The Visual Computer
https://doi.org/10.1007/s00371-024-03448-6

ORIGINAL ARTICLE

ZMNet: feature fusion and semantic boundary supervision for real-time semantic segmentation

Ya Li[1] · Ziming Li[1] · Huiwang Liu[1] · Qing Wang[2]

Accepted: 1 May 2024

Abstract
Feature fusion module is an essential component of real-time semantic segmentation networks to bridge the semantic gap among different feature layers. However, many networks are inefficient in multi-level feature fusion. In this paper, we propose a simple yet effective decoder that consists of a series of multi-level attention feature fusion modules (MLA-FFMs) aimed at fusing multi-level features in a top-down manner. Specifically, MLA-FFM is a lightweight attention-based module. Therefore, it can not only efficiently fuse features to bridge the semantic gap at different levels, but also be applied to real-time segmentation tasks. In addition, to solve the problem of low accuracy of existing real-time segmentation methods at semantic boundaries, we propose a semantic boundary supervision module (BSM) to improve the accuracy by supervising the prediction of semantic boundaries. Extensive experiments demonstrate that our network achieves a state-of-the-art trade-off between segmentation accuracy and inference speed on both Cityscapes and CamVid datasets. On a single NVIDIA GeForce 1080Ti GPU, our model achieves 77.4% mIoU with a speed of 97.5 FPS on the Cityscapes test dataset, and 74% mIoU with a speed of 156.6 FPS on the CamVid test dataset, which is superior to most state-of-the-art real-time methods.

Keywords Real-time semantic segmentation · Multi-level feature fusion · Attention mechanism · Encoder–decoder architecture · Boundary supervision

1 Introduction

Semantic segmentation is a fundamental task in computer vision that aims to precisely predict the label of each pixel in an image. It has been widely applied in many fields, such as autonomous driving, medical image segmentation, video surveillance, and more. As the demand for mobile device deployment grows, real-time semantic segmentation has become a hot research field in recent years (Fig. 1).

✉ Qing Wang
wangq79@mail.sysu.edu.cn

Ya Li
liya@gzhu.edu.cn

Ziming Li
liziming_maxlee@outlook.com

Huiwang Liu
liuhuiwang1025@outlook.com

[1] School of Computer Science and Cyber Engineering, Guangzhou University, Guangzhou 510006, China

[2] School of Computer and Engineering, Sun Yat-sen University, Guangzhou 510275, China

Real-time semantic segmentation is a challenging task that requires considering both segmentation accuracy and inference speed. To balance both accuracy and speed, MobileNet [1] reduces computation by using depthwise separable convolutions. BiSeNet [2, 3] proposes a two-pathway architecture to capture spatial and semantic information separately. STDC [4] designs a lightweight backbone and proposes a detail aggregation module to preserve the spatial details in low-level layers. Despite the development and impressive performance of real-time semantic segmentation driven by these methods, it still faces the following challenges:

1) Most of methods are inefficient in multi-level feature fusion. The semantic gap is common among different levels of features. Simple feature fusion methods, such as element-wise addition or channel-wise concatenation operation, require less computation, but they are not capable of effectively bridging the semantic gap, leading to suboptimal segmentation performance. Some fusion methods employing attention mechanisms, such as [2, 5], unilaterally utilize spatial or channel context information to generate attention weights, ignoring the effectiveness improvement brought by their joint generation. Moreover, sophisticated feature fusion

Published online: 20 June 2024

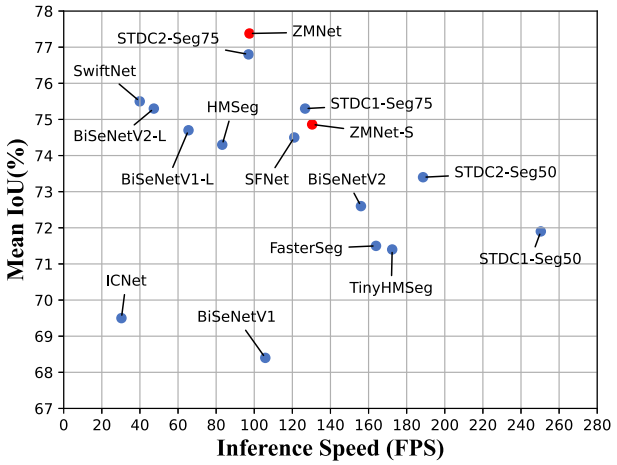

Fig. 1 The comparison of speed–accuracy performance on the Cityscapes *test* set. Red dots indicate our methods, while blue dots represent other methods. Notably, our proposed ZMNet achieves a state-of-the-art trade-off between segmentation accuracy and inference speed

methods [6, 7] can effectively bridge the semantic gap, but they come with high computational costs and are difficult to optimize, rendering them unsuitable for deployment on mobile devices.

2) Most semantic boundary optimizations require additional inference overhead. Semantic boundaries are crucial for segmentation accuracy, and further optimizing semantic boundary prediction can enhance segmentation performance. However, current real-time semantic segmentation methods [2, 3, 8–10] usually optimize semantic boundary prediction by adding extra learnable modules or branches, undoubtedly leading to additional training costs and inference overhead.

Inspired by the above observations, we propose a novel real-time semantic segmentation network, ZMNet, that balances accuracy and speed by using a lightweight decoder and semantic boundary supervision module. ZMNet adopts an encoder–decoder structure. In the encoding stage, different levels of convolutional blocks generate feature maps of different scales. Specifically, shallow feature maps contain more spatial detail information while deep feature maps contain more semantic information. In the decoding stage, we propose a new feature fusion module to fuse deep and shallow features in a top-down manner with multi-dimensional contextual information of input features. In addition, the semantic boundaries are refined further to improve accuracy. These two processes in the decoding stage are implemented by two modules: the multi-level attention feature fusion module (MLA-FFM) and the semantic boundary supervision module (BSM).

Our main contributions can be summarized as follows:

- We propose a lightweight multi-level feature fusion module called MLA-FFM, which is the critical component of our decoder. This module can bridge the semantic gap and fuse multi-level features in an effective and efficient way.
- We propose a semantic boundary supervision module called BSM, which aims to improve the accuracy of semantic boundaries without increasing the computational cost of the inference stage.
- Extensive experiments demonstrate that ZMNet achieves a state-of-the-art trade-off between accuracy and speed on both Cityscapes and CamVid datasets. More specifically, on one NVIDIA GTX 1080Ti card, ZMNet achieves **77.4% mIoU** on Cityscapes *test* set at **97.5 FPS**, and **74% mIoU** on CamVid *test* set at **156.6 FPS**.

