LMFFNet: A Well-Balanced Lightweight Network for Fast and Accurate Semantic Segmentation

Min Shi, Jialin Shen, Qingming Yi, Jian Weng, *Member, IEEE*, Zunkai Huang, Aiwen Luo, *Member, IEEE*, and Yicong Zhou, *Senior Member, IEEE*

***Abstract*— Real-time semantic segmentation is widely used in autonomous driving and robotics. Most previous networks achieved great accuracy based on a complicated model involving mass computing. The existing lightweight networks generally reduce the parameter sizes by sacrificing the segmentation accuracy. It is critical to balance the parameters and accuracy for real-time semantic segmentation. In this article, we propose a lightweight multiscale-feature-fusion network (LMFFNet) mainly composed of three types of components: split-extract-merge bottleneck (SEM-B) block, feature fusion module (FFM), and multiscale attention decoder (MAD), where the SEM-B block extracts sufficient features with fewer parameters. FFMs fuse multiscale semantic features to effectively improve the segmentation accuracy and the MAD well recovers the details of the input images through the attention mechanism. Without pretraining, LMFFNet-3-8 achieves 75.1% mean intersection over union (mIoU) with 1.4 M parameters at 118.9 frames/s using RTX 3090 GPU. More experiments are investigated extensively on various resolutions on other three datasets of CamVid, KITTI, and WildDash2. The experiments verify that the proposed LMFFNet model makes a decent tradeoff between segmentation accuracy and inference speed for real-time tasks. The source code is publicly available at https://github.com/Greak-1124/LMFFNet.**

***Index Terms*— Fast semantic segmentation, lightweight network, multiscale attention decoder (MAD), multiscale feature fusion, split-extract-merge bottleneck (SEM-B).**

I. Introduction

EDGE computing on mobile phones, automotive systems, wearable devices, the Internet of Things (IoT) devices, and so on evolves to be a new computing paradigm and becomes more reliable and practical for real-time tasks executed with extremely low transmission latency based on local data sources. Nevertheless, restricted computing resources and battery capacity of edge devices bring new challenges to real-time applications as well. Lightweight network architecture with an appropriate tradeoff between accuracy and inference speed becomes a challenging task for real-time application scenarios, especially the semantic segmentation tasks that receive significant attention on many important applications such as autonomous driving and robotics in the current decade.

The deployment of artificial neural networks (ANNs) for real-time semantic image segmentation is mostly constrained by: 1) accuracy; 2) model size; and 3) inference speed. Nevertheless, ANNs are generally built toward a primary trend for achieving higher accuracy by employing deeper convolutional layers and larger feature channels. The high-quality results largely rely on sophisticated models that involve mass computing operations. Most previous deep neural networks (DNNs) mainly focused on improving the model accuracy regardless of computational efficiency. For example, in semantic segmentation, PSPNet [1] introduced a pyramid pooling module (PPM) achieving 80.2% mean intersection over union (mIoU) with 65.7 million parameters on Cityscapes [2] test set. DeepLabV3+ [3] achieved a better performance of 82.1% mIoU with 54.6 million parameters resulting mainly from the "atrous convolution." DRANet [4] introduced both spatial attention and channel attention in the network, which reached 82.9% mIoU, while the reasoning speed was far less than the real-time standard. Besides, to capture more spatial details, high-resolution images are employed in various tasks based on DNNs, which brings higher computational cost.

Therefore, accurate networks built by existing approaches usually request support from powerful computing platforms with rich hardware resources since the large model sizes always involve enormous parameters and computing operations. In other words, most DNN models that only concern accuracy are unlikely to enable practical tasks on resource-constrained edge devices such as mobile phones. The requirements, such as timing, power consumption, and reliability, are also extremely important to practical applications, especially to the autonomous driving scene.

Many lightweight-oriented technologies are developed currently to reduce the model size and improve the inference speed. Li *et al.* [39] provided an overview of impressive achievements made by various CNN models and surveyed some methods to refine CNNs for consuming fewer computing

Manuscript received 14 June 2021; revised 26 February 2022; accepted 10 May 2022. Date of publication 27 May 2022; date of current version 2 June 2023. This work was supported in part by the National Natural Science Foundation of China (NSFC) under Grant 62002134; in part by the Guangdong Basic and Applied Basic Research Foundation under Grant 2020A1515110645; in part by the Fundamental Research Funds for the Central Universities at Jinan University under Grant 21620353; in part by the Science and Technology Development Fund, Macau, under Grant 189/2017/A3; and in part by the University of Macau under Grant MYRG2018-00136-FST. *(Min Shi and Jialin Shen contributed equally to this work.) (Corresponding author: Aiwen Luo.)*

Min Shi, Jialin Shen, Qingming Yi, and Jian Weng are with the Department of Electronic Engineering, College of Information Science and Technology, Jinan University, Guangzhou 510632, China.

Zunkai Huang is with the Shanghai Advanced Research Institute, Chinese Academy of Sciences, Shanghai 201210, China.

Aiwen Luo is with the Department of Electronic Engineering, College of Information Science and Technology, Jinan University, Guangzhou 510632, China, and also with the Department of Computer and Information Science, University of Macau, Macau 999078, China (e-mail: faith.awluo@gmail.com).

Yicong Zhou is with the Department of Computer and Information Science, University of Macau, Macau 999078, China.

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TNNLS.2022.3176493.

Digital Object Identifier 10.1109/TNNLS.2022.3176493

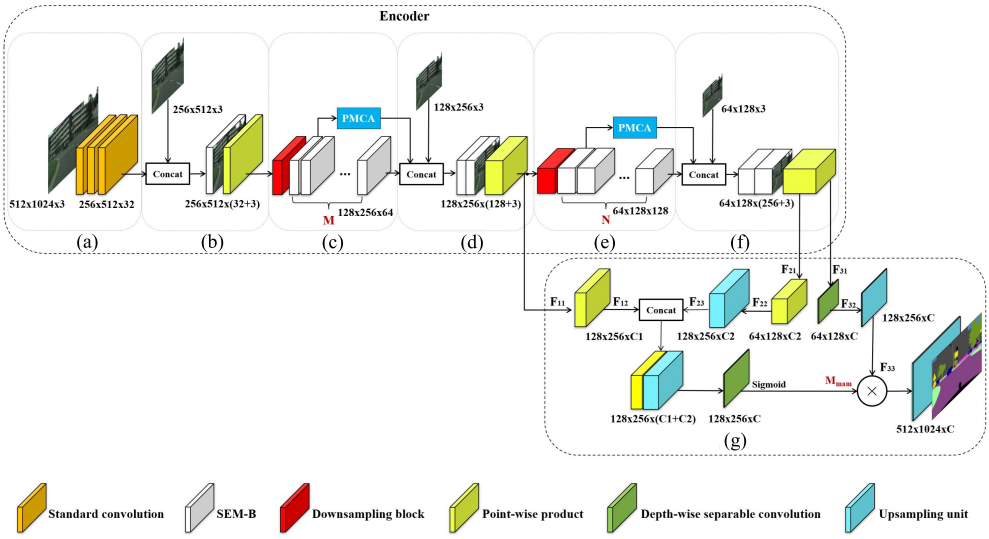

Fig. 1. Architecture of the proposed LMFFNet. M and N denote the number of SEM-B in SEM-B Block1 and SEM-B Block2, respectively. $C1$ and $C2$ represent the number of output channels of FFM-B1 and FFM-B2 after 1×1 convolution operation, respectively. C denotes the number of categories. PMCA: partition-merge channel attention module. (a) Initial block. (b) FFM-A1. (c) SEM-B block1. (d) FFM-B1. (e) SEM-B block2. (f) FFM-B2. (g) MAD (decoder).

resources, such as low-rank approximation, network pruning, and network quantization. Pooling technology can attain fast processing speed for the high-resolution input images by cutting down the resolution of feature maps. However, the pooling operation generally leads to the loss of image spatial information. Thus, many existing approaches [3], [5], [6] tried to replace the later pooling layers by dilated convolutions, obtaining a larger receptive field with less loss of spatial information.

Besides, convolution factorization [7], depthwise separable convolution (ds-Conv) [8], and attention mechanism [9]–[12] are often used to reduce the parameter number of semantic segmentation networks. For instance, BiSeNet [12], [40] introduced channel attention into the attention refinement module (ARM) to aggregate the feature maps, and CGNet [9] used the same attention technology to construct context-guided (CG) block for feature extraction. LRNNet [10] used a self-attention mechanism to construct a lightweight decoder to reduce the model parameters. In this work, we propose a novel semantic segmentation structure called lightweight multiscale-feature-fusion network (LMFFNet) whose overall architecture is shown in Fig. 1. Compared with the existing methods, our main contributions are summarized as follows.

1) A novel module of split-extract-merge bottleneck (SEM-B), which combines standard convolution with ds-Conv, is exploited in an efficient way for building a lightweight backbone.
2) A novel channel attention module with partition-merge channel attention (PMCA) mechanism is proposed for improving the feature fusion ability of the feature fusion module (FFM) module.
3) FFMs are further developed and inserted in the backbone to capture multiscale features, fusing the context information between feature maps of adjacent layers and combining multiscale semantic information from different depths.
4) A lightweight multiscale attention decoder (MAD) constructed on an attention mechanism using only 0.09 M parameters is introduced to process multiscale features and recover the spatial details efficiently.
5) The SEM-Bs and FFMs in the backbone are united in the encoder phase, while the MAD dominates in the decoder phase for constructing the overall architecture of LMFFNet based on the encoder–decoder strategy, achieving an adequate tradeoff between accuracy and efficiency for edge computing.

Extensive ablation experiments are implemented progressively to investigate the impacts of each core component of the LMFFNet model on the accuracy–efficiency tradeoff (AET) in this article.

The rest of this article is organized as follows. Section II gives an overview of related work for real-time semantic segmentation. Section III introduces the key components and the overall architecture of the proposed semantic segmentation model of LMFFNet in detail. A series of ablation experiments is implemented to verify the effectiveness of our networks in Section IV. Finally, Section V concludes this article.

II. Related Work

In this section, we summarize related methods for real-time semantic segmentation based on the encoder–decoder architecture. To improve the accuracy of semantic segmentation, the encoder usually needs to make full use of the context information to improve the classification accuracy, while the decoder needs to effectively recover the spatial information to improve the segmentation accuracy.

A. Existing Methods for Encoder

The current design of the real-time semantic segmentation encoder can be divided into two streams. The first one introduces the multibranch structure or connects the shallow features and deep features based on the existing backbone networks (e.g., ResNet [13] and Xception [14]). For example, SwiftNet [15] modified ResNet and introduced a light-aggregation encoder deploying reversed subpixel and showcased a pixel-adaptive memory strategy for

real-time semisupervised video object segmentation, achieving 70 frames/s and state-of-the-art accuracy on the DAVIS 2017 validation dataset with pretraining. ICNet [16] proposed a multipath cascade structure based on ResNet, which captured more context information and thus greatly improved the classification accuracy. Yoltrack [17] built a segmentation network applied to the field of autopilot based on ShuffleNet v2 [18] as the backbone network. The C-DCNN [19] used VGG16 [20] as the backbone network to improve the accuracy by introducing depth image information to generate high-order features. Nevertheless, although the encoders based on the existing backbone networks gain strong feature extraction ability, it always includes large network parameters and thus is generally unaffordable for mobile scenarios using edge devices with limited computational resources.

