Few-Shot Segmentation Propagation with Guided Networks

Anonymous Author(s)
Affiliation
Address
email

Abstract

Learning-based methods for visual segmentation have made progress on particular types of segmentation, but are limited by the necessary supervision, the narrow definitions of fixed tasks, and the lack of control during inference for correcting errors. To remedy the rigidity and annotation burden of standard approaches, we address the problem of few-shot segmentation: given few image and few pixel supervision, segment any images accordingly. We propose guided networks, which extract a latent task representation from any amount of supervision and are optimized end-to-end for fast, accurate few-shot segmentation. Our method can switch tasks without further optimization and quickly update with more guidance. We report the first results for segmentation from one pixel per concept and for real-time interactive video segmentation. Our unified approach propagates pixel annotations across space for interactive segmentation, across time for video segmentation, and across scenes for semantic segmentation. Our guided segmentor is state-of-the-art in accuracy given the amount of annotation and time.

1 Introduction

Learning a particular type of segmentation, or even extending an existing model to a new task like a new semantic class, generally requires collecting and annotating a large amount of data and (re-)training the model for many iterations. Current methods are supervised by fully annotated images numbering in the thousands or tens of thousands, such that even a "small" dataset contains billions of pixelwise annotations. Collecting these dense annotations is time-consuming, tedious, and error-prone. There are many tasks of practical and scientific interest for which annotation on this scale is impractical or even infeasible, such as graphic design, medical imaging, and more.

Semi- and weakly-supervised segmentation methods can propagate annotations across inputs within a task (instance segmentation throughout a video) or across different types of annotations (tags, boxes, and masks), but current approaches are specific to tasks or forms of supervision, and are often inefficient in computation or data. Once learned, these methods are difficult to guide or correct, and are insensitive to small amounts of further annotation. On the other hand, interactive segmentation methods adjust to the task at hand with few annotations, and can be corrected. However, annotations only control inference on that same image and cannot inform the segmentation of a new input.

We instead address the problem of few-shot segmentation: given only a few images with spatially sparse pixelwise annotations that indicate the task, segment unannotated images accordingly. Our framework is "pixels in, pixels out," for propagating any set of pixel annotations, from within and across images, to unannotated pixels for inference. We directly optimize a guided network to infer a latent task defined by sparse annotations and segment new inputs conditioned on that task. Our few-shot segmentor segments new concepts from as little as one pixel per concept and incorporates further annotations near-instantaneously to update and improve inference. Existing methods, designed for specific segmentation tasks, fail in the extremely sparse regime while our method can propagate

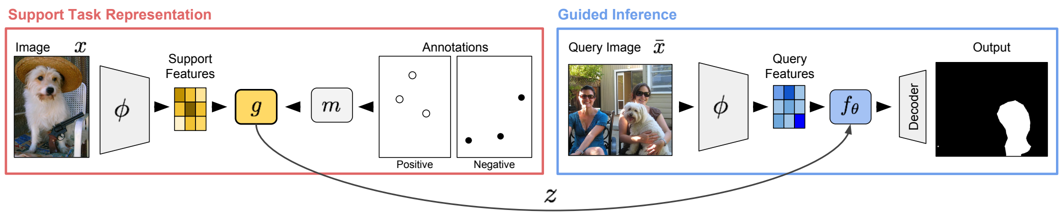


Figure 1: Guided network for few-shot segmentation.

with one annotated pixel per class or dense masks. Our few-shot segmentor is task-agnostic in switching as directed by annotations, data efficient in learning from few pixel-wise annotations, and correctable in incrementally incorporating more supervision.

The few-shot setting divides the input into an annotated support, which supervises the task to be done, and an unannotated query that should be segmented accordingly. In this work we address these key parts of the few-shot segmentation problem: (1) how to summarize the sparse, structured support into a task representation, (2) how to condition pixelwise inference on the given task representation, and (3) how to synthesize segmentation tasks for accuracy and generality. The structured output poses challenges for each of these aspects due to its high-dimensional, statistically dependent, and skewed input and output distributions We make connections to few-shot methods in the image classification setting as we adapt them to segmentation for comparison with our approach.

We propose a new class of *guided* network that extend few-shot and fully convolutional architectures, see Figure 1. Given an annotated support set and query image, the support encoder extracts a latent representation of the task, which guides the segmentation of the query. We carry out a comprehensive comparison of how to encode the support (Section 4.1), and introduce a new late fusion mechanism that improves both learning time and inference accuracy. We examine different choices of conditional inference (Section 4.2) to identify which is best for structured output. Once trained, our model requires no further optimization to perform few-shot tasks, and can update predictions according to additional annotations in real time.