2 Related work

Real-time Semantic Segmentation. With the increasing demands for mobile device deployment, real-time semantic segmentation has been an active research area in computer vision for the past few years, with significant progress achieved in terms of accuracy and speed. So far, a lot of deep convolutional neural network-based methods have been proposed, and have made great progress in the semantic segmentation of street scenes. MobileNet [1], ShuffleNet [11], and STDC [4] propose a lightweight and effective backbone to reduce parameters and computation of models. BiSeNet [2, 3] proposes a two-pathway architecture which a context path for high-level context information and a spatial path as a supplement to strengthen spatial information. Similarly, BFMNet [10] adopts a bilateral network structure to separately encode semantic feature information and detailed feature information, and introduces the attention enhancement fusion module (AEFM) to promote bilateral feature fusion. Fast-SCNN [12] proposes shallow learning to a down-sample module that contains three layers using stride 2 for fast and efficient multi-branch low-level feature extraction. GDN [13] proposes a guided down-sampling method that decomposes the original image into a set of compressed images, reducing the size of feature maps while retaining most of the spatial information of the original image, thus greatly reducing computational costs. DDRNet [14] maintains high-resolution representations and harvests contextual information simultaneously with a dual-resolution network structure and a deep aggregation pyramid pooling module (DAPPM). Method [15] proposes token pyramid module for processing high-resolution images to quickly produce a local feature pyramid. SFANet [16] proposes a Stage-aware Feature Alignment module (SFA) to efficiently align and aggregate two adjacent levels of feature maps. Method [17] develops a lattice enhanced residual block (LERB) to address the issue of inferior feature extraction capability of the lightweight backbone network, and proposes a feature

transformation block (FTB) that mitigates the problem of feature interference between different layers at a lower computational cost. Moreover, image segmentation under rain or fog conditions is very practical in real application, especially in real-time scenario [18–20].

Attention mechanism. The key concept of attention mechanism is to enhance feature representation and improve network performance by telling the network "what" and "where" to focus on. Attention mechanism has achieved great success and been widely applied in neural language processing tasks due to its high efficiency and simplicity. In recent years, it has also been used in the computer vision field. Wang et al. [21] bridge self-attention mechanism and non-local operators for capturing long-range dependencies with deep neural networks. Method [22] exploits the relationships between different channels by channel attention. Go further, BAM [23] and CBAM [24] introduce spatial attention, and combine it with channel attention to improve the representation power of CNNs. The ResNeSt [25] proposes a split-attention block that can perform feature map attention across different feature map groups. SENet [26] strengthens the spatial information representation by enhancing the information correlation between feature maps of different resolutions through an attention mechanism. Method [27] proposes sparse attention module (SAM) and class attention module (CAM) to optimize the computational cost of self-attention.

Feature Fusion Module. Feature fusion module is widely used in encoder–decoder structure networks. As the network layers deepen, the semantic gap between shallow features and deep features also increases: deep features contain high-level semantic information with low resolution, while shallow features contain high-resolution spatial information. Although traditional feature fusion methods such as element-wise addition or channel-wise concatenation can effectively reduce computational and parameter complexity, they may result in poor fusion performance. To better fuse features of different levels, BiSeNet [2] proposes a feature fusion module (FFM) to fuse the output features of the spatial path and context path to make the final prediction. SFNet [28] proposes a novel flow-based align module (FAM) to promote broadcasting high-level features to high-resolution features. DFANet [29] proposes sub-network aggregation and sub-stage aggregation modules to obtain sufficient receptive fields and enhance the model learning ability. Dai et al. [5] propose an attentional feature fusion (AFF) module, which fuses features of inconsistent semantics and scales by utilizing multi-scale contextual information along the channel dimension. In this paper, we propose multi-level feature fusion modules (MLA-FFM), which fuse features of different levels by utilizing multi-dimensional contextual information from different levels of features.

3 Proposed method

In this section, we present our proposed methods in detail. We first introduce the overall architecture of ZMNet in Sect. 3.1. Then, we present MLA-FFM in Sect. 3.2, which is used to fuse features from adjacent levels. Finally, we introduce BSM in Sect. 3.3, which is used to optimize the semantic boundary segmentation.

3.1 Network architecture

As shown in Fig. 2, the architecture of ZMNet follows the encoder–decoder paradigm. The backbone of encoder can be any computationally efficient CNN architecture. We choose STDC [4] as the backbone because of its outstanding performance in real-time semantic segmentation. The decoder is composed of a sequence of MLA-FFMs that integrate features in a top-down manner. To enhance the segmentation accuracy, we incorporate BSM as a supervision signal to guide the learning of semantic boundary results during training. The overall process is as follows:

Firstly, as illustrated in Fig. 2a, the encoder generates a series of multi-level feature maps, denoted as $\{x_1, \ldots, x_5\}$. It should be noted that during the decoding stage, involving x_1 in feature fusion would result in a significant computational burden due to its large spatial size, and the abundant spatial information from shallow feature map may hinder feature aggregation [30]. Therefore, in the decoding stage, we only utilize x_2, x_3, x_4, and x_5 for feature fusion.

Secondly, as shown in Fig. 2b, to integrate spatial details of shallow features during the upsampling process, the decoder utilizes lateral connections to perform top-down feature fusion with MLA-FFMs. To bridge the semantic gap, MLA-FFM exploits input feature maps' multi-dimensional contextual information to generate a soft attention map, then fuses inputs in a soft selection manner. The decoder of ZMNet comprises three decoding stages, each of which utilizes MLA-FFM to fuse feature maps from different layers and generate corresponding fused feature map, i.e., $\{x_6, x_7, x_8\}$. We denote the process of MLA-FFM as $\mathcal{F}(\cdot, \cdot)$:

$$\begin{aligned} x_6 &= \mathcal{F}(x_4, x_5) \\ x_7 &= \mathcal{F}(x_3, x_6) \\ x_8 &= \mathcal{F}(x_2, x_7) \end{aligned} \tag{1}$$

In the training phase, each MLA-FFM is followed by a segmentation head (SegHead) and an upsampling operation to produce segmentation prediction. All segmentation predictions are supervised by the ground-truth semantic labels. In this paper, we adopt cross-entropy loss with online hard example mining (OHEM) [31] to optimize the models. The

