Pattern Analysis and Applications (2024) 27:2
https://doi.org/10.1007/s10044-023-01207-2

ORIGINAL PAPER

FBRNet: a feature fusion and border refinement network for real-time semantic segmentation

ShaoJun Qu[1] · Zhuo Wang[1] · Jie Wu[1] · YueWen Feng[1]

Received: 20 September 2023 / Accepted: 21 December 2023 / Published online: 24 January 2024

Abstract
Existing semantic segmentation networks perform well in accuracy by spending much computation. However, for practical applications, not only high segmentation accuracy but also high inference speed is required. To solve the problem of the difficult balance between accuracy and speed, we propose a new real-time semantic segmentation network (FBRNet). To extract multi-scale semantic information more quickly, we propose a lightly weighted reinforced atrous spatial pyramid pooling module (arASPP) based on the attention mechanism, which can extract richer and more advanced features with less computation than the original ASPP. To eliminate the semantic gap between high- and low-level features, we propose a new feature fusion module (CSFM), in which a shuffling mechanism is introduced to enhance robustness, and a parallel contextual information enhancement module and detail information enhancement module are built to facilitate the information exchange between high- and low-level features, achieving the effect of improving the model feature representation. Finally, we also introduce high-level features, fusing Laplace convolution and spatial attention mechanisms, and design the edge feature reinforcement module (LABRM) to eliminate the noise of low-level features and compensate for the model's segmentation effect target boundary. In the Cityscapes validation set and test set, FBRNet achieves 77.63% and 75.3% mIoU, and 101.9 FPS on a single tesla-T4 GPU, also achieved 72.4% mIoU and 89.8 FPS on the CamVid dataset and 55.2% mIoU and 100.8 FPS on the BDD100K dataset, which is a better balance of accuracy and speed compared with existing networks. The code is available at https://github.com/little5570/FBRNet.

Keywords Real-time semantic segmentation · Attention mechanism · Feature fusion · Boundary reinforcement

1 Introduction

Semantic segmentation is one of the classic tasks of computer vision, with a wide range of application scenarios, such as graphics processing, autonomous driving, and medical image analysis [1–3], and its task is to segment the image produced in a specific scene, that is, to classify each pixel in the image according to the pre-defined semantic category, to achieve the purpose of segmenting the image [4, 5]. Based on the traditional convolutional network, Long et al. [6] proposed a fully convolutional network (FCN), replacing the final fully connected layer with the convolution and using FCN for semantic segmentation for the first time, which greatly improved the effect of model segmentation of images, since then, image segmentation has entered the deep learning stage from the traditional stage. However, these models typically require long inference speeds and are not suitable for practical applications that require both good accuracy and high speed, such as unmanned driving and video surveillance. As a result, researchers have started to favor the design real-time semantic segmentation networks that can achieve a good balance between accuracy and speed simultaneously. To achieve this goal, they have designed some approaches as follows.

One is to design a lightweight computing module. MobileNet [7] proposed the concept of depth-separable

✉ ShaoJun Qu
qshj@hunnu.edu.cn

Zhuo Wang
1351893798@qq.com

Jie Wu
1749415531@qq.com

YueWen Feng
feng_yuewen@163.com

[1] College of Information Science and Engineering, Hunan Normal University, 36 Lushan Rd, Changsha 410081, Hunan, China

convolution, which divides a standard 3×3 convolution into a depthwise convolution and a pointwise convolution with only 1/3 of the original standard convolution in terms of the number of parameters, reducing the computational complexity of the model and reducing the inference time, but the computing power of this convolution is inferior to that of the original convolution, especially for small models, the performance of the model will be seriously lost due to the substantial decrease in the number of parameters.

Another one is to design a new network structure. Includes two-branch structure and multi-branch structure. The two-branch structure model, such as BiSeNetV1 [8] designs two different branches for extracting semantic features and detail information, respectively, and finally merges the two. However, the increased computational effort of a two-branch network undoubtedly slows down model inference speed compared to a general network. The multi-branch structure models include ICNet [9], DFANet [10], etc. To extract feature information at different scales of the input image, ICNet [9] downsamples the input image to three different sizes by cropping; the calculation and fusion of the three branches also increase the calculation amount of the network, which adds a lot of pressure to the inference speed.

In summary, the lightweight computation module can improve the speed of model inference, but the accuracy is not as good as the general module, and the added branching of the new network structure will slow down the inference speed. Many of the methods mentioned above cannot achieve a good balance between segmentation accuracy and speed. To solve such problems, we design a new real-time semantic segmentation network, which includes an attention refinement atrous spatial pyramid pooling module (arASPP) that can extract multi-scale features with less computational effort, a feature fusion module based on mutual guidance of high- and low-level features, and a lightweight boundary refinement module based on Laplace and spatial attention mechanism. Our contributions can be summarized as follows:

- The plug-and-play attention refinement spatial pyramid pooling module arASPP is proposed, and the channels of each branch are only 1/4 of the original ASPP [11], which greatly reduces the amount of computation; besides, an attention refinement module is added after each branch to exceedingly enhance the expression of the information of the network.
- A new feature fusion module CSFM based on the mutual guidance of high- and low-level features is proposed, using feature guidance to further extract semantic and detail information before fusion, and effectively fusing features of each layer under the premise of less computation.
- The boundary reinforcement module, LABRM, is proposed to enhance the representation of low-level features and then creatively utilizes parallel Laplace convolution and spatial attention mechanism to extract boundary information, resulting in a significantly smoother boundary segmentation effect.
- A new real-time semantic segmentation network FBRNet is designed by assembling arASPP, CSFM, and LABRM, and the balance of high accuracy and high speed is achieved on Cityscapes [12], CamVid [13], and BDD100K [14] datasets.

The rest of this paper is structured as follows: Section 2 reviews the relevant work. Section 3 describes the proposed method. Section 4 presents the experimental results and comparisons. Finally, we conclude in Sect. 5.

2 Related work

Based on the composition of FBRNet, we will discuss the related work of encoder–decoder, multi-scale feature extraction, feature fusion, and boundary reinforcement in this section.

2.1 Encoder–decoder

Although convolution can extract certain semantic features, it can also lose some information. To compensate for the problem of information loss caused by continuous convolution, the researchers designed the encoder–decoder structure. UNet [15] is one of the most typical encoder–decoder models, which downsampled the input image during the encoding and supplemented the information lost in the decoding. UNet [15] densely integrates high- and low-layer features, making it suitable for processing medical images. However, UNet [15] integrates features at all levels through cascading, which leads to a large amount of computation in the decoding and will reduce the inference speed.

To solve the problem of complex and challenging division of remote sensing image content, SU et al. [16] proposed an optimized version of UNet, using combinatorial dilated convolution as the encoder to increase the receptive field, which can help the model extract more advanced semantic features. But in the decoding stage, the network uses a transpose convolution with strides of 2 to restore the feature, which undoubtedly increases the number of parameters. XNet [17] cleverly integrates convolution neural network (CNN) and transformer into the encoder–decoder structure, using convolution and transformer simultaneously in the encoding part to extract both local and global features, achieving excellent results in the field of medical image segmentation.

Due to the ability of the encoder–decoder to fuse features captured at each level during the downsampling, this paper adopts an encoder–decoder. In the decoding stage, it is crucial to fill in the information lost during downsampling, as this is essential for the classification of pixels in subsequent prediction. Therefore, it is necessary to design a reasonable decoder. Methods like UNet [15] use a cascaded approach, which leads to a large computational burden and hinders feature fusion. The optimized version of UNet [16], which uses transpose convolution, undoubtedly increases trainable parameters. To avoid information loss while ensuring low computational complexity, FBRNet uses a guided fusion approach between high- and low-level features, employing regular convolutions in the computation. This not only achieves high segmentation accuracy but also improves speed, making it more suitable for real-time semantic segmentation.

2.2 Multi-scale feature extraction

Semantic segmentation aims to classify every pixel of the input image. Therefore, the model not only needs to extract large-scale target information but also needs to obtain target features of other scales. PPM [18] uses four different scales of pooling to obtain multi-scale features, which can play a good role in image classification tasks, and pooling does not contain trainable parameters, so the addition of PPM will not slow down the inference speed [19]. However, in complex scenarios, global average pooling is not enough to cover key information, which is not conducive to improving accuracy. To handle a large number of medical images with limited computational resources, He et al. [20] proposed a medical image segmentation method that combines transformer with CNN, which extracts both global and local features of the image from multiple perspectives, achieving the goal of completing image segmentation in a short period of time.

To obtain the target features under different receptive fields, Chen et al. [11] designed ASPP in deeplabV2, which uses four parallel branches, and the four branches, respectively, use the scales of [6, 12, 18, 24] for dilated convolution to obtain feature, which effectively improves the receptive fields of the network and enhance the network's ability to recognize objects at different scales. However, the input of these four parallel branches is the highest layer feature of the network, with a large number of channels, which will slow down the inference speed of the model.

The PPM only uses simple average pooling to extract multi-scale features, resulting in fewer trainable parameters and limited optimization space for the network. In this paper, the arASPP module is introduced, which utilizes dilated convolution to extract features, aiming to enhance the feature extraction capability of FBRNet through multiple training iterations. Similarly, ASPP also employs dilated convolution to extract multi-scale features, but its branches have excessively large dilation rates, leading to a higher parameter count and slower inference speed, so the convolution of the arASPP module with a smaller dilation rate, which does not slow down the inference speed of the model, and we also add the attention reinforcement module after each parallel dilated convolution branch, so that the extracted features are further strengthened.

2.3 Feature fusion

The deep neural network increases the number of channels and reduces the size of the feature map by continuous downsampling, and obtains the high-level feature map. However, the top-level feature map is not enough to cover all the information of the input image. It is necessary to fuse the feature map of each layer to obtain rich contextual information and details [21, 22]. Commonly used feature fusion methods include element-wise addition and cascading.

Using element-wise addition, different layer features can be fused by simple addition, so that the size of the feature map participating in the calculation will not change, and the inference speed of the model will not increase. But there is a semantic gap between high and low features. Fusing them by simple addition will not only lead to mutual interference between the two features but also will lose the previously extracted information.

Using the cascade method to fuse the features is equivalent to connecting all levels of features from the dimension of channels. This allows the feature information to be processed from a higher dimension and increases the optimization space of the model. However, it also has some disadvantages, the decoder treats all channels and positions evenly and cannot pay more attention to important channels or locations, which weakens the context and spatial expression ability of the model to a certain extent.

In the field of super-resolution image segmentation, MFFN [23] obtains a set of high-level features and a set of low-level features by designing low-level feature mapping and high-level feature mapping, respectively. Then in the feature reconstruction module, the high- and low-level features of each level are corresponded and fused with each other by convolution and sub-pixel fusion. It not only effectively reduces the computational amount of the network, but also ensures the effective fusion of features at all levels and improves the model performance. The element-wise addition fuses features too simply, and cascade will make the network treat each channel and position evenly so that the model has no focus on the fused features. The CSFM we designed is more complex than element-wise addition, and ContextR and DetailR are added to the CSFM so that the model can focus on the context features and details from different perspectives, with a clear focus.