We evaluate our method on a variety of challenging segmentation problems: interactive segmentation in 5.1, semantic segmentation in 5.2, video object segmentation in 5.3, and real-time interactive video segmentation in 5.4. See Figure 2 for an illustration of the problems we consider. The focus of our results is in the sparse regime, for which it is practical to collect annotations. In all cases our accuracy is state-of-the-art for the amount of annotations and time required. The speed with which our method incorporates new annotations makes it suitable for real-time interactive video segmentation.

2 Related Work

Our framework extends and bridges segmentation and few-shot learning. Segmentation is a vast subject with many current directions for deep learning techniques [11]. We focus on one-shot, semi-supervised, and interactive methods, which are addressed separately in existing work. We review few-shot learning methods and how they relate to structured output.

Segmentation There are many modes of segmentation. We take up semantic [7, 16], interactive [3, 13], and video object segmentation [17] as our challenge problems. See Fig. 2 for summary.

For semantic seg., Shaban et al. [21] proposes a pioneering but limited one-shot segmentor (OSLSM), which requires few images but dense annotations and needs semantic supervision at training time. Our few-shot segmentor can segment a class from as few as two points for positive and negative, and even few-shot segment classes from instance supervision, due to our guided architecture. For video object seg., one-shot video object segmentation (OSVOS) by Caelles et al. [4] achieve high accuracy by fine-tuning during inference, but this optimization is too costly in time and fails with sparse annotations. By contrast, we learn to guide segmentation propagation from sparse annotations by synthesizing tasks with that kind of annotation. We rely on feedforward guidance to few-shot learn, making our method much faster. For interactive seg, Xu et al. [27] learn state-of-the-art interactive object segmentation (DIOS), but cannot propagate annotations across different images as we do. This is a bottleneck on annotation efficiency, since it requires $>= 2$ annotations for every input, while our method can segment new inputs on its own. Note that interactive is a special case of few-shot.

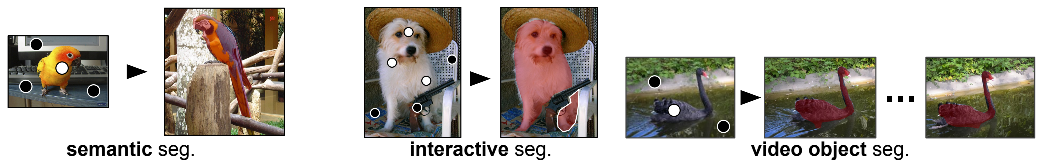


Figure 2: Few-shot segmentation subsumes semantic, interactive, and video object segmentation.

Few-shot learning Few-shot learning [8, 15] holds the promise of data efficiency: in the extreme case, one-shot learning requires only a single annotation of a new concept. The present wave of methods [9, 14, 18, 20, 24, 25] frame it as direct optimization for the few-shot setting: they synthesize few-shot tasks, define a task loss, and learn a task model given the few-shot supervision. Existing works focus on classification, not structured output, and have given little attention to sparse and imbalanced supervision. We are the first to address general few-shot segmentation, ranging from extremely sparse to dense annotations, with efficiency for use on image collections and video.

To situate our work, we summarize approaches as they relate to visual structured output tasks like segmentation. Few-shot as optimization [9, 18] optimizes during inference by gradients on pre-training for fine-tuneability (MAML) [9] or a learned recurrent optimizer [18]. While remarkable in generality by lack of task and architecture assumptions, the second-order optimization and learned optimization of these methods are computationally impractical on large-scale networks for visual recognition, and are unproven for the high dimensionality and skewed distributions of segmentation. Few-shot as embedding [14, 20, 24, 25] learns a metric and retrieves the closest item from the support, inspired by siamese networks for metric learning [6, 12]. These methods are fast on small problems, and can be admirably simple [24], but degrade with higher shot and way. Structured output tasks can have 10^6 elements in one support and query, and the full way of instance recognition can be in the $1000+$. Few-shot as modulation [2, 26] regresses task model parameters based on the support. While adopted by OSLSM, it is difficult to merge the modulations for > 1 shot (and they do not).

3 Few-Shot Segmentation

Few-shot learning divides the input into an annotated support, which supervises the task to be done, and an unannotated query on which to do the task. The common setting in which the support contains K distinct classes and S examples of each is referred to as K-way, S-shot learning [8, 15, 25]. For few-shot segmentation tasks we add a further pixel dimension to this setting, as we must now consider the number of support pixel annotations for each image, as well as the number of annotated support images. We denote the number of annotated pixels per image as P, and consider the settings of (S, P)-shot learning for various S and P. In particular, we focus on sparse annotations where P is small, as these are more practical to collect, and merely require the annotator to point to the segment(s) of interest. This type of data collection is more efficient than collecting dense masks by at least an order of magnitude [1]. As a further step, we handle mixed-shot learning where the amount of annotation varies by class and task since structured output distributions are imbalanced.

