FBSNet: A Fast Bilateral Symmetrical Network for Real-Time Semantic Segmentation

Guangwei Gao, Member, IEEE, Guoan Xu, Juncheng Li, Yi Yu, Senior Member, IEEE, Huimin Lu, Senior Member, IEEE, and Jian Yang, Member, IEEE

Abstract—Real-time semantic segmentation, which can be visually understood as the pixel-level classification task on the input image, currently has broad application prospects, especially in the fast-developing fields of autonomous driving and drone navigation. However, the huge burden of calculation together with redundant parameters are still the obstacles to its technological development. In this article, we propose a Fast Bilateral Symmetrical Network (FBSNet) to alleviate the above challenges. Specifically, FBSNet employs a symmetrical encoder-decoder structure with two branches, semantic information branch and spatial detail branch. The Semantic Information Branch (SIB) is the main branch with semantic architectural modules to acquire the contextual information of the input image and meanwhile acquire sufficient receptive field. The Spatial Detail Branch (SDB) is a shallow and simple network used to establish local dependencies of each pixel for preserving details, which is essential for restoring the original resolution during the decoding phase. Meanwhile, a Feature Aggregation Module (FAM) is designed to effectively combine the output of these two branches. Experimental results of Cityscapes and CamVid show that the proposed FBSNet can strike a good balance between accuracy and efficiency. Specifically, it obtains 70.9% and 68.9% mIoU along with the inference speed of 90 fps and 120 fps on these two test datasets, respectively, with only 0.62 million parameters on a single RTX 2080Ti GPU. The code is available at https://github.com/IVIPLab/FBSNet.

Index Terms—Feature aggregation, local dependencies, real-time, semantic segmentation.

Manuscript received 15 November 2021; revised 23 January 2022 and 25 February 2022; accepted 1 March 2022. Date of publication 9 March 2022; date of current version 8 August 2022. This work was supported in part by the National Natural Science Foundation of China under Grants 61972212, 61772568, and 61833011, in part by the Natural Science Foundation of Jiangsu Province under Grant BK20190089, and in part by the Six Talent Peaks Project in Jiangsu Province under Grant KYFW-011. The Associate Editor coordinating the review of this manuscript and approving its publication was Dr. Chang Xu. (Guangwei Gao and Guoan Xu contributed equally to this work.) (Corresponding authors: Juncheng Li; Yi Yu.)

Guangwei Gao and Guoan Xu are with the Institute of Advanced Technology, Nanjing University of Posts and Telecommunications, Nanjing 210023, China, and also with the Digital Content and Media Sciences Research Division, National Institute of Informatics, Tokyo 101-8430, Japan (e-mail: csg-gao@gmail.com; xga_njupt@163.com).

Juncheng Li is with the Center for Mathematical Artificial Intelligence, Department of Mathematics, The Chinese University of Hong Kong, Hong Kong (e-mail: cvjunchengli@gmail.com).

Yi Yu is with the Digital Content and Media Sciences Research Division, National Institute of Informatics, Tokyo 101-8430, Japan (e-mail: yiyu@nii.ac.jp).

Huimin Lu is with the Department of Mechanical and Control Engineering, Kyushu Institute of Technology, Kitakyushu 804-8550, Japan (e-mail: dr.huimin.lu@ieee.org).

Jian Yang is with the School of Computer Science and Technology, Nanjing University of Science and Technology, Nanjing 210094, China (e-mail: csysyang@njust.edu.cn).

Digital Object Identifier 10.1109/TMM.2022.3157995

I. INTRODUCTION

SEMANTIC segmentation, as one of the three fundamental computer vision tasks, is responsible for assigning a label to each pixel in an input image [1]–[3]. It can be viewed as a dense prediction task and what cannot be overlooked is the parameter burden. However, to embed it in real-world terminal equipment like augmented reality devices or autonomous driving chips, it is necessary to ensure that the model size and calculation cost of the proposed model are as small as possible. In comparison with developing large networks (e.g., VGGNet [4] and ResNet [5], [6]) that pursue precision, designing real-time semantic segmentation structures is the proper choice that meets the needs of current real-world edge applications requiring fast interaction speed.

Recently, many lightweight real-time segmentation networks have been proposed to address the balance problem between accuracy and inference speed (Fig. 1). For example, Paszke et al. [7] proposed an Efficient Neural Network (ENet), which drops the last stage to achieve a compact encoder-decoder framework. However, the disadvantage of this model is that the receptive field is too small to capture large objects. In order to collect multi-scale contextual information, Mehta et al. [8] proposed the Efficient Spatial Pyramid Network (ESPNet), which adopts the efficient spatial pyramid module and convolution factorization strategy. ESPNet achieved better accuracy than ENet with similar parameters. The Image Cascade Network (ICNet [9]) is improved from the Pyramid Scene Parsing Network (PSPNet [10]), which uses three cascading branches to process images. Nevertheless, this network is not suitable for low-resolution input images. The Bilateral Segmentation Network (BiSeNet [11]) is a dual-path model, which contains two branches which can preserve the spatial details and obtain semantic information, respectively. Meanwhile, a feature fusion module was proposed to effectively aggregate the different features of these two branches. Based on BiSeNet, the fusion method of different features in BiSeNet-v2 [12] is further optimized. Although the effect of BiSeNet-v2 is much better than BiSeNet, the number of parameters and calculations has also increased a lot, which slows down the inference speed. DFANet [13] proposed a feature reuse strategy, which stacks Xception [14] models three times to yield the purpose of expanding the receptive field and the interactive integration of features. Fast-SCNN [15] proposed a “learning to downsample” module and merged the two-branch setup in the encoder phase.
A lightweight Bottleneck Residual Unit (BRU) is proposed to drastically distill significant semantic information. BRU contains a small number of parameters and only needs less calculation cost, but can extract rich features with different receptive fields.

