# MSDNet: Multi-Scale Decoder for Few-Shot Semantic Segmentation via Transformer-Guided Prototyping

Amirreza Fateh<sup>1</sup>, Mohammad Reza Mohammadi<sup>1,\*</sup>, Mohammad Reza Jahed Motlagh<sup>1</sup> School of Computer Engineering, Iran University of Science and Technology (IUST), Tehran, Iran \* Corresponding author. mrmohammadi@iust.ac.ir

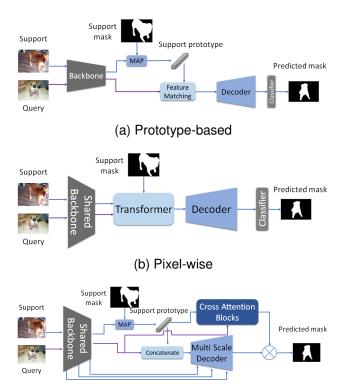
Abstract—Few-shot Semantic Segmentation addresses the challenge of segmenting objects in query images with only a handful of annotated examples. However, many previous state-of-the-art methods either have to discard intricate local semantic features or suffer from high computational complexity. To address these challenges, we propose a new Few-shot Semantic Segmentation framework based on the transformer architecture. Our approach introduces the spatial transformer decoder and the contextual mask generation module to improve the relational understanding between support and query images. Moreover, we introduce a multi-scale decoder to refine the segmentation mask by incorporating features from different resolutions in a hierarchical manner. Additionally, our approach integrates global features from intermediate encoder stages to improve contextual understanding, while maintaining a lightweight structure to reduce complexity. This balance between performance and efficiency enables our method to achieve state-of-the-art results on benchmark datasets such as  $PASCAL-5^{i}$  and  $COCO-20^{i}$ in both 1-shot and 5-shot settings. Notably, our model with only 1.5 million parameters demonstrates competitive performance while overcoming limitations of existing methodologies. https://github.com/amirrezafateh/MSDNet

Index Terms—Few-shot learning, few-shot segmentation, Semantic Segmentation, Prototype generation

## I. INTRODUCTION

Semantic segmentation is a key task in computer vision, where each pixel of an image is labeled as part of a specific category. This is important in many areas like OCR, autonomous driving, medical imaging, and scene understanding [1]–[3]. To perform this task well, models need to learn detailed object boundaries. In recent years, deep Convolutional Neural Networks (CNNs) have made big improvements in this area [4]. However, these high-performing models usually need large datasets with lots of labeled examples [5], [6], which takes a lot of time and effort to create. In real-world scenarios, like in medical imaging or other fields where labeled data is limited, this becomes a big problem [7], [8]. To solve this, Few-shot Semantic Segmentation (FSS) has become a useful approach.

FSS tries to segment new object classes in images using only a few labeled examples, called support images, that show the target class [9]. This method helps reduce the need for large datasets, making it more practical for real-world use [10]. Addressing the challenges of FSS requires handling differences in texture or appearance between the target object



(c) Cross-Domain Multi scale decoder with transformer guided prototyping

Fig. 1. Comparison among existing methods and our proposed method for FSS. (a) Prototype-based methods; (b) Pixel-wise approach; (c) Cross-Domain Multi scale decoder with transformer guided prototyping.

in the query image and similar objects depicted in the support examples. Effectively using the relationship between the query image and the support examples is essential in tackling FSS.

FSS can be widely categorized into two groups: Prototype-based approaches and Pixel-wise methods. As shown in Figure 1-(a), prototype-based approaches involve abstracting semantic features of the target class from support images through a shared backbone network [11]. This process results in feature vectors called class-wise prototypes, which are obtained using techniques such as class-wise average pooling or clustering. These prototypes are then combined with query features through operations like element-wise summation or channel-wise concatenation. The combined features are refined by a

decoder module to classify each pixel as either the target class or background [12]. In contrast, as shown in Figure 1-(b), pixel-wise methods take a different approach by focusing directly on pixel-level information rather than compressing it into prototypes. These methods aim to predict the target class for each pixel in the query image by comparing it directly with corresponding pixels in the support images. To achieve this, they establish pixel-to-pixel correlations between the support and query features, which allows the model to find precise matches even when the object's appearance varies [13]. This process is often enhanced by attention mechanisms, like those found in Transformer models, which help the model focus on important relationships between pixels. By avoiding the need for prototypes, Pixel-wise methods aim to preserve more detailed information, allowing for finer-grained segmentation [14], [15].

While both groups have demonstrated efficacy, they also have certain limitations. Prototype-based methods may inadvertently discard complex local semantic features specific to the target class in support images. This can lead to coarse segmentation of the target class in query images, especially for objects with complex appearances. On the other hand, while pixel-wise methods have notably improved performance compared to prototype-based approaches, they grapple with computational complexity due to dot-product attention calculations across all pixels of support features and query features. Moreover, a large amount of pixel-wise support information can lead to confusion in attention mechanisms [13]. Also, a shared limitation across both approaches is the lack of use of encoder middle features in the decoder section. Many methods in both categories employ straightforward decoders that fail to incorporate encoder middle features. However, in few-shot scenarios where data samples are limited, leveraging the global features captured by the encoder in the decoder phase can prove to be highly beneficial.

Inspired by recent developments, we aim to develop a straightforward and effective framework to address limitations in FSS methods. A notable approach gaining traction is the Query-based<sup>1</sup> Transformer architecture, which has demonstrated versatility across various computer vision tasks, including few-shot learning scenarios [16], [17]. This architecture utilizes learnable Query embeddings derived from support prototypes, enabling nuanced analysis of their relationships within the query feature map.

Inspired by previous works, as shown in Figure 1-(c), we have designed a novel Transformer-based module, known as the Spatial Transformer Decoder (STD), to enhance the relational understanding between support images and the query image. This module operates concurrently with the multi-scale decoder. Within the STD module, we introduce a common strategy: Using the prototype of support images as a Query, while utilizing the features extracted from the query image as both Value and Key embeddings inputted into the Transformer decoder. This formulation allows the Query to effectively focus on the semantic features of the target class within the query

<sup>1</sup>For differentiating it from the conventional term "query" frequently employed in FSS, we capitalize "Query" when referring to the query sequence within the Transformer architecture.

image. Furthermore, to reduce the impact of information loss resulting from the abstraction of support images into a feature vector named the 'support prototype,' we integrate global features from the intermediate stages of the encoder, which are fed with the support images, into our decoder. Incorporating these features allows us to leverage features from different stages of the encoder, thereby enriching the decoder's contextual understanding. Additionally, we introduce the Contextual Mask Generation Module (CMGM) to further augment the model's relational understanding, operating alongside the STD and enhancing the model's capacity to capture relevant contextual information.

#### II. RELATED WORKS

#### A. Semantic Segmentation

Semantic segmentation, a crucial task in computer vision, involves labeling each pixel in an image with a corresponding class [18], [19]. CNNs significantly advanced semantic segmentation by replacing fully connected layers with convolutional layers, enabling the processing of images of various sizes [20]. Since then, subsequent advancements have focused on enhancing the receptive field and aggregating long-range context in feature maps. Techniques such as dilated convolutions [21], spatial pyramid pooling [22], and non-local blocks [23] have been employed to capture contextual information at multiple scales. More recently, Transformer-based backbones, including SegFormer [24], Segmenter [25], and SETR [26], have been introduced to better capture long-range context in semantic segmentation tasks. Further enhancing this approach, hierarchical architectures like the Swin Transformer [27] have achieved state-of-the-art performance by using shifted windows in their general-purpose backbones. In parallel, selfsupervised pretraining strategies, such as the masked image modeling used in BEiT [28], have also shown strong results, fine-tuning directly on the semantic segmentation task and pushing the boundaries of model performance.

