Dense Gaussian Processes for Few-Shot Segmentation

Joakim Johnander[1,4], Johan Edstedt[1],
Michael Felsberg[1], Fahad Shahbaz Khan[1,2], and Martin Danelljan[3]

[1] Dept. of Electrical Engineering, Linköping University
first.last@liu.se
[2] Mohamed bin Zayed University of AI
[3] Computer Vision Lab, ETH Zürich
[4] Zenseact AB, Sweden

Abstract. Few-shot segmentation is a challenging dense prediction task, which entails segmenting a novel query image given only a small annotated support set. The key problem is thus to design a method that aggregates detailed information from the support set, while being robust to large variations in appearance and context. To this end, we propose a few-shot segmentation method based on dense Gaussian process (GP) regression. Given the support set, our dense GP learns the mapping from local deep image features to mask values, capable of capturing complex appearance distributions. Furthermore, it provides a principled means of capturing uncertainty, which serves as another powerful cue for the final segmentation, obtained by a CNN decoder. Instead of a one-dimensional mask output, we further exploit the end-to-end learning capabilities of our approach to learn a high-dimensional output space for the GP. Our approach sets a new state-of-the-art on the PASCAL-5^i and COCO-20^i benchmarks, achieving an absolute gain of +8.4 mIoU in the COCO-20^i 5-shot setting. Furthermore, the segmentation quality of our approach scales gracefully when increasing the support set size, while achieving robust cross-dataset transfer. Code and trained models are available at `https://github.com/joakimjohnander/dgpnet`.

1 Introduction

Image few-shot segmentation (FSS) of semantic classes [28] has received increased attention in recent years. The aim is to segment novel *query* images based on only a handful annotated training samples, usually referred to as the *support set*. The FSS method thus needs to extract information from the support set in order to accurately segment a given query image. The problem is highly challenging, since the query image may present radically different views, contexts, scenes, and objects than what is represented in the support set.

The core component in any FSS framework is the mechanism that extracts information from the support set to guide the segmentation of the query image. However, the design of this module presents several challenges. First, it needs

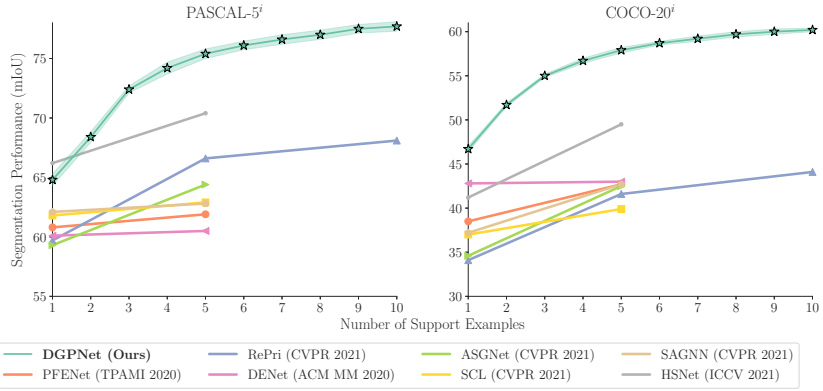

Fig. 1. Performance of the proposed DGPNet approach on the PASCAL-5^i and COCO-20^i benchmarks, compared to the state-of-the-art. We plot the mIoU (higher is better) for different number of support samples. For our approach, we show the mean and standard deviation over 5 experiments. Our GP-based method effectively leverages larger support sets, achieving substantial improvements in segmentation accuracy. Our method also excels in the extreme one-shot case, even outperforming all previously reported results on COCO-20^i.

to aggregate detailed yet generalizable information from the support set, which requires a flexible representation. Second, the FSS method should effectively leverage larger support sets, achieving scalable segmentation performance when increasing its size. While perhaps trivial at first glance, this has proved to be a major obstacle for many state-of-the-art methods, as visualized in Fig. 1. Third, the method is bound to be queried with appearances not included in the support set. To achieve robust predictions even in such common cases, the method needs to assess the relevance of the information in the support images in order to gracefully revert to e.g. learned segmentation priors when necessary.

We address the aforementioned challenges by densely aggregating information in the support set using Gaussian Processes (GPs). Specifically, we use a GP to learn a mapping between dense local deep feature vectors and their corresponding mask values. The mask values are assumed to have a jointly Gaussian distribution with covariance based on the similarity between the corresponding feature vectors. This allows us to extract detailed relations from the support set, with the capability of modeling complex, non-linear mappings. As a non-parametric model [25], the GP further effectively benefits from additional support samples, since all given data is retained. As shown in Fig. 1, the segmentation accuracy of our approach improves consistently with the number of support samples. Lastly, the predictive covariance from the GP provides a principled measure of the uncertainty based on the similarity with local features in the support set.

Our FSS approach is learned end-to-end through episodic training, treating the GP as a layer in a neural network. This further allows us to learn the output space of the GP. To this end, we encode the given support masks with a neural network in order to achieve a multi-dimensional output representation. In order

to generate the final masks, our decoder module employs the predicted mean query encodings, together with the covariance information. Our decoder is thus capable of reasoning about the uncertainty when fusing the predicted mask encodings with learned segmentation priors. Lastly, we further improve our FSS method by integrating dense GPs at multiple scales.

We perform comprehensive experiments on two benchmarks: PASCAL-5^i [28] and COCO-20^i [22]. Our proposed DGPNet outperforms existing methods for 5-shot by a large margin, setting a new state-of-the-art on both benchmarks. When using the ResNet101 backbone, our DGPNet achieves an absolute gain of 8.4 for 5-shot segmentation on the challenging COCO-20^i benchmark, compared to the best reported results in the literature. We further demonstrate the cross-dataset transfer capabilities of our DGPNet approach from COCO-20^i to PASCAL and perform detailed ablative studies to probe the effectiveness of our contributions.

2 Related Work

Few-Shot Segmentation The earliest work in few-shot segmentation (FSS), by Shaban *et al*. [28], proposed a method for predicting the weights of a linear classifier based on the support set, which was further built upon in later works [29,15,4]. Instead of learning the classifier directly, Rakelly *et al*. [24] proposed to construct a global conditioning *prototype* from the support set and concatenate it to the query representation, with several subsequent works [6,48,35,22,46]. Recent works have strived to improve prototype-based approaches. Azad *et al*. [2] reduced the texture bias; Liu *et al*. [16] cross-referenced query and support features with a neural network and iteratively refined the predicted masks; Xie *et al*. [38] combined information at multiple feature levels with a graph neural network (with nodes being feature maps); and Wang *et al*. [33] proposed to model the prototype with a unimodal normal distribution. A major limitation of these methods is the *unimodality* assumption. Wu *et al*. [37] identified that additional support images may actually contaminate the unimodal target class representation, and proposed to weight the different support images. Zhang *et al*. [44] instead opted for more flexible target class models, comprising multiple feature vectors. Yang *et al*. [40], Liu *et al*. [17], and Li *et al*. [12] clustered feature vectors to create a multi-modal target class representation. However, clustering introduces extra hyperparameters, such as the number of clusters, as well as optimization difficulties. In this work, we propose a mechanism that increases the flexibility. The mechanism has few hyperparameters, primarily the choice of covariance function. More closely related to this work are methods that consider pointwise correspondences between the support and query set. These works have mostly focused on attention or attention-like mechanisms [45,42,10,31,34,47]. The work of Min *et al*. [21] proposed a pointwise comparison that was then processed with hypercorrelation layers. In contrast with these methods, we construct a principled posterior over functions, which greatly aids the decoder.