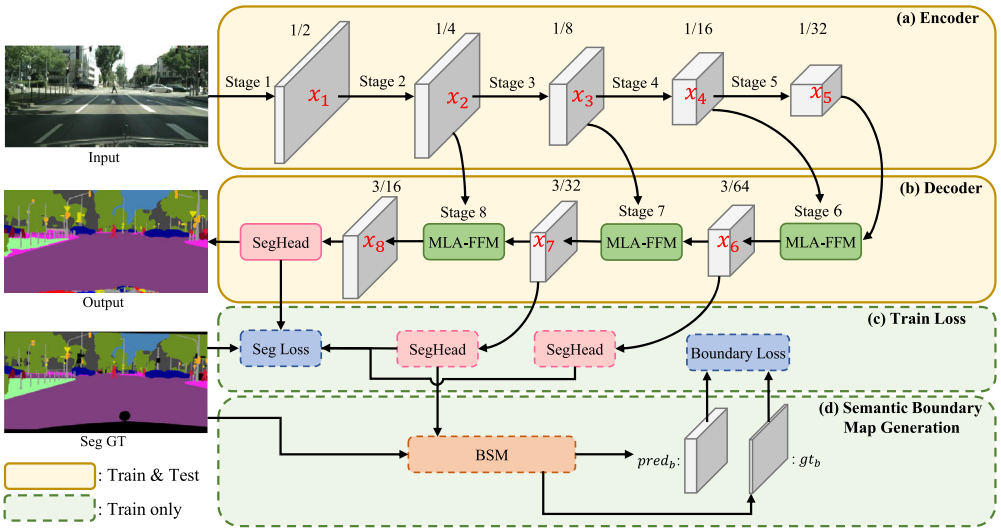

Fig. 2 The architecture overview of the ZMNet. "MLA-FFM" denotes the multi-level feature fusion module, and "BSM" denotes the semantic boundary supervision module

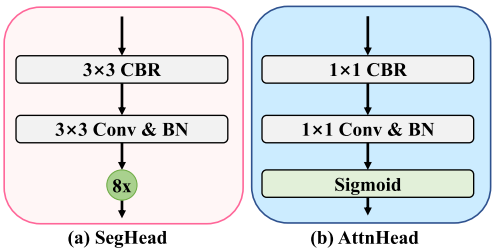

Fig. 3 The components of **a** SegHead and **b** AttnHead. "CBR" denotes the convolution with batch normalization (BN) and ReLU

joint segmentation loss l_s is calculated as follows:

$$\begin{aligned} pred_i &= \mathcal{I}_u(\mathcal{S}_i(x_i)) \\ l_s &= \sum_{i=6}^{8} \mathcal{L}_s(pred_i, gt) \end{aligned} \tag{2}$$

where x_i denotes the output feature map at the i-th stage, and $pred_i \in \mathbb{R}^{N\times H\times W}$ is the corresponding segmentation prediction probability map, where N is the number of classes. $\mathcal{S}_i(\cdot)$ denotes the i-th SegHead operation, and its structure is illustrated in Fig. 3a. Note that $\mathcal{S}_6(\cdot)$ and $\mathcal{S}_7(\cdot)$ are auxiliary SegHeads intended to enhance feature representation during training and they are discarded in the inference phase. $\mathcal{I}_u(\cdot)$ denotes the bilinear upsampling operation. $gt \in \mathbb{R}^{1\times H\times W}$ denotes the ground-truth semantic labels. $\mathcal{L}_s(\cdot)$ denotes the OHEM [31].

Thirdly, as shown in Fig. 2d, to obtain more accurate semantic boundaries, we use BSM to derive the semantic boundary ground-truth gt_b and the semantic boundary probability map $pred_b$ from the ground-truth gt and the segmentation prediction probability map $pred_7$ of the 7-th stage. As the pixels on the semantic boundary represent hard samples due to their relatively small proportion compared to non-semantic boundary pixels, we utilize focal loss [32] to increase the weight of hard samples and optimize the model's learning of semantic boundaries. Notably, BSM is a module designed to guide the learning of semantic boundary prediction during training and is discarded in the inference phase. The above procedure can be formulated as follows:

$$\begin{aligned} \{gt_b, pred_b\} &= \mathcal{B}(gt, pred_7) \\ l_b &= \mathcal{L}_b(gt_b, pred_b) \end{aligned} \tag{3}$$

where $\mathcal{B}(\cdot,\cdot)$ denotes the operation of BSM. $\mathcal{L}_b(\cdot)$ denotes the focal loss [32] function. l_b denotes the semantic boundary loss.

Finally, the overall objective function l is a combination of segmentation loss l_s and semantic boundary loss l_b. We use the trade-off parameter ξ to balance the weight of the segmentation loss and semantic boundary loss. In our paper, we set $\xi = 0.6$.

$$l = l_s + \xi \cdot l_b \tag{4}$$

3.2 Multi-level attention feature fusion module

As illustrated in Fig. 4, MLA-FFM receives two different levels of feature maps as inputs, where the shallow-level feature map x_h has a higher resolution than the deep-level feature map x_l. The size of x_h and x_l is first unified through channel compression and bilinear interpolation, and the unified feature maps are denoted as $x_{h'}$ and $x_{l'}$.

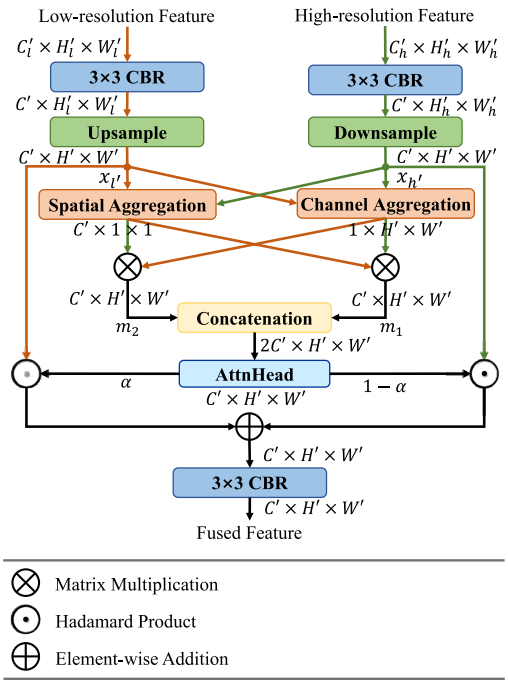

Fig. 4 The components of MLA-FFM. Where "Spatial Aggregation" refers to pooling operations along the spatial dimension, "Channel Aggregation" refers to pooling operations along the channel dimension. The symbol α represents the weight of the linear combination between $x_{l'}$ and $x_{h'}$

Next is information aggregation, through pooling operations to obtain a larger receptive field and capture multi-dimensional contextual information. Spatial aggregation and channel aggregation are performed, respectively. Spatial aggregation performs max-pooling operation on spatial dimension to aggregate spatial-wise context information. Channel aggregation performs average-pooling on channel dimension to aggregate channel-wise context information and avoid the loss of spatial location information in feature maps. After information aggregation, four feature maps are generated, i.e., $x_{l'}^{sp} \in \mathbb{R}^{C' \times 1 \times 1}$, $x_{l'}^{cp} \in \mathbb{R}^{1 \times H' \times W'}$, $x_{h'}^{sp} \in \mathbb{R}^{C' \times 1 \times 1}$, and $x_{h'}^{cp} \in \mathbb{R}^{1 \times H' \times W'}$.