Semantic segmentation tasks typically involve per-pixel classification. as demonstrated by approaches like Mask-Former [29] and Mask2Former [30], which predict binary masks corresponding to individual class labels. Older architectures, such as UNet [31], PSPNet [32], and Deeplab [33], [34], have also significantly contributed to the field by incorporating features like global and local context aggregation and dilated convolutions to increase the receptive field without reducing resolution. Building upon these foundational approaches, more recent studies, including CRGNet [35] and SAM [36], have focused on further improving model performance, exploring new techniques to enhance accuracy in segmentation tasks. Despite the progress made in per-pixel classification, addressing the challenge of segmenting unseen classes remains an open area for future research

## B. Few-Shot Semantic Segmentation

FSS is a challenging task in computer vision, wherein the objective is to segment images with limited annotated examples, known as support images. Approaches to FSS can be categorized into various groups based on their primary aims and methodologies employed [37]. One significant challenge in FSS is addressing the imbalance in details between support and query images. Methods like PGNet [38] and PANet [39] aim to eliminate inconsistent regions between support and query images by associating each query pixel with relevant parts of the support image or by regularizing the network to ensure its success regardless of the roles of support and query. But methods like ASGNet [37], on the other hand, focuses on finding an adaptive quantity of prototypes and their spatial expanses determined by image content, utilizing a boundary-conscious superpixel algorithm.

Another critical aspect of FSS is bridging the inter-class gap between base and novel datasets. Approaches like RePRI [40] and CWT [41] address this gap by fine-tuning over support images or episodically training self-attention blocks to adapt classifier weights during both training and testing phases. Additionally, architectures designed for supervised learning often trouble recognizing objects at different scales in few-shot scenarios. To address this issue, new methods have been developed to allow information exchange between different resolutions [42].

Moreover, ensuring the reliability of correlations between support and query images is essential in FSS. Methods like HSNet [43] and CyCTR [44] utilize attention mechanisms to filter out erroneous support features and focus on beneficial information. VAT [45], meanwhile, employs a cost aggregation network to aggregate information between query and support features, leveraging a high-dimensional Swin Transformer to impart local context to all pixels.

Overall, the field of FSS is advancing rapidly with innovative methods aimed at enhancing model performance and overcoming challenges in adapting segmentation models to novel classes with limited annotated data. These efforts are driven by the ongoing need to improve the effectiveness and versatility of segmentation models in real-world applications.

#### III. PROPOSED METHOD

## A. Problem Definition

In FSS, the task involves segmenting images belonging to novel classes with limited annotated data. We operate with two datasets,  $D_{train}$  and  $D_{test}$ , each associated with class sets  $C_{train}$  and  $C_{test}$ , respectively. Notably, these class sets are disjoint  $(C_{train} \cap C_{test} = \emptyset)$ , ensuring that there is no overlap between the classes in the training and test datasets. Each training episode consists of a support set S and a query set S, where S includes a set of S support images along with their corresponding binary segmentation masks, while S contains a single query image. The model is trained to predict the segmentation mask for the query image based on the support set.

Both  $D_{train}$  and  $D_{test}$  consist of a series of randomly sampled episodes (an episode is defined as a set comprising support images and a query image. During each epoch, we can have many episodes (e.g., 1000 episodes), each containing its own set of support and query images). During training, the model learns to predict the segmentation mask for the query image based on the support set. Similarly, during testing, the

model's performance is evaluated on the  $D_{test}$  dataset, where it predicts the segmentation mask for query images from the test dataset using the knowledge learned during training.

Overall, the goal of FSS is to develop a model that can accurately segment images from novel classes with only a few annotated samples, demonstrating robust generalization capabilities across different datasets and unseen classes.

## B. Overview

Given a support set  $S = \left\{I_s^i, M_s^i\right\}$  and a query image  $I_q$ , the objective is to generate the binary segmentation mask for  $I_q$ , identifying the same class as the support examples. To address this task, we introduce a straightforward yet robust framework, outlined in Figure 2. For simplicity, we illustrate a 1-shot setting within the framework, but this can be easily generalized to a 5-shot setting as well. The proposed method comprises several key components, including a shared pretrained backbone, support prototype, CMGM, a multi-scale decoder, and STD. These elements collectively contribute to the model's ability to accurately segment objects of interest in the query image based on contextual information provided by the support set. In the following, we'll take a closer look at each component, explaining its role and how it interacts within our framework.

1) Backbone: In our proposed framework, we adopt a modified ResNet architecture, initially pre-trained on the ImageNet dataset, to serve as the backbone for feature extraction from raw input images, ensuring that the size of the output of each block does not reduce below a specified dimension. For instance, like [46], we define that the output sizes from  $conv2\_x$  to  $conv5\_x$  are maintained at  $60 \times 60$  pixels. Specifically, we utilize a ResNet with shared weights between support and query images. This type of ResNet maintains the spatial resolution of feature maps at  $60 \times 60$  pixels from the  $conv2\_x$  stage forward, preserving finer details crucial for accurate segmentation. We extract high-level features  $(conv5\_x)$ , as well as mid-level features  $(conv3\_x)$  and  $conv4\_x)$  from both support and query images using the backbone.

The mid-level features of the support image are denoted as  $X_s^{conv3}$  and  $X_s^{conv4}$ , while the high-level features are denoted as  $X_s^{conv5}$ . Similarly, for the query image, the mid-level features are represented as  $X_q^{conv3}$  and  $X_q^{conv4}$ , and the high-level features as  $X_q^{conv5}$ . To integrate mid-level features across different stages, we concatenate the mid-level feature maps from  $conv3\_x$  and  $conv4\_x$  stages and apply a  $1\times 1$  convolution layer to yield a merged mid-level feature map, denoted as  $X_s^{merged}$ . This merging process ensures that the resultant feature map retains essential information from both mid-level stages, enhancing the model's ability to capture diverse contextual information (Equation 1, Equation 2).

$$X_s^{merged} = C_{1 \times 1}(Cat(X_s^{conv3}, X_s^{conv4})) \tag{1}$$

$$X_q^{merged} = C_{1\times 1}(Cat(X_q^{conv3}, X_q^{conv4}))$$
 (2)

Where Cat denotes concatenation along the channel dimension, and  $C_{1\times 1}$  denotes the  $1\times 1$  convolution operation. These equations illustrate the process of merging mid-level features from different stages of the backbone network, resulting in a

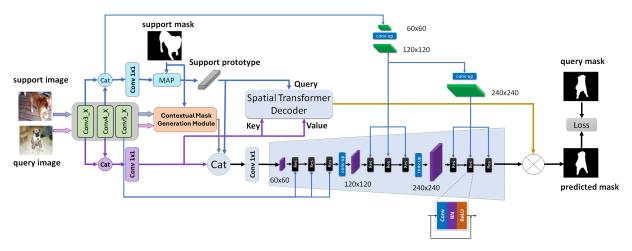


Fig. 2. the overview of the proposed method

combined mid-level feature map that retains crucial information from both stages.

