Boundary-Guided Lightweight Semantic Segmentation With Multi-Scale Semantic Context

Quan Zhou, *Senior Member, IEEE*, Linjie Wang, Guangwei Gao, *Senior Member, IEEE*, Bin Kang, *Member, IEEE*, Weihua Ou, and Huimin Lu, *Senior Member, IEEE*

***Abstract*—Lightweight semantic segmentation plays an essential role in image signal processing that is beneficial to many multimedia applications, such as self-driving, robotic vision, and virtual reality. Due to the powerful capability to encode image details and semantics, many lightweight dual-resolution networks have been proposed in recent years for semantic segmentation. In spite of achieving remarkable progresses, they often ignore semantic context ranged from different scales. Furthermore, most of them always neglect the object boundaries, serving as a significant assistance for lightweight semantic segmentation. To alleviate these problems, this paper develops a Boundary-guide dual-resolution lightweight network with multi-scale Semantic Context, called BSCNet, for semantic segmentation. Specifically, to enhance the capability of feature representation, an Extremely Lightweight Pyramid Pooling Module (ELPPM) is designed to capture multi-scale semantic context at the top of low-resolution branch of BSCNet. In addition, to increase feature similarity of the same object while keeping feature discrimination of different objects, pixel information is propagated throughout the entire object area using a simple Boundary Auxiliary Fusion Module (BAFM), where the predicted object boundaries are served as high-level guidance to refine low-level convolutional features. The comprehensive experimental results have demonstrated that our BSCNet is simple and effective, achieving state-of-the-art trade-off in terms of segmentation accuracy and running efficiency on CityScapes, CamVid, and KITTI datasets.**

***Index Terms*—Lightweight semantic segmentation, boundary detection, feature propagation, convolutional neural networks (CNNs).**

Manuscript received 30 September 2023; revised 6 February 2024; accepted 26 February 2024. Date of publication 12 March 2024; date of current version 24 April 2024. This work was supported in part by the National Natural Science Foundation of China under Grant 61876093, Grant 62171232, and Grant 62262005. The Associate Editor coordinating the review of this manuscript and approving it for publication was Dr. Yi Yu. *(Corresponding authors: Quan Zhou; Guangwei Gao.)*

Quan Zhou is with the National Engineering Research Center of Communications and Networking, Nanjing University of Posts & Telecommunications, Nanjing 210003, China, and also with the Jiangsu Key Lab of Image and Video Understanding for Social Security, and Key Lab of Intelligent Perception and Systems for High-Dimensional Information of Ministry of Education, Nanjing University of Science and Technology, Nanjing 210094, China (e-mail: quan.zhou@njupt.edu.cn).

Linjie Wang is with the New H3C Technologies Company Ltd., Hangzhou 310052, China (e-mail: 1020010524@njupt.edu.cn).

Guangwei Gao is with the Institute of Advanced Technology, Nanjing University of Posts & Telecommunications, Nanjing 210003, China (e-mail: csgwgao@njupt.edu.cn).

Bin Kang is with the Department of Internet of Things, Nanjing University of Posts & Telecommunications, Nanjing 210003, China (e-mail: kangbin@njupt.edu.cn).

Weihua Ou is with the School of Big Data and Computer Science, Guizhou Normal University, Guiyang 550001, China (e-mail: ouweihuahust@gmail.com).

Huimin Lu is with the School of Automation, Southeast University, Nanjing 210096, China (e-mail: dr.huimin.lu@ieee.org).

Digital Object Identifier 10.1109/TMM.2024.3372835

I. Introduction

LIGHTWEIGHT semantic segmentation, as a fundamental and challenging task in the field of signal processing community, plays a vital role in many real-world multimedia applications, such as self-driving, robotic vision, and virtual reality. The goal of semantic segmentation is to assign a unique semantic category label to each pixel in images. With the rapid development of convolutional neural networks (CNNs), various advanced high-accuracy CNNs [1], [2], [3], [4], [5] have been proposed for image semantic segmentation. In spite of achieving remarkable progresses, those high-accuracy networks often involve heavy model size and huge amount of computational costs, which are unsuitable for many real-world applications that require online estimations and real-time decisions [6], [7].

To this end, many researchers prefer to design lightweight networks [6], [7], [8], [9], leveraging segmentation accuracy and implementing efficiency at the same time. Most of them employ lightweight encoder-decoder architecture in a single path manner [6], [7], where the convolutional features are firstly down-sampled in encoder, and then gradually recovered in decoder. Among them, depth-wise [7], group-wise [10] and factorized-wise [6] convolutions are widely utilizes to compress network size. Considering image details are hard to accurately recover via deconvolution and upsampling operators in decoder, some recent studies [8], [9], often called dual-resolution networks, design an extra parallel high-resolution branch to maintain spatial detailed information. For instance, BiSeNet [8] divides the network into spatial and context paths separately, both of which involve lightweight architecture. BiSeNetV2 [9], as an extension of [8], proposes a finer way to fuse features from two branches, leading to the great reduction of model size and computational costs. Although these advanced and lightweight networks have achieved impressive segmentation results, they inherently suffer from following limitations:

- Whether single-path [11], [12] nor dual-path lightweight networks [8], [9], they often capture contextual features solely relying on single scale clues. Correctly classifying image pixels, however, not only depends on short-ranged low-level details, but also may rely on long-ranged high-level semantics. Although some high-accuracy networks have integrated intermediate convolutional features [1], [2] via element-wise addition or concatenation, such fusion