We define a few-shot segmentation task as the set of input-output pairs $(\mathcal{T}_i, \mathcal{Y}_i)$ sampled from a task distribution $\mathcal{P}$, adopting and extending the notation of Garcia & Bruna [10]. The task inputs are

$$\mathcal{T} = \big\{\{(x_1, L_1), \dots (x_S, L_S)\}, \{\bar{x}_1, \dots, \bar{x}_Q\}\ ;\ x_s, \bar{x}_q \sim \mathcal{P}_l(\mathbb{R}^N)\big\}$$
$$L_s = \{(p_j, l_j) : j \in \{1\,..\,P\}\}\ ,\ l \in \{1\,..\,K\} \cup \{\varnothing\}$$

where S is the number of annotated support images x_s, Q is the number of unannotated query images $\bar{x}_q$, and L_s are the support annotations. The annotations are sets of point-label pairs (p, l) with $|L_s| = P$ per image, where every label l is one of the K classes or unknown ($\varnothing$). The task outputs, that is the targets for the support-defined segmentation task on the queries, are

$$\mathcal{Y} = (y_1, \dots, y_Q), \quad y_q = \{(p_j, l_j) : j \in \bar{x}_q\}$$

Without loss of generality we consider every task to be binary with $K = 2$, or $L_s = (+_s, -_s)$, where each task defines its own positive and negative is the complement (background, as it is known in segmentation). The choice of binary tasks is a natural one for the problems of interactive segmentation and video object segmentation, in which cases the tasks consist of a single object to be segmented.

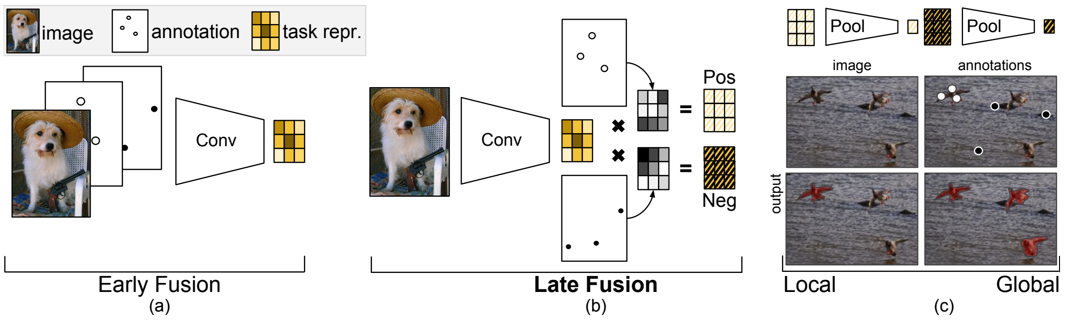


Figure 3: Extracting a task representation or "guidance" from the support. (a) Early fusion simply concatenates the image and annotations. (b) Our late fusion factorizes into image and annotation streams, improves accuracy, and updates quickly given new annotations. (c) Globalizing the task representation propagates appearance non-locally: a single bird is annotated in this example, but global guidance causes all the similar-looking birds to be segmented (red) regardless of location.

Note that higher-way tasks can be handled as a union of binary tasks. We let $Q = 1$ throughout, since inference for each query is independent in our model.

Our approach for the few-shot segmentation problem has two parts: (1) extracting a task representation from the semi-supervised, structured support and (2) segmenting the query given the task representation. We define the task representation as $z = g(x, +, -)$, and the query segmentation guided by that representation as $\hat{y} = f(\bar{x}, z)$. The design of the task representation z and its encoder g is crucial for few-shot segmentation to handle the hierarchical structure of images and pixels, the high and variable dimensions of images and their pixelwise annotations, the semi-supervised nature of support with many unannotated pixels, and skewed support distributions, which we address in Section 4.1. For segmentation given the task representation we meld few-shot techniques to dense, pixelwise inference via fully convolutional networks. While related to few-shot methods on simple, low-dimensional data, our evaluation is the first comparison of these methods on large-scale visual recognition problems that stress the limits of shot, task diversity, and efficiency.