Then, mix the aggregated information on the spatial and channel of different resolution features to generate attention weight tensor. That is, we multiply $x_{l'}^{sp}$ and $x_{h'}^{cp}$ to obtain m_1, and multiply $x_{h'}^{sp}$ and $x_{l'}^{cp}$ to obtain m_2. Then, m_1 and m_2 are concatenated and fed into the attention head (AttnHead) to generate an attention weight tensor α:

$$\begin{aligned} m_1 &= \mathcal{SP}(x_{l'}) \otimes \mathcal{CP}(x_{h'}) \\ m_2 &= \mathcal{SP}(x_{h'}) \otimes \mathcal{CP}(x_{l'}) \\ \alpha &= \mathcal{T}([m_1, m_2]) \end{aligned} \tag{5}$$

where $\mathcal{SP}(\cdot)$ and $\mathcal{CP}(\cdot)$ denote the spatial aggregation and channel aggregation, respectively. $[\cdot, \cdot]$ denotes the concatenation operation along the channel dimension. $\mathcal{T}(\cdot)$ denotes the AttnHead operation, and its structure is illustrated in Fig. 3b.

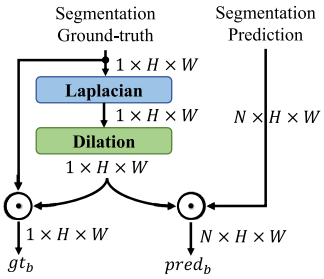

Fig. 5 The components of BSM. Where "Laplacian" denotes to the application of 2D convolution using the Laplacian kernel as the weight, "Dilation" denotes the dilation operation

Finally, we perform soft selection on $x_{h'}$ and $x_{l'}$, and then use element-wise addition followed by a 3×3 CBR to obtain fused feature map. The merge procedure can be formulated as follows:

$$x_{h+1} = CBR(x_{h'} \odot \alpha + x_{l'} \odot (1 - \alpha)) \tag{6}$$

3.3 Semantic boundary supervision module

The semantic boundary is crucial for semantic segmentation tasks. However, down-sampling operations in deep convolutional neural networks cause the loss of spatial detail information, resulting in rough prediction results of semantic boundaries and their adjacent pixels. To alleviate this problem, we propose BSM supervise the semantic boundary of segmentation results. Figure 5 illustrates the structure of BSM.

Firstly, a binary boundary mask $m_b \in \mathbb{R}^{1 \times H \times W}$ is extracted from the ground-truth semantic labels using a 2D convolution with the Laplacian kernel as the weight. In the binary boundary mask, the value is 1 if it is a semantic boundary, otherwise it is 0.

Then, a dilation operation on the binary semantic boundary mask is performed to expand the range of boundary pixels, in order to treat pixels near the semantic boundary as part of the semantic boundary. This operation produces a dilated binary boundary mask, denoted as $m_d \in \mathbb{R}^{1 \times H \times W}$.

Finally, the dilated binary boundary mask is element-wise multiplied with both the ground-truth semantic labels and the predicted probability map to obtain two semantic boundary maps, i.e., the semantic boundary ground-truth $gt_b \in \mathbb{R}^{1 \times H \times W}$ and the semantic boundary prediction $pred_b \in \mathbb{R}^{N \times H \times W}$. It is worth noting that those semantic boundary maps only retain the classification information of boundary elements, while non-boundary elements are set to 0. Therefore, unlike [4], BSM can not only optimize the

semantic boundary but also further optimize the classification information of the semantic boundary. We believe this can achieve better semantic boundary results. The above procedure can be formulated as follows:

$$\begin{aligned} m_d &= \Upsilon(\Gamma(gt)) \\ gt_b &= m_d \odot gt \\ pred_b &= m_d \odot pred \end{aligned} \tag{7}$$

where $\Gamma(\cdot)$ denotes the 2D convolution with the Laplacian kernel as the weight. $\Upsilon(\cdot)$ denotes the dilation operation.

4 Experiments

4.1 Datasets

Cityscapes. Cityscapes [33] is one of the most popular complex urban street scene datasets. It contains 5,000 fine annotated images and is split into 2,975, 500, and 1,525 images for training, validation, and testing, respectively. The annotation includes 19 classes of annotation for the semantic segmentation task. These images have the same challenging resolution ($1,024 \times 2,048$) for real-time semantic segmentation. For a fair comparison, we only use fine annotated images in our experiments.

CamVid. Cambridge-driving Labeled Video Database (Camvid) [34] is a road scene dataset that contains 701 images and is partitioned into 367 training, 101 validation, and 233 test images. All images share the same resolution of 720×960. The annotation includes 32 categories, of which a subset of 11 categories are used in our experiments. Same setting as [2, 14, 29, 35], we train our model on both the *training* and *validation* sets and validate on the *test* set.

4.2 Implementation details

Data Augmentation. In the training phase, we all apply color jittering, random horizontal flip, random resize, random crop, etc. For the Cityscapes, the random scale ranges in [0.125, 1.5], and the cropped resolution is $512 \times 1,024$. For the CamVid, the random scale ranges in [0.5, 2.5], and the cropped resolution is 720×960.

Training Settings. As a common configuration, we use mini-batch stochastic gradient descent (SGD) [36] as an optimizer which momentum set as 0.9 and the weight decay is $5e^{-4}$. Similar to [37, 38], we also utilize "poly" learning rate policy. In our paper, we set the initial learning rate and power is 0.01 and 0.9, respectively, and the initial rate is multiplied by $(1-\frac{iter}{max\_iter})^{power}$. In addition, we use the warm-up strategy at the first 1,000, and 300 iterations for Cityscapes and CamVid, respectively. For the Cityscapes, we set the batch size as 40, the max iterations are 70,000. For the CamVid, we

Table 1 Performance comparison with and without BSM on Cityscapes *test* set

Model	Road	Sidewalk	Building	Wall	Fence	Pole	Traffic light	Traffic sign	Vegetation	Terrain	Sky	Person	Rider	Car	Truck	Bus	Train	Motorcycle	Bicycle	mIoU (%)
ZMNet-S†	**98.4**	**84.6**	91.6	**47.2**	**50.7**	59.2	**68.3**	72.5	92.6	**70.3**	94.7	82.6	**65.3**	95.1	68.3	74.7	67.3	**62.4**	71.0	74.6
ZMNet-S	98.4	84.4	**91.6**	46.7	50.1	**59.6**	68.2	**72.9**	**92.6**	70.0	**94.8**	**82.7**	65.1	**95.1**	**69.2**	**77.3**	**69.9**	62.2	**71.1**	**74.9**
ZMNet†	**98.4**	**85.0**	92.3	54.5	**55.5**	60.6	69.2	72.9	92.8	**71.0**	95.1	83.2	65.3	95.4	72.5	82.2	75.2	62.8	72.1	76.7
ZMNet	98.4	84.9	**92.4**	**55.9**	55.0	**61.6**	**69.9**	**73.6**	**92.9**	69.9	**95.1**	**83.6**	**66.4**	**95.5**	**73.2**	**85.3**	**79.4**	**63.8**	**72.4**	**77.4**