The decision to employ this modified ResNet architecture is grounded in its ability to balance computational efficiency with feature representation. By maintaining the feature map size at  $60 \times 60$  pixels, the backbone effectively captures detailed spatial information while avoiding excessive computational overhead. This approach strikes a pragmatic balance between model complexity and segmentation performance, making it well-suited for our few-shot segmentation task, where computational efficiency is paramount.

2) Support Prototype: In our proposed framework, the Support Prototype serves as a condensed representation of the mid-level features extracted from the support example  $(X_s^{merged})$ . The Support Prototype is obtained by applying a Masked Average Pooling (MAP) operation, which selectively aggregates information based on the support mask. Mathematically, the Support Prototype  $P_s$  is defined in Equation 3.

$$P_s = F_{pool}(X_s^{merged} \odot M_s) \tag{3}$$

Where  $F_{pool}$  represents the average pooling operation, and  $\odot$  signifies element-wise multiplication (Hadamard product) with the support mask  $M_s$ . The MAP operation involves computing the average pooling of the masked feature map, focusing solely on regions of interest specified by the support mask. This results in the generation of the Support Prototype, which encapsulates essential semantic information from the support example, facilitating effective few-shot segmentation.

3) Contextual Mask Generation Module (CMGM): The CMGM is a novel component introduced by our framework, designed to enhance the contextual understanding between support and query images in FSS tasks. As shown in Figure 3, CMGM leverages the feature representations extracted from both the support and query images to generate a contextual mask that encapsulates pixel-wise relations indicative of the target object. This process involves computing the cosine similarity between the query feature vector and the support feature vector. Mathematically, cosine similarity cos(q,s) is calculated as the dot product of the normalized query and support feature vectors. In a five-shot scenario, where there

are five support examples, five cosine similarities are computed and subsequently averaged, yielding a novel cosine similarity measure representative of the collective support set.

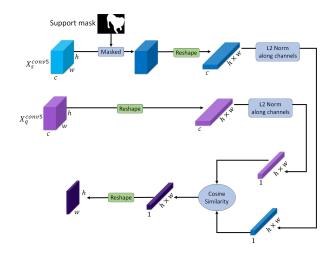


Fig. 3. the overview of CMGM

4) Multi Scale Decoder: The multi scale decoder in our proposed method is a critical component designed to refine the segmentation mask by incorporating features from different resolutions in a hierarchical manner. The decoder consists of three stages, each comprising two residual layers. Input feature map undergoes a sequence of convolutional operations within residual layers to gradually upsample the mask image.

As shown in Figure 2, in the first stage of the decoder, the input feature map has a size of  $60 \times 60$  pixels. This stage begins with two residual layers applied to the input feature map. Each residual layer receives input from combination of the previous layer's output and  $X_s^{conv5}$ . Following these layers, a convolutional operation is employed to upsample the mask image to a resolution of  $120 \times 120$  pixels.

Second stage of the decoder, which operates on a feature map size of  $120 \times 120$  pixels, has two residual layer like the first stage. Each residual layer takes input from combination of the previous layer's output and the merged mid-level features  $(X_s^{merged})$  obtained from the support image's encoder. Since

the size of  $X_s^{merged}$  remains at  $60 \times 60$  pixels, it is upsampled to  $120 \times 120$  pixel resolution using a convolutional layer. This upsampled feature map, denoted as  $X_{s(120 \times 120)}^{merged}$ .

Finally, in the third stage of the decoder, which operates on a feature map size of  $240\times240$  pixels, the input to each residual layer comprises the output from the combination of preceding layer and the upsampled  $X_s^{merged}$  feature map. in this stage  $X_{s(120\times120)}^{merged}$ , upsamples to  $240\times240$  pixel resolution, denoted as  $X_{s(240\times240)}^{merged}$ . This upsampled feature map is then integrated with the output from the preceding layer to form the input for subsequent processing.

Notably, one of the distinctive aspects of our multi-scale decoder is the incorporation of mid-level and high-level features from the encoder, like U-Net architecture. Specifically, in each stage of the decoder, the input to the residual layers combines the output from the previous layer with either the  $conv5\_x$  features (the output of the last block of the encoder) or the merged mid-level features ( $X_s^{merged}$ ) extracted from the support image's encoder. This fusion of features from different levels of abstraction enhances the decoder's ability to capture both detailed and contextual information essential for accurate segmentation.

5) Spatial Transformer Decoder (STD): In parallel with the multi-scale decoder module, STD plays a pivotal role in refining the final segmentation mask. As illustrated in Figure 4, the STD module operates by leveraging multi-head cross-attention, focusing on target objects within the query features to generate semantic-aware dynamic kernels. This process begins by treating the support features as the Query embeddings, while the query features are utilized as the Key and Value embeddings within the STD. Through this strategic integration, the STD module adeptly captures intricate relationships between target objects present in the query features and their corresponding representations in the support features.

The architecture of the STD module employs multi-head cross-attention, rather than a traditional Transformer decoder paradigm. The prototype vector, representing the support features, is integrated as a Ouery, enriched with learnable positional encodings for heightened spatial context awareness. The query feature map serves as Key and Value embeddings for multi-head cross-attention, enabling comprehensive exploration of their interplay with the support features. Through this multi-head cross-attention process, the STD dynamically generates semantic-aware dynamic kernels crucial for finetuning segmentation predictions. The output of the STD module represents a segmentation mask embedding that captures the semantic information of the target objects within the query features. This embedding is crucial for refining the segmentation results. To integrate this information into the final segmentation output, the segmentation mask embedding is combined with the feature map of the output from the multiscale decoder using a dot-product operation. This operation efficiently merges the information from both modules, enhancing the overall segmentation accuracy.

#### C. Loss function

In our method, we employ the Dice loss function to train our model. This loss function measures the dissimilarity between the predicted segmentation mask M and the corresponding ground truth query mask  $M_q$ . The Dice loss is formulated in 4.

$$Dice\ Loss = 1 - \frac{2 \times |M \cap M_q|}{|M| + |M_q|} \tag{4}$$

Where  $|M \cap M_q|$  represents the intersection between the predicted and ground truth masks, and |M| and  $|M_q|$  denote the cardinality of the predicted and ground truth masks, respectively. Minimizing the Dice loss encourages the model to generate segmentation masks that closely match the ground truth masks, leading to more accurate segmentation results during training.

#### IV. EXPERIMENTAL RESULTS

#### A. Datasets

We evaluated our proposed method on two widely used datasets commonly employed in few-shot segmentation tasks:  $PASCAL - 5^{i}$  [57] and  $COCO - 20^{i}$  [53].

**PASCAL-5<sup>i</sup> Dataset.** The  $PASCAL-5^i$  dataset, introduced by Shaban et al. [57], is derived from the PASCAL VOC dataset [58], and augmented with the SDS [59]. The original PASCAL VOC dataset comprises 20 object categories. For  $PASCAL-5^i$ , these 20 categories are evenly divided into 4 subsets, each denoted as  $PASCAL-5^i$ . Consequently, each subset consists of 5 distinct object categories.

