Highlights

A Learnable Support Selection Scheme for Boosting Few-Shot Segmentation

Wenxuan Shao, Hao Qi, Xinghui Dong

- Proposing to use a similarity-based support selection scheme to boost the few-shot segmentation task and demonstrating the effectiveness of this scheme by experimentation.

- Introducing a Siamese Support Selection Network (SSSN) which can be inserted to a few-shot segmentation network. Both networks can be end-to-end trained together.

- Jointly exploiting the local and global features extracted using a Convolutional Neural Network (CNN) and a Transformer network, respectively, on top of a new differentiable feature fusion method, to further improve the segmentation performance.

A Learnable Support Selection Scheme for Boosting Few-Shot Segmentation

Wenxuan Shao[a], Hao Qi[a], Xinghui Dong[a,*]

[a]School of Computer Science and Technology, Ocean University of China, 238 Songling Road, Qingdao, 266100, Shandong, China

Abstract

Upon reevaluating recent studies of Few-Shot Segmentation (FSS), a key observation is that the random selection of support images is not always the optimal. In this situation, the support images cannot provide the useful guidance for the segmentation task. Therefore, we argue that a similarity-based support selection scheme, which selects support images according to the similarity between the query and candidate support images, is able to boost the performance of an FSS network. To this end, we propose a Siamese Support Selection Network (SSSN) which can be end-to-end trained along with an FSS network. We also leverage the joint utilization of a Convolutional Neural Network (CNN) and a Transformer network on top of a new feature fusion method to further improve the performance. To our knowledge, none of the similarity-based support selection scheme and the dual-stream network have been utilized for the FSS task before. Experimental results show that our FSS approach outperforms its counterparts on three data sets. In particular, the SSSN is able to boost the performance of an FSS network. We believe that these promising results should be due to the ability of the SSSN to select the top similar support images, which are useful for the FSS task.

Keywords: Image segmentation, Few-shot segmentation, Support selection, Few-shot learning, Meta learning

1. Introduction

In recent years, great progress has been made in the field of semantic segmentation based on deep learning techniques [1, 2, 3, 4, 5]. Normally, semantic segmentation aims to classify objects at the pixel level. However, deep semantic segmentation methods usually encounter the challenge of insufficient annotated data. This dilemma should be attributed to the fact that the acquisition of these data is expensive and time-consuming. To address that challenge, Few-Shot Segmentation (FSS) [6, 7, 8, 9], which normally segments a query image based on a small number of support images and the corresponding masks, has drawn more and more attention because this task depends on the annotated data less than ordinary semantic segmentation methods.

The FSS task can be fulfilled on top of semantic correspondence [10], which seeks to identify the pixel-level correspondence between the query image and a set of semantically relevant support images. These images may manifest the significant intra-class appearance and geometry variations. According to the classical Gestalt law [11], however, the pixels belong to the objects within the same category are more similar than those of different categories of objects [12]. This explains why humans are able to recognize unseen objects shown in different instances of the known category promptly. In this context, the key to FSS is to make effective use of support images, which provide the template appearance and geometry information of the objects within the same category. Therefore, the performance of an FSS network severely relies on the similarity between the query and support images, which measures the semantic correspondence.

Existing FSS methods normally select support images randomly without considering the similarity between these images and the query image (see Fig. 1(a)). However, the random selection scheme cannot guarantee that the top similar support images are selected. To address this issue, we argue that a similarity-based support selection scheme can boost the performance of an FSS network. This scheme picks out top similar support images based on the similarity between the query and candidate support images. To this purpose, we introduce a novel Siamese Support Selection Network (SSSN). The SSSN can be inserted into an FSS network (see Fig. 1(b)). Both networks can be end-to-end trained together.

Previous studies [13], which used dense intermediate features and high-dimensional convolutions to process correlation tensors, have produced promising results. However, they usually did not utilize a multi-scale feature representation. As a result, repetitive patterns or background clutter may cause ambiguities in the correlation data [10, 14]. To address this issue, Min et al. [15] computed the multi-level feature correlation using 4D convolutions. Nevertheless, limited receptive fields impaired the learning of long-range dependencies. Recently, Vision Transformer (ViT) [16, 17] techniques have been widely applied to different vision tasks because they are able to capture long-range dependencies. However, these techniques cannot learn the local inductive bias. This issue limits their ability to distinguish between the background and the foreground. Therefore, we introduce a multi-scale dual-stream FSS network which comprises a CNN stream and a Transformer stream. In

*Corresponding author

Email addresses: shaowenxuan@stu.ouc.edu.cn (Wenxuan Shao), qihao@stu.ouc.edu.cn (Hao Qi), xinghui.dong@ouc.edu.cn (Xinghui Dong)

[1]Code is available at: https://github.com/INDTLab/SSSN_FSS/.

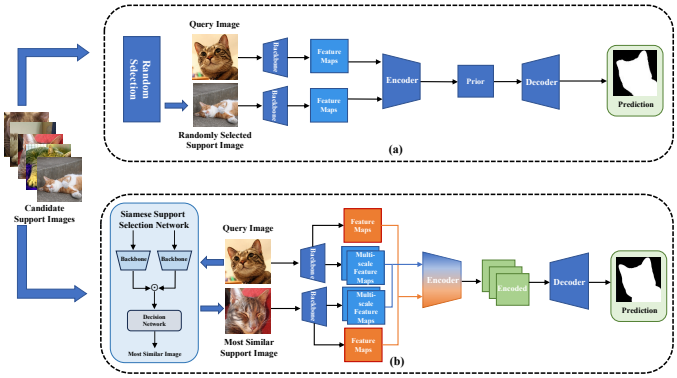

Figure 1: Comparison of the existing FSS method and the proposed approach in the case of the 1-shot segmentation. (a) The usual practice for the FSS task where a support image is randomly selected. (b) The proposed FSS approach which is built on top of a Siamese Support Selection Network (SSSN) and a Dual-stream FSS Network. Here, $\odot$ denotes the Hadamard product operation.

particular, we develop a simple differentiable fusion method in order to jointly exploit the features extracted using the two streams.

To our knowledge, none of the similarity-based support selection scheme and the dual-stream network have been used for the FSS task before. The contributions of this study can be summarized into fourfold. (1) We propose to use a similarity-based support selection scheme to boost the FSS task and show the effectiveness of this scheme by experimentation. (2) We introduce a Siamese Support Selection Network (SSSN) which can be end-to-end trained along with an FSS network. The SSSN is able to select top similar support images to the query image. (3) We adopt a dual-stream FSS network on top of a new differentiable feature fusion approach, which jointly exploits both the global and local features extracted using the CNN and Transformer streams respectively. (4) The proposed approach is tested on three data sets. The results can be used as baselines by the community.

