use a batch size of 16, SGD as our optimizer, and a learning rate of 0.01 with a learning rate decay of 0.00001 and a momentum of 0.9. We train this stage for 50 epochs.

**Stage II: Fine-Tuning.** In the second training stage, we freeze the backbone and fine-tune the classifier. Our loss function, \( L \), for this second stage of training is:

\[ L = L_{\text{main}} \]  

(2)

Note that, compared to the base training stage, we omit the auxiliary loss. That is because the base weights are no longer updated.

For training, we use the same batch size, optimizer, and learning rate as in **Stage I**. Unlike **Stage I**, we use as our training data a random sample of \( K \) images for each base and novel class, where \( K \) is the desired number of shots. The motivation for including base classes is to prevent the model from forgetting the knowledge of base classes learned in the first training stage. This sampling approach was shown to be successful for a fine-tuning approach proposed for few-shot object detection [43]. Unlike **Stage I**, we also train for a maximum of 1000 epochs, and keep the model with the best total mIoU. Of note, this convergence typically happens much earlier than the 1000 epoch mark.

### 3.3. Training with an Augmented Triplet Loss

We next propose triplet loss as a form of regularization. Intuitively, because it implicitly includes a notion of similarity and difference, triplet loss should help base and novel classes to become more separable in the feature space. We experiment with using this loss for (a) both stages of training and (b) only for the second fine-tuning stage. An illustration of our triplet loss is shown in Figure 2.

Generally, triplet loss takes in an anchor sample \((a)\), a positive sample \((p)\), and a negative sample \((n)\) and then aims to pull the anchor and positive points close together in feature space while pushing the anchor and negative points away from each other up to a specified margin, \( \mu \). Towards this aim, we use the following loss function from [40]:

\[ L_{\text{triplet}}(a, p, n) = \max(0, \delta(a, p) - \delta(a, n) + \mu) \]  

(3)

where \( \delta(\cdot) = ||x_1 - y_1||_2 \) and \( \cdot ||_2 \) is the \( \ell_2 \) norm.

We apply triplet loss to the penultimate features of the network, \( F \), since they have the highest semantic resolution before the output layer and that layer has a fixed dimensionality across all datasets (unlike the last layer, which has a dimensionality that is determined by the number of classes relevant for each dataset). Formally, let \( C = C_b \cup C_n \) and \( T_c, \forall c \in C \), denote the set of triplets extracted during a training epoch. Since the amount of possible triplets for each class is large, for latency reasons we only sample min(|\( T_c |, \tau) \) triplets for each class.\(^2\) Let \( C^+ \) be the set of all points of the same arbitrary class, and let \( C^- \) be the set of all points from other classes such that \( C^+ \cap C^- = \emptyset \). To construct \( T_c \), we first randomly sample \( \tau \) points from \( C^+ \) for our anchor points, \( \tau \) points from \( C^- \) that are disjoint from our anchor points to act as positive examples, and \( \tau \) points from \( C^- \) for our negative points. From here we create \( T_c \) by randomly pairing anchor, positive, and negative points without replacement.

**Stage I: Base Training with Triplet Loss.** When augmenting triplet loss to our base training, we arrive at the following loss function:

\[ L = L_{\text{main}} + \lambda_{\text{aux}} L_{\text{aux}} + \lambda_{\text{triplet}} L_{\text{triplet}} \]  

(4)

\(^1\)In preliminary experiments, we observed that fine-tuning the backbone led to worse results due to overfitting to the small fine-tuning set.

\(^2\)We set \( \tau = 50 \) and \( \mu = 1 \) for our experiments.

---

![Figure 2. Illustration showing that our proposed triplet framework is based on features extracted at the penultimate layer of our architecture’s classifier, where features are semantically rich. To compute triplet loss, the algorithm uses each triplet—consisting of an anchor sample \((a)\), positive sample \((p)\) that matches the class of the anchor sample, and negative sample \((n)\) that differs from the class of the anchor sample—to learn to push features describing the anchor and positive points closer together while pushing features describing the anchor and negative samples farther apart (up to a distance of \( \mu \)).](image-url)
where $L_{\text{main}}$ and $L_{\text{aux}}$ are the same as in Equation 1, while $L_{\text{triplet}}$ from Equation 3 is the triplet loss applied to the penultimate layer of the network.

**Stage II: Fine-Tuning with Triplet Loss.** When augmenting triplet loss for fine-tuning, we arrive at the following loss function:

$$L = L_{\text{main}} + \lambda_{\text{triplet}} L_{\text{triplet}}$$

(5)

Given that at the start of fine-tuning our feature space is fit to base classes, we assign a larger weight to triplet loss in order to enforce the notions of similarity and difference. Specifically, by doing so, we prioritize that feature vectors of the same class should be similar and feature vectors from different classes should be dissimilar.