**COCO-20<sup>i</sup> Dataset.** The  $COCO-20^i$  dataset, introduced by Nguyen et al. [53], is derived from MSCOCO dataset [60]. The  $COCO-20^i$  dataset includes a total of 80 object categories. Similar to  $PASCAL-5^i$ , these 80 categories are divided into 4 subsets, with each subset denoted as  $COCO-20^i$ . Each subset contains 20 distinct object categories. Notably,  $COCO-20^i$  presents a greater challenge due to its larger number of categories and images compared to  $PASCAL-5^i$ .

Cross-Validation Training. To ensure robust evaluation, we adopted a cross-validation training strategy commonly employed in few-shot segmentation literature. Specifically, we divided each dataset into four subsets. Three subsets were utilized as training sets, while the remaining subset served as the test set for model evaluation. During testing, we randomly selected 1000 support-query pairs from the test set for evaluation.

## B. Experimental Setting

We implemented our proposed method using PyTorch version 1.8.1. For feature extraction, we employed pretrained ResNet-50 and ResNet-101 backbones, which were originally trained on the ImageNet dataset. During training, the parameters of these pretrained models were frozen, and only the newly added modules were trainable. For training on the  $COCO-20^i$  dataset, we conducted training for each fold over 30 epochs. Conversely, for the  $PASCAL-5^i$  dataset, training was extended to 60 epochs to ensure optimal convergence.

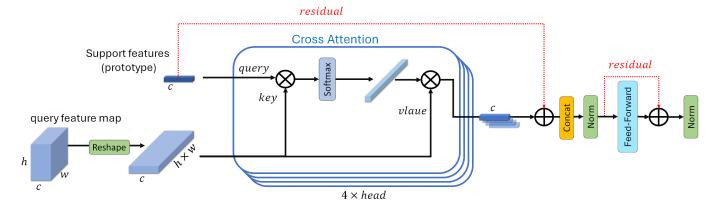


Fig. 4. Spatial Transformer Decoder

TABLE I Performance on  $PASCAL-5^i$  in terms of mIoU and FB-IoU. Numbers in bold represent the best performance, while underlined values denote the second-best performance.

Backbone	Methods	Publication			1-	shot					5-	shot			# learnable
Баскоопе	Wiethous	Publication	fold0	fold1	fold2	fold3	mean	FB-IoU	fold0	fold1	fold2	fold3	mean	FB-IoU	params
	PANet [39]	ICCV19	44.0	57.5	50.8	44.0	49.1	-	55.3	67.2	61.3	53.2	59.3	-	23.5M
ResNet50	PGNet [38]	ICCV19	56.0	66.9	50.6	50.4	56.0	69.9	57.7	68.7	52.9	54.6	58.5	70.5	17.2M
	PFENet [47]	TPAMI20	61.7	69.5	55.4	56.3	60.8	73.3	63.1	70.7	55.8	57.9	61.9	73.9	10.3M
	PPNet [48]	ECCV20	48.6	60.6	55.7	46.5	52.8	69.2	58.9	68.3	66.8	58.0	63.0	75.8	31.5M
	RePRI [40]	CVPR21	59.8	68.3	62.1	48.5	59.7	-	64.6	71.4	71.1	59.3	66.6	-	-
	HSNet [43]	ICCV21	64.3	70.7	60.3	60.5	64	76.7	70.3	73.2	67.4	67.1	69.5	80.6	2.5M
	CyCTR [44]	NeurIPS21	65.7	71.0	59.5	59.7	64.0	-	69.3	73.5	63.8	63.5	67.5	-	15.4M
	NTRENet [49]	CVPR22	65.4	72.3	59.4	59.8	64.2	77.0	66.2	72.8	61.7	62.2	65.7	78.4	19.9M
	ABCNet [50]	CVPR23	62.5	70.8	57.2	58.1	62.2	74.1	64.7	73.0	57.1	59.5	63.6	74.2	-
	QGPLNet [51]	ACM TOMM23	56.95	68.99	60.1	54.98	60.25	-	61.78	70.96	69.56	58.26	65.14	-	-
	DRNet [52]	IEEE Trans. CSVT24	<u>66.1</u>	68.8	61.3	58.2	63.6	76.9	69.2	<u>73.9</u>	65.4	65.3	68.5	81.6	-
	MSDNet (our)	-	66.3	71.9	57.2	62.0	64.3	77.1	73.2	75.4	59.9	66.3	68.7	82.1	1.5M
	FWB [53]	ICCV19	51.3	64.5	56.7	52.2	56.2	-	54.8	67.4	62.2	55.3	59.9	-	43.0M
	PPNet [48]	ECCV20	52.7	62.8	57.4	47.7	55.2	70.9	60.3	70.0	<u>69.4</u>	60.7	65.1	77.5	50.5M
	PFENet [47]	TPAMI20	60.5	69.4	54.4	55.9	60.1	72.9	62.8	70.4	54.9	57.6	61.4	73.5	10.3M
ResNet101	HSNet [43]	ICCV21	67.3	72.3	62.0	63.1	66.2	77.6	71.8	<u>74.4</u>	67.0	68.3	<u>70.4</u>	80.6	2.5M
Resilettut	CWT [41]	ICCV21	56.9	65.2	61.2	48.8	58	-	62.6	70.2	68.8	57.2	64.7	-	-
	CyCTR [44]	NeurIPS21	69.3	72.7	56.5	58.6	64.3	73.0	73.5	74.0	58.6	60.2	66.6	75.4	15.4M
	NTRENet [49]	CVPR22	65.5	71.8	59.1	58.3	63.7	75.3	67.9	73.2	60.1	66.8	67.0	78.2	19.9M
	ABCNet [50]	CVPR23	62.7	70.0	55.1	57.5	61.3	73.7	63.4	71.8	56.4	57.7	62.3	74	-
	QGPLNet [51]	ACM TOMM23	59.66	69.77	65.15	55.9	62.64	-	65.05	72.75	71.12	59.85	67.19	-	-
	DRNet [52]	IEEE Trans. CSVT24	66.4	70.7	64.9	59.8	<u>65.3</u>	79.2	69.3	74.1	66.7	66.5	69.2	84.5	-
	MSDNet(our) -			72.8	58.2	60.0	64.7	77.3	75.5	77.2	62.5	68.1	70.8	85.0	1.5M

We utilized the Adam optimizer with a fixed learning rate of  $10^{-3}$ . All input images were resized to  $473 \times 473$  pixels, and the training batch size was set to 32 for the 1-shot setting and 16 for the 5-shot setting. Our training pipeline did not incorporate any data augmentation strategies. After prediction, the binary segmentation masks were resized to match the original dimensions of the input images for evaluation purposes. To ensure robustness and mitigate the effects of randomness, we averaged the results of three trials conducted with different random seeds. All experiments were performed on NVIDIA RTX 4090 GPU.

## C. Evaluation Metrics

We employ the following evaluation metrics to assess the performance of our proposed method:

**Mean Intersection over Union (mIoU).** mIoU is a widely used metric for evaluating segmentation performance. It calculates the average intersection over union (IoU) across all classes in the target dataset (Equation 5).

$$mIoU = \frac{1}{C} \sum_{i=1}^{C} IoU_i$$
 (5)

TABLE II PERFORMANCE ON  $COCO-20^i$  in terms of MIoU and FB-IoU. Numbers in bold represent the best performance, while underlined values denote the second-best performance.

