Deep learning solutions for domain-specific image segmentation

Vadineanu, S.

Chapter 4

Few-Shot Cell Segmentation

4.1 Introduction

Recently, deep learning (DL) has become an integral part of many imaging tasks, showing accurate results for problems such as image segmentation [106], a process that labels every pixel of an image into categories. Despite their potential, DL solutions are less applicable in scenarios where the annotated data are scarce [16], such as medical or biological image segmentation. These settings require trained experts to produce the annotations needed to train DL algorithms. Few-shot learning (FSL) techniques present potential solutions for these data-scarce domains by exploiting supervised information from a data-rich source task to adapt to the target task by only utilizing a limited number of labelled samples of the target task [163]. Despite the apparent suitability for cell segmentation, there is a lack of research targeting few-shot segmentation of new structures in cell data sets when other labelled structures are available. Moreover, the particularities of cell imaging make existing few-shot medical image segmentation approaches [160, 127, 138] unsuitable for cell segmentation. Thus, there is a need for a few-shot segmentation method targeted towards cell segmentation.

In FSL, we assume the availability of a relatively large amount of annotated samples

This chapter is based on:

> Ș. Vădineanu, D. M. Pelt, O. Dzyubachyk, and K.J. Batenburg. "From Feature Maps to Few-Shot Cell Segmentation". *International Workshop on Medical Optical Imaging and Virtual Microscopy Image Analysis*. Springer Nature (2025).

for a source task as a *training set*. For a different target task, only a few annotated samples are available, called the *support set*. The challenge is to effectively derive unique representations for the target task from the support set. Subsequently, these representations are leveraged to predict unlabelled samples, known as the *query set*. Given its broad definition, tackling the FSL problem can include domain adaptation [81, 39], image augmentation [27], or visual prompting [79]. Here, we focus on semantic segmentation, where the goal is to segment structures with limited annotations within a data set, leveraging sufficient annotations for other classes of structures. Such a scenario may prove especially suitable for cell segmentation, since, besides requiring domain expert annotators, the number of structures to be annotated within an image is large and the process is tedious due to the varying cell size. Additionally, adapting from one cell type to another may be possible with the limited amount of annotations involved in FSL due to morphological similarities among certain cells.

Despite the promising applicability of FSL techniques to cell segmentation, there is a limited amount of research targeting few-shot segmentation of new classes in cell data sets. Segmenting new classes with FSL is, however, more widely attempted in medical imaging. In this case, one technique that many works rely on is attention-guided segmentation [127, 55, 170], where the activations generated from the support images are used to weigh the activations of the query images. Another popular category of works uses prototype learning [160, 141, 112], where prototype vectors learned from the support set are compared against the features extracted from the query images to generate predictions. Although these approaches perform well in organ segmentation, where the structures are relatively large, morphologically dissimilar, and located in relatively fixed positions, they are not entirely suitable for cell segmentation, where structures do not necessarily fit into the aforementioned pattern. For instance, one difference from organ segmentation lies in the varying cell positions within tissues. This affects attention-based FSL methods since guiding the segmentation of the query based on the attention provided by the support requires alignment between the target structures from these images. This alignment issue is also acknowledged in [122], which motivates the authors to employ the prototype learning paradigm. Prototype learning solutions compare prototype vectors against the feature maps generated in the last layers of an encoder, which results in low-resolution predictions. This can hinder the segmentation of cell microscopy images, which generally contain clusters of cells, since the lack of resolution would not allow for the delineation the individual cells within the clusters. Besides the methods designed for few-shot medical image segmentation, there are many developed for natural images [122, 173, 90]. However,

their applicability has not been extensively explored for (bio)medical imaging.