4 Guided Networks

Guided networks reconcile autonomous and interactive modes of inference: a "guided" model is both able to make predictions on its own and incorporate expert guidance for directing the task or correcting errors. Guidance is extracted from the support as a latent task representation z through a guide $g(x, L)$. The model carries out inference in the form of $\hat{y} = f(\bar{x}, z)$ for inputs $\bar{x}$, without further need for the support. That is, $(\hat{y}, \bar{x}) \perp (x, L) \mid z$. The guidance, unlike static model parameters, is not fixed: it can be extended or corrected as directed by an annotator such as a human-in-the-loop.

We examine how to best design the guide g and inference f functions as deep networks. Our method is one part architecture and one part optimization. For architecture, we define branched fully convolutional networks, with a guide branch for extracting the task representation from the support (Section 4.1), and an inference branch for segmenting queries given the guidance (Section 4.2). For optimization, we adapt episodic training of few-shot learning to image-to-image learning for structured output (Section 4.3).

The backbone of our networks is VGG-16 [23], pre-trained on ILSVRC [19], and cast into fully convolutional form [22]. The same is true for the methods we compare against in our results.

4.1 Guidance: From Support to Task Representation

The task representation z must fuse the visual information from the image with the annotations in order to capture what should be segmented in the query. As images with (partial) segmentations, our support is statistically dependent because pixels are spatially correlated, semi-supervised because the full supervision is arduous to annotate, and high dimensional and class-skewed because scenes are sizable and complicated. For simplicity, we first consider a $(1, P)$-shot support consisting of one image with an arbitrary number of annotated pixels P, and then extend to general (S, P)-shot support. As a first step we decompose the support encoder $g(x_s, +_s, -_s)$ across receptive fields indexed by i

for local task representations $z_i = g(x_{si}, +_{si}, -_{si})$; this is the same independence assumption made by existing approaches to structured output. See Figure 3 for an overview.

Early Fusion (prior work) Stacking the image and annotations channel-wise at the input makes $z_{si} = g_{\text{early}}(x, +, -) = \phi_S(x \oplus + \oplus -)$ with a support feature extractor ϕ_S. This early fusion strategy, employed by [27], gives end-to-end learning full control of how to fuse. Masking the image by the positive pixels [21, 28] instead forces invariance to context, potentially speeding up learning, but makes unnatural images and precludes learning from the background. Both early fusion techniques suffer from an inherent modeling issue: incompatibility of the support and query representations. Stacking requires distinct ϕ_S, ϕ_Q while masking divides the input distribution. Early fusion is slow, since changes in annotations trigger a full pass through the network.

Late Fusion (ours) We resolve the learning and inference issues of early fusion by architecturally inducing structure to make $z_{si} = g_{\text{late}}(x, +, -) = \psi(\phi(\bar{x}), m(+), m(-))$. We first extract visual features from the image alone by $\phi(x)$, map the annotations into masks in the feature layer coordinates $m(+), m(-)$, and then fuse both by ψ chosen to be element-wise product, as shown in Figure 3. This factorization into visual and annotation branches defines the spatial relationship between image and annotations, improving data efficiency by not learning and computation time by caching. Fixing m to interpolation and ψ to multiplication, the task representation can be updated quickly by only recomputing the masking and not features ϕ. The visual representations of the support and query are unified by design in a shared feature extractor ϕ, improving learning efficiency. Late fusion improves task accuracy, with 60% relative improvement for video object segmentation (sec. 5.3). Furthermore it reduces inference time, as only the masking needs to be recomputed to incorporate new annotations, making real-time interactive video segmentation possible (section 5.4). By contrast, existing methods require seconds or even minutes to update between interactions.

At first glance, late fusion might seem limited in generality and resolution; however, we can improve these while keeping its advantages. The annotation-to-mask mapping m can be learned end-to-end to capture characteristic patterns of annotations, but more efficiently than early fusion, because it does not have to learn to mask exhaustively across visual variations, unlike the monolithic ϕ_S. The spatial resolution of the support can be improved by feature interpolation in ψ or dilation [5, 29] in ϕ. Note this can be done solely for the support without altering query inference time and output resolution.

Locality We are generally interested in segmentation tasks that are determined by visual characteristics and not absolute location in space or time, i.e. the task is to group pixels of an object and not pixels in the bottom-left of an image. When the support and query images differ, there is no known spatial correspondence, and the only mapping between support and query should be through features. To fit the architecture to this task structure, we merge the local task representations by $m_P(\{z_{si} : \forall i\})$ for all positions i. Choosing global pooling for m_P globalizes the task by discarding the spatial dimensions. The pooling step can be done by averaging, our choice, or other reductions.

However, when the support and query images are the same (e.g. interactive segmentation), feature location can be informative, and m_P can instead be the identity. The effect of this decision is seen in the segmentation of a single image with multiple objects shown by Figure 3.