The organization of the rest of this paper is described as follows. In Section 2, we review the related work. The proposed approach is introduced in Section 3. The experimental settings and results are reported in Sections 4 and 5 respectively. Finally, we draw our conclusion in Section 6.

2. Related work

2.1. Few-shot segmentation

Few-shot segmentation (FSS) is a challenging task which aims at segmenting a novel query image based on a small set of annotated support images [18]. In essence, this task is a dense prediction application [19]. Early FSS networks normally used a two-branch architecture to extract class prototypes from support images [7, 20]. The prototypes extracted from a query image were then segmented in order to produce the segmentation mask. However, there was an unavoidable loss when all the information contained in the support images was condensed into only one or a few prototypes [21]. To address this problem, the research emphasis was put on making effective use of the category information of support images, in order that the better guidance could be obtained for segmentation of query images [22]. Different approaches [7, 23] were developed in order to extract the richer information from support images by building multiple prototypes for each category in terms of different areas of the query image.

To utilize the information manifested in the support images more adequately, the pixel-level correspondence between the query and support images has been studied in recent years. Min et al. [15] used the 4D correlation tensors computed using 4D convolutions to encode this type of correspondence data. In [24], it was demonstrated that more support images were useful for learning the representation of the single-modal target class. Vision Transformers (ViTs) have been used to improve the class information transfer between the query and support images. To identify the hidden patterns encoded in the affinity maps, Hong et al. [10] employed both the 4D convolutions and Transformers.

However, none of the above-mentioned studies determined support images by taking account into the similarity between these images and the query image. In contrast, we argue the importance of the similarity-based support selection scheme to the FSS task. To this end, we design a novel Siamese Support Selection Network (SSSN), which can be inserted into an FSS network. Both networks can be end-to-end trained together.

2.2. Vision transformers

Vision Transformers (ViTs) [16] have achieved many successes in different vision tasks. Recent studies [25, 26, 27, 10] investigated the application of Transformers to the FSS task. Zhang et al. [27] computed query and key sequences from the query and support images based on the cycle-consistent attention mechanism, to reduce the potential damage to the features extracted from the support images. In [26], each query pixel was processed as a single token. The similarity to each support pixel was then calculated.

Efficient Transformers [28, 29] were normally used to reduce the computational cost caused by the quadratic complexity of long sequences through a simplified self-attention process. To aggregate the features computed from the query and support images, Hong et al. [10] built a cost aggregation network, referred to as the 4D Convolutional SwinTransformer [17]. A high-dimensional patch embedding was constructed on top of a series of small-kernel convolutions, which captured the local context information.

Considering the advantages of CNNs and Transformers, which are able to extract local characteristics and long-range dependencies, respectively, they have been brought together in the same network. Motivated by Conformer [30] and the following studies [31], we build a dual-stream FSS network on top of the pre-trained Conformer model. This network exploits both the local and global features, extracted using a CNN stream and a Transformer stream, respectively, based on a simple feature fusion method.

3. Methodology

Considering that the support images randomly selected are often not the optimal, we argue that a similarity-based support selection scheme is useful for boosting the performance of an FSS network. We therefore propose a Siamese Support Selection Network (SSSN). This network can be inserted into an FSS network. Both the SSSN and the FSS network can be end-to-end trained together. Given a query image, the SSSN is able to select a set of top similar support images according to the similarity computed between this image and candidate support images. To further improve the segmentation performance, we also introduce a dual-stream FSS network which jointly exploits the features extracted using a CNN stream and a Transformer stream. The FSS network contains a feature extraction module, a correlation computation module, a dual-stream encoder and a 2D SwinTransformer decoder. Both the SSSN and the dual-stream FSS network are comprised of the proposed FSS approach. The architecture of this approach is shown in Fig. 2. To our knowledge, either the similarity-based support selection scheme or the dual-stream FSS network has not been applied to the FSS task before.

3.1. Problem formulation

The FSS task has to conquer the challenge that semantic segmentation of novel classes encounters under the constraint of the limited annotated data. This task aims to segment objects, which belong to unseen classes, based on a small set of annotated support images. To comply with the common FSS setup, the training and testing data sets are denoted as D_{Train} and D_{Test} respectively. Both the sets do not share the same class. Normally, the episodic training scheme is employed. Each episode set corresponds to a particular class. This set contains a query sample (I_q, M_q) and K support samples $S = \{(I_s^k, M_s^k)\}_{k=1}^K$, where I_* and M_* stand for an image and the corresponding mask respectively. During the training stage, both I_q and I_s^k are randomly sampled from D_{Train}. An FSS network learns from I_s^k and uses I_q to make a prediction of M_q. On the other hand, I_s^k and I_q are randomly selected from D_{Test} during the testing stage, in order to predict $\hat{M}_q$.

3.2. Motivation

The performance of an FSS network heavily relies on the support images. Our observation is that the random support selection scheme cannot guarantee that the top similar support images to the query image are returned. As a result, the performance of the network will be impaired. Due to this observation, we argue that a similarity-based support selection scheme is useful for boosting the performance of an FSS network. On top of the similarity calculated between the query and candidate support images, this scheme selects a set of top similar support images.

Siamese networks have been widely used in similarity measurement tasks. These networks normally consist of two identical subnetworks which share the same architecture and weights. This design allows the subnetworks to learn and extract the same features. As a result, the network are able to capture the meaningful relationship between two input samples. Therefore, we are inspired to introduce a siamese network for support image selection, which is referred to as Siamese Support Selection Network (SSSN).

Apart from local characteristics, it has been demonstrated that long-range dependencies/interactions are important to image similarity [32]. As known, CNNs are able to capture characteristics at multiple levels through a series of convolutional operations. On the other hand, Transformers use the self-attention mechanism to aggregate patch embeddings. In this case, the long-range dependencies can be captured. Motivated by the success that Conformer [30] achieved in the image classification task, we adopt a dual-stream FSS network, to exploit both the local characteristics and long-range dependencies extracted using the CNN and Transform streams respectively.

3.3. Siamese support selection network

The Siamese Support Selection Network (SSSN) uses the same training/testing split as that utilized for the FSS network. To enhance the ability of the SSSN to select similar images, we employ an episodic training scheme. Regarding each episode, an image x_a is randomly selected from D_{Train}. The images which have been used as x_a cannot be selected as x_a in the following episodes. The rest of the images in D_{Train} are then divided into two distinct subsets. Given the class that x_a belongs to, one subset consists of the images from the same class while the other subset contains the images from different classes. The other image x_b is randomly selected from the two subsets. If x_a and x_b are contained in the same class, we have $l = 1$; otherwise, we have $l = 0$, where l indicates whether or not both the images belong to the same class. Thanks to this selection strategy, the SSSN is able to learn not only the intra-class similarity but also the inter-class similarity.