In this chapter, we propose a novel few-shot segmentation solution designed for cell segmentation. We train mixed-scale dense (MSD) networks [116] as feature extractors on the training set and then we use the support set to learn a linear combination of the extracted features that can be applied to segment a new class of entities. We account for limitations of previous works, such as the low resolution of the prototype-learning predictions, by producing features of the same spatial dimensions as the input image. Moreover, unlike attention-guided methods, we do not require similar positioning between support and query structures since we disentangle the adaptation step on the support from the query prediction. Also, since we only learn a low number of weights for the linear combination, the adaptation step can be performed rapidly, enabling easier prototyping.

4.2 Background and Methodology

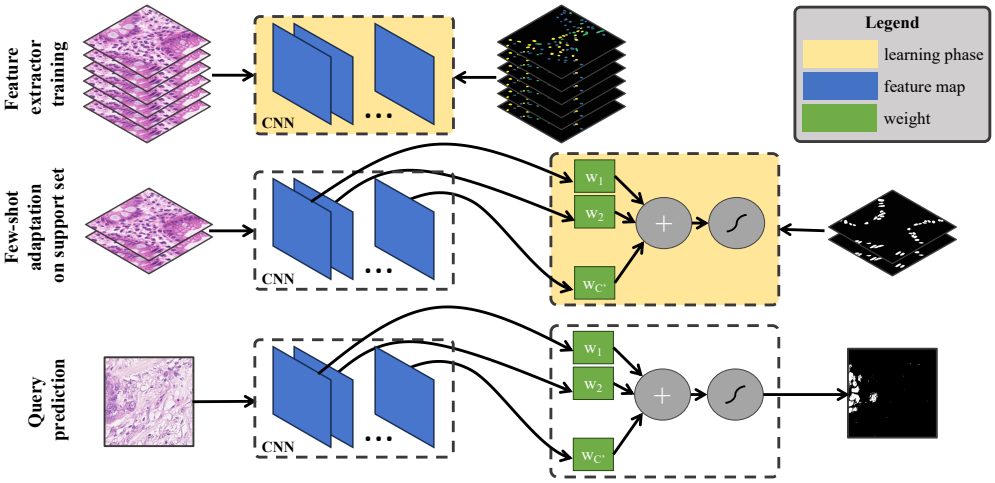

Figure 4.1: Workflow. We train feature extractors on the known classes. Consequently, we extract feature maps from the support set to learn a recombination for predicting the new class. Lastly, we apply the learned recombination to the query images.

We consider the case of segmentation of 2D vector-valued images, e.g., RGB or grayscale, where the aim is to learn a mapping from a matrix of pixels with L rows, M columns, and C channels $x \in \mathbb{R}^{L \times M \times C}$ to the target $y \in \mathbb{Z}^{L \times M}$, where each pixel of y has a value reflecting the entity it belongs to. One method of approximating

this is with convolutional neural networks (CNNs) $f_\delta : \mathbb{R}^{L\times M\times C} \to \mathbb{R}^{L\times M\times N_c}$, where N_c is the number of classes and whose parameters δ are learned from a training set $X_{\text{tr}} = \{(x_1, y_1), (x_2, y_2), ..., (x_{N_{\text{tr}}}, y_{N_{\text{tr}}})\}$, with $N_{\text{tr}} = |X_{\text{tr}}|$.

Few-shot image segmentation requires an initial training set X_{tr}, with the set of classes C_{tr}, which is used to train the model's parameters. The model is employed to predict a new set of classes $C_{\text{te}} : C_{\text{te}} \cap C_{\text{tr}} = \emptyset$ by only relying on K annotated samples (shots) for each of the new N classes (ways), where $N = |C_{\text{te}}|$, making the segmentation task an N-way K-shot problem. For the remainder of this chapter, we will consider 1-way segmentation problems, i.e., the binary segmentation of the new class. The K annotated shots comprise the *support set* $X_s = \{(x_s, y_s)\}$ from which the model distils knowledge about the new class to produce a segmentation of the unannotated *query set* $X_q = \{x_q\}$.