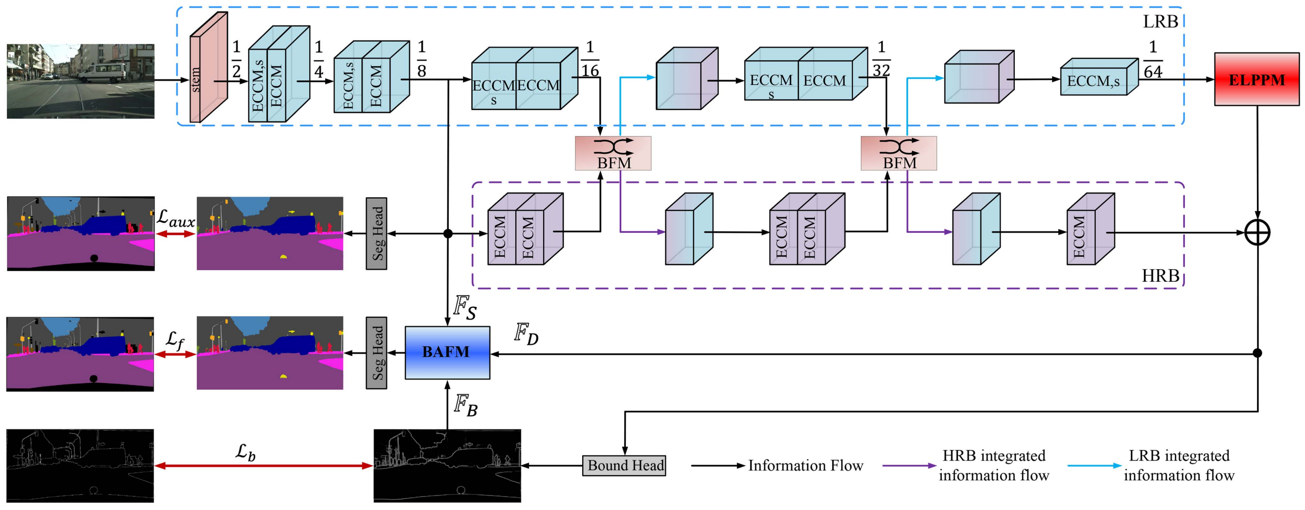


Fig. 1. Overall architecture of BSCNet. More specifically, the BSCNet is mainly composed of high-resolution branch (HRB, marked by purple dashed box) and low-resolution branch (LRB, marked by blue dashed box). The ECCM, ELPPM, BFM, and BAFM stand for efficient compact convolutions module, extremely lightweight Pyramid pooling module, bilateral fusion module, and boundary auxiliary fusion module, respectively. (Best viewed in color).

strategy is too weak to represent multi-scale context. The alternative approaches usually utilize image pyramid [13] or self-attention [14], [15] to encode multi-scale context, yet they both suffer from heavy computations that are unsuitable for lightweight semantic segmentation.

- As an important auxiliary clue, object boundary information has been widely used to develop recent high-accuracy segmentation networks [16], [17], where object shapes are well delineated. These networks, however, require to design an extra modules [18] or sub-networks [16], [19] to learn boundary cues, inevitably leading to great increase of model size and computational costs. As a result, how to encode object boundaries as an additional assistance in an effective and efficient manner still remains unclear for lightweight semantic segmentation.

This paper makes an effort to address these limitations by developing a dual-resolution network, named BSCNet, leveraging multi-scale semantic context and object boundary auxiliary for lightweight semantic segmentation. As shown in Fig. 1, BSCNet also inherits dual-resolution architecture [8], [9], including high-resolution branch (HRB) and low-resolution branch (LRB). HRB maintains low-level fine details as well as LRB encodes high-level image semantics. Instead of using inverted bottlenecks in BiseNet families [8], [9], our BSCNet is mainly composed of a series of Efficient Compact Convolution Modules (ECCMs), which employ multiple depth-wise convolutions to enlarge receptive fields, while remaining smaller computational costs. In addition, different from BiseNets [8], [9] that extract convolution features in two branches independently, two Bilateral Fusion Modules (BFMs) are employed to enhance information communication between HRB and LRB, benefiting to the information flow among features that have variable-resolutions.

To effectively investigate multi-scale semantic context in an efficient manner, as shown in Fig. 1, an Extremely Lightweight Pyramid Pooling Module (ELPPM) is designed on the top of LRB. Inspired from PSPNet [2], ELPPM employs pyramid feature representation to capture semantic context from different scales. It is worthy that ELPPM designs a global-to-local context fusion strategy that integrates neighbouring scale features step-by-step, rather than directly employing simple concatenation operation [2] that is too weak to merge convolution features with different resolutions. The details structure of ELPPM is also motivated from Res2Net [20] that investigates multi-scale information via group-wise channel convolution. Nevertheless, ELPPM achieves more powerful representation capability from resolution perspective, which harvests multi-scale semantic context from hierarchical pyramid pooling features. Most importantly, ELPPM is computationally efficient as it is performed only once, instead of stacking many times in backbone as well as involving heavier 3×3 convolutions used in Res2Net [20].

To explore object boundary cues, BSCNet employs a simple Boundary Auxiliary Fusion Module (BAFM) using estimated boundaries as semantic guidance to help lightweight semantic segmentation. More specifically, a binary boundary map is first predicted through a boundary detection head. To increase the feature similarity of the same object while keeping the feature discrimination of different objects, we explore to propagate information throughout entire object area under the control of the estimated boundary map. As shown in Fig. 1, there are several merits using BAFM. Firstly, as image fine details have already been discarded in deep convolution layers, the shallow layers often contain more richer object shape clues that complement to high-level semantics. Secondly, in contrast to some previous methods [18], [19] that employ additional decoder network or complicated module [16] to learn object boundaries, BAFM still keeps a lightweight design that requires small amount of model parameters and computational costs. In nutshell, the major contributions of our paper are three-fold:

- A lightweight pyramid module, ELPPM, is designed to leverage multi-scale context representation and computational efficiency. Instead of directly integrating intermediate convolution features [1], [2] or self-attention [14], [15], our ELPPM adopts a more powerful pyramid feature representation that fuses multi-scale semantics step-by-step.

