SSNet: A Novel Transformer and CNN Hybrid Network for Remote Sensing Semantic Segmentation

Min Yao, Yaozu Zhang, Guofeng Liu, and Dongdong Pang

***Abstract*—There are still various challenges in remote sensing semantic segmentation due to objects diversity and complexity. Transformer-based models have significant advantages in capturing global feature dependencies for segmentation. However, it unfortunately ignores local feature details. On the other hand, convolutional neural network (CNN), with a different interaction mechanism from transformer-based models, captures more small-scale local features instead of global features. In this article, a new semantic segmentation net framework named SSNet is proposed, which incorporates an encoder–decoder structure, optimizing the advantages of both local and global features. In addition, we build feature fuse module and feature inject module to largely fuse these two-style features. The former module captures the dependencies between different positions and channels to extract multiscale features, which promotes the segmentation precision on similar objects. The latter module condenses the global information in transformer and injects it into CNN to obtain a broad global field of view, in which the depthwise strip convolution improves the segmentation accuracy on tiny objects. A CNN-based decoder progressively recovers the feature map size, and a block called atrous spatial pyramid pooling is adopted in decoder to obtain a multiscale context. The skip connection is established between the decoder and the encoder, which retains important feature information of the shallow layer network and is conducive to achieving flow of multiscale features. To evaluate our model, we compare it with current state-of-the-art models on WHDLD and Potsdam datasets. The experimental results indicate that our proposed model achieves more precise semantic segmentation.**

***Index Terms*—Fusion features, multiscale features, remote sensing (RS), semantic segmentation.**

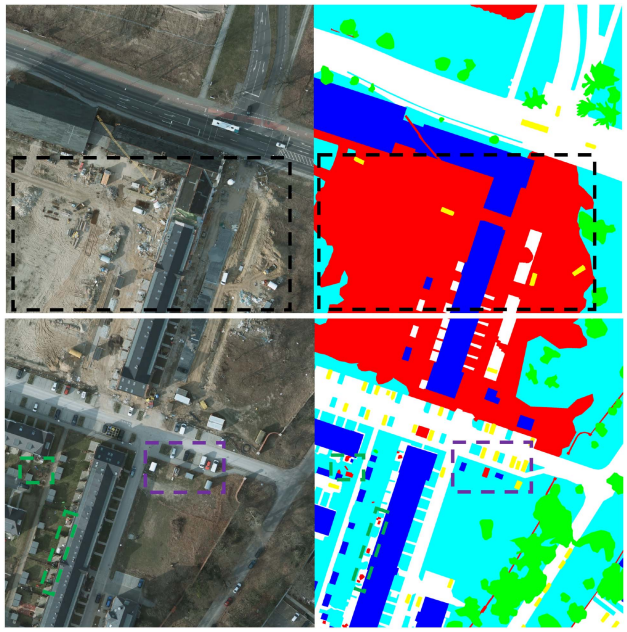

Fig. 1. RS image segmentation example. The black box frames out the complex backgrounds of the given image while the purple box and the green box show high similarity objects and tiny objects in high-resolution images. The cars, buildings and clutter in the purple box are highly similar in both shape and size, and the green boxes show some tiny objects.

I. Introduction

SEMANTIC segmentation is one of the most basic and important topics research in the fields of image processing, which is widely applied to various segmentation tasks, such as remote sensing (RS) image segmentation [2], [3] and medical image segmentation [3], [4]. For RS, semantic segmentation is also named ground-objects identification, referring to a dense prediction task. In other words, it needs to categorize each pixel in an image and then corresponds it to ground-objects of different categories. With the fast development of computer vision technology, semantic segmentation of RS image becomes a current research hotspot and a useful tool in wide ranges of applications, such as building extraction [6], land cover mapping [7], urban planning [8], environmental change monitoring [9], and agricultural production [10]. RS images contain various objects, such as airplanes, cars, roads, buildings, and trees. Due to the diverse and complex nature of ground objects, RS image segmentation still faces great challenges, including high background complexity [11], high similarity objects [12], tiny objects in high resolution images [13], as shown in Fig. 1. Recently, as an important theory to extract image features, convolutional neural network (CNN) has attracted wide attention to RS image semantic segmentation. Many works based on CNN theory have already achieved encouraging results. Take fully convolutional

Manuscript received 2 August 2023; revised 16 November 2023 and 10 December 2023; accepted 27 December 2023. Date of publication 4 January 2024; date of current version 16 January 2024. This work was supported by National Natural Science Foundation of China under Grant 61603245. *(Corresponding author: Min Yao.)*

Min Yao and Yaozu Zhang are with the College of Information Engineering, Shanghai Maritime University, Shanghai 200120, China (e-mail: minyao@shmtu.edu.cn; z_here@qq.com).

Guofeng Liu is with Baidu, Beijing 100000, China (e-mail: 937413907@qq.com).

Dongdong Pang is with the College of Resources and Environmental Engineering, Tianshui Normal University, Tianshui 741000, China (e-mail: pangdongdong1994@163.com).

The code of this work can be downloaded at https://github.com/stuyzZ/SSNet.

Digital Object Identifier 10.1109/JSTARS.2024.3349657

network (FCN) [6] for example, it is a seminal study in semantic segmentation using CNN, which performs pixel-level categorization of images. Since then, FCN has inspired many following works [24], [25], [26] and encoder–decoder structure has become a main choice for semantic segmentation. Furthermore, researchers have focused on enhancing such structure in various aspects. Specifically, [24], [26], [28], [29], [30] explored expanding the receptive field. Contextual information has been acknowledged as a crucial feature, with [31], [32], [33], [34], [35] delving into this domain. In addition, there are works on auxiliary networks, which aimed at extracting boundary information to assist in pixel classification, such as [36]. Recently, SegNext [1], based on the traditional convolutional improvement, has been proposed, which employs a similar pyramid structure to SegFormer [18] and uses multiscale convolutional features to evoke spatial attention mechanism.

On the other hand, the outstanding modeling capabilities of self-attention have brought new achievements in computer vision. Computer vision models, such as [40], [41], and [42], employed self-attention to capture dependencies between features in spatial and channel dimensions. Recently, vision transformer (ViT) [15] is a typical representation for image classification proposed by Dosovitskiy et al., which was inspired by transformer scaling successes of natural language processing (NLP) [14]. To improve ViT, various works have explored a variety of theories for modifying ViT and presented some excellent performance, such as [16], [18], [19], [43], [44], [45], [46]. Both the decoder and encoder of DETR [43] use the transformer structure, and deformable DETR [44] improves on this structure. A pure transformer backbone network in semantic segmentation, was introduced in SETR [45], and CNN-based decoders were proposed. In addition, related PE problems were explored in CPVT [46], and a dynamic positional embeddings (PEs) theory was proposed. Meanwhile, multiscale feature maps are crucial in visual tasks, so PVT1 [19] proposed a transformer backbone with multiscale, which is a significant improvement. SegFormer [18] improved ViT by introducing a hierarchical network, while using mix-FFN instead of PE, which is a remarkable achievement.

Following ViT models usually divide an image into multiple linearly embedded patches and input them into a standard transformer with PEs, resulting in encouraging performance. Transformer models are powerful at modeling global context [20], and require global features reasoning by computing self-attentions among all the tokens [21], but unfortunately deteriorate local feature details [22]. Due to this characteristic, great problems may exist, especially in semantic segmentation of RS images with complex backgrounds, or with small and highly similar objects. Here are several reasons. First, complex backgrounds can fool small objects, that is, self-attentional learning of pixels on small objects absorbs complex backgrounds noise, resulting in poor segmentation results [13]. Second, it is difficult to classify highly similar objects based on their shape, color, and pattern. Last but not least, the image local continuity can be easily lost [18].

While both CNN and transformer models excel in the field of semantic segmentation, there are differences between them. One major difference is the feature interaction mechanism. In CNN, convolutional kernels are locally connected to the input feature maps, where features only interact with their local neighbors [21]. In other words, the modeling range of CNN is limited by the receptive field of convolution kernels. Although the limited receptive fields of convolutions inevitably result in the neglect of global image features, they are more sensitive to local information, hence more sensitive to tiny objects. In addition, it is mentioned in [27] that CNNs are "texture biased" and make predictions mainly from texture in an image. Texture information plays a more important role, when distinguishing between highly similar objects and between objects with complex backgrounds. Therefore, the feature interaction mechanism of CNN has the benefit of remedying such fine details in the image that are deteriorated or neglected in transformer. Various works [17], [20], [47], [48], [49] explored the theory of fusing CNN and transformer.

In this article, to compensate the limitations of transformer in local modeling, a novel semantic segmentation network for RS images called SSNet was constructed, which obtains feature maps from two complementary mechanism, SegFormer and SegNext, thus selectively promote the convergence and flow of both. In addition, feature fuse module (FFM) and feature inject module (FIM) are designed in our network. The purpose of FFM is to enhance the local details in transformer by fusing the features of CNN and transformer, then inject the fused features into transformer. FIM selectively acquires multiscale feature information to inject into CNN as to enhance the flow between global information and local information in each stage. Furthermore, a CNN-based decoder is constructed to restore the feature map size and acquire the semantic segmentation results. The primary contributions of this work are as follows.

1) We present a novel network designed specifically for RS semantic segmentation, which retains local and global information in both branches. It maximizes the injection of complementary information from CNNs into self-attention to obtain excellent segmentation results.
2) To enhance the detail information of transformer, four branches are designed in FFM to handle two-style features. A combination of attention mechanisms and pooling layers is used to wake up the module's ability to orchestrate global and local information.
3) We apply strip convolution and squeeze-and-excitation (SE) module [40] to compensate for CNN's neglect of global information during downsampling in FIM. SE module allows selective focus on channels, while depthwise strip convolution increases the ability to capture tiny objects.
4) We propose a CNN-based decoder that combines ASPP module to preserve more details. With skip connections within encoder and decoder, it improves deep features, recover feature maps to the original image resolution, and achieves competitive results.

II. Related Works

In this section, we make a compendium of semantic segmentation models from three different perspectives. Section II-A introduces the use of CNN models in semantic segmentation. Section II-B is an introduction to the model of self-attention and

transformer. Section II-C describes the model combining both self-attention and CNN.