For our method, summarized in Figure 4.1, we consider feature extractor CNNs capable of generating feature maps at the same spatial resolution as the input image. We train the feature extractors on the classes known in the training set. Specifically, in this work, we train a binary segmentation model for each of these known classes. Consequently, we use the feature maps of the images from the support set generated by the trained feature extractors to learn a set of weights for recombining the maps to predict the new class. Finally, we employ the feature extractors and the learned weights to predict the query images. In this work, we choose MSD networks [116] as feature extractors. MSD bypasses the need for downscaling and upscaling the feature maps for capturing features at different scales by replacing standard convolutional kernels with dilated convolutions [70]. Since each feature map has the same spatial dimensions as the input image, this network can localize well the individual cells, generating activation areas that correspond to the actual position and shape of the cells within the image. Preserving the spatial dimensions of the feature maps also enables the network to densely connect its layers, allowing MSD to produce accurate results with relatively few trainable parameters. The low parameter count implies that MSD is less prone to overfitting [158], making it a well-suited feature extractor for data-scarce domains, such as medical image or cell segmentation.

We decompose the feature extractor network f_δ into a feature maps generator g_ϕ and a predictor o_ϵ. Therefore, we have $f_\delta = o_\epsilon \circ g_\phi$, where $g_\phi : \mathbb{R}^{L\times M\times C} \to \mathbb{R}^{L\times M\times C'}$ uses the parameters ϕ to generate C' feature maps from the input image and $o_\epsilon : \mathbb{R}^{L\times M\times C'} \to \mathbb{R}^{L\times M\times N_c}$, parametrized by ϵ, outputs the prediction from the feature

maps, with $\phi \cup \epsilon = \delta$. We begin by training the feature extractor on the training set:

$$\widehat{\phi}, \widehat{\epsilon} = \underset{\phi,\epsilon}{\operatorname{argmin}} \sum_{(x,y)\in X_{\mathrm{tr}}} L(o_\epsilon(g_\phi(x)), y), \tag{4.1}$$

where L is a loss function. Consequently, we employ the feature maps generator to learn the weights $W \in \mathbb{R}^{C'}$ and intercept $b \in \mathbb{R}$ of a perceptron in the few-shot adaptation step:

$$\widehat{b}, \widehat{W} = \underset{b,W}{\operatorname{argmin}} \sum_{(x,y)\in X_{\mathrm{s}}} L(\sigma(b + g_{\widehat{\phi}}(x) \cdot W), y) + \lambda \|W\|_2, \tag{4.2}$$

where $\sigma : \mathbb{R}^{L\times M} \to \mathbb{R}^{L\times M}$ is an element-wise activation function. The $\|W\|_2$ regularization term is included because we noticed its benefit in 1-shot cases, where overfitting can become more likely. Equation 4.2 enables us to create a new linear combination of the feature maps, suitable for the new class in the support set. Finally, we apply the weights to predict the query images as $\widehat{y} = \sigma(\widehat{b} + g_{\widehat{\phi}}(x) \cdot \widehat{W})\ \forall x \in X_q$.

When utilizing a cross-entropy loss, Equation 4.2 becomes a logistic regression task [108] for which highly efficient implementations are available [53]. For other loss functions, e.g., Dice loss, we use a second-order optimizer, which has several advantages compared to first-order approaches (e.g., faster convergence and better robustness to hyperparameter settings [171]). Second-order optimizers are typically not suitable for deep learning due to their high computational costs when optimizing a large number of parameters. However, the number of weights of our perceptron is relatively low, making second-order optimization a viable choice. Since second-order optimization methods perform best when the initial guess of the parameters is close to the optimum [12], we use logistic regression to provide this initial guess.