Combining GPs and Neural Networks While early work focused on combining GPs and neural networks in the standard supervised classification set-

ting [27,36,5], there has recently been an increased interest in utilizing Gaussian processes in the context of *few-shot classification* [23,30]. These works employ the GP as the final output layer. This is problematic as the GP assumes the output to have a Gaussian distribution. In contrast, the output in classification has a *categorical* distribution. Thus, these works are forced to optimize proxies of either the predictive or marginal log-likelihood.

In this work, we instead adopt the GP as an internal layer and train a decoder to interpret the Gaussian output and make categorical predictions. As the GP is no longer used to make the final categorical predictions, we have the option to also *learn* the GP output space, further increasing its expressive power. Furthermore, we use a *dense* GP model to model the mapping from individual support features to corresponding encoded mask values. Compared to few-shot image classification, this leads to a high number of points and a high computational cost. We show that this can be addressed by downsampling and compensating the reduced resolution with a high-dimensional output space.

3 Method

3.1 Few-shot Segmentation

Few-shot segmentation is a *dense* few-shot learning task [28]. The aim is to learn to segment objects from novel classes, given only a small set of annotated images. A single instance of this problem, referred to as an *episode*, comprises a small set of annotated samples, called the *support set*, and a set of samples on which prediction is to be made, the *query set*. Formally, we denote the support set as $\{(I_{\mathcal{S}k}, M_{\mathcal{S}k})\}_{k=1}^{K}$, comprising K image-mask pairs $I_{\mathcal{S}k} \in \mathbb{R}^{H_0 \times W_0 \times 3}$ and $M_{\mathcal{S}k} \in \{0,1\}^{H_0 \times W_0}$. A query image is denoted as $I_{\mathcal{Q}} \in \mathbb{R}^{H_0 \times W_0 \times 3}$ and the aim is to predict its corresponding segmentation mask $M_{\mathcal{Q}} \in \{0,1\}^{H_0 \times W_0}$.

To develop our approach, we first provide a general formulation for addressing the FSS problem, which applies to several recent methods, including prototype-based [12,15] and correlation-based [31,34] ones. Our formulation proceeds in three steps: feature extraction, few-shot learning, and prediction. In the first step, deep features are extracted from the given images,

$$\mathbf{x} = F(I) \in \mathbb{R}^{H \times W \times D} \quad . \tag{1}$$

These features provide a more disentangled and invariant representation, which greatly aids the problem of learning from a limited number of samples.

The main challenge in FSS, namely how to most effectively leverage the annotated support samples, is encapsulated in the second step. As a general approach, we consider a learner module Λ that employs the support set in order to find a function f, which associates each query feature $x_{\mathcal{Q}} \in \mathbb{R}^D$ with an *output* $y_{\mathcal{Q}} \in \mathbb{R}^E$. Note that it is common to use a downsampled mask, setting $E = 1$, but we are not restricted to do so. The goal is to achieve an output $y_{\mathcal{Q}}$ that is strongly correlated with the ground-truth query mask, allowing it to act as a

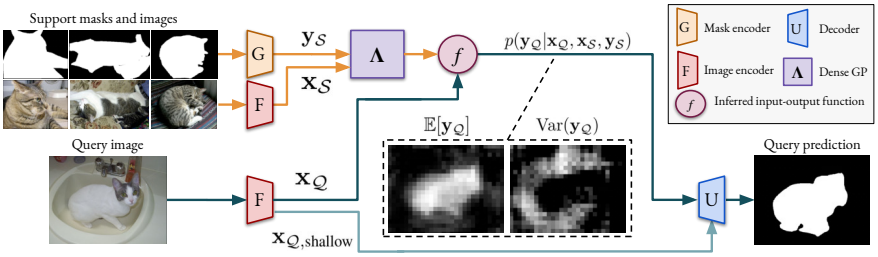

Fig. 2. Overview of our approach. Support and query images are fed through an encoder to produce deep features $\mathbf{x}_S$ and $\mathbf{x}_Q$ respectively. The support masks are fed through another encoder to produce $\mathbf{y}_S$ ($E = 1$ in this figure). Using Gaussian process regression, we infer the probability distribution of the query mask encodings $\mathbf{y}_Q$ given the support set and the query features (see equations 5-7). We create a representation of this distribution and feed it through a decoder. The decoder then predicts a segmentation at the original resolution.

strong cue in the final prediction. Formally, we express this general formulation as,

$$f = \Lambda(\{\mathbf{x}_{Sk}, \mathbf{M}_{Sk}\}_k), \quad y_Q = f(x_Q) \ . \tag{2}$$

The learner Λ aggregates information in the support set $\{\mathbf{x}_{Sk}, \mathbf{M}_{Sk}\}_k$ in order to predict the function f. This function is then applied to query features (2). In the final step of our formulation, output from the function f on the query set is decoded by a separate network as $\widehat{M}_Q = U(\mathbf{y}_Q, \mathbf{x}_Q)$ to predict the segmentation $\widehat{M}_Q$. An overview of our formulation is shown in Fig. 2.

The general formulation in (2) encapsulates several recent approaches for few-shot segmentation. In particular, prototype-based methods, for instance PANet [35], are retrieved by letting Λ represent a mask-pooling operation. The function f then computes the cosine-similarity between the pooled feature vector and the input query features. In general, the design of the learner Λ represents the central problem in few-shot segmentation, since it is the module that extracts information from the support set. Next, we distinguish three key desirable properties of this module.

3.2 Key Properties of Few-Shot Learners

As discussed above, the core component in few-shot segmentation is the few-shot learner Λ. Much research effort has therefore been diverted into its design [22,15,34,17,40,12]. To motivate our approach, we first identify three important properties that the learner should possess.

Flexibility of f The intent in few-shot segmentation is to be able to segment a wide range of classes, unseen during training. The image feature distributions of different unseen classes are not necessarily linearly separable [1]. Prototypical few-shot learners, which are essentially linear classifiers, would fail in such scenarios. Instead, we need a mechanism that can learn and represent more complex functions f.