A. CNN-Based Methods for Semantic Segmentation

With the prevalence of computer vision in RS, semantic segmentation for RS image based on CNN has garnered significant attention. Semantic segmentation is systematically studied in a seminal work FCN [6]. FCN has an encoder–decoder architecture, which has inspired many works adopting this architecture in semantic segmentation. The encoder plays a crucial role in feature representation learning and like most other CNNs designed for computer vision, it comprises stacked convolutional layers. Considering the computational cost, a strategy of gradually reducing the feature maps resolution is applied, allowing the encoder to learn more semantic information by gradually increasing the receptive field. Although FCN has high efficiency and low complexity, the semantic segmentation results are not satisfying enough, because this network ignores the relationship between pixels and does not consider spatial consistency, leading to misclassification of object categories. After FCN, researchers focused on improving it in different ways.

PSPnet [26] and Deeplab series [24], [28], [29], [30] are examples that improve FCN, in terms of expanding the receptive field. PSPnet proposed the pyramid pooling module which incorporates a (1×1) convolution operation for global pooling and upsampling and concat operations, incorporating contextual information at different scales and increasing the perceptual field. PSPnet reduces the probability of missegmenting image categories in the FCN network.

The Google team has proposed a series of semantic segmentation algorithms, among which [24], [28], [29], [30] are very widely used. DeepLabV1 [24] is based on the improvement of CNN, which solves the problem of repeated pooling and downsampling leading to resolution degradation. This makes the image location information unrecoverable, and the use of conditional random field (CRF) improves the segmentation of fine details. DeepLabV2 [28] improves the network architecture based on DeepLabV1, mainly proposing the ASPP to address the problem of the existence of multiscale objects in images. DeepLabV3 [29] builds on DeepLabV2 by deleting the fully connected CRFs and improves the ASPP module by parallelizing the dilated convolution with different dilate rates. DeepLabV3+ [30] proposes a depthwise separable convolution, which has been highly influential in the field, and the model is obtained by adding decoder on top of DeepLabV3, significantly improving network performance. Depthwise separable convolution can greatly reduce computation while maintaining performance and allowing better recovery of object edge information.

Contextual information is also important for semantic segmentation performance, and many methods [31], [32], [33], [34], [35] explore context dependencies to obtain better segmentation results. EncNet [33] presents a new context encoding module and enhances model effectiveness by incorporating global contextual information. The main process is to do semantic segmentation by first predicting the category information present in the image and then executing contextual encoding with feature attention mechanism. Authors of adaptive pyramid context network (APCNet) [35] argue that global guided local affinity is vital for construction of semantic features, which was neglected in previous research works. Considering this, the authors proposed a novel solution, the APCNet, dedicated to advancing semantic segmentation, which uses multiple adaptive context modules (ACMs) to adaptively construct multiscale contextual representations. In particular, each ACM uses the global image as a guide to every subregion, then uses these affinities to compute the context vector. This favors the further construction of adaptive and multiscale contextual representations.

The prediction of object boundary is likewise an aspect worth exploring. In GSCNN [36], the authors argue that color, shape, and texture contain various kinds of information that are critical to understanding an image, so it may not be ideal to process them together in CNN. Hence, GSCNN proposes a novel dual-stream CNN architecture with a shape stream and a classical stream processing information in parallel, where the shape stream shape information apart. This architecture introduces a novel type of gate to establish connections between the intermediate layers of the two streams. This new type of gate is the key component in the architecture. Also, thanks to the sharper boundary prediction, GSCNN greatly improves the segmentation performance on thin and smaller objects.

In conclusion, FCN and corresponding variants were widely used in RS semantic segmentation works [37], [38], [39]. The exploration of semantic segmentation models in different aspects has also contributed to the progress of RS images. CNN has demonstrated significant potential in the realm of RS semantic segmentation, owing to its modeling effectiveness and extensiveness.

B. Self-Attention and Transformer in Vision

Recently, self-attention mechanisms have been prevalent in computer vision tasks. Inductive bias of CNN may impose limitations on the model's ability to extract long-range spatial dependencies, thereby degrading its performance. To address this issue, researchers have explored the incorporation of self-attention to aid CNN in feature extraction.

SENet [40] leverages a global average pooling layer to establish connections between channels and completes the recalibrating of original features of the channel dimension dynamically and adaptively, which pays attention to the dependence at the channel level of the model for the first time. CBAM [41] uses channel-level and spatial-level attention modules to refine adaptive features. The channel attention module (CAM) emphasizes the relationship of feature maps between different channels, and the spatial attention generates a spatial attention map, which allows the model to focus on important features. DANet [42] proposes a dual attention network, including position attention module (PAM) and CAM. These two modules are employed to capture the feature dependencies in spatial dimension and channel dimension, respectively.

The work in [14] is one of the dominant architectures in NLP that utilizes multihead attention to establish long-range dependencies. Inspired by the transformer scaling successes in NLP, ViT [15] is a typical representative of image classification task and has achieved outstanding performance. But for dense prediction tasks, it requires a higher training cost and only outputs lower solution features.

To accommodate dense prediction tasks, some researchers modify ViT architecture and achieved state-of-the-art performance. For example, DETR [43] leverages the transformer decoder to frame object detection as an end-to-end dictionary lookup problem, where learnable queries are used, effectively eliminating the requirement for handcrafted processes, such as nonmaximum suppression. Building upon DETR, deformable DETR [44] incorporates a deformable attention layer to emphasize a sparse set of contextual elements, leading to faster convergence and improved performance. SETR [45] is the pioneering network that employs a pure transformer structure as backbone in semantic segmentation, which constructs three different CNN-based decoders for backbone to obtain dense predicted results. By combining the transformer-based encoder with a simple decoder, a powerful segmentation model can be achieved. CPVT [46] advocated a novel positional encoding (PE) scheme, named conditional positional encoding (CPE). Different from PE used in previous works, such as ViT, which are predefined and input-agnostic, CPE is generated in a dynamic way and is conditional on the local neighborhood of an input token. As a result, PE varies depending on input size and ensures the desired property of translation-invariance.

PVT1 [19] generates multiscale feature maps by introducing pyramid modules to the transformer framework, based on which a pure transformer backbone is proposed for dense prediction tasks. Although PVT1 reduces the computational cost to some extent, its complexity remains quadratic with the image size. SegFormer proposes a hierarchical transformer structure without PE and significantly reduce computational complexity. It avoids PE interpolation, as it can adversely affect the model's performance, under the condition where the test resolution differs from the training one. In semantic segmentation, SegFormer demonstrates impressive performance as a dedicated transformer.

C. Self-Attention and CNN

Local feature details are tended to be neglected by the self-attention mechanism in transformer [22], [47]. In contrast, CNN possesses distinct advantages in local modeling and translation-invariance. Since TransUNet [20] generates a new encoder for improved semantic segmentation by sequentially concatenating the two, this would allow it to benefit from transformer self-attention while retaining the effective encoder–decoder structure of U-Net. TransFuse [48] performs parallel concatenation of them and fuses the relevant features by a BiFusion module which incorporates both self-attention and multimodal fusion mechanisms. Furthermore, TransFuse employs a simple progressive upsampling method to recover the spatial resolution. UNetFormer [17] selects CNN as the encoder and also proposes an efficient global–local attention mechanism to model global and local information in the transformer-based decoder. ST-UNet [49] uses swin transformer [16] to assist UNet [25], where CNN is used as the primary encoder and swin transformer encoder is used as an auxiliary encoder. DS-Net [47] applies self-attention and convolution as dual-resolution processing paths, in which the self-attention path is designed to capture local fine-grained details, while the convolution path aims to explore features from a global perspective.

Different from existing excellent works, we design a novel network that preserves both CNN and transformer features. In this design, both local and global information can be extracted at each stage, and the advantages of the two were fully utilized to obtain better semantic segmentation results.

III. Methodology

In this section, we provide a comprehensive overview of the architecture of SSNet. In Section III-A, we generally introduce the design of our pipeline. Subsequently, in Section III-B and III-C, we provide an in-depth introduction to the general architecture schemes for SegFormer and SegNext. Then, in Section III-D and III-E, we introduce FIM and FFM, respectively, in terms of both mathematical principles and workflows. Finally, in Section III-F, we illustrate the structure of the CNN-based decoder.

A. Overall Architecture

Fig. 2 illustrates the proposed network, which follows the encoder–decoder paradigm, connecting both modules through skip connection layers. As depicted in Fig. 2, this framework mainly contains three modules as follows.

1) A hybrid encoder of the SegFormer and SegNext.
2) FFM used to fuse the feature from transformer and CNN; FIM utilized to inject multiscale information into the CNN feature map from the transformer branch.
3) A CNN-based decoder for progressive recovery of feature map size and prediction of segmentation results.

Unlike transformer, such as ViT and STER, where backbones only generate single-resolution feature maps when given an input RS image, our backbone is to generate multiscale features in four stages. These different resolution feature maps can enhance the property of semantic information extraction. In detail, for a given RS image $(3 \times H \times W)$, a hierarchical feature map F_n with a resolution of $(C_n \times H/2^{n+1} \times W/2^{n+1})$ are obtained.

For FFM, it fuses two-style features in each stage, then passes the output features into the SegFormer to increase the transformer abilities. And FIM injects the multiscale features into the SegNext to enhance the CNN global perception capability. CNN-based decoder is used to restore the feature map size and inject the shallower informative features from the shallower layers of the encoder into the decoder by skipping connections to obtain a more detailed feature map.