2.4 Boundary reinforcement

The low-level features contain rich information on small targets and key details, but also much noise because the downsampling depth of the low-level feature is not enough, and the receptive field is limited. Suppose the low-level features are directly fused with the high-level features. In that case, the noise will interfere with the model classification, leading to the misclassification and the loss of necessary details [24, 25]. Therefore, it has also become one of the difficulties in semantic segmentation to perform some enhancement of the low-level features to remove the noise and optimize the target boundary segmentation effect.

To prevent the loss of low-level information, the JPANet [26] uses parallel dilated convolution and maximum pooling to extract low-level features, and then directly fuse the low-level features with the high-level features. However, the low-level features are only downsampled by 1/2, which contain complex information that is not conducive to the guidance of high-level features. To get clear boundary information, FPANet [27] uses the invert function to highlight the segmentation of the target edge information and strengthen the detailed representation of the image. However, the invert function also increases the calculation to a certain extent, making the inference slower. Zhu et al. [28] designed an edge spatial attention block (ESAB) module to address the issue of insufficient consideration of edge information in image segmentation. In this module, a Sobel operator is introduced for edge detection, and a mechanism similar to attention is connected after the Sobel operator to improve boundary extraction and enhance the performance of the model in tumor image segmentation. DAM [29] has designed the network with a dual attention mechanism to solve the problem of removing rain marks and raindrops from the image simultaneously. Heavy rain features are obtained using the first attention mechanism, and then light rain features are obtained using inverse values. This combines the two as global features and then adjusts the attention by subsequent convolution to obtain a clear rain removal image. This shows that the attention mechanism is advantageous in small object feature extraction and is good for edge enhancement. To solve the problem of motion image blurring, Zhang et al. [30] designed a GramNet, which uses the GramNet matrix to refine the subtle movements between consecutive frames and subsequently uses HeptaGAN to supplement the exposure time and restore the continuity to achieve the effect of enriching the details of the motion image. This detail-aware network is effective for detail segmentation of images.

The low-level features involved in feature fusion in JPANet are too low and contain considerable noise. To avoid excessive noise in the low-level features participating in boundary reinforcement, we use feature maps downsampled to 1/4 the size of the original image as input to our boundary reinforcement module. Similar to the concept of BRM in FPANet, we also designed LABRM to highlight the segmentation target boundary information, but instead of using cumbersome confidence vectors and invert function, we just use Laplace convolution with fixed parameters, which has a smaller number of module parameters and is more conducive to the improvement of inference speed.

3 Method

This section first introduces the individual modules in FBRNet and then describes the network structure of FBRNet.

3.1 ArASPP module

The ASPP used in the Deeplab series uses convolution with a large dilation rate, resulting in increased computational complexity. Therefore, we use a smaller convolution dilation rate, and since using composite numbers as dilation rates may cause duplicate sample point positions and introduce redundant information, as shown in Fig. 1. Our dilation rate of arASPP is set to continuous prime numbers such as [3, 5, 7]. To obtain the global information at this stage, in addition to four convolutions, we add a global average pooling as a component of parallel branches. And to ensure that the number of parameters of the module does not increase excessively, the channels of the feature after dilated convolution are only 1/4 of the original feature map. If only one convolution is used per branch, the extracted features will still be too abstract and will not allocate attention appropriately, resulting in a lot of wasted available information, so we design an attention reinforcement module (ARM), as shown in Fig. 1. To obtain the global feature of each channel, we use global averaging pooling for the input features, and the global feature vector of $c\times1\times1$ is obtained, to keep the module pay more attention on important channels while suppressing redundant channels, a convolution of 1×1 is used for this feature vector, due to the trainability of the convolution parameters, after iterative training, the parameters of the convolution are gradually optimized, so that attention will be allocated appropriately, the obtained $c\times1\times1$ attention vector represents the weights of each channel after strengthening, and finally, the weights of each channel $c\times1\times1$ after strengthening are multiplied by element-wise with the input so that the original feature map can be adjusted adaptively according to the weight of each channel. After ARM, the enhanced parallel output is fused by cascading to obtain a feature map with the number of channels ($c\times5/4$). To facilitate subsequent fusion with the input features, the number of channels is changed from ($c\times5/4$) to c using

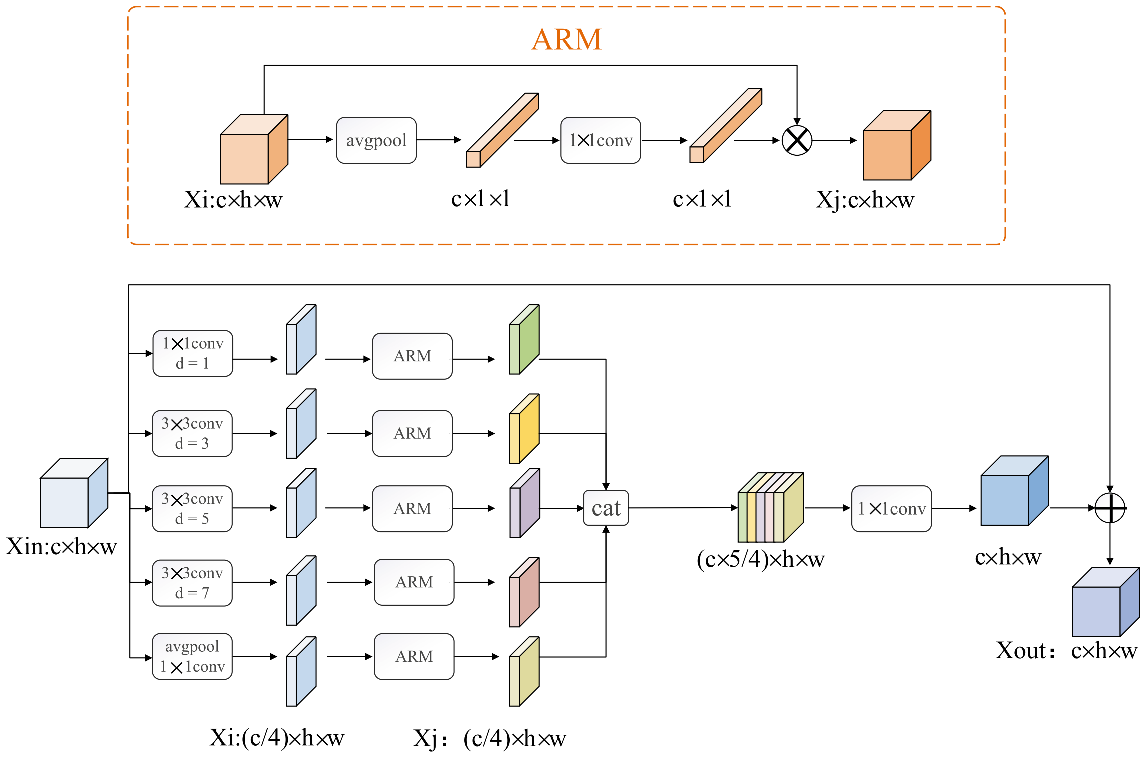


Fig. 1 Araspp module

1×1 convolution, and finally, the element-wise addition method is used to fuse the input features and the convoluted features to make up for the lost feature information after parallel convolution.

In Fig. 1, c×h×w represents the size of the feature map, and c, h, and w represent the number of channels, height, and width of the feature map, respectively. Avgpool represents global average pooling, 1×1 conv represents the convolution of kernel_size=1, and × represents element-level multiplication; 3×3 conv (d=3) represents the dilation=3 and kernel_size=3 dilated convolution, and cat represents cascade. The specific operation process of arASPP can be formulated as follows:

$$Xi = opi(Xin)) \tag{1}$$

$$Xj = conv(avg(Xi)) \times Xi \tag{2}$$

$$Xout = Xin + conv(cat(Xj)) \tag{3}$$

In Eq. (1), X*in* denotes the input of the arASPP module, *opi* represents the five parallel operations in arASPP. Equation (2) represents the operations performed in ARM, X*i* represents the input of ARM, *conv* represents the 1×1 convolution, *avg* represents the global average pooling along the channel, "×" represents element-wise multiplication, and X*j* represents the output of ARM. In Eq. (3), *cat* denotes the cascade and X*out* is the output of the entire arASPP module.

3.2 CSFM module

As mentioned above, there is a semantic gap between high- and low-level features. If only simple addition or cascading are used to fuse features, the certain error information will be introduced, resulting in mutual interference between features, and the model segmentation effect is not ideal. To solve such problems, we propose a context- and detail-based feature fusion module CSFM, as shown in Fig. 2. To make the size of the high-level feature consistent with the low-level feature, we first use upsampling for the high-level feature to expand its size; However, the number of channels of the high-level feature is twice that of the low-level feature, so we use 3×3 convolution to halve the number of channels of the high-level feature (the general model uses 1×1 convolution to change the channels of high-level feature, but to ensure that the receptive field of the high-level feature map is large enough, we use 3×3 convolution to further extract the information contained in the high-level features). To increase the space for optimization of low-level features, and cooperate with high-level features after upsampling, we also use 3×3 convolution for low-level features, so that through