The second one proposes lightweight modules to build an efficient backbone. ENet [21] is one remarkable lightweight network for real-time semantic segmentation applied in practical scenarios. To make an available tradeoff between accuracy and efficiency, ENet constructed a lightweight bottleneck by employing asymmetric convolution. Referring to the bottleneck of ResNet, ERFNet [22] employed factorized convolutions (1×3, 3×1) to replace the 3×3 convolution and proposed nonbottleneck-1D (Non-bt-1D), greatly reducing the number of parameters. LEDNet [23] proposed a split-shuffle-nonbottleneck that introduced split-shuffle operations to reduce the number of parameters and improved the accuracy. Convolution factorization was also employed in [23], improving the accuracy with a few parameters. ESNet [24] adopted three branches in a parallel factorized convolution unit (PFCU). Each branch in [24] corresponds to a convolution with different dilated rates, which aggregated multiscale context information. The above networks are usually lighter in weight (i.e., fewer model parameters), but the achieved accuracy is not satisfactory compared with the former approaches employed by existing backbone networks. In this article, our LMFFNet can keep high accuracy, but the parameters of its backbone network constructed by SEM-Bs are much smaller than those based on existing networks, such as ResNet and Xception.

B. Existing Methods for Decoder

The decoder could be divided into two types: symmetric decoder and asymmetric decoder. SegNet [25] introduced the pooling indices to build the upsampling module to construct the symmetric decoder. ERFNet [22] and ESNet [24] used Non-bt-1D and PFCU to build symmetric decoders, respectively. The symmetric decoder could recover spatial information finely and then improve the model accuracy. However, it usually includes multiple feature extractions and upsampling steps due to the symmetry between encoder and decoder, leading to complex computation and slow inference speed. In other words, the symmetric encoder–decoder-based architectures typically cannot provide a decent tradeoff between inference speed and accuracy.

The asymmetric decoder is frequently used in recent works since the asymmetric decoder is more suitable for real-time semantic segmentation due to its better tradeoff between accuracy and computing efficiency with fewer parameters and faster inference speed. DFANet [26] designed a simple asymmetric decoding module that aggregated the feature information of each layer of the encoder and efficiently recovered the spatial information. DABNet [11] and LRNNet [10] proposed the efficient reduced nonlocal module and pointwise aggregation decoder (PAD) based on the attention mechanism to achieve better spatial information recovery with smaller parameters. AGLNet [45] with 1.12 M parameters employed two types of attention mechanisms subsequently in the decoder to upsample feature maps to match input resolution and achieved maximal 71.3% mIoU. Peng *et al.* [46] developed a lightweight decoder using bilateral spatial–channel attention to combine the high- and low-level feature maps without changing of backbone network, obtaining 75.9% mIoU and 38.5-frames/s inference speed on the Cityscapes test dataset and 72.9% mIoU and 254.7-frames/s speed on the CamVid test dataset. However, its parameter amount and FLOPs were unrevealed. MSCFNet [47] employed a two-step decoder that is asymmetrical relative to the encoder by using a $\times 4$ upsampling and a final $\times 2$ deconvolution operation to restore the image size, resulting in 1.15 M parameters. Likewise, LRDNet [48] proposed a refined dual-attention decoder to reduce the complexity of the model and improve the semantic segmentation accuracy, resulting in 0.66 M parameters. Inspired by these works, we explore a new attention mechanism to build an ultralightweight decoder and recover spatial details efficiently in our proposed LMFFNet.

C. Real-Time Semantic Segmentation Networks

There are fast-growing network models designed for the real-time semantic segmentation task in practical application scenarios in recent years. The core issue for real-time semantic segmentation task turns into how to strike a good tradeoff between accuracy and inference speed.

It is a primary trend to design a lightweight and efficient network for real-time applications. For this purpose, Sem2Ins [41] leveraged a lightweight generator based on conditional generative adversarial networks (cGANs), least-squares loss, deep supervision, and weighted fusion, reaching up to 40-frames/s inference speed. However, the accuracy and inference speed for the instance segmentation task combined the Sem2Ins with other modules leaved much to be desired. WFDCNet [43] introduced a front-end network and designed a lightweight encoder, which is mainly composed of a full-dimensional continuous separation (FCS) convolution module and lateral asymmetric pyramid fusion (LAPF) module, resulting in 0.5 M parameters. STDC network [42] proposed an efficient lightweight backbone to provide scalable receptive field and set a single path decoder using detail information guidance to learn the low-level details. A satisfactory result of 250.4-frames/s inference speed and 71.9% mIoU, concerning the results of Cityscapes dataset, was presented in the STDC network [42]. A patchwise hypernetwork named HyperSeg [44] was constructed by an encoder with a nested U-Net [54] and a decoder consisted of dynamic blocks with spatially varying weights.

In the last two years, more lightweight networks were designed and achieved better performance in terms of the

tradeoff between inference speed and accuracy. MSCFNet [47] applied a simple deconvolution in the decoder and explored an efficient asymmetric residual (EAR) module in the encoder, achieving 71.9% mIoU and 50 frames/s on the Cityscapes testing set. Zhuang *et al.* [48] proposed an asymmetric and refined encoder–decoder model named LRDNet, which improved the feature extraction efficiency and reduced the boundary loss with low parameters by adopting decomposition convolution, deep convolution, and dual-attention mechanism, obtaining up to 77 frames/s and 70.1% mIoU on the Cityscapes test set. EFRNet [49] adopted a single branch strategy, which combined feature fusion for multistage features with a lightweight channel attention refinement, achieving 70.02% mIoU and 50.2 frames/s with 0.48 M parameters on CamVid. The LAANet [50] solved the intraclass inconsistency and interclass indistinction problems and achieved 73.6% mIoU and 95.8 frames/s on the Cityscapes.

To enable the feature extraction capability and detail recovery capability of high-resolution remote sensing images in the land cover segmentation algorithm, Pang *et al.* [51] proposed a semantic-guided bottleneck network (SGBNet) to balance accuracy and reasoning speed using a two-branch architecture. Zhang *et al.* [52] proposed an asymmetric network called LEANet, which only consumed 0.74 M parameters. LEANet [51] could process images at 98.6 frames/s inference speed and achieved the accuracy of 67.5% mIoU on the CamVid test set, while 71.9% mIoU and 77.3 frames/s were gained on the Cityscapes test set. Likewise, Liu *et al.* [53] tried to extract both local and contextual information by building up a RELAXNet with 1.9 M parameters, using residual modules combined of depthwise convolution, dilated convolution, factorized convolution, and channel shuffle. RELAXNet [52] achieved 74.8%-mIoU and 64-frames/s inference speed on the Cityscapes dataset and 71.2%-mIoU and 79-frames/s speed on the CamVid dataset. Note that the inference speeds of different models are closely related to the size of testing images and the employed GPU types.

For achieving real-time performance with a decent tradeoff between accuracy and inference speed, we dedicate to explore an elaborate encoder–decoder architecture for semantic segmentation tasks with around 1 M parameters by employing a lightweight encoder based on an efficient backbone and an asymmetric decoder with a relatively small size in this work.

III. LMFFNet

Based on the asymmetric encoder–decoder strategy, we elaborately design an LMFFNet for real-time semantic segmentation. SEM-B and FFM are two main types of components that compose the LMFFNet backbone. The SEM-B block consists of a variable number of SEM-Bs in different layers of the LMFFNet architecture. Moreover, a lightweight decoder, namely MAD, is developed in this work, consuming only 0.09 M parameters. The overall architecture of LMFFNet is shown in Fig. 1 and listed in Table I.

A. Split-Extract-Merge Bottleneck

Multiple successful instances of lightweight residual layers have been witnessed in recent years. ResNet [13] proposes a widely used residual structure as shown in Fig. 2(a) for real-time semantic segmentation. LEDNet [23] introduces split-shuffle-nonbottleneck (SS-nbt) [Fig. 2(b)], which adopts the split-transform-merge strategy. In the SS-nbt, convolution factorization is employed in two feature extraction branches. However, the factorized convolution is not conductive enough to the GPU parallel processing, which leads to a slow inference speed of the model under the same computational budget. Similar works in [10], [11], and [24] use the split-transform-merge strategy to build a feature extraction module as well.

TABLE I
OVERALL ARCHITECTURE OF LMFFNET

Module	Operator	Mode	Channel	Output size
Initial Block	3×3 Conv 3×3 Conv 3×3 Conv	Stride2 Stride1 Stride1	32 32 32	256×512 256×512 256×512
FFM-A1	Concatenation, 1×1 Conv	-	32+3	256×512
Downsampling	Downsampling	-	64	128×256
SEM-B Block1	SEM-B×M	Dilated2	64	128×256
FFM-B1	Concatenation, 1×1 Conv	-	128+3	128×256
Downsampling	Downsampling	-	128	64×128
SEM-B Block2	SEM-B×N	Dilated {4,8,16,...,}	128	64×128
FFM-B2	Concatenation, 1×1 Conv	-	256+3	64×128
MAD	-	-	C *	512×1024

* C is the number of predicted classes.

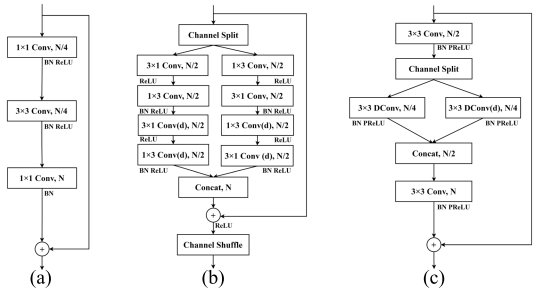

Fig. 2. Comparison of different bottleneck structures. (a) ResNet bottleneck. (b) SS-nbt of LEDNet. (c) Our SEM-B. Note that "Conv" denotes the standard convolution, "Conv(d)" denotes dilated convolution with "d," and "DConv(d)" indicates depthwise dilated convolution with dilated "d." "*N*" refers to the number of feature channels.

Inspired by the above residual structures, we propose a new bottleneck referred to SEM-B in our LMFFNet, as shown in Fig. 2(c). To improve the computing efficiency, we build up our bottleneck with both standard convolution and ds-Conv. Besides, the activation function scheme is very important to convolution operation. PReLU and batch normalization (BN) are both applied in this work before every convolution operation in the SEM-B because the PReLU normally performs better than ReLU in lightweight networks [27] and BN helps to increase the convergence speed of the model [28].

At the beginning of each SEM-B, a standard 3×3 convolution is employed to generate features and halves the number of input channels. The convolutional output is then split into two branches, where each one has 1/4 channels of the input. To avoid convolution factorization, the transformation is performed in the two branches, individually using depthwise convolution and depthwise dilated convolution. The convolutional outputs of the two branches are merged through concatenation so as to obtain multiscale feature information and restore the

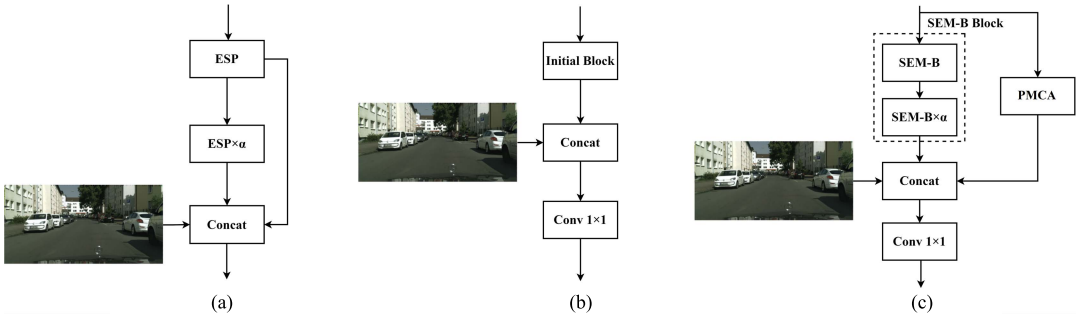

Fig. 3. Comparison of different FFMs. (a) Skip connection of ESPNet. (b) Proposed FFM-A. (c) Proposed FFM-B. Note that "ESP" is the efficient spatial pyramid module, and "α" represents the number of ESP modules or SEM-Bs. The SEM-B block bounded in the dashed box in (c) is composed of $(\alpha + 1)$ SEM-Bs, where the input feature maps of the FFM-B are concatenated to the output of the $(\alpha + 1)$ SEM-Bs and a downsampled raw image through the PMCA module.

number of channels to half of the input channels. Then, another standard 3×3 convolution is employed to integrate the feature maps of different scales and finally restore the number of channels that are consistent with the number of input channels. Besides, the convolutional output with restored channels is further added up with the input by another branch for identity mapping.