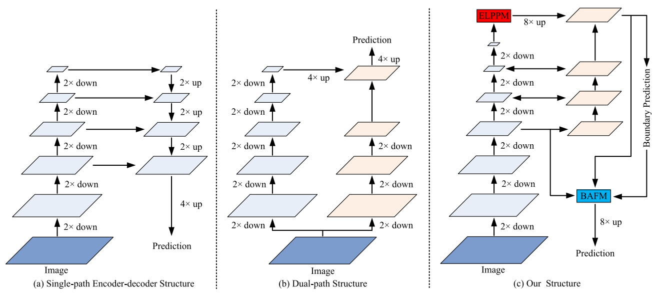


Fig. 2. Lightweight architecture comparison between (a) single-path encoder-decoder networks, (b) dual-path networks, and (c) our method. (Best viewed in color).

Even so, ELPPM is computationally cheap as it only requires very few model size and computational costs.

- Boundary cues are employed in BSCNet as an essential assistance for lightweight semantic segmentation. Through feature propagation, BAFM explicitly encourages feature consistency inside the area of an object, thus implicitly achieving feature inconsistency on both sides of estimated boundaries. Although boundary clues have been widely-used for high-accuracy segmentation network [21], [22], to our best knowledge, it is rarely explored for lightweight semantic segmentation.
- We test BSCNet on three widely-used semantic segmentation datasets: Cityscapes [23], CamVid [24], and KITTI [25]. The exhausted experimental results demonstrate that our method achieves state-of-the-art trade-off in terms of segmentation accuracy and running efficiency. Specifically, BSCNet achieves 78.3%, 79.8%, and 52.4% mIoU on three datasets, respectively, with only 1.5 M model size and 96 FPS and 319 FPS inference speed for 1024×2048 and 960×760 input images, respectively.

The remainder of this paper is organized as follows. Section II briefly reviews the related works. The detailed architecture of BSCNet is introduced in Section III. Section IV shows the experimental settings and results on all datasets. Finally, the concluding remarks and future work are given in Section V.

II. Realted Work

To adapt to real-time applications, a vast number of methods have been proposed to compress semantic segmentation networks, such as quantization [26], pruning [27], knowledge distillation [28], and lightweight model design [8], [9], [29]. As our method belongs to the final category, we briefly review related work in this direction.

A. Lightweight Semantic Segmentation

The existing lightweight semantic segmentation networks can be roughly divided into two categories: single-path [6], [29] and dual-path networks [8], [9]. As shown in Fig. 2(a), the first category often adopts compact convolutions using lightweight encoder-decoder architectures [6], [29], where encoder produces feature representations with repeated spatial reductions, while decoder restores feature resolutions step-by-step to predict segmentation outputs. For instance, SegNet [29] employs a symmetrical encoder-decoder structure with skipped connections to achieve high speed. In [30], a parallel complement network is designed for segmenting road scenes. ERFNet [6] decomposes a 3×3 standard convolution into two successive 1D-factorized convolutions (e.g., 3×1 and 1×3). ESPNet [31] adopts spatial pyramid with dilated convolutions to improve running efficiency. ESPNetV2 [11], as an extension of [31], proposes extremely efficient spatial pyramid unit to enlarge receptive fields. STDCNet [32] designs short-term dense concatenates module to obtain scale-variant receptive fields. HyperSeg [33] combines auto-encoders and hyper-networks for lightweight semantic segmentation. An alternative approach of single path architecture is designing lightweight Transformers [34], [35], [36] for real-time semantic segmentation. For example, CvT [37] incorporates depth-wise convolution into Transformer backbone to compute multi-head self-attention. LETNet [38] designs an efficient U-shape network that inherits the merit of Transformer and CNN. In [35], a pure Transformer backbone is designed to capture hierarchical features using shifted windows. PVT [36] and P2T [34] adopt the pooling operation to accelerate inference speed by reducing the number of input tokens.

On the other hand, as illustrated in Fig. 2(b), the second category usually employs compact dual-resolution architecture, where the resolution of LRB is gradually reduced as usual, while the resolution of HRB is shrank at first, and then remains unchanged to provide rich spatial details as a supplement. For example, the BiSeNet families [8], [9] decouple lightweight network into spatial and context paths separately, parallelly abstracting high-level context semantics and low-level spatial details. ICNet [13] utilizes image pyramid as inputs and builds cascaded networks that incorporate high-level label guidance. Lite-HRNet [39] is another multi-branch network, where an 1×1 convolution is replaced by cross-resolution weighting module in HRNet [40]. MLFNet [41] encodes target contours using a spatial compensation branch. In [42], a triple branch segmentation network is designed by connecting CNNs and proportional-integral derivative controllers.

In contrast to these advanced and lightweight networks that ignore to take object boundary information into account, our BSCNet, as exhibited in Fig. 2(c), designs a BAFM to investigate object boundary clues. In addition, ELPPM harvests various scale semantic context that show enough evidence to make a discriminative decision for each pixel. Finally, whether BAFM nor ELPPM, they are both lightweight and computationally efficient so that BSCNet satisfies the requirement of real-time applications.

B. Multi-Scale Context Fusion

Capturing context information plays a significant role for semantic segmentation [1], [2]. Therefore, there are vast number of high-accuracy [1], [2] and lightweight networks [8], [9], [11] used to combine multi-scale context clues.

In high-accuracy networks [1], [2], the most commonly-used method to capture contextual cues is integrating feature maps from intermediate layers. For instance, FCN [1] captures context using skip connections, where high-level semantics from deep layers are combined with the low-level details from shallow layers. SegNet [29] designs a mirror encoder-decoder structure to harvest multi-scale context, where the convolution features in encoder are duplicated to decoder. Another methods [15], [43] enlarge receptive fields to capture long-ranged contextual dependencies. One solution is to use large kernel size or dilated convolutions. For example, RepLKNet [44] captures context using convolution filters with 31×31 kernel size. The atrous dilated convolution is introduced in [43], [45] to increase the sampling steps in convolution. The alternative solution is self-attention [14] that has shown great success in natural language processing. The non-local network [14] firstly introduces self-attention to CNNs for capturing global context, where the response at each position is a weighted sum of all other positions. DANet [15] captures long-ranged interactions using spatial- and channel-wise attentions. In [3], a channel-wise attention is well explored to capture global context. ANNet [46] proposes spatial pyramid pooling in [14] to accelerate inference speed.