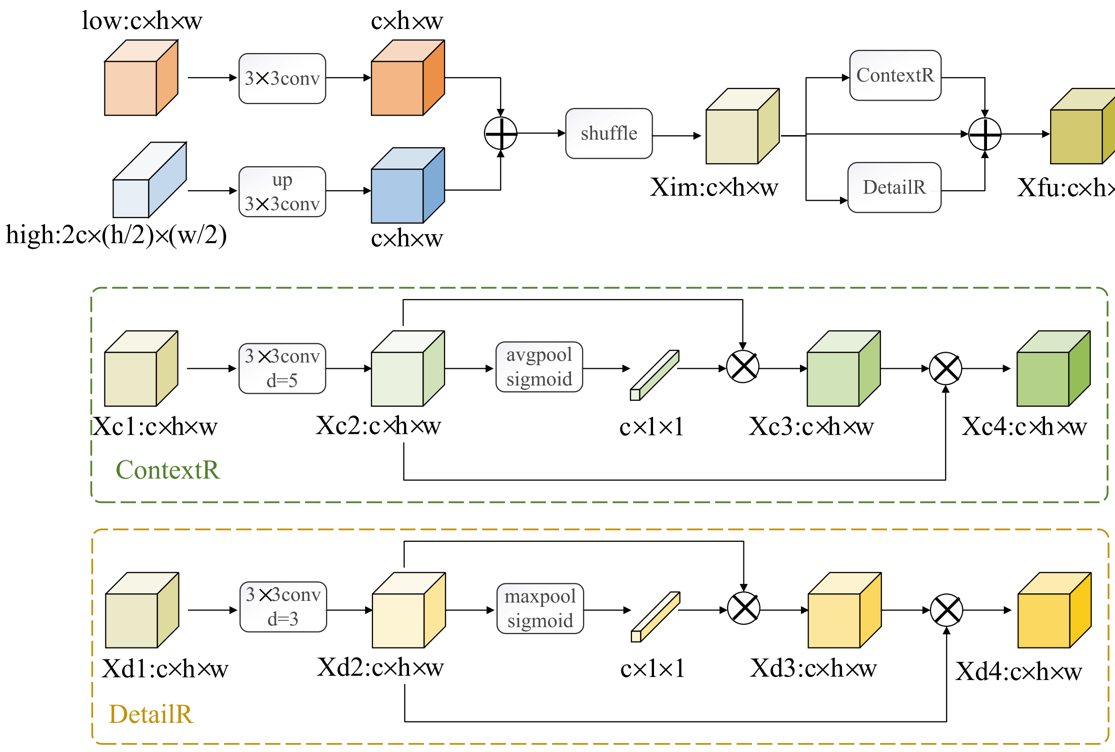


Fig. 2 CSFM module

multiple training optimizations, the convolution can filter out the noise in the low-level features and obtain the necessary information. To ensure that the channels after fusion are not much and the parameters are reduced for subsequent calculations, we use the fusion of element-wise addition for the high and low features. To prevent overfitting and strengthen the communication between channels, we performed a channel shuffle on the fused features after element-wise addition. After channel shuffle, to obtain further contextual and detailed information from different perspectives, the feature map passes through the ContextR and the DetailR in parallel (as information may be lost due to gradient disappearance if a cascade approach is used).

As shown in Fig. 2, ContextR is used to strengthen the context representation of the features after fusion; first, we use 3×3 convolution with a dilation rate of 5 for the feature map, which has a larger receptive field than the general convolution and can extract more contextual information, and then use global average pooling for the convoluted feature map to obtain a c×1×1 feature vector, which represents the weight occupied by each channel in all channels, which is beneficial to the redistribution of model attention. This feature vector is then multiplied element-wise with the convolved feature map to highlight key channels and suppress unimportant channels, and finally, the attention-enhanced feature map is multiplied element-wise with the input feature map to prevent the loss of original feature information.

DetailR has the same structure as ContextR, but the difference lies in the pooling method and the dilation rate used in the convolution. ContextR uses global average pooling, while DetailR uses global maximum pooling. Global average pooling is obtained as the mean value of each feature map, which is beneficial to the extraction of contextual information, while global maximum pooling is obtained as the maximum value of each feature map, which is beneficial to extract discriminative information from the feature map and facilitates the model's segmentation of the target boundary. The dilation rate of the first 3×3 convolution of ContextR is 5, larger dilated convolution means larger receptive fields, which is more conducive to obtaining global features, and the dilation rate of the first 3×3 convolution of DetailR is 3, which means smaller receptive fields, which can capture more local features and facilitate the extraction of detailed information.

After obtaining the global contextual information and boundary details, we add the features before entering the two modules, making these three features fused in the form of elemental-wise addition, which can fully exploit each other's strengths and compensate for each other. The high- and low-level feature maps entered into CSFM also play to the best advantage, preventing the two from interfering with each other.

In Fig. 2, low represents the low-level feature map, high represents the high-level feature map, up represents upsampling, 3×3 conv represents the ordinary convolution of kernel_size=3, " +" represents element-wise addition, shuffle represents channel shuffle, "×" represents element-wise multiplication, avgpool represents global average pooling along the channel, and maxpool represents global maximum pooling along the channel. The specific operation process of CSFM can be formulated as follows:

$$Xim = shuffle(conv3(\text{low}) + conv3(up(\text{high}))) \tag{4}$$

$$Xfu = ContextR(Xim) + DetailR(Xim) + Xim \tag{5}$$

$$Xc2 = convd(Xc1) \tag{6}$$

$$Xc3 = Xc2 \times sig(avg(Xc2)) \tag{7}$$

$$Xc4 = Xc3 \times Xc2 \tag{8}$$

Equation (4) represents the calculation process of X*im*, low denotes input low-level features, high denotes input high-level features, *up* denotes upsampling, and *conv3* denotes a convolution of size 3×3. In Eq (5), *ContextR* and *DetailR* denote the two submodules, respectively, and X*fu* represents the output of CSFM. Equations (6) – (8) denote the operations performed in ContextR, *convd* denotes dilated convolution, *avg* denotes global average pooling, and *sig* denotes sigmoid linear activation; as DetailR runs similarly to ContextR, no other equations are used to denote DetailR.

3.3 LABRM module

When the extracted semantic information is rich enough, it is beneficial to segment large goals. However, if the boundary detail information is missing, it is not friendly to the classification of target boundary pixels, resulting in blurred boundaries and low segmentation performance. The primary feature map has a small respective field, and most of the extracted features are small target object features, which also contain unavoidable noise. If the primary features are directly fused with the high-level features, the noise will be introduced, resulting in classification errors. Based on this, we propose LABRM based on Laplace and an attention mechanism for strengthening primary features. A single reinforcement of primary features alone is insufficient, as shown in Fig. 3, where we guide the primary features through the high-level features to ensure smoothness and continuity of the segmentation boundary. Since the high-level features are not the same size as the low-level features, the high-level features need to be recovered to the size of the low-level feature map by upsampling before guiding the low-level feature map learning. After fusing the high- and low-level feature maps by cascading, a feature map Xcat of size c2×h×w is obtained, and an auxiliary loss is used on Xcat for optimization. After the auxiliary loss calculation, the number of channels is changed using 1×1 convolution to the number of categories in the final segmentation of the model, for example, Cityscapes contains 19 categories, the returned feature map Xc is 19×h×w in size, and then the max operation is performed on Xc along the channels to get the maximum value to obtain a feature map Xmax of

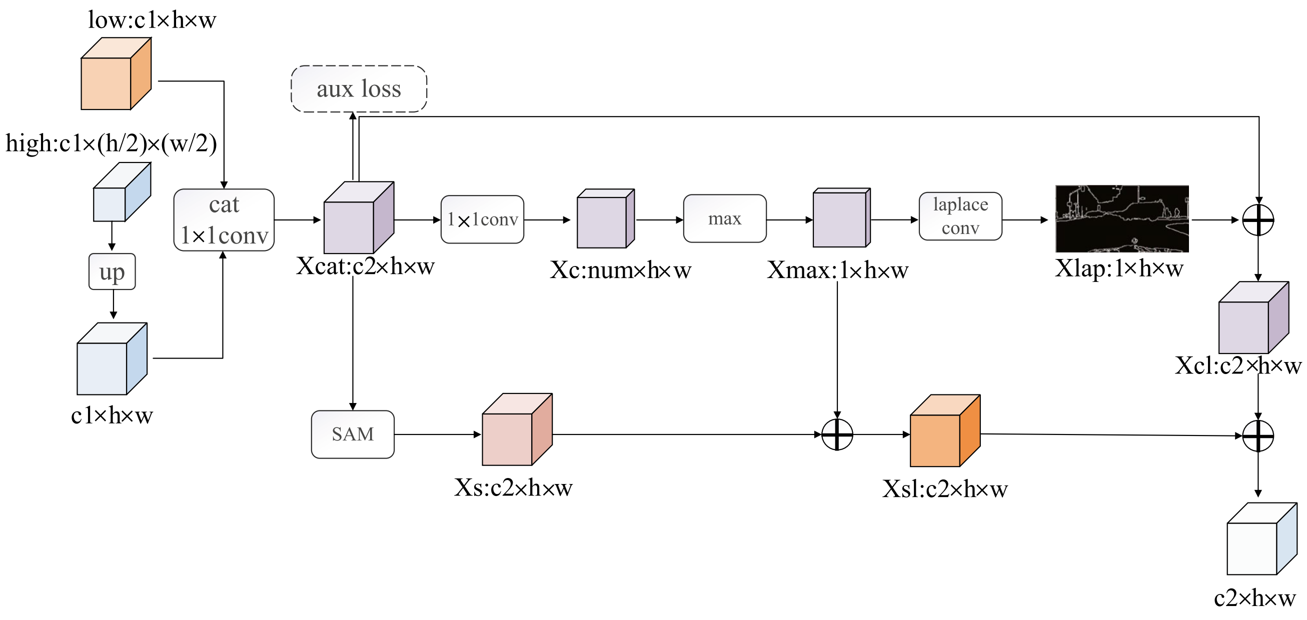


Fig. 3 LABRM module

1×h×w. The pixel value at each position of Xmax indicates the category to which the pixel belongs.

In Fig. 3, c1 represents the number of input feature map channels, c2 represents the number of output feature map channels, and num represents the number of classes finally predicted.

Since Xmax is obtained by taking the maximum value of Xc along the channel, the higher the probability that the point with the larger value in Xmax belongs to a certain semantic class, conversely, the lower the probability that the point with the smaller value in Xmax belongs to a certain class, that is, the point with the smaller value belongs to the target edge has a higher probability. To improve the spatial expression of the target edges, LABRM uses Laplace convolution on Xmax, which is a classical edge detection operator, as shown in Fig. 4. Compared with the original image, the edges of the feature map obtained using Laplace convolution are obvious. The feature map Xlap after Laplace convolution contains all the feature map edge information so that Xcat and Xlap are added element-wise to obtain the edge-enhanced feature map Xcl.

Transformer [31] has been designed for natural language processing to reduce the amount of model computation, omit recursion and convolution, and design a network for natural language processing based on a complete attention mechanism. Since then, the attention mechanism has been widely used in the field of deep learning due to its smaller computation and efficient performance. Spatial attention mechanism, as a kind of attention mechanism, adaptively adjusts the weight of each position by pooling. To guide the low-level features to strengthen the edge representation, we use attention mechanism as another parallel branch after Xcat in addition to Laplace convolution. The high-level feature map Xcat has an extensive feature map size. To pay more attention to important positions while suppressing redundant information, We use the spatial attention module (SAM) for Xcat. The structure of SAM is shown in Fig. 5, which performs global maximum pooling and global average pooling on the feature map along the spatial channel to obtain global information on the feature map, then the pooled feature map is convoluted and linearly activated to obtain a 1×h×w feature map, which represents the weights of each position of the input feature map, then the 1×h×w feature map and the input feature map are multiplied by element-wise to obtain the feature map Xs, that is, the weight effect is carried out on each position of the input feature map to strengthen important positions and suppress redundant positions. The enhanced feature map Xs is subjected to element-wise addition with Xmax, which corresponds to fusing the pixels at each location of Xs with the probability value of the category to which that location most likely belongs to obtain Xsl.