Backbone	Methods	Publication	1-shot						5-shot						# learnable
васкоопе	Methods	Publication	fold0	fold1	fold2	fold3	mean	FB-IoU	fold0	fold1	fold2	fold3	mean	FB-IoU	params
	PPNet [48]	ECCV20	28.1	30.8	29.5	27.7	29.0	-	39.0	40.8	37.1	37.3	38.5	-	31.5M
	PFENet [47]	TPAMI20	36.5	38.6	34.5	33.8	35.8	-	36.5	43.3	37.8	38.4	39.0	-	10.3M
	HSNet [43]	ICCV21	36.3	43.1	38.7	38.7	39.2	68.2	43.3	51.3	48.2	45.0	46.9	70.7	2.5M
	CyCTR [44]	NeurIPS21	38.9	43.0	39.6	39.8	40.3	-	41.1	48.9	45.2	47.0	45.6	-	15.4M
	NTRENet [49]	CVPR22	36.8	42.6	39.9	37.9	39.3	68.5	38.2	44.1	40.4	38.4	40.3	69.2	19.9M
	BAM [54]	CVPR22	43.4	50.6	47.5	<u>43.4</u>	46.2	-	<u>49.3</u>	<u>54.2</u>	<u>51.6</u>	49.6	<u>51.2</u>	-	26.7M
	DCAMA [15]	ECCV22	41.9	45.1	44.4	41.7	43.3	69.5	45.9	50.5	50.7	46.0	48.3	71.7	47.7M
ResNet50	ABCNet [50]	CVPR23	36.5	35.7	34.7	31.4	34.6	59.2	40.1	40.1	39.0	35.9	38.8	62.8	-
Resirees	DRNet [52]	IEEE Trans. CSVT24	42.1	42.8	42.7	41.3	42.2	68.6	47.7	51.7	47.0	49.3	49.0	71.8	-
	QPENet [55]	IEEE Trans. Multimedia24	41.5	47.3	40.9	39.4	42.3	67.4	47.3	52.4	44.3	44.9	47.2	69.5	-
	PFENet++ [56]	TPAMI24	40.9	46.0	42.3	40.1	42.3	65.7	47.5	53.3	47.3	46.4	48.6	70.3	-
	MSDNet (our)	-	43.7	<u>49.1</u>	<u>46.9</u>	46.2	46.5	70.4	50.1	58.5	56.3	53.1	54.5	74.5	1.5M
	FWB [53]	ICCV19	17.0	18.0	21.0	28.9	21.2	-	19.1	21.5	23.9	30.1	23.7	-	43.0M
	PFENet [47]	TPAMI20	36.8	41.8	38.7	36.7	38.5	63.0	40.4	46.8	43.2	40.5	42.7	65.8	10.3M
	HSNet [43]	ICCV21	37.2	44.1	42.4	41.3	41.2	69.1	45.9	53.0	51.8	47.1	49.5	72.4	2.5M
	NTRENet [49]	CVPR22	38.3	40.4	39.5	38.1	39.1	67.5	42.3	44.4	44.2	41.7	43.2	69.6	19.9M
	DCAMA [15]	ECCV22	41.5	46.2	45.2	41.3	43.5	69.9	48.0	<u>58.0</u>	54.3	47.1	<u>51.9</u>	73.3	47.7M
ResNet101	ABCNet [50]	CVPR23	40.7	45.9	41.6	40.6	42.2	66.7	43.2	50.8	45.8	47.1	46.7	62.8	-
RESINCTIVI	DRNet [52]	IEEE Trans. CSVT24	43.2	43.9	43.3	43.9	43.6	69.2	52.0	54.5	47.9	49.8	51.1	73	-
	QPENet [55]	IEEE Trans. Multimedia24	39.8	45.4	40.5	40.0	41.4	67.8	47.2	54.9	43.4	45.4	47.7	70.6	-
	PFENet++ [56]	TPAMI24	42.0	44.1	41.0	39.4	41.6	65.4	47.3	55.1	50.1	<u>50.1</u>	50.7	70.9	-
	MSDNet (our)	-	44.5	52.5	48.9	48.1	48.5	71.3	50.4	59.9	57.6	53.3	55.3	75.1	1.5M

Here, C represents the number of classes in the target fold, and  $IoU_i$  denotes the intersection over union of class i.

**Foreground-Background IoU** (**FB-IoU**). FB-IoU measures the intersection over union specifically for the foreground and background classes. While FB-IoU provides insights into the model's ability to distinguish between foreground and background regions, we primarily focus on mIoU as our main evaluation metric due to its comprehensive assessment of segmentation performance.

# D. Comparison with SOTA

In this subsection, we compare our proposed method with several SOTA methods on both the  $PASCAL-5^i$  and  $COCO-20^i$  datasets. We present the results in Table I and Table II, respectively, where we report the mIoU and FB-IoU scores under both 1-shot and 5-shot settings, along with the final FB-IoU value. The results of other methods are obtained from their respective original papers.

**Results on PASCAL-5<sup>i</sup> Dataset.** As shown in Table I, our proposed method, utilizing ResNet50 and ResNet101 backbones, consistently surpasses SOTA methods in both 1-shot and 5-shot scenarios across all four folds of the  $PASCAL-5^i$  dataset. Notably, our method achieves the one of the highest performance across all folds.

**Results on COCO-20<sup>i</sup> Dataset.** Similarly, Table II presents the results on the  $COCO - 20^i$  dataset, where our pro-

posed method demonstrates superior performance under both ResNet50 and ResNet101 backbones in both 1-shot and 5-shot settings. Our method consistently outperforms all competing approaches across all four folds of the  $COCO-20^i$  dataset, consistently achieving either first or second rank. We obtained the highest mIoU scores in several folds and secured the second rank in others. Notably, our method exhibits superior performance in terms of mean and FB-IoU scores, further emphasizing its effectiveness and robustness.

It is important to highlight that our proposed method maintains a remarkably low number of learnable parameters, with only 1.5 million parameters. This stands in stark contrast to some SOTA methods, which possess significantly higher parameter counts, exceeding 40 million parameters in certain cases. This demonstrates the efficiency and effectiveness of our approach in achieving superior segmentation performance while maintaining a compact model architecture.

## E. Cross-dataset task

In this study, we investigate the cross-domain generalization capabilities of our proposed few-shot segmentation method through rigorous domain shift testing. Specifically, we trained our model on the  $COCO-20^i$  dataset and conducted testing on the  $PASCAL-5^i$  dataset to evaluate its adaptability across different datasets and domain settings.

TABLE III

Few-shot segmentation performance on cross-dataset task, " $COCO - 20^i \rightarrow PASCAL - 5^i$ ", in terms of mIoU, with different backbones (ResNet-50 and ResNet-101). Numbers in bold represent the best performance, while underlined values denote the second-best performance.