A Detail Residual Module (DRM) is proposed to better obtain shallow spatial features to compensate for the lost details in the semantic information branch. Meanwhile, a Feature Aggregation Module (FAM) is proposed to effectively fuse image features from different branches, with the goal to increase the global and local dependencies.

A novel Fast Bilateral Symmetrical Network (FBSNet) is presented for real-time image semantic segmentation. FBSNet is a symmetrical encoder-decoder structure, composed of a Semantic Information Branch (SIB) and a Spatial Detail Branch (SDB), which can effectively extract deep semantic information and preserve shallow boundary details, respectively.

In summary, the contributions of this paper are as follows:

- **A lightweight Bottleneck Residual Unit (BRU)**: Proposed to drastically distill significant semantic information. BRU contains a small number of parameters and only needs less calculation cost, but can extract rich features with different receptive fields.
- **A Detail Residual Module (DRM)**: Proposed to better obtain shallow spatial features to compensate for the lost details in the semantic information branch. Meanwhile, a Feature Aggregation Module (FAM) is proposed to effectively fuse image features from different branches, with the goal to increase the global and local dependencies.
- **A novel Fast Bilateral Symmetrical Network (FBSNet)**: Presented for real-time image semantic segmentation. FBSNet is a symmetrical encoder-decoder structure, composed of a Semantic Information Branch (SIB) and a Spatial Detail Branch (SDB), which can effectively extract deep semantic information and preserve shallow boundary details, respectively.

### II. RELATED WORKS

#### A. Contextual Information

Contextual information is crucial in the field of deep learning-based image processing methods. It contains a large number of image features that can be used for subsequent tasks like classification, detection, segmentation, and other tasks to predict high-quality results [16]–[20]. For example, ParseNet [21] utilized global average pooling to generate weight feature maps for enhancing contextual information. Methods like [22]–[24] employed dilated convolution with various dilation rates to capture diversified contextual information. PSPNet [10], inspired from DeepLab-V3 [24], applied spatial pyramid pooling (SPP) to capture global contexts. DFN [25] encoded the global context by adding the global pooling on the head of the U-shape structure. Based on the above observations, it can be found that increasing the receptive field and acquiring a larger range of global contextual information is what every method cares about, and it is also beneficial to the final segmentation results.

#### B. Spatial Information

Spatial information is often ignored in the segmentation task. One is that with the deepening of the network layer and the use of consecutive down-sampling modules, the loss of spatial information is considered normal and unavoidable. The other is that restoring spatial information is a difficult task, which often consumes a lot of computing resources, so many methods ignore it [26]–[28]. In fact, the preservation of spatial details has a positive effect on the prediction of the final results. Common methods, such as UNet [29], directly concatenate the shallow details and the features with corresponding resolutions. This operation will bring a large number of parameters and calculations, which is not good for the inference speed. In order to supplement the spatial detail information, DABNet [30] inserted features after different multiples of downsampling at different stages in the phase of the encoder. ICNet [9] applied a cascading
method to supplement the missing boundary information. Some models [22], [24] adopted dense connections, but cannot fulfill the real-time requirement. These models are all effective methods, but the decoder lacks the guidance of semantic information, resulting in unsatisfactory performance.

C. General Structure

In the past ten years, many researchers have made a lot of efforts in network structure. However, many models still use the network structure of image classification (e.g., VGGNet [4], ResNet [5]) as the backbone of the segmentation model. Although this is the easiest way, two problems cannot be ignored: one is that this type of model has too many parameters, the other is that frequent use of downsampling operations will cause a large amount of feature loss, which is not conducive to feature discrimination. To solve these problems, some representative network structures came into being, and many subsequent networks are improved based on previous methods, such as U-shape structure [29], [31], feature reuse [13], [21], and two-branch structure [11], [12], [15]. In Fig. 2, we provide the structural comparison of some commonly-used image semantic segmentation models. Among them, the U-shape structure used the symmetrical encoder-decoder, whose strategy is to merge the feature maps of the corresponding stages. However, this kind of network is bound to bring huge additional computation. Feature reuse can enhance the network learning capacity and expand receptive fields by reusing high-level features. However, refining the spatial details is what prevents it from going further. Two-branch structure performs separate extraction of semantic information and spatial information in the encoder phase, and finally use a feature fusion method before prediction. But this method still lacks the interaction between the two branches, so there is still a lot of room for improvement.

In contrast to previous methods, the differences of our method are two-fold: (a) as for the two-branch structure, our main branch (semantic information branch) has a symmetrical encoder-decoder structure, while the spatial detail branch well preserves the shallow boundary details without the downsampling operations; (b) as for the feature fusion, to promote the representation capability of the fused features, except for the simple element-wise operation, we successively aggregate features at both semantic and spatial levels with negligible parameters and computational overhead.

III. PROPOSED METHOD

In this part, we will introduce the proposed FBSNet with the following parts: 1) Bottleneck Residual Unit (BRU); 2) Semantic Information Branch (SIB) with channel attention module; 3)
Spatial Detail Branch (SDB) with spatial attention module; 4) Network with initial block and Feature Aggregation Module (FAM).