Finally, Xsl is fused with Xcl as the output of the whole LABRM module. LABRM gets both the advantages of the edge-enhanced feature map Xcl and contains the spatial information of the enhanced spatial location representation of the feature map Xsl, taking full advantage of the

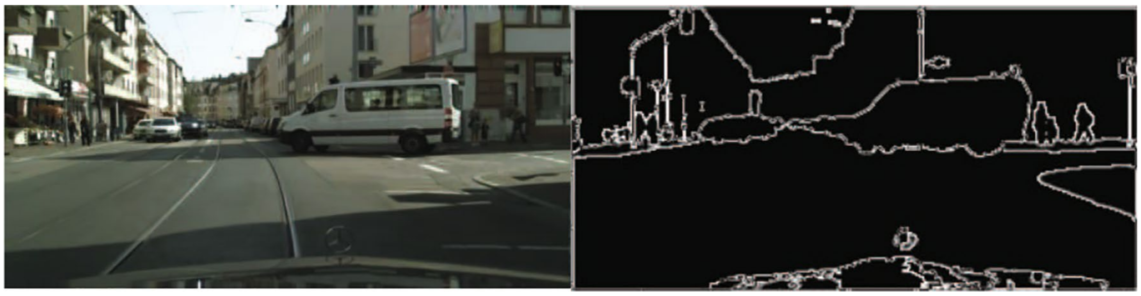
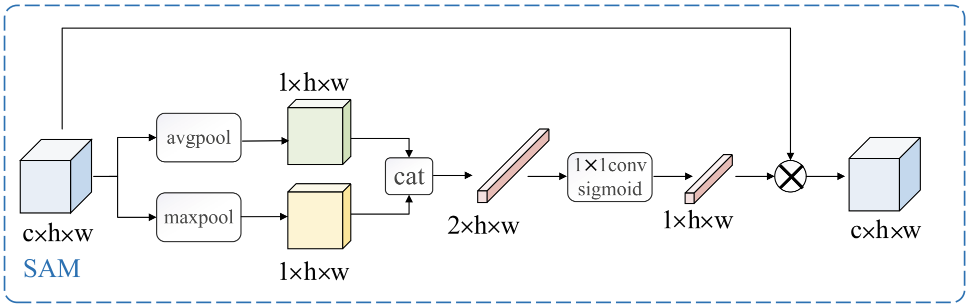


Fig. 4 visualization of boundary reinforcement module

Fig. 5 spatial attention module



low-level features. The specific operations of LABRM can be formulated as follows:

$$Xcat = aux(cat(\mathrm{low}, \mathrm{high})) \tag{9}$$

$$Xmax = max(conv(Xcat)) \tag{10}$$

$$Xcl = lap(Xmax) + Xcat \tag{11}$$

$$Xsl = sam(Xcat) + Xmax \tag{12}$$

$$Xout = Xcl + Xsl \tag{13}$$

In Eqs. (9) – (13), *cat* denotes cascade operation, *aux* denotes auxiliary loss calculation, *conv* denotes 1×1 convolution, *max* denotes maximum pooling along the channel, *lap* denotes Laplace convolution, *sam* denotes spatial attention mechanism, and "+" denotes element-wise addition.

3.4 FBRNet

The network structure in this paper is shown in Fig. 6. Resnet18, a lightweight backbone network pre-trained in the ImageNet dataset, is used as the FBRNet encoder structure, aux denotes the auxiliary loss, and FBRNet performs loss calculation on the layer3 and layer4 outputs of the encoder, the location settings for auxiliary losses to the backbone network can be found in Sect. 4.3.5. To further extract multi-target features, FBRNet adds arASPP after the output of layer 4. ArASPP does not change the size of the feature map and then fuses the enhanced top-level features with the output of layer 3, both of which are jointly used as input to the CSFM module. The output feature map of the first CSFM is then fused with the output of layer 2 in the same way to obtain the advanced fused features. To guide the enhanced edge representation of the low-level features, the fused features are used as input to the two LABRM layers together with the maxpool layer output and layer1 output, respectively. Auxiliary loss calculation is also available in the LABRM module, and the specific loss factor settings are described in detail in section 4.3.6. The output feature

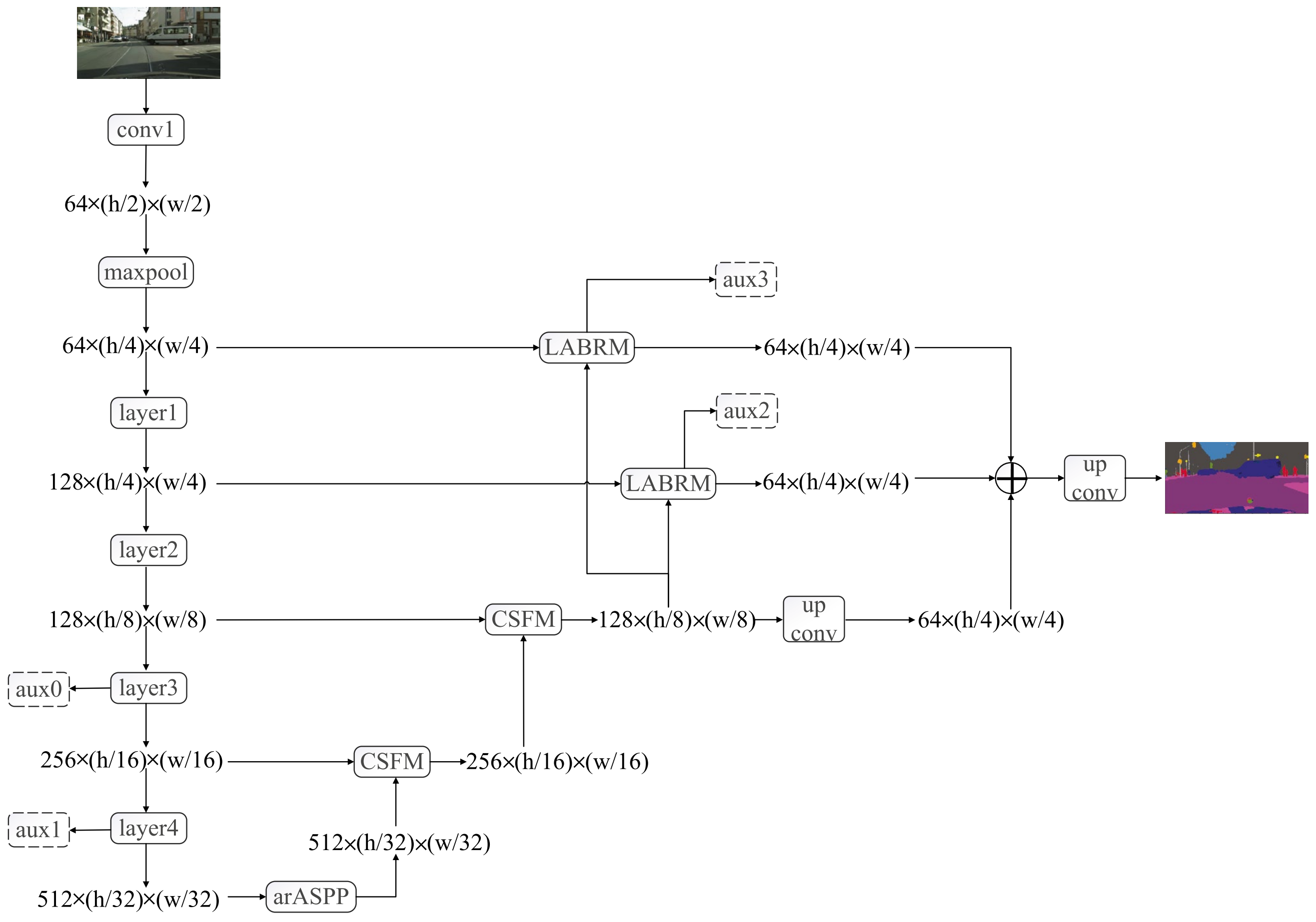


Fig. 6 architecture of network

map size of both LABRM modules is 64×h/4×w/4. To fuse the two enhanced primary feature maps after LABRM, the advanced fusion features are upsampled and convolved to make the advanced feature map consistent with the primary feature size. Finally, the upsampled advanced features and the two enhanced primary features are added at the elemental level, and the final output is obtained by upsampling and convolution.

4 Experiments

In this section, we will first discuss the dataset on which the experiment was performed, then introduce the experimental details, and finally discuss the ablation experiment and comparison experiment.

4.1 Datasets

The Cityscapes dataset is a dataset of image segmentation in a driverless environment. It contains street scenes from 50 cities worldwide with different environments. Only 19 categories for this training. The dataset is divided into training set, validation and sets by the number of 2975, 500, and 1000, and each image has a resolution of 1024×2048.

The CamVid dataset is the first semantic segmentation dataset to be used in autonomous driving, which includes 701 images and 32 classes of annotated objects. Only 11 classes are of interest for this training. The set is divided into training, validation and test sets by the number of 367, 101 and 233; each image has a resolution size of 960×720.

The BDD100K dataset covers real driving scenarios with diverse environments. The data were annotated with 8000 images for semantic segmentation, of which 7000 images were used for training and 1000 images for validation. Each image has a resolution size of 720×1280 and uses 19 classes labeled for semantic segmentation.

4.2 Implement details

In this section, we will first introduce the metrics used to evaluate the experimental results and then introduce the parameter settings of the experiment.

4.2.1 Metrics

- Use the average intersection ratio mIoU as an accuracy measure.
- The number of frames processed per second(FPS) is used as a speed measure.
- Use Params for the number of parameters in each network.
- Use GFLOPs for the number of computations in each network.

4.2.2 Settings

The preprocessing of the input image during training includes random cropping, horizontal rotation, and randomly changing the brightness, hue, and contrast of the image. The batch size is set to 4, and the stochastic gradient descent method (SGD) with momentum is used for model training, and Eq. (14) is the learning rate decline formula. We use the weighted cross-entropy loss function for primary loss and auxiliary loss calculation; the specific loss calculation is shown in Eq. (15). FBRNet is implemented using the deep learning framework PyTorch−1.3. A Tesla-T4 GPU and 16 CPU cores are used for model training and testing.

$$lr = lr_{base} \times (1 - \frac{iter}{maxiter})^{power} \tag{14}$$

$$loss = loss_{pre} + aux0 + aux1 + \alpha \times aux2 + \alpha \times aux3 \tag{15}$$

lr_{base} represents the initial learning rate, set to 0.01, *iter* represents the current number of iterations, maxiter represents the maximum number of iterations, set to 80000, momentum power is set to the default value of 0.9, and the attenuation factor is 0.00005. In Eq (15), $loss_{pre}$ represents the loss of the entire network, aux0 represents the aux0 auxiliary loss in Fig. 6, aux1, aux2, and aux3 also represent the corresponding position auxiliary loss in Fig. 6, α represents the coefficients of aux2 and aux3, set to 3, and the numerical setting of α will be discussed in section 4.3.6.