4.3 Experiments

4.3.1 Experimental Setup

We implemented our experiments in PyTorch [114]. For training the feature extractors, we partition our data into training and validation with an 80/20 ratio and stop the training when the validation score does not improve for more than 10 consecutive epochs. We use the Dice loss function and ADAM [82] optimizer. For obtaining the perceptron's weights W and intercept b, we employ BFGS [169] with Dice loss as the objective function. We choose the regularization parameter in Equation 4.2 by visually

Method	ABD-MRI*	ABD-CT*	Lizard†	MoNuSAC†
PANet	46.75	28.95	10.08	27.77
SE-Net	47.45	39.242	10.16	23.12
GCN-DE	67.3	61.73	19.04	21.79
SSL-ALPNet	70.12	65.05	6.01	18.59
BAM	-	-	5.4	9.77
Ours	-	-	48.76	48.27

Table 4.1: The average Dice score [%] on the test set of state-of-the-art medical and natural image few-shot segmentation models. $*$: Results taken from [170]. †: Results generated by following the open-source implementation of the methods.

assessing the predictions of several random selections of support and query images. The optimization stops when the gradient norm is lower than 10^{-5}. To report the results, we use the Dice coefficient on a separate test set, using 5 instances of trained feature extractors with 10 randomly sampled support sets (50 results) per experiment.

4.3.2 Data

We chose Lizard [59] and MoNuSAC [153] segmentation data sets, containing, respectively, 291 (191 train, 100 test) and 410 (310 train, 100 test) 8-bit RGB H&E stained tissue images of various sizes. From Lizard, we keep epithelial (E), connective (C), lymphocyte (L), and plasma (P) classes, whereas from MoNuSAC we use epithelial (E), lymphocyte (L), macrophage (M), and neutrophil (N). For both data sets, we separate each image into multiple 256×256 patches via a sliding window technique with a stride of 64 pixels. Additionally, to show the performance of the FSL methods designed for medical image segmentation, we use ABD-CT from [89] (30 3D CT scans with 1755 slices) and ABD-MRI from [80] (20 3D MRI scans with 492 slices). From both data sets, we report the results on four classes: liver, spleen, left kidney, and right kidney.

4.3.3 Results

Existing Works on Cell Segmentation. We used open-source implementations, provided by their respective authors, of SE-Net [127] and PANet [160], two methods that constitute the seminal works in medical imaging for attention-guided few-shot segmentation, and for prototype learning, respectively, as well as of GCN-DE [138] and SSL-ALPNet [112], two well-performing derivations of SE-Net and PANet, respectively. Also, we explore the results of BAM [90], a recent method with state-of-the-art

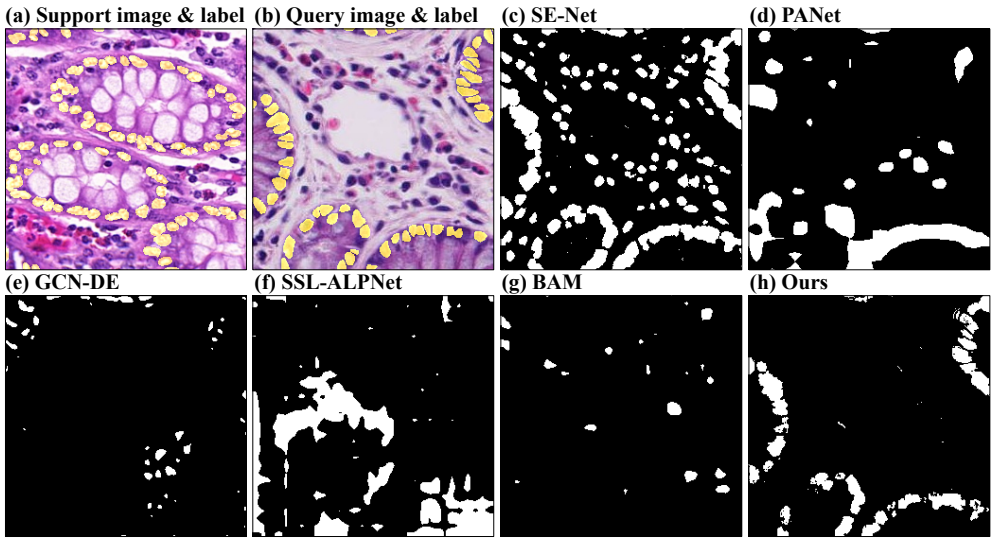

Figure 4.2: Visual comparison of predictions of epithelial cells from Lizard by methods designed for natural and/or medical image segmentation, and our method.