Multi-Shot The full (S, P)-shot setting requires summarizing the entire support with a variable number of images with varying amounts of pixelwise annotations. Note in this case that the annotations might be divided across the support, for instance one frame of a video may only have positives while a different frame has only negatives, so S-shot cannot simply be reduced to 1-shot, as done in prior work [21]. We form the full task representation $z_S = m_S(\{z_1, \ldots, z_S\})$ by merging the shot-wise image-annotation representations z_s. As long as the merge m_S is differentiable the full (S, P)-shot task representation can be learned end-to-end. Alternatively, optimizing for 1-shot then merging by a fixed m_S such as averaging can suffice, and requires less computation during training.

4.2 Guiding Inference

Inference in a fixed segmentation model is simply $\hat{y} = f_\theta(\bar{x})$ for output y, parameters θ, and input $\bar{x}$. Guided inference is a function $\hat{y} = f(\bar{x}, z)$ of both the query and the guidance extracted from the support. In our case inference has the further form of $f(\phi(\bar{x}), z)$, where ϕ is a fully convolutional encoder from input pixels to visual features. We explore a variety of conditioning approaches, note connections to recently proposed few-shot learning methods for image classification, and determine which most effectively copes with structured output across several types of segmentation tasks.

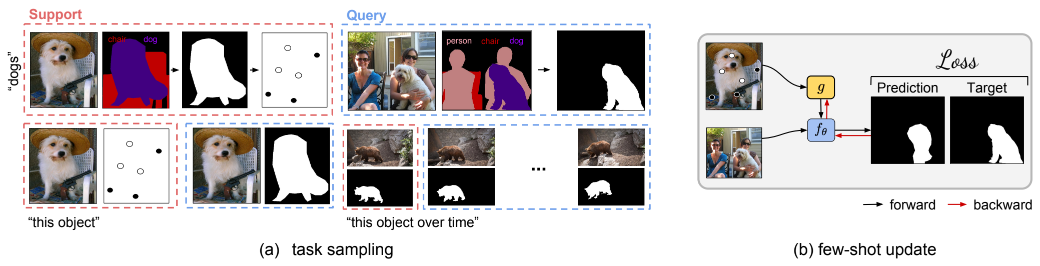


Figure 4: Optimization for few-shot segmentation propagation. (a) Synthesizing tasks from densely annotated segmentation data. (b) One task update: episodic training reduces to supervised learning.

Feature Fusion $\hat{y} = f_\theta(m_f(\phi(\bar{x}), z))$ for fusion operation m_f. In particular we consider fusing by $m_f = \phi(x) \oplus \text{tile}(z)$ which concatenates the task representation with the query features (while tiling z to the spatial dimensions of the query if need be). The fused query-support feature is then decoded to a binary segmentation by a small convolutional network f_θ that can be interpreted as a learned distance metric for retrieval from support to query. Note that the pixel-level matching architecture of Yoon et al. [28] resembles our instantiation of this approach, but without addressing multi-shot and sparse pixel settings, and furthermore they require optimization during inference for few-shot usage.

Parameter Regression $\hat{y} = f_{\theta,\eta(z)}(\phi(x))$ for static inference parameters θ and dynamic task parameters regressed by $\eta(z)$. The task parameters can correspond to any differentiable model, but in existing methods and this work they control the output layer of a fully convolutional network. The remaining majority of the weights θ for other layers are learned by backpropagation during training then fixed. Note that Shaban et al. [21] regress parameters, but their regressor is global, instead of local and fully convolutional. We find that this approach is unstable to optimize, and can sometimes converge to constant task weights that do not vary with the support.

Nearest Neighbors and Prototypes $\hat{y} = \arg\min_j f(\phi(\bar{x}), z_j)$ for any distance metric f. If the local task representations are kept, the query can be inferred by nearest neighbors. Globalizing the task representation class-wise amounts to clustering fully convolutional features into prototypes. In this case the distance from the query to the prototypes implicitly defines the task model. This is a natural extension of [24] to segmentation, but we find it is difficult to optimize for structured output, perhaps due to multi-modality or fine-tuning from pre-training.

To decide how to guide inference, we carry out pilot experiments on interactive and few-shot semantic segmentation. In comparing feature fusion, parameter regression, and prototypes we find feature fusion is more accurate and simpler to tune. This is the inference strategy used for all of our results.