A. Bottleneck Residual Unit (BRU)

In Fig. 4, we show some classic residual units that widely used in semantic segmentation. All of these modules are designed for feature feature extraction. Among them, (a) and (b) use the two-branch structure, one is responsible for local information extraction, while the other is dedicated to expanding the receptive field to obtain more contextual information. As for the pyramid cascade module in (c), it uses a series of dilation convolutions to gradually assemble multi-scale flexible representations. However, these models ignore the utilization of original features and the fusion of features from different branches is insufficient. To solve these problems, we proposed the Bottleneck Residual Unit (BRU). As shown in Fig. 4, BRU is a three-branch module, the left branch is responsible for extracting local and short-distance feature information, the right branch is designed for enlarging the receptive field to acquire long-distance feature information, and the middle branch is dedicated to saving the input information. Specifically, we absorb the advantages of the bottleneck, which can not only ensure the model effect but also greatly reduce the calculation budget. Meanwhile, the convolution factorization strategy also be adopted, which means that a standard 2-dimensional convolution kernel $K \times K$ is factored into two 1-dimensional convolution kernels, i.e., $1 \times K$ and $K \times 1$ followed by the batch normalization [32] and ReLU [33] operations. This strategy has been verified in many previous works such as Inception-v3 [34], Xception [14], MobileNet [35], [36], and ShuffleNet [37], [38], which can significantly reduce the model parameters while maintaining the model performance. Meanwhile, the use of depth-wise separable convolution is based on the same reason and a $1 \times 1$ point-wise convolution is used to recover the final channel dependency. Dilation convolution is applied for enlarging the receptive fields. The rule is that as the network deepens, the dilation rate gradually becomes larger. At the end of the module, the information from three branches is merged together and a channel shuffle operation is applied to enhance the module representation capability. The complete operation of the module can be defined as follows:

$$x_{out} = C_{1 \times 1}(x_{in}),$$  \hfill (1) 
$$y_{1,1} = f_{CAM}(C_{1 \times 3}(C_{3 \times 1}(x_{out}))),$$  \hfill (2) 
$$y_1 = C_{1 \times 3}(C_{3 \times 1}(y_{1,1})),$$  \hfill (3) 
$$y_{2,1} = f_{CAM}(C_{1 \times 3,r}(C_{5 \times 1,r}(x_{out}))),$$  \hfill (4) 
$$y_2 = C_{3 \times 1,r}(C_{3 \times 1,r}(y_{2,1})),$$  \hfill (5) 
$$y_3 = y_{1,1} + f_{CAM}(x_{out}) + y_{2,1},$$  \hfill (6) 
$$y_{out} = f_{shuffle}(f_{CAM}(C_{1 \times 1}(y_1 + y_2 + y_3)) + x_{in}),$$  \hfill (7)

where $x_{in}$ and $y_{out}$ mean the input and output of the BRU, respectively. $y_1$ and $y_2$ represent the output of the left, right, and middle branch, respectively. The first half of the outputs are represented by $y_{1,1}$ and $y_{2,1}$. Meanwhile, $f_{CAM}()$ denotes the CAM operation, $C_{m \times n}$ denotes the convolutional layer with a $m \times n$ convolution kernel, and $f_{shuffle}()$ is the channel shuffle operation. The activation function and batch normalization operations are omitted in the formulas.

B. Semantic Information Branch (SIB)

Recently, some modern approaches attempt to obtain sufficient receptive fields and enough contextual information by applying atrous spatial pyramid pooling, large kernel, or dense connection [10], [22], [24]. Although complex networks can bring good accuracy, the complex network and heavy computational overhead will make them incompatible with real-time requirements in real-world applications. The dual-branch structure network proposed by BiseNet [11] used ResNet [5] as the backbone of the model to extract contextual information. The
disadvantage is that it will take up a lot of GPU memory consumption and the used ResNet [5] is pre-trained by the ImageNet [39] which cannot be regarded as a end-to-end model. Different from previous works, we build a deep Semantic Information Branch (SIB) with our proposed BRU. This not only ensures that more semantic information can be captured and larger receptive fields can be obtained, but also maintain the number of parameters and calculations is very low. Different stages of the convolutional neural networks have different representation abilities: the shallow stages preserve abundant spatial information, such as edges and corners but poor contextual information; while the deep stages have enough semantic consistency but a coarse prediction. Therefore, we set different dilation rates in BRU at different stages of the branch.

1) Channel Attention Module (CAM): Since the channels contain abundant feature information and interference noise, we use the Channel Attention Module (CAM) in both BRU and the semantic information branch to emphasize the features that need to be highlighted. Meanwhile, this method can suppress the interference noise thus facilitate feature extraction. Although CBAM [40], Non-local [41], DANet [42], and CCNet [43] can bring good results, they are potentially not suitable for real-time segmentation tasks due to the complex calculation and a large number of parameters. Therefore, we use the lightweight attention method described in [44]. CAM uses global average pooling to obtain global contexts and generates an attention map to guide the feature extraction with negligible computational cost, which is a good way to improve the model performance. This method has been generally explored in the tasks of detection, recognition, and segmentation. The process can be given as

\[ M_c(X) = \sigma(C_{k \times k} (f_{\text{Trans}} (f_{\text{AvgPool}}(X)))) , \]

where \( M_c \in \mathbb{R}^{C \times 1 \times 1} \) is the channel attention map, \( X \in \mathbb{R}^{C \times H \times W} \) are the input features, \( C_{k \times k} \) represents the standard convolution operation with kernel size \( k \), \( f_{\text{AvgPool}}(\cdot) \) represents the average pooling operation, \( f_{\text{Trans}}(\cdot) \) denotes the compression and re-weighting operations, and \( \sigma \) is the Sigmoid function.

2) Downsampling Block: Inspired by the initial block in ENet [7], we adopt it as our downsampling block. Through the downsampling operation, we can achieve larger receptive fields. However, too many downsampling operations will make the boundary details loss too serious to be restored. This is not conducive to feature extraction and utilization. Therefore, we take a compromised strategy and only use the downsampling module twice. That is, each time it is downsampled to 1/2 of the input features using a downsampling block. So we end up downsampling the image to 1/8 of the original given size.