Scalability in support set size K An FSS method should be able to effectively leverage additional support samples and therefore achieve substantially better accuracy and robustness for larger support sets. However, many prior works show little to no benefit in the 5-shot setting compared to 1-shot. As shown by Li *et al*. [12] and Boudiaf *et al*. [4], it is crucial that new information is effectively incorporated into the model without *averaging* out useful cues.

Uncertainty Modeling Since only a small number of support samples are available in FSS, the network needs to regularly handle unseen appearances in the query images. For instance, the query may include novel backgrounds, scenes, and objects. Since the function f predicted by the learner is not expected to generalize to such unseen scenarios, the network should instead utilize neighboring predictions or learned priors. However, this can only be achieved if f models and communicates the uncertainty of its prediction to the decoder.

3.3 Dense Gaussian Process Few-Shot Learner

As our key contribution, we propose a dense Gaussian Process (GP) learner for few-shot segmentation. As a few-shot learning component, the GP possesses all three properties identified in the previous section. It can represent flexible and highly non-linear functions by selecting an appropriate kernel. It further effectively benefits from additional support samples since all given data is retained. Third, the GP explicitly models the uncertainty in the prediction by learning a probabilistic distribution over a space of functions. Note that both the learning and inference steps of the GP are differentiable functions. While computational cost is a well-known challenge when deploying GPs, we demonstrate that even simple strategies can be used to keep the number of training samples at a tractable level for the FSS problem.

Our dense GP learner predicts a distribution of functions f from the input features x to an output y. First, the support feature maps are stacked and reshaped as a matrix $\mathbf{x}_\mathcal{S} \in \mathbb{R}^{KHW \times D}$. The query feature maps are reshaped as $\mathbf{x}_\mathcal{Q} \in \mathbb{R}^{HW \times D}$. Let $\mathbf{y}_\mathcal{S}$ and $\mathbf{y}_\mathcal{Q}$ be the corresponding outputs. The former is obtained from the support masks and the latter is to be predicted with the GP. The key assumption in the Gaussian process is that the support and query outputs $\mathbf{y}_\mathcal{S}, \mathbf{y}_\mathcal{Q}$ are jointly Gaussian according to,

$$\begin{pmatrix} \mathbf{y}_\mathcal{S} \\ \mathbf{y}_\mathcal{Q} \end{pmatrix} \sim \mathcal{N}\left(\begin{pmatrix} \boldsymbol{\mu}_\mathcal{S} \\ \boldsymbol{\mu}_\mathcal{Q} \end{pmatrix}, \begin{pmatrix} \mathbf{K}_{\mathcal{S}\mathcal{S}} & \mathbf{K}_{\mathcal{S}\mathcal{Q}} \\ \mathbf{K}_{\mathcal{S}\mathcal{Q}}^\top & \mathbf{K}_{\mathcal{Q}\mathcal{Q}} \end{pmatrix} \right) \ . \tag{3}$$

For simplicity, we set the output prior means, $\boldsymbol{\mu}_\mathcal{S}$ and $\boldsymbol{\mu}_\mathcal{Q}$, to zero. The covariance matrix $\boldsymbol{K}$ in (3) is defined by the input features at those points and a *kernel* $\kappa : \mathbb{R}^D \times \mathbb{R}^D \to \mathbb{R}$. In our experiments, we adopt the commonly used *squared exponential* (SE) kernel

$$\kappa(x^m, x^n) = \sigma_f^2 \exp(-\frac{1}{2\ell^2} \|x^m - x^n\|_2^2) \ , \tag{4}$$

with scale parameter σ_f and length parameter ℓ. The kernel can be viewed as a similarity measure. If two features are similar, then the corresponding outputs are correlated.

Next, the posterior probability distribution of the query outputs is inferred (see Fig. 2). The rules for conditioning in a joint Gaussian give us [25]

$$\mathbf{y}_\mathcal{Q}|\mathbf{y}_\mathcal{S}, \mathbf{x}_\mathcal{S}, \mathbf{x}_\mathcal{Q} \sim \mathcal{N}(\boldsymbol{\mu}_{\mathcal{Q}|\mathcal{S}}, \boldsymbol{\Sigma}_{\mathcal{Q}|\mathcal{S}}) \ , \tag{5}$$

where

$$\boldsymbol{\mu}_{\mathcal{Q}|\mathcal{S}} = \mathbf{K}_{\mathcal{S}\mathcal{Q}}^\top (\mathbf{K}_{\mathcal{S}\mathcal{S}} + \sigma_y^2 \mathbf{I})^{-1} \mathbf{y}_\mathcal{S} \ , \tag{6}$$

$$\boldsymbol{\Sigma}_{\mathcal{Q}|\mathcal{S}} = \mathbf{K}_{\mathcal{Q}\mathcal{Q}} - \mathbf{K}_{\mathcal{S}\mathcal{Q}}^\top (\mathbf{K}_{\mathcal{S}\mathcal{S}} + \sigma_y^2 \mathbf{I})^{-1} \mathbf{K}_{\mathcal{S}\mathcal{Q}} \ . \tag{7}$$

The measurements $\mathbf{y}_\mathcal{S}$ are assumed to have been obtained with some additive i.i.d. Gaussian noise with variance σ_y^2. This corresponds to adding a scaled identity matrix $\sigma_y^2 \mathbf{I}$ to the support covariance matrix $\mathbf{K}_{\mathcal{S}\mathcal{S}}$. The equations 4-7 thus predict the distribution of the query outputs, represented by the mean value and the covariance. Note that the asymptotic complexity is $\mathcal{O}((KWH)^3)$. We therefore need to work on a sufficiently low resolution. Next, we introduce a way to incorporate more information into the predicted outputs while maintaining a low resolution.

3.4 Learning the GP Output Space

The decoder strives to transform the query outputs predicted by the few-shot learner into an accurate mask. This is a challenging task given the low resolution and the desire to generalize to classes unseen during offline training. We therefore explore whether additional information can be encoded into the outputs, $\mathbf{y}_\mathcal{Q}$, in order to guide the decoder. This information could for instance include the shape of the mask or which of the object parts are at a given location. To this end, we train a mask-encoder to construct the outputs,

$$\mathbf{y}_{\mathcal{S}k} = G(M_{\mathcal{S}k}) \ . \tag{8}$$

Here, G is a neural network with learnable parameters.

The aforementioned formulation allows us to learn the GP output space during the meta-training stage. To the best of our knowledge, this has not previously been explored in the context of GPs. There is some reminiscence to the attention mechanism in transformers [32]. The transformer queries, keys, and values correspond to our query features $\mathbf{x}_\mathcal{Q}$, support features $\mathbf{x}_\mathcal{S}$, and support outputs $\mathbf{y}_\mathcal{S}$. A difference is that in few-shot segmentation, there is a distinction between the features used for matching and the output – the features stem from the image whereas the output stems from the mask.