Backbone	Methods	Publication	1-shot					5-shot				
Dackboile	Methous	rublication	fold0	fold1	fold2	fold3	mean	fold0	fold1	fold2	fold3	mean
	PFENet [47]	TPAMI20	43.2	65.1	66.6	69.7	61.1	45.1	66.8	68.5	73.1	63.4
	RePRI [40]	CVPR21	52.2	64.3	64.8	71.6	63.2	56.5	68.2	70.0	76.2	67.7
	HSNet [43]	ICCV21	45.4	61.2	63.4	75.9	61.6	56.9	65.9	71.3	80.8	68.7
	HSNet-HM [61]	ECCV22	43.4	68.2	<u>69.4</u>	79.9	65.2	50.7	71.4	<u>73.4</u>	83.1	69.7
ResNet50	VAT-HM [61]	ECCV22	68.3	64.9	67.5	<u>79.8</u>	65.1	55.6	68.1	72.4	<u>82.8</u>	69.7
	RTD [62]	ECCV22	57.4	62.2	68.0	74.8	65.6	65.7	69.7	70.8	75.0	70.1
	PMNet [9]	WACV24	<u>68.8</u>	<u>70.0</u>	65.1	62.3	<u>66.6</u>	73.9	<u>74.5</u>	73.3	72.1	<u>73.4</u>
	MSDNet (our)	-	70.7	73.2	71.1	73.2	72.1	<u>72.5</u>	75.0	73.8	75.5	74.2
	HSNet [43]	ICCV21	47.0	65.2	67.1	<u>77.1</u>	64.1	57.2	69.5	72.0	82.4	70.3
	HSNetT-HM [61]	ECCV22	46.7	68.6	<u>71.1</u>	<b>79.7</b>	66.5	53.7	70.7	75.2	83.9	70.9
ResNet101	RTD [62]	ECCV22	59.4	64.3	70.8	72.0	66.6	67.2	72.7	72.0	78.9	72.7
	PMNet [9]	WACV24	<u>71.0</u>	<u>72.3</u>	66.6	63.8	<u>68.4</u>	75.2	<u>76.3</u>	77.0	72.6	<u>75.3</u>
	MSDNet (our)	-	71.6	75.6	73.0	75.2	73.9	<u>71.5</u>	79.6	<u>76.4</u>	77.9	76.4

The  $COCO-20^i$  dataset used in our experiments was modified to exclude classes and associated images that overlap with those present in  $PASCAL-5^i$ . This adaptation ensured that the training process focused on distinct visual concepts, thereby enhancing the model's exposure to novel classes during testing.

For our experiments, we adopted a cross-dataset evaluation protocol where models trained on each fold of  $COCO-20^i$  were repurposed for testing on the entire  $PASCAL-5^i$  dataset. Notably, during training, the model was exposed only to specific classes within  $COCO-20^i$ , ensuring no overlap with the classes present in  $PASCAL-5^i$ . This setup effectively simulates a scenario where the model encounters novel classes during testing that were not part of its training curriculum.

For instance, in the fold-0 setting, the model was exclusively trained on fold-0 of  $COCO-20^i$  and then assessed on the entirety of  $PASCAL-5^i$  after filtering out any classes that were encountered during training. This approach tests the model's ability to generalize to new and unseen classes in a different dataset domain.

Our experimental results, as detailed in Table III, demonstrate the superior performance of our proposed method compared to existing SOTA approaches under both 1-shot and 5-shot evaluation scenarios. This underscores the robustness and effectiveness of our few-shot segmentation framework in handling cross-dataset challenges and domain shifts.

#### F. Ablation Study

For the ablation study, we conduct experiments on the  $COCO-20^i$  dataset using the ResNet50 backbone in 1-shot scenario. Our first aim is to investigate the individual impact of various components on the segmentation performance. Table

IV show the impact of each component of the proposed method.

The first row of Table IV represents the performance of the baseline model, consisting solely of the backbone architecture and support prototype mechanism. Subsequent rows introduce additional components incrementally, including the CMGM, STD, and multi-scale decoder.

Baseline	CMGM	STD	Multi Scale			1	-shot		
Daseillie	CIVIGIVI	310	Decoder	fold0	fold1	fold2	fold3	mean	FB-IoU
<b>√</b>				30.1	34.2	33.4	33.8	32.9	59.7
✓	✓			31.5	35.9	34.8	34.2	34.1	60.8
✓	✓	✓		43.0	45.2	43.1	41.4	43.2	67.6
✓	✓	✓	✓	43.7	49.1	46.9	46.2	46.5	70.4

As shown in Table IV, each component contributes to an improvement in segmentation performance, with the Multi Scale Decoder showcasing the most substantial impact. The progressive integration of these components results in a notable enhancement in mIoU scores across all folds, underscoring their significance in refining segmentation masks and capturing contextual information effectively.

Furthermore, in Figure 5, we present the qualitative results obtained by incorporating each component into the baseline model on the  $COCO-20^i$  dataset. As illustrated in Figure 5, the addition of each component leads to noticeable improvements in the segmentation results. Particularly, the integration of the multi-scale decoder component demonstrates significant enhancement in segmentation accuracy.

To further explore the influence of the architecture within the multi-scale decoder, we conducted an ablation study varying the number of residual blocks in each stage. Figure

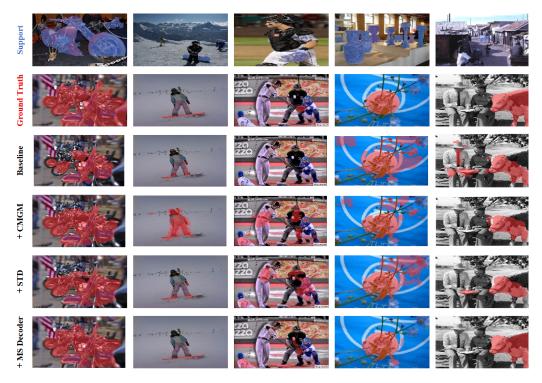


Fig. 5. Qualitative comparison of component effects on  $COCO - 20^i$  dataset in 1-shot scenario

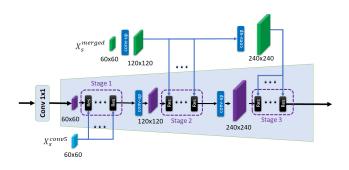


Fig. 6. The overview of Multi Scale Decoder with different number of residual blocks in each stage (1-4)

6 provides an overview of the Multi-Scale Decoder with different numbers of residual blocks in each stage. The experiment involved evaluating the segmentation performance on the  $COCO - 20^i$  dataset using the ResNet50 backbone in a 1-shot scenario. As depicted in Table V, we examined configurations ranging from one to four residual blocks per stage. Interestingly, the results revealed that the optimal segmentation performance was achieved with three residual blocks in each stage. This finding suggests that an appropriate balance in the depth of the decoder architecture plays a crucial role in enhancing segmentation accuracy. Too few blocks may limit the model's capacity to capture intricate features, while an excessive number of blocks could lead to overfitting or computational inefficiency. Therefore, our results underscore the importance of carefully tuning the architecture parameters to achieve optimal performance in few-shot segmentation tasks.