C. Spatial Detail Branch (SDB)

Spatial information will be inevitably lost during the processing of the semantic information branch. The reason is that the extraction of deep-level semantic information and the preservation of shallow-level boundary information are a pair of contradictory relationships. To solve this problem, we design the spatial detail branch, which is actually a supplement to the detailed information lost by the semantic information branch to help the model achieve better accuracy in prediction process. Different from SIB, we only use a simple and effective Detail Residual Module (DRM, Fig. 5(a)) and a Spatial Attention Module (SAM) in this branch. Among them, DRM is specially designed to supplement the details lost in the semantic branch, which consists of three \( 3 \times 3 \) convolutional layers and one \( 1 \times 1 \) point-wise convolutional layer. In order to obtain more features, we increase the number of channels of the second and third convolutional layers to 4 times (4 C) the original input. Finally, we use a \( 1 \times 1 \) convolutional layer to reduce the number of channels to C again. This operation can remove redundant features and extract effective features. In addition, to reduce the number of parameters and computational cost, we replace the later two \( 3 \times 3 \) convolutional layers with the depth-wise separable convolution. Therefore, our DRM can extract rich shallow spatial features with few parameters and computational cost.

3) Spatial Attention Module (SAM): Different from channel attention focusing on “what”, spatial attention pays more attention to “where”. Therefore, SAM is used to extract and preserve the shallow spatial information of the entire model. Specifically, the practice of spatial attention is to apply both max-pooling and average-pooling along the channel axis, then concatenate and convolve them by a standard convolution to generate an effective feature description. The process of spatial attention [40] can be described as follows:

\[ M_s(X) = \sigma(C_{k \times k} ([f_{\text{AvgPool}}(X); f_{\text{MaxPool}}(X)])), \]

where \( M_s \in \mathbb{R}^{1 \times H \times W} \) is the desired spatial attention map, \( X \in \mathbb{R}^{C \times H \times W} \) are the input features, \( C_{k \times k} \) represents the standard convolutional layer with kernel size \( k \), \([\cdot] \) means concatenation operation, \( f_{\text{AvgPool}}(\cdot) \) represents the average pooling operation, \( f_{\text{MaxPool}}(\cdot) \) represents the max pooling operation, and \( \sigma \) is the Sigmoid function.
D. Fast Bilateral Symmetrical Network (FBSNet)

The whole network structure can be divided into three parts: initial block, dual-branch backbone, and feature aggregation module (the symmetry is mainly reflected in the resolution of the encoding and decoding procedure in SIB). The complete network structure can be found in Fig. 3 and the detailed architectural configuration can be found in Table I.

1) Initial Block: The initial block includes three \(3 \times 3\) convolutional layers, where the stride of the first convolution is set to 2 to collect initial features. Different from Fast-SCNN [15], our initial block performs downsampling operation only once, thus the spatial information will be well preserved. In addition, in typical two-branches networks [11, 12], they often branched from the original input, where the feature information of the two branches is completely independent. In this paper, the initial block is used as the dividing point of two branches so that the semantic and spatial information are partially correlated, which facilitates subsequent feature merging.

2) Feature Aggregation Module (FAM): How to effectively integrate the information of the semantic branch and the spatial branch is also the key issue of the two-branch structure. Directly element-wise add or concatenate them are the most widely used methods. However, these methods ignore the differences between the features provided by the two branches. To solve this problem, we used the strategy motivated by the latest attention mechanism [53]. This method can not only capture cross-channel information but also grab information about direction and position perception. The most precious thing is that it is not computation-intensive, which means fewer parameters are in exchange for more gains, especially for dense predictions. Specifically, the global average pooling is factorized into a pair of 1D feature operations. Given an input \(X \in \mathbb{R}^{C \times H \times W}\), which is obtained by adding the two output features \(X_1 \in \mathbb{R}^{C \times H \times W}\), \(X_2 \in \mathbb{R}^{C \times H \times W}\) from the two branches. Along the horizontal and the vertical coordinates, we employ two spatial pooling kernels with size \((H, 1)\) and \((1, W)\) to encode the features, respectively. After that, we can get two transformations, which respectively merge features along the two spatial dimensions to correspondingly generate attention maps from two directions. One direction is responsible for capturing long-range dependencies, and the other helps to retain precise positional information. For the generated maps, we concatenate them and send them to a shared \(1 \times 1\) convolutional layer for channel reduction yielding a \(C/r \times 1 \times (W + H)\) output. The procedures can be formulated as follows:

\[
F = C_{1 \times 1} \left( \left[ f_{\text{AvgPool}}^H (X), f_{\text{AvgPool}}^W (X) \right] \right),
\]

where \([\cdot]\) represents the concatenation operation, \(f_{\text{AvgPool}}^H (\cdot)\) and \(f_{\text{AvgPool}}^W (\cdot)\) represent the pooling operation along the \(H\) and \(W\) directions. Batch normalization and non-linear activation function are also used before we split the intermediate features \(F\) into two tensors, \(F^h \in \mathbb{R}^{C/r \times H}\) and \(F^w \in \mathbb{R}^{C/r \times W}\), respectively. After that, two \(1 \times 1\) convolutional layers \(T_h\) and \(T_w\) are applied to restore \(F^h\) and \(F^w\) to have the same channel numbers as the input \(X\). After passing through two Sigmoid functions, we expand the two outputs and take them as attention weights.