As mask encoder G, we employ a lightweight residual convolutional neural network. The network predicts multi-dimensional outputs for the support masks that are then reshaped into $\mathbf{y}_\mathcal{S} \in \mathbb{R}^{KHW \times E}$. These are fed into (6) when the posterior probability distribution of the query outputs is inferred. The two matrix-vector multiplications are transformed into matrix-matrix multiplications and

the result $\boldsymbol{\mu}_{\mathcal{Q}|\mathcal{S}} \in \mathbb{R}^{HW \times E}$ is a matrix containing the multi-dimensional mean. The covariance in (7) is kept unchanged. This setup corresponds to the assumption that the covariance is isotropic over the output feature channels and that the different output feature channels are independent.

3.5 Final Mask Prediction

Next, we decode the predicted distributions of the query outputs in order to predict the final mask (see Fig. 2). Since the output of the GP is richer, including uncertainty and covariance information, we first need to consider what information to give to the final decoder network. We employ the following output features from the GP.

GP output mean The multi-dimensional mean $\boldsymbol{\mu}_{\mathcal{Q}|\mathcal{S}}$ represents the best guess of $\mathbf{y}_\mathcal{Q}$. Ideally, it contains a representation of the mask $M_\mathcal{Q}$ to be predicted. The decoder works on 2D feature maps and $\boldsymbol{\mu}_{\mathcal{Q}|\mathcal{S}}$ is therefore reshaped as $\mathbf{z}_\mu \in \mathbb{R}^{H \times W \times E}$.

GP output covariance The covariance $\boldsymbol{\Sigma}_{\mathcal{Q}|\mathcal{S}}$ captures both the uncertainty in the predicted query output $y_\mathcal{Q}$ and the correlation between different query outputs. The former lets the decoder network identify uncertain regions in the image, and instead rely on, e.g., learnt priors or neighbouring predictions. Moreover, local correlations can tell whether two locations of the image are similar. For each query output, we employ the covariance between it and each of its spatial neighbours in an $N \times N$ region. We represent the covariance as a 2D feature map $\mathbf{z}_\Sigma \in \mathbb{R}^{H \times W \times (N^2)}$ by vectorizing the N^2 covariance values in the $N \times N$ neighborhood. For more details, see Appendix E.

Shallow image features Image features extracted from early layers of deep neural networks are of high resolution and permit precise localization of object boundaries [3]. We therefore store feature maps extracted from earlier layers of F and feed them into the decoder. These serve to guide the decoder as it transforms the low-resolution output mean into a precise, high-resolution mask.

We finally predict the query mask by feeding the above information into a decoder U,

$$\widehat{M}_\mathcal{Q} = U(\mathbf{z}_\mu, \mathbf{z}_\Sigma, \mathbf{x}_{\mathcal{Q},\text{shallow}}) \ . \tag{9}$$

Note that while it would be possible to sample from the inferred distribution of $\mathbf{y}_\mathcal{Q}$, we instead feed its parameters into the decoder that predicts the final mask. As a decoder U, we adopt DFN [43]. DFN processes its input at one scale at a time, starting with the coarsest scale. In our case, this is the concatenated mean and covariance. At each scale, the result at the previous scale is upsampled and processed together with any input at that scale. After processing the finest scaled input, the result is upsampled to the mask scale and classified with a linear layer and a `softmax` function.

3.6 FSS Learner Pyramid

Many computer vision tasks benefit from processing at multiple scales or multiple feature levels. While deep, high-level features directly capture the presence

of objects or semantic classes, the mid-level features capture object parts [31]. In semantic segmentation, methods begin with high-level features and then successively add mid-level features while upsampling [18]. In object detection, a detection head is applied to each level in a feature pyramid [13]. Tian *et al.* [31] discuss and demonstrate the benefit of using both mid-level and high-level features for few-shot segmentation. We therefore adapt our framework to be able to process features at different levels.

We take features from our image encoder F extracted at multiple levels A. Let the support and query features at level $a \in A$ be denoted as $\mathbf{x}_{\mathcal{S}}^a$ and $\mathbf{x}_{\mathcal{Q}}^a$. From the mask encoder G we extract corresponding support outputs $\mathbf{y}_{\mathcal{S}}^a$. We then introduce a few-shot learner Λ^a for each level a. For efficient inference, the support features and outputs are sampled on a grid such that the total stride compared to the original image is 32. The query features retain their resolution. Each few-shot learner then infers a posterior distribution over the query outputs $\mathbf{y}_{\mathcal{Q}}^a$, parameterized by $\boldsymbol{\mu}_{\mathcal{Q}|\mathcal{S}}^a$ and $\boldsymbol{\Sigma}_{\mathcal{Q}|\mathcal{S}}^a$. These are then fed into the decoder, which processes them one scale at a time.

4 Experiments

We validate our proposed approach by conducting comprehensive experiments on two FSS benchmarks: PASCAL-5^i [28] and COCO-20^i [22].

4.1 Experimental Setup

Datasets We conduct experiments on the PASCAL-5^i [28] and COCO-20^i [22] benchmarks. PASCAL-5^i is composed of PASCAL VOC 2012 [7] with additional SBD [8] annotations. The dataset comprises 20 categories split into 4 folds. For each fold, 15 categories are used for training and the remaining 5 for testing. COCO-20^i [22] is more challenging and is built from MS-COCO [14]. Similar to PASCAL-5^i, the COCO-20^i benchmark is split into 4 folds. For each fold, 60 base classes are used for training and the remaining 20 for testing.

Implementation Details Following previous works, we employ ResNet-50 and ResNet-101 [9] backbones pre-trained on ImageNet [26] as image encoders. We let the dense GP work with feature maps produced by the third and the fourth residual module. In addition, we place a single convolutional projection layer that reduces the feature map down to 512 dimensions. As mask encoder, we use a light-weight CNN (see Appendix E). We use $\sigma_y^2 = 0.1$, $\sigma_f^2 = 1$, and $\ell^2 = \sqrt{D}$ in our GP. We train our models for 20k and 40k iterations à 8 episodes for PASCAL-5^i and COCO-20^i, respectively. On one NVIDIA A100 GPU, this takes up to 10 hours. We use the AdamW [11,19] optimizer with a weight decay factor of 10^{-3} and a cross-entropy loss with 1 to 4 background to foreground weighting. We use a learning rate of $5 \cdot 10^{-5}$ for all parameters save for the image encoder, which uses a learning rate of 10^{-6}. The learning rate is decayed with a factor of 0.1 when 10k iterations remain. We freeze the batch normalization layers of the image encoder. Episodes are sampled in the same way as we evaluate. We

Table 1. State-of-the-art comparison on the PASCAL-5^i and COCO-20^i benchmarks in terms of mIoU (higher is better). In each case, the best two results are shown in magenta and cyan font, respectively. Asterisk * denotes re-implementation by Liu *et al*. [15]. Our DGPNet achieves state-of-the-art results for 1-shot and 5-shot on both benchmarks. When using ResNet101, our DGPNet achieves absolute gains of 5.5 and 8.4 for 1-shot and 5-shot segmentation, respectively, on the challenging COCO-20^i benchmark, compared to the best reported results in the literature.