On the other hand, lightweight networks often integrate various-scale context using an extra designed modules [11], [32]. Some approaches [31], [32] prefer to design context capturing module in backbone. For example, ESPNet families [11], [31] adopt spatial pyramid to learn the representations from a larger receptive field. STDCNet [32] adopts a dense-like structure to collect multi-scale context with scalable receptive fields. The alternative approaches [8], [9] fuse context information at the end of backbone. For instance, BiSeNetV1 [8] proposes a feature fusion module to integrate the high-level and low-level cues from context and spatial path. BiSeNetV2 [9] proposes a aggregation layer that enhances mutual connections between detailed and semantic feature representations.

In contrast to high-accuracy networks that always involve huge computational costs to encode context cues, our ELPPM leverages powerful multi-scale context representation and implementation efficiency. Similar to [9], ELPPM is also conducted only once at the end of backbone, yet employing a more lightweight design that integrates various-resolution features step-by-step from one input, instead of exploring correlations among multiple inputs [8], [9].

C. Object Boundary for Semantic Segmentation

As a fundamental low-level visual task in computer vision, boundary detection has been widely-used to help semantic segmentation with the development of CNNs [16], [17], [21]. For instance, OBGNet [21] designs a fully convolution network (FCN) that integrates object boundaries and shapes for semantic segmentation. In [19], object shapes are predicted using an equipotential learning method. GSCNN [16] describes a dual streams network, where shape stream is used to extract object boundaries, while segmentation stream is used to produce segmentation results. RPCNet [17] proposes an iterative pyramid context module to couples semantic segmentation and boundary detection. In contrast to design an additional edge detection branch to estimate object boundaries, some methods prefer to directly produce boundary maps using pre-trained edge detection networks [47], [48], [49], [50]. For example, in [47], the authors integrate side outputs of different convolution layers to predict object boundary. Deng et al. [48] designs a loss function that is capable of penalizing the distance of contour structure similarity between edge prediction and ground-truth. FCL-Net [49] exploits semantic information from deep layers to facilitate fine-scale feature learning. In [50], a PNT-Edge network learns a pixel-wise transition to detect edges under the scenario of noisy labels. The final alternative approaches that investigate object boundaries mainly focus on post-processing, scuh as DeepLab models [45] that restrict segmentation outputs through a fully connected conditional random fields (CRFs).

Unlike these high-accuracy networks [16], [17], [19], [21] that always have heavy model size and huge computational overheads, our BSCNet aims at exploring boundary information in a lightweight manner. Specifically, the proposed lightweight BAFM utilizes the estimated binary boundary map to guide convolution features, leading to the improvement of segmentation performance with limited increase of model parameters and computational costs.

III. Method

This section first elaborates on the detailed architecture of BSCNet, together with its core units ECCM and BFM. Thereafter, we introduces ELPPM and BAFM, used to extract multi-scale context and object boundaries as additional assistance.

A. Network Architecture

The overall architecture of BSCNet is depicted in Fig. 1. BSCNet still inherits the dual-resolution architecture [8], [9], where HRB remains low-level fine details as well as LRB captures high-level semantic cues. Except ELPPM and BAFM, two paths are mainly constructed by a series of ECCMs and two BFMs. ECCM can explore larger receptive fields with very fewer model size and smaller computational costs. On the other hand, BFM strengthens information exchange between two paths using cross-resolution feature integration.

The detailed structure of BSCNet backbone is given in Table I. Similar to traditional single-path segmentation network [6], [29], LRB employs a stem and multiple ECCMs wtih stride 2, gradually producing convolution feature maps that have resolutions of $\frac{1}{2}$, $\frac{1}{4}$, $\frac{1}{8}$, $\frac{1}{16}$, $\frac{1}{32}$ and $\frac{1}{64}$ with respect to input image. On the other hand, HRB keeps a relatively high resolution of LRB, maintaining unchanged feature resolution that is $\frac{1}{8}$ of input size. Among two paths, two BFMs are used to enhance cross-resolution feature integration and communication. At the top of LRB, ELPPM is used to encodes multi-scale semantic context, whose resolution is enlarged 8× with equal dimension for exactly integrating with the output of HRB, used to predict boundary map that are helpful to accurately delineated object shapes. As shown in Fig. 1, the proposed BAFM is added at the end of entire BSCNet to estimate final segmentation outputs, receiving the supervision from the associated ground truths. Furthermore, as preliminary features are always inaccurate and coarse [2], [45], we introduce an auxiliary supervision head to learn more accurate

TABLE I
DETAILED ARCHITECTURE OF BACKBONE IN BSCNET

Layer	LRB	HRB	Operation (LRB)	Operation (HRB)
1	$512 \times 512 \times 16$		stem(3×3, s)	
2	$256 \times 256 \times 32$		ECCM, s	
3	$256 \times 256 \times 32$		ECCM	
4	$128 \times 128 \times 64$		ECCM, s	
5	$128 \times 128 \times 64$		ECCM	
6	$64 \times 64 \times 128$	$128 \times 128 \times 64$	ECCM, s	ECCM
7	$64 \times 64 \times 128$	$128 \times 128 \times 64$	ECCM	ECCM
8	$64 \times 64 \times 128$	$128 \times 128 \times 64$	BFM	
9	$32 \times 32 \times 256$	$128 \times 128 \times 64$	ECCM, s	ECCM
10	$32 \times 32 \times 256$	$128 \times 128 \times 64$	ECCM	ECCM
11	$32 \times 32 \times 256$	$128 \times 128 \times 64$	BFM	
12	$16 \times 16 \times 512$	$128 \times 128 \times 128$	ECCM, s	ECCM

s Denotes stride 2, LRB and HRB denote low-irresolution and high-irresolution branch, respectively.