The merits of SEM-B can be summarized as follows. First, the standard 3×3 convolution is used in the head and tail of each SEM-B, which enlarges the receptive field of the model and makes the model get more context information. Second, in each SEM-B, split operation enables a lower computational complexity for the overall architecture. Meanwhile, the concatenation of multiscale convolutional results provides abundant feature information. Therefore, SEM-B greatly improves the feature extraction efficiency with a few parameters, which is verified with extensive experiments in Section IV.

B. SEM-B Block

In CNN networks, shallow features usually contain more detailed information, while deep features contain more abstract information. Therefore, we stack up two SEM-B blocks in different depths for extracting both shallow features by the former SEM-B block and deep features by the latter one in the LMFFNet. The first SEM-B block, which is composed of M $(M > 0)$ SEM-Bs, is primarily used to extract shallow features, while the second one, which consists of N $(N > 0)$ SEM-Bs, dedicates to unite with the front SEM-B block to extract deep features. The more detailed architectures of these two SEM-B blocks of LMFFNet are shown in Fig. 1(c) and (e).

The SEM-B block that consists of a number of consecutive SEM-Bs is the key component of FFM-B. There are $(\alpha + 1)$ SEM-Bs sequentially connected if the depth of an SEM-B block is supposed to be $\alpha + 1$. As the depth $(\alpha + 1)$ of the SEM-B block in FFM-B determines the feature level of the spatial information, it has a great impact on the accuracy of the LMFFNet, while the feature fusion strategy is implemented by the skip connection within the FFM-B modules. Since the lightweight-oriented model asks for fewer model parameters, the LMFFNet should not employ too many SEM-B blocks, and the depth of each SEM-B block can be determined by a finite number of experiments within a narrow range. Nevertheless, we further adopt an automatic screening method based on the Bayesian optimization [58] for searching the best depth $(\alpha + 1)$ during the training phase for the SEM-B block. We have introduced a new metric for performance evaluation of neural network in (9) and make the training phase as an iteration to achieve a sufficiently good value calculated according to new metric based on the Bayesian optimization. We carried out extensive experiments as described in Section IV to choose the most appropriate depth of the SEM-B block.

C. Feature Fusion Module

The skip-connection and shortcut connection strategies are found to be effective for various vision tasks in many models, such as ResNet [13], ESPNet [29], EFRNet [49], U-net [54], FCN [55], and DenseNet [56], for reusing or fusing features in different levels. The skip-connection strategy to connect the output layer to all of the hidden units as well as the input layers to fuse multistage features has been investigated and achieved very competitive accuracy based on a solid theoretical analysis in AdaNet [57]. AdaNet [57] derived a theoretical framework for networks with cross-layer connections and constructed an automatically learn network structure, balancing model complexity with empirical risk minimization. Likewise, DenseNet [56] reported a dense connection that the feature maps of all preceding layers were used as inputs for each layer, while its own output feature maps were used as inputs into all subsequent layers. Many state-of-the-art works exhibited the advantages of skip connection between layers close to the input and those close to the output to build up a more accurate and efficient convolution network for semantic segmentation [56]. Specifically, ESPNet [29] adopted a skip-connection strategy that concatenated the downsampled image with prior ESP module, as shown in Fig. 3(a).

Motivated by the above skip-connection strategies, we design two types of FFMs for fusing multiscale features, as shown in Fig. 3(b) and (c). Specifically, FFM-A directly integrates the downsampled image with the convolutional result of the initial block in the input stage. By contrast, FFM-B not only establishes a long-range skip connection to concatenate the downsampled image from the input but also establishes a short-range skip connection to fuse the convolution results of the first SEM-B and the last SEM-B in the SEM-B block bounded in the dashed box, as shown in Fig. 3(c).

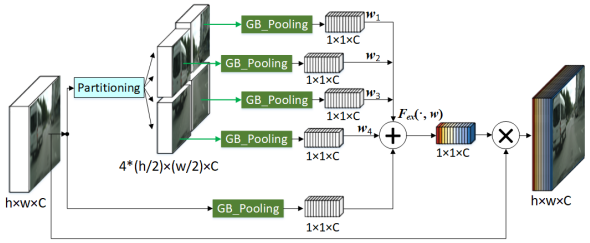

Fig. 4. Architecture of the proposed PMCA module.

Different from the ESP module as shown in Fig. 3(a), a pointwise convolution (Conv 1 × 1) is employed to make further integration in FFM-A and FFM-B in order to extract more useful information with fewer model parameters. FFM-A fuses the downsampled image information and the convolutional result of the initial block and efficiently avoids too much loss of useful information for further processing. Besides, the skip connection in FFM-B with a dilated convolution branch can learn both global and local features that capture long- and short-range spatial relationships between pixels. Through the use of skip connections where feature maps from the final layers of the model are downsampled and fused with feature maps of earlier layers in each FFM-B, the LMFFNet combines appearance information from fine layers with shallow features and semantic information from coarse layers with deep features by stacking a number of FFM-B modules.

Although the initial skip-connection strategy is not first launched in LMFFNet, our FFM-Bs not only integrate the downsampled image information but also introduce the information of SEM-B. In addition, instead of using the identity mapping with shortcut connection to directly fuse the input feature to the output of block as many existing methods do, we introduce a novel PMCA mechanism to focus on the features in important channels while connecting the input and output of the SEM-B block in an FFM-B. However, as a lightweight-oriented model, it is an inadvisable choice to stack too many SEM-Bs in FFM-B considering the model size. As a lightweight network, the LMFFNet would prefer to stack finite several FFM-Bs to build up the network architecture. Thus, simple enumeration can also be employed to train the LMFFNet. Eventually, we apply three FFMs (i.e., FFM-A1, FFM-B1, and FFM-B2) to construct the backbone of LMFFNet, as shown in Table I and Fig. 1.

D. Partition-Merge Channel Attention

We introduce the PMCA during the skip connection between the input and output of the SEM-B block. As shown in Fig. 4, the input feature maps are partitioned to a number of regions. Specifically, the regional global average pooling layer in PMCA can apply four partitioned regions of the input feature maps to obtain four groups of regional average pooling values in parallel. Then, each branch can adaptively and respectively learn the weights of four groups of regional pooling values through the neural network. Finally, a set of final pooled values is obtained by the weighted sum to achieve more detailed attention to specific channels. To pay attention to the specific feature channels and capture the important information, the PMCA obtains a set of pooled values after global average pooling and then learns the channel weights through several layers of neural networks. More regions can be partitioned in the input feature maps, but more computational operations are needed for a slight improvement of accuracy due to the similarity of adjacent regions according to our investigation in extensive experiments.

E. Multiscale Attention Decoder

In an encoder–decoder-based semantic segmentation architecture, the encoder produces dense feature maps, while the decoder upsamples the feature maps to match the original input resolution. A well-designed decoder can effectively recover the spatial details and improve the segmentation accuracy performance based on a small parameter size.

To improve the model performance with fewer parameters, a PAD [11] shown in Fig. 5(a) achieves generally good accuracy. The main idea of PAD is to transform the 1/4 feature map into an attention map, guides the 1/8 feature map to recover the detail information, and finally recovers more spatial information by pixel-level addition with the 1/2 feature map. However, in PAD, only 1/4 feature map is used to generate an attention map that it is difficult for PAD to extract multiscale spatial details for accurate semantic segmentation. Therefore, we design a new decoder named MAD shown in Fig. 5(b), combining two-scale features in one stage to refine and generate more accurate attention maps for our LMFFNet.

First, the output feature map of FFM-B1 F_{11} is transformed to F_{12} with C1 channels by

$$F_{12} = f_{1\times 1\text{Conv}}(F_{11}) \tag{1}$$

where $f_{1\times 1\text{Conv}}$ represents the 1 × 1 convolution operation. By contrast, the output feature map F_{21} of FFM-B2 is transformed to F_{22} with $C2$ channels by an individual 1 × 1 pointwise convolution. We further apply the upsampling operation to double the size of feature map F_{21}–F_{23}, which can be expressed as

$$F_{23} = f_{\text{up}}\Big(f_{1\times 1\text{conv}}\big(F_{21}\big)\Big) \tag{2}$$

where f_{up} is the upsampling function implemented by bilinear interpolation.

Second, the converted feature maps of F_{12} and F_{23} are fused together through a concatenation operation. Finally, a depthwise separable 3 × 3 convolution is followed by the concatenation operation to get F_{Dwconv} as

$$F_{\text{Dwconv}} = f_{\text{Dwconv}}\Big(W_{3\times 3}, f_{\text{concat}}\big(F_{12}, F_{23}\big)\Big) \tag{3}$$

where f_{concat} refers to the concatenation operation, f_{DwConv} represents the depthwise convolution, and $W_{3\times 3}$ is a depthwise convolutional kernel, i.e., a parameter matrix in size of 3 × 3.

Then, a sigmoid activation function is utilized to generate a multiscale attention map M_{mam} according to

$$M_{\text{mam}} = \delta\Big(f_{1\times 1\text{conv}}\Big(\sigma\Big(f_{\text{nor}}\big(F_{\text{DwConv}}\big)\Big)\Big)\Big) \tag{4}$$

where f_{nor} means the BN, σ indicates the PReLU activation, and δ denotes the sigmoid activation.

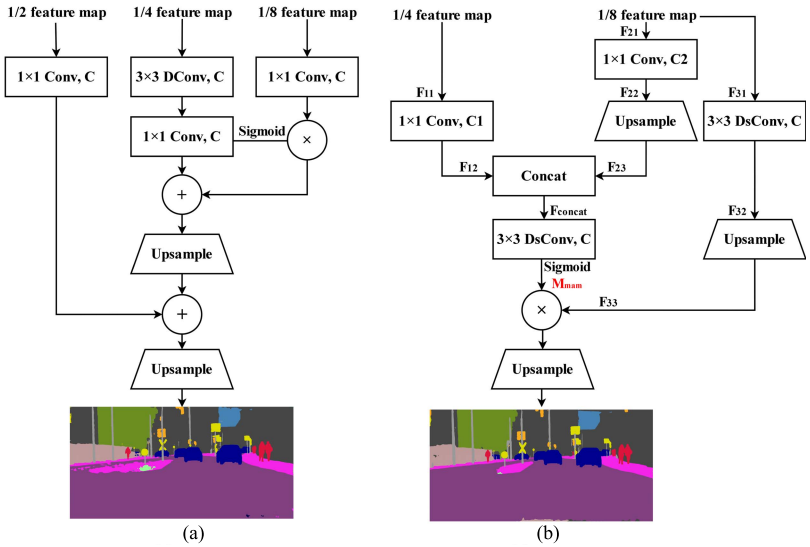

Fig. 5. Comparison of different decoders. (a) PAD. (b) MAD. Note that "1/*n* feature map" represents the feature map in different sizes in the feature extraction phase. "Conv" is the standard convolution, "DConv" denotes depthwise convolution, and "DsConv" indicates ds-Conv. "*C*" is the number of predict classes. "*C*1" and "*C*2" are the number of feature channels of the 1/4 feature map and 1/8 feature map, respectively.

In addition, the input 1/8 feature map from the FFM-B2 is applied to both the depthwise convolution and another branch for upsampling in MAD. The updated feature map F_{33} can be calculated by

$$F_{33} = f_{\text{up}}\left(\sigma\left(f_{\text{nor}}\left(f_{\text{DwConv}}\left(W_{3\times3}, F_{31}\right)\right)\right)\right) \tag{5}$$

which is further utilized for a pointwise multiplication with the attention map M_{mam} calculated by the aforementioned sigmoid activation operation.