The procedures can be formulated as follows:

\[
K^h = \delta (T_h (F^h)), \quad K^w = \delta (T_w (F^w)).
\]

Finally, the output \(Y\) of the FAM can be calculated as:

\[
Y = (X_1 + X_2) \times K^h \times K^w.
\]

Through this method, the features of the two branches can be fully integrated, and the feature information can be adaptively highlighted under the channel and spatial directions simultaneously. The complete structure of FAM is provided in Fig. 5(b).
IV. EXPERIMENTS

In this section, we will introduce the benchmarks datasets we used, the parameter settings during training, the ablation experiments, as well as the comparison with other state-of-the-art image semantic segmentation methods to fully illustrate the advantages of our FBSNet.

A. Datasets

1) Cityscapes: The Cityscapes dataset records street scenes in 50 different European cities with the size of $2048 \times 1024$, containing 5,000 finely annotated images and 19,998 coarsely annotated images. The dataset involves 30 categories, and we only employ 19 of them with fine-annotated images for training and testing, which is divided into three parts: 2975 samples for training, 500 samples for validation, and 1525 samples for testing.

2) CamVid: The CamVid dataset is another vehicle application dataset collected from video sequences. It has 11 categories and 701 finely annotated images that are split into 367 training samples, 101 validation samples, and 233 testing samples with the same size of $960 \times 720$.

B. Implementation Details

We implement our model with the PyTorch framework and all our experiments are performed on a single RTX 2080 Ti GPU. For Cityscapes, we train our model from scratch by utilizing cross-entropy loss function with OHEM, as well as the stochastic gradient descent (SGD) method with momentum 0.9 and the related weight decay.

The initial learning rate is configured as $1 \times 10^{-4}$, and the related weight decay $1 \times 10^{-4}$. As for the CamVid dataset, due to different input resolutions, we adapt the optimization method to Adam with momentum 0.9 and related weight decay $2 \times 10^{-4}$. At the same time, to stabilize the training, the ‘poly’ policy is also utilized for the learning rate strategy. The initial learning rate is configured as $4.5 \times 10^{-2}$ and $1 \times 10^{-3}$ for Cityscapes and CamVid, respectively. The learning rate is generally related to the iteration and can be calculated as $l_{\text{initial}} \times (1 - \frac{\text{iteration}}{\text{max} \_\text{iteration}})^{0.9}$. Both datasets are trained for 1000 epochs with a batch size of 4 for Cityscapes and 6 for CamVid.

C. Ablation Studies

The main purpose of the ablation studies conducted on the CamVid test set is to verify the performance improvement of each module in the entire network. Similar to the controlled variable methods in physics experiments, we add modules step by step to see their impact on the prediction results, the number of parameters, and the amount of computation. All results are provided in Table II.

1) CAM: As we can observe from the first two lines of the data field in Table II, the prediction results of the network are worse without using CAM. In other words, the CAM can provide a 0.6% improvement in accuracy with almost no increase in the number of parameters or computation. This experiment fully proves the effectiveness of CAM.

2) FAM: Feature fusion methods have always been the key research topic for multi-semantic aggregation. Among them, the “Add” and “Concatenate” operations are the most widely used methods. Therefore, we provide the comparison of “Add,” “Concatenate,” and FAM in Table II. According to the table, we can clearly observe that our FAM achieves the best performance of 68.58%, which is 0.71% and 0.32% better than that of the “Add” and “Concatenate” operations, respectively. Compared with the “Add” operation, FAM only increases negligible parameters (0.000556 M) and FLOPs (0.0003 G). This is entirely acceptable. Moreover, compared with the “Concatenate” operation, FAM achieves better results with fewer parameters and FLOPs. This is a big breakthrough that can effectively improve the performance of the model without increasing the complexity of the model. Comprehensive consideration of these indicators can prove the effectiveness of our proposed FAM.

3) SAM: This part is to verify the benefits of SAM in enhancing spatial information. According to Table II, we can see that SAM can take up only 0.01 G of computation and very few parameters, resulting in a performance improvement from 68.58% to 68.86%.

As can be seen from the above three ablation experiments, each module plays a unique role in the network. It can also be found that due to the addition of SDB, the computation of the whole network is increased by nearly a half while the performance is improved from 66.72% to 67.87%. The reason

| Models       | SIB | SDB | CAM | SAM | Fusion  | Parameters (M) | FLOPs (G) | mIoU (%) |
|--------------|-----|-----|-----|-----|---------|---------------|-----------|----------|
| (a) CAM      |     |     |     |     |         |               |           |          |
| FBSNet       | ✓   | ✓   |     |     | Add     | 0.592270      | 6.7433    | 66.13    |
| FBSNet       | ✓   | ✓   | ✓   |     | Concatenate | 0.592285      | 6.7434    | 66.72    |
| (b) FAM      |     |     |     |     |         |               |           |          |
| FBSNet       | ✓   | ✓   | ✓   | ✓   |         | 0.614765      | 9.6889    | 67.87    |
| FBSNet       | ✓   | ✓   | ✓   | ✓   | FAM     | 0.615093      | 9.7745    | 68.26    |
| FBSNet       | ✓   | ✓   | ✓   | ✓   |         | 0.615321      | 9.6892    | 68.58    |
| (c) SAM      |     |     |     |     |         |               |           |          |
| FBSNet       | ✓   | ✓   | ✓   | ✓   |         | 0.615321      | 9.6892    | 68.58    |
| FBSNet       | ✓   | ✓   | ✓   | ✓   | FAM     | 0.615419      | 9.7037    | 68.86    |
why many methods are reluctant to use it is that their semantic branches are already so computationally heavy that there is no room left to consider the boundary detail information. On the contrary, due to the lightweight structure of our SIB, we can integrate the SIB and SDB to achieve promising performance.