Symbol † denotes results without BSM. The best results in each category are highlighted in bold

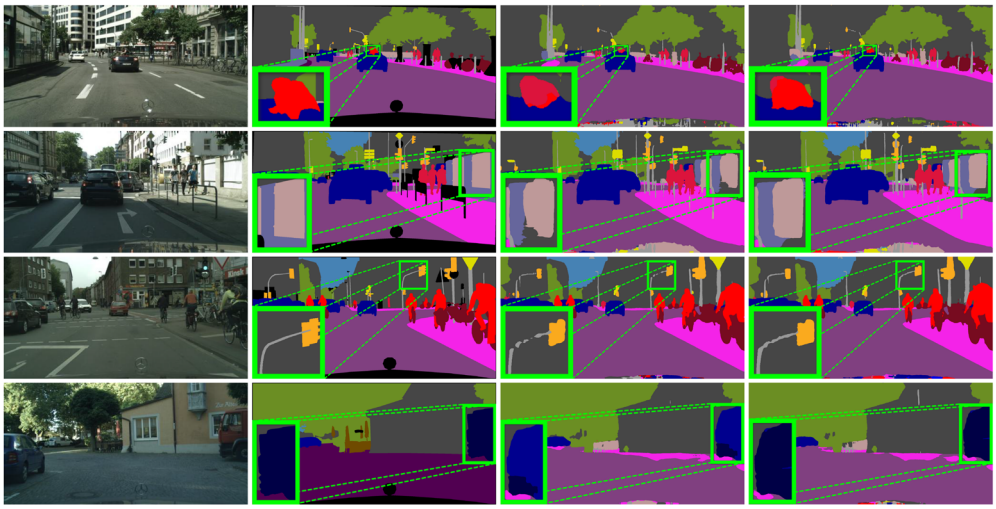

Fig. 6 Visualization of segmentation results with and without BSM on Cityscapes *validation* set. The column with subscript **B** denotes predictions with BSM. Column **a** shows the input images, **b** are the ground-truths of input images, **c**, **d** demonstrate the predictions without and with BSM

Table 2 Performance comparison of using different loss functions in BSM and inserting BSM at different stages

	w/o BSM	$Stage_6$		$Stage_7$		$Stage_8$	
		FL	OHEM	FL	OHEM	FL	OHEM
mIoU(%)	77.2	77.4	77.3	**77.6**	77.3	77.4	77.3

"FL" denotes the BSM with focal loss [32] function. "OHEM" denotes the BSM with OHEM [31] loss function. $Stage_i$ indicates that the BSM is inserted into the i-th stage

set the batch size as 30 and the max iterations are 30,000. We conduct our all training experiments on NVIDIA GeForce 2080Ti.

Inference Settings. Similar to [2, 4], for the Cityscapes dataset, we initially resize the image resolution from the original $1,024 \times 2,048$ to $768 \times 1,536$ for inference, and subsequently, we resize the prediction to the original size of the input. The time taken for resizing is included in our reported inference time. For the CamVid dataset, we take the original image as input. We measure the inference time under PyTorch−1.6, CUDA 10.2, CUDNN 7.6, and TensorRT on a single NVIDIA GeForce 1080Ti GPU with a batch size of 1.

Evaluation Metrics. In this paper, we adopt the mean of class-wise Intersection over Union (mIoU) to evaluate segmentation accuracy and frames per second (FPS) to evaluate inference speed.

4.3 Ablation study

Effectiveness of BSM. To investigate the effectiveness of BSM, we conduct ablation experiments with and without BSM. As shown in Table 1, the model with BSM improves the segmentation accuracy for most categories. Through careful observation, we find that the categories with clear edges and lines, such as poles, traffic signs, trucks, buses, and trains, show the most significant improvement in segmentation accuracy. Specifically, the train category has the most noticeable improvement, with mIoU increasing from 75.2% to 79.4%, a 4.2% improvement.

To further demonstrate the effectiveness of BSM, we present some segmentation examples of models with and without BSM for visual comparison. As shown in Fig. 6, comparing columns (c) and (d), we can see that the model with BSM achieves better semantic segmentation performance than the model without BSM.

Comparison of different loss functions in BSM. In order to validate the rationality of the selected loss function, we conduct experimental comparisons in BSM using both the focal [32] and OHEM [31] loss functions. As shown in Table 2, although both OHEM and focal loss functions are suitable for addressing issues like class imbalance and hard example learning, the experimental results consistently indicate higher accuracy when employing focal loss function compared to OHEM loss function. Therefore, in this paper, we choose focal loss as the loss function for BSM.

Comparison of BSM Application Positions. It is worth noting that BSM is a module specifically designed to improve the accuracy of semantic boundaries during model training, which can be incorporated at various stages. As shown in

Table 3 Compare the effect of using auxiliary segmentation supervision in different decoding stages

$Stage_6$	$Stage_7$	mIoU(%)
		76.8
	✓	76.9
✓		76.9
✓	✓	**77.2**

$Stage_6$ and $Stage_7$, respectively, represent the application of auxiliary segmentation supervision in the 6-th and 7-th stages

Table 4 Comparison with other feature fusion methods on the Cityscapes *validation* set

Methods	mIoU(%)	Params (M)	FPS
STDC2 + EA	76.6	20.8	**100.1**
STDC2 + CAT	76.7	20.7	93.2
STDC2 + AFF [5]	77.1	20.8	94.5
STDC2 + FFM [2]	76.2	20.6	84.1
STDC2 + MLA-FFM (ours)	**77.2**	**20.6**	97.5

EA: Feature fusion implemented by element-wise addition. **CAT**: Feature fusion implemented by channel-wise concatenation. **AFF** and **FFM** represent the fusion method proposed in [5], [2], respectively.

Table 2, when BSM is applied in the 7-th stage, its segmentation accuracy is higher than that of other stages, which increases the mIoU on the Cityscapes *validation* set from 77.2% to 77.6%.