4.3 Ablation experiments

In this section, we will discuss the ablation experimental effects of arASPP, CSFM and LABRM modules in turn. The ablation experiments of each module will display experimental data and visualization effects.

4.3.1 ArASPP

To verify the effectiveness of the arASPP module in FBRNet, we conducted experiments on the arASPP module. The experimental results are shown in Table 1. When the arASPP module without ARM was added, although the inference speed was reduced by 48 frames, the segmentation accuracy of the network was improved by 9.2% due to the module's strong multi-scale feature capture capability. To further strengthen the features after the dilated convolution in arASPP, we add the attention reinforcement module ARM

Table 1 Compare of arASPP module

Method	Weight (M)	Params (G)	GFLOPs (G)	mIoU	FPS
res18	**46.10**	**11.70**	**81.82**	61.31	178.2
res18+ASPP	80.68	20.78	101.73	70.87	110.0
res18+arASPP	55.22	14.11	86.81	**71.47**	**116.1**

after each branch, so that the features in each branch can adaptively adjust the weight according to the attention vector obtained after global pooling, as can be seen in Table 1, after adding ARM, the accuracy of the overall network reaches 71.47%. We also added the ASPP module to the backbone for experiments, and the segmentation accuracy was 70.87%, and the inference speed was 110.0 frames. Due to the large dilation rate of the dilated convolution in ASPP and the large number of feature map channels participating in convolution, their inference speed is not as good as arASPP. And because our arASPP design has more parallel branches than ASPP, and the addition of a well-designed attention reinforcement module ARM can enhance the feature representation, the segmentation accuracy of arASPP is also higher than that of ASPP. All bolded data in the tables in this paper indicate the best experimental results.

4.3.2 CSFM

As shown in Table 2, the mIoU obtained by res18+arASPP is 71.47%. Since the CSFM module in FBRNet is composed of two modules, ContextR and DetailR, to verify the effectiveness of these two submodules, we performed ablation experiments on both modules. Compared with the network that did not use DetailR, the number of parameters increased by 2.7M, the calculation amount increased by 39.3G, and the segmentation speed was also reduced by 5 frames, but its re-extraction of low-level detail information helped the network improve the segmentation accuracy by 0.62%. The network using ContextR also has a certain improvement in the number of parameters and calculations because it uses the dilated convolution with dilation 5, the segmentation speed decreases more, but it also strengthens the further extraction of global information by the network, and the accuracy of network segmentation is increased to 72.54%. The CSFM is assembled by DetailR and ContextR, and CSFM absorbs the advantages of DetailR and contextR, and the mIoU is increased to 72.92%, and the model inference speed is still 102.9 frames.

4.3.3 LABRM

To verify the effectiveness of the LABRM module in FBRNet, the LABRM module, and its submodules were ablated experimentally. As shown in Table 3, the network segmentation accuracy composed of res18+arASPP reaches 71.47%, and only the LAP branch (refers to using only the Laplace branch in this module) in the LABRM module is used to fuse the lowest two-layer features, and the model segmentation accuracy is increased to 73.12% due to its powerful edge feature extraction ability, but the use of Laplace convolution also reduces the network inference speed to 105.9 frames. Only the SAM fusion of the lowest two layers of features (refers to using only the SAM branch in this module), the obtained network segmentation accuracy is 73.44%, and there are fewer convolution operations in the spatial attention mechanism, so the FPS of the network is faster than that of the res18+arASPP+LAP combination. The LABRM module is obtained by fusing Laplace convolution and SAM, which has both the edge extraction ability of the Laplace operator and the position reweight redistribution ability of the spatial attention mechanism, and the segmentation accuracy reaches 73.78%, and the segmentation speed is still 104.4 frames.

Table 3 Ablation experiments of LABRM

Method	Params (M)	GFLOPs (G)	mIoU	FPS
res18+arASPP	**14.11**	**86.81**	71.47	**116.1**
res18+arASPP+LAP	14.13	89.32	73.12	105.9
res18+arASPP+SAM	14.16	93.32	73.44	107.7
res18+arASPP+LABRM	14.19	101.32	**73.78**	104.4

Table 2 Ablation experiments of CSFM

Method	Params (M)	GFLOPs (G)	mIoU	FPS
res18+arASPP	**14.11**	**86.81**	71.47	**116.1**
res18+arASPP+DetailR	16.88	125.30	72.09	111.8
res18+arASPP+ContextR	16.90	128.02	72.54	104.2
res18+arASPP+CSFM	17.71	137.90	**72.92**	102.9

4.3.4 FBRNet

To verify the performance impact of various decoding methods on the network, we did several sets of experiments, the experimental data shown in Table 4 are summarized. Res18+arASPP achieved 71.47% mIoU for the coded network while decoding using element-wise addition and cascading yielded mIoU even inferior to baseline, which shows that neither decoding method is suitable for the network. In contrast, the decoding structure in FBRNet can still improve mIoU by 6.2% based on res18+arASPP.

To deeply analyze the impact of the modules on the network, we conducted ablation experiments. As shown in Table 5, the segmentation accuracy of the network with only resnet18 is 61.31%, and with the addition of the arASPP module, the segmentation accuracy of the model is improved to 71.47% due to the powerful multi-scale feature extraction capability of arASPP, which can also be seen in Column 4 of Fig. 7, where the model's segmentation of the objects is more complete and continuous. Base on resnet18+arASPP, we add CSFM and LABRM in turn. From Table 5, we can see that the segmentation accuracy of the model increases by 1.5% after adding CSFM module due to the stronger feature fusion ability of CSFM, and the segmentation effect for each object is more smooth as can be seen in Column 5 of Fig. 7. Since the LABRM module is more sensitive to edge features, the combination of resnet18+arASPP+LABRM improves the segmentation accuracy by 2.3% compared with resnet18+arASPP, and it can also be visualized from column 6 of Fig. 7 that, for the small objects and the street lights with stronger geometric properties, resnet18+arASPP+LABRM has a better segmentation effect. Combining resnet18 with arASPP, CSFM, and LABRM together, the network experimental effect obtained is even as high as 77.63%. It can also be seen from Table 5 that the segmentation accuracy of the last row of the network is higher than that of the previous rows, which is because that the network combines the

Table 4 Compare experiments of decoder

Method	Params (M)	GFLOPs (G)	mIoU	FPS
res18+arASPP	**14.11**	**86.81**	71.47	**116.1**
res18+arASPP+addition	14.19	93.99	69.95	110.4
res18+arASPP+cat	14.18	95.59	70.11	110.2
FBRNet(ours)	17.62	147.71	**77.63**	101.9

Table 6 Compare experiments of decoder

layer4	layer3	layer2	mIoU
√			72.02
√	√		**72.92**
√	√	√	72.41

Table 5 Results of ablation experiments

resnet18	arASPP	CSFM	LABRM	Params (M)	GFLOPs (G)	mIoU	FPS
√				**11.70**	**81.82**	61.31	**178.2**
√	√			14.11	86.61	71.47	116.1
√	√	√		17.71	137.90	72.92	102.9
√	√		√	14.19	101.32	73.78	104.4
√	√	√	√	17.62	147.71	**77.63**	101.9

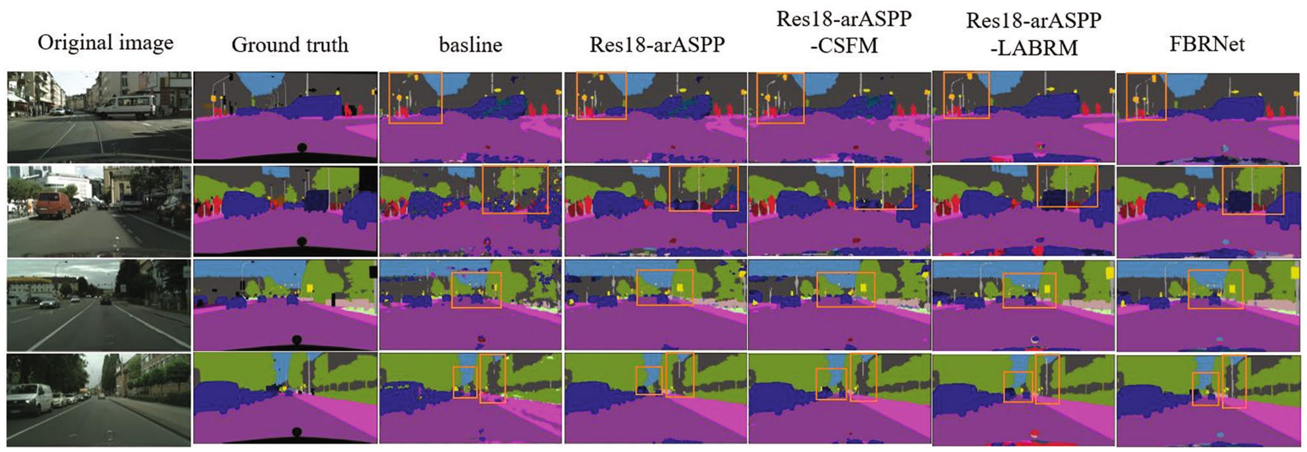


Fig. 7 visualization of ablation experiments

advantages of multi-scale features, feature fusion and edge enhancement, and it can also be seen from the last column of Fig. 7 that the network has the most complete segmentation effect when compared with the previous columns.

4.3.5 Baseline auxiliary loss experiments

During the training process, we tried to set the auxiliary loss at different positions of the encoder to obtain the optimal coefficient of auxiliary loss, as shown in Table 6. The segmentation accuracy obtained by setting the auxiliary loss at layer 4 was 72.02%. The segmentation accuracy was 72.92% when the auxiliary loss at both layer 4 and layer 3; compared to setting the auxiliary loss at layer 4 only, the segmentation accuracy is 0.9% higher; we also tried to set the auxiliary loss was at layer 4, layer 3, and layer 2 at the same time, but the accuracy is 0.5% lower. Therefore, FBRNet only sets the auxiliary loss at layer 4 and layer 3 to obtain the optimal performance of the model.

4.3.6 LABRM auxiliary loss factor experiments

FBRNet sets the auxiliary loss at LABRM to obtain the optimal experimental results, and we investigated the effect of different auxiliary loss factors α on the experimental results, as shown in Table 7. The optimal segmentation accuracy is achieved when α is set to 3 in both LABRM modules.

4.4 Comparison experiments

To verify the effectiveness of the algorithm in this paper, we will conduct comparative experiments on cityscapes, CamVid and BDD100K datasets in turn, and discuss the results of the experiments.