4. Experiments

We now evaluate the performance of our fine-tuning approaches for generalized few-shot segmentation.

4.1. Design

**Datasets:** We conduct experiments using two datasets: PASCAL-5$^i$ [31] and COCO-20$^i$ [31]. For both datasets, the few-shot scenario is mimicked by reserving a subset of classes, called folds, to act as novel classes.

PASCAL-5$^i$ [31] is built from PASCAL VOC 2012 [9]. We follow the dataset split in [31] to evenly split 20 object categories into four folds. The set of class indexes contained in fold $i$ is written as $\{5i + j\}$ where $i \in \{0, 1, 2, 3\}$, $j \in \{1, 2, 3, 4, 5\}$. We perform cross-validation, treating background class and three folds as base classes while deeming the remaining fold as novel classes. At test time, we evaluate with all the images in the validation set on both base and novel classes. Of note, PASCAL [9] has 120-887 annotated images per category. The few-shot setting is mimicked by sampling a subset of the annotated images for the “novel” classes.

COCO-20$^i$ [31] is built from MSCOCO [18]. We follow the dataset split in [38] to evenly split 80 object categories into four folds, each fold with 20 categories. The set of class indexes contained in fold $i$ is written as $\{4j - 3 + i\}$ where $j \in \{1, 2, \ldots, 20\}$, $i \in \{0, 1, 2, 3\}$. As done for PASCAL-5$^i$, we perform cross-validation, treating the background class and three folds as base classes while deeming the remaining fold as novel classes. We use all the images in the validation set for evaluation. Similar to PASCAL-5$^i$, the few-shot setting is mimicked by sampling a subset of the annotated images for the “novel” classes.

**Evaluation Metrics:** As is standard in the literature, we measure the Intersection-over-Union (IoU) for each class and then average the IoU over all relevant classes to obtain mean Intersection-over-Union (mIoU). Final performance scores are computed by averaging mIoU over all the folds in cross validation. We compute the average IoU over all the classes as well as for the base and novel classes separately.

4.2. Analysis of Our Vanilla Fine-tuning Approach

We first evaluate our vanilla, two-stage fine-tuning approach described in Section 3.2. We compare its performance against existing approaches for different numbers of shots by varying the number available for learning. This analysis underscores the base performance of a simple fine-tuning approach, without any bells and whistles.

**Baselines:** For comparison, we evaluate the state-of-the-art method in generalized few-shot semantic segmentation: GFS-Seg [37]. Recall, this is a meta-learning approach and is the only method benchmarked for the generalized setting. Of note, the authors did not publicly-share their code, so we report numbers directly from their paper.

While comparison to state-of-the-art methods [39,42,49] for the traditional few-shot semantic segmentation problem is also relevant, prior work [37] has already demonstrated that these methods perform worse overall compared to GFS-Seg. Recall, such methods were only evaluated on novel classes when they were originally published. The three benchmarked methods [39,42,49] are prototype based, and so generalized performance could be computed by averaging the prototypes of base classes in order to detect base categories within an input.\(^3\)

**Few-shot Learning Settings:** Results comparing the performance of our baseline approach (i.e., Ours-Vanilla) and GFS-Seg [37] for the common few-shot settings—1, 5, and 10 shots—are reported in Tables 1 and 2 for PASCAL-5$^i$ and COCO-20$^i$ respectively.

Overall, our method outperforms GFS-Seg by large margins in all shot settings (1, 5, and 10) on both datasets, overall as well as with respect to base classes and novel classes. For example, in the 5-shot setting on PASCAL-5$^i$, the observed percentage point increase is 12.64 overall with a 8.60 boost on the base classes and a 25.56 boost on the novel classes. Similarly, in the 5-shot setting on COCO-20$^i$, the observed percentage point increase is 7.54 overall with a 5.13 boost on base classes and a 14.91 boost on novel classes. The large improvements across all shots for both datasets demonstrate the benefit of our approach to generalize to novel classes while maintaining the learned knowledge of base classes. Our results highlight that a backbone architecture trained only on base classes can produce a feature representation that generalizes to novel classes as well.

---

\(^3\)While a new state-of-the-art approach has been published since the timing of that analysis, i.e., HSNet [26], there is no clear way for evaluating its generalized performance. It is not based on prototypes but rather employs multi-scaled features to predict a binary mask for the target class. There is no uniform or fair way to evaluate few-shot segmentation models in the generalized case like there is for prototype-based approaches.
Impact of Training Data: We next compare the ability of our baseline approach (i.e., \textit{Ours-Vanilla}) and GFS-Seg [37] to benefit from additional training examples. To do so, we report the performance of our method when varying the number of training examples from 1 to 100. Results for both datasets are shown in Tables 1 and 2 respectively.