During a training episode, two images x_a and x_b are first fed into the two subnetworks of the SSSN, respectively. Two sets of features, denoted as F_{x_a} and F_{x_b}, are then extracted using the two subnetworks, respectively, which are fused using the Hadamard product. The fused features are further sent to a Multi-layer Perception (MLP) unit, which contains a series of linear and sigmoid layers. The result is a similarity score S, which measures the similarity between x_a and x_b. The computation can be expressed as:

$$S = Sigmoid(\mathrm{MLP}(F_{x_a} \odot F_{x_b})), \tag{1}$$

where $Sigmoid(\cdot)$ represents the sigmoid function, $\odot$ denotes the Hadamard product operation and MLP stands for the MLP classification head.

Once the SSSN has been trained, it can be used to select top K similar support images. To reduce the computational cost of support image selection, a set of candidate support images $\{I_s^i\}$ $(i \in \{1...n\}, n \geqslant K)$ are randomly selected from the same class as that of I_q. The SSSN predicts the probability that I_q and I_s^i belong to the same class. This value is used as the similarity between the two images. Finally, K images with regard to the top K similarity scores, respectively, are selected.

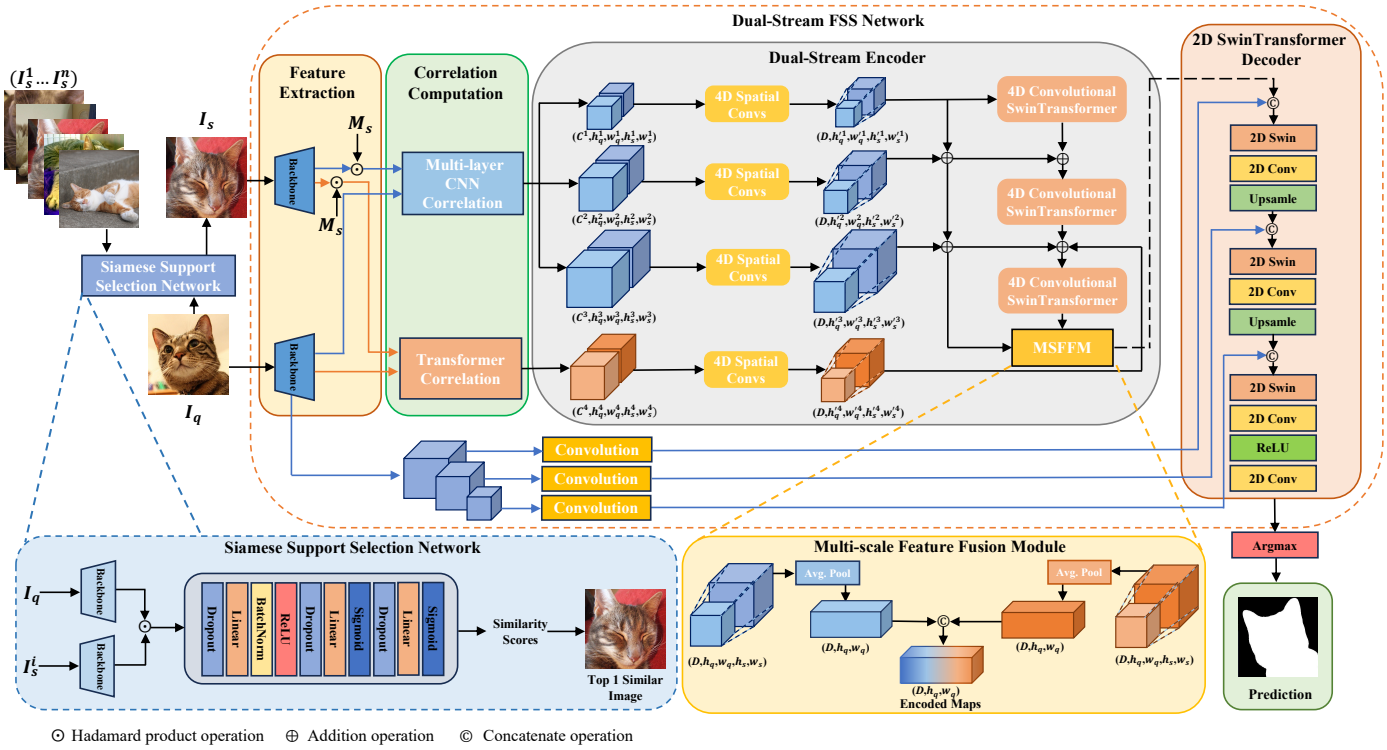

Figure 2: The architecture of the proposed FSS approach, which contains a Siamese Support Selection Network (SSSN) and a dual-stream FSS network, in the 1-shot segmentation scenario. The FSS network comprises a feature extraction module, a correlation computation module, a dual-stream encoder and a 2D SwinTransformer [17] decoder.

3.4. Feature extraction

Regarding the feature extraction module, we use the pre-trained Conformer [30] model as the backbone. In total, L layers of the CNN stream are utilized. Given a query image I_q and the support image I_s selected using the SSSN, two sets of feature maps are extracted using the l-th layer of the CNN stream, denoted as $CNN\_F_q^l$ and $CNN\_F_s^l$ ($l \in \{1...L\}$), respectively. Meanwhile, two sets of feature maps are extracted from the two images using the Transformer stream, denoted as $Trans\_F_q$ and $Trans\_F_s$, respectively.

To eliminate the interference of the background area and exploit the context information, we mask each feature map extracted from the support image using the associated mask by following the existing studies [10, 15]. The feature map masking process can be expressed as:

$$\hat{CNN\_F_s^l} = CNN\_F_s^l \odot \varphi(M_s), \tag{2}$$

$$\hat{Trans\_F_s} = Trans\_F_s \odot \varphi(M_s), \tag{3}$$

where $\odot$ is the Hadamard product operation and $\varphi(\cdot)$ is a function which resizes the support mask M_s and expands it along the channel dimension in order to match the shape of the associated support feature maps. In this study, the $CNN\_F_s^l$ maps had the shapes of $(256 \times 128 \times 128)$, $(512 \times 64 \times 64)$, $(1024 \times 32 \times 32)$ and $(1024 \times 16 \times 16)$ in turn, while the $Trans\_F_s$ maps had the shape of $(384 \times 32 \times 32)$.