4.4.1 Cityscapes datasets

Table 8 shows the segmentation performance of the best models on the Cityscapes dataset in recent years. Compared to the classic network BiSeNetV1 [8], which uses the same backbone, FBRNet does not add unnecessary branches, resulting in faster segmentation speed. Additionally, BiSeNetV1 [8] only merges information from two branches to obtain the final features. In contrast, the decoder of FBRNet fuses features at each layer, resulting in a more comprehensive feature representation. As a result, FBRNet achieves better segmentation accuracy than BiSeNetV1 [8] on both the validation set and the test set. Despite CSRNet [32] using a multi-stage hierarchical approach to extract features at different levels, its feature fusion method is not comprehensive enough. In comparison, FBRNet outperforms CSRNet [32] on both the validation set and the test set. Compared to classic networks such as ICNet [9], ERFNet [33], and FasterSeg [34], FBRNet performs better in terms of both segmentation accuracy and speed. This makes FBRNet a more efficient and effective choice for various segmentation tasks. To compare the performance of each network, Fig. 8 shows the visualization of each network on the cityscapes dataset. From the first row of images in Fig. 8, it can be observed that FBRNet performs better in the segmentation of streetlights compared to the previous networks, and the segmented objects are more complete. In the second row of images in Fig. 8, only FBRNet successfully segments the streetlight within the rectangular box, indicating that FBRNet has better segmentation results compared to the other networks. Since some networks do not have open codes, the corresponding network properties cannot be obtained, so we fill "–" in the corresponding cell, FPS represents the image segmentation speed in the original paper of the network, and FPS+ represents the segmentation speed tested using the same GPU as FBRNet.

Table 7 Experiments of LABRM auxloss

brm_aux2	brm_aux3	mIoU
0.5	0.5	71.40
1	1	72.92
2	2	73.44
3	**3**	**73.78**
4	4	73.68

4.4.2 CamVid datasets

It can be seen in Table 9 that FBRNet has the fastest inference speed under the same GPU conditions, while other networks that cannot test FPS due to non-open sources are not as accurate as FBRNet. Compared with BiSeNetV1 [8] using the same backbone network, FBRNet's segmentation accuracy is improved by 3.7%, and the segmentation speed is also improved by 13 FPS. Compared with RGPNet [52] and DCNet [44], FBRNet has improved segmentation accuracy by 5.5% and 6.2%, respectively. Although BiSeNetV2 [41] has reached the same mIoU as FBRNet in the CamVid dataset, and the inference speed is also important for real-time semantic segmentation, the inference speed of BiSeNetV2 [41] is only 46.6FPS, which is approximately 1/2 of FBRNet; it can be seen that FBRNet has also achieved good performance in the CamVid dataset. LARNet [48] adopts a similar approach to ICNet [9] in extracting features at different scales. However, this network only utilizes a cascaded fusion method to merge these features, which can lead to interference between features and loss of essential information. In comparison, FBRNet's feature fusion method is more reasonable. As a result, it achieves a 5.3% improvement in

Table 8 Compare experiments of Cityscapes datasets

Method	Year	Input size	Baseline	Params (M)	GFLOPs (G)	mIoU		GPU	FPS	FPS+
						val	test			
ICNet [9]	2018	1024×2048	PSP50	28.3	26.5	71.7	69.5	TitanX	30.3	15.2
ERFNet [33]	2018	1024×2048	No	2.1	30.06	70.0	69.7	TitanX	83	79.9
BiSeNetV1 [8]	2018	1024×2048	Xception	2.41	53.1	69.0	68.4	TitanX	105.8	60.9
BiSeNetV1 [8]	2018	1024×2048	res18	61.1	30.14	74.8	74.7	TitanX	65.5	83.0
DFANet-A [10]	2019	1024×2048	Xception	7.8	**1.7**	71.9	71.3	TitanX	100	25.2
FasterSeg [34]	2019	1024×2048	No	3.4	28.2	73.1	71.5	1080Ti	163.9	94.2
LEDNet [35]	2019	512×1024	No	0.94	11.44	–	70.6	1080Ti	71.0	40.9
DABNet [36]	2019	1024×2048	No	0.76	42.43	–	70.1	1080Ti	104	78.8
LiteSeg [37]	2019	360×640	MobileNet	4.38	4.9	–	67.8	1080Ti	161	78.0
FDDWNet [38]	2020	512×1024	No	0.8	14.17	–	71.5	2080Ti	60.0	41.0
DSNet [39]	2020	512×1024	DenseNet	11.9	–	–	69.1	1080Ti	68.0	–
AGLNet [40]	2020	512×1024	No	1.12	13.88	69.39	70.1	1080Ti	52.0	–
BiSeNetV2 [41]	2021	512×1024	No	201.6	21.15	73.4	69.1	1080Ti	156.0	–
JPANet [26]	2021	512×1024	GhostNet	3.49	10.9	–	71.6	1080Ti	109	–
LEANet [42]	2021	512×1024	No	0.74	–	–	72.9	1080Ti	98.6	83.4
Lite-HRNet [43]	2021	512×1024	HRNet18	1.1	1.95	73.8	72.8	V100	–	–
DCNet [44]	2021	512×1024	res18	–	–	–	71.2	2080Ti	142	–
DSANet [45]	2021	512×1024	No	11.9	37.4	79.8	71.4	1080Ti	34.0	–
RELAXNet [46]	2021	512×1024	No	1.9	22.84	–	74.8	2080Ti	64.0	–
LSNet [47]	2022	1024×1024	res18	–	–	73.9	73.9	RTX3080	130	–
CSRNet-L [32]	2023	768×768	res18	–	–	76.1	74.0	1080Ti	56.0	–
CSRNet-M [32]	2023	768×768	res18	–	–	76.6	75.3	1080Ti	52.5	–
LARNet [48]	2023	512×1024	No	0.95	–	–	71.1	2080Ti	105	–
RCNet [49]	2023	1024×2048	No	1.96	20.92	72.49	–	V100	59.2	–
LBNet [50]	2023	512×1024	No	0.76	9.83	–	69.6	1080Ti	70.0	–
ELANet [51]	2023	512×1024	No	**0.67**	9.7	–	74.7	1080Ti	47.0	45.2
FBRNet(ours)		1024×2048	res18	17.62	147.7	**77.6**	**75.3**	Tesla-4	101.9	**101.9**

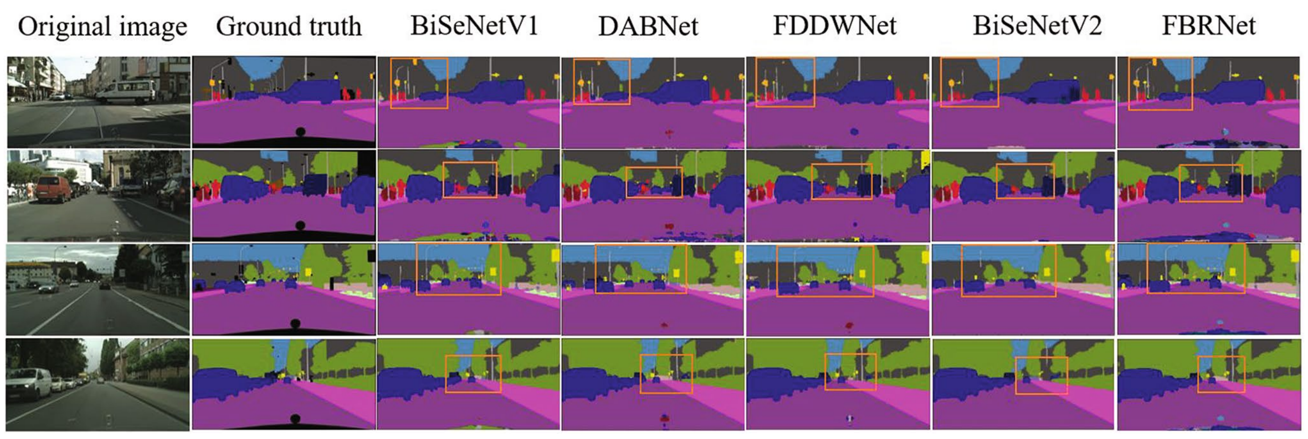


Fig. 8 visualization of compare experiments in cityscapes datasets

segmentation accuracy on the CamVid dataset. Compared to all the open-source models in Table 9, FBRNet achieves the best results in terms of speed and accuracy. Figure 9 shows the visualization of each network on the CamVid dataset. From the second row of images in Fig. 9, it can be observed that only BiSeNetV1 [8] and FBRNet successfully segment

Table 9 Compare experiments of CamVid datasets

Method	Year	Input size	Baseline	Params (M)	GFLOPs (G)	mIoU	GPU	FPS	FPS+
ICNet [9]	2018	720×960	PSP50	97.94	26.5	67.1	TitanX	27.8	41.9
BiSeNetV1 [8]	2018	720×960	Xception	17.28	2.41	65.6	TitanX	175	45.7
BiSeNetV1 [8]	2018	720×960	res18	9.85	61.10	68.7	TitanX	116	76.8
DFANet-A [10]	2019	720×960	Xception	5.14	2.17	64.7	TitanX	120	27.4
FasterSeg [34]	2019	720×960	no	9.2	4.4	71.1	1080Ti	398	34.7
DABNet [36]	2019	360×480	no	3.49	**0.76**	66.4	1080Ti	–	10.3
FANet-18 [53]	2020	720×960	res18	15.7	13.65	69.0	TitanX	154	78.2
FDDWNet [38]	2020	360×480	no	4.66	0.8	66.9	2080Ti	79	10.38
BiSeNetV2 [41]	2021	720×960	no	17.5	2.41	72.4	1080Ti	125	46.6
RGPNet [52]	2021	352×480	res18	–	17.7	66.9	V100	190	–
DCNet [44]	2021	360×480	res18	–	–	66.2	2080Ti	166	–
DSANet [45]	2021	360×480	no	101	3.47	69.9	1080Ti	75.3	34.9
FPANet-A [27]	2021	720×960	res18	69.21	14.11	68.6	2080Ti	151	–
JPANet [26]	2021	360×480	GhostNet	–	3.49	67.45	1080Ti	294	–
RELAXNet [46]	2021	360×480	no	–	1.9	71.2	2080Ti	79	–
LSNet [47]	2022	720×960	res18	–	–	72.3	RTX3080	105	–
FEENet [54]	2023	360×480	no	–	29.96	68.1	2080Ti	225	–
LARNet [48]	2023	360×480	no	0.95	–	67.1	2080Ti	204	–
ELANet [51]	2023	360×480	–	**0.76**	9.7	67.9	1080Ti	–	32.7
FBRNet(ours)		720×960	res18	47.57	17.62	**72.4**	Tesla-4	89.8	**89.8**