B. Transformer-Based Encoder

Unlike the generic transformer, the SegFormer is more flexible and efficient, so we adopt SegFormer to extract global

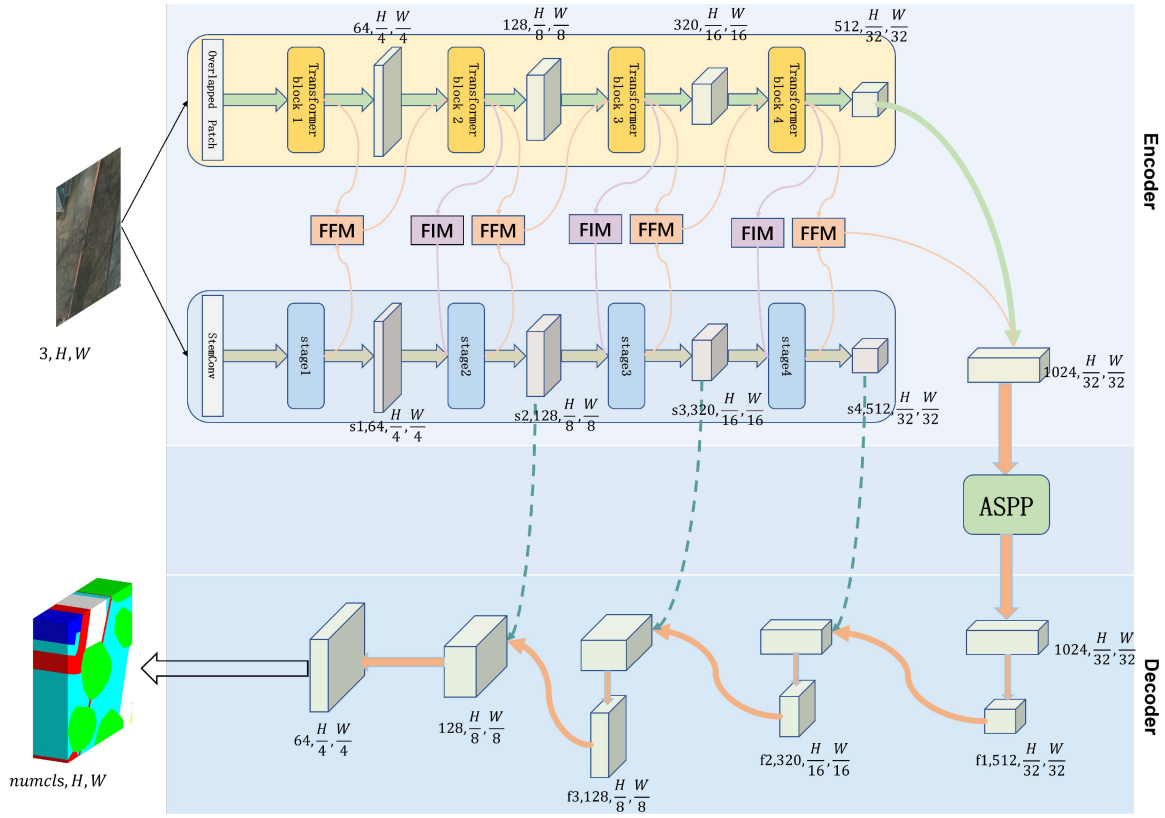

Fig. 2. Proposed framework consists of three main modules: A hierarchical hybrid encoder of transformer and CNN to extract coarse and fine features; FFM and FIM ensure the flow of information between CNN and transformer; a CNN-based decoder to recover the size of the feature map.

features, which uses four stages to extract the feature map. Each stage generates feature maps at different resolution ratios than the original image, including 1/4, 1/8, 1/16, 1/32. For an RS image $(3 \times H \times W)$, it was first split into patches of size (4×4). We use such small patches instead of size (16×16) employed in ViT, since smaller patches perform better for intensive prediction tasks. These patches serve as inputs to the hierarchical transformer encoder, resulting in the generation of multilevel feature maps.

As shown in Fig. 3, the main component of SegFormer encoder is the transformer block, which mainly comprises efficient self-attention and mix-FFN. During the original process, the self-attention is estimated as follows:

$$\text{Attention}(Q, K, V) = \text{Softmax}\left(\frac{QK^T)}{\sqrt{d_{\text{head}}}}\right) V. \quad (1)$$

In (1), the vectors query(Q), key(K), and value(V) are fundamental components of the attention mechanism, commonly used as building blocks. Each of these vectors Q, K, and V has dimensions of $(C \times H \times W)$, where $(H \times W)$ represents the sequence length. Wang et al. [19] proposed the sequence reduction process to improve original multihead self-attention, which brings the computational complexity down from $O(N^2)$ to $O(N^2/R)$, where R is a reduction ratio. In our experiments, we use the process and set the parameter R to [64, 16, 8, 1]. Moreover, for semantic segmentation, PEs are not required. They introduce mix-FFN which combines a (3×3) convolution and multilayer perceptron (MLP) into each feedforward network

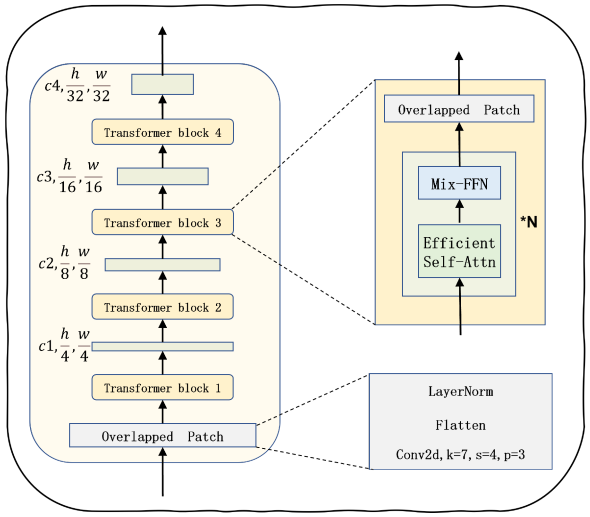

Fig. 3. Structure of the SegFormer encoder. The main component of SegFormer encoder is the transformer block, which mainly comprises efficient self-attention and Mix-FFN.

(FFN). Mix-FFN can be represented as follows:

$$X_{\text{out}} = \text{MLP}\left(\text{GELU}\left(\text{Conv}_{3\times3}(\text{MLP}(X_{\text{in}}))\right)\right) + X_{\text{in}} \quad (2)$$

where X_{in} and X_{out} represent the input and output feature maps, so the feature map with approximate PE can be obtained by mix-FFN. By choosing diverse hyperparameters and varying the number of transformer blocks in each stage, we can obtain five

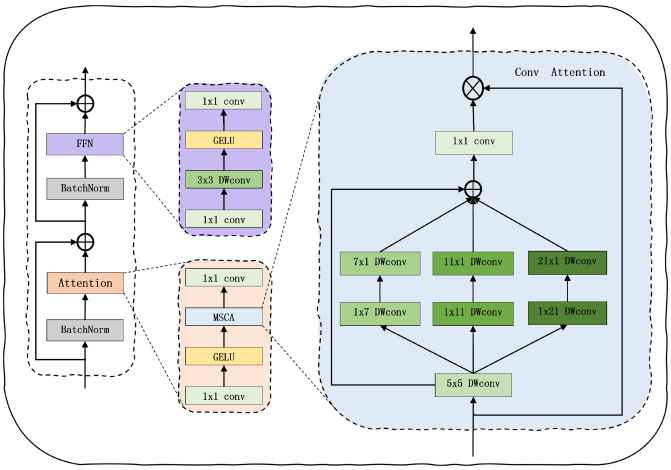

Fig. 4. Structure of the core component MSCA in SegNext consisting of traditional convolution.

SegFormer backbones with various complexities. We choose mit-B5 as our transformer backbone, which has a deeper network layer and the best performance.

C. CNN-Based Encoder

Thanks to the improvement of traditional convolutional blocks and the use of multiscale feature convolution following [50], SegNext achieves a simple and efficient performance.

As shown in Fig. 4, the SegNext encoder is composed of three key components as follows.

1) A depthwise convolution to consolidate local information.
2) Multibranch depthwise strip convolutions, which enable the capture of multiscale feature maps. Strip convolutions are a form of 1-D convolution operation, distinct from the common 2-D convolution kernels, typically taking the shape of $(n \times 1)$ or $(1 \times n)$. The specific description can be found in the FIM presentation and Fig. 8. They are commonly applied for horizontal or vertical processing of feature maps.
3) A (1×1) convolution facilitating the modeling of relationships in different channels. It can be written as follows:

$$\mathrm{Out} = \mathrm{Conv}_{1\times1}\left(\sum_{i=0}^{2}(\mathrm{Scale}_i(\mathrm{DW{-}Conv}(F)))\right) \otimes F \tag{3}$$

where $\mathrm{DW{-}Conv}$ is depthwise convolution (5×5), and Scale_i denotes the convolution of three different scales respectively. Like some common semantic segmentation networks, SegNext adopts four stages and each produces feature maps at different resolutions. We choose SegNext-B as the CNN encoder because it has the optimal performance with lower computational requirements and can generate feature maps of the same size as SegFormer.

D. Feature Fuse Module

Highly similar tiny foreground and confusing objects are prevalent in RS images, which may significantly impact the semantic segmentation quality. For such images, an adequate combination of local and global information, such as shape, color and texture, is needed to achieve better segmentation effects. However, SegFormer approach weakens the local feature detail to a certain extent, even if it uses smaller patches, unlike ViT.

To improve the quality of segmentation, by taking full advantages of local and global information, we propose FFM. The FFM fuses features from both the CNN and transformer branches and injects them back into the transformer branch, enriching the local details of the transformer branch. We have designed two distinct streams to handle features from the CNN and transformer branches separately.

As shown in Fig. 5, the features from transformer go through the pooling layer and PAM [42] to obtain more local information, where PAM captures the spatial relationships among different positions within the feature map. Max pooling can retain more texture features, and avg pooling can better retain the overall features and highlight the background information. On the second stream, the CAM and strip convolution receive the features from CNN, where CAM explicitly models interdependencies between channels, and the depthwise strip convolution can obtain fine feature representations. Finally, we multiply the features computed by the two streams and change the dimensions by an (1×1) convolution as the final output. This can be represented as follows:

$$\mathrm{Out1} = (\mathrm{MaxPool}(f_t) + \mathrm{Avgpool}(f_t)) \cdot \mathrm{CAM}(f_c) \tag{4}$$

$$\mathrm{Out2} = \left(\sum \mathrm{str{-}Conv}(f_c)\right) \cdot \mathrm{PAM}(f_t) \tag{5}$$

$$\mathrm{Output} = \mathrm{Conv}_{1\times1}(\mathrm{Out1} \cdot \mathrm{Out2}) \tag{6}$$

where f_t and f_c represent the feature maps from transformer and CNN, respectively. $\mathrm{str{-}Conv}$ is depthwise strip convolution of different scales. The final results of the two streams are $\mathrm{Out1}$ and $\mathrm{Out2}$, respectively.