Effectiveness of Auxiliary Segmentation Supervision. The decoder of ZMNet consists of three fusion stages, namely stage 6, stage 7, and stage 8. During the training phase, the main segmentation supervision is conducted in stage 8, while the auxiliary segmentation supervision can be performed in stages 6 and 7. In order to evaluate the impact of auxiliary segmentation supervision on segmentation accuracy, we conduct ablation experiments on the Cityscapes *validation* set. The results of the auxiliary segmentation supervision are shown in Table 3. It can be observed that the incorporation of auxiliary segmentation supervision in either stage 6 or stage 7 contributes to the improvement of segmentation accuracy. The best segmentation performance is achieved when auxiliary segmentation supervision is applied in both stage 6 and stage 7, resulting in an increase of mIoU from 76.8% to 77.2%, which is an improvement of over 0.4%.

Effectiveness of MLA-FFM. To validate the effectiveness of MLA-FFM, we compare it with previously popular feature fusion methods. For a fair comparison, we set the input resolution to $768 \times 1,536$ for all methods and only use the feature maps generated by the last three stages of the encoder for feature fusion. The input and output feature maps are aligned using 3×3 CBR. The settings of the auxiliary segmentation supervision are also consistent.

As shown in Table 4, the feature fusion method of element-wise addition (EA) is the fastest fusion method among all methods. The channel-wise concatenation (CAT) fusion method performs similarly to EA, but both methods ignored the importance of multi-dimension contextual information during fusion. Compared with EA and CAT methods, our proposed MLA-FFM, respectively, improved 0.6% and 0.5% mIoU at the cost of a small speed sacrifice.

AFF [5] aggregates spatial contextual information through spatial pooling during feature fusion. Our method not only aggregates spatial contextual information through spatial aggregation but also aggregates channel contextual information through channel aggregation, and generates a soft attention map by combining these two pieces of contextual information to better fuse features from different levels. By comparison, we can find that MLA-FFM is faster and more accurate than AFF [5].

FFM [2] uses contextual information of the channel dimension to perform attention weighting on the fused features. Compared with FFM [2], our method is 13.4 FPS faster and has a higher mIoU by 0.7%. We believe that such improvement is due to our use of multi-dimensional contextual information to fuse feature information.

Comparison of different pooling operations in MLA-FFM. Firstly, as shown in the first row of Table 5, we remove all pooling operations and corresponding matrix multiplication operations in MLA-FFM. We can find that the modified MLA-FFM does not show any significant impact on speed but significantly reduce the segmentation accuracy (by 1.5% mIoU). This indicates that multi-dimensional contextual information is beneficial for feature fusion.

Additionally, we compare the effects of different pooling operations in MLA-FFM. As shown in rows two to five of Table 5, the impact of different pooling operations on the results is not significant. Among them, spatial dimension aggregation implemented by max-pooling and channel dimension aggregation implemented by average-pooling achieve the best performance.

Comparison of different trade-off parameter ξ values. In our paper, semantic segmentation loss and semantic boundary loss jointly optimize our model. We employ a parameter ξ to balance the trade-off between semantic segmentation loss and semantic boundary loss. As shown in Fig. 7, the accuracy of our network is improved when ξ takes smaller value such as 0.3, 0.6, or 1.0. However, when the value of ξ is set to a larger value, such as 1.5 or 2.0, the accuracy of the network decreases. This is because when ξ is larger, the network may pay too much attention to the semantic boundary area while reducing attention to non-semantic boundary areas, resulting in a decrease in overall segmentation accuracy. Experimental results show that when ξ is set to 0.6, its segmentation performance is optimal.

Table 5 Comparison of different pooling operations in MLA-FFM

Spatial aggregation	Channel aggregation	mIoU(%)	Params (M)	FPS
–	–	75.7	20.6	**98.1**
Max-pooling	Max-pooling	77.2	20.6	97.5
Average-pooling	Average-pooling	77.5	20.6	97.5
Max-pooling	Average-pooling	**77.6**	20.6	97.5
Average-pooling	Max-pooling	77.5	20.6	97.5

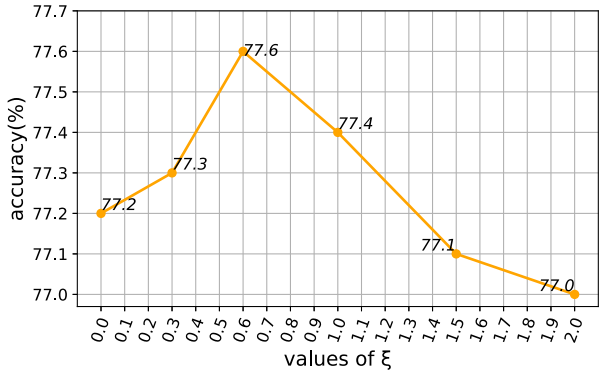

Fig. 7 Comparison of different trade-off parameter ξ values

Table 6 Comparison with other state-of-the-art real-time methods on the Cityscapes dataset

Model	Backbone	mIoU(%)		
		Val	Test	FPS
SwiftNet [35]	ResNet-18	75.4	75.5	39.9
BiSeNetV1 [2]	Xception39	69.0	68.4	105.8
BiSeNetV1-L [2]	ResNet-18	74.8	74.7	65.5
BiSeNetV2 [3]	–	73.4	72.6	156
BiSeNetV2-L [3]	–	75.8	75.3	47.3
SFNet [28]	DF1	–	74.5	121
HMSeg [39]	–	–	74.3	83.2
TinyHMSeg [39]	–	–	71.4	172.4
STDC1-Seg50 [4]	STDC1	72.2	71.9	**250.4**
STDC2-Seg50 [4]	STDC2	74.2	73.4	188.6
STDC1-Seg75 [4]	STDC1	74.5	75.3	126.7
STDC2-Seg75 [4]	STDC2	77.0	76.8	97.0
GDN [13]	GDN	-	75.6	113
LSNet [40]	ResNet-18	-	73.9	130.2
SENet [26]	MobileNetV2	-	77.2	30.8
BFMNet1-M† [10]	MobileNetV3	76.2	75.7	72.1
BFMNet2-M† [10]	ResNet-18	**77.9**	77.7	63.7
ZMNet-R18 (ours)	ResNet-18	75.0	75.2	80
ZMNet-S (ours)	STDC1	75.0	74.9	130.4
ZMNet (ours)	STDC2	77.6	77.4	97.5

Methods marked with "†" indicate that they measure FPS on a single NVIDIA GeForce 2080ti GPU