4.3 Optimization and Task Sampling

In standard supervised learning, the model parameters θ are optimized according to the loss between prediction $\hat{y} = f_\theta(x)$ and target $\hat{y}$. In our case, we optimize the parameters of the guide $z = g(x_s, +_s, -_s)$, as well as the parameters of the query segmentor $f_\theta(\bar{x}, z)$. We reduce the problem to supervised learning by training episodically to directly optimize for the few-shot task while learning the parameters of both branches jointly and end-to-end.

Inputs (annotated support and query) and (query) targets are synthesized from a fully labeled dataset. For clarity, we distinguish between binary segmentation problems (tasks) and tasks grouped by problem distribution (modes). For example, semantic segmentation is a mode while segmenting the category of birds is a task. To construct a training episode, we first sample a task (e.g. segment "dogs"), then a subset of images containing that task that we divide into support and query. We binarize support and query annotations to encode the task, and spatially sample support annotations for sparsity. Given inputs and targets, we train the network with cross-entropy loss between the prediction and the (dense) query target. We distinguish between optimizing the parameters of the model during training (learning) and performing few-shot learning during inference (guidance). After learning, the model parameters are fixed, and few-shot learning is achieved via guidance and guided inference. While we evaluate the trained network for varying support size S, as described in 4.2, we train with $S = 1$ for efficiency. Once learned, our guided networks can operate at different (S, P) shots and so we address sparse and dense pixelwise annotations with the same model. In our

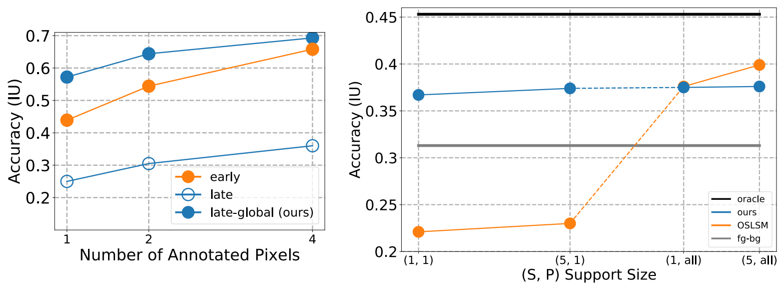


Figure 5: (left) Interactive segmentation of objects in images. (right) Few-shot semantic segmentation of held-out classes: we are state-of-the-art with only two points and competitive with full annotations.

experiments, we train with tasks sampled from a single mode, but co- or cross-supervision of modes is possible. Intriguingly, we see some transfer between modes when evaluating a guided network on a different mode than it was trained on in Section 5.2.

5 Results

We evaluate our few-shot segmentor on a variety of problems that are representative of segmentation as a whole: interactive segmentation, semantic segmentation, and video object segmentation. These are conventionally regarded as separate, with their own aspects and proper approaches, but we reduce each to few-shot segmentation propagation. As a further demonstration of our framework we present the first deep learning approach to *real-time, interactive* video segmentation from dot annotations. Every task is framed as binary segmentation of the support (+) and its complement (-).

To standardize evaluation we select one metric for all tasks: the intersection-over-union (IU) of the positives. This is the usual choice of metric for many segmentation problems, including semantic segmentation [7] and video object segmentation [17], and so we adopt it here. As a further plus, this choice allows us to compare scores across the different problems and tasks we consider.

We include fine-tuning and foreground-background segmentation as baselines for all problems. Fine-tuning simply attempts to optimize the model on the support. Foreground-background verifies that few-shot methods learn, and that their output co-varies with the support. Although the tasks for few-shot segmentation vary with episodes, episodic evaluation cannot give a thorough measure of performance if the data is too simple (if there is only one object in an image for instance).

5.1 Interactive Image Segmentation

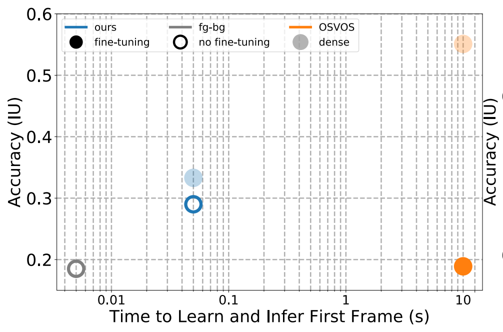
We recover this problem as a special case of few-shot segmentation when the support and query images are identical. We compare to deep interactive object selection (DIOS) [27], because it is state-of-the-art and shares our focus on learning for label efficiency and generality. Our approaches differ in support encoding: DIOS fuses early while we fuse late and globally. Our method is more accurate with extreme sparsity and intrinsically faster to update, as DIOS must do a full forward pass. See Figure 5. From this we decide on late-global guidance throughout.