Finally, another upsampling operation is utilized to restore the feature map to its original size. Thus, the final output of the MAD can be computed by

$$f_{\text{out}} = M_{\text{mam}} \odot F_{33} \tag{6}$$

where $\odot$ is a pointwise multiplication.

Although we employ an asymmetry encoder–decoder architecture for efficient networks, multilevel feature information is still critical for improving accuracy performance. Thus, we apply multiple branches to gather multilevel features. Different from the PAD strategy that only uses a single scale of the 1/4 feature map for generating attention maps, the 1/4 feature map and 1/8 feature map are fused to capture more multiscale spatial information and achieve better recovery of the original image objects in our MAD. Corresponding to the two FFM-Bs in the encoder with fast downsampling, MAD aims to recover the input information by gathering low- and high-level features with less computational complexity, to achieve fast inference speed. Since we employ the attention mechanism to each branch of the MAD, each branch focuses on learning different information and considers it as much important features for restoration. As a result, it is possible to get worse accuracy performance even though more levels of features from the encoder are used in MAD. Based on the investigation with extensive experiments, we found that the attention map with fused features has already contained sufficient information to recover the spatial information of the input image with significant accuracy. The 1/2 feature map as in PAD causes more unnecessary computation that it is removed in our work for reducing the information weight of the attention map. Besides, we discard all pixel-level addition operations in MAD. Consequently, the lightweight decoder with a multiscale attention mechanism can well recover the spatial details of the feature map with only 0.09 M parameters.

F. Overview of LMFFNet Architecture

The overall architecture of LMFFNet based on the asymmetric encoder–decoder scheme is shown in Fig. 1.

An initial block is employed at the beginning of the real-time segmentation network LMFFNet to change the size of the original input image and remove its redundant information. In the initial block, first, we use a 3 × 3 standard convolution module with a stride of "2" to reduce the original input image size by half and expand the channel number of the feature map to 32. Then, two 3 × 3 standard convolutions are adopted for extracting rich context information. Besides, we use a downsampling module in LMFFNet to increase the receptive field. The downsampling module consists of two parallel components: a 3 × 3 standard convolution with stride "2" and a 2 × 2 maximum pooling layer. Then, a concatenation operation is performed to the outputs of the above two parallel components. The combination of 3 × 3 convolution and maximum pooling layer can retain more spatial information in a larger receptive field and alleviate the problem of spatial information loss.

Consequently, in the encoder phase in our LMFFNet model, the initial block captures more useful information at the beginning of the network, the downsampling operation enlarges the receptive field of the model, the SEM-B block extracts more context information, and the FFM fuses multiscale features. In the decoder phase, the MAD introduces the attention mechanism and efficiently restores the spatial details with a more accurate output.

IV. Experimental Results and Analysis

In this section, we evaluate our LMFFNet on four benchmarks: Cityscapes [2], CamVid [30], KITTI [59], and WildDash2 [60]. We first introduce brief information about these four datasets. Then, we implement a series of ablation experiments on the Cityscapes and CamVid datasets to prove the effectiveness of the proposed network. We report the model performance in terms of parameter size, accuracy (mIoU), and inference speed and compare them with many existing models for real-time semantic segmentation. All the architectures were trained with PyTorch on an RTX 2080Ti GPU, while inference evaluations are performed on a single RTX 3090 GPU if without special statement.

A. Implementation Protocol

All the training experiments are conducted with one 2080Ti GPU, CUDA 10.0, and cuDNN V8 on the PyTorch platform. Mini-batch stochastic gradient descent (SGD) [31] is employed in our optimization strategy, where we set the batch size to 8, the momentum to 0.9, and the weight decay to $1e^{-4}$. Besides, the learning rate decay policy we use is "poly" [6] and we set its initial learning rate to $4.5e^{-2}$ with a power of 0.9. The formula can be expressed as follows:

$$\mathrm{lr} = \mathrm{lr}_{\mathrm{init}} \times \left(1 - \frac{\mathrm{iter}}{\mathrm{max\_iter}}\right)^{\mathrm{power}} \tag{7}$$

where lr refers to the current iter learning rate, $\mathrm{lr}_{\mathrm{init}}$ indicates the initial learning rate, iter means the iteration, and max_iter is the maximum iteration.

We set the maximum epochs to 1000 when training on Cityscapes and CamVid. Also, the techniques of random scale, mean subtraction, and horizontal flip are all employed in the training phase. We set the random parameters as 0.75, 1.0, 1.25, 1.5, 1.75, and 2.0 to transform the training images to different scales. For the Cityscapes dataset, we randomly crop the training images into 512×1024 resolution in the training phase due to the limitation of GPU memory size. For the CamVid dataset, we estimate two resolutions of 720×960 and 360×480 for ablation experiments. Finally, the online-hard-example-mining (OHEM) loss is employed on Cityscapes and the class weighting scheme is used on CamVid to handle the category imbalance problem. The class weight W_{class} can be calculated as follows:

$$W_{\mathrm{class}} = \frac{1}{\ln(c + P_{\mathrm{class}})} \tag{8}$$

where c is a hyperparameter that is set to 1.10 and P_{class} is the distribution of class samples.

The depth of LMFFNet can be automatically determined by the Bayesian algorithm [58]. Therefore, we define a Bayesian optimization method for designating the depth of SEM-B block while introducing a new evaluation index I_{auto} as

$$I_{\mathrm{auto}} = \frac{w \times (m_i - m_b) + (f_i - f_b)}{\ln(p_i + d)} \tag{9}$$

where m_i, f_i, and p_i are mIoU, frames per second, and parameter corresponding to the model in the ith iteration of Bayesian optimization algorithm, respectively. w is the weight coefficient of mIoU, d denotes the suppression coefficient of sensitivity to parameter quantity, m_b represents the baseline of mIoU, and f_b is the baseline of frames per second. m_b and f_b are used to measure the lowest tolerable mIoU and frames per second in the search process, respectively. We set w as 20, m_b and f_b as 65, and d as 20 in our experiments. In addition, referring to LEANet [51], we propose the index I_i to weigh the mIoU, frames per second, and parameter of lightweight network, which is expressed as follows:

$$I_i = \frac{m_i^{\mathrm{norm}} + f_i^{\mathrm{norm}}}{1 + \ln\left(p_i^{\mathrm{norm}} + 1\right)} \tag{10}$$

where $m_i^{\mathrm{norm}} = ((m_i - m_b)/(m_{\max} - m_b))$, $f_i^{\mathrm{norm}} = (f_i/f_{\max})$, $p_i^{\mathrm{norm}} = (p_i/p_{\max})$, and $m_{\max}$, $f_{\max}$, and $p_{\max}$ are the maximal mIoU, frames per second, and parameter, respectively, within all compared networks.

B. Datasets

1) Cityscapes: The Cityscapes dataset is a large urban street scene dataset that contains 5000 fine annotated images and 20 000 coarse annotated images with a resolution of 1024×2048, 19 semantic categories. The fine annotated images are split into three sets: a train set of 2975 images, a validation set of 500 images, and a test set of 1525 images.

2) CamVid: The CamVid dataset contains a total of 701 images, which is another street scene dataset and has a resolution of 720×960. We also divide it into three sets: 367 images for training, 101 images for validation, and 233 images for testing. Following the general setting, we annotated the ground truth to 11 semantic categories before starting the experiments.

3) KITTI: KITTI is an automatic driving dataset collected in the rural areas of a certain city in Germany, which contains 200 annotated images, and the semantic categories are compatible with Cityscapes. To verify the robustness of our semantic segmentation network, the model LMFFNet trained on Cityscapes is applied to evaluate its performance on KITTI.

4) WildDash2: WildDash2 is another dataset that covers road scenes under different environmental conditions, such as night and rainy days. It contains 4256 annotated images and 19 semantic categories. Similarly, we use the model trained on Cityscapes to evaluate the performance of LMFFNet.

C. Ablation Studies

In this section, we design a series of ablation experiments to evaluate the effectiveness of each component of our proposed LMFFNet. All the experiments in this section are evaluated on the Cityscapes validation set and the inference speed is evaluated at an input resolution of 512×1024.

1) Influence of SEM-B Block Depth: Two parameters of M and N are used for indicating the number of SEM-B inside the SEM-B Block1 and the SEM-B Block2 in LMFFNet, as shown in Fig. 1, respectively. We did a series of ablation experiments by changing the values of M and N manually or automatically to measure the model performance with respect to segmentation accuracy (mIoU), inference speed (frames/s), and model size (parameters amount). Extensive experimental results on the Cityscapes validation set are listed

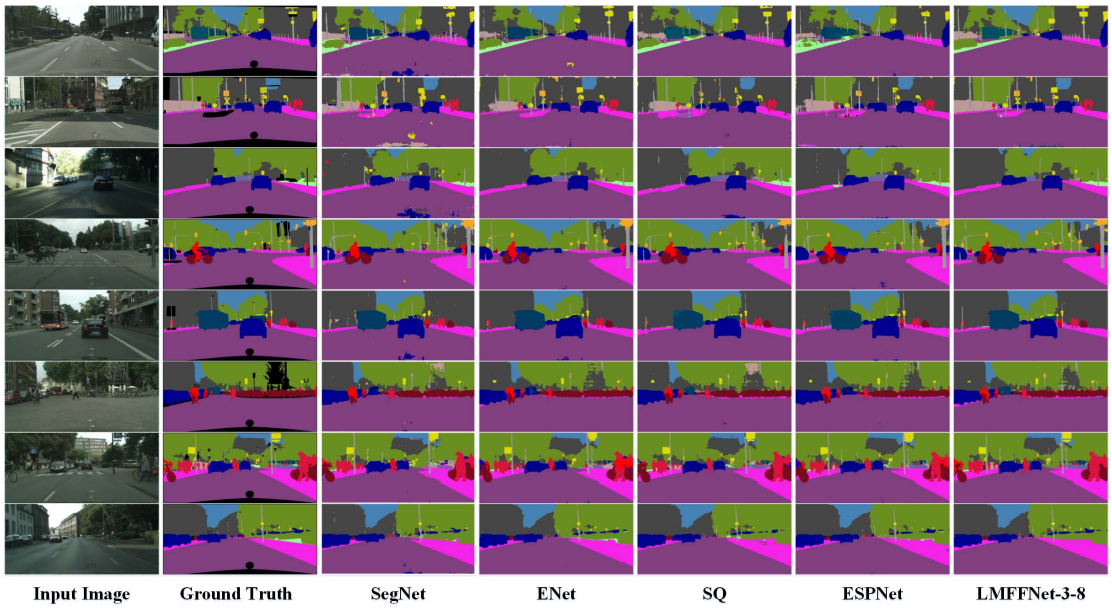

Fig. 6. Comparison of segmentation effectiveness of the proposed network LMFFNet-3-8 with four other models using the Cityscapes validation set.