A key distinction between our method and GFS-Seg [37] is the ability to improve learning of the novel categories from observing more shots during training. For example, when the number of shots increase from 1 to 10 on PASCAL-5\textsuperscript{i}, we observe a 33.73 percentage point increase for \textit{Ours-Vanilla} while the increase is only 5.31 percentage points for GFS-Seg. The performance improvement of GFS-Seg tapers in its learning at around 5 shots and saturates after about 10 training shots, only gaining 2.35 percentage points in novel \textit{mIoU} despite doubling the training samples. This saturation so quickly after only so few examples suggests that meta-learning based approaches may be prone to underfitting to novel classes for this task. In contrast, \textit{Ours-Vanilla} learns steeply from the first few-shots and continues to benefit from additional training examples, yielding a 126% improvement on PASCAL-5\textsuperscript{i} and a 189% improvement on COCO-20\textsuperscript{i} over GFS-Seg when observing 10 shots. We extend our results to show the improvement of \textit{Ours-Vanilla} from 10 to 100 shots as well. Here we observe a further 12.32 percentage point improvement on PASCAL-5\textsuperscript{i}, overall as well as with respect to the base and novel classes separately for COCO-20\textsuperscript{i}.

We offer our results as promising evidence that a fine-tuning approach may be preferable to a meta-learning approach for the generalized few-shot semantic segmentation task. We note that this work does not negate the merit of meta-learning and instead suggests that other approaches may be favorable in generalized few-shot semantic segmentation.

### Table 1. Performance (mIoU) of our models compared to GFS-Seg [37], overall as well as with respect to the base and novel classes separately for PASCAL-5\textsuperscript{i}.

| Method | # shots | Base mIoU | Novel mIoU | Total mIoU |
|--------|---------|-----------|------------|-----------|
| GFS-Seg [37] | 1 | 62.17 | 17.88 | 51.63 |
| Ours-Vanilla | 1 | 66.84 | 18.82 | 55.41 |
| Ours-ObjDetFT | 1 | 68.89 | 19.96 | 57.01 |
| Ours-TripletFT | 1 | 65.46 | 18.87 | 54.36 |
| Ours-TripletBaseFTLast | 1 | 68.99 | 25.40 | 58.61 |
| Ours-TripletAll | 1 | 66.41 | 19.71 | 55.31 |
| GFS-Seg [37] | 5 | 63.43 | 20.84 | 53.29 |
| Ours-Vanilla | 5 | 72.03 | 46.40 | 65.93 |
| Ours-ObjDetFT | 5 | 71.72 | 39.20 | 63.98 |
| Ours-TripletFT | 5 | 70.66 | 44.47 | 64.41 |
| Ours-TripletBaseFTLast | 5 | 71.88 | 47.24 | 66.01 |
| Ours-TripletAll | 5 | 71.31 | 50.46 | 66.35 |
| GFS-Seg [37] | 10 | 64.52 | 23.19 | 54.68 |
| Ours-Vanilla | 10 | 73.02 | 52.55 | 68.14 |
| Ours-ObjDetFT | 10 | 72.75 | 46.40 | 66.45 |
| Ours-TripletFT | 10 | 72.06 | 53.56 | 67.66 |
| Ours-TripletBaseFTLast | 10 | 73.19 | 51.54 | 68.04 |
| Ours-TripletAll | 10 | 72.87 | 57.00 | 69.10 |
| Ours-Vanilla | 100 | 75.33 | 64.87 | 72.84 |
| Ours-ObjDetFT | 100 | 74.25 | 52.70 | 69.12 |
| Ours-TripletFT | 100 | 74.52 | 65.18 | 72.03 |
| Ours-TripletBaseFTLast | 100 | 75.12 | 61.35 | 71.83 |
| Ours-TripletAll | 100 | 75.37 | 68.35 | 73.70 |

Table 1. Performance (mIoU) of our approach compared to GFS-Seg [37], overall as well as with respect to the base and novel classes separately for COCO-20\textsuperscript{i}.