4.4 Compare with state-of-the-arts

Results on Cityscapes. To demonstrate the ability of our proposed ZMNet for tackling segmentation on complex street scenes, in Table 6, we illustrate the accuracy and speed of ZMNet on Cityscapes *validation* and *test* sets. We propose ZMNet-S and ZMNet, which use STDC1 [4] and STDC2 [4] as backbones, respectively. ZMNet-S has a faster inference speed, while ZMNet has higher segmentation accuracy. To further demonstrate the effectiveness of our decoder, we also propose ZMNet-R18, which uses ResNet-18 [41] as the backbone. The backbones used in our experiments are pre-trained on the ImageNet [42] dataset. To make a fair comparison, at the test phase, we use both *training* and *validation* sets and make the evaluation on the *test* set. In the end, we submit our *test* set predictions to the Cityscapes online evaluation server for detailed accuracy results. As shown in Table 6, ZMNet-R18 achieves 75% and 75.2% mIoU with 79.7 FPS on *validation* set and *test* set, respectively. Although its accuracy is slightly lower than SwiftNet [35], its inference speed is almost twice as fast as SwiftNet [35]. ZMNet-S with 130.4 FPS achieves 75% and 74.9% mIoU on the *validation* set and *test* set, respectively, showing competitive performance compared to most methods. Moreover, our ZMNet achieves 77.6% mIoU on the *validation* set and 77.4% mIoU on *test* set with 97.5 FPS, respectively. Compared to SENet [26], not only does ZMNet have higher segmentation accuracy than SENet, but its inference speed is also three times that of SENet. Although ZMNet is slightly inferior to BFMNet2-M [10] on segmentation accuracy, its inference speed is much faster than BFMNet2-M.

Results on CamVid. We also present the segmentation accuracy and inference speed results of our proposed methods on the CamVid dataset. As shown in Table 7, ZMNet-R18 achieves a 73.8% mIoU at 133 FPS, which achieves a more competitive speed–accuracy trade-off than other methods that use the same ResNet-18 backbone. ZMNet-S achieve 72.6% mIoU on the CamVid *test* set at a speed of 213.7 FPS, which is 8.1% faster than STDC1-Seg [4] while sacrificing a small amount of accuracy. ZMNet achieved a remarkable trade-off between accuracy and speed among all methods

Table 7 Comparison with other state-of-the-art real-time methods on the CamVid dataset

Model	Encoder	mIoU(%)	FPS
ENet [43]	–	51.3	61.2
BiSeNetV1 [2]	Xception39	65.6	175
BiSeNetV1-L [2]	ResNet-18	68.7	116.3
BiSeNetV2 [3]	–	72.4	124.5
BiSeNetV2-L [3]	–	73.2	32.7
STDC1-Seg [4]	STDC1	73.0	197.6
STDC2-Seg [4]	STDC2	73.9	152.2
LSNet [40]	ResNet-18	72.3	104.5
BFMNet1† [10]	MobileNetV3	74.4	118.6
BFMNet2† [10]	ResNet-18	**75.6**	98.8
ZMNet-R18 (ours)	ResNet-18	73.8	133
ZMNet-S (ours)	STDC1	72.6	**213.7**
ZMNet (ours)	STDC2	74.0	156.6

Methods marked with "†" indicate that they measure FPS on a single NVIDIA GeForce 2080ti GPU

by achieving 74% mIoU at a speed of 156.6 FPS. Compared to state-of-the-art methods like STDC [4] and BFMNet [10], our method demonstrates competitiveness in either segmentation accuracy or inference speed. The experiment results demonstrate that our method achieves a good balance between inference speed and segmentation accuracy.

5 Conclusions

In this paper, we propose a novel real-time network ZMNet to balance segmentation accuracy and inference speed in real-time semantic segmentation. First, we design a lightweight feature fusion module MLA-FFM to effectively fuse features from adjacent levels. Second, we propose a semantic boundary supervision module BSM to further improve the accuracy of semantic boundaries. Experiments show that our proposed ZMNet achieves a state-of-the-art balance between segmentation accuracy and inference speed. Currently, our method has only been tested for real-time street scene segmentation in ideal conditions. In the future, we plan to expand our approach to encompass real-time scene segmentation in non-ideal conditions such as rainy or foggy weather, aiming to meet the demands of a wider range of practical application scenarios.

Acknowledgements This work was supported in part by National Natural Science Foundation of China (No. 61906049), and in part by Guangzhou Higher Education Teaching Reform Project (No. 2022JXGG016).

Declarations

Conflict of interest The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

1. Howard, A.G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., Adam, H.: Mobilenets: efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv:1704.04861 (2017)
2. Yu, C., Wang, J., Peng, C., Gao, C., Yu, G., Sang, N.: Bisenet: bilateral segmentation network for real-time semantic segmentation. In: Proceedings of the European Conference on Computer Vision (ECCV), pp. 325–341 (2018)
3. Yu, C., Gao, C., Wang, J., Yu, G., Shen, C., Sang, N.: Bisenet v2: bilateral network with guided aggregation for real-time semantic segmentation. Int. J. Comput. Vis. **129**(11), 3051–3068 (2021)
4. Fan, M., Lai, S., Huang, J., Wei, X., Chai, Z., Luo, J., Wei, X.: Rethinking bisenet for real-time semantic segmentation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 9716–9725 (2021)
5. Dai, Y., Gieseke, F., Oehmcke, S., Wu, Y., Barnard, K.: Attentional feature fusion. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, pp. 3560–3569 (2021)
6. Yang, M., Yu, K., Zhang, C., Li, Z., Yang, K.: Denseaspp for semantic segmentation in street scenes. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3684–3692 (2018)
7. Yu, F., Wang, D., Shelhamer, E., Darrell, T.: Deep layer aggregation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2403–2412 (2018)
8. Yin, H., Xie, W., Zhang, J., Zhang, Y., Zhu, W., Gao, J., Shao, Y., Li, Y.: Dual context network for real-time semantic segmentation. Mach. Vis. Appl. **34**(2), 22 (2023)
9. Zhen, M., Wang, J., Zhou, L., Li, S., Shen, T., Shang, J., Fang, T., Quan, L.: Joint semantic segmentation and boundary detection using iterative pyramid contexts. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 13666–13675 (2020)
10. Liu, J., Zhang, F., Zhou, Z., Wang, J.: Bfmnet: bilateral feature fusion network with multi-scale context aggregation for real-time semantic segmentation. Neurocomputing **521**, 27–40 (2023)
11. Zhang, X., Zhou, X., Lin, M., Sun, J.: Shufflenet: an extremely efficient convolutional neural network for mobile devices. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 6848–6856 (2018)
12. Poudel, R.P., Liwicki, S., Cipolla, R.: Fast-scnn: fast semantic segmentation network. arXiv preprint arXiv:1902.04502 (2019)
13. Luo, D., Kang, H., Long, J., Zhang, J., Liu, X., Quan, T.: Gdn: guided down-sampling network for real-time semantic segmentation. Neurocomputing **520**, 205–215 (2023)
14. Hong, Y., Pan, H., Sun, W., Jia, Y.: Deep dual-resolution networks for real-time and accurate semantic segmentation of road scenes. arXiv preprint arXiv:2101.06085 (2021)