3.5. Correlation computation

The pixel-level correspondence between a query image and the support image selected can be computed as the 4D correlation [15, 10]. Given two sets of CNN feature maps, i.e., $CNN\_F_q^l$ and $CNN\_F_s^l$, the 4D correlation is calculated using the cosine similarity:

$$C_{CNN}^l(p^q, p^s) = ReLU\left(\frac{CNN\_F_q^l(p^q) \cdot CNN\_F_s^l(p^s)}{\|CNN\_F_q^l(p^q)\| \cdot \|CNN\_F_s^l(p^s)\|}\right), \tag{4}$$

where p^q and p^s denote the 2D spatial positions in the two sets of feature maps respectively. Similarly, the 4D correlation can be computed for two sets of Transformer feature maps $Trans\_F_q$ and $Trans\_F_s$ using the equation below:

$$C_{Trans}(p^q, p^s) = ReLU\left(\frac{Trans\_F_q(p^q) \cdot Trans\_F_s(p^s)}{\|Trans\_F_q(p^q)\| \cdot \|Trans\_F_s(p^s)\|}\right). \tag{5}$$

Following the previous studies [15, 10], we collect the 4D correlations computed from all the intermediate features of the same spatial size. These correlations are then stacked in order to obtain a set of stacked 4D correlation maps $C^l \in \mathbb{R}^{C \times h_q^l \times w_q^l \times h_s^l \times w_s^l}$, where (h_q^l, w_q^l) and (h_s^l, w_s^l) are the height and width of the feature maps extracted from the query and support images, respectively, and $l \in \{1...L\}$ is the layer index for the CNN feature maps and $l = L + 1$ is the index for the Transformer feature maps.

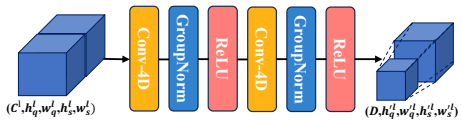

Figure 3: The 4D spatial convolutional block, which contains two consecutive sequences of operations.

3.6. Dual-stream encoder

The L+1 sets of 4D correlation maps produced by the correlation computation module are passed through a two-stream encoder. The encoder compresses these maps into a set of 2D maps. A CNN stream and a Transformer stream are comprised of the encoder. Both the streams process the two types of 4D correlation maps, computed in the multi-layer CNN correlation and Transformer correlation units respectively.

The CNN stream comprises L (L=3) branches in terms of different layers in the CNN stream of the backbone. In each branch, a 4D spatial convolutional block is used to reduce the spatial size of the 4D correlation maps $C^l \in \mathbb{R}^{C\times h_q^l\times w_q^l\times h_s^l\times w_s^l}$. The result is denoted as $C^{l'} \in \mathbb{R}^{D\times h_q'^l\times w_q'^l\times h_s'^l\times w_s'^l}$. Regarding the first branch, the 4D correlation maps $C^{1'}$ are sent to a 4D convolutional SwinTransformer [17] block. For each of the other branches, the related 4D correlation maps $C^{l'}$ are added with the $C^{(l-1)'}$ produced in the upper branch after the upsampling operation has been performed. The resultant maps are further added with the output of the upper branch while the sum is fed into the 4D SwinTransformer block in the current branch.

On the other hand, the Transformer stream receives the 4D correlation maps computed in the Transformer correlation unit. These maps are fed into a 4D spatial convolutional block. The result is added with the output of the second CNN branch and the output of the 4D spatial convolutional block in the third CNN branch. Finally, the sum is sent to the 4D SwinTransformer block in the third CNN branch.

Both the output of the 4D convolutional block in the third CNN branch and the output of this branch, denoted as $C^{L'}$ and $C^{L''}$, are sent to a Multi-scale Feature Fusion Module (MSFFM). The result is a set of 2D encoded maps, which will be processed by a 2D SwinTransformer decoder.

3.6.1. 4D spatial convolutional block

To address the challenge of resource requirements due to the 4D convolutions, a 4D spatial convolutional block (see Fig. 3) was introduced [15]. The block normally comprises two or three sequences of operations. Each sequence includes a multi-channel 4D convolution, a group normalization and an ReLU activation. Within each block, we periodically use large strides to compress the last two spatial dimensions of $C^l \in \mathbb{R}^{C\times h_q^l\times w_q^l\times h_s^l\times w_s^l}$ to 8, while we reduce the first two spatial dimensions to one half of each. As a result, a set of compressed maps are generated, denoted as $C^{l'} \in \mathbb{R}^{D\times h_q'^l\times w_q'^l\times h_s'^l\times w_s'^l}$, where $h_q^l > h_q'^l$, $w_q^l > w_q'^l$, $h_s^l > h_s'^l$ and $w_s^l > w_s'^l$.

3.6.2. 4D convolutional SwinTransformer

The 4D convolutional SwinTransformer comprised a Volumetric Convolution Module (VCM) and a Volumetric Transformer Module (VTM) [10]. As an extension of the SwinTransformer [17], the VTM was used to process 4D correlation maps [10]. A modified self-attention mechanism was adopted. The correlation map was partitioned into non-overlapping submaps. The self-attention mechanism was performed in each submap. This strategy enabled localized interactions and was able to capture the spatial relationship. Shifting windows were also used to capture different spatial relationships. In addition, the relative position bias was employed in order to enhance the self-attention mechanism by deriving values from a parameterized bias matrix. In this context, the attention could be paid to different positions on top of the spatial structure of the correlation maps. In contrast, the VCM contained a series of operations, including the overlapping convolution, 4D spatial max-pooling, ReLU activation and Group Normalization, for the purpose of decreasing the spatial dimensions of the correlation maps.

3.6.3. Multi-scale feature fusion module

As mentioned in [10], the features extracted using CNNs and Transformers cannot be effectively fused using a residual connection-like approach. To jointly exploit the features processed by the 4D spatial convolutional block and the 4D convolutional SwinTransformer, we adopt a simple Multi-scale Feature Fusion Module (MSFFM). Given that the output of the 4D spatial convolutional block in the last CNN branch and the output of this branch are denoted as $C^{L'}$ and $C^{L''} \in \mathbb{R}^{D\times h_q'^L\times w_q'^L\times h_s'^L\times w_s'^L}$, they are fed into the MSFFM. $C^{L'}$ and $C^{L''}$ are first compressed using an average pooling operation into $D\times h_q'^L\times w_q'^L$ maps. The two sets of compressed maps are then concatenated into a single set of maps. A 1×1 convolution is further applied to these maps. Finally, a set of $D\times h_q'^L\times w_q'^L$ encoded maps are produced, which capture the correspondence between the query and support images.