Backbone	Method	PASCAL-5^i		COCO-20^i	
		1-shot	5-shot	1-shot	5-shot
ResNet50	CANet [46] CVPR'19	55.4	57.1	40.8*	42.0*
	PGNet [45] ICCV'19	56.0	58.5	36.7*	37.5*
	RPMM [40] ECCV'20	56.3	57.3	30.6	35.5
	CRNet [16] CVPR'20	55.7	58.8	-	-
	DENet [15] MM'20	60.1	60.5	*42.8*	43.0
	LTM [42] MM'20	57.0	60.6	-	-
	PPNet [17] ECCV'20	51.5	62.0	25.7	36.2
	PFENet [31] TPAMI'20	60.8	61.9	-	-
	RePri [4] CVPR'21	59.1	66.8	34.0	42.1
	SCL [44] CVPR'21	61.8	62.9	-	-
	SAGNN [38] CVPR'21	62.1	62.8	-	-
	CWT [20] ICCV'21	56.4	63.7	32.9	41.3
	CMN [39] ICCV'21	62.8	63.7	39.3	43.1
	MLC [41] ICCV'21	62.1	66.1	33.9	40.6
	MMNet [37] ICCV'21	60.2	61.8	37.2	37.4
	HSNet [21] ICCV'21	*64.0*	*69.5*	39.2	*46.9*
	CyCTR [47] NeurIps'21	**64.2**	65.6	40.3	45.6
	DGPNet (Ours)	63.5±0.4	**73.5±0.3**	**45.0±0.4**	**56.2±0.4**
ResNet101	FWB [22] CVPR'19	56.2	60.0	21.2	23.7
	DAN [34] ECCV'20	58.2	60.5	24.4	29.6
	PFENet [31] TPAMI'20	60.1	61.4	38.5	42.7
	RePri [4] CVPR'21	59.4	65.6	-	-
	VPI [33] WACV'21	57.3	60.4	23.4	27.8
	ASGNet [12] CVPR'21	59.3	64.4	34.6	42.5
	SCL [44] CVPR'21	-	-	37.0	39.9
	SAGNN [38] CVPR'21	-	-	37.2	42.7
	CWT [20] ICCV'21	58.0	64.7	32.4	42.0
	MLC [41] ICCV'21	62.6	68.8	36.4	44.4
	HSNet [21] ICCV'21	**66.2**	*70.4*	*41.2*	*49.5*
	CyCTR [47] NeurIps'21	64.3	66.6	-	-
	DGPNet (Ours)	*64.8±0.5*	**75.4±0.4**	**46.7±0.3**	**57.9±0.3**

randomly flip images horizontally followed by resizing to a size of 384×384 for PASCAL-5^i and 512×512 for COCO-20^i.

Evaluation We evaluate our approach on each fold by randomly sampling 5k and 20k episodes respectively, for PASCAL-5^i and COCO-20^i. This follows the work of Tian *et al*. [31] in which it is observed that the procedure employed by many prior works, using only 1000 episodes, yields fairly high variance. Additionally, following Wang *et al*. [35], Liu *et al*. [17], Boudiaf *et al*. [4], and Zhang *et al*. [44], our results in the state-of-the-art comparison are computed as the

Table 2. Performance comparison (mIoU, higher is better), when performing cross-dataset evaluation from COCO-20^i to PASCAL. When using the same ResNet50 backbone, our approach achieves significant improvements for both 1-shot and 5-shot settings, with absolute gains of 5.8 and 11.3 mIoU over RePRI [4].

Backbone	Method	1-Shot	5-Shot
ResNet50	RPMM [40]	49.6	53.8
	PFENet [31]	61.1	63.4
	RePRI [4]	*63.1*	66.2
	HSNet [21]	61.6	*68.7*
	DGPNet (Ours)	**68.9±0.4**	**77.5±0.2**
ResNet101	HSNet [21]	64.1	70.3
	DGPNet (Ours)	**70.1±0.3**	**78.5±0.3**

average of 5 runs with different seeds. We also report the standard deviation. Performance is measured in terms of mean Intersection over Union (mIoU). First, the IoU is calculated per class over all episodes in a fold. The mIoU is then obtained by averaging the IoU over the classes. As in the original work on FSS by Shaban *et al*. [28], we calculate the IoU on the original resolution of the images.

4.2 State-of-the-Art Comparison

We compare our proposed DGPNet with state-of-the-art FSS approaches on the PASCAL-5^i and COCO-20^i benchmarks, see Table 1. Following prior works, we report results given a single support example, *1-shot*, and given five support examples, *5-shot*. On PASCAL-5^i with a ResNet50 backbone, recently proposed approaches – RePri [4], SCL [44], SAGNN [38], CWT [20], CMN [39], MLC [41], MMNet [37], HSNet [21], and CyCTR [47] – range from 56.4 mIoU to 64.2 mIoU. Our proposed DGPNet obtains a competitive performance of 63.5 mIoU. Where the powerful dense Gaussian process of DGPNet really shines, however, is when additional support examples are given. In the 5-shot setting, these recently proposed approaches range from 61.8 mIoU to 69.5 mIoU. Our proposed DGPNet obtains 73.5 mIoU, setting a new state-of-the-art for 5-shot few-shot segmentation on PASCAL-5^i.

On the challenging COCO-20^i benchmark, the previous best reported 1-shot result comes from the work of Liu *et al*. [15], 42.8 mIoU. In their work, however, no significant improvement is reported when additional support examples are given. Three other approaches – CMN [39], HSNet [21], and CyCTR [47] – obtain performance close to the work of Liu *et al*. [15], from 39.3 mIoU to 40.3 mIoU. Compared to the work of Liu *et al*. [15], CMN, HSNet, and CyCTR scale better with additional support examples, obtaining between 43.1 mIoU and 46.9 mIoU, with HSNet reporting the largest gain, 7.7 mIoU. Our proposed DGPNet obtains a 1-shot performance of 45.0 mIoU, setting a new state-of-the-art. As additional support examples are given, the gap to prior works increases. In the 5-shot setting, DGPNet obtains 56.2 mIoU, beating the previous best approach, HSNet, by 9.7 mIoU. Our per-fold results, including the standard deviation for

each fold, are provided in Appendix A. Qualitative examples are found in Fig. 3 and Appendix F.