15. Zhang, W., Huang, Z., Luo, G., Chen, T., Wang, X., Liu, W., Yu, G., Shen, C.: Topformer: Token pyramid transformer for mobile semantic segmentation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 12083–12093 (2022)
16. Weng, X., Yan, Y., Chen, S., Xue, J.-H., Wang, H.: Stage-aware feature alignment network for real-time semantic segmentation of street scenes. IEEE Trans. Circuits Syst. Video Technol. **32**(7), 4444–4459 (2021)
17. Weng, X., Yan, Y., Dong, G., Shu, C., Wang, B., Wang, H., Zhang, J.: Deep multi-branch aggregation network for real-time semantic segmentation in street scenes. IEEE Trans. Intell. Transp. Syst. **23**(10), 17224–17240 (2022)
18. Li, Y., Chang, Y., Yu, C., Yan, L.: Close the loop: a unified bottom-up and top-down paradigm for joint image deraining and segmentation. Proc. AAAI Conf. Artif. Intell. **36**, 1438–1446 (2022)
19. Zhao, S., Huang, W., Yang, M., Liu, W.: real rainy scene analysis: A dual-module benchmark for image deraining and segmentation. In: 2023 IEEE International Conference on Multimedia and Expo Workshops (ICMEW), 69–74 (2023). IEEE
20. Sun, S., Ren, W., Li, J., Zhang, K., Liang, M., Cao, X.: Event-aware video deraining via multi-patch progressive learning. IEEE Trans. Image Process. (2023)
21. Wang, F., Jiang, M., Qian, C., Yang, S., Li, C., Zhang, H., Wang, X., Tang, X.: Residual attention network for image classification. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3156–3164 (2017)
22. Hu, J., Shen, L., Sun, G.: Squeeze-and-excitation networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 7132–7141 (2018)
23. Park, J., Woo, S., Lee, J.-Y., Kweon, I.S.: Bam: Bottleneck attention module. arXiv preprint arXiv:1807.06514 (2018)
24. Woo, S., Park, J., Lee, J.-Y., Kweon, I.S.: Cbam: convolutional block attention module. In: Proceedings of the European Conference on Computer Vision (ECCV), pp. 3–19 (2018)
25. Zhang, H., Wu, C., Zhang, Z., Zhu, Y., Lin, H., Zhang, Z., Sun, Y., He, T., Mueller, J., Manmatha, R., : Resnest: split-attention networks. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 2736–2746 (2022)
26. Huang, Y., Shi, P., He, H., He, H., Zhao, B.: Senet: spatial information enhancement for semantic segmentation neural networks. Vis. Comput. 1–14 (2023)
27. Jiang, M., Zhai, F., Kong, J.: Sparse attention module for optimizing semantic segmentation performance combined with a multi-task feature extraction network. Vis. Comput. **38**(7), 2473–2488 (2022)
28. Li, X., You, A., Zhu, Z., Zhao, H., Yang, M., Yang, K., Tan, S., Tong, Y.: Semantic flow for fast and accurate scene parsing. In: European Conference on Computer Vision, pp. 775–793 (2020). Springer
29. Li, H., Xiong, P., Fan, H., Sun, J.: Dfanet: deep feature aggregation for real-time semantic segmentation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 9522–9531 (2019)
30. Huang, Z., Wei, Y., Wang, X., Liu, W., Huang, T.S., Shi, H.: Alignseg: feature-aligned segmentation networks. IEEE Trans. Pattern Anal. Mach. Intell. **44**(1), 550–557 (2021)
31. Shrivastava, A., Gupta, A., Girshick, R.: Training region-based object detectors with online hard example mining. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 761–769 (2016)
32. Lin, T.-Y., Goyal, P., Girshick, R., He, K., Dollár, P.: Focal loss for dense object detection. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 2980–2988 (2017)
33. Cordts, M., Omran, M., Ramos, S., Rehfeld, T., Enzweiler, M., Benenson, R., Franke, U., Roth, S., Schiele, B.: The cityscapes dataset for semantic urban scene understanding. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3213–3223 (2016)
34. Brostow, G.J., Shotton, J., Fauqueur, J., Cipolla, R.: Segmentation and recognition using structure from motion point clouds. In: European Conference on Computer Vision, pp. 44–57 (2008). Springer
35. Orsic, M., Kreso, I., Bevandic, P., Segvic, S.: In defense of pre-trained imagenet architectures for real-time semantic segmentation of road-driving images. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 12607–12616 (2019)
36. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. Commun. ACM **60**(6), 84–90 (2017)
37. Chen, L.-C., Papandreou, G., Kokkinos, I., Murphy, K., Yuille, A.L.: Deeplab: semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. IEEE Trans. Pattern Anal. Mach. Intell. **40**(4), 834–848 (2017)
38. Chen, L.-C., Papandreou, G., Schroff, F., Adam, H.: rethinking atrous convolution for semantic image segmentation. arXiv preprint arXiv:1706.05587 (2017)
39. Li, P., Dong, X., Yu, X., Yang, Y.: When humans meet machines: towards efficient segmentation networks. In: The 31st British Machine Vision Virtual Conference (2020)
40. Sheng, P., Shi, Y., Liu, X., Jin, H.: Lsnet: real-time attention semantic segmentation network with linear complexity. Neurocomputing **509**, 94–101 (2022)
41. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778 (2016)
42. Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., Fei-Fei, L.: Imagenet: a large-scale hierarchical image database. In: 2009 IEEE Conference on Computer Vision and Pattern Recognition, pp. 248–255 (2009). IEEE
43. Paszke, A., Chaurasia, A., Kim, S., Culurciello, E.: Enet: a deep neural network architecture for real-time semantic segmentation. arXiv preprint arXiv:1606.02147 (2016)

Ya Li is an associate professor in School of Computer Science and Cyber Engineering, Guangzhou University, Guangzhou, China. She received her B.E. degree from Zhengzhou University, Zhengzhou, China, in 2002, M.E. degree from Southwest Jiaotong University, Chengdu, China, in 2006, and Ph.D. degree from Sun Yat-sen University, Guangzhou, in 2015. Her current research focuses on computer vision and machine learning.

Ziming Li received his B.E. degree from the School of Computer Science and Educational Software, Guangzhou University, Guangzhou, China, in 2018. He is currently pursuing the M.E. degree in the School of Computer Science and Network Engineering, Guangzhou University, Guangzhou, China. His current research interests include computer vision and machine learning.

Huiwang Liu received the B.E. degree from the School of Software Engineering, Jiangxi Normal University, Nanchang, China, in 2020. He received the M.E. degree in the School of Computer Science and Network Engineering, Guangzhou University, China, in 2023. His research interests include computer vision and machine learning..

Qing Wang is an associate professor of School of Computer and Engineering of Sun Yat-sen University. His research interests include software engineering and machine learning.