PAM, illustrated in Fig. 6, first takes the input feature map A and passes it through a convolution layer, resulting in three different feature maps, namely B, C, and D, where B, C, and D are all of size $(C \times H \times W)$, subsequently reshaping them from $(C \times H \times W)$ to $(C \times N)$, where $N = H \times W$. Second, the transposed features $B^T \in R^{(N\times C)}$ of $B \in R^{(C\times N)}$ are multiplied with $C \in R^{(C\times N)}$ and the weights $S \in R^{(N\times N)}$ are obtained by softmax. Its calculation process can be described as follows:

$$\mathrm{S}_{ij} = \frac{\exp(B_i \cdot C_j)}{\sum_{i=1}^{N}(\exp(B_i \cdot C_j)} \tag{7}$$

where S_{ij} measures the ith position's impact on jth position.

Then, the feature map $D \in R^{(C\times N)}$ and the transpose of the weight $S \in R^{(N\times N)}$ are multiplied together with a scale parameter α. After reshaping the result from $(C \times N)$ to $(C \times H \times W)$, we perform an elementwise sum operation with the original input feature map A to obtain the final output

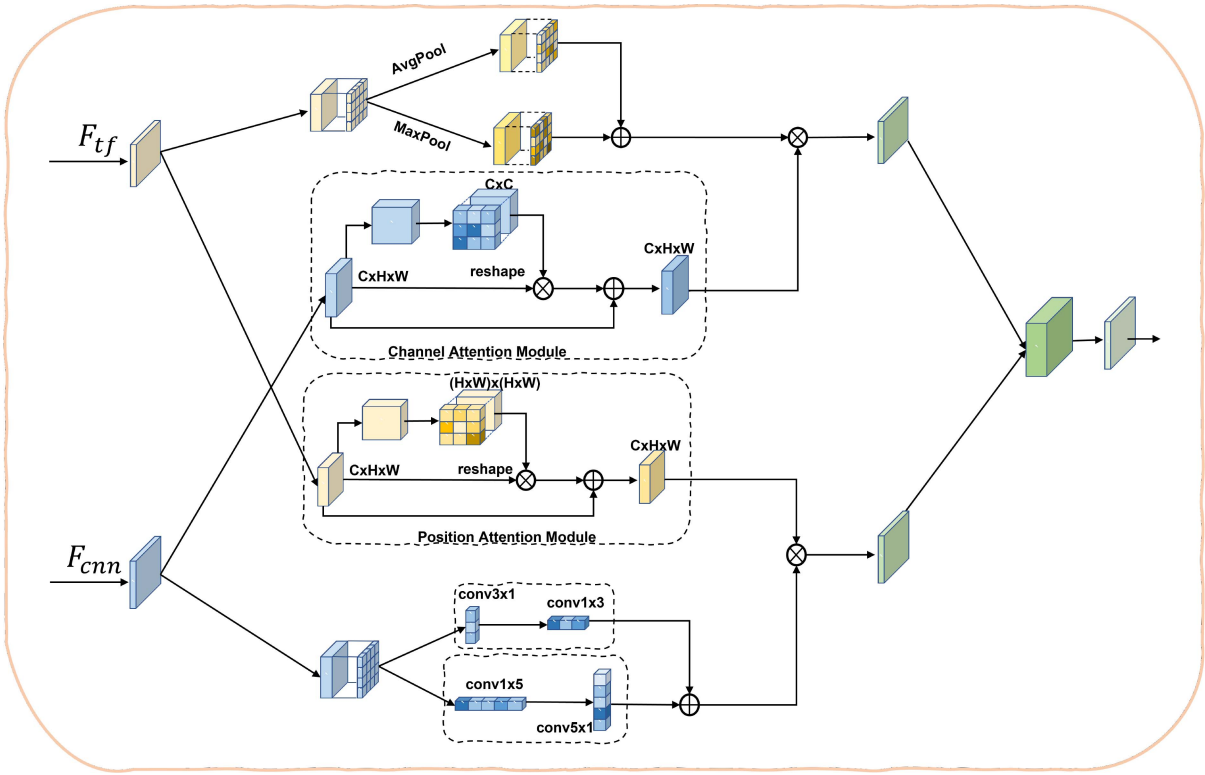

Fig. 5. Structure of FFM. FFM takes full advantage of local features and global information by two different streams.

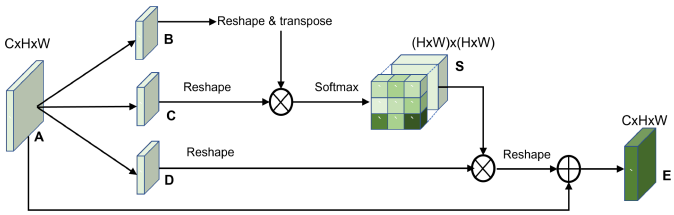

Fig. 6. Structure of PAM.

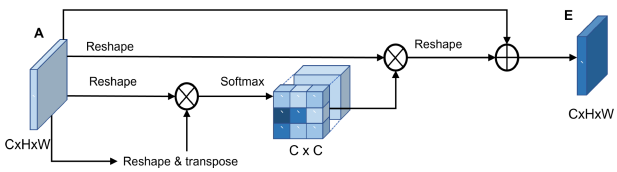

Fig. 7. Structure of CAM.

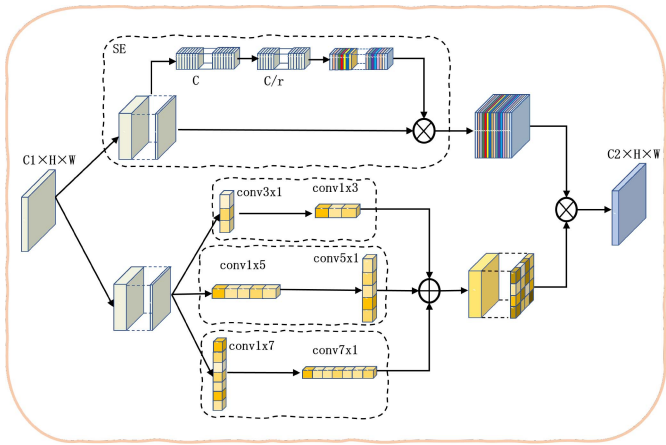

Fig. 8. Structure of FIM.

$E \in R^{(C\times H\times W)}$, as depicted in the following:

$$\mathrm{E}_j = \alpha \sum_{i=1}^{N} (\mathrm{S}_{ij}\mathrm{D}_i) + \mathrm{A}_j \tag{8}$$

where α is initialized as 0 and gradually learns to assign more weighs by (8). It can be inferred that the resulting feature E for each location is a weighted sum of all location features and the original features [42].

The structure of CAM is depicted in Fig. 7. We first reshape the input feature map A from $(C \times H \times W)$ to $(C \times N)$, where $N = H \times W$. Then, we perform a matrix multiplication between $A \in R^{(C\times N)}$ and the transpose of $A^T \in R^{(N\times C)}$. By applying the softmax layer on the result, we obtain the channel attention map $X \in R^{(C\times C)}$, which can be represented as follows:

$$\mathrm{X}_{ij} = \frac{\exp(A_i \cdot A_j)}{\sum_{i=1}^{C} (\exp(A_i \cdot A_j)} \tag{9}$$

where X_{ij} measures the ith channel's impact on the jth channel.

In addition, we perform a multiplication between $X \in R^{(C\times C)}$ and $A \in R^{(C\times N)}$, and then reshape the result to obtain $R \in R^{(C\times H\times W)}$. Afterward, we multiply the $R \in R^{(C\times H\times W)}$ by β which is a scale parameter and perform an elementwise sum operation with A to compute the final output $E \in R^{(C\times H\times W)}$.

This process can be represented as follows:

$$\mathrm{E}_j = \beta \sum_{i=1}^{C} (\mathrm{X}_{ij} \mathrm{A}_i) + \mathrm{A}_j \tag{10}$$

where β learns a weight from 0. Equation (10) implies that the final features of each channel are computed as the weighted sum of the features from all channels and the original features [42].

E. Feature Inject Module

With the aim of leveraging the complementary nature of the two-style features, our work consecutively establishes FIM to integrate multiscale information from the transformer branch into the feature maps of CNN. This approach strengthens the global perception capability of the CNN branch and facilitates the efficient flow of feature maps between CNN and transformer, as depicted in Fig. 8.

As done in [1], [51], for every FIM branch, we employ three depthwise strip convolutions to extract information, which can approximate the standard depthwise convolution of a large kernel and thus remains lightweight. That is to say we only need a pair of (7×1) and (1×7) convolutions to approximate the effect of a convolution with (7×7). This is one reason for the usage of depthwise strip convolution. In addition, there are many strip-like objects in the segmentation scenes. Therefore, strip convolution can be used as a complement to the mesh convolution and helps extract strip-like features [51], [52]. In addition, we introduce the SE module [40]. For SE, the main focus is to calculate the channelwise weights for each feature map that enters. With the addition of SE, by learning the correlation between channels, the network is supposed to focus on those channels that need more attention. FIM can be written as follows:

$$\mathrm{Out} = \left(\sum \mathrm{str\text{-}Conv}(f) \right) \cdot \mathrm{SE}(f) \tag{11}$$

where $\mathrm{str\text{-}Conv}$ means depthwise strip convolution of different sizes.

The SE module includes two main processes which are SE operations. The feature $f \in R^{(C \times H \times W)}$ is squeezed to produce a tensor of $p \in R^{(C \times 1 \times 1)}$. Therefore, each element in the vector encodes the global information of its corresponding channel, and the cth element of p is calculated using the following formula:

$$p_c = \mathrm{F}_{\mathrm{gp}}(f_c) = \frac{1}{\mathrm{H} \times \mathrm{W}} \sum_{i=1}^{H} \sum_{j=1}^{W} f_c(i, j) \tag{12}$$

where i and j refer to the position coordinates of the elements in feature f, and F_{gp} means global pooling layer.