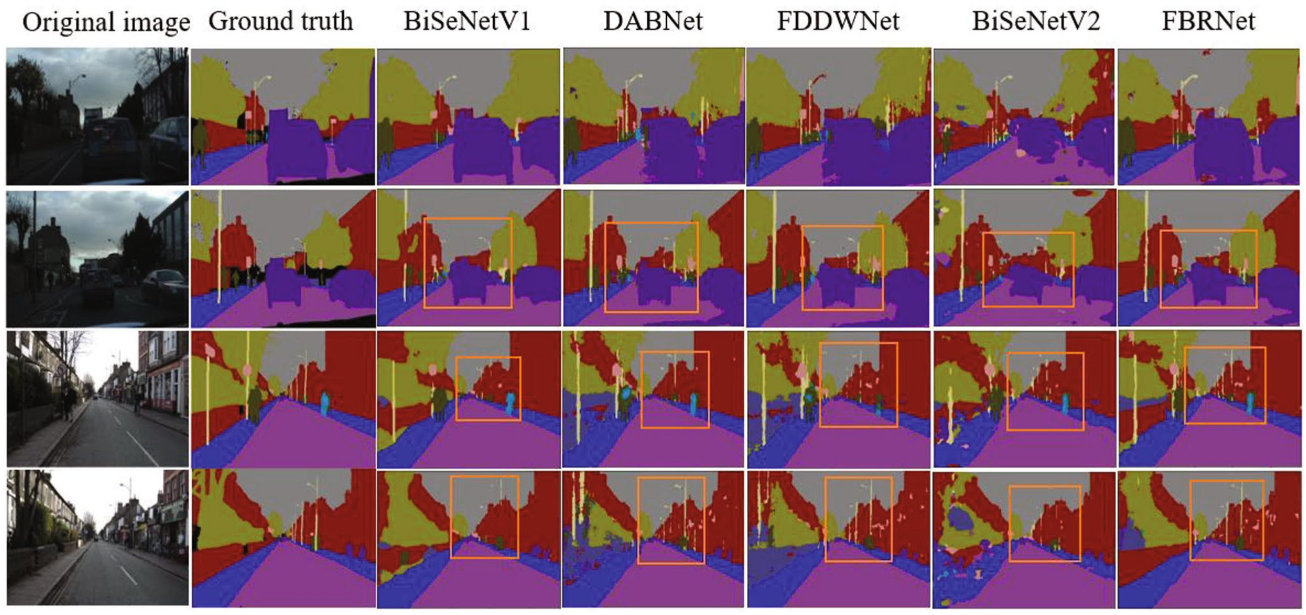


Fig. 9 visualization of compare experiments in CamVid datasets

the vehicles completely. Compared to BiSeNet [8], FBRNet provides more accurate segmentation of the streetlights within the rectangular box. In the third row of images, it is also evident that FBRNet has the best segmentation results for the streetlights within the rectangular box.

4.4.3 BDD100K datasets

As seen in Table 10, FBRNet has the highest segmentation accuracy, reaching 55.2% mIoU. Although DRN-D-38 achieves the same segmentation accuracy as FBRNet, it also comes at a huge cost, with an inference speed of only 4.5

Table 10 Compare experiments of BDD100k datasets

Method	Year	Input size	Baseline	Params (M)	GFLOPs (G)	mIoU	FPS+
DRN-D-22 [55]	2018	720×1280	DRN22	15.90	244.5	53.2	6.5
DRN-D-38 [55]	2018	720×1280	DRN38	26.50	388.8	**55.2**	4.5
BiSeNetV1 [8]	2018	720×1280	res18	13.43	107.69	52.4	28.9
FasterSeg [34]	2019	720×1280	No	4.4	**12.4**	55.1	43.7
DABNet [36]	2019	720×1280	No	**0.76**	37.29	51.8	44.9
FDDWNet [38]	2020	720×1280	No	0.81	24.90	52.8	7.9
BiSeNetV2 [41]	2021	720×1280	No	5.23	128.18	48.3	86.2
FBRNet(ours)		720×1280	res18	17.62	65.07	**55.2**	**100.8**

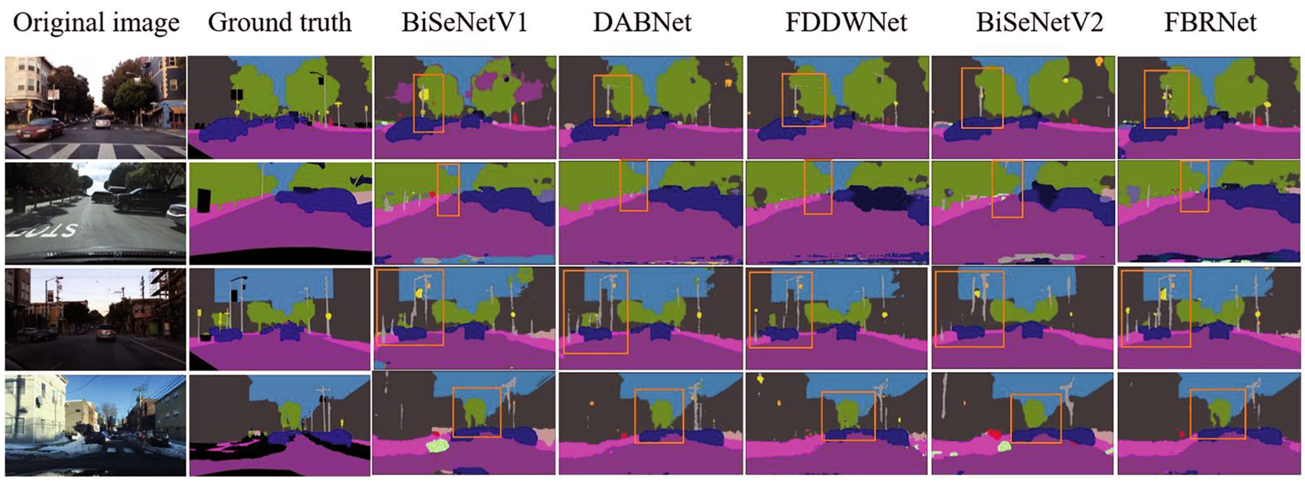


Fig. 10 visualization of compare experiments in BDD100k datasets

FPS, which is far from meeting the real-time requirements. Compared with BiSeNetV1 using the same backbone network, the segmentation accuracy of FBRNet is improved by 2.8%, and the inference speed is also equivalent to 3 times that of BiSeNetV1. Due to its multi-scale feature fusion method and deep supervised training strategy based on feature pyramid network FPN, FasterSeg also has a segmentation accuracy of 55.1%, which is only 0.1% lower than FBRNet, but it is precise because of its complex network structure that its inference speed is only 1/2 of FBRNet. Figure 10 shows the visual segmentation results of each network on the BDD100K dataset. Due to the use of Laplacian as an edge detection operator in FBRNet, it can be observed from the third row of rectangular boxes in Fig. 10 that FBRNet achieves better contour segmentation results for various objects compared to the previous networks.

5 Conclusion

Based on previous research, we have designed a new real-time semantic segmentation network, FBRNet, which achieves a better balance between segmentation accuracy and speed. Our designed arASPP module can extract multi-scale features in a short time, with smaller weights and higher segmentation accuracy. The CSFM module further extracts features from both contextual and detailed perspectives, helping to better integrate features from different layers. The LABRM module combines the Laplacian operator and spatial attention mechanism for the first time, fully utilizing the capabilities of existing edge information extraction algorithms, and improving the segmentation accuracy of the model. We have conducted comparative experiments on three datasets to validate that our algorithm performs better than existing ones. Real-time semantic segmentation has a wide range of research possibilities, such as occlusion segmentation, real-time semantic segmentation of small objects, and real-time semantic segmentation with limited data. All of these studies require more efficient real-time semantic segmentation networks. In the method of this paper, we introduce several new modules, which increase the complexity of the model to some extent, and in the future, we will further simplify the network structure while maintaining the high performance of the network, focusing on further exploration and design of a lightweight backbone network that can efficiently extract multi-scale information while preserving edge features, and we will apply this network to various domains mentioned above.

Author contributions S.Q. contributed to conceptualization and writing—review and editing. Z.W. contributed to conceptualization, methodology, and writing—original draft. J.W. involved in supervision and investigation. Y.F. involved in supervision and investigation.

Funding The work was supported by the National Natural Science Foundation of China (No. 12071126), the Scientific Research Fund of Hunan Provincial Education Department, China (23A0081).

Data availability The Cityscapes datasets are openly available in https://www.cityscapes-dataset.com. The CamVid datasets are openly available in http://mi.eng.cam.ac.uk/research/projects/VideoRec/CamVid. The BDD100k datasets openly available in https://www.bdd100k.com. Our results of the cityscapes test set can be seen at https://www.cityscapes-dataset.com/method-details/?submissionID=16865&back=mysubmissions.

Declarations

Conflicts of interests The authors declare no competing interests.

Ethical approval Written informed consent for publication of this paper was obtained from all authors.