3.7. 2D SwinTransformer decoder

To predict the segmentation mask, the encoded maps produced by the dual-stream encoder are fed into a decoder [10]. The decoder contains three consecutive blocks. The appearance embedding is applied at the beginning of each block. That is to say, the input feature maps of each block are concatenated with the feature maps extracted using a single layer of the CNN stream of the backbone. Thanks to the appearance embedding, the decoder is able to focus on the relevant features and suppress the irrelevant information. Thus, the accuracy of segmentation is likely to be improved. The embeddings are sent to a 2D SwinTransformer [17], which learns the long-range dependencies between different regions. Then a 2D convolution is applied in order to capture local characteristics. Within the third block, the features extracted using the first convolution are passed through an Relu layer and a second convolution. As a result, the encoded maps are converted into a two-channel segmentation map. An *argmax* function is further applied to the map. The result is the final segmentation mask.

3.8. Extension to the K-shot scenario

Given a query image I_q and K support images $S = \{(I_s^k, M_s^k)\}_{k=1}^K$, we perform K forward passes in order to obtain K different query masks $\hat{M}_q$, respectively. All the K masks are summed up at each pixel location and the resultant map is divided by K. If the value of a point in the map exceeds the threshold τ, we label this point as the foreground; otherwise, we label it as the background.

4. Experimental settings

In this section, we present the data sets, evaluation metrics and implementation details used in our experiments.

4.1. Data sets

We evaluated the proposed approach using three data sets, including PASCAL-5^i [6], COCO-20^i [33] and FSS-1000 [8]. The PASCAL-5^i data set was created on top of the PASCAL VOC 2012 [34]. This data set contained 20 object classes, which were evenly divided into four folds: 5^i ($i \in \{0, 1, 2, 3\}$), along with the corresponding annotated masks. The COCO-20^i [33] data set comprised 80 object classes, which were also divided into four folds. Each fold consisted of 20 classes. An annotated mask was provided with each image. Regarding the PASCAL-5^i [6] and COCO-20^i [33] data sets, we performed a cross-validation experiment across all the folds using a training/testing scheme. In terms of each fold i, the images contained in the other folds were utilized for training, while only 1,000 episodes within this fold were randomly selected for testing, by following the setup used in [15, 10]. Besides, the FSS-1000 [8] data set comprised 1,000 different object classes. Following the setup utilized in [15, 10], we divided these classes into three subsets, which contained 520, 240 and 240 classes, respectively, for training, validation and testing in turn.

4.2. Evaluation metrics

Regarding evaluation metrics, we followed the common practice [15, 10, 27]. Specifically, both the mean Intersection over Union (mIoU) and the Foreground-Background Intersection over Union (FB-IoU) measures were utilized. The mIoU metric is defined as the average IoU value calculated across all the classes in a fold. The computation can be expressed as:

$$mIoU = \frac{1}{C} \sum_{c=1}^{C} IoU_c, \tag{6}$$

where C is the number of classes contained in the current fold and IoU_c is the IoU value computed in terms of class c. Given that IoU_F and IoU_B denote the foreground and background IoU values, computed in the current fold, respectively, the FB-IoU metric can be calculated as:

$$FB\text{-}IoU = \frac{1}{2} \left(IoU_F + IoU_B \right). \tag{7}$$

This metric ignores object classes and only considers the average of both the foreground and background IoU values.

4.3. Implementation details

The pre-trained Conformer-S [30] model was employed as the backbone network by default. To ensure the consistency for different data sets and avoid overfitting, we froze the weights of the pre-trained model during the training process. Data augmentation was not used for the training operation. We adjusted the resolutions of the training images in the PASCAL-5^i [6], FSS-1000 [8] and COCO-20^i [33] data sets to 480×480 pixels, 480×480 pixels and 400×400 pixels, respectively. The window size was set to 4 for SwinTransformer [17]. We used the AdamW optimizer. The learning rate was set to 5e-4. The K-shot threshold τ was set to 0.5 and the embedding dimension D was set to 128. The number of candidate support images was set to 9. The binary cross-entropy loss and cross-entropy loss functions were used for the SSSN and the dual-stream FSS network, respectively. In each epoch, the two losses were updated separately.

5. Experimental results

We evaluated the proposed FSS approach using three data sets. Our approach was compared with state-of-the-art FSS methods. To examine the effect of different components of our approach, we further conducted a series of ablation studies.

5.1. Few-shot segmentation results

The proposed approach was tested using three data sets, including PASCAL5^i [6], COCO-20^i [33] and FSS-1000 [8].

PASCAL-5^i We compared our approach with eight baselines on the PASCAL5^i [6] data set. The results obtained for the 1-shot and 5-shot segmentation tasks are shown in Tab. 1. Although the backbone that we used contained less parameters, our approach still outperformed its counterparts, which used the heavier backbone, with large margins.

In particular, the mIoU value, produced by our method for the 1-shot segmentation task, was greater than those produced by HSNet [15] and VAT [10] with the margins of 8.2 and 6.5, respectively. It is suggested that the proposed approach was superior to these state-of-the-art methods.

Fig. 4 presents six sets of results derived using HSNet [15], VAT [10] and our approach. It can be observed that our approach produced the more accurate segmentation results than the baselines. Also, we compared those methods in terms of the convergence speed when PASCAL5^0 [6] was used for testing. As shown in Fig. 5, our method converged faster than both HSNet [15] and VAT [10] but the mIoU value that we derived was much larger than those produced by the baselines at the beginning of the training stage.

We used an NVIDIA RTX 3090, achieving an average inference time of 0.37 seconds per image in the 1-shot setting and 0.98 seconds per image in the 5-shot setting. The model without SSSN achieving an average inference time of 0.21 seconds per image in the 1-shot setting and 0.82 seconds per image in the 5-shot setting.

COCO-20^i As can be seen in Tab. 2, our approach normally outperformed seven baselines on the COCO-20^i data set [33]

Backbone	Method	1-Shot						5-Shot					
		mIoU					FB-IoU	mIoU					FB-IoU
		5^0	5^1	5^2	5^3	Mean		5^0	5^1	5^2	5^3	Mean	
ResNet50	PANet [20]	44.0	57.5	50.8	44.0	49.1	-	55.3	67.2	61.3	53.2	59.3	-
ResNet50	PFENet [35]	61.7	69.5	55.4	56.3	60.8	73.3	63.1	70.7	55.8	57.9	61.9	73.9
ResNet101	FWB [18]	51.3	64.5	56.7	52.2	56.2	-	54.8	67.4	62.2	55.3	59.9	-
ResNet101	DAN [19]	54.7	68.6	57.8	51.6	58.2	71.9	57.9	69.0	60.1	54.9	60.5	72.3
ResNet101	HSNet [15]	67.3	72.3	62.0	63.1	66.2	77.6	71.8	74.4	67.0	68.3	70.4	80.6
ResNet101	VAT [10]	70.0	72.5	64.8	64.2	67.9	79.6	75.0	75.2	68.4	69.5	72.0	83.2
ResNet101	CyCTR [27]	67.2	71.1	57.6	59.0	63.7	73.0	71.0	75.0	58.5	65.0	67.4	75.4
ViT-B	FPTrans [25]	67.1	69.8	65.6	56.4	64.7	-	73.5	75.7	**77.4**	68.3	73.7	-
Conformer-S	Ours	**76.6**	**77.4**	**70.1**	**73.4**	**74.4**	**83.5**	**78.2**	**79.7**	72.0	**76.9**	**76.7**	**85.0**

Table 1: The comparison of the proposed FSS approach and eight baselines on the PASCAL-5^i [6] data set. The best and second best results are indicated in the **bold** and underline fonts, respectively, while the results which have not been reported in the original publications are indicted using "-".