Next, we change the dimensionality of the features through two different fully connected layers. Then, the weights of each channel in the input feature layer are obtained through the sigmoid function. In the end, the final output is obtained by multiplying the weights with the original input feature f, and its detailed structure can be expressed as follows:

$$\mathrm{V} = \mathrm{Sigmoid}(\mathrm{FC}_2(\mathrm{ReLu}(\mathrm{FC}_1(p)))) \otimes f \tag{13}$$

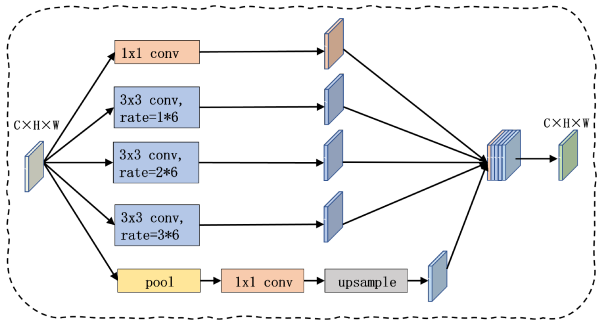

Fig. 9. Structure of ASPP.

where FC means fully connected layer. The differences between FC_2 and FC_1 are that numbers of neurons are set to (C/r) and C, respectively. In our SE modules, the reduction ratio r is set to 16.

F. CNN-Based Decoder

Classical semantic segmentation U-shaped architecture is adopted in our work, and we employ a CNN-based decoder to progressively recover the feature map size and predict the segmentation result. But encoders and decoders play different roles in feature extraction, with encoders supporting the extraction of shallow features, such as color, texture and edges, while decoders are better at deep information, that is, semantic information. To keep the transmission of details and to promote the interaction of multiscale characteristics, this study uses skip connections to fuse shallow features with deep features between CNN-based encoder and decoder. By employing skip connections, we keep the transmission of details and improve the communication of multiscale features, achieving the fusion of shallow and deep features between the CNN-based encoder and decoder. We aggregate feature maps from four stages with different resolutions.

In addition, ASPP is an excellent module in semantic segmentation which samples the given inputs in parallel with dilation convolutions of different dilation rates. In our work, to obtain multiscale contextual information, ASPP is added after the encoder. ASPP is shown in Fig. 9, which includes global pooling operation, (1×1) convolution, upsampling and concat operations. In detail, the feature maps passed to ASPP are first passed through four convolutional layers with different dilation rates and a pooling layer. Then, the obtained feature maps are concat to obtain feature maps containing multiscale contextual information for subsequent segmentation prediction.

The structure of the CNN-based decoder is depicted in Fig. 2. First, it keeps the resolution of the feature maps in CNN encoder stages 2–4, which are (128, H/8, W/8), (320, H/16, W/16), (512, H/32, W/32), respectively. The output feature size of the ASPP module is (1024, H/32, W/32). To change the output feature channel dimension of ASPP from 1024 to 512, a (3×3) convolution is utilized, while keeping the resolution. Let us denote the obtained feature as $f1$, then $f1 \in R^{(512, H/32, W/32)}$. At this point, $f1$ and $s4$ which are the feature maps output by CNN encoder stage 4, have the same size. Then, we merge $f1$

with $s4$ in a way of elementwise addition, and the result size is (512, H/32, W/32). To gradually recover to the original image size and fuse the feature maps of $s3$, we first change the channel dimension of result from 512 to 320 by (1×1) convolution, and subsequently upsample that to (H/16, W/16) using bilinear interpolation and the result can be named as $f2$ whose size is (320, H/16, W/16). Similarly, then we merge $f2$ with $s3$, and the result size is (320, H/16, W/16). Next, the size is altered to (128, H/8, W/8) by an (1×1) convolution and bilinear interpolation, denoted as $f3$. Similarly, we fuse the features of $f3$ and $s2$ by way of elementwise addition to obtain a feature map (128, H/8, W/8), and we then restored the feature map by convolution and bilinear interpolation and output the final result.

IV. Experimental Setup

A. Datasets

The proposed model has been evaluated on datasets of WHDLD and Potsdam, respectively.

1) WHDLD: WHDLD, released by Wuhan University in 2018, is a dense labeling dataset suitable for multilabel tasks, such as RS image retrieval and classification, as well as pixel-based tasks, such as semantic segmentation [53], [54]. Each image in the dataset is labeled with six class labels, including building, road, pavement, vegetation, bare soil, and water. The dataset comprises a total of 4940 images, each sized at 256 × 256 pixels. To ensure proper training, validation and testing, we split the dataset into a training set (3000 images), a validation set (1000 images), and a test set (940 images).

2) Potsdam: Potsdam, situated in northeastern Germany, is characterized by large buildings, narrow streets, and dense traffic. The Potsdam dataset has a ground sampling distance of 5 cm and consists of 38 patches, each measuring 6000 × 6000 pixels. Within this dataset, six classes of objects are labeled, namely building (blue), car (yellow), low vegetation (cyan), impervious surface (white), tree (green), and background (red). Following [49] and [56], we utilized 24 images from the dataset for training, which were cropped into 13 824 images of size 256 × 256 pixels. The remaining 14 images were used for verification and were likewise cropped into 8064 images, each measuring 256 × 256 pixels.

B. Evaluation Metrics

For each method, we evaluate model performance using the mean intersection over union (MIoU) and accuracy (Acc) metrics. These evaluation indicators are computed using the accumulated confusion matrix, which are calculated as follows:

$$\mathrm{MIoU} = \frac{1}{N}\sum_{k=1}^{N}\frac{\mathrm{TP}_k}{\mathrm{TP}_k + \mathrm{FP}_k + \mathrm{FN}_k} \tag{14}$$

$$\mathrm{Acc} = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{TP} + \mathrm{FN} + \mathrm{FP} + \mathrm{FN}} \tag{15}$$

where TP, FP, TN, and FN indicate the true positive, false positive, true negative, and false negatives, respectively.

C. Implementation Details

For our trials in present research, all experiments were performed using PyTorch on our lab device, which is equipped with an Intel i9-10900 processor, an RTX3090 graphics processor, and 64G of RAM. All models are our own PyTorch-based implementations, and we did not pretrain them. The AdamW algorithm was selected as the gradient decent optimizer algorithm for setting model parameters. The initial learning rate is 0.0006, and weight decay is 0.001 with loss function of soft crossentropy.

V. Experiments and Results

We would like to emphasize that we trained all the networks on our lab devices and did not use any pretrained weights. The ablation experiments were conducted using the WHDLD and Potsdam datasets with 300 epochs.

A. Effect of Encoder Structure

We first analyzed the influence of various encoders, including SegNext and various shallower SegFormer. The results are presented in Table III, which demonstrates the performance of various types of encoders. As the Table III show, we can initially observe that the best-performing architecture is the fusion of SegFormer (MiT-B5) and SegNext, while the shallowest SegFormer (MiT-B0) exhibits the lowest performance. In addition, the standalone SegNext outperforms SegFormer (MiT-B4) slightly. In terms of MIoU and ACC, the fused architecture, which is SegFormer (MiT-B5) and SegNext, have MIoU and ACC scores of 60.63% and 87.16% on WHDLD, and 76.04% and 84.50% on Potsdam, respectively. On WHDLD the fused architecture outperforms the standalone SegNext by 0.8% and 1.45%, and it also surpasses the deepest SegFormer by 0.22% and 0.64%. Similarly, on Potsdam, it outperforms the other two standalone architectures by 2.54% and 2.61%, and 0.48% and 0.82%, respectively. It turns out that the encoder combining CNN and transformer can capture more information favorable for semantic segmentation.

B. Effect of FFM

To explore the impact of FFM in our model, we conducted a comparison of the semantic segmentation results with and without the incorporation of FFM. Table I shows these data, on WHDLD and Potsdam, where MIoU and ACC improve by 0.67% and 1.02%, 0.84%, and 0.99%, respectively. From the point of view of details, on WHDLD we find that FFM better facilitates the segmentation of road, pavement and build, with 1.57% and 1.08% and 0.92% improvement in MIoU for road and pavement and build, respectively. From the visualization in Fig. 10, it can be found that road and pavement are highly similar and indistinguishable in some ways, and after integrating FFM, the segmentation performance of confusing objects is significantly improved.

TABLE I
ABLATION EXPERIMENT OF THE PROPOSED MODULES ON WHDLD AND POTSDAM VALIDATION SET

Network	Modules		IoU%(WHDLD)						Evaluation metrics			
	FFM	FIM	Build	Road	Pavement	Vegetation	Bare Soil	Water	WHDLD		Potsdam	
									MIoU%	Acc%	MIoU%	Acc%
SegFormer+SegNext			54.82	57.71	41.25	81.33	36.76	93.34	60.86	87.36	76.33	84.69
SegFormer+SegNext+FFM	✓		55.74	59.28	42.33	81.52	37.02	93.31	61.53	88.38	77.17	85.68
SegFormer+SegNext+FIM		✓	55.26	58.88	41.91	81.42	36.98	93.30	61.29	87.53	76.68	85.35
SegFormer+SegNext +FFM+FIM	✓	✓	56.00	59.76	42.64	82.03	37.28	93.90	61.93	88.62	77.67	85.92

TABLE II
ABLATION EXPERIMENT OF SKIP CONNECTION ON WHDLD VALIDATION SET

Network	skip connection	Evaluation metrics			
		WHDLD		Potsdam	
		MIoU%	Acc%	MIoU%	Acc%
SegFormer +SegNext +FFM+FIM	No Skip	60.85	85.10	76.32	82.51
	s4+f1	61.02	86.44	76.39	83.76
	s3+f2	61.62	88.31	77.28	85.62
	s2+f3	61.28	86.77	76.65	84.03
	Three Skips	**61.93**	**88.62**	**77.67**	**85.92**

Bold values in the table are the highest values among data in the column.

TABLE III
ABLATION EXPERIMENT OF ENCODER ON WHDLD AND POTSDAM VALIDATION SET

Encoder	Decoder	WHDLD		Potsdam	
		MIoU%	Acc%	MIoU%	Acc%
SegNext	MLP	59.83	85.71	73.50	81.89
SegFormer(Mit-B0)	MLP	44.30	65.93	56.54	64.91
SegFormer(Mit-B1)		49.98	74.46	62.87	72.18
SegFormer(Mit-B2)		55.07	82.62	69.45	80.21
SegFormer(Mit-B3)		58.51	84.29	73.33	81.70
SegFormer(Mit-B4)		59.58	86.15	74.61	83.52
SegFormer(Mit-B5)		60.41	86.52	75.56	83.68
SegFormer(Mit-B5) +SegNext	MLP	**60.63**	**87.16**	**76.04**	**84.50**

Bold values in the table are the highest values among data in the column.