| Method | # shots | Base mIoU | Novel mIoU | Total mIoU |
|--------|---------|-----------|------------|-----------|
| GFS-Seg [37] | 1 | 40.64 | 7.36 | 32.42 |
| Ours-Vanilla | 1 | 43.42 | 8.94 | 34.90 |
| Ours-ObjDetFT | 1 | 46.02 | 8.28 | 36.70 |
| Ours-TripletFT | 1 | 44.06 | 7.53 | 35.04 |
| Ours-TripletBaseFTLast | 1 | 45.26 | 11.62 | 36.95 |
| Ours-TripletAll | 1 | 43.64 | 9.23 | 35.14 |
| GFS-Seg [37] | 5 | 42.05 | 9.81 | 34.09 |
| Ours-Vanilla | 5 | 47.18 | 24.72 | 41.63 |
| Ours-ObjDetFT | 5 | 47.57 | 20.59 | 40.91 |
| Ours-TripletFT | 5 | 45.62 | 22.97 | 39.94 |
| Ours-TripletBaseFTLast | 5 | 47.38 | 27.87 | 42.57 |
| Ours-TripletAll | 5 | 46.61 | 28.84 | 41.36 |
| GFS-Seg [37] | 10 | 42.81 | 10.39 | 34.81 |
| Ours-Vanilla | 10 | 48.18 | 30.03 | 48.37 |
| Ours-ObjDetFT | 10 | 48.52 | 25.46 | 42.82 |
| Ours-TripletFT | 10 | 46.65 | 28.28 | 42.12 |
| Ours-TripletBaseFTLast | 10 | 48.06 | 31.15 | 43.88 |
| Ours-TripletAll | 10 | 46.61 | 34.49 | 43.27 |
| Ours-Vanilla | 100 | 50.94 | 40.23 | 48.30 |
| Ours-ObjDetFT | 100 | 50.14 | 32.04 | 45.67 |
| Ours-TripletFT | 100 | 50.95 | 40.71 | 48.41 |
| Ours-TripletBaseFTLast | 100 | 50.33 | 39.06 | 47.55 |
| Ours-TripletAll | 100 | 50.01 | 45.19 | 48.81 |

Table 2. Performance (mIoU) of our approach compared to GFS-Seg [37] in all common few shot settings, specifically when there are 1, 5 and 10 shots of the novel classes available.
PASCAL-5\textsuperscript{i} and a further 10.20 percentage point improvement on COCO-20\textsuperscript{i}, again reinforcing that our approach does not saturate as we increase the number of training samples. We show qualitative results for intervals between 1 and 100 shots for \textit{Ours-Vanilla} in Figure 3. These results exemplify that, as we increase in the number of training examples, we see more fine-grained segmentations and better novel class recognition.

### 4.3. Analysis of Different Fine-Tuning Approaches

We next compare our fine-tuning approach to that used by prior work for the few-shot object detection problem \cite{carion2020end}. Specifically, we only fine-tune the last convolutional layer in our network rather than the multiple layers of the classifier that are fine-tuned by our vanilla approach. To do so, we freeze all other layers in the network and then follow the fine-tuning approach described in Section 3.2. We refer to this variant as \textit{Ours-ObjDetFT}. Note that less than 0.2\% of the model parameters fine-tuned for \textit{Ours-Vanilla} are fine-tuned for \textit{Ours-ObjDetFT}; i.e., 10,773 vs 23, 082, 517 parameters for PASCAL-5\textsuperscript{i} and 41,553 vs 23, 113, 297 parameters for COCO-20\textsuperscript{i}. Quantitative results for PASCAL-5\textsuperscript{i} and COCO-20\textsuperscript{i} are shown in Tables 1 and 2 respectively.

We observe that \textit{Ours-ObjDetFT}, even with much fewer learnable parameters than \textit{Ours-Vanilla}, still considerably outperforms meta-learning in the generalized few-shot semantic segmentation task overall as well as with respect to base and novel classes. This reinforces our argument for fine-tuning as a preferred solution over meta-learning.

Our results also offer initial insights into which features are most useful to fine-tune. In the 1-shot scenario, we observe fine-tuning only the last layer leads to a slight overall boost in performance across both datasets. However, this boost only stems from a performance gain on the base classes and not on the novel classes. We suspect that fine-tuning a smaller number of parameters may lack enough representational power to generalize to novel classes. In contrast, we almost always observe better results when fine-tuning more layers for the 5-shot and 10-shot settings, including for both base and novel classes.

An interesting area for future work is examining how to find a single fine-tuning approach that would generalize well across all shots. Further analysis could also be conducted to assess whether a single fine-tuning approach is best-suited for different tasks, given that different tasks have different requirements; e.g., intuitively, object detection may need to fine-tune fewer parameters since they produce fewer predictions overall (i.e., bounding box and classification for each object) than for semantic segmentation (i.e., per-pixel predictions). We show qualitative results comparing \textit{Ours-Vanilla} to \textit{Ours-ObjDetFT} in Figure 4. This exemplifies that fine-tuning with more layers leads to slightly better novel class identification and overall segmentation quality, especially in the case when multiple classes are present (as shown in the first and second to last rows).

### 4.4. Analysis of Augmenting Triplet Loss

We next examine how augmenting triplet loss to our baseline approach (i.e., \textit{Ours-Vanilla}) performs both when using it only for fine-tuning at the second stage (i.e., \textit{Ours-TripletFT}) as well as using it for both stages of our approach (i.e., \textit{Ours-TripletAll}). Results are reported in Tables 1 and 2 for PASCAL-5\textsuperscript{i} and COCO-20\textsuperscript{i} respectively.