results for natural image segmentation. We applied these techniques on the cell data sets whose results we report in Table 4.1. The results on ABD-MRI and ABD-CT, taken from [170], correspond to a one-shot setting, while for Lizard and MoNuSAC we used five shots in the support set for PANet, BAM, GCN-DE and our method, while for SE-Net and SSL-ALPNet, we employed one support image since these methods do not allow pairing a query image with multiple support images. Since BAM has not been applied to medical imaging, we do not show results for it on ABD-MRI and ABD-CT. We notice that although these methods show good results for the task they were designed for, i.e., few-shot organ segmentation, their performances do not translate to few-shot cell segmentation. In this context, they achieve considerably lower scores compared to our method. Figure 4.2 shows a qualitative comparison between the aforementioned methods and our solution on the Lizard data set where the unknown class is the epithelial cell type. The other cell classes were used during training. For the methods allowing multiple shots, we used five. For the others, we used one shot. In Figure 4.2, we only show the support image common to all methods. We observe that SE-Net, GCN-DE, and BAM show difficulties in adapting to the new cell type. SE-Net segments most cell-like structures within the query image, whereas the predictions of GCN-DE and BAM contain structures belonging to the cell types from the training set. The prototype-based solutions, i.e., PANet and SSL-ALPNet, show

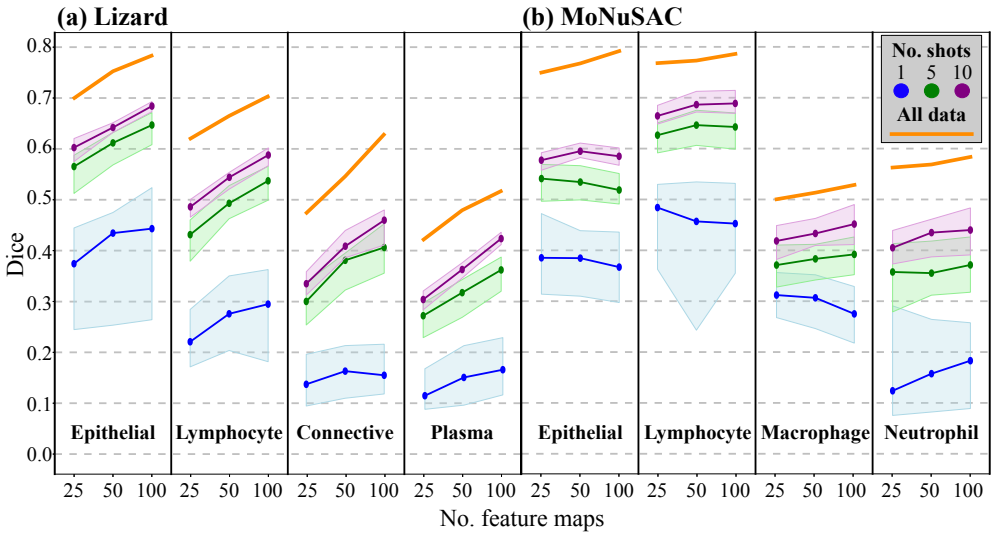

Figure 4.3: Figures (**a, b**) show the Dice score of our method for different numbers of shots and feature maps. The orange line shows the average Dice score on the test set of the models trained on all labelled data. The data points reflect the median Dice score on the test set, while the shaded area is defined by the first and the third quartiles.

comparatively better adaptability. PANet includes a region with the epithelial cells in its predictions, with the same area also covered by SSL-ALPNet. However, both methods present relatively large areas of false positive segmentation. In this example, our method shows the best coverage of the target cells, while introducing the least amount of false positives.