Figure 4: Qualitative comparison of the results obtained using HSNet [15], VAT [10] and our approach on the PASCAL-5^i [6] data set in the 1-shot FSS scenario. In each column, a query image, the associated ground-truth label image and the results produced by the three FSS methods are shown from top to bottom. Besides, the IoU value computed against the ground-truth mask is displayed below each resultant image.

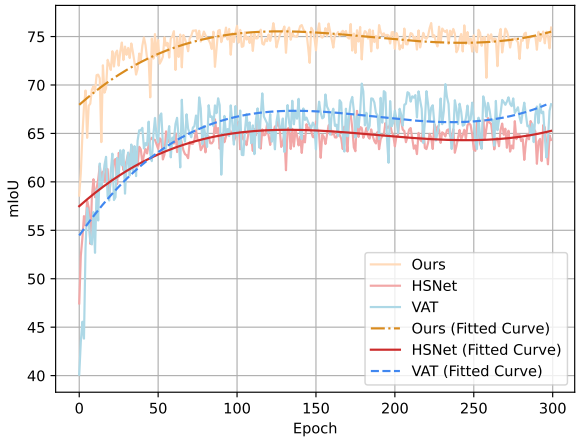

Figure 5: The comparison of HSNet [15], VAT [10] and our approach in terms of the convergence speed, when PASCAL5^0 [6] was used for testing.

with large margins. Compared with HSNet [15] and VAT [10], the mIoU derived using our approach was 12.3 and 12.2 greater than those produced by these methods. It is noteworthy that the Conformer-S [30] model that we used contains fewer parameters, compared with both the ResNet101 [36] and ViT-B [37] that the baselines usually employed.

FSS-1000 As reported in Tab. 3, our approach performed better than five baselines. This finding further demonstrates the effectiveness of the proposed approach for the FSS task.

5.2. Ablation study

To investigate the impact of different components of the proposed approach, a comprehensive ablation study was conducted. For simplicity, only the first round of cross-validation was performed on PASCAL-5^i [6].

5.2.1. Effect of different backbones

We examined the impact of different backbones on the performance of the proposed FSS approach. The pre-trained ResNet50 [36], ResNet101 [36], Conformer-T [30] and Conformer-S [30] were utilized. The CNN streams of the Conformer-T [30] and Conformer-S [30] were also tested individually. The results are shown in Tab. 4. As can be seen, our approach together with the Conformer-S [30] model performed better than it did using the pre-trained ResNet50 [36], ResNet101 [36] and Conformer-T models in both the 1-shot and 5-shot experiments. Given that both the CNN and Transformer streams were employed, the mIoU value obtained was higher than that derived using only the CNN stream. It should be noted that our method produced the worse result using ResNet101, compared with that generated using Conformer-S, even if the former used more parameters than the latter.

5.2.2. Effect of different components of the dual-stream encoder

We removed the MSFFM, the Transformer stream and the SSSN from the proposed FSS method separately. In total, three variants were derived. The results obtained using our method and these variants are reported in Tab. 5. It can be observed that our method was superior to those variants in both the 1-Shot and 5-Shot experiments. Regarding the influence to our method, the MSFFM performed the least while the SSSN operated the most among the three components.

Backbone	Method	1-Shot						5-Shot					
		mIoU					FB-IoU	mIoU					FB-IoU
		20^0	20^1	20^2	20^3	Mean		20^0	20^1	20^2	20^3	Mean	
ResNet50	CMN [38]	37.9	44.8	38.7	35.6	39.3	61.7	42.0	50.5	41.0	38.9	43.1	63.3
ResNet50	VAT [10]	39.0	43.8	42.6	39.7	41.3	68.8	44.1	51.1	50.2	46.1	47.9	72.4
ResNet101	FWB [18]	17.0	18.0	21.0	28.9	21.2	-	19.1	21.5	23.9	30.1	23.7	-
ResNet101	PFENet [35]	36.8	41.8	38.7	36.7	38.5	63.0	40.4	46.8	43.2	40.5	42.7	65.8
ResNet101	HSNet [15]	37.2	44.1	42.4	41.3	41.2	69.1	45.9	53.0	51.8	47.1	49.5	72.4
ViT-B	FPTrans [25]	39.7	44.1	44.4	39.7	42.0	-	49.9	56.5	55.4	**53.2**	53.8	-
Conformer-S	Ours	**52.0**	**58.0**	**54.5**	**49.6**	**53.5**	**75.5**	**58.0**	**60.9**	**58.4**	53.0	**57.6**	**77.4**

Table 2: The comparison of the proposed FSS approach and seven baselines on the COCO-20^i [33] data set.

Backbone	Method	mIoU		FB-IoU	
		1-Shot	5-Shot	1-Shot	5-Shot
ResNet50	FSOT [39]	82.5	83.8	-	-
ResNet101	DAN [19]	85.2	88.1	-	-
ResNet101	HSNet [15]	86.5	88.5	91.6	92.9
ResNet101	VAT [10]	90.3	90.8	94.0	94.4
Swin-B	DCAMA [26]	90.1	90.4	93.8	94.1
Conformer-S	Ours	**90.6**	**91.1**	**94.2**	**94.6**

Table 3: The comparison of our approach and five baselines on the FSS-1000 [8] data set.

Backbone	Number of	mIoU		FB-IoU	
	Parameters	1-Shot	5-Shot	1-Shot	5-Shot
ResNet50	25.6M	69.6	73.3	84.1	86.3
ResNet101	44.5M	70.6	74.5	85.0	87.5
Conformer-T (CNN)	23.5M	71.0	72.5	84.1	85.0
Conformer-T (CNN+Trans.)	23.5M	73.0	75.5	85.9	87.2
Conformer-S (CNN)	37.7M	75.7	77.9	87.4	**88.7**
Conformer-S (CNN+Trans.)	37.7M	**76.6**	**78.2**	**88.2**	88.6

Table 4: Effect of different backbones on the proposed FSS approach.