Our first observation is that it is better to use triplet loss for both stages (i.e., \textit{Ours-TripletAll}) than only in the fine-tuning stage (i.e., \textit{Ours-TripletFT}). We observe that this performance boost stems predominantly from the novel cate-
Probability

Vanilla Triplet Vanilla Triplet
PASCAL COCO

Figure 5. Boxplot showing the softmax scores from Ours-Vanilla and Ours-TripletAll on points sampled from activations of correct novel class predictions for both COCO-20i and PASCAL-5i validation sets. Each box denotes the median score as the central mark, the 25th and 75th percentiles scores as the box edges, and the most extreme data points not considered outliers as the whiskers. Means are denoted with pink boxes. Overall, we observe that triplet loss leads to more confident correct predictions.

Figure 6. Comparison of results from our method to those in GFS-Seg [37], where we leverage results from the GFS-Seg [37] paper to enable comparison (since that code base is not publicly-available to support further comparisons). From left to right: input image, ground truth segmentation of base and novel classes, results from GFS-Seg [37], and then results from Ours-TripletAll. The novel classes are: car, bus, chair, and cat with white meaning ignore those pixels. The first three rows exemplify the advantage of our approach over GFS-Seg, while the last row demonstrates a failure case for our method.

We next examine how augmenting triplet loss for both stages (i.e., Ours-TripletAll) compares to our baseline approach (i.e., Ours-Vanilla). For novel categories, we consistently observe a boost in performance across all shots (1, 5, 10, and 100) on both datasets. In contrast, for base categories, we observe the opposite effect where there typically is a slight drop in performance across all shots (1, 5, 10, and 100) on both datasets. Altogether, these findings suggest that it is beneficial to apply triplet loss during base training because it can reshape the feature space for base classes such that novel classes can more easily be separated from them during fine-tuning.

We provide qualitative results to exemplify how our more balanced fine-tuning approach (i.e., Ours-TripletAll) compares to the prior state-of-the-art method, GFS-Seg [37]. Since prior work [37] did not publish their code, we focus only on examples that those authors provided in their paper. Results are shown in Figure 6. In the first example (i.e., row 1), we observe that our approach generates a better segmentation of the car (i.e., the novel class) and a better gap between the person’s arm and the car. More gen-
erally, this suggests that our approach may be better able to distinguish novel classes from the background class while better capturing fine-grained boundary details. The second example (i.e., row 2) shows that neither Ours-TripletAll nor GFS-Seg [37] are able to segment the car and people from the bus (i.e., the novel class), but our approach is able to more appropriately segment the bus by not introducing holes to the segmentation. The third example (i.e., row 3) shows that Ours-TripletAll again avoids mistakenly segmenting the background class for the novel class of chair as is done by GFS-Seg [37]. Finally, the final example (i.e., row 4) shows a failure case of Ours-TripletAll compared to GFS-Seg [37]. As shown, our approach is not able to as cleanly segment the cat (i.e., the novel class) and confuses some of the cat with the person class, highlighting that while globally our approach outperforms GFS-Seg [37], local failure cases still occur.

To motivate the general purpose benefit of triplet loss, we also report results when applying triplet loss to the second fine-tuning approach we analyze in this paper: Ours-ObjDetFT. We augment triplet loss to the base training stage, and refer to this model as Ours-TripBaseFTLast. Results for PASCAL-5i and COCO-20i are shown in Tables 1 and 2 respectively. These findings reinforce the benefits we observed from augmenting triplet loss to our vanilla fine-tuning approach. First, across all shots, there is a significant increase in performance across both datasets when augmenting triplet loss, due to boosts on novel categories rather than base categories. This further substantiates our claim that including triplet loss during the base training stage serves as a useful step for better segmentation of novel classes downstream.

To demonstrate the benefit of triplet loss over our vanilla approach, we show qualitative results from Ours-Vanilla and Ours-TripletAll in Figure 11. Compared to Ours-Vanilla, Ours-TripletAll is able to more correctly identify novel classes within an image as well as produce finer-grained segmentations. However, while Ours-TripletAll is more accurate at identifying novel classes, there can be a trade off in the granularity of the segmentation (as shown in the sheep example).

5. Ethical Considerations

As discussed in Chan et al. [2], the broader issue among all semantic segmentation algorithms is that decision boundaries must be made when performing classification. As a consequence of this, in some applications (e.g. autonomous vehicles), lives could be at stake over the decision boundary learned. Researchers, engineers, and more generally anyone who utilizes semantic segmentation algorithms should take this into account when applying our (or anyone else’s) models to an application.

A positive consideration of few-shot segmentation, however, is the ability to segment items that may not be abundant in traditional datasets. Specifically, given that most datasets are Euro or East Asian-centric in content, few-shot segmentation has the potential to identify and preserve items from cultures that are sparse and underrepresented in traditional datasets.