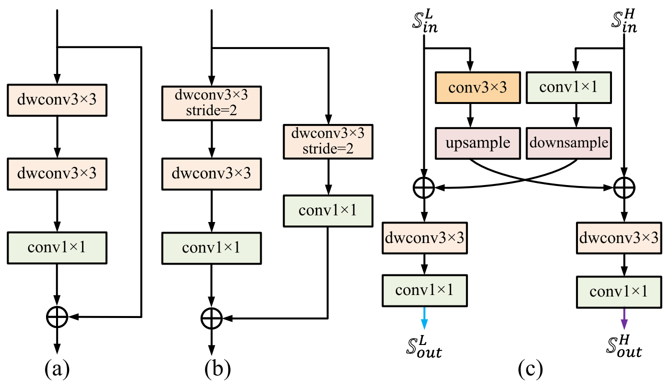
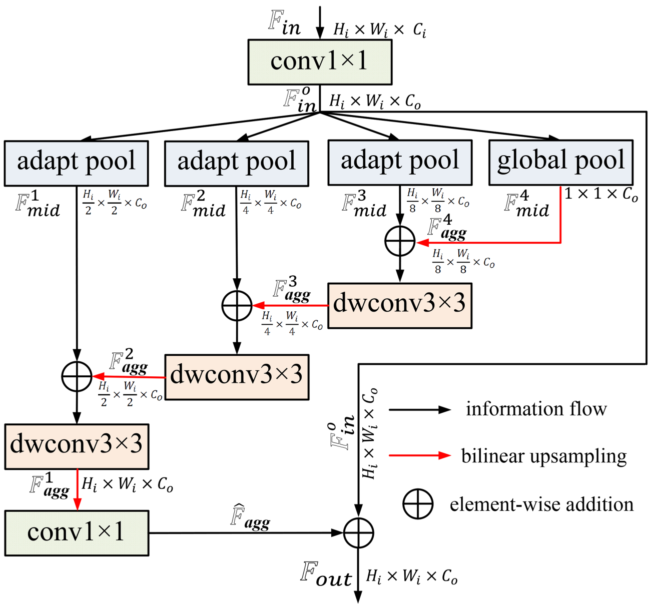


Fig. 3. Overview of the units used in backbone of BSCNet. (a) ECCM; (b) Stride ECCM ($s = 2$); And (c) BFM. The blue and purple arrows in (c) are consistent with the information flow shown in Fig. 1. (Best viewed in color).



$\mathbb{F}_{in}$
$H_i \times W_i \times C_i$
conv1×1
$\mathbb{F}_{in}^{o}$
$H_i \times W_i \times C_o$
adapt pool
adapt pool
adapt pool
global pool
$\mathbb{F}_{mid}^{1}$
$\frac{H_i}{2} \times \frac{W_i}{2} \times C_o$
$\mathbb{F}_{mid}^{2}$
$\frac{H_i}{4} \times \frac{W_i}{4} \times C_o$
$\mathbb{F}_{mid}^{3}$
$\frac{H_i}{8} \times \frac{W_i}{8} \times C_o$
$\mathbb{F}_{mid}^{4}$
$1 \times 1 \times C_o$
$\mathbb{F}_{agg}^{4}$
$\frac{H_i}{8} \times \frac{W_i}{8} \times C_o$
dwconv3×3
$\mathbb{F}_{agg}^{3}$
$\frac{H_i}{4} \times \frac{W_i}{4} \times C_o$
dwconv3×3
$\mathbb{F}_{agg}^{2}$
$\frac{H_i}{2} \times \frac{W_i}{2} \times C_o$
dwconv3×3
$\mathbb{F}_{agg}^{1}$
$H_i \times W_i \times C_o$
conv1×1
$\widehat{\mathbb{F}}_{agg}$
$\mathbb{F}_{in}^{o}$
$H_i \times W_i \times C_o$
$\mathbb{F}_{out}$
$H_i \times W_i \times C_o$
information flow
bilinear upsampling
element-wise addition

Fig. 4. Detailed architecture of ELPPM. (Best viewed in color).

features from shallow convolution layers. Next, we introduce ECCM and BFM in details, respectively. **ECCM.** As illustrated in Fig. 3(a), ECCM follows a residual structure, including transformed and identity branch that leverage compact convolutions and residual connections. The transformed branch serves as a residual function, while the identity branch is used to facilitate end-to-end model training. In transformed branch, two 3×3 depth-wise convolutions are sequentially adopted to enlarge receptive fields, yet resulting in the independent features among all channels. Thus an 1×1 point-wise convolution is followed, recovering channel dependencies using a linear combination. Note the two successive 3×3 depth-wise convolutions achieve same receptive field of 5×5 depth-wise convolutions [51]. However, ECCM is more lightweight, as it only requires $2 \times 3 \times 3 = 18$ parameters, while [51] needs $5 \times 5 = 25$ parameters. Fig. 3(b) also exhibits the stride version of ECCM, used to reduce feature resolutions and synchronously increase channel numbers. Concretely, two 3×3 stride depth-wise convolutions are first utilized in both branches, respectively, resulting in the reduced $2\times$ resolution of outputs. Thereafter, the channel numbers of the outputs are expanded $2\times$, fulfilled by the following 1×1 point-wise convolutions in two branches.

BFM. As depicted in Fig. 3(c), BFM serves as a bridge to enable communications between HRB and LRB. Let $\mathbb{S}_{in}^{H} \in \mathbb{R}^{H \times W \times C}$ and $\mathbb{S}_{in}^{L} \in \mathbb{R}^{\frac{H}{n} \times \frac{W}{n} \times nC}, n \in \{2, 4\}$, be inputs of BFM, and $\mathbb{S}_{out}^{H} \in \mathbb{R}^{H \times W \times C}$ and $\mathbb{S}_{out}^{L} \in \mathbb{R}^{\frac{H}{n} \times \frac{W}{n} \times nC}$ be outputs of BFM, respectively, where $H \times W$ stands for input resolution, and C denotes channel number. Concretely, $\mathbb{S}_{in}^{H}$ first passes through an 1×1 convolution, and then downsampled with equal dimensions for following fusion with $\mathbb{S}_{in}^{L}$. On the other hand, $\mathbb{S}_{in}^{L}$ is fed into a 3×3 convolution, and then upsampled with equal dimensions for next feature integration with $\mathbb{S}_{in}^{H}$. Finally, the combined features sequentially undergo a depth-wise 3×3 and point-wise 1×1 convolutions, producing the outputs $\mathbb{S}_{out}^{L}$ and $\mathbb{S}_{out}^{H}$ for LRB and HRB, respectively.