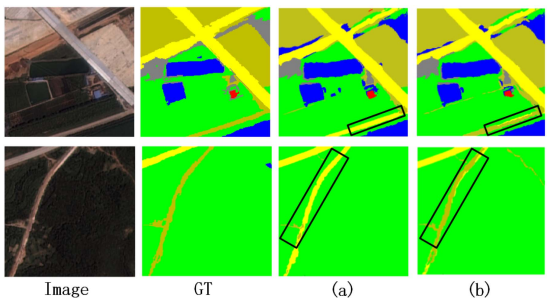

Fig. 10. Comparison of segmentation results before and after using FFM in the proposed network. The first column has the original RS images and the second column has ground truth segmentation results. (a) Results without FFM. (b) Results with FFM.

C. Effect of FIM

We conducted an analysis of the effect of FIM, as presented in Table I. First, when using the FIM independently, the segmentation results on WHDLD show an improvement of 0.43% on MIoU and 0.17% on Acc. On Potsdam, MIoU and ACC were 76.68% and 84.65%, an increase of 0.35% and 0.66%, respectively. Moreover, the segmentation results of road and pavement on WHDLD are significantly improved after integrating FIM; MIoU achieves growth of 1.17% and 0.66% for road and pavement, respectively, thanks to the depthwise strip convolutions in FIM, which are more sensitive to the strip-like objects. Although the results are better than the original model after adding FIM, they are still slightly inferior to the result using FFM. This is because FFM incorporates both global and local information, while FIM is more focused on strip-like objects.

In addition, we can observe the joint effect in Table I. Adding both FFM and FIM results in an increase in the MIOU and Acc indicators of the two datasets, respectively, which exceeds the performance when adding FFM or FIM alone. Specifically, on WHDLD, MIoU and Acc increase by 1.07% and 1.26%, respectively, and on Potsdam, they increase by 1.43% and 1.23%, respectively. These results indicate that incorporating both FFM and FIM enhances the flow of different information between CNN and transformer, leading to improved semantic segmentation performance of the model.

D. Effect of the Skip Connection

In the present part, we evaluate the effectiveness of the skip connections in our model. As shown in Fig. 2, we introduce a total of three skip connections in the decoder, s2, s3, s4, which are combined with f3, f2, f1 in turn. To avoid additional interference, we keep the layers of the decoder unchanged and only disable the integration of skip connections. We conducted five experiments. The first one did not use any skip connections, and then we added (s4+f1), (s3+f2), and (s2+f3) separately, and finally, we added three skips to the model.

In a word, the addition of any skip connection leads to an improvement in the evaluation indicators of the model. Specifically, the proposed model with three skip connections achieves the highest performance on WHDLD and Potsdam. First, the highest MIoU and ACC on WHDLD are 61.93% and 88.62%, respectively, which are 1.08% and 3.52% higher than the model without

TABLE IV
ABLATION EXPERIMENT OF ASPP ON WHDLD VALIDATION SET

Network	Modules	IoU%(WHDLD)						Evaluation metrics			
	ASPP	Build	Road	Pavement	Vegetation	Bare Soil	Water	WHDLD		Potsdam	
								MIoU%	Acc%	MIoU%	Acc%
SegFormer+SegNext +FFM+FIM+Decoder	×	55.79	58.76	41.97	81.51	37.76	93.54	61.56	88.43	77.20	85.73
	✓	56.00	59.76	42.64	82.03	37.28	93.90	61.93	88.62	77.67	85.92

any skip connections. And on Potsdam, our model achieves the highest MIoU and ACC of 77.67% and 85.92%, respectively. Similarly, this result is 1.35% and 3.41% higher than the lowest scores in MIoU and ACC, respectively. As shown in Table II, for WHDLD, we find that between not using any skip connection and using (s4+f1), the model is not significantly improved. The addition of (s3+f2) improves the MIoU and ACC of the model the most, by 0.77% and 3.21%, respectively. After adding (s2+f3), the model behaves better than adding (s4+f1), but worse than adding (s3+f2), meanwhile, MIoU and ACC are increased by 0.43% and 1.67%, respectively, compared with adding (s4+f1). For Potsdam, we can clearly observe that the experimental results have a similar trend to the results on WHDLD. (s3+f2) has strong support for model performance; it is 0.96% and 3.11% higher than no skip in MIoU and ACC. (s4+f1) and (s2+f3) contribute slightly more to the model than no skip. According to the experimental results, there are significant differences in the information transmitted by different connections. Therefore, the connection scheme should be optimized to effectively improve the performance of the model.

E. Effect of ASPP

The ASPP block effectively captures multiscale features that are crucial for semantic segmentation results by using dilation convolutions with various dilation rates. To investigate the impact of ASPP in our model, we compared the semantic segmentation results when the ASPP block was added and when it was not included.

As shown in Table IV, it can be seen that the metric increases when including ASPP blocks in our network. On WHDLD and Potsdam, MIoU and ACC increase by 0.37% and 0.19%, 0.47% and 0.19%, respectively. This demonstrates the positive effect of the ASPP block on improving the segmentation performance of our model. In terms of specific classes, ASPP is more significant for road and pavement, with IoU increasing 1% and 0.67%, respectively. However, for classes like build and water, the improvement was not as prominent. In general, the overall impact of the ASPP block on the entire network was positive, so ASPP is incorporated into our network.

F. Comparison With Other Methods

To demonstrate the effectiveness of the proposed network SSNet, we compared it with a bunch of present methods, including FCN [6], DeeplabV3 [29], UNet [25], DANet [42], TransUnet [20], all of which we used Resnet101 [55] for the backbone of the network except TransUnet and Unet. All models are based on CNN except TransUnet, which is a hybrid network of Unet and ViT.

To be fair, we trained these models, using the same hyperparameters and devices. It is important to note that we did not use any pretrained weights for these models.

1) Results on WHDLD Dataset: Table V presents the quantitative results on the WHDLD dataset, which further validates the effectiveness of our proposed model. From the quantitative point of view, our model reaches 61.93% in MIoU and 88.62% in ACC; it is evident that the proposed model is superior to other methods in both MIoU and ACC and outperforms other models in terms of segmentation results for each category, except for the category of bare soil.

Fig. 11 illustrates the segmentation results of all models from a visual perspective. Thanks to the addition of FFM and FIM, our network not only improves the segmentation performance of strip-like objects, but also better preserves the detailed contours of some complex and irregular objects.

2) Results on Potsdam Dataset: We conducted experiments on the Potsdam dataset to further evaluate the effectiveness of our method. Table VI presents the quantitative results on the Potsdam dataset. Our model achieves MIoU of 77.67% and ACC of 85.92%, which significantly outperforms previous methods. Compared to other models, our network achieves the highest enhancement of 2.93% and 1.62% in MIoU and ACC, respectively, and it demonstrates an increase in the segmentation performance of each category.

To visually showcase the segmentation performance of our model, we present the segmentation results in Fig. 12. In the first and third rows, our model is more likely to identify the impervious surface. The second and fourth rows are similar and mainly show the results of our model's partitioning of the tree, and the fifth row mainly shows our model's segmentation results for the background or clutter.

3) Results for FLOPs, Params, and MIoU on WHDLD Dataset: When evaluating a deep learning model, two crucial factors to consider are the number of floating-point operations (FLOPs) and the number of learnable parameters in the model. FLOPs offer an estimate of the model's computational complexity, while params represent the number of parameters that require learning during model training, reflecting the model's space complexity.

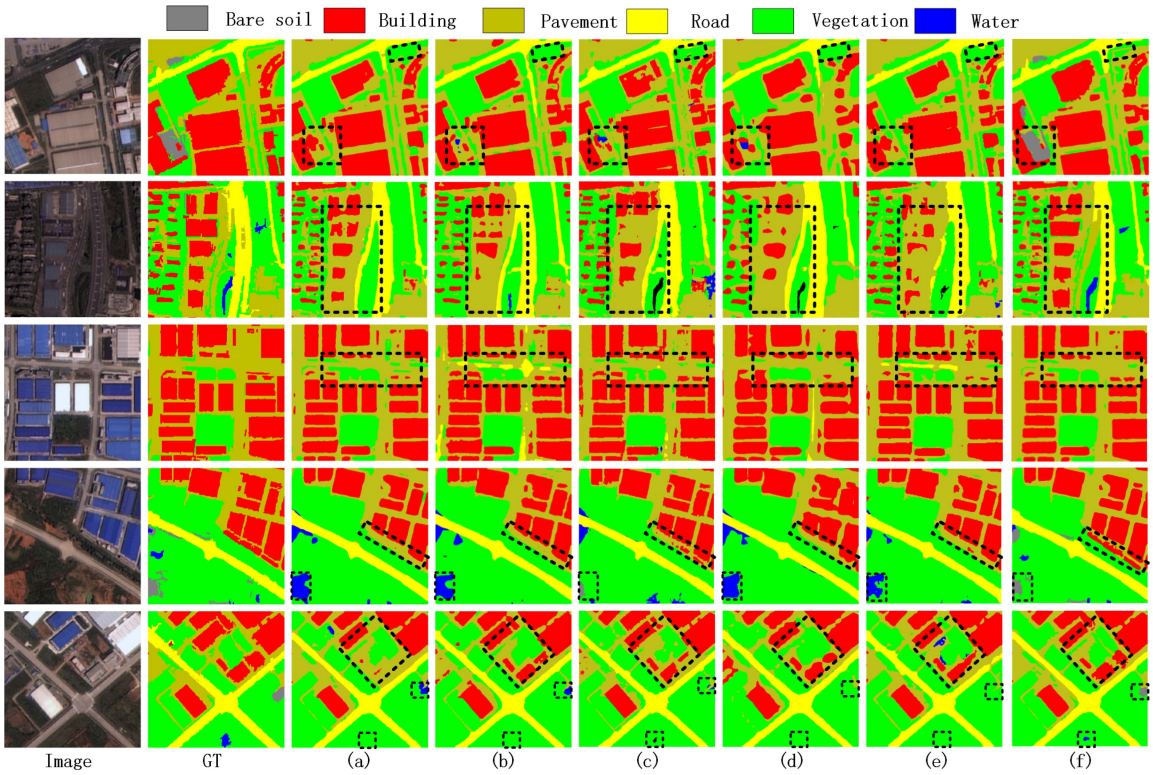

Fig. 11. Comparison between the proposed network and other methods on WHDLD dataset. Examples are selected randomly from the validation set of the WHDLD dataset. (a) FCN. (b) DeepLabV3. (c) UNet. (d) DANet. (e) TransUnet. (f) Our network.