TABLE V THE IMPACT OF NUMBER OF RESIDUAL BLOCKS IN EACH STAGE OF MULTI SCALE DECODER ON SEGMENTATION PERFORMANCE IN THE  $COCO-20^i\ {\rm Dataset}$ 

# residual		# learnable					
blocks	fold0	fold1	fold2	fold3	mean	FB-IoU	params
1	42.4	48.2	46.0	45.1	45.4	69.4	1.0M
2	43.9	48.9	46.7	45.5	46.2	69.7	1.2M
3	43.7	49.1	46.9	46.2	46.5	70.4	1.5M
4	41.9	47.4	46.4	45.8	45.4	69.5	1.7M

## V. Conclusion

In conclusion, our proposed few-shot segmentation framework, leveraging a combination of components including a shared pretrained backbone, support prototype mechanism, CMGM, STD, and multi-scale decoder, has demonstrated remarkable efficacy in achieving SOTA performance on both  $PASCAL-5^{i}$  and  $COCO-20^{i}$  datasets. Through extensive experimentation and ablation studies, we have highlighted the critical contributions of each component, particularly emphasizing the significant impact of the multi-scale decoder in enhancing segmentation accuracy while maintaining computational efficiency. Looking ahead, further investigation into the dynamic adaptation of prototype representations and the exploration of additional attention mechanisms could offer avenues for improving the adaptability and robustness of our method across diverse datasets and scenarios. Additionally, exploring semi-supervised learning paradigms could enhance the generalization capability of our framework, enabling effective segmentation in scenarios with limited labeled data. These avenues for future work hold promise for advancing

the effectiveness and applicability of few-shot segmentation methods in real-world scenarios.

#### REFERENCES

- A. Fateh, R. T. Birgani, M. Fateh, and V. Abolghasemi, "Advancing multilingual handwritten numeral recognition with attention-driven transfer learning," *IEEE Access*, vol. 12, pp. 41381–41395, 2024.
- [2] Y. Zhang, Z. Shen, and R. Jiao, "Segment anything model for medical image segmentation: Current applications and future directions," *Computers in Biology and Medicine*, p. 108238, 2024.
- [3] A. Saber, P. Parhami, A. Siahkarzadeh, and A. Fateh, "Efficient and accurate pneumonia detection using a novel multi-scale transformer approach," arXiv preprint arXiv:2408.04290, 2024.
- [4] S. Sun, W. Wang, A. Howard, Q. Yu, P. Torr, and L.-C. Chen, "Remax: Relaxing for better training on efficient panoptic segmentation," Advances in Neural Information Processing Systems, vol. 36, 2024.
- [5] I. B. Barcelos, F. d. C. Belém, L. d. M. João, Z. K. d. P. Jr, A. X. Falcão, and S. J. F. Guimarães, "A comprehensive review and new taxonomy on superpixel segmentation," ACM Computing Surveys, 2024.
- [6] X. Gu, Y. Cui, J. Huang, A. Rashwan, X. Yang, X. Zhou, G. Ghiasi, W. Kuo, H. Chen, L.-C. Chen et al., "Dataseg: Taming a universal multidataset multi-task segmentation model," Advances in Neural Information Processing Systems, vol. 36, 2024.
- [7] Z. Marinov, P. F. Jäger, J. Egger, J. Kleesiek, and R. Stiefelhagen, "Deep interactive segmentation of medical images: A systematic review and taxonomy," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024.
- [8] F. Askari, A. Fateh, and M. R. Mohammadi, "Enhancing few-shot image classification through learnable multi-scale embedding and attention mechanisms," arXiv preprint arXiv:2409.07989, 2024.
- [9] H. Chen, Y. Dong, Z. Lu, Y. Yu, and J. Han, "Pixel matching network for cross-domain few-shot segmentation," in *Proceedings of the IEEE/CVF* Winter Conference on Applications of Computer Vision, 2024, pp. 978– 987.
- [10] C. Lang, G. Cheng, B. Tu, and J. Han, "Few-shot segmentation via divide-and-conquer proxies," *International Journal of Computer Vision*, vol. 132, no. 1, pp. 261–283, 2024.
- [11] S.-A. Liu, Y. Zhang, Z. Qiu, H. Xie, Y. Zhang, and T. Yao, "Learning orthogonal prototypes for generalized few-shot semantic segmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 11319–11328.
- [12] H. Ding, H. Zhang, and X. Jiang, "Self-regularized prototypical network for few-shot semantic segmentation," *Pattern Recognition*, vol. 133, p. 109018, 2023.
- [13] Q. Xu, W. Zhao, G. Lin, and C. Long, "Self-calibrated cross attention network for few-shot segmentation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 655–665.
- [14] D. Kang, P. Koniusz, M. Cho, and N. Murray, "Distilling self-supervised vision transformers for weakly-supervised few-shot classification & segmentation," in *Proceedings of the IEEE/CVF Conference on Computer* Vision and Pattern Recognition, 2023, pp. 19627–19638.
- [15] X. Shi, D. Wei, Y. Zhang, D. Lu, M. Ning, J. Chen, K. Ma, and Y. Zheng, "Dense cross-query-and-support attention weighted mask aggregation for few-shot segmentation," in *European Conference on Computer Vision*. Springer, 2022, pp. 151–168.
- [16] J. Su, M. Ahmed, Y. Lu, S. Pan, W. Bo, and Y. Liu, "Roformer: Enhanced transformer with rotary position embedding," *Neurocomputing*, vol. 568, p. 127063, 2024.
- [17] S. Tian, L. Li, W. Li, H. Ran, X. Ning, and P. Tiwari, "A survey on few-shot class-incremental learning," *Neural Networks*, vol. 169, pp. 307–324, 2024.
- [18] G. Rizzoli, D. Shenaj, and P. Zanuttigh, "Source-free domain adaptation for rgb-d semantic segmentation with vision transformers," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2024, pp. 615–624.
- [19] T. Zhou and W. Wang, "Cross-image pixel contrasting for semantic segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024.
- [20] A. Akter, N. Nosheen, S. Ahmed, M. Hossain, M. A. Yousuf, M. A. A. Almoyad, K. F. Hasan, and M. A. Moni, "Robust clinical applicable cnn and u-net based algorithm for mri classification and segmentation for brain tumor," *Expert Systems with Applications*, vol. 238, p. 122347, 2024.
- [21] F. Yu and V. Koltun, "Multi-scale context aggregation by dilated convolutions," arXiv preprint arXiv:1511.07122, 2015.