Scaling Segmentation Performance with Increased Support Size As discussed earlier, the FSS approach is desired to effectively leverage additional support samples, thereby achieving superior accuracy and robustness for larger shot-sizes. We therefore analyze the effectiveness of the proposed DGPNet by increasing the number of shots on PASCAL-5^i and COCO-20^i. Fig. 1 shows the results on the two benchmarks. We use the model trained for 5-shot in all cases, except for 1-shot. Compared to most existing works, our DGPNet effectively benefits from additional support samples, achieving remarkable improvement in segmentation accuracy with larger support sizes.

We provide a qualitative comparison between different support set sizes in Fig. 3. For this comparison, we use COCO-20^i. In general, the 1-shot setting is quite challenging for several classes due to major variations in where the object appears and what it looks like. If the single support sample is too different from the query sample, the model tends to struggle. With five support samples, the model performs far better. At five, the benefit of additional support samples seems to saturate, with ten samples often bringing only marginal improvements. An example is the train episode in Fig. 3, where minor improvements are obtained when going from five to ten support samples.

Cross-dataset Evaluation

In Table 2, we present the cross-dataset evaluation capabilities of our DGPNet from COCO-20^i to PASCAL. For this cross-dataset evaluation experiment, we followed the same protocol as in Boudiaf *et al*. [4], where the folds are constructed to ensure that there is no overlap with COCO-20^i training folds. When using the same ResNet50 backbone, our approach obtains 1-shot and 5-shot mIoU of 68.9 and 77.5, respectively, with absolute gains of 5.8 and 11.3 over RePRI [4]. Further, DGPNet achieves segmentation mIoU of 70.1 and 78.5 in 1-shot and 5-shot setting, respectively, using ResNet101. For more details on the experiment, see Appendix A.

4.3 Ablation Study

Here, we analyze the impact of the key components in our proposed architecture. We first investigate the choice of kernel, κ. Then, we analyze the impact of the predictive covariance provided by the GP and the benefits of learning the GP output space. Last, we experiment with dense GPs at multiple feature levels. All experiments in this section are conducted with the ResNet50 backbone. Detailed experiments are provided in the Appendix. In each experiment, we also report the mean gain over a baseline (Mean Δ), where the mean is computed over the 1-shot and 5-shot performance on PASCAL-5^i and COCO-20^i.

Choice of Kernel Going from a linear few-shot learner to a more flexible function requires an appropriate choice of kernel. We consider the *homogenous linear kernel* as our baseline. Note that the homogenous linear kernel is equivalent to Bayesian linear regression under appropriate priors [25]. To make our learner more flexible, we consider two additional choices of kernels, the *Exponential* and

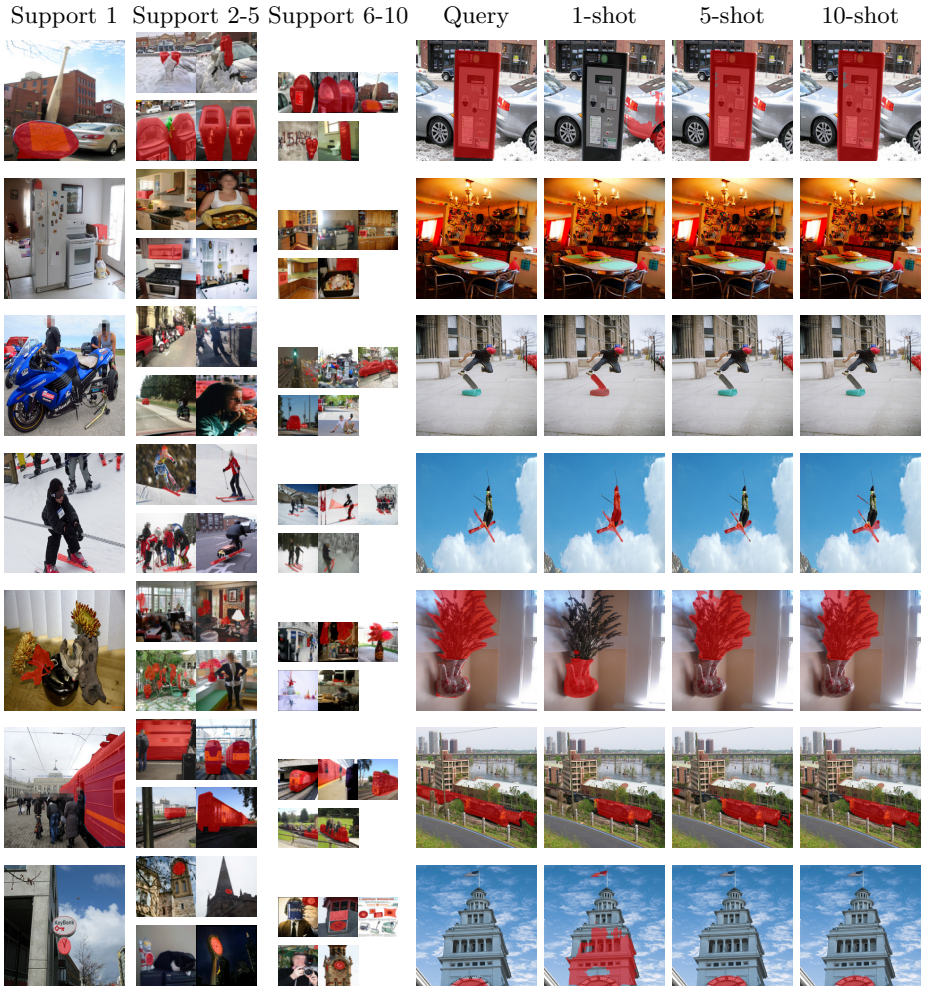

Fig. 3. Right: Qualitative results of our approach given 1, 5, and 10 support samples. The 1-shot results are based on (see left) Support 1; the 5-shot results on Support 1 and Support 2-5; and the 10-shot Support 1, Support 2-5, and Support 6-10. The results are from COCO-20^i and human faces have been pixelized in the visualization, but the model makes predictions on the non-pixelized images.

Squared Exponential (SE) kernels. The kernel equations are given in Appendix C. In Table 3, results on both the PASCAL and COCO benchmarks are presented. Both the Exponential and SE kernels greatly outperform the linear kernel, with the SE kernel leading to a mean gain of 5.5 in mIoU. These results show the benefit of a more flexible and scalable learner.