B. ELPPM

Harvesting context information within different ranges plays a significant role for lightweight semantic segmentation [8], [9]. Some previous methods [1], [2] aggregate intermediate convolution features that have different resolutions, yet they are too weak to capture multi-scale context. Lots of alternative efforts have been proposed to encode multi-scale context by enlarging respective fields [43], [44], or computing dense attention maps [15], [46]. These methods, however, require large amount of computations that are not suitable for lightweight semantic segmentation. In order to address these limitations, this section describes ELPPM that leverages computational efficiency and multi-scale contextual representations. More specifically, ELPPM adopts a simple yet powerful hierarchical feature pyramid to represent multi-scale context, facilitating context fusion in a global-to-local manner. To reduce model size and accelerate computing speed, the depth-wise convolution is used in ELPPM to integrate neighboring-scale context features step-by-step.

The detailed structure of ELPPM is shown in Fig. 4. Given input features $\mathbb{F}_{in} \in \mathbb{R}^{H_i \times W_i \times C_i}$, where H_i, W_i, and C_i stand for height, width, and channel number of $\mathbb{F}_{in}$, respectively, our

ELPPM begins at an 1×1 convolution that projects $\mathbb{F}_{in}$ into a low-dimensional embedding $\mathbb{F}_{in}^{o} \in \mathbb{R}^{H_i \times W_i \times C_o}$, where C_o is set as $\frac{1}{4}C_i$, empirically. In order to produce hierarchical representations, a series of adaptive stride pooling operations $\mathbf{P}_s^i(\cdot)$ are employed, parallelly producing hierarchical features $\mathbb{F}_{mid}^{i}$, $i \in \{1,2,3\}$ that have resolutions of $\frac{1}{2}$, $\frac{1}{4}$, and $\frac{1}{8}$ with respect to $\mathbb{F}_{in}^{o}$. Moreover, to capture global context, we also use a global pooling operation $\mathbf{P}_g^i(\cdot)$, producing an additional features $\mathbb{F}_{mid}^{4}$ that has the resolution of $1 \times 1 \times C_o$ to represent the entirety of $\mathbb{F}_{in}^{o}$:

$$\mathbb{F}_{mid}^{i} = \begin{cases} \mathbf{P_s^i}(\mathbb{F}_{\mathbf{in}}^{\mathbf{o}}), & i \in \{1,2,3\}; \\ \mathbf{P}_g^i(\mathbb{F}_{in}^{o}), & i = 4; \end{cases} \tag{1}$$

Given these intermediate features $\mathbb{F}_{mid}^{i}$ that encode multi-scale semantic context, a global-to-local fusion strategy is designed to sequentially integrate neighboring scale context. Taking the combination of $\mathbb{F}_{mid}^{3}$ and $\mathbb{F}_{mid}^{4}$ into account, the resolution of $\mathbb{F}_{mid}^{4}$ has to be firstly enlarged with the equal dimensions of $\mathbb{F}_{mid}^{3}$ using bilinear upsampling operation $\mathbf{U}(\cdot)$. Then, the enlarged features $\mathbb{F}_{agg}^{4} \in \mathbb{R}^{\frac{H_i}{8} \times \frac{W_i}{8} \times C_o}$ are combined with $\mathbb{F}_{mid}^{3}$ by element-wise addition, which is ready for following 3×3 depth-wise convolution $\mathbf{DC}(\cdot)$. The output of depth-wise convolution are further upsampled, producing aggregated features $\mathbb{F}_{agg}^{3} \in \mathbb{R}^{\frac{H_i}{4} \times \frac{W_i}{4} \times C_o}$ used for next integration. The entire procedure is repeatedly performed until all scale semantic context features are encoded step-by-step:

$$\mathbb{F}_{agg}^{i} = \begin{cases} \mathbf{U}(\mathbf{DC}(\mathbb{F}_{\mathbf{agg}}^{\mathbf{i+1}} \oplus \mathbb{F}_{\mathbf{mid}}^{\mathbf{i}})), & i \in \{1,2,3\}; \\ \mathbf{U}(\mathbb{F}_{mid}^{i}), & i = 4; \end{cases} \tag{2}$$

Note the depth-wise convolution leads to the independence among feature channels during the whole process, thus $\mathbb{F}_{agg}^{1}$ has to be fed into an 1×1 convolution, producing features $\hat{\mathbb{F}}_{agg}$ that the channel dependencies are well recovered. Finally, the entire context fusion process serves as a residual function that facilitates training ELPPM in an end-to-end manner:

$$\mathbb{F}_{out} = \mathbb{F}_{in}^{o} \oplus \hat{\mathbb{F}}_{agg}, \tag{3}$$

Benefiting from the pyramid feature representation and global-to-local context fusion strategy, ELPPM is able to sequentially capture multi-scale semantic context in an efficient manner. In contrast to Res2Net [20] that encodes multi-scale context using channel-wise group convolution and integration, our ELPPM investigates hierarchical semantic context from the perspective of resolution-wise convolution and combination. Furthermore, ELPPM employs depth-wise convolutions to fuse neighboring scale semantic context, leading to the great reduction of model size and computational costs.

C. Lightweight Segmentation With Boundary Assistance

In spite of widely-used for existing high-accuracy semantic segmentation networks [21], [22], object boundaries are rarely explored for lightweight semantic segmentation. Thus this section focuses on how does BAFM improve the segmentation outputs with the aid of object boundaries. We first elaborate on the detailed architecture of BAFM, and then introduce the loss functions used to supervise object boundary estimations.