- [22] K. He, X. Zhang, S. Ren, and J. Sun, "Spatial pyramid pooling in deep convolutional networks for visual recognition," *IEEE transactions on* pattern analysis and machine intelligence, vol. 37, no. 9, pp. 1904– 1916, 2015.
- [23] X. Wang, R. Girshick, A. Gupta, and K. He, "Non-local neural networks," in *Proceedings of the IEEE conference on computer vision and* pattern recognition, 2018, pp. 7794–7803.
- [24] E. Xie, W. Wang, Z. Yu, A. Anandkumar, J. M. Alvarez, and P. Luo, "Segformer: Simple and efficient design for semantic segmentation with transformers," *Advances in neural information processing systems*, vol. 34, pp. 12 077–12 090, 2021.
- [25] R. Strudel, R. Garcia, I. Laptev, and C. Schmid, "Segmenter: Transformer for semantic segmentation," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pp. 7262–7272.
- [26] S. Zheng, J. Lu, H. Zhao, X. Zhu, Z. Luo, Y. Wang, Y. Fu, J. Feng, T. Xiang, P. H. Torr et al., "Rethinking semantic segmentation from a sequence-to-sequence perspective with transformers," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 6881–6890.
- [27] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, "Swin transformer: Hierarchical vision transformer using shifted windows," in *Proceedings of the IEEE/CVF international conference on* computer vision, 2021, pp. 10012–10022.
- [28] H. Bao, L. Dong, S. Piao, and F. Wei, "Beit: Bert pre-training of image transformers," arXiv preprint arXiv:2106.08254, 2021.
- [29] B. Cheng, A. Schwing, and A. Kirillov, "Per-pixel classification is not all you need for semantic segmentation," *Advances in neural information* processing systems, vol. 34, pp. 17864–17875, 2021.
- [30] B. Cheng, I. Misra, A. G. Schwing, A. Kirillov, and R. Girdhar, "Masked-attention mask transformer for universal image segmentation," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 1290–1299.
- [31] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Medical image computing and* computer-assisted intervention–MICCAI 2015: 18th international conference, Munich, Germany, October 5-9, 2015, proceedings, part III 18. Springer, 2015, pp. 234–241.
- [32] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, "Pyramid scene parsing network," in *Proceedings of the IEEE conference on computer vision* and pattern recognition, 2017, pp. 2881–2890.
- [33] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs," *IEEE transactions on* pattern analysis and machine intelligence, vol. 40, no. 4, pp. 834–848, 2017.
- [34] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, "Encoder-decoder with atrous separable convolution for semantic image segmentation," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 801–818.
- [35] Y. Xu and P. Ghamisi, "Consistency-regularized region-growing network for semantic segmentation of urban scenes with point-level annotations," *IEEE Transactions on Image Processing*, vol. 31, pp. 5038–5051, 2022.
- [36] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo et al., "Segment anything," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 4015–4026.
- [37] G. Li, V. Jampani, L. Sevilla-Lara, D. Sun, J. Kim, and J. Kim, "Adaptive prototype learning and allocation for few-shot segmentation," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 8334–8343.
- [38] C. Zhang, G. Lin, F. Liu, J. Guo, Q. Wu, and R. Yao, "Pyramid graph networks with connection attentions for region-based one-shot semantic segmentation," in *Proceedings of the IEEE/CVF International* Conference on Computer Vision, 2019, pp. 9587–9595.
- [39] K. Wang, J. H. Liew, Y. Zou, D. Zhou, and J. Feng, "Panet: Few-shot image semantic segmentation with prototype alignment," in proceedings of the IEEE/CVF international conference on computer vision, 2019, pp. 9197–9206.
- [40] M. Boudiaf, H. Kervadec, Z. I. Masud, P. Piantanida, I. Ben Ayed, and J. Dolz, "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*, 2021, pp. 13 979–13 988.
- [41] Z. Lu, S. He, X. Zhu, L. Zhang, Y.-Z. Song, and T. Xiang, "Simpler is better: Few-shot semantic segmentation with classifier weight transformer," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 8741–8750.

- [42] A. Kayabaşı, G. Tüfekci, and İ. Ulusoy, "Elimination of non-novel segments at multi-scale for few-shot segmentation," in *Proceedings of* the IEEE/CVF Winter Conference on Applications of Computer Vision, 2023, pp. 2559–2567.
- [43] J. Min, D. Kang, and M. Cho, "Hypercorrelation squeeze for few-shot segmentation," in *Proceedings of the IEEE/CVF international conference* on computer vision, 2021, pp. 6941–6952.
- [44] G. Zhang, G. Kang, Y. Yang, and Y. Wei, "Few-shot segmentation via cycle-consistent transformer," *Advances in Neural Information Process*ing Systems, vol. 34, pp. 21 984–21 996, 2021.
- [45] S. Hong, S. Cho, J. Nam, S. Lin, and S. Kim, "Cost aggregation with 4d convolutional swin transformer for few-shot segmentation," in *European Conference on Computer Vision*. Springer, 2022, pp. 108–126.
- [46] L. Cao, Y. Guo, Y. Yuan, and Q. Jin, "Prototype as query for few shot semantic segmentation," arXiv preprint arXiv:2211.14764, 2022.
- [47] Z. Tian, H. Zhao, M. Shu, Z. Yang, R. Li, and J. Jia, "Prior guided feature enrichment network for few-shot segmentation," *IEEE transactions on* pattern analysis and machine intelligence, vol. 44, no. 2, pp. 1050–1065, 2020.
- [48] Y. Liu, X. Zhang, S. Zhang, and X. He, "Part-aware prototype network for few-shot semantic segmentation," in Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part IX 16. Springer, 2020, pp. 142–158.
- [49] Y. Liu, N. Liu, Q. Cao, X. Yao, J. Han, and L. Shao, "Learning non-target knowledge for few-shot semantic segmentation," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 11573–11582.
- [50] Y. Wang, R. Sun, and T. Zhang, "Rethinking the correlation in fewshot segmentation: A buoys view," in *Proceedings of the IEEE/CVF* Conference on Computer Vision and Pattern Recognition, 2023, pp. 7183–7192.
- [51] Y. Tang and Y. Yu, "Query-guided prototype learning with decoder alignment and dynamic fusion in few-shot segmentation," ACM Transactions on Multimedia Computing, Communications and Applications, vol. 19, no. 2s, pp. 1–20, 2023.
- [52] Z. Chang, X. Gao, N. Li, H. Zhou, and Y. Lu, "Drnet: Disentanglement and recombination network for few-shot semantic segmentation," *IEEE Transactions on Circuits and Systems for Video Technology*, 2024.
- [53] K. Nguyen and S. Todorovic, "Feature weighting and boosting for few-shot segmentation," in *Proceedings of the IEEE/CVF International* Conference on Computer Vision, 2019, pp. 622–631.
- [54] C. Lang, G. Cheng, B. Tu, and J. Han, "Learning what not to segment: A new perspective on few-shot segmentation," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 8057–8067.
- [55] R. Cong, H. Xiong, J. Chen, W. Zhang, Q. Huang, and Y. Zhao, "Query-guided prototype evolution network for few-shot segmentation," *IEEE Transactions on Multimedia*, 2024.
- [56] X. Luo, Z. Tian, T. Zhang, B. Yu, Y. Y. Tang, and J. Jia, "Pfenet++: Boosting few-shot semantic segmentation with the noise-filtered contextaware prior mask," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2023.
- [57] A. Shaban, S. Bansal, Z. Liu, I. Essa, and B. Boots, "One-shot learning for semantic segmentation," arXiv preprint arXiv:1709.03410, 2017.
- [58] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes (voc) challenge," *International journal of computer vision*, vol. 88, pp. 303–338, 2010.
- [59] B. Hariharan, P. Arbeláez, L. Bourdev, S. Maji, and J. Malik, "Semantic contours from inverse detectors," in 2011 international conference on computer vision. IEEE, 2011, pp. 991–998.
- [60] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13. Springer, 2014, pp. 740–755.
- [61] W. Liu, C. Zhang, H. Ding, T.-Y. Hung, and G. Lin, "Few-shot segmentation with optimal transport matching and message flow," *IEEE Transactions on Multimedia*, vol. 25, pp. 5130–5141, 2022.
- [62] W. Wang, L. Duan, Y. Wang, Q. En, J. Fan, and Z. Zhang, "Remember the difference: Cross-domain few-shot semantic segmentation via metamemory transfer," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 7065–7074.