D. Comparison With SOTA Methods

During the training phase, we crop the size of original images and corresponding labels to $512 \times 1024$ for Cityscapes and $480 \times 360$ for CamVid as input to reduce the computational consumption and the GPU memory occupation. For testing, we use the original size of the datasets for inference. For a fair comparison, we do not employ any other post-process operations, like conditional random field (CRF). The evaluation indicators are mainly from four aspects: the mean intersection over union (mIoU), the number of parameters, the number of float-point operations (FLOPs), and the inference speed (FPS).

1) Results on Cityscapes: In Table III, we provide a quantitative comparison with other state-of-the-art image semantic segmentation methods on the Cityscapes test dataset. Moreover, we provide the per-class IoU (%) results on the Cityscapes in Table IV. According to these results, we can observe that when encountered with fewer parameters, our network can well ensure better accuracy and faster inference speed. The models with the same number of parameters as our method do not have the same effects, and the models with the same effects will need more parameters than ours. Specifically, from the perspective of "Parameters," ENet [7], ESPNet [8], and NDNet [46] have fewer parameters but their segmentation accuracy is 5.6% lower than our FBSNet, which is a big gap in the segmentation field. This is in the case that our number of parameters is only 0.26 M more than theirs. From the perspective of "FLOPs," the
computation cost of ESPNet-v2 [47] is 7 G smaller than ours at the cost of 4.7% segmentation performance penalty. From the perspective of “Speed,” our method is in a position above the medium level at 90 FPS. While in terms of “mIoU,” there are several methods have higher performance than our method, such as DFANet [13] and BiseNet-v2 [12]. Indeed, DFANet [13] is an outstanding model, which achieves slightly better mIoU result than our FBSNet. However, we should not ignore that the number of parameters of DFANet is 7.8 M (M = Million), while the number of ours model is 0.62 M, which is only 1/12 of DFANet. Such a large-scale DFANet is difficult to be promoted in practical applications. As for the inference speed, the results of the two models are very close (DFANet: 100 vs. FBSNet: 90). The reason why FBSNet has fewer parameters but slower inference speed is that the attention mechanisms are used in FBSNet. These attention mechanisms will bring some computational overhead, resulting in slower inference speed. Fortunately, this effect is acceptable. To better compare these models, we provide the Accuracy-Speed-Parameters comparisons in Fig. 1. Obviously, our FBSNet achieves the best trade-off between the accuracy, model size, and inference speed of the model. Moreover, we provide the visual comparisons with LEDNet [28], DABNet [30], ERFNet [26], NDNet [46], and ENet [7] in Fig. 6. Through a horizontal comparison of the segmentation results, we can also qualitatively see the advantages of our proposed method. The yellow dotted boxes in the ground truth distinctly depict that our method is obviously superior to other methods.

2) Results on CamVid: To further verify the performance of FBSNet, we also provide a quantitative comparison with other methods on the CamVid test dataset in Table V. According to the table, we can observe that (a) FBSNet achieves the second-best segmentation results, which is slightly worse than MSCFNet. However, it should be noticed that FBSNet requires only half the amount of parameters and FLOPs of MSCFNet, and its inference speed is twice that of MSCFNet; (b) Compared with methods of similar model size, the performance of FBSNet has been significantly improved; (c) Compared with other large models, FBSNet achieves better results with fewer parameters.

| Methods      | Input Size | Backbone | Parameters (M) | mIoU (%) |
|--------------|------------|----------|----------------|----------|
| ENet [5]     | 360 × 480  | No       | 0.36           | 51.3     |
| SegNet [27]  | 360 × 480  | VGG16    | 29.50          | 55.6     |
| NDN [42]     | 360 × 480  | No       | 0.50           | 57.2     |
| DFANet [11]  | 720 × 960  | Xception | 7.80           | 64.7     |
| DilationNet [50] | 720 × 960 | VGG16 | 14.80          | 65.3     |
| CGNet [17]   | 360 × 480  | No       | 0.50           | 65.6     |
| BiseNet [9]  | 720 × 960  | Xception39 | 5.80          | 65.6     |
| DABNet [26]  | 360 × 480  | No       | 0.76           | 66.4     |
| FDDWN [48]   | 360 × 480  | No       | 0.80           | 66.9     |
| ICNet [7]    | 720 × 960  | PSPNet50 | 26.50          | 67.1     |
| BCPNet [46]  | 720 × 720  | No       | 0.61           | 67.8     |
| BiseNet-v2 [10] | 720 × 960 | ResNet18 | 49.00          | 68.7     |
| MSCFNet [2]  | 360 × 480  | No       | 1.15           | 69.3     |
| FBSNet (ours)| 360 × 480  | No       | 0.62           | 68.9     |

Similarly, we provide the visual comparison results of these methods on the CamVid test dataset in Fig. 7. Obviously, our FBSNet still achieves the best segmentation results. All the above experiments have fully proved the effectiveness and excellence of FBSNet, which strikes a good balance between accuracy and efficiency.
V. CONCLUSION

In this paper, we proposed a Fast Bilateral Symmetrical Network (FBSNet) for real-time image semantic segmentation. Compared to some top-accurate methods struggling for complex architecture and extreme lightweight networks aiming at compressing convolutional layers, our solution mainly focuses on getting a relatively good balance between the accuracy, model size, and inference speed with elaborately designed modules. Especially, our proposed Bottleneck Residual Unit (BRU) can enlarge the receptive field to distill significant semantic information comprehensively. Our extensive investigations have shown that spatial information is critical to recovering the resolution during the decoder process. Therefore, we designed a symmetrical encoder-decoder model, which is composed of a Semantic Information Branch (SIB) and a Spatial Detail Branch (SDB). These two branches can extract deep semantic information and preserve shallow boundary details, respectively. In summary, our model, which is only 0.62 M, can process a $512 \times 1024$ input size and achieves 70.9% at a speed of 90 FPS on Cityscapes test data with a single RTX 2080TI GPU. Extensive experiments have revealed that our proposed FBSNet achieved a good balance between the accuracy and efficiency of the model.