References

1. Sun Y, Zheng W (2023) Hrnet-and pspnet-based multiband semantic segmentation of remote sensing images. Neural Comput Appl 35(12):8667–8675. https://doi.org/10.1007/s00521-022-07737-w
2. Mo Y, Wu Y, Yang X, Liu F, Liao Y (2022) Review the state-of-the-art technologies of semantic segmentation based on deep learning. Neurocomputing 493:626–646. https://doi.org/10.1016/j.neucom.2022.01.005
3. Luo D, Kang H, Long J, Zhang J, Liu X, Quan T (2023) Gdn: guided down-sampling network for real-time semantic segmentation. Neurocomputing 520:205–215. https://doi.org/10.1016/j.neucom.2022.11.075
4. Li Y, Zhang W, Liu Y, Shao X (2022) A lightweight network for real-time smoke semantic segmentation based on dual paths. Neurocomputing 501:258–269. https://doi.org/10.1016/j.neucom.2022.06.026
5. Zhu H, Zhang M, Zhang X, Zhang L (2021) Two-branch encoding and iterative attention decoding network for semantic segmentation. Neural Comput Appl 33:5151–5166. https://doi.org/10.1007/s00521-020-05312-9
6. Long J, Shelhamer E, Darrell T (2015) Fully convolutional networks for semantic segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 3431–3440. https://doi.org/10.1109/CVPR.2015.7298965
7. Sandler M, Howard A, Zhu M, Zhmoginov A, Chen L-C (2018) Mobilenetv2: Inverted residuals and linear bottlenecks. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 4510–4520. https://doi.org/10.1109/CVPR.2018.00474
8. Yu C, Wang J, Peng C, Gao C, Yu G, Sang N (2018) Bisenet: Bilateral segmentation network for real-time semantic segmentation. In: Proceedings of the European conference on computer vision, pp. 325–341. https://doi.org/10.1007/978-3-030-01261-8_20
9. Zhao H, Qi X, Shen X, Shi J, Jia J (2018) Icnet for real-time semantic segmentation on high-resolution images. In: Proceedings of the European conference on computer vision, pp. 405–420. https://doi.org/10.1007/978-3-030-01219-9_25
10. Li H, Xiong P, Fan H, Sun J (2019) Dfanet: Deep feature aggregation for real-time semantic segmentation. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp. 9522–9531. https://doi.org/10.1109/cvpr.2019.00975
11. Chen L-C, Papandreou G, Kokkinos I, Murphy K, Yuille AL (2017) Deeplab: semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. IEEE Trans Pattern Anal Mach Intell 40(4):834–848. https://doi.org/10.1109/tpami.2017.2699184
12. Cordts M, Omran M, Ramos S, Rehfeld T, Enzweiler M, Benenson R, Franke U, Roth S, Schiele B (2016) The cityscapes dataset for semantic urban scene understanding. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 3213–3223. https://doi.org/10.1109/CVPR.2016.350
13. Brostow GJ, Fauqueur J, Cipolla R (2009) Semantic object classes in video: a high-definition ground truth database. Pattern Recogn Lett 30(2):88–97. https://doi.org/10.1016/j.patrec.2008.04.005
14. Yu F, Chen H, Wang X, Xian W, Chen Y, Liu F, Madhavan V, Darrell T (2020) Bdd100k: A diverse driving dataset for heterogeneous multitask learning. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp. 2636–2645. https://doi.org/10.48550/arXiv.1805.04687
15. Ronneberger O, Fischer P, Brox T (2015) U-net: Convolutional networks for biomedical image segmentation. In: Medical image computing and computer-assisted intervention, pp. 234–241. https://doi.org/10.1007/978-3-319-24574-4_28
16. Su Z, Li W, Ma Z, Gao R (2022) An improved u-net method for the semantic segmentation of remote sensing images. Appl Intell 52(3):3276–3288. https://doi.org/10.1007/s10489-021-02542-9
17. Li Y, Wang Z, Yin L, Zhu Z, Qi G, Liu Y (2023) X-net: a dual encoding-decoding method in medical image segmentation. Vis Comput 39(6):2223–2233. https://doi.org/10.1007/s00371-021-02328-7
18. Zhao H, Shi J, Qi X, Wang X, Jia J (2017) Pyramid scene parsing network. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 2881–2890. https://doi.org/10.48550/arXiv.1612.01105
19. Xiao C, Hao X, Li H, Li Y, Zhang W (2022) Real-time semantic segmentation with local spatial pixel adjustment. Image Vis Comput 123:104470. https://doi.org/10.1016/j.imavis.2022.104470
20. He X, Qi G, Zhu Z, Li Y, Cong B, Bai L (2023) Medical image segmentation method based on multi-feature interaction and fusion over cloud computing. Simul Model Pract Theory 126:102769. https://doi.org/10.1016/j.simpat.2023.102769
21. Xiao C, Hao X, Li H, Li Y, Zhang W (2022) Real-time semantic segmentation with local spatial pixel adjustment. Image Vis Comput 123:104470. https://doi.org/10.1016/j.imavis.2022.104470
22. Zhang B, Li W, Hui Y, Liu J, Guan Y (2020) Mfenet: multi-level feature enhancement network for real-time semantic segmentation. Neurocomputing 393:54–65. https://doi.org/10.1016/j.neucom.2020.02.019
23. Chen Y, Xia R, Yang K, Zou K (2023) Mffn: image super-resolution via multi-level features fusion network. Vis Comput 52:1–16. https://doi.org/10.1007/s00371-023-02795-0
24. Liu M, Yin H (2021) Efficient pyramid context encoding and feature embedding for semantic segmentation. Image Vis Comput 111:104195. https://doi.org/10.1016/j.imavis.2021.104195
25. Zhang X, Du B, Wu Z, Wan T (2022) Laanet: lightweight attention-guided asymmetric network for real-time semantic

segmentation. Neural Comput Appl 34(5):3573–3587. https://doi.org/10.1007/s00521-022-06932-z
26. Hu X, Jing L, Sehar U (2022) Joint pyramid attention network for real-time semantic segmentation of urban scenes. Appl Intell 52(1):580–594. https://doi.org/10.1007/s10489-021-02446-8
27. Wu Y, Jiang J, Huang Z, Tian Y (2022) Fpanet: feature pyramid aggregation network for real-time semantic segmentation. Appl Intell 52(3):3319–3336. https://doi.org/10.1007/s10489-021-02603-z
28. Zhu Z, He X, Qi G, Li Y, Cong B, Liu Y (2023) Brain tumor segmentation based on the fusion of deep semantics and edge information in multimodal MRI. Inform Fusion 91:376–387. https://doi.org/10.1016/j.inffus.2022.10.022
29. Zhang K, Li D, Luo W, Ren W (2021) Dual attention-in-attention model for joint rain streak and raindrop removal. IEEE Trans Image Process 30:7608–7619. https://doi.org/10.1109/TIP.2021.3108019
30. Zhang K, Luo W, Stenger B, Ren W, Ma L, Li H (2020) Every moment matters: Detail-aware networks to bring a blurry image alive. In: Proceedings of the 28th ACM international conference on multimedia, pp. 384–392 . https://doi.org/10.1145/3394171.3413929
31. Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, Kaiser Ł, Polosukhin I (2017) Attention is all you need. Adv Neural Inf Process Syst 30:329655. https://doi.org/10.5555/3295222.3295349
32. Xiong J, Po L-M, Yu W-Y, Zhou C, Xian P, Ou W (2023) Csrnet: cascaded selective resolution network for real-time semantic segmentation. Expert Syst Appl 211:118537. https://doi.org/10.1016/j.eswa.2022.118537
33. Romera E, Alvarez JM, Bergasa LM, Arroyo R (2017) Erfnet: efficient residual factorized convnet for real-time semantic segmentation. IEEE Trans Intell Transp Syst 19(1):263–272. https://doi.org/10.1109/tits.2017.2750080
34. Chen W, Gong X, Liu X, Zhang Q, Li Y, Wang Z (2019) Fasterseg: Searching for faster real-time semantic segmentation. arXiv preprint arXiv:1912.10917
35. Wang Y, Zhou Q, Liu J, Xiong J, Gao G, Wu X, Latecki LJ (2019) Lednet: A lightweight encoder-decoder network for real-time semantic segmentation. In: IEEE international conference on image processing, pp. 1860–1864. https://doi.org/10.1109/icip.2019.8803154
36. Li G, Yun I, Kim J, Kim J (2019) Dabnet: Depth-wise asymmetric bottleneck for real-time semantic segmentation. arXiv preprint arXiv:1907.11357
37. Emara T, Abd El Munim HE, Abbas HM (2019) Liteseg: A novel lightweight convnet for semantic segmentation. In: Digital image computing: techniques and applications, pp. 1–7. https://doi.org/10.1109/dicta47822.2019.8945975
38. Liu J, Zhou Q, Qiang Y, Kang B, Wu X, Zheng B (2020) Fddwnet: a lightweight convolutional neural network for real-time semantic segmentation. In: IEEE international conference on acoustics, speech and signal processing, pp. 2373–2377. https://doi.org/10.1109/icassp40776.2020.9053838
39. Wang W, Pan Z (2018) Dsnet for real-time driving scene semantic segmentation. arXiv preprint arXiv:1812.07049
40. Zhou Q, Wang Y, Fan Y, Wu X, Zhang S, Kang B, Latecki LJ (2020) Aglnet: towards real-time semantic segmentation of self-driving images via attention-guided lightweight network. Appl Soft Comput 96:106682. https://doi.org/10.1016/j.asoc.2020.106682
41. Yu C, Gao C, Wang J, Yu G, Shen C, Sang N (2021) Bisenet v2: bilateral network with guided aggregation for real-time semantic segmentation. Int J Comput Vis 129:3051–3068. https://doi.org/10.1007/s11263-021-01515-2
42. Zhang X-L, Du B-C, Luo Z-C, Ma K (2022) Lightweight and efficient asymmetric network design for real-time semantic segmentation. Appl Intell 52(1):564–579. https://doi.org/10.1007/s10489-021-02437-9
43. Yu C, Xiao B, Gao C, Yuan L, Zhang L, Sang N, Wang J (2021) Lite-hrnet: A lightweight high-resolution network. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp. 10440–10450. https://doi.org/10.1109/CVPR46437.2021.01030
44. Li Y, Liu Y, Sun Q (2021) Real-time semantic segmentation via region and pixel context network. In: International conference on pattern recognition, pp. 7043–7049. https://doi.org/10.1109/icpr48806.2021.9413018
45. Elhassan MA, Huang C, Yang C, Munea TL (2021) Dsanet: Dilated spatial attention for real-time semantic segmentation in urban street scenes. Expert Syst Appl 183:115090. https://doi.org/10.1016/j.eswa.2021.115090
46. Liu J, Xu X, Shi Y, Deng C, Shi M (2022) Relaxnet: residual efficient learning and attention expected fusion network for real-time semantic segmentation. Neurocomputing 474:115–127. https://doi.org/10.1016/j.neucom.2021.12.003
47. Sheng P, Shi Y, Liu X, Jin H (2022) Lsnet: real-time attention semantic segmentation network with linear complexity. Neurocomputing 509:94–101. https://doi.org/10.1016/j.neucom.2022.08.049
48. Hu X, Xu S, Jing L (2023) Lightweight attention-guided redundancy-reuse network for real-time semantic segmentation. IET Image Process 17:2649–2658. https://doi.org/10.1049/ipr2.12816
49. Dong Y, Zhao K, Zheng L, Yang H, Liu Q, Pei Y (2023) Refinement co-supervision network for real-time semantic segmentation. IET Comput Vis 17:652–662. https://doi.org/10.1049/cvi2.12187
50. Wang P, Li L, Pan F, Wang L (2023) Lightweight bilateral network for real-time semantic segmentation. J Adv Comput Intell 27(4), 673–682. https://doi.org/10.20965/jaciii.2023.p0673
51. Yi Q, Dai G, Shi M, Huang Z, Luo A (2023) Elanet: effective lightweight attention-guided network for real-time semantic segmentation. Neural Process Lett 55:6425–6442. https://doi.org/10.1007/s11063-023-11145-z
52. Arani E, Marzban S, Pata A, Zonooz B (2021) Rgpnet: A real-time general purpose semantic segmentation. In: Proceedings of the IEEE/CVF winter conference on applications of computer vision, pp. 3009–3018 . https://doi.org/10.1109/wacv48630.2021.00305
53. Hu P, Perazzi F, Heilbron FC, Wang O, Lin Z, Saenko K, Sclaroff S (2020) Real-time semantic segmentation with fast attention. IEEE Robot Autom Lett 6(1):263–270. https://doi.org/10.1109/LRA.2020.3039744
54. Tan S, Yang W, Lin J, Yu W (2023) Feature extraction and enhancement for real-time semantic segmentation. Concurr Comput 35(17):6573. https://doi.org/10.1002/cpe.6573
55. Yu F, Koltun V, Funkhouser T (2017) Dilated residual networks. In: Proceedings of the IEEE conference on cComputer vision and pattern recognition, pp. 472–480. https://doi.org/10.48550/arXiv.1705.09914