5.2.3. *Effect of different support selection schemes*

Given the proposed FSS approach, we compared our Siamese Support Selection Network (SSSN) with two support selection schemes, including the random and the SSIM-based selection schemes. Regarding the training and inference stages, different support selection schemes were applied. It is noteworthy that the selection scheme which utilizes the random selection in both the training and inference stages is normally used by the existing FSS methods [15, 10]. The results of the comparison are shown in Tab. 6. It can be observed that the best result was derived when the proposed SSSN was applied to both the training and inference stages. Given that the SSSN was only used in the inference stage, the results were still superior to those obtained using the random and/or SSIM-based selection schemes.

5.2.4. *Effect of the proposed SSSN on different FSS networks*

The SSSN was applied to two state-of-the-art FSS networks, including HSNet [15] and VAT [10]. The results produced by HSNet [15], VAT [10] and our approach with or without the SSSN are reported in Tab. 7. As can be seen, the SSSN normally boosted the performance of the three FSS networks in either the 1-shot or the 5-shot experiment. In particular, our approach still outperformed its counterparts when the SSSN was replaced by the random selection scheme. In this case, our approach was usually superior to both HSNet [15] and VAT [10] which had been boosted using the SSSN. The advantage of the proposed dual-stream FSS network was indicated.

Variant	mIoU		FB-IoU	
	1-Shot	5-Shot	1-Shot	5-Shot
Ours	**76.6**	**78.2**	**88.2**	88.6
without MSFFM	76.2	77.1	87.4	88.0
without Trans. Stream	75.7	77.9	87.4	**88.7**
without SSSN	73.8	76.8	86.6	88.3

Table 5: Effect of different components of the dual-stream encoder on our approach.

Support Selection Scheme		mIoU		FB-IoU	
Training	Inference	1-Shot	5-Shot	1-Shot	5-Shot
Random	Random	74.8	77.5	86.5	88.0
Random	SSIM	74.6	77.3	86.9	88.2
SSIM	SSIM	75.3	77.4	86.7	88.0
Random	SSSN	75.3	77.6	86.9	88.3
SSSN	SSSN	**76.6**	**78.2**	**88.2**	**88.6**

Table 6: Effect of different support selection schemes on our approach.

5.2.5. *Effect of different fusion strategies for the Transformer stream*

We compared three fusion strategies used for the Transformer stream in the dual-stream FSS network. Regarding each strategy, an individual CNN branch was fused with the Transformer stream. It can be seen in Tab. 8 that the best performance was achieved when the Transformer stream was fused with the second CNN branch.

5.2.6. *Effect of different K-shot settings*

Given that 5 or 9 candidate support images were used, the effect of different K-shot settings on the HSNet [15], VAT [10] and our method was investigated. Both the HSNet [15] and VAT [10] used the random support selection scheme while our method utilized the SSSN for support selection. The results are reported in Tab. 9. As can be seen, the performance of the FSS task was improved with the increase of the n or K values. In other words, the better result was produced when the more support images were used. The proposed method achieved the better performance than both the HSNet and VAT in the case that K support images were selected from the same n candidate support images. The superiority of our method to its two

Method	mIoU		FB-IoU	
	1-Shot	5-Shot	1-Shot	5-Shot
HSNet	67.3	71.8	82.4	85.0
HSNet with SSSN	68.2	70.6	82.8	84.5
VAT	70.0	75.0	84.2	86.8
VAT with SSSN	74.1	75.5	87.1	88.0
Ours without SSSN	74.8	77.5	86.5	88.0
Ours	**76.6**	**78.2**	**88.2**	**88.6**

Table 7: Effect of the proposed SSSN on different FSS networks.

Method	mIoU		FB-IoU	
	1-Shot	5-Shot	1-Shot	5-Shot
Transformer + CNN branch 1	75.5	77.7	87.2	88.5
Transformer + CNN branch 2	**76.6**	**78.2**	**88.2**	**88.6**
Transformer + CNN branch 3	75.3	77.5	87.3	88.4

Table 8: Effect of different fusion strategies for the Transformer stream.

Method	$n = 5$		$n = 9$				
	$K = 1$	$K = 5$	$K = 1$	$K = 3$	$K = 5$	$K = 7$	$K = 9$
HSNet	66.8	70.4	67.4	69.7	71.4	71.9	71.9
VAT	69.3	72.3	70.1	73.3	73.5	73.9	74.5
Ours	**75.2**	**77.4**	**76.6**	**77.8**	**78.2**	**78.8**	**78.3**

Table 9: Effect of different K-shot settings on the HSNet [15], VAT [10] and our method in the case that K support images were selected from the same n candidate support images.

counterparts was further demonstrated.

5.2.7. Computational efficiency

Given that an NVIDIA RTX 3090 graphics card was used, the inference operation performed using our method in the 1-shot and 5-shot segmentation experiments took around 0.37 and 0.98 seconds per image, respectively. When the SSSN was removed from our network, the two digits were 0.21 and 0.82, respectively.

6. Conclusion

We observed that the support images randomly selected often did not provide the useful guidance for the Few-Shot Segmentation (FSS) task. We therefore argued that a similarity-based support selection scheme, which selects the support images based on the similarity between the query and candidate support images, is useful for boosting the performance of an FSS network. To this purpose, we introduced a Siamese Support Selection Network (SSSN). This network can be end-to-end trained together with an FSS network. To further improve the performance, we jointly exploited a Convolutional Neural Network (CNN) and a Transformer network using a new feature fusion approach.

Our FSS approach was applied to three different data sets. The results demonstrated that our approach normally outperformed its counterparts with large margins. In particular, the SSSN improved the performance of two state-of-the-art FSS networks, including HSNet and VAT. The promising results should be due to the capability of the SSSN to decide the top similar support images, which are useful for guiding the FSS task.

However, the joint use of the SSSN increases the computational load of the FSS network. Therefore, our future work aims to improve the efficiency of the SSSN.

Acknowledgement

This study was in part supported by the National Natural Science Foundation of China (NSFC) (No. 42176196) and was in part supported by the Young Taishan Scholars Program (No. tsqn201909060).

References

[1] H. Min, Y. Zhang, Y. Zhao, W. Jia, Y. Lei, C. Fan, Hybrid feature enhancement network for few-shot semantic segmentation, Pattern Recognit. 137 (2023) 109291.

[2] S. Kim, P. Chikontwe, S. An, S. H. Park, Uncertainty-aware semi-supervised few shot segmentation, Pattern Recognition (2023) 109292.