TABLE II

RESULTS OF USING DIFFERENT M AND N IN SEM-B BLOCK ON THE CITYSCAPES VALIDATION SET

iter	M	N	mIoU (%)	Speed (fps)	Params (M)	I_{auto}
1	3	1	53.1	209.1	0.51	-31.10
2	4	5	66.5	133.7	1.02	32.48
3	4	8	69.6	112.0	1.38	45.21
5	8	12	70.7	72.1	1.99	39.19
5	5	12	70.5	83.6	1.90	41.33
6	5	9	70.3	91.4	1.53	43.13
7	3	8	69.5	120.5	1.35	47.34
8	3	15	70.7	78.9	2.20	41.38
9	1	11	68.5	113.6	1.65	38.64
10	3	10	69.6	102.7	1.59	45.64

TABLE III

EXPERIMENTAL RESULTS OF ABLATION STUDY WITH FFM-B ON THE CITYSCAPES VALIDATION SET

α	mIoU	Speed (fps)	Param (M)	mIoU(%)	Speed (fps)	Params (M)
	(M=α+1)			(N=α+1)		
3	72.7	125.2	1.26	66.6	189.0	0.62
4	72.7	114.4	1.29	69.3	173.2	0.74
5	72.7	106.1	1.32	69.9	157.1	0.86
6	72.8	100.4	1.35	71.1	143.2	0.98
7	73.9	95.3	1.38	72.6	135.9	1.10
8	73.2	92.1	1.41	72.8	124.2	1.22

TABLE IV

RESULTS OF USING DIFFERENT BOTTLENECKS ON THE CITYSCAPES VALIDATION SET

Module	mIoU(%)	Speed (fps)	Params(M)
ResNet bottleneck [13]	60.6	171.3	0.77
SS-nbt module [23]	73.6	107.9	1.04
DAB modules [11]	71.7	109.2	1.04
Non-bt-1D [22]	74.3	171.3	2.0
SEM-B	74.9	118.9	1.35

in Table II. We have investigated the network performance if LMFFNet only uses the FFM-B1 or FFM-B2 individually, i.e., either (FFM-A1 and FFM-B1) or (FFM-A1 and FFM-B2) pair is used in LMFFNet. Besides, we evaluate the impact of SEM-B block depth if there are two FFM-B modules in our model. As shown in Table II, the accuracy tends to be better, while the depth of SEM-B block increases. However, the accuracy can be only slightly improved by simply increasing the SEM-B block depth. Note that the experimental results listed in Table II were trained in 200 epochs, while the other results were obtained in 1000 epochs. Up to 75.1% mIoU with 1000 epochs can be achieved by combining two FFM-B modules for LMFFNet-3-8. As the SEM-B block depth increases, LMFFNet can achieve better accuracy at the cost of a slower inference speed and a larger model size. Besides, the Bayesian optimization [58] is employed to automatically determine the optimal network depth M and N for SEM-B Block1 and SEM-B Block2, respectively. The depth of SEM-B in FFM-B1 and FFM-B2 always affects the overall performance of the network. We set the parameter α to control the number of SEM-B in FFM-B1 and FFM-B2, as shown in Table III. For making an appropriate tradeoff between parameters and accuracy, we set $M = 3$ and $N = 8$ to construct the backbone of LMFFNet-3-8.

2) Model Performance Analysis With SEM-B Block: To evaluate the effectiveness of our SEM-B, we apply four different bottlenecks, i.e., ResNet bottleneck [13], Non-bt-1D [22], SS-nbt [23], and depthwise asymmetric bottleneck (DAB) [11], to replace the SEM-B in LMFFNet to build four segmentation networks. The comparative experimental results are listed in Table IV. The ResNet bottleneck results in the smallest number of parameters and the fastest inference speed for the segmentation network at the expense of significant accuracy loss. The network with Non-bt-1D achieves considerable accuracy of 74.3% mIoU but with the largest number of parameters. Compared to the other four bottlenecks, our SEM-B achieves the best accuracy of 74.9% mIoU at a high inference speed at 118.9 frames/s with only 1.35 M model parameters, creating an increasing tradeoff between the segmentation accuracy, inference speed, and model parameters.

3) Model Performance Analysis With FFM: We find that adding one or more short-, middle-, and long-rang connections to FFM-B greatly improved the accuracy of the model. At the same time, the model connecting short- and long-range features reached 74.9% mIoU in the Cityscapes validation set, while adding middle-range connection on this basis reduced the mIoU of the model by 1.2%. It demon-

TABLE V

EXPERIMENTAL RESULTS OF ABLATION STUDY WITH ATTENTION MODULES OF FFM-B ON THE CITYSCAPES VALIDATION SET

Attention on FFM	mIoU (%)	Speed (fps)	Params(M)
Channel Attention	74.2	120.8	1.35
Partition Fusion Attention (PMCA)	74.9	118.9	1.35
No Attention	74.0	125.2	1.35

TABLE VI

EXPERIMENTAL RESULTS OF ABLATION STUDY WITH FFM-B ON THE CITYSCAPES VALIDATION SET

short	middle layer	long	mIoU(%)	Speed (fps)	Params(M)
			72.7	125.3	1.16
√			73.4	122.1	1.34
	√		73.1	122.1	1.35
		√	73.0	124.1	1.16
√	√		73.2	110.4	1.39
	√	√	73.3	120.4	1.35
√		√	74.9	118.9	1.35
√	√	√	73.7	113.3	1.40

TABLE VII

EXPERIMENTAL RESULTS OF ABLATION STUDY WITH THE NUMBER OF FFM-B ON THE CITYSCAPES VALIDATION SET

Number of FFM-B	mIoU (%)	Speed (fps)	Params(M)
1	72.7	125.2	1.26
2	74.9	118.9	1.35
3	76.3	108.6	4.04
4	75.6	101.5	10.42

strates that FFM-B achieves better performance by fusing the shallow feature from the short-range connection and the downsampled original image from the long-range connection. The feature of the middle connection is close to the output feature map of SEM-B block, resulting in the similarity of the scales of the two feature maps. Therefore, FFM-B has disadvantages when using the middle-range connection for multiscale fusion. To prove the effectiveness of our proposed PMCA, we insert different attention modules into FFM to evaluate its parameter performance, as shown in Table V. It can be seen that the PMCA is 0.7% higher than the channel attention proposed by SENet with little loss of inference speed. This shows that our proposed attention module can replace the channel attention module proposed by SENet in real-time application scenarios. Since the context information impacts significantly on the accuracy of the model, the FFM is applied to fuse multiscale context information in semantic segmentation tasks. To verify the effectiveness of FFM, we conduct a series of ablation experiments and summarize the comparative results in Tables V–VII based on the Cityscapes dataset.

4) Model Performance Analysis With MAD: To explore the impact of merging different depth features to generate attention feature maps on the performance of the decoder, we fuse different depth features to generate attention feature maps, as shown in Table VIII. To verify the effectiveness of MAD and FFM, we set up a series of experiments. First, we build the "Base" network for the LMFFNet with a variable number of SEM-B blocks. Then, the FFM to the "Base" network is to build up the LMFFNet backbone. MAD, PAD [11], and ERFD [22] are individually added to the LMFFNet backbone for further observation of the performance variation of the LMFFNet. The comparative experimental results are concluded in Table IX. We find that the accuracy is getting better when the FFM is combined with the networks.

TABLE VIII

EXPERIMENTAL RESULTS OF ABLATION STUDY WITH MAD ON THE CITYSCAPES VALIDATION SET

shallow	middle	deep	mIoU(%)	Speed (fps)	Params(M)
			73.6	127.2	1.26
√			72.8	114.0	1.33
	√		73.0	120.8	1.35
		√	72.2	119.2	1.34
√	√		74.9	118.9	1.35
	√	√	72.4	117.8	1.34
√		√	74.9	118.9	1.35
√	√	√	73.8	116.3	1.35

TABLE IX

RESULTS OF INCLUDING DIFFERENT FUNCTION MODULES FOR ABLATION ON THE CITYSCAPES VALIDATION SET

Method	mIoU(%)	Speed (fps)	Params(M)
Base	72.5	141.2	1.12
Base+FFM	73.6	127.2	1.26
Base+FFM+MAD	74.9	118.9	1.35
Base+FFM+PAD	74.0	119.7	1.27
Base+FFM+ERFD	74.3	83.4	2.0

When adding FFM to the "Base" network, it can improve by approximately 1.1% mIoU at a cost of more model parameters and lower inference speed. The combined network "Base + FFM + PAD" achieves about 74.0% mIoU, while another combined network "Base + FFM + ERFD" obtains 74.3% mIoU. The "Base + FFM + MAD" network achieves the best accuracy performance of 74.9% mIoU, which has increased by 1.3% mIoU to the "Base + FFM" network. It demonstrates the significant effect of the combination of FFM and MAD for improving the segmentation accuracy with an affordable increment of model size for real-time segmentation tasks. The LMFFNet constructed by FFM and MAD achieves a better tradeoff among the accuracy, inference speed, and model size for the segmentation network when compared to the PAD or ERFD. Finally, to explore the impact of the number of FFM-B on the network, different numbers of FFM-B modules are employed in LMFFNet and evaluate its performance, as shown in Table VII. With the increase of the number of SEM-B, the mIoU of the model increased continuously. However, when the number of FFM-B increased to 4, the mIoU began to decline. The possible reason is that the increase of the number of FFM-B leads to more downsampling of the model, which affects the recovery of spatial information and damage accuracy. Besides, it is noteworthy that with the increase of the number of FFM-B, the amount of model parameters increases rapidly, which is not conducive to lightweight application scenarios. Therefore, we set the FFM-B number to 2 to build the real-time network.

D. Comparison With the State-of-the-Art Approaches

We compare the performance of our LMFFNet with the state-of-prior-art segmentation models on the Cityscapes dataset, CamVid dataset, KITTI dataset, and WildDash2 dataset. LMFFNet-3-8 is estimated on the same Cityscapes validation and test set without pretraining.

First, the individual category results of different models on the Cityscapes test set are estimated in this article, which is summarized in Table X. We find that LMFFNet-3-8 achieves outstanding segmentation accuracy performance on the Cityscapes dataset compared to existing methods. In particular, our LMFFNet-3-8 performs better on small objects

TABLE X

PRECLASS RESULTS (%) OF DIFFERENT SEGMENTATION MODELS ON CITYSCAPES TEST SET

Method	Roa	Sid	Bui	Wal	Fen	Pol	TLi	TSi	Veg	Ter	Sky	Ped	Rid	Car	Tru	Bus	Tra	Mot	Bic	Class	Cat
SegNet [25]	96.4	73.2	84	28.4	29	35.7	39.8	45.1	87	63.8	91.8	62.8	42.8	89.3	38.1	43.1	44.1	35.8	51.9	57	79.1
ENet [21]	96.3	74.2	75	32.2	33.2	43.4	34.1	44	88.6	61.4	90.6	65.5	38.4	90.6	36.9	50.5	48.1	38.8	55.4	58.3	80.4
SQ [32]	96.9	75.4	87.9	31.6	35.7	50.9	52	61.7	90.9	65.8	93	73.8	42.6	91.5	18.8	41.2	33.3	34	59.9	59.8	84.3
ESPNet [29]	97	77.5	76.2	35	36.1	45	35.6	46.3	90.8	63.2	92.6	67	40.9	92.3	38.1	52.5	50.1	41.8	57.2	60.3	82.2
CGNet [9]	95.5	78.7	88.1	40	43	54.1	59.8	63.9	89.6	67.6	92.9	74.9	54.9	90.2	44.1	59.5	25.2	47.3	60.2	64.8	85.7
EDANet[34]	97.8	80.6	89.5	42	46	52.3	59.8	65	91.4	68.7	93.6	75.7	54.3	92.4	40.9	58.7	56	50.2	64	67.3	85.8
ERFNet [22]	97.2	80	89.5	41.6	45.3	56.4	60.5	64.6	91.4	68.7	94.2	76.1	56.4	92.4	45.7	60.6	27	48.7	61.8	66.3	85.2
ICNet [16]	97.1	79.2	89.7	43.2	48.9	61.5	60.4	63.4	91.5	68.3	93.5	74.6	56.1	92.6	51.3	72.7	51.3	53.6	70.5	69.5	86.4
LEDNet [23]	98.1	79.5	91.6	47.7	49.9	**62.8**	61.3	72.8	92.6	61.2	94.9	76.2	53.7	90.9	**64.4**	64	52.7	44.4	71.6	70.6	87.1
ESNet [24]	98.1	80.4	**92.4**	48.3	49.2	61.5	62.5	72.3	92.5	61.5	94.4	76.6	53.2	94.4	62.5	74.3	52.4	45.5	71.4	70.7	87.4
LMFFNet-3-8	**98.3**	**84.1**	92.0	**56.1**	**55.1**	62.6	**69.0**	**72.7**	**93.0**	**71.3**	**95.0**	**83.8**	**66.1**	**95.0**	60.6	**76.7**	**64.1**	**60.0**	**71.7**	**75.1**	**89.3**