5.2 Few-shot Semantic Segmentation

Semantic segmentation is a challenge for few-shot learning by the high intra-class variance of appearance. For this problem it is crucial to evaluate on not only held-out inputs, but held-out classes, to be certain the few-shot learner has not covertly learned to be an unguided semantic segmentor. To do so we follow the experimental protocol of [21] and score by averaging across four class-wise splits of PASCAL VOC [7], with has 21 classes (including background), and compare to OSLSM.

Our method achieves state-of-the-art sparse results that rival the most accurate dense results with just two labeled pixels: see Figure 5. Foreground-background is a strong baseline, and we were unable to improve on it with fine-tuning. The oracle is trained on all classes (nothing is held-out).

For this problem our method is curiously insensitive to the amount of annotation. On inspection we determined that the guidance our method extracts has exceptionally low variance for any support in the



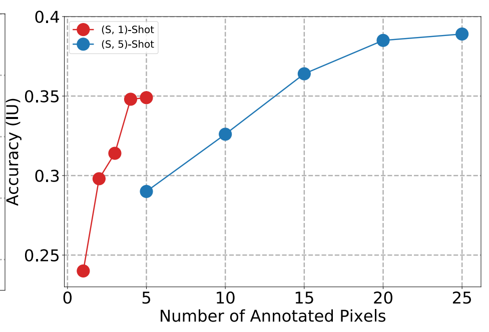


Figure 6: (left) Accuracy-time evaluation for sparse and dense video object segmentation on DAVIS'17 val. (right) Real-time interactive video segmentation on simulated dot interactions.

same class. This is a pathological solution to optimizing for class generalization from insufficient shot. Fundamentally, one shot cannot cover the visual variation present in a class. Consider segmenting a sleek, black dog running given a fluffy, white dog sitting—the guide is forced to be too invariant to color, texture, pose, and more—crippling the few-shot segmentor from improving with shot.

A guided segmentor supervised by instances and trained in the interactive mode learns differently. Since the guide must discriminate among instances, it is more sensitive to visual variation. Learning from *instances* lets our model guide *semantic* tasks from supports that consist of class instances. Without training on classes and with only one +/- pixel of semantic guidance it is more accurate at 26.4% than OSLSM with the same support, and shows steeper $+6$ improvement at 10 pixels while the accuracy of our class-supervised segmentor is near constant. In essence this is an enumerative definition of class as the limit of a collection of instances.

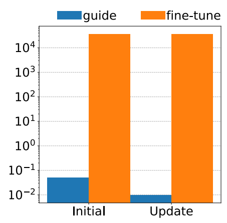
5.3 Video Object Segmentation

We evaluate our approach on the DAVIS 2017 benchmark [17] of 2–3 second videos. We then extend the benchmark to sparse annotations to gauge how methods degrade less. We compare to OSVOS [4], a state-of-the-art method that fine-tunes on the annotated frame and then segments the video frame-by-frame. Our method is state-of-the-art in accuracy in the sparse regime, and in the dense regime when compared to other approaches of comparable speed. See Figure 6.

In the dense regime our method achieves 33.3% accuracy for 80% relative improvement over methods in the same time envelope. Given (much) more time fine-tuning significantly improves in accuracy, but takes 10+min/video. Our method is $\sim 200\times$ faster at 3sec/video. Our method handles extreme sparsity with little degradation, maintaining 87% of the dense accuracy with only 5 points for positive and negative. Fine-tuning struggles to optimize over so few annotations.

5.4 Interactive Video Segmentation

By dividing guidance and inference, our few-shot segmentor can perform real-time interactive video segmentation. As an initial evaluation, we simulate interactions with randomly-sampled dot annotations. We define a benchmark by fixing the amount of annotation and measuring accuracy as the annotations are given. The accuracy-annotation tradeoff curve and timing are plotted in Figure 6. Our guided network improves with both dimensions of shot, whether images (S) or pixels (P). Our architecture is feedforward and fast, and faster still to update on changes to annotations (see right).



6 Discussion

Few-shot segmentation unifies annotation-bound segmentation problems. Guided networks reconcile task-driven and interactive inference by extracting guidance, a latent task representation, from any amount of supervision given. With guidance our few-shot segmentor can switch tasks, improve its accuracy near-instantly with more annotations, and segment new inputs without the supervisor.

References

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

[2] Bertinetto, Luca, Henriques, João F, Valmadre, Jack, Torr, Philip, and Vedaldi, Andrea. Learning feed-forward one-shot learners. In *NIPS*, 2016.

[3] Boykov, Yuri Y and Jolly, M-P. Interactive graph cuts for optimal boundary & region segmentation of objects in nd images. In *Computer Vision, 2001. ICCV 2001. Proceedings. Eighth IEEE International Conference on*, volume 1, pp. 105–112. IEEE, 2001.