6. Conclusion

In this work, we present a simple, yet effective, two-stage fine-tuning based approach for generalized few-shot semantic segmentation. We show that multiple fine-tuning based approaches can achieve state-of-art results and benefit from increasing numbers of shots available in training, despite major differences in representational power. To support generalization of our findings, we demonstrate these results on two datasets across 1, 5, 10, and 100 shots, which shows that performance does not saturate with increased training examples as is seen for the meta-learning based approach. Further experiments demonstrate a promising research direction for the use of contrastive learning in this task. By augmenting triplet loss, we observe a redistribution in overall performance such that the performance on novel categories increases to narrow the gap in performance between the novel and base categories.

Acknowledgments. We gratefully acknowledge support from Microsoft AI for Accessibility for donating cloud computing credits.
References

[1] Malik Boudiaf, Hoel Kervadec, Imtiaz Masud Ziko, Pablo Plantanida, Ismail Ben Ayed, and José Dolz. Few-shot segmentation without meta-learning: A good transductive inference is all you need? 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pages 13974–13983, 2021.

[2] Robin Chan, Matthias Rottmann, Radin Dardashli, Fabian Huger, Peter Schlicht, and Hannu Gotschalk. The ethical dilemma when (not) setting up cost-based decision rules in semantic segmentation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, pages 0–0, 2019.

[3] Jiacheng Chen, Bin-Bin Gao, Zongqing Lu, Jing-Hao Xue, Chengjie Wang, and Qingmin Liao. Scnet: Enhancing few-shot semantic segmentation by self-contrastive background prototypes. ArXiv, abs/2104.09216, 2021.

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

[5] 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.

[6] Xiaocong Chen, Lina Yao, Tao Zhou, Jinming Dong, and Yu Zhang. Momentum contrastive learning for few-shot covid-19 diagnosis from chest ct images. Pattern Recognition, 113:107826, May 2021.

[7] Guneeet S Dhillon, Pratik Chaudhari, Avinash Ravichandran, and Stefano Soatto. A baseline for few-shot image classification. arXiv preprint arXiv:1909.02729, 2019.

[8] Nanqing Dong and Eric Xing. Few-shot semantic segmentation with prototype learning. In BMVC, volume 3, 2018.

[9] Mark Everingham, SM Ali Eslami, Luc Van Gool, Christopher I Williams, John Winn, and Andrew Zisserman. The pascal visual object classes challenge: A retrospective. International journal of computer vision, 111(1):98–136, 2015.

[10] Damien Fourure, Rémi Emonet, Elisa Fromont, Damien Muselet, Alain Trémeau, and Christian Wolf. Residual conv-deconv grid network for semantic segmentation. In Proceedings of the British Machine Vision Conference, 2017, 2017.

[11] Jun Fu, Jing Liu, Haijie Tian, Yong Li, Yongjun Bao, Zhiwei Fang, and Hanqing Lu. Dual attention network for scene classification. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 3146–3154, 2019.

[12] Siddhartha Gairola, Mayur Hemani, Ayush Chopra, and Balaji Krishnamurthy. Simpropnet: Improved similarity propagation for few-shot image segmentation. In Christian Bessiere, editor, Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020, pages 573–579. ijcai.org, 2020.

[13] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 770–778, 2016.

[14] Hanzhe Hu, Jinshi Cui, and Liwei Wang. Region-aware contrastive learning for semantic segmentation. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), pages 16291–16301, October 2021.

[15] Tao Hu, Pengwan Yang, Chiliang Zhang, Gang Yu, Yadong Mu, and Cees GM Snoek. Attention-based multi-context guiding for few-shot semantic segmentation. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 33, pages 8441–8448, 2019.

[16] Jianyi Li and Guizhong Liu. Few-shot image classification via contrastive self-supervised learning. ArXiv, abs/2008.09942, 2020.

[17] Xiaomin Li, Daqian Shi, Xiaolei Diao, and Hao Xu. Selmnet: Boosting few-shot remote sensing scene classification via self-supervised contrastive learning. IEEE Transactions on Geoscience and Remote Sensing, pages 1–12, 2021.

[18] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In European conference on computer vision, pages 740–755. Springer, 2014.

[19] Chen Liu, Yanwei Fu, Chengming Xu, Siqian Yang, Jilin Li, Chengjie Wang, and Li Zhang. Learning a few-shot embedding model with contrastive learning. Proceedings of the AAAI Conference on Artificial Intelligence, 35(10):8635–8643, May 2021.

[20] Wei Liu, Andrew Rabinovich, and Alexander C Berg. Parsenet: Looking wider to see better. arXiv preprint arXiv:1506.04579, 2015.

[21] Weide Liu, Zhonghua Wu, Henghui Ding, Fayao Liu, Jie Lin, and Guosheng Lin. Few-shot segmentation with global and local contrastive learning. arXiv preprint arXiv:2108.05293, 2021.

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

[23] Yongfei Liu, Xiangyi Zhang, Songyang Zhang, and Xuming He. Part-aware prototype network for few-shot semantic segmentation. In European Conference on Computer Vision, pages 142–158. Springer, 2020.