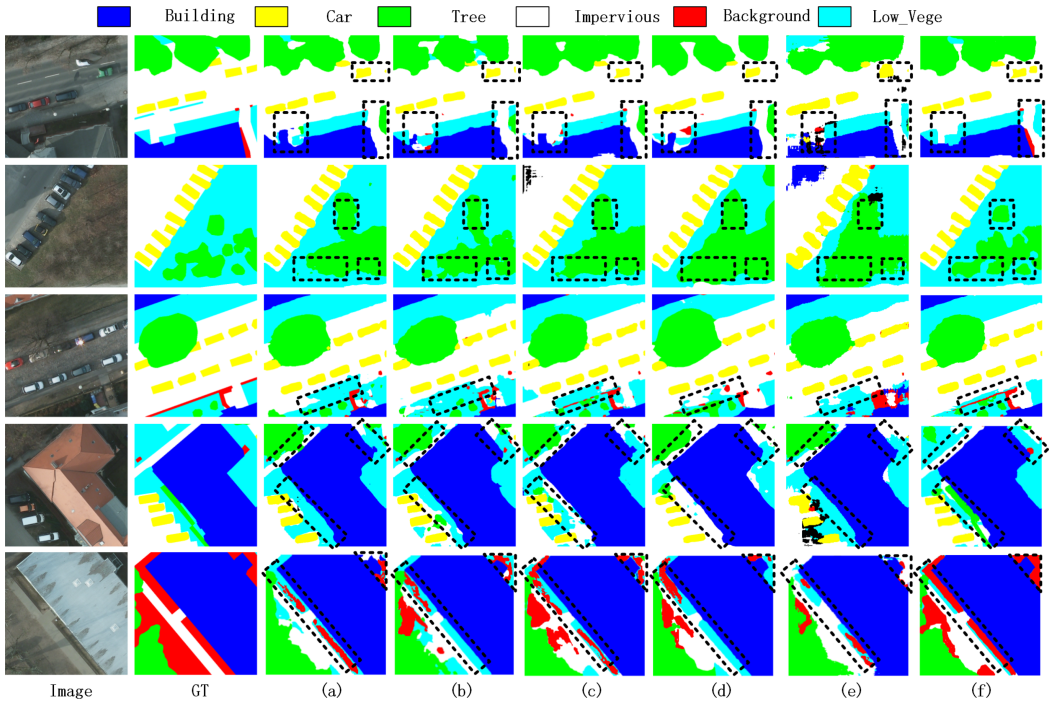

Fig. 12. Comparison between the proposed network and other methods on Potsdam dataset. Examples are selected randomly from the validation set of the Potsdam dataset. (a) FCN. (b) DeepLabV3. (c) UNet. (d) DANet. (e) TransUnet. (f) Our network.

TABLE V
RESULTS ON WHDLD VALIDATION SET

Method	IoU%						Evaluation metrics	
	Build	Road	Pavement	Vegetation	Bare Soil	Water	MIoU%	Acc%
FCN [6]	53.52	56.15	39.27	80.10	**38.37**	92.66	60.01	87.09
DeepLabV3 [29]	54.29	56.20	39.04	80.11	37.52	92.67	59.97	87.12
UNet [25]	54.56	56.01	39.98	80.60	38.30	93.28	60.46	87.39
DANet [42]	48.44	54.52	36.30	77.97	36.61	91.11	57.49	85.70
TransUnet [20]	51.62	52.39	37.64	79.61	37.69	92.56	58.58	86.56
Our method	**56.00**	**59.76**	**42.64**	**82.03**	37.28	**93.90**	**61.93**	**88.62**

Bold values in the table are the highest values among data in the column. The underlined values mean the second highest value among data in the column.

TABLE VI
RESULTS ON THE POTSDAM VALIDATION SET

Method	IoU%					Evaluation metrics	
	Impervious surface	Building	Low vegetation	Tree	Car	MIoU%	Acc%
FCN [6]	78.92	85.71	68.27	70.41	76.29	75.92	85.33
DeepLabV3 [29]	80.10	87.53	69.17	71.76	76.34	76.68	85.91
UNet [25]	79.33	87.01	69.54	72.56	77.02	77.09	85.90
DANet [42]	79.23	86.57	68.42	70.24	74.22	75.74	85.42
TransUnet [20]	77.03	82.41	68.32	70.46	75.52	74.74	84.27
Our method	**80.35**	**88.09**	**69.58**	**72.81**	**77.54**	**77.67**	**85.92**

Bold values in the table are the highest values among data in the column. The underlined values mean the second highest value among data in the column.

TABLE VII
COMPARISON RESULTS OF FLOPS AND PARAMS AND MIOU ON WHDLD VALIDATION SET

Method	FLOPs(G)	Params(M)	MIoU%
FCN	54.22	51.94	60.01
DeepLabV3	60.53	58.62	59.97
Unet	40.21	**17.26**	60.46
DANet	19.18	66.42	57.49
TransUnet	25.00	66.81	58.58
SSNet	**10.73**	54.00	**61.93**

Bold values in the table are the highest values among data in the column. The underlined values mean the second highest value among data in the column.

Table VII shows data of the various models on WHDLD. First and foremost, it is clear that SSNet has the best segmentation performance of 61.93% and the lowest model complexity, although the number of parameters is in the middle of the pack. For FLOPs, SSNet only needs 10.73G FLOPs, which is about two to four times less than UNet, DANet, and TransUnet, and more than five times less than DeepLabV3 and FCN. Compared to standard models like FCN, the SSNet model delivers state-of-the-art efficiency while maintaining high accuracy and strong performance for semantic segmentation.

Since transformer performs semantic computation based on self-attention, SSNet does not have the least number of parameters, but it strikes an optimal balance between parameters number and efficiency as well as property. SSNet achieves this level of performance with up to about six times fewer FLOPs than other models with similar parameter counts. For instance, DeepLabV3 has 58.62 M parameters but requires 60.53G FLOPs compared to SSNet's 10.73 G FLOPs, while SSNet's MIoU is 2% higher than DeepLabV3. And with 54 M parameters, SSNet retains sufficient capacity for handling complex segmentation tasks, unlike extremely lightweight models like Unet (17.26 M parameters). Benefiting from the advantages of the CNN-transformer hybrid architecture, SSNet demonstrates state-of-the-art MIoU performance while maintaining relatively lower FLOPs and Params, showcasing its feasibility and potential in RS applications.

Finally, for accuracy as measured by MIoU, SSNet achieves 61.93% MIoU. This edges out prior top models by up to 2%, representing a substantial accuracy gain. Fig. 13 visualizes the results of different models in the three performance dimensions-efficiency, computation cost, and precision.

In a word, the model effectively balances tradeoffs between efficiency, size, and accuracy for superior overall performance compared to prior methods.

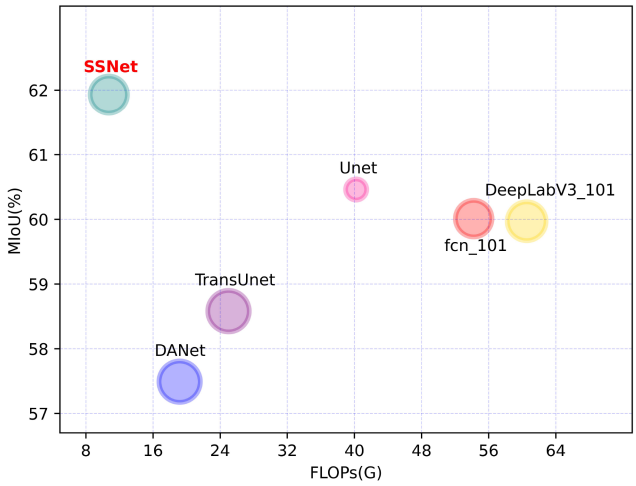

Fig. 13. Performance versus model efficiency of different models. The vertical axis represents the MIoU. The horizontal axis indicates the FLOPs. The diameter of the circle indicates the number of model parameter.

VI. Conclusion

In this research, we propose an innovative network for semantic segmentation of RS images. Our focus is on effectively integrating the benefits of local and global information to enhance the feature discrimination of ground objects. The proposed model follows the classical encoder–decoder mode, with the encoder combining CNN and transformer. It consists of four stages, producing feature maps with different resolutions, while the decoder gradually restores feature maps to the original map resolution size and predicts the semantic segmentation results. Between the decoder and the encoder, we use skip connections to keep the shallow and deep features fused with each other, enhancing the communication of multiscale features. Moreover, we proposed FFM to improve the quality of segmentation, which takes full advantages of local features and global information. FIM managed to help extract strip-like features as much as possible and learn the correlation between channels. Although our proposed model achieves encouraging performance on the Potsdam and WHDLD datasets, we remain to be concerned with the parameters number and speed, and it is unclear whether it can work well in small mobile devices. Also, we do not have a separate design for boundary detection, which we will verify in our future work.

References

[1] M.-H. Guo, C.-Z. Lu, Q. Hou, Z. Liu, M.-M. Cheng, and S.-M. Hu, "SegNeXt: Rethinking convolutional attention design for semantic segmentation," *Adv. Neural Inf. Process. Syst.*, vol. 35, pp. 1140–1156, 2022.

[2] C. Zhang, W. Jiang, Y. Zhang, W. Wang, Q. Zhao, and C. Wang, "Transformer and CNN hybrid deep neural network for semantic segmentation of very-high-resolution remote sensing imagery," *IEEE Trans. Geosci. Remote Sens.*, vol. 60, pp. 1–20, 2022.