TABLE XI

PERFORMANCE COMPARISON OF LMFFNET AGAINST STATE-OF-THE-ART SEMANTIC SEGMENTATION NETWORKS ESTIMATED ON THE CITYSCAPES DATASET

Method	Pretrain	Input Size	mIoU(%) val	mIoU(%) test	Params (M)	Speed (fps)	GPU	GFLOPs	GFLOPs@1024*	I_i
SegNet [25]	ImageNet	360×640	57.8	56.1	29.5	52.8	3090	143.4	652.5	0.24
ENet [21]	No	512×1024	58.8	58.3	**0.36**	47.7	3090	4.4	8.7	0.42
SQ [32]	ImageNet	1024×2048	59.9	59.8	16.3	6.1	3090	576.4	288.2	0.21
ESPNet [29]	No	512×1024	60	60.3	**0.36**	**155.7**	3090	3.5	6.9	1.02
CGNet [9]	No	1024×2048	63.5	64.8	0.49	24.9	3090	28	14	0.61
ContextNet [33]	No	1024×2048	67.3	66.1	0.85	79.8	3090	7.2	3.6	0.97
EDANet [34]	No	512×1024	67.8	67.3	0.68	109.9	3090	9	17.9	1.20
ERFNet [22]	No	512×1024	70.4	68	2.1	62.7	3090	26.9	53.7	0.95
Fast-SCNN [35]	ImageNet	1024×2048	68.6	68	1.1	-	3090	7	3.5	1.59
BiSeNetV1_Xception39 [12]	ImageNet	768×1536	69	68.4	5.8	105.8	1080Ti	14.9	13.2	-
LEDNet [23]	No	512×1024	70.6	69.2	0.95	57.1	3090	11.5	23	0.99
ICNet [16]	ImageNet	1024×2048	-	69.5	7.8	15.4	3090	28.4	14.2	0.69
DABNet [11]	No	1024×2048	69.6	70.1	0.76	21.0	3090	41.8	20.9	0.84
GUN [36]	ImageNet	512×1024	69.6	70.4	-	33.3	1080Ti	-	-	-
ESNet [24]	No	512×1024	70.4	70.7	1.66	53.1	3090	24.4	48.7	1.03
DFANet [26]	ImageNet	1024×1024	-	71.3	7.8	100	1080Ti	**3.4**	**3.4**	1.18
LRNNet [10]	No	512×1024	71.4	72.2	0.68	71	1080Ti	-	-	-
ShelfNet50 [37]	ImageNet	1024×2048	-	72.4	-	36.9	1080Ti	-	-	-
BiSeNetV1_ResNet18 [12]	ImageNet	768×1536	74.8	74.7	49	65.5	1080Ti	55.3	49.1	-
SwiftNet [15]	ImageNet	1024×2048	**75.4**	**75.5**	11.8	39.9	1080Ti	104	52	-
SwiftNet [15]	No	1024×2048	70.4	-	11.8	39.9	1080Ti	104	52	-
AGLNet [45]	ImageNet	512×1024	-	71.3	1.1	52	1080Ti	13.9	27.8	-
WFDCNet [43]	No	512×1024	72.2	73.7	0.5	88.5	3090	5.8	11.6	1.39
LMFFNet-3-8	No	512×1024	74.9	75.1	1.4	118.9	3090	16.7	33.3	**1.60**

TABLE XII

PERFORMANCE COMPARISON OF OUR METHOD AGAINST STATE-OF-THE-ART SEMANTIC SEGMENTATION NETWORKS ON THE CAMVID TEST SET

Method	Pretrain	Input Size	mIoU(%)	Params(M)	Speed(fps)	GPU	GFLOPs	GFLOPs@1024*	I_i
ENet [21]	No	360×480	52.8	**0.36**	60.0	3090	1.43	8.7	0.14
SegNet [25]	ImageNet	360×480	59.7	29.5	67.7	3090	107.5	652.5	0.14
ESPNet [29]	No	360×480	55.6	**0.36**	**219.8**	3090	1.1	6.9	1.02
SwiftNet [15]	No	720×960	63.3	11.8	-	3090	34.3	52	-
DFANet [26]	ImageNet	720×960	64.7	7.8	120	1080Ti	2.2	3.4	-
FPENet [38]	No	360×480	65.4	0.4	82.4	3090	**0.5**	**3.15**	**-**
CGNet [9]	No	360×480	65.6	0.5	66.1	3090	2.3	14	0.93
BiSeNetV1_Xception39 [12]	ImageNet	720×960	65.6	49	175	1080Ti	8.7	13.2	-
EDANet [34]	No	360×480	66.4	0.68	110.3	3090	2.9	17.9	1.17
DABNet [11]	No	360×480	66.4	0.81	97.8	3090	3.4	20.9	1.11
LEDNet [23]	No	360×480	66.6	0.95	61.1	3090	3.8	23	0.96
ICNet [16]	ImageNet	720×960	67.1	7.8	27.8	1080Ti	9.4	14.2	-
BiSeNetV1_ResNet18 [12]	ImageNet	720×960	68.7	5.8	116.3	1080Ti	32.4	49.1	-
AGLNet [45]	No	360×480	69.4	1.1	90.1	3090	4.6	27.9	-
WFDCNet [43]	No	360×480	67.5	0.5	88.7	3090	1.9	11.6	1.15
LMFFNet-3-8	No	360×480	69.1	1.4	120.8	3090	5.5	33.3	1.35
LMFFNet-3-8	No	720×960	**72.0**	1.4	116.4	3090	22	33.3	**1.46**

TABLE XIII

RECALL, PRECISION, AND F1-SCORE PERFORMANCE COMPARISON OF LMFFNET AND STATE-OF-THE-ART SEMANTIC SEGMENTATION NETWORKS ON THE CITYSCAPES AND CAMVID VALIDATION SETS ON RTX 3090 GPU

Method	Input size	Recall(%)	Precision(%)	F1 Score(%)	Method	Input Size	Recall(%)	Precision(%)	F1 Score(%)
		Cityscapes					**CamVid**		
ENet [21]	512×1024	68.4	76.6	72.2	ENet [21]	360×480	73.1	61.5	66.8
EDANet [34]	512×1024	78.1	81.6	79.8	CGNet [9]	360×480	76.7	78.8	77.8
DABNet [11]	512×1024	78.7	84.3	81.4	DABNet [11]	360×480	78.6	77.8	78.2
ERFNet [12]	512×1024	79.5	84.5	82.1	ERFNet [12]	360×480	80.0	77.3	78.8
ESNet [24]	512×1024	78.8	85.5	82.0	ESNet [24]	360×480	79.5	78.4	78.9
LEDNet [23]	512×1024	80.4	84.6	82.4	LEDNet [23]	360×480	79.3	77.3	78.1
SegNet [25]	512×1024	67.8	74.2	70.8	SegNet [25]	360×480	76.7	78.8	77.8
ESPNet [29]	512×1024	70.0	77.0	73.3	LMFFNet-3-6	360×480	78.6	82.7	80.6
CGNet [9]	512×1024	72.5	80.7	76.4	LMFFNet-3-6	720×960	82.3	82.2	82.3
LMFFNet-3-6	512×1024	81.5	86.0	83.7	LMFFNet-3-8	360×480	78.6	82.7	80.6
LMFFNet-3-8	512×1024	**83.6**	**86.4**	**85.0**	LMFFNet-3-8	720×960	**82.3**	**83.0**	**82.7**

such as traffic lights and traffic signs and segment uncertain shape objects such as terrain and vegetation excellently compared to most of the other networks. Note that the metric GFLOPs@1024 in Tables XI–XV represents the normalized GFLOPs of the model when the input resolution is 1024×1024.

TABLE XIV

PERFORMANCE COMPARISON OF LMFFNET AGAINST STATE-OF-THE-ART SEMANTIC SEGMENTATION NETWORKS ON THE KITTI DATASET ON RTX 3090 GPU

Method	Input Size	mIoU(%)	Reacall(%)	Precision(%)	F1 Score(%)	Params(M)	Speed(fps)	GFLOPs	GFLOPs@1024*	I_i
ENet [21]	512×1024	24.0	38.4	44.0	41.0	**0.36**	60.0	4.4	8.7	0.78
SegNet [25]	512×1024	31.3	45.1	50.5	47.6	29.5	49.8	326.3	652.5	0.47
ESPNet [29]	512×1024	28.5	40.9	39.3	40.1	**0.36**	**219.8**	**3.5**	**6.9**	1.56
CGNet [9]	512×1024	36.9	48.0	53.7	50.7	0.49	71.3	7.0	14.0	1.19
EDANet [34]	512×1024	35.3	49.2	48.6	48.9	0.68	110.3	9.0	17.9	1.39
DABNet [11]	512×1024	37.2	50.0	58.0	53.7	0.81	97.8	10.5	20.9	1.31
LEDNet [23]	512×1024	40.3	53.9	60.4	56.9	0.95	61.1	11.5	23	1.15
ERFNet [12]	512×1024	42.9	56.5	61.3	58.8	2.07	104.6	26.9	53.7	1.19
ESNet [24]	512×1024	38.0	53.1	59.1	55.9	1.66	115.4	24.4	48.7	1.05
LMFFNet-3-8	512×1024	**49.3**	**63.9**	**65.1**	**64.5**	1.35	118.9	16.7	33.3	**1.69**

TABLE XV

PERFORMANCE COMPARISON OF LMFFNET AGAINST STATE-OF-THE-ART SEMANTIC SEGMENTATION NETWORKS ON THE WILDDASH2 DATASET ON RTX 3090 GPU

Method	Input Size	mIoU(%)	Recall(%)	Precision(%)	F1 Score(%)	Params(M)	Speed(fps)	GFLOPs	GFLOPs@1024*	I_i
ENet [21]	1080×1920	14.8	13.7	35.6	30.5	**0.36**	13.8	17.2	8.7	0.89
SegNet [25]	1080×1920	15.5	28.1	31.6	29.7	29.5	5.9	1290.0	652.5	29.7
ESPNet [29]	1080×1920	15.1	27.3	28.1	27.3	**0.36**	30.6	**13.6**	**6.9**	1.20
CGNet [9]	1080×1920	20.5	32.5	34.5	33.5	0.49	24.9	27.7	14.0	1.33
EDANet [34]	1080×1920	16.7	29.7	30.3	30.0	0.68	30.0	35.4	17.9	1.25
DABNet [11]	1080×1920	21.3	33.4	38.5	35.8	0.81	21.0	41.4	20.9	1.28
LEDNet [23]	1080×1920	22.1	35.3	38.4	36.7	0.95	13.4	45.5	23	1.17
ERFNet_ResNet18 [12]	1080×1920	21.2	34.1	38.9	36.4	2.07	14.0	106.2	53.7	1.11
ESNet [24]	1080×1920	20.3	32.8	35.3	34.0	1.66	12.4	96.3	48.7	1.05
LMFFNet-3-8	1080×1920	**23.1**	**35.6**	**45.8**	**40.1**	1.35	**49.4**	65.9	33.3	**1.83**

In addition, we give the visualized comparison of segmentation results of our model against other four state-of-the-art models, as shown in Fig. 6. We could find that LMFFNet-3-8 can achieve comparable visual segmentation results to other networks. In particular, LMFFNet-3-8 appears to segment the images a little bit more precisely than the LMFFNet-3-6 ($M = 3$ and $N = 6$) with fewer parameters in encoder. Thus, we conclude that LMFFNet-3-8 is suitable to be applied for segmentation tasks in the field of automatic driving in complex urban road scenes based on high-resolution images such as the samples in the Cityscapes dataset. Similarly, we give the visual effect of CamVid, as shown in Fig. 7. It can be seen that our segmentation effect on the CamVid dataset is also more refined than several other segmentation networks.