Incorporating Uncertainty and Learning the GP Output Space We adopt the SE kernel and analyze the performance of letting the decoder process the predictive covariance provided by the dense GP. With the covariance in a

Table 3. Performance of different kernels on the PASCAL-5^i and COCO-20^i benchmarks. Notably, both the Exponential and SE kernels significantly outperform the linear kernel, confirming the need for a flexible learner. Measured in mIoU (higher is better). Best results are in bold.

Kernel	PASCAL-5^i		COCO-20^i		
	1-shot	5-shot	1-shot	5-shot	Mean Δ
Linear	58.6	61.8	37.1	45.3	0.0
Exponential	59.3	67.3	40.9	**51.8**	4.1
SE	**62.1**	**69.9**	**41.7**	51.2	**5.5**

Table 4. Analysis of learning the GP output space (GPO) and incorporating covariance (Cov). Performance in mIoU (higher is better). Best results are in bold.

Cov	GPO	PASCAL-5^i		COCO-20^i		
		1-shot	5-shot	1-shot	5-shot	Mean Δ
		62.1	69.9	41.7	51.2	0.0
✓		**62.5**	71.8	43.8	53.7	1.7
	✓	61.7	**72.7**	43.1	54.2	1.7
✓	✓	**62.5**	**72.6**	**44.7**	**55.0**	**2.5**

Table 5. Performance of different multilevel configurations of the dense GP few-shot learner on the PASCAL-5^i and COCO-20^i benchmarks. Measured in mIoU (higher is better). Best results are in bold.

Str. 16	Str. 32	PASCAL-5^i		COCO-20^i		
		1-shot	5-shot	1-shot	5-shot	Mean Δ
	✓	62.5	72.6	44.7	55.0	0.0
✓		60.4	69.3	40.7	51.1	-3.9
✓	✓	**63.9**	**73.6**	**45.3**	**56.4**	**1.4**

5×5 window, we obtain a gain of 1.7 mIoU. Then, we add the mask-encoder and learn the GP output space (GPO). This leads to a 1.7 improvement in isolation, or a 2.5 mIoU gain together with the covariance.

Multilevel Representations Finally, we investigate the effect of introducing a multilevel hierarchy of dense GPs. Specifically we investigate using combinations of stride 16 and 32 features. The results are reported in Table 5. The results show the benefit of using dense GPs at different feature levels, leading to an average gain of 1.4 mIoU.

5 Conclusion

We have proposed a few-shot learner based on Gaussian process regression for the few-shot segmentation task. The GP models the support set in deep feature space and its flexibility permits it to capture complex feature distributions. It makes probabilistic predictions on the query image, providing both a point estimate and additional uncertainty information. These predictions are fed into a CNN decoder that predicts the final segmentation. The resulting approach sets a new state-of-the-art on 5-shot PASCAL-5^i and COCO-20^i, with absolute improvements of up to 8.4 mIoU. Our approach scales well with larger support sets during inference, even when trained for a fixed number of shots.

Acknowledgements : This work was partially supported by the Wallenberg Artificial Intelligence, Autonomous Systems and Software Program (WASP) funded by Knut and Alice Wallenberg Foundation; ELLIIT; and the ETH Future Computing Laboratory (EFCL), financed by a donation from Huawei Technologies. The computations were enabled by resources provided by the Swedish National Infrastructure for Computing (SNIC), partially funded by the Swedish Research Council through grant agreement no. 2018-05973.

References

1. Allen, K., Shelhamer, E., Shin, H., Tenenbaum, J.: Infinite mixture prototypes for few-shot learning. In: International Conference on Machine Learning. pp. 232–241. PMLR (2019)
2. Azad, R., Fayjie, A.R., Kauffmann, C., Ben Ayed, I., Pedersoli, M., Dolz, J.: On the texture bias for few-shot CNN segmentation. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision. pp. 2674–2683 (2021)
3. Bhat, G., Johnander, J., Danelljan, M., Khan, F.S., Felsberg, M.: Unveiling the power of deep tracking. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 483–498 (2018)
4. Boudiaf, M., Kervadec, H., Masud, Z.I., Piantanida, P., Ben Ayed, I., Dolz, J.: Few-shot segmentation without meta-learning: A good transductive inference is all you need? In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 13979–13988 (2021)
5. Calandra, R., Peters, J., Rasmussen, C.E., Deisenroth, M.P.: Manifold Gaussian Processes for regression. In: Proceedings of the International Joint Conference on Neural Networks (2016). https://doi.org/10.1109/IJCNN.2016.7727626
6. Dong, N., Xing, E.P.: Few-shot semantic segmentation with prototype learning. In: British Machine Vision Conference 2018, BMVC 2018 (2019)
7. Everingham, M., Van Gool, L., Williams, C.K., Winn, J., Zisserman, A.: The pascal visual object classes (VOC) challenge. International Journal of Computer Vision (2010). https://doi.org/10.1007/s11263-009-0275-4
8. Hariharan, B., Arbeláez, P., Bourdev, L., Maji, S., Malik, J.: Semantic contours from inverse detectors. In: Proceedings of the IEEE International Conference on Computer Vision (2011). https://doi.org/10.1109/ICCV.2011.6126343
9. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778 (2016)
10. Hu, T., Yang, P., Zhang, C., Yu, G., Mu, Y., Snoek, C.G.: Attention-based multi-context guiding for few-shot semantic segmentation. In: Proceedings of the AAAI conference on artificial intelligence. vol. 33, pp. 8441–8448 (2019)
11. Kingma, D.P., Ba, J.L.: Adam: A method for stochastic optimization. In: 3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings (2015)
12. Li, G., Jampani, V., Sevilla-Lara, L., Sun, D., Kim, J., Kim, J.: Adaptive prototype learning and allocation for few-shot segmentation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 8334–8343 (2021)
13. Lin, T.Y., Dollár, P., Girshick, R., He, K., Hariharan, B., Belongie, S.: Feature pyramid networks for object detection. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 2117–2125 (2017)
14. Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft COCO: Common objects in context. In: European Conference on Computer Vision. pp. 740–755. Springer (2014). https://doi.org/10.1007/978-3-319-10602-1_48
15. Liu, L., Cao, J., Liu, M., Guo, Y., Chen, Q., Tan, M.: Dynamic extension nets for few-shot semantic segmentation. In: Proceedings of the 28th ACM international conference on multimedia. pp. 1441–1449 (2020)