REFERENCES

[1] Y. Li, Y. Guo, Y. Kao, and R. He, “Image piece learning for weakly supervised semantic segmentation,” IEEE Trans. Syst., Man, Cybern. Syst., vol. 47, no. 4, pp. 648–659, Apr. 2017.
[2] T. Zhang et al., “Decoupled spatial neural attention for weakly supervised semantic segmentation,” IEEE Trans. Multimedia, vol. 21, no. 11, pp. 2930–2941, Nov. 2019.
[3] G. Gao et al., “MSCFNet: A lightweight network with multi-scale context fusion for real-time semantic segmentation,” IEEE Trans. Intell. Transp. Syst., to be published. doi: 10.1109/TITS.2021.3098355.
[4] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” in Proc. IEEE Int. Conf. Learn. Represent., 2015, pp. 1–14.
[5] 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.
[6] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. Alemi, “Inception-v4, inception-resnet and the impact of residual connections on learning,” in Proc. AAAI Conf. Artif. Intell., 2017, pp. 4278–4284.
[7] A. Paszke, A. Chaurasia, S. Kim, and E. Culurciello, “ENet: A deep neural network architecture for real-time semantic segmentation,” in Proc. IEEE Int. Conf. Learn. Represent., 2017, pp. 1–10.
[8] S. Mehta, M. Rastegari, A. Caspi, L. Shapiro, and H. Hajishirzi, “ESPNet: Efficient spatial pyramid of dilated convolutions for semantic segmentation,” in Proc. Eur. Conf. Comput. Vis., 2018, pp. 552–568.
[9] H. Zhao, X. Qi, X. Shen, J. Shi, and J. Jia, “ICNet for real-time semantic segmentation on high-resolution images,” in Proc. Eur. Conf. Comput. Vis., 2018, pp. 405–420.
[10] 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. 2881–2890.
[11] C. Yu et al., “BiSeNet: Bilateral segmentation network for real-time semantic segmentation,” in Proc. Eur. Conf. Comput. Vis., 2018, pp. 325–341.
[12] C. Yu et al., “BiSeNet V2: Bilateral network with guided aggregation for real-time semantic segmentation,” Int. J. Comput. Vis., vol. 129, pp. 3051–3068, Sep., 2021.
[13] H. Li, P. Xiong, H. Fan, and J. Sun, “DFANet: Deep feature aggregation for real-time semantic segmentation,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2019, pp. 9522–9531.
[14] F. Chollet, “Xception: Deep learning with depthwise separable convolutions,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2017, pp. 1251–1258.
[15] R. P. Poudel, S. Liwicki, and R. Cipolla, “Fast-SCNN: Fast semantic segmentation network,” in Proc. Brit. Mach. Vis. Conf., 2019, pp. 1–12.
[16] G. Gao et al., “Learning robust and discriminative low-rank representations for face recognition with occlusion,” Pattern Recognit., vol. 66, pp. 129–143, 2017.
[17] G. Gao et al., “Constructing multilayer locality-constrained matrix regression framework for noise robust face super-resolution,” Pattern Recognit., vol. 110, 2021, Art. no. 107539.
[18] G. Gao, Y. Yu, J. Yang, G.-J. Qi, and M. Yang, “Hierarchical deep CNN feature set-based representation learning for robust cross-resolution face recognition,” IEEE Trans. Circuits Syst. Video Technol., to be published, doi: 10.1109/TCSVT.2020.3042178.
[19] X. Zhang et al., “Fast semantic segmentation for scene perception,” IEEE Trans. Ind. Inform., vol. 15, no. 2, pp. 1183–1192, Feb. 2019.
[20] T. Wu, S. Tang, R. Zhang, J. Cao, and Y. Zhang, “CGNet: A lightweight context guided network for semantic segmentation,” IEEE Trans. Image Process., vol. 30, pp. 1169–1179, 2020.
[21] W. Liu, A. Rabinovich, and A. C. Berg, “Parsenet: Looking wider to see better,” in Proc. IEEE Int. Conf. Learn. Representation Workshop, 2016, pp. 1–11.
[22] M. Yang, K. Yu, C. Zhang, Z. Li, and K. Yang, “Denseaspp for semantic segmentation in street scenes,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2018, pp. 3684–3692.
[23] 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. 2017.
[24] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam, “Rethinking atrous convolution for semantic image segmentation,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2017, pp. 1–14.
[25] C. Yu et al., “Learning a discriminative feature network for semantic segmentation,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2018, pp. 1857–1866.
[26] E. Romera, J. M. Alvarez, L. M. Bergasa, and R. Arroyo, “ERFNet: Efficient residual factorized convnet for real-time semantic segmentation,” IEEE Trans. Intell. Transp. Syst., vol. 19, no. 1, pp. 263–272, Jan. 2017.
[27] E. Shelhamer, J. Long, and T. Darrell, “Fully convolutional networks for semantic segmentation,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 39, no. 4, pp. 640–651, Apr. 2017.
[28] Y. Wang et al., “U-Net: A lightweight encoder-decoder network for real-time semantic segmentation,” in Proc. IEEE Int. Conf. Image Process., 2019, pp. 1860–1864.
[29] O. Ronneberger, P. Fischer, and T. Brox, “U-Net: Convolutional networks for biomedical image segmentation,” in Proc. Med. Image Comput. Comput. Assist. Interv., 2015, pp. 234–241.
[30] G. Li, I. Yun, J. Kim, and J. Kim, “DabNet: Depth-wise asymmetric bottleneck for real-time semantic segmentation,” in Proc. Brit. Mach. Vis. Conf., 2019, pp. 1–12.
[31] V. Badrinarayanan, A. Kendall, and R. Cipolla, “SegNet: A deep convolutional encoder-decoder architecture for image segmentation,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 39, no. 12, pp. 2481–2495, Dec. 2017.
[32] S. Ioffe and C. Szegedy, “Batch normalization: Accelerating deep network training by reducing internal covariate shift,” in Proc. Int. Conf. Mach. Learn., 2015, pp. 448–456.
[33] V. Nair and G. E. Hinton, “Rectified linear units improve restricted Boltzmann machines,” in Proc. Int. Conf. Mach. Learn., 2010, pp. 1–8.
[34] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, “Rethinking the inception architecture for computer vision,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2016, pp. 2818–2826.
[35] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, “Mobilenetv3: Inverted residuals and linear bottlenecks,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2018, pp. 4510–4520.
[36] A. Howard et al., “Searching for mobilenetv3,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2019, pp. 1314–1324.
[37] X. Zhang, X. Zhou, M. Lin, and J. Sun, “ShuffleNet: An extremely efficient convolutional neural network for mobile devices,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2018, pp. 6848–6856.
[38] N. Ma, X. Zhang, H.-T. Zheng, and J. Sun, “ShuffleNetV2: Practical guidelines for efficient CNN architecture design,” in Proc. Eur. Conf. Comput. Vis., 2018, pp. 116–131.
[39] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “ImageNet classification with deep convolutional neural networks,” in Proc. Adv. Int. Conf. Neural Inf. Process. Syst., 2012, pp. 1097–1105.
et al. received the Ph.D. degree in computer science and technology from East China Normal University, Shanghai, China, in 2021. He is currently a Postdoctoral Fellow with the Center for Mathematical Artificial Intelligence, The Chinese University of Hong Kong, Hong Kong. He has authored or coauthored more than 20 scientific papers in IEEE TRANSACTIONS ON IMAGE PROCESSING, IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING, IEEE TRANSACTIONS ON MULTIMEDIA, IEEE TRANSACTIONS ON CYBERNETICS, IEEE TRANSACTIONS ON VISUAL COMMUNICATION AND IMAGE PROCESSING, IEEE TRANSACTIONS ON ROBOTICS, IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, and IEEE TRANSACTIONS ON AUTOMATION SCIENCE AND ENGINEERING. He was the recipient of the Best Paper Award from the IEEE ISM 2012, 2nd prize in Yahoo Flickr Grand Challenge 2015, were in the top winners (out of 29 teams) from ACM SIGSPATIAL GIS Cup 2013, and Best Paper Runner-Up in APWeb-WAIM 2017, recognized as finalist of the World's FIRST 10 K Best Paper Award in ICME 2017.