[24] 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.

[25] Orchid Majumder, Avinash Ravichandran, Subransu Maji, Alessandro Achille, Marzia Polito, and Stefano Soatto. Supervised momentum contrastive learning for few-shot classification. ArXiv, abs/2101.11058, 2021.

[26] Juhong Min, Dahyun Kang, and Minsu Cho. Hypercorrelation squeeze for few-shot segmentation. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), 2021.
[27] Akihiro Nakamura and Tatsuya Harada. Revisiting fine-tuning for few-shot learning. CoRR, abs/1910.00216, 2019.

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

[29] Yassine Ouali, Céline Hudelot, and Myriam Tami. Spatial contrastive learning for few-shot classification. In Nuria Oliver, Fernando Pérez-Cruz, Stefan Kramer, Jesse Read, and Jose A. Lozano, editors, Machine Learning and Knowledge Discovery in Databases. Research Track, page 671–686. Springer International Publishing, 2021.

[30] Kate Rakelly, Evan Shelhamer, Trevor Darrell, Alexei A. Efros, and Sergey Levine. Conditional networks for few-shot semantic segmentation. In ICLR, 2018.

[31] Amirreza Shaban, Shray Bansal, Zhen Liu, Irfan Essa, and Byron Boots. One-shot learning for semantic segmentation. In Gabriel Brostow Tae-Kyun Kim, Stefanos Zafeiriou and Krystian Mikolajczyk, editors, Proceedings of the British Machine Vision Conference (BMVC), pages 167.1–167.13. BMVA Press, September 2017.

[32] Memnataullah Siam, Boris N Oreshkin, and Martin Jagersand. Amp: Adaptive masked proxies for few-shot segmentation. In Proceedings of the IEEE International Conference on Computer Vision, pages 5249–5258, 2019.

[33] Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. In Advances in neural information processing systems, pages 4077–4087, 2017.

[34] Bo Sun, Banghui Li, Shengcai Cai, Ye Yuan, and Chi Zhang. Fsoc: Few-shot object detection via contrastive proposal encoding. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pages 7352–7362, June 2021.

[35] Ke Sun, Bin Xiao, Dong Liu, and Jingdong Wang. Deep high-resolution representation learning for human pose estimation. In Proceedings of the IEEE International Conference on Computer Vision, 2019.

[36] Yonglong Tian, Yue Wang, Dilip Krishnan, Joshua B Tenenbaum, and Phillip Isola. Rethinking few-shot image classification: a good embedding is all you need? In Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XIV 16, pages 266–282. Springer, 2020.

[37] Zhuotao Tian, Xin Lai, Li Jiang, Michelle Shu, Hengshuang Zhao, and Jiaya Jia. Generalized few-shot semantic segmentation. arXiv preprint arXiv:2010.05210, 2020.

[38] Zhuotao Tian, Hengshuang Zhao, Michelle Shu, Zhicheng Yang, Ruiyu Li, and Jiaya Jia. Prior guided feature enrichment network for few-shot segmentation. IEEE Annals of the History of Computing. (01):1–1, 2020.

[39] Zhuotao Tian, Hengshuang Zhao, Michelle Shu, Zhicheng Yang, Ruiyu Li, and Jiaya Jia. Prior guided feature enrichment network for few-shot segmentation. TPAMI, 2020.

[40] Daniel Ponsa Vassileios Balntas, Edgar Riba and Krystian Mikolajczyk. Learning local feature descriptors with triplets and shallow convolutional neural networks. In Edwin R. Hancock Richard C. Wilson and William A. P. Smith, editors, Proceedings of the British Machine Vision Conference (BMVC), pages 119.1–119.11. BMVA Press, September 2016.

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

[42] 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.

[43] Xin Wang, Thomas E. Huang, Trevor Darrell, Joseph E Gonzalez, and Fisher Yu. Frustratingly simple few-shot object detection. International Conference on Machine Learning (ICML), July 2020.

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

[45] Maoke Yang, Kun Yu, Chi Zhang, Zhiwei Li, and Kuiyuan Yang. Densenaspp for semantic segmentation in street scenes. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 3684–3692, 2018.

[46] Yuhui Yuan and Jingdong Wang. Ocnet: Object context network for scene parsing. arXiv preprint arXiv:1809.00916, 2018.

[47] Xiangyu Yue, Zangwei Zheng, Shanghang Zhang, Yang Gao, Trevor Darrell, Kurt Keutzer, and Alberto Sangiovanni Vincentelli. Prototypical cross-domain self-supervised learning for few-shot unsupervised domain adaptation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pages 13834–13844, June 2021.

[48] 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.

[49] 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.

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

[51] Hengshuang Zhao, Jiaping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiashi Feng. Adaptive masked proxies for few-shot segmentation. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 2881–2890, 2017.