Table XI shows the quantitative experimental results of the LMFFNets compared to other 23 existing segmentation models on the Cityscapes dataset. Approximately 75.1% mIoU can be achieved LMFFNet-3-8. Compared to other high-quality real-time semantic segmentation networks, such as BiSeNetV1_ResNet18 [12], SwiftNet [15], and ShelfNet [37], our achieves a decent accuracy without employing any pretraining. BiSeNetV1_ResNet18 achieves 74.7% mIoU with as many as 49.0 M parameters. The proposed LMFFNet-3-8 achieves a higher accuracy on a slightly smaller image resolution but yields extremely fewer parameters, i.e., approximately 44× fewer parameters than BiSeNetV1_ResNet18. SwiftNet reports a better segmentation accuracy on the Cityscapes validation and test set, while pretraining is employed. However, SwiftNet [15] only achieves a smaller accuracy of 70.4% mIoU without pretraining on the Cityscapes validation set and the model size in terms of parameters of SwiftNet is more than ten times to our LMFFNet-3-8 network. Nevertheless, to obtain stronger feature extraction ability, more previous accurate networks usually adopt the existing backbones such as ResNet18 [13] and Xception [14], which leads to lots of model parameters.

Table XI also presents the results of inference speed and GFLOPs for each model. For a fair comparison of speed, we evaluate the inference speed for all networks on the PyTorch platform with the same 3090 cards. In other words, the LMFFNets meet the speed requirement of real-time tasks for high-resolution input images. Besides, we set GFLOPs@1024 to represent the GFLOPs of each model when all input image resolutions are normalized to 1024 × 1024 for a fair comparison. Obviously, LMFFNets have a comparatively small number of operations in terms of GFLOPs@1024 compared with many other high-accuracy networks such as SwiftNet [15], ESNet [24], and BiSeNetV1_ResNet18 [12].

Furthermore, we estimate the model performance on images with lower resolutions on the CamVid dataset. Since the current receptive field of the LMFFNets is a little bit large for the CamVid dataset, it may lead to loss of local information that an appropriate receptive field has been adjusted for LMFFNet-3-8 when they are estimated based on the CamVid dataset. Dilated convolution turns to be unnecessary for experiments using the CamVid dataset. The experimental results performing on the CamVid dataset are summarized in Table XII. Specifically, our models are estimated at two resolutions using the CamVid dataset. Besides, LMFFNet-3-8 is better than in terms of accuracy, achieving 69.1% and 72.0% mIoU, respectively, for 360 × 480 and 720 × 960 images. Nevertheless, the inference speeds of the LMFFNet-3-8 are yet sufficient enough for real-time segmentation tasks.

In addition, to verify the robustness of our network, we also evaluated it on KITTI and WildDash2 datasets. We employ the model trained in Cityscapes to make predictions on the KITTI dataset and WildDash2 dataset. The results are shown in Tables XIV and XV. In the KITTI dataset, the mIoU of LMFFNet-3-8 is as high as 49.3%, which is 6.4% higher than that of ERFNet, and the best tradeoff indicated by the index I_i. In the WildDash2 dataset, LMFFNet-3-8 achieves the highest accuracy and the fastest inference speed, and the tradeoff index I_i also achieves the best. Finally, we conclude the performance of the proposed LMFFNet model intuitively in Fig. 8 using the Cityscapes dataset. Our two networks achieve significantly high accuracy and keep a decent inference speed

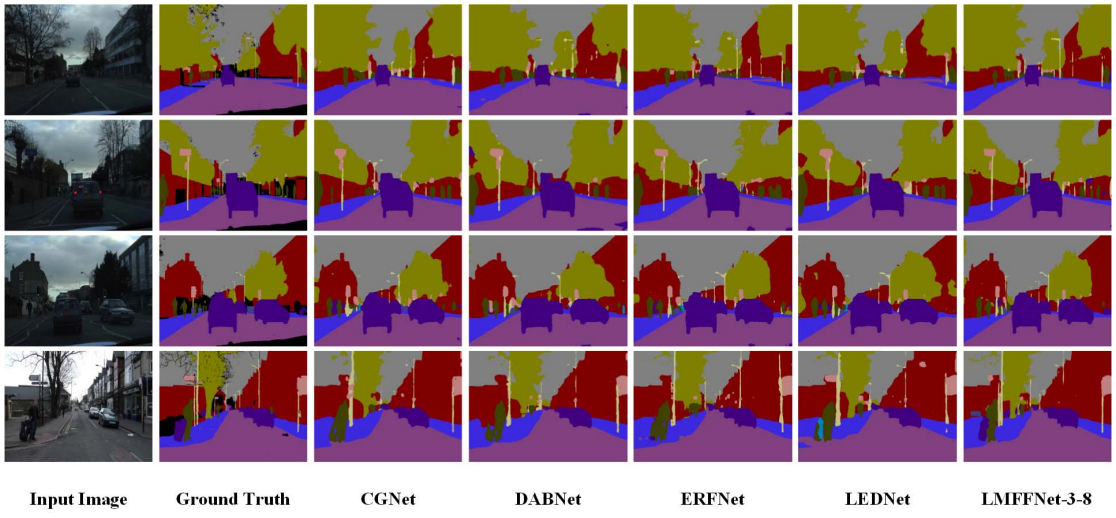

Fig. 7. Segmentation effectiveness using the CamVid test set.

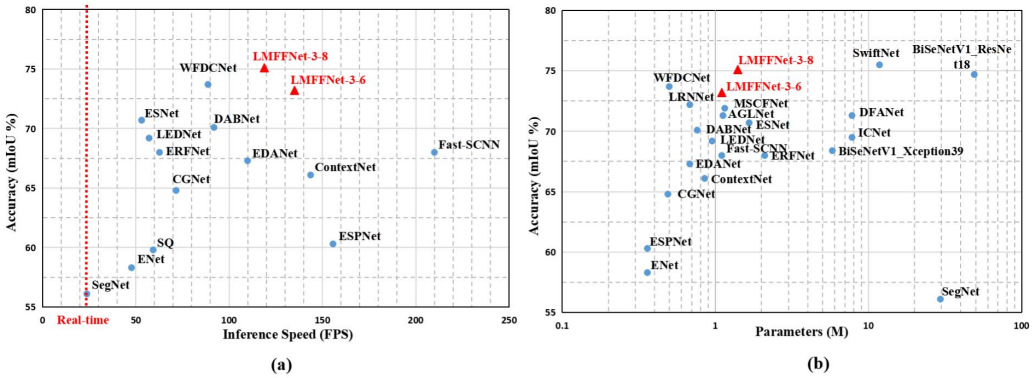

Fig. 8. Comparison with state-of-the-art neural networks in terms of tradeoff between accuracy and computing efficiency on the Cityscapes dataset evaluated on GeForce RTX 3090 GPU. (a) Results of segmentation accuracy versus inference speed. (b) Results of segmentation accuracy versus parameters amount.

for real-time segmentation tasks compared to the state-of-prior-art frameworks, as shown in Fig. 8(a). Besides, Fig. 8(b) shows that our models gain an adequate tradeoff between accuracy and parameter size for edge computing platforms equipped with limited hardware resources.

V. Conclusion and Future Work

In this article, we propose an LMFFNet for real-time semantic segmentation. It is composed of three types of components: SEM-B blocks, FFMs, and MAD. The SEM-B blocks extract contextual features efficiently and the FFMs are applied to fuse the long-range features and the short-range features to generate multiscale local features. The MAD aggregates multiscale features and introduces a new attention mechanism to gain a better recovery of spatial details. Then, a series of ablation studies on Cityscapes and CamVid datasets is performed to estimate the influence of each component of the LMFFNet and demonstrate the effectiveness of our proposed network for real-time semantic segmentation. Two networks combined with different components are verified with significant segmentation performance based on the LMFFNet model without pretraining. Compared with the existing semantic segmentation networks, our proposed network LMFFNet-3-8 achieves state-of-the-art tradeoffs among accuracy, parameter size, and inference speed for real-time segmentation tasks.

Nevertheless, there are still many tough issues for us to solve in the near future. The existing lightweight models for semantic segmentation lost much useful information that the model sizes were obtained at the cost of significant accuracy loss. The segmentation accuracy is still unsatisfactory. Attention mechanism is frequently used to channel transformation or spatial transformation currently, but there is very little room for accuracy improvement. In addition, the inference speed is not yet satisfactory to process high-resolution images. Besides, the power consumption of the semantic segmentation networks, which is extremely important for edge devices, does not achieve enough attention in existing research. Therefore, we dedicate to explore a novel architecture for semantic segmentation to gain a better tradeoff between inference speed, accuracy, and power consumption in the future.

References

[1] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, "Pyramid scene parsing network," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 2881–2890.

[2] M. Cordts *et al.*, "The cityscapes dataset for semantic urban scene understanding," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 3213–3223.

[3] L. C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, "Encoder decoder with atrous separable convolution for semantic image segmentation," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, Munich, Germany, 2018, pp. 801–818.

[4] J. Fu, J. Liu, J. Jiang, Y. Li, Y. Bao, and H. Lu, "Scene segmentation with dual relation-aware attention network," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 6, pp. 1–14, Aug. 2020.

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

[6] L. C. Chen, G. Papandreou, and I. Kokkinos, "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, Jun. 2017.

[7] 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. (CVPR)*, Jun. 2016, pp. 2818–2826.

[8] D. Mazzini, "Guided upsampling network for real-time semantic segmentation," 2018, *arXiv:1807.07466*.

[9] T. Wu, S. Tang, R. Zhang, J. Cao, and Y. Zhang, "CGNet: A light-weight context guided network for semantic segmentation," *IEEE Trans. Image Process.*, vol. 30, pp. 1169–1179, 2021.

[10] W. Jiang, Z. Xie, Y. Li, C. Liu, and H. Lu, "LRNNET: A light-weighted network with efficient reduced non-local operation for real-time semantic segmentation," in *Proc. IEEE Int. Conf. Multimedia Exp. Workshops (ICMEW)*, Jul. 2020, pp. 1–6.

[11] G. Li, I. Yun, J. Kim, and J. Kim, "DABNet: Depth-wise asymmetric bottleneck for real-time semantic segmentation," 2019, *arXiv:1907.11357*.

[12] C. Yu, J. Wang, C. Peng, C. Gao, G. Yu, and N. Sang, "BiSeNet: Bilateral segmentation network for real-time semantic segmentation," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 325–341.

[13] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.

[14] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 1251–1258.

[15] M. Orsic, I. Kreso, P. Bevandic, and S. Segvic, "In defense of pre-trained ImageNet architectures for real-time semantic segmentation of road-driving images," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, p. 12607.

[16] 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. (ECCV)*, 2018, pp. 405–420.

[17] X. Chang, H. Pan, W. Sun, and H. Gao, "YolTrack: Multitask learning based real-time multiobject tracking and segmentation for autonomous vehicles," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 12, pp. 1–11, Dec. 2021.

[18] N. Ma, X. Zhang, H.-T. Zheng, and J. Sun, "ShuffleNet v2: Practical guidelines for efficient CNN architecture design," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, Sep. 2018, pp. 116–131.

[19] J. Liu, Y. Wang, Y. Li, J. Fu, J. Li, and H. Lu, "Collaborative deconvolutional neural networks for joint depth estimation and semantic segmentation," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 11, pp. 5655–5666, Nov. 2018.

[20] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, *arXiv:1409.1556*.

[21] A. Paszke, A. Chaurasia, S. Kim, and E. Culurciello, "ENet: A deep neural network architecture for real-time semantic segmentation," 2016, *arXiv:1606.02147*.

[22] E. Romera *et al.*, "ERFNet: Efficient residual factorized convnet for real-time semantic segmentation," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 1, pp. 263–272, Oct. 2017.