Yi Yu (Senior Member, IEEE) received the Ph.D. degree in information and computer science from Nara Women’s University, Nara, Japan. She is currently an Assistant Professor with the National Institute of Informatics (NII), Tokyo, Japan. Before joining NII, she was a Senior Research Fellow with the School of Computing, National University of Singapore, Singapore. Her main research interests include machine learning and deep learning for multimedia understanding and knowledge discovery. She and her team was the recipient of the Best Paper Award from the IEEE ISM 2012, 2nd prize in Yahoo Flickr Grand Challenge 2015, were in the top winners (out of 29 teams) from ACM SIGSPATIAL GIS Cup 2013, and Best Paper Runner-Up in APWeb-WAIM 2017, recognized as finalist of the World’s FIRST 10 K Best Paper Award in ICME 2017.

Huimin Lu (Senior Member, IEEE) received the B.S. degree in electronics information science and technology from Yangzhou University, Yangzhou, China, in 2008, the M.S. degree in electrical engineering from the Kyushu Institute of Technology, Kitakyushu, Japan, and Yangzhou University, in 2011, and the Ph.D. degree in electrical engineering from the Kyushu Institute of Technology, in 2014. From 2013 to 2016, he was a JSPS Research Fellow (DC2, PD, and PD) with the Kyushu Institute of Technology. He is currently an Assistant Professor with the Kyushu Institute of Technology and an Excellent Young Researcher with MEXT-Japan. His research interests include computer vision, robotics, artificial intelligence, and ocean observing.

Jian Yang (Member, IEEE) received the Ph.D. degree from the Nanjing University of Science and Technology (NUST), Nanjing, China, on the subject of pattern recognition and intelligence systems in 2002. In 2003, he was a Postdoctoral Researcher with the University of Zaragoza, Zaragoza, Spain. From 2004 to 2006, he was a Postdoctoral Fellow with Biometrics Centre, The Hong Kong Polytechnic University, Hong Kong. From 2006 to 2007, he was a Postdoctoral Fellow with the Department of Computer Science, New Jersey Institute of Technology, Newark, NJ, USA. He is currently a Chang-Jiang Professor with the School of Computer Science and Engineering, NUST. He is the Author of more than 100 scientific papers in pattern recognition and computer vision. His papers have been cited more than 9000 times in the Web of Science, and 9000 times in the Scholar Google. His research interests include pattern recognition, computer vision and machine learning. He is/was an Associate Editor for the Pattern Recognition Letters, IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, IEEE TRANSACTIONS ON COMPUTATIONAL INTELLIGENCE AND AI, and IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS, and Neurocomputing. He is a Fellow of IAPR.

Authorized licensed use limited to: Nanjing Univ of Post & Telecommunications. Downloaded on August 09,2023 at 07:36:48 UTC from IEEE Xplora. Restrictions apply.