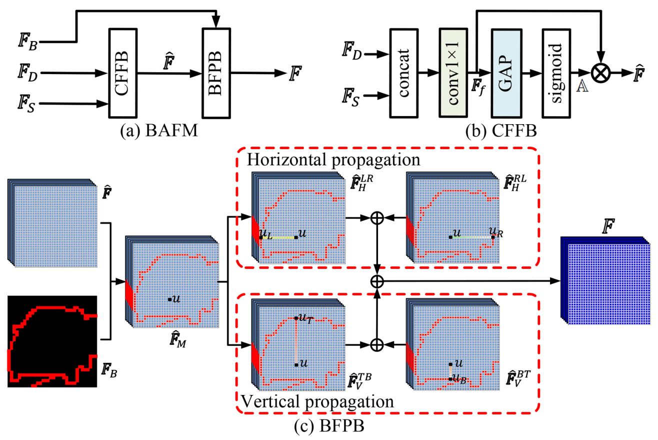


Fig. 5. Detailed architecture of BAFM is shown in (a), which consists of (b) convolution feature fusion block (CFFB) and (c) boundary-guided feature propagation block (BFPB). (Best viewed in color).

1) BAFM: The recent segmentation networks with boundary auxiliary can be roughly divided into two categories. One is employing boundary auxiliary head to guide internal features [17], [19], while the others introduce binary boundaries into segmentation networks [16], [21], [52]. Following second one, we utilize binary boundaries as high-level semantic guidance to enhance object discriminative capability by feature propagation, yet with lightweight design that is restricted by very few model parameters and computational complexity.

As can be seen in Fig. 5, our BAFM has three inputs: $\mathbb{F}_S$, $\mathbb{F}_D$, and $\mathbb{F}_B$. Specifically, low-level features $\mathbb{F}_S \in \mathbb{R}^{H \times W \times C}$ comes from the shallow layer of BSCNet that contains richer object shape details, where W, H, and C stand for feature width, height, and channel numbers, respectively. On the contrary, $\mathbb{F}_D \in \mathbb{R}^{H \times W \times 2C}$ denotes high-level features, produced by merging the outputs of ELPPM and HRB. Finally, $\mathbb{F}_B \in \mathbb{R}^{H \times W \times 1}$ represents a one-channel binary feature map, estimated from $\mathbb{F}_D$ via an auxiliary boundary supervision head. As shown in Fig. 5(a), BAFM composes of two components: convolution feature fusion block (CFFB) and boundary-guided feature propagation block (BFPB). Immediately below, we elaborate on the details of these two blocks.

CFFB. The detailed structure of CFFB is illustrated in Fig. 5(b). CFFB first fuses the deep layer features $\mathbb{F}_D$ and shallow layer features $\mathbb{F}_S$, and then reweighted by channel-wise attention [53]. More specifically, two inputs $\mathbb{F}_D$ and $\mathbb{F}_S$ are concatenated together, and then fed into an 1×1 convolution $\mathbf{f}(\cdot)$, producing an intermediate feature map $\mathbb{F}_f \in \mathbb{R}^{H \times W \times C}$:

$$\mathbb{F}_f = \mathbf{f}(\mathbb{F}_D \odot \mathbb{F}_S), \tag{4}$$

where $\odot$ indicates concatenation operation. Next, a squeeze attention [53] is employed to encode channel importance, and utilized to reweight intermediate feature $\mathbb{F}_f$. As illustrated in Fig. 5(b), $\mathbb{F}_f$ is directly fed into a global average pooling $\mathbf{P}_g(\cdot)$ to represent global entirety. Thereafter, a sigmoid function $\sigma(\cdot)$ is employed as usual to produce a channel-wise attention maps

$\mathbb{A} \in \mathbb{R}^{1\times 1\times C}$, used to reweight intermediate feature $\mathbb{F}_f$ to produce the output $\hat{\mathbb{F}} \in \mathbb{R}^{H\times W\times C}$ of CFFB:

$$\hat{\mathbb{F}} = \mathbb{F}_f \otimes \mathbb{A} = \mathbb{F}_f \otimes \sigma(\mathbf{P}_g(\mathbb{F}_f)), \tag{5}$$

where $\otimes$ stands for channel-wise multiplication.

BFPB. The goal of BFPB is to propagate information among feature maps. Undifferentiated propagation, however, would make the features assimilated, leading to the smooth representation that weakens the discrimination power. To classify features in different objects and stuff for lightweight semantic segmentation, it is beneficial to improve the feature similarity of the same object while keeping the feature discrimination of different objects. Therefore, we introduce the boundary information into feature propagation to control the information flow between different objects. As shown in Fig. 5(c), with the guide of estimated binary boundaries, the information is only allowed to pass through pixels belonging to the same segment, without interference from pixels of other objects. The detailed propagation process is presented below.

As illustrated in Fig. 5(c), given the output $\hat{\mathbb{F}}$ of CFFB and predicted binary boundaries $\mathbb{F}_B$, we first produce a masked boundary-aware features $\hat{\mathbb{F}}_M$, where the pixels with red color indicates estimated boundary pixels. As feature propagation has to ergodic all positions within one segment, one specific pixel will receive different information if we propagate features from different directions. To achieve high computational efficiency, the features of $\hat{\mathbb{F}}_M$ are propagated horizontally and vertically in a bi-directional manner, producing four propagated feature maps $\hat{\mathbb{F}}_H^{LR}$, $\hat{\mathbb{F}}_H^{RL}$, $\hat{\mathbb{F}}_V^{TB}$, and $\hat{\mathbb{F}}_V^{BT}$ that represent propagated directions from left to right, right to left, top to bottom, and bottom to top, respectively. Let $\mathbf{u}(x,y)$ be a pixel with horizontal and vertical coordinate (x,y), $\mathbf{u}_L(x_L,y)$ and $\mathbf{u}_R(x_R,y)$ be the most left and right pixels in the predicted boundary that share the same vertical coordinate of $\mathbf{u}(x,y)$, where $x_L < x < x_R$. Similarly, $\mathbf{v}_T(x,y_T)$ and $\mathbf{v}_B(x,y_B)$ are the most top and bottom pixels in the estimated boundary that share the same horizontal coordinate of $\mathbf{u}(x,y)$, where $y_T < y < y_B$. Then, to enhance feature similarity in the same segment, the propagation process for $\mathbf{u}(x,y)$ are defined as the average mean of feature responses from four directions:

$$\begin{aligned}
\hat{\mathbb{F}}_H^{LR}(x,y) &= \frac{\hat{\mathbb{F}}_M(x,y) + (x-x_L-1)\hat{\mathbb{F}}_H^{LR}(x-1,y)}{x-x_L}\\
\hat{\mathbb{F}}_H^{RL}(x,y) &= \frac{\hat{\mathbb{F}}_M(x,y) + (x_R-x-1)\hat{\mathbb{F}}_H^{RL}(x+1,y)}{x_R-x}\\
\hat{\mathbb{F}}_V^{TB}(x,y) &= \frac{\hat{\mathbb{F}}_M(x,y) + (y-y_T-1)\hat{\mathbb{F}}_V^{TB}(x,y-1)}{y-y_T}\\
\hat{\mathbb{F}}_V^{BT}(x,y) &= \frac{\hat{\mathbb{F}}_M(x,y) + (y_B-y-1)\hat{\mathbb{F}}_V^{BT}(x,y+1)}{y_B-y},
\end{aligned} \tag{6}$$

where numerator computes the accumulated sum, and denominator calculates how many pixels have been propagated in each direction. Computing average mean is also benefit for feature discrimination, as the consistent feature responses for different segments are more easily supervised by ground truth.

Although the propagation process works in a bi-directional manner, pixel $\mathbf{u}(x,y)$ actually receives all information from other pixels that have the same segment of $\mathbf{u}(x,y)$, as the average means of the propagated pixels have been computed and restored no matter which propagated directions come from. From (6), the step-by-step alternative accumulation will lead to different propagated information, whether left to right or right to left in horizon direction, nor up to bottom or bottom to up in vertical direction. As a result, four propagated features $\hat{\mathbb{F}}_H^{LR}$, $\hat{\mathbb{F}}_H^{RL}$, $\hat{\mathbb{F}}_V^{TB}$, and $\hat{\mathbb{F}}_V^{BT}$ have to be equally combined, producing outputs $\mathbb{F}$ of BFPB:

$$\mathbb{F} = \frac{1}{4}\left(\hat{\mathbb{F}}_H^{LR} + \hat{\mathbb{F}}_H^{RL} + \hat{\mathbb{F}}_V^{TB} + \hat{\mathbb{F}}_V^{BT}\right), \tag{7}$$

In (6), it can be seen that the information propagated to current pixel is only associated with its previous pixel, indicating that all pixels need to be ergodiced only once in each direction. As a result, the entire propagation process is computationally cheap, which is economic and desired in practice, especially for applications that require to compute high-resolution feature maps, such as self-driving. Moreover, traditional graph-based propagation methods [54] prone to the problem of propagation vanish, due to the pixel-by-pixel loop operation. However, our method relies on unidirectional computation, thus effectively alleviating this problem.

2) Boundary Loss: Considering the class imbalance problem between boundary and non-boundary pixels, the boundary loss $\mathcal{L}_b$ adopts widely-used cross-entropy $\mathcal{L}_{ce}$ [15] and dice loss $\mathcal{L}_{dice}$ [13] to jointly learn boundary features, as $\mathcal{L}_{dice}$ is not sensitive to the number of foreground/background pixels:

$$\mathcal{L}_b(\mathbb{P},\mathbb{B}) = \mathcal{L}_{ce}(\mathbb{P},\mathbb{B}) + \mathcal{L}_{dice}(\mathbb{P},\mathbb{B}), \tag{8}$$

where $\mathbb{P}$ and $\mathbb{B}$ stand for binary boundary predictions, and its corresponding ground truth, respectively.

IV. Experiments

To evaluate our method, we have conducted exhausted experiments on three semantic segmentation datasets: Cityscapes [23], CamVid [24], and KITTI [25]. We have compared with recent state-of-the-art lightweight segmentation networks in terms of segmentation accuracy and execution speed. In addition, a series of ablation studies have been carried on to reveal the potential impact of various components, and better understand the underlying behavior of BSCNet.

A. Datasets and Evaluation Metrics

1) Cityscapes: The Cityscapes dataset [23] contains 30 object categories, yet only 19 classes are used to evaluate semantic segmentation. It contains 5K pixel-wise well annotated images with 2048×1024 resolution, which are divided into 2,975/500/1,525 images for training, validation and testing, respectively. This dataset also includes nearly 20K coarse annotated training images. These images are not used to train our BSCNet, as the produced boundary ground truths are not accurate due to coarse annotations.

2) CamVid: Compared with Cityscapes [23], the CamVid dataset [24] is a smaller road scene dataset. It contains 11 object categories. All images are collected from the video sequences, producing 701 densely annotated frames that have resolution of 960×720. Following [8], [32], we split it into 367 for training, 101 for validation, and 233 for testing.

3) KITTI: The KITTI dataset [25] is another small self-driving dataset collected in the rural areas of a certain city in Germany. It only includes 200 annotated training images, and additional 200 images without annotation for testing. The semantic categories are compatible with Cityscapes [23].

4) Evaluation Metrics: For fair comparison with other state-of-the-art lightweight segmentation networks, we employ the standard evaluation metric mIoU score (mean intersection over union between segmentation estimations and ground truth maps) to measure segmentation accuracy. On the other hand, the commonly-used floating-point operations per second (FLOPs), model size (number of parameters) and frames per second (FPS) are used to measure implementation efficiency.

B. Implementation Details

1) Training Settings: BSCNet is implemented in the hardware server platform with a single RTX 2080Ti GPU card. The software code is based on the MMseg toolbox [55] that is an open-source