16. Liu, W., Zhang, C., Lin, G., Liu, F.: CRNet: Cross-Reference Networks for Few-Shot Segmentation. In: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition. pp. 4164–4172 (2020). https://doi.org/10.1109/CVPR42600.2020.00422
17. Liu, Y., Zhang, X., Zhang, S., He, X.: Part-Aware Prototype Network for Few-Shot Semantic Segmentation. In: European Conference on Computer Vision. pp. 142—-158 (2020). https://doi.org/10.1007/978-3-030-58545-7_9
18. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 3431–3440 (2015)
19. Loshchilov, I., Hutter, F.: Decoupled weight decay regularization. In: International Conference on Learning Representations (2019), https://openreview.net/forum?id=Bkg6RiCqY7
20. Lu, Z., He, S., Zhu, X., Zhang, L., Song, Y.Z., Xiang, T.: Simpler is better: Few-shot semantic segmentation with classifier weight transformer. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 8741–8750 (2021)
21. Min, J., Kang, D., Cho, M.: Hypercorrelation squeeze for few-shot segmentation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 6941–6952 (2021)
22. Nguyen, K., Todorovic, S.: Feature weighting and boosting for few-shot segmentation. In: Proceedings of the IEEE International Conference on Computer Vision. vol. 2019-Octob, pp. 622–631 (2019). https://doi.org/10.1109/ICCV.2019.00071
23. Patacchiola, M., Turner, J., Crowley, E.J., Storkey, A.: Bayesian meta-learning for the few-shot setting via deep kernels. In: Advances in Neural Information Processing Systems (2020)
24. Rakelly, K., Shelhamer, E., Darrell, T., Efros, A., Levine, S.: Conditional networks for few-shot semantic segmentation. In: 6th International Conference on Learning Representations, ICLR 2018 - Workshop Track Proceedings (2018)
25. Rasmussen, C.E., Williams, C.K.I.: Gaussian Processes for Machine Learning (2006). https://doi.org/10.7551/mitpress/3206.001.0001
26. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A.C., Fei-Fei, L.: ImageNet Large Scale Visual Recognition Challenge. IJCV pp. 1–42 (April 2015). https://doi.org/10.1007/s11263-015-0816-y
27. Salakhutdinov, R., Hinton, G.: Using deep belief nets to learn covariance kernels for Gaussian processes. In: Advances in Neural Information Processing Systems 20 - Proceedings of the 2007 Conference (2009)
28. Shaban, A., Bansal, S., Liu, Z., Essa, I., Boots, B.: One-shot learning for semantic segmentation. In: British Machine Vision Conference 2017, BMVC 2017 (2017). https://doi.org/10.5244/c.31.167
29. Siam, M., Oreshkin, B., Jagersand, M.: AMP: Adaptive masked proxies for few-shot segmentation. In: Proceedings of the IEEE International Conference on Computer Vision. vol. 2019-Octob, pp. 5248–5257 (2019). https://doi.org/10.1109/ICCV.2019.00535
30. Snell, J., Zemel, R.: Bayesian few-shot classification with one-vs-each pólya-gamma augmented gaussian processes. In: International Conference on Learning Representations (2021), https://openreview.net/forum?id=lgNx56yZh8a
31. Tian, Z., Zhao, H., Shu, M., Yang, Z., Li, R., Jia, J.: Prior Guided Feature Enrichment Network for few-shot segmentation. IEEE Transactions on Pattern Analysis & Machine Intelligence (2020). https://doi.org/10.1109/tpami.2020.3013717

32. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. In: Advances in neural information processing systems. pp. 5998–6008 (2017)
33. Wang, H., Yang, Y., Cao, X., Zhen, X., Snoek, C., Shao, L.: Variational prototype inference for few-shot semantic segmentation. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision. pp. 525–534 (2021)
34. Wang, H., Zhang, X., Hu, Y., Yang, Y., Cao, X., Zhen, X.: Few-Shot Semantic Segmentation with Democratic Attention Networks. In: European Conference on Computer Vision. pp. 730–746 (2020). https://doi.org/10.1007/978-3-030-58601-0_43
35. Wang, K., Liew, J.H., Zou, Y., Zhou, D., Feng, J.: PANet: Few-shot image semantic segmentation with prototype alignment. In: Proceedings of the IEEE International Conference on Computer Vision. vol. 2019-Octob, pp. 9196–9205 (2019). https://doi.org/10.1109/ICCV.2019.00929
36. Wilson, A.G., Hu, Z., Salakhutdinov, R., Xing, E.P.: Deep kernel learning. In: Proceedings of the 19th International Conference on Artificial Intelligence and Statistics, AISTATS 2016 (2016)
37. Wu, Z., Shi, X., Lin, G., Cai, J.: Learning meta-class memory for few-shot semantic segmentation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 517–526 (2021)
38. Xie, G.S., Liu, J., Xiong, H., Shao, L.: Scale-aware graph neural network for few-shot semantic segmentation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 5475–5484 (2021)
39. Xie, G.S., Xiong, H., Liu, J., Yao, Y., Shao, L.: Few-shot semantic segmentation with cyclic memory network. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 7293–7302 (2021)
40. Yang, B., Liu, C., Li, B., Jiao, J., Ye, Q.: Prototype Mixture Models for Few-Shot Semantic Segmentation. In: European Conference on Computer Vision. pp. 763–778. Springer (2020)
41. Yang, L., Zhuo, W., Qi, L., Shi, Y., Gao, Y.: Mining latent classes for few-shot segmentation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 8721–8730 (2021)
42. Yang, Y., Meng, F., Li, H., Wu, Q., Xu, X., Chen, S.: A new local transformation module for few-shot segmentation. In: International Conference on Multimedia Modeling. pp. 76–87. Springer (2020)
43. Yu, C., Wang, J., Peng, C., Gao, C., Yu, G., Sang, N.: Learning a Discriminative Feature Network for Semantic Segmentation. In: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (2018). https://doi.org/10.1109/CVPR.2018.00199
44. Zhang, B., Xiao, J., Qin, T.: Self-guided and cross-guided learning for few-shot segmentation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 8312–8321 (2021)
45. Zhang, C., Lin, G., Liu, F., Guo, J., Wu, Q., Yao, R.: Pyramid graph networks with connection attentions for region-based one-shot semantic segmentation. In: Proceedings of the IEEE International Conference on Computer Vision. vol. 2019-Octob, pp. 9586–9594 (2019). https://doi.org/10.1109/ICCV.2019.00968
46. Zhang, C., Lin, G., Liu, F., Yao, R., Shen, C.: CANET: Class-agnostic segmentation networks with iterative refinement and attentive few-shot learning. In: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (2019). https://doi.org/10.1109/CVPR.2019.00536

47. Zhang, G., Kang, G., Yang, Y., Wei, Y.: Few-shot segmentation via cycle-consistent transformer. Advances in Neural Information Processing Systems **34** (2021)
48. Zhang, X., Wei, Y., Yang, Y., Huang, T.S.: Sg-one: Similarity guidance network for one-shot semantic segmentation. IEEE Transactions on Cybernetics **50**(9), 3855–3865 (2020)