[3] Y. Hua, D. Marcos, L. Mou, X. X. Zhu, and D. Tuia, "Semantic segmentation of remote sensing images with sparse annotations," *IEEE Geosci. Remote Sens. Lett.*, vol. 19, pp. 1–5, 2022.

[4] J. Li et al., "A 2.5 D semantic segmentation of the pancreas using attention guided dual context embedded U-Net," *Neurocomputing*, vol. 480, pp. 14–26, 2022.

[5] R. O. Dogan, H. Dogan, C. Bayrak, and T. Kayikcioglu, "A two-phase approach using mask R-CNN and 3D U-Net for high-accuracy automatic segmentation of pancreas in CT imaging," *Comput. Methods Programs Biomed.*, vol. 207, 2021, Art. no. 106141.

[6] S. Ji, S. Wei, and M. Lu, "Fully convolutional networks for multisource building extraction from an open aerial and satellite imagery data set," *IEEE Trans. Geosci. Remote Sens.*, vol. 57, no. 1, pp. 574–586, Jan. 2019.

[7] B. Chen, M. Xia, and J. Huang, "MFANet: A multi-level feature aggregation network for semantic segmentation of land cover," *Remote Sens.*, vol. 13, no. 4, 2021, Art. no. 731.

[8] Z. Guo et al., "Semantic segmentation for urban planning maps based on U-Net," in *Proc. IEEE Int. Geosci. Remote Sens. Symp.*, 2018, pp. 6187–6190.

[9] L. Bragagnolo, R. V. d. Silva, and J. M. V. Grzybowski, "Amazon forest cover change mapping based on semantic segmentation by U-Nets," *Ecological Inform.*, vol. 62, 2021, Art. no. 101279.

[10] Y. Yu et al., "Crop row segmentation and detection in paddy fields based on treble-classification Otsu and double-dimensional clustering method," *Remote Sens.*, vol. 13, no. 5, 2021, Art. no. 901.

[11] Z. Zheng, Y. Zhong, J. Wang, and A. Ma, "Foreground-aware relation network for geospatial object segmentation in high spatial resolution remote sensing imagery," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 4095–4104.

[12] J. Wang, Y. Wang, Y. Wu, K. Zhang, and Q. Wang, "FRPNet: A feature-reflowing pyramid network for object detection of remote sensing images," *IEEE Geosci. Remote Sens. Lett.*, vol. 19, pp. 1–5, 2022.

[13] X. Li et al., "Pointflow: Flowing semantics through points for aerial image segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 4215–4224.

[14] A. Vaswani et al., "Attention is all you need," in *Proc. Annu. Conf. Neural Inf. Process. Syst.*, vol. 30, 2017, pp. 5998–6009.

[15] A. Dosovitskiy et al., "An image is worth 16x16 words: Transformers for image recognition at scale," 2020, *arXiv:2010.11929*.

[16] Z. Liu et al., "Swin transformer: Hierarchical vision transformer using shifted windows," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2021, pp. 9992–10002.

[17] L. Wang et al., "UNetFormer: A UNet-like transformer for efficient semantic segmentation of remote sensing urban scene imagery," *ISPRS J. Photogrammetry Remote Sens.*, vol. 190, pp. 196–214, 2022.

[18] E. Xie, W. Wang, Z. Yu, A. Anandkumar, J. M. Alvarez, and P. Luo, "Segformer: Simple and efficient design for semantic segmentation with transformers," in *Proc. Annu. Conf. Neural Inf. Process. Syst.*, 2021, pp. 12077–12090.

[19] W. Wang et al., "Pyramid vision transformer: A versatile backbone for dense prediction without convolutions," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2021, pp. 548–558.

[20] J. Chen et al., "TransUNet: Transformers make strong encoders for medical image segmentation," 2021, *arXiv:2102.04306*.

[21] J. Li, Y. Yan, S. Liao, X. Yang, and L. Shao, "Local-to-global self-attention in vision transformers," 2021, *arXiv:2107.04735*.

[22] Z. Peng et al., "Conformer: Local features coupling global representations for visual recognition," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2021, pp. 367–376.

[23] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2015, pp. 3431–3440.

[24] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Semantic image segmentation with deep convolutional nets and fully connected CRFs," 2014, *arXiv:1412.7062*.

[25] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional networks for biomedical image segmentation," in *Proc. Int. Conf. Med. Image Comput. Comput.- Assist. Intervention*, 2015, pp. 234–241.

[26] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, "Pyramid scene parsing network," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 6230–6239.

[27] X. Huang and S. Belongie, "Arbitrary style transfer in real-time with adaptive instance normalization," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 1510–1519.

[28] 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 Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 4, pp. 834–848, Apr. 2018.

[29] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam, "Rethinking atrous convolution for semantic image segmentation," 2017, *arXiv:1706.05587*.

[30] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, "Encoder-decoder with atrous separable convolution for semantic image segmentation," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 801–818.
[31] C. Yu, J. Wang, C. Gao, G. Yu, and N. Sang, "Context Prior for Scene Segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 12413–12422.
[32] Y. Yuan, X. Chen, and J. Wang, "Object-contextual representations for semantic segmentation," in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 173–190.
[33] H. Zhang et al., "Context encoding for semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 7151–7160.
[34] Y. Zhou, X. Sun, Z.-J. Zha, and W. Zeng, "Context-reinforced semantic segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 4041–4050.
[35] J. He, Z. Deng, L. Zhou, Y. Wang, and Y. Qiao, "Adaptive pyramid context network for semantic segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 7511–7520.
[36] T. Takikawa, D. Acuna, V. Jampani, and S. Fidler, "Gated-SCNN: Gated shape CNNs for semantic segmentation," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2019, pp. 5228–5237.
[37] M. Dickenson and L. Gueguen, "Rotated rectangles for symbolized building footprint extraction," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops*, 2018, pp. 215–2153.
[38] T.-S. Kuo, K.-S. Tseng, J.-W. Yan, Y.-C. Liu, and Y.-C. F. Wang, "Deep aggregation net for land cover classification," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops*, 2018, pp. 247–2474.
[39] S. Aich, W. v. d. Kamp, and I. Stavness, "Semantic binary segmentation using convolutional networks without decoders," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops*, 2018, pp. 182–1824.
[40] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 7132–7141.
[41] S. Woo, J. Park, J.-Y. Lee, and I. S. Kweon, "CBAM: Convolutional block attention module," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 3–19.
[42] J. Fu et al., "Dual attention network for scene segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 3141–3149.
[43] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, "End-to-end object detection with transformers," in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 213–229.
[44] X. Zhu, W. Su, L. Lu, B. Li, X. Wang, and J. Dai, "Deformable DETR: Deformable transformers for end-to-end object detection," 2020, *arXiv:2010.04159*.
[45] S. Zheng et al., "Rethinking semantic segmentation from a sequence-to-sequence perspective with transformers," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 6877–6886.
[46] X. Chu et al., "Conditional positional encodings for vision transformers," 2021, *arXiv:2102.10882*.
[47] M. Mao et al., "Dual-stream network for visual recognition," in *Proc. Annu. Conf. Neural Inf. Process. Syst.*, 2021, pp. 25346–25358.
[48] Y. Zhang, H. Liu, and Q. Hu, "Transfuse: Fusing transformers and CNNs for medical image segmentation," in *Proc. Int. Conf. Med. Image Comput. Comput.- Assist. Intervention*, 2021, pp. 14–24.
[49] B. Yu, H. Yin, and Z. Zhu, "ST-UNet: A spatio-temporal u-network for graph-structured time series modeling," 2019, *arXiv:1903.05631*.
[50] M.-H. Guo, C.-Z. Lu, Z.-N. Liu, M.-M. Cheng, and S.-M. Hu, "Visual attention network," *Comput. Vis. Media*, vol. 9, no. 4, pp. 733–752, 2023.
[51] C. Peng, X. Zhang, G. Yu, G. Luo, and J. Sun, "Large kernel matters–improve semantic segmentation by global convolutional network," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 1743–1751.
[52] Q. Hou, L. Zhang, M.-M. Cheng, and J. Feng, "Strip pooling: Rethinking spatial pooling for scene parsing," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 4002–4011.
[53] Z. Shao, K. Yang, and W. Zhou, "Performance evaluation of single-label and multi-label remote sensing image retrieval using a dense labeling dataset," *Remote Sens.*, vol. 10, no. 6, 2018, Art. no. 964.
[54] Z. Shao, W. Zhou, X. Deng, M. Zhang, and Q. Cheng, "Multilabel remote sensing image retrieval based on fully convolutional network," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 13, pp. 318–328, 2020.
[55] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 770–778.
[56] X. Chen et al., "Adaptive effective receptive field convolution for semantic segmentation of VHR remote sensing images," *IEEE Trans. Geosci. Remote Sens.*, vol. 59, no. 4, pp. 3532–3546, Apr. 2021.

Min Yao received the Ph.D. degree in information processing from the Tokyo Institute of Technology, Tokyo, Japan, in 2014.

She is currently a Lecturer with the College of Information Engineering, Shanghai Maritime University, Shanghai, China. Her research interests include remote sensing image processing, object detection and segmentation, and pattern recognition.

Yaozu Zhang received the B.S. degree in Internet of Things engineering from Shanghai DianJi University, Shanghai, China, in 2018. He is currently working toward the M.S. degree in computer science and technology with the College of Information Engineering, Shanghai Maritime University, Shanghai, China.

His research interests include deep learning, computer vision, semantic segmentation, and remote sensing image processing.

Guofeng Liu received B.S. degree in computer science from Jilin University, Changchun, China, in 2019, and the M.S. degree in software engineering from Shanghai Maritime University, Shanghai, China, in 2022.

He is currently a Software Engineer with Baidu, Beijing, China. His research interests include deep learning, object detection, and knowledge distillation.

Dongdong Pang received M.S. degree in surveying and mapping science and technology from the Chengdu University of Technology, Chengdu, China, in 2022.

He is currently a Teacher with the School of Resources and Environmental Engineering, Tianshui Normal University, Tianshui, China. His research interests include remote sensing image processing, deep learning, and geospatial analysis.