[4] Caelles, Sergi, Maninis, Kevis-Kokitsi, Pont-Tuset, Jordi, Leal-Taixé, Laura, Cremers, Daniel, and Van Gool, Luc. One-shot video object segmentation. In *CVPR*, 2017.

[5] Chen, Liang-Chieh, Papandreou, George, Kokkinos, Iasonas, Murphy, Kevin, and Yuille, Alan L. Semantic image segmentation with deep convolutional nets and fully connected crfs. In *ICLR*, 2015.

[6] Chopra, Sumit, Hadsell, Raia, and LeCun, Yann. Learning a similarity metric discriminatively, with application to face verification. In *CVPR*, volume 1, pp. 539–546. IEEE, 2005.

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

[8] Fei-Fei, Li, Fergus, Rob, and Perona, Pietro. One-shot learning of object categories. *PAMI*, 2006.

[9] Finn, Chelsea, Abbeel, Pieter, and Levine, Sergey. Model-agnostic meta-learning for fast adaptation of deep networks. *arXiv preprint arXiv:1703.03400*, 2017.

[10] Garcia, Victor and Bruna, Joan. Few-shot learning with graph neural networks. In *ICLR*, 2018.

[11] Garcia-Garcia, Alberto, Orts-Escolano, Sergio, Oprea, Sergiu, Villena-Martinez, Victor, and Garcia-Rodriguez, Jose. A review on deep learning techniques applied to semantic segmentation. *arXiv preprint arXiv:1704.06857*, 2017.

[12] Hadsell, Raia, Chopra, Sumit, and LeCun, Yann. Dimensionality reduction by learning an invariant mapping. In *CVPR*, volume 2, pp. 1735–1742. IEEE, 2006.

[13] Kass, Michael, Witkin, Andrew, and Terzopoulos, Demetri. Snakes: Active contour models. *International journal of computer vision*, 1(4):321–331, 1988.

[14] Koch, Gregory, Zemel, Richard, and Salakhutdinov, Ruslan. Siamese neural networks for one-shot image recognition. In *ICML Deep Learning Workshop*, volume 2, 2015.

[15] Lake, Brenden M, Salakhutdinov, Ruslan, and Tenenbaum, Joshua B. Human-level concept learning through probabilistic program induction. *Science*, 2015.

[16] Liu, C., Yuen, J., and Torralba, A. Sift flow: Dense correspondence across scenes and its applications. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 33(5):978–994, 2011.

[17] Pont-Tuset, Jordi, Perazzi, Federico, Caelles, Sergi, Arbeláez, Pablo, Sorkine-Hornung, Alexander, and Van Gool, Luc. The 2017 davis challenge on video object segmentation. *arXiv:1704.00675*, 2017.

[18] Ravi, Sachin and Larochelle, Hugo. Optimization as a model for few-shot learning. In *ICLR*, 2017.

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

[20] Santoro, Adam, Bartunov, Sergey, Botvinick, Matthew, Wierstra, Daan, and Lillicrap, Timothy. Meta-learning with memory-augmented neural networks. In *ICML*, 2016.

[21] Shaban, Amirreza, Bansal, Shray, Liu, Zhen, Essa, Irfan, and Boots, Byron. One-shot learning for semantic segmentation. In *BMVC*, 2017.

[22] Shelhamer, Evan, Long, Jonathan, and Darrell, Trevor. Fully convolutional networks for semantic segmentation. *PAMI*, 2016.

[23] Simonyan, Karen and Zisserman, Andrew. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015.

[24] Snell, Jake, Swersky, Kevin, and Zemel, Richard. Prototypical networks for few-shot learning. In *Advances in Neural Information Processing Systems*, pp. 4080–4090, 2017.

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

[26] Wang, Yu-Xiong and Hebert, Martial. Learning to learn: Model regression networks for easy small sample learning. In *ECCV*, 2016.

[27] Xu, Ning, Price, Brian, Cohen, Scott, Yang, Jimei, and Huang, Thomas S. Deep interactive object selection. In *CVPR*, pp. 373–381, 2016.

[28] Yoon, Jae Shin, Rameau, Francois, Kim, Junsik, Lee, Seokju, Shin, Seunghak, and Kweon, In So. Pixel-level matching for video object segmentation using convolutional neural networks. In *ICCV*, pp. 2186–2195, 2017.

[29] Yu, Fisher and Koltun, Vladlen. Multi-scale context aggregation by dilated convolutions. In *ICLR*, 2016.