Performance Analysis. For this experiment, we trained feature extractors of 25, 50, and 100 layers, on three classes, while selecting 1, 5, and 10 images whose labels we include in the support set. We trained the feature extractors as individual binary segmentation networks, each covering one of the classes; four binary segmentation networks in total. For every target class, we include the feature extractors trained on the other classes within the few-shot adaptation step. We present the results generated by the feature extractor trained on the target class as an upper-bound baseline. The results reported in Figure 4.3 show that offering the perceptron more features for recombination, i.e., training deeper feature extractors, generally results in improved Dice score, especially for higher numbers of shots. This behaviour is expected, since a richer set of features means more flexibility in choosing a more general combination separating the new cell type from the other structures in the image. Also, we notice that, due to the random nature of the few-shot allocation, adapting the perceptron

Optimization	Lizard				MoNuSAC			
	E	L	C	P	E	L	M	N
Logistic [%]	61.37	50.26	34.57	32.48	49.1	63.63	37.6	37.54
BFGS [%]	**64.66**	**53.89**	**40.27**	**36.21**	**51.86**	**64.45**	**39.18**	**37.57**

Table 4.2: Median Dice score results on the test with weights trained by logistic regression and further refined by BFGS.

on only one shot is detrimental to its performance because of the risk of picking a less representative or less informative support image, such as an image with a small amount of annotated cells. This issue is reflected in the large blue-shaded areas corresponding to 1-shot results, showing high variability in the performance of our method. However, when utilizing 5 shots, we notice more robust results that are less affected by the chosen support set. Also, since we learn the perceptron's weights using a second-order optimizer, the adaptation step is performed quickly, averaging 9 seconds on a Nvidia RTX 3070 GPU, allowing eventual flaws in choosing the support set to be quickly detected and corrected in practice.

Few-Shot Adaptation Choice. We assessed the benefit of utilizing the weights obtained via logistic regression as a starting point for a second-order optimization algorithm with Dice loss. We conducted the experiments by randomly selecting 5 support images of the target class on which the perceptron was trained via logistic regression. Consequently, we employed BFGS to optimize the resulting weights with the Dice loss as the optimization function. Table 4.2 illustrates that the additional optimization step is beneficial for the performance of our method with an average Dice score gain of 10.14% for Lizard and 2.8% for MoNuSAC.

4.4 Conclusion

Cell segmentation is an important annotation-scarce task that can benefit from few-shot learning, but for which existing methods are unsuitable. Here, we present a novel few-shot segmentation method designed to account for the particularities of cell segmentation, such as the varying position of the target structures and their proximity to each other. To achieve this, we utilize the high-resolution feature maps generated by MSD networks [116], trained on the known classes, as input to a perceptron, which we adapt to the few shots of the new class. We showed that our method can be successfully applied to cell images, requiring as little as five annotated images in the support set for producing Dice scores less than 20% lower than of models trained on several

hundred annotated images. In the future, we aim to improve the reliability of our solution by exploring other types of feature extractors, incorporating additional regularization techniques, or using ensemble methods. Moreover, to better contextualize our results, we intend to provide additional comparisons with popular fully-supervised cell segmentation methods such as UNet [125] and Hover-Net [60].

Besides being used as a standalone cell segmentation tool, our solution can also be embedded into an active learning setup where the quick adaptation step would enable the user to immediately choose an appropriate support set where its predictions can constitute the base for a further refinement step, e.g. as in [159]. In both cases, our method can significantly reduce the amount of training annotations necessary for costly segmentation tasks. For instance, within a semi-automated annotation tool, our solution can produce initial suggestions of annotations, which can then quickly be corrected by experts, while a fully-supervised model trains in the background on the rectified annotations such as in [61].