[23] Y. Wang *et al.*, "LEDNet: A lightweight encoder-decoder network for real-time semantic segmentation," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Sep. 2019, pp. 1860–1864.

[24] Y. Wang, Q. Zhou, J. Xiong, X. Wu, and X. Jin, "ESNet: An efficient symmetric network for real-time semantic segmentation," in *Proc. Chin. Conf. Pattern Recognit. Comput. Vis. (PRCV)*, 2019, pp. 41–52.

[25] 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.

[26] H. Li, P. Xiong, H. Fan, and J. Sun, "DFANet: Deep feature aggregation for real-time semantic segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Long Beach, CA, USA, Jun. 2019, pp. 9522–9531.

[27] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Santiago, Chile, Dec. 2015, pp. 1026–1034.

[28] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," 2015, *arXiv:1502.03167*.

[29] 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. (ECCV)*, Munich, Germany, 2018, pp. 552–568.

[30] G. J. Brostow, J. Fauqueur, and R. Cipolla, "Semantic object classes in video: A high-definition ground truth database," *Pattern Recognit. Lett.*, vol. 30, no. 2, pp. 88–97, 2009.

[31] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Commun. ACM*, vol. 60, no. 2, pp. 84–90, Jun. 2017.

[32] M. Treml *et al.*, "Speeding up semantic segmentation for autonomous driving," in *Proc. NeurIPS*, Barcelona, Spain, 2016, vol. 2, no. 7, pp. 1–7.

[33] R. P. Poudel, U. Bonde, S. Liwicki, and C. Zach, "ContextNet: Exploring context and detail for semantic segmentation in real-time," 2018, *arXiv:1805.04554*.

[34] S. Y. Lo, H. M. Hang, S. W. Chan, and J. J. Lin, "Efficient dense modules of asymmetric convolution for real-time semantic segmentation," in *Proc. ACM Multimedia Asia*, 2019, pp. 1–6.

[35] R. P. Poudel, S. Liwicki, and R. Cipolla, "Fast-SCNN: Fast semantic segmentation network," 2019, *arXiv:1902.04502*.

[36] D. Mazzini, "Guided upsampling network for real-time semantic segmentation," 2018, *arXiv:1807.07466*.

[37] J. Zhuang, J. Yang, L. Gu, and N. Dvornek, "ShelfNet for fast semantic segmentation," in *Proc. IEEE/CVF Int. Conf. (ICCV)*, Seoul, South Korea, 2019, pp. 1–10.

[38] M. Liu and H. Yin, "Feature pyramid encoding network for real-time semantic segmentation," 2019, *arXiv:1909.08599*.

[39] Z. Li, F. Liu, W. Yang, S. Peng, and J. Zhou, "A survey of convolutional neural networks: Analysis, applications, and prospects," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, Jun. 10, 2021, doi: 10.1109/TNNLS.2021.3084827.

[40] C. Yu, C. Gao, J. Wang, G. Yu, C. Shen, and N. Sang, "BiseNet V2: Bilateral network with guided aggregation for real-time semantic segmentation," *Int. J. Comput. Vis.*, vol. 129, pp. 3051–3068, Sep. 2021.

[41] C. Yin, J. Tang, T. Yuan, Z. Xu, and Y. Wang, "Bridging the gap between semantic segmentation and instance segmentation," *IEEE Trans. Multimedia*, early access, Sep. 22, 2021, doi: 10.1109/TMM.2021.3114541.

[42] M. Fan *et al.*, "Rethinking BiSeNet for real-time semantic segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 9716–9725.

[43] X. Hao, X. Hao, Y. Zhang, Y. Li, and C. Wu, "Real-time semantic segmentation with weighted factorized-depthwise convolution," *Image Vis. Comput.*, vol. 114, Oct. 2021, Art. no. 104269.

[44] Y. Nirkin, L. Wolf, and T. Hassner, "HyperSeg: Patch-wise hypernetwork for real-time semantic segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 4061–4070.

[45] Q. Zhou *et al.*, "AGLNet: Towards real-time semantic segmentation of self-driving images via attention-guided lightweight network," *Appl. Soft Comput.*, vol. 96, Nov. 2020, Art. no. 106682.

[46] C. Peng, T. Tian, C. Chen, X. Guo, and J. Ma, "Bilateral attention decoder: A lightweight decoder for real-time semantic segmentation," *Neural Netw.*, vol. 137, pp. 188–199, May 2021.

[47] G. Gao, G. Xu, Y. Yu, J. Xie, J. Yang, and D. Yue, "MSCFNet: A lightweight network with multi-scale context fusion for real-time semantic segmentation," *IEEE Trans. Intell. Transp. Syst.*, early access, Aug. 2, 2021, doi: 10.1109/TITS.2021.3098355.

[48] M. Zhuang, X. Zhong, D. Gu, L. Feng, X. Zhong, and H. Hu, "LRDNet: A lightweight and efficient network with refined dual attention decorder for real-time semantic segmentation," *Neurocomputing*, vol. 459, pp. 349–360, Oct. 2021.

[49] K. Zhang, Q. Liao, J. Zhang, S. Liu, H. Ma, and J. H. Xue, "EFRNet: A lightweight network with efficient feature fusion and refinement for real-time semantic segmentation," in *Proc. IEEE Int. Conf. Multimedia Expo. (ICME)*, Jul. 2021, pp. 1–6.

[50] X. Zhang, B. Du, Z. Wu, and T. Wan, "LAANet: Lightweight attention-guided asymmetric network for real-time semantic segmentation," *Neur. Comp. Appl.*, vol. 34, pp. 1–15, Jan. 2022.

[51] K. Pang, L. Weng, Y. Zhang, J. Liu, H. Lin, and M. Xia, "SGBNet: An ultra light-weight network for real-time semantic segmentation of land cover," *Int. J. Remote Sens.*, pp. 1–23, Jan. 2022, doi: 10.1080/01431161.2021.2022805.

[52] X.-L. Zhang, B.-C. Du, Z.-C. Luo, and K. Ma, "Lightweight and efficient asymmetric network design for real-time semantic segmentation," *Int. J. Speech Technol.*, vol. 52, no. 1, pp. 564–579, Jan. 2022.

[53] J. Liu, X. Xu, Y. Shi, C. Deng, and M. Shi, "RELAXNet: Residual efficient learning and attention expected fusion network for real-time semantic segmentation," *Neurocomputing*, vol. 474, pp. 115–127, Feb. 2022.
[54] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional networks for biomedical image segmentation," in *Proc. Med. Image Comput. Comput.-Assist. Intervent.*, 2015, pp. 234–241.
[55] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 3431–3440.
[56] G. Huang, Z. Liu, L. V. D. Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proc. IEEE/CVF Conf. CVPR*, 2017, pp. 4700–4708.
[57] C. Cortes, X. Gonzalvo, V. Kuznetsov, M. Mohri, and S. Yang, "AdaNet: Adaptive structural learning of artificial neural networks," in *Proc. Int. Conf. Mach. Learn. (ICML)*, Jul. 2017, pp. 874–883.
[58] J. Snoek, H. Larochelle, and R. P. Adams, "Practical Bayesian optimization of machine learning algorithms," in *Proc. Adv. Neural Inform. Proce. Syst.*, vol. 25, 2012, pp. 1–9.
[59] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The KITTI dataset," *Int. J. Robot. Res.*, vol. 32, no. 11, pp. 1231–1237, 2013.
[60] O. Zendel, K. Honauer, M. Murschitz, D. Steininger, and G. F. Dominguez, "Wilddash-creating hazard-aware benchmarks," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 402–416.

Min Shi was born in Hubei, China. She received M.S. degree in electronic engineering from the Wuhan University of Technology, Wuhan, China, in 2002, and the Ph.D. degree in signal processing and wireless communication from the South China University of Technology, Guangzhou, China, in 2005.

She is currently an Associate Professor with Jinan University, Guangzhou, and the Director of the Technology Research Center for Satellite Navigation Chips and Applications, Guangdong University of Science and Technology, Guangzhou. Her research interests include machine learning, nonnegative signal processing, image processing, and satellite navigation.

Jialin Shen received the B.E. degree from Guangdong Ocean University, Zhanjiang, China, in 2019. He is currently pursuing the M.S. degree with Jinan University, Guangzhou, China.

His current research interests include computer vision, machine learning, and deep learning.

Qingming Yi received the B.S. degree from Xiangtan University, Xiangtan, China, in 1984, the M.S. degree from Jinan University, Guangzhou, China, in 1990, and the D.Eng. degree from the South China University of Technology, Guangzhou, in 2008, respectively.

She is currently a Full Professor with Jinan University. Her research interests include image processing, computer vision, multimedia security, and digital integrated circuit (IC) design.

Jian Weng (Member, IEEE) received the B.S. and M.S. degrees in computer science and engineering from the South China University of Technology, Guangzhou, China, in 2000 and 2004, respectively, and the Ph.D. degree in computer science and engineering from Shanghai Jiao Tong University, Shanghai, China, in 2008.

From 2008 to 2010, he held a post-doctoral position at the School of Information Systems, Singapore Management University, Singapore. He is currently a Professor and the Dean of the College of Information Science and Technology, Jinan University, Guangzhou. He has published over 100 papers in cryptography and security conferences and journals, such as CRYPTO, EUROCRYPT, ASIACRYPT, IEEE TRANSACTIONS ON CLOUD COMPUTING, IACR International Workshop on Public Key Cryptography (PKC), IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY, and IEEE TRANSACTIONS ON DEPENDABLE AND SECURE COMPUTING. His research interests include public key cryptography, cloud security, and blockchain.

Dr. Weng served as the PC co-chair or a PC member for over 30 international conferences. He also serves as an Associate Editor for the IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY.

Zunkai Huang received the B.S. degree in electronics engineering from Tianjin University, Tianjin, China, in 2013, and the Ph.D. degree in microelectronics from the Shanghai Advanced Research Institute, Chinese Academy of Sciences, Shanghai, China, in 2018.

He was with Hiroshima University, Higashihiroshima, Japan, as a Special Research Student, from 2016 to 2017. Since July 2018, he has been with the Shanghai Advanced Research Institute, Chinese Academy of Sciences, where he is currently an Associate Professor. His research focuses on CMOS image sensors, X-ray detectors, driver integrated circuits (ICs) for OLED/LED displays, and intelligent IC design.

Aiwen Luo (Member, IEEE) received the B.Eng. degree from Beijing Jiaotong University, Beijing, China, in 2009, the M.Eng. degree from Jinan University, Guangzhou, China in 2012, and the D.Eng. degree from Hiroshima University, Higashihiroshima, Japan, in March 2018.

From April 2018 to August 2019, she worked as a Post-Doctoral Researcher with Hiroshima University. She currently works with Jinan University and the University of Macau, Macau, China. Her research interests include computer vision, pattern recognition, robotics, and intelligent integrated circuit (IC) design.

Yicong Zhou (Senior Member, IEEE) received the B.S. degree from Hunan University, Changsha, China, in 1992, and the M.S. and Ph.D. degrees from Tufts University, Medford, MA, USA, in 2008 and 2010, respectively.

He is currently a Professor with the Department of Computer and Information Science, University of Macau, Macau, China. His research interests include image processing, computer vision, machine learning, and multimedia security.

Dr. Zhou is a fellow of the Society of Photo-Optical Instrumentation Engineers (SPIE). He was recognized as one of "Highly Cited Researchers" in 2020 and 2021. He received the Third Price of Macao Natural Science Award as a sole winner in 2020 and was a co-recipient in 2014. He has been the leading Co-Chair of the Technical Committee on Cognitive Computing of the IEEE Systems, Man, and Cybernetics Society since 2015. He also serves as an Associate Editor for IEEE TRANSACTIONS ON CYBERNETICS, IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS, IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY, and IEEE TRANSACTIONS ON GEOSCIENCE AND REMOTE SENSING.