[3] H. Ding, H. Zhang, X. Jiang, Self-regularized prototypical network for few-shot semantic segmentation, Pattern Recognition 133 (2023) 109018.

[4] D. Liu, Y. Cui, W. Tan, Y. Chen, Sg-net: Spatial granularity network for one-stage video instance segmentation, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, pp. 9816–9825.

[5] Z. Qin, X. Lu, X. Nie, D. Liu, Y. Yin, W. Wang, Coarse-to-fine video instance segmentation with factorized conditional appearance flows, IEEE/CAA Journal of Automatica Sinica 10 (2023) 1192–1208.

[6] A. Shaban, S. Bansal, Z. Liu, I. Essa, B. Boots, One-shot learning for semantic segmentation, in: Procedings of the British Machine Vision Conference 2017, 2019.

[7] N. Dong, E. P. Xing, Few-shot semantic segmentation with prototype learning., in: BMVC, volume 3, 2018.

[8] X. Li, T. Wei, Y. P. Chen, Y.-W. Tai, C.-K. Tang, Fss-1000: A 1000-class dataset for few-shot segmentation, in: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020.

[9] H. Sun, X. Lu, H. Wang, Y. Yin, X. Zhen, C. G. Snoek, L. Shao, Attentional prototype inference for few-shot segmentation, Pattern Recognition (2023) 109726.

[10] S. Hong, S. Cho, J. Nam, S. Lin, S. Kim, Cost aggregation with 4d convolutional swin transformer for few-shot segmentation, in: Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXIX, Springer, 2022, pp. 108–126.

[11] K. Koffka, Principles of gestalt psychology, routledge & kegan paulltd, 1935.

[12] Q. Fan, W. Pei, Y.-W. Tai, C.-K. Tang, Self-support few-shot semantic segmentation, in: Computer Vision–ECCV 2022: 17th European Conference, 2022, Proceedings, Part XIX, Springer, 2022, pp. 701–719.

[13] J. Min, J. Lee, J. Ponce, M. Cho, Hyperpixel flow: Semantic correspondence with multi-layer neural features, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2019, pp. 3395–3404.

[14] S. Hong, S. Kim, Deep matching prior: Test-time optimization for dense correspondence, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2021, pp. 9907–9917.

[15] J. Min, D. Kang, M. Cho, Hypercorrelation squeeze for few-shot segmentation, in: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), 2021.

[16] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, et al., An image is worth 16x16 words: Transformers for image recognition at scale, arXiv preprint arXiv:2010.11929 (2020).

[17] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, 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.

[18] K. D. M. Nguyen, S. Todorovic, Feature weighting and boosting for few-shot segmentation, 2019 IEEE/CVF International Conference on Computer Vision (ICCV) (2019) 622–631.

[19] H. Wang, X. Zhang, Y. Hu, Y. Yang, X. Cao, X. Zhen, Few-shot semantic segmentation with democratic attention networks, in: European Conference on Computer Vision, 2020.

[20] K. Wang, J. H. Liew, Y. Zou, D. Zhou, 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.

[21] X. Zhang, Y. Wei, Y. Yang, T. S. Huang, Sg-one: Similarity guidance network for one-shot semantic segmentation, IEEE transactions on cybernetics 50 (2020) 3855–3865.

[22] E. Iqbal, S. Safarov, S. Bang, Msanet: Multi-similarity and attention guidance for boosting few-shot segmentation, arXiv preprint arXiv:2206.09667 (2022).

[23] J. Liu, Y. Bao, G.-S. Xie, H. Xiong, J.-J. Sonke, E. Gavves, Dynamic prototype convolution network for few-shot semantic segmentation, in:

2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2022, pp. 11543–11552.

[24] Z. Wu, X. Shi, G. Lin, J. Cai, Learning meta-class memory for few-shot semantic segmentation, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2021, pp. 517–526.

[25] J.-W. Zhang, Y. Sun, Y. Yang, W. Chen, Feature-proxy transformer for few-shot segmentation, arXiv preprint arXiv:2210.06908 (2022).

[26] X. Shi, D. Wei, Y. Zhang, D. Lu, M. Ning, J. Chen, K. Ma, 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.

[27] G. Zhang, G. Kang, Y. Yang, Y. Wei, Few-shot segmentation via cycle-consistent transformer, Advances in Neural Information Processing Systems 34 (2021) 21984–21996.

[28] S. Wang, B. Z. Li, M. Khabsa, H. Fang, H. Ma, Linformer: Self-attention with linear complexity, arXiv preprint arXiv:2006.04768 (2020).

[29] C. Wu, F. Wu, T. Qi, Y. Huang, X. Xie, Fastformer: Additive attention can be all you need, arXiv preprint arXiv:2108.09084 (2021).

[30] Z. Peng, W. Huang, S. Gu, L. Xie, Y. Wang, J. Jiao, Q. Ye, Conformer: Local features coupling global representations for visual recognition, arXiv preprint arXiv:2105.03889 (2021).

[31] L. Yang, Y. Yang, J. Yang, N. Zhao, L. Wu, L. Wang, T. Wang, Fusionnet: A convolution–transformer fusion network for hyperspectral image classification, Remote Sensing 14 (2022) 4066.

[32] X. Dong, J. Dong, M. J. Chantler, Perceptual texture similarity estimation: An evaluation of computational features, IEEE Transactions on Pattern Analysis and Machine Intelligence 43 (2020) 2429–2448.

[33] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, 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.

[34] M. Everingham, S. M. A. Eslami, L. Van Gool, C. K. I. Williams, J. Winn, A. Zisserman, The pascal visual object classes challenge: A retrospective, International Journal of Computer Vision (2014) 98–136.

[35] Z. Tian, H. Zhao, M. Shu, Z. Yang, R. Li, J. Jia, Prior guided feature enrichment network for few-shot segmentation, IEEE transactions on pattern analysis and machine intelligence 44 (2020) 1050–1065.

[36] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016.

[37] H. Touvron, M. Cord, M. Douze, F. Massa, A. Sablayrolles, H. Jégou, Training data-efficient image transformers & distillation through attention, in: International conference on machine learning, PMLR, 2021, pp. 10347–10357.

[38] G.-S. Xie, H. Xiong, J. Liu, Y. Yao, L. Shao, Few-shot semantic segmentation with cyclic memory network, in: 2021 IEEE/CVF International Conference on Computer Vision (ICCV), 2021, pp. 7273–7282.

[39] W. Liu, C. Zhang, H. Ding, T.-Y. Hung, G. Lin, Few-shot segmentation with optimal transport matching and message flow, arXiv preprint arXiv:2108.08518 (2021).