[52] Hengshuang Zhao, Yi Zhang, Shu Liu, Jianping Shi, Chen Change Loy, Dahua Lin, and Jiaya Jia. Psenet: Point-wise spatial attention network for scene parsing. In Proceedings of the European Conference on Computer Vision (ECCV), pages 267–283, 2018.
Supplementary Materials

This document supplements the main paper with the following:

1. Explanation of choice for $\lambda_{\text{triplet}}$ (supplements Section 3.3).

2. Fine-grained analysis of the effect on base classes when including triplet loss (supplements Section 4.4).

3. Qualitative results on COCO-20$^i$ to highlight the differences in segmentation quality using Ours-Vanilla for shots 1, 5, 10, and 100 (supplements Section 4.2).

4. Qualitative results on COCO-20$^i$ that highlight the differences between Ours-Vanilla and Ours-ObjDetFT for 5 and 10 shots (supplements Section 4.3).

5. Qualitative results on COCO-20$^i$ demonstrating the differences between Ours-Vanilla and Ours-TripletAll (supplements Section 4.4).

6. Additional qualitative results on PASCAL-5$^i$ further highlighting how the inclusion of triplet loss increases novel segmentation quality (supplements Section 4.4).

7. Training Implementation Details

For Equations 4 and 5, we chose $\lambda_{\text{triplet}} = 0.5$ for base training and $\lambda_{\text{triplet}} = 1.0$ for fine-tuning. We share these values to support the reproducibility of our findings.

8. Fine-Grained Analysis of Triplet Loss

Table 3 reinforces our findings of the impact of triplet loss in the main paper; i.e., comparison of Ours-Vanilla, Ours-TripletFT, and Ours-TripletAll in Tables 1 and 2 from the main paper. Specifically, across the different folds, we do not observe a significant gain in base mIoU but the addition of triplet loss during the first-stage of training is useful downstream when generalizing to novel classes.

| Method           | Fold | PASCAL | COCO |
|------------------|------|--------|------|
| Base Stage       | 0    | 72.92  | 42.65|
| Base Stage + Triplet | 0   | 72.25  | 43.25|
| Base Stage       | 1    | 62.91  | 47.67|
| Base Stage + Triplet | 1   | 63.62  | 48.25|
| Base Stage       | 2    | 64.33  | 51.25|
| Base Stage + Triplet | 2   | 65.18  | 50.61|
| Base Stage       | 3    | 73.38  | 49.41|
| Base Stage + Triplet | 3   | 74.41  | 49.40|
| Average Base     | Avg. | 68.38  | 47.75|
| Average Base + Triplet | Avg.| 68.86  | 47.88|

Table 3. Comparison of base training with and without triplet loss for both PASCAL-5$^i$ and COCO-20$^i$. The addition of triplet loss does not have a significant impact on base category performance after the first-stage of training.

9. Additional Qualitative Results

9.1. COCO Dataset

Here we show qualitative results for COCO-20$^i$ on three scenarios (Figures 8, 9, 10): shots 1, 5, 10, 100 for Ours-Vanilla, differences between Ours-Vanilla and Ours-ObjDetFT, and differences between Ours-Vanilla and Ours-TripletAll. For the first scenario (i.e., shot comparison for Ours-Vanilla) we observe better novel class segmentation and detection as the number of shots increase, further suggesting that our approach does not saturate as the number of shots increases. The second scenario (i.e., Ours-Vanilla vs. Ours-ObjDetFT) highlights that when we fine-tune more layers, novel classes experience better segmentations. Finally, in the last scenario (i.e., Ours-Vanilla vs. Ours-TripletAll), we mirror our observation from...
Figure 7 in the main paper that novel classes are segmented better when triplet loss is added to training in both the single and multi-object scenarios.

Figure 8. Results for all shots from Ours-Vanilla on COCO-20. As the number of shots increases, the novel class segmentation quality and localization improves.

Figure 9. Results for 5 and 10 shots from Ours-Vanilla and Ours-ObjDetFT on COCO-20. We continue to observe slightly better novel class identification and segmentation boundaries when fine-tuning more layers (i.e., Ours-Vanilla). Note that the novel categories the bottom of the figure correspond to each row.
Figure 10. Comparison of *Ours-Vanilla* and *Ours-TripletAll* for COCO-20\(^4\). Overall, the addition of triplet loss helps improve the novel class segmentations for both single object and multi-object scenarios.
9.2. PASCAL Dataset

Here we provide additional examples on PASCAL-5 to further substantiate that the addition of triplet loss leads to better novel class segmentation. With the addition of triplet loss, novel classes predictions are more appropriately disjoint from other classes and the overall segmentation is better as well for the novel object.

Figure 11. Additional comparisons of Ours-Vanilla and Ours-TripletAll. Overall, the addition of triplet loss helps improve the novel class segmentations for both single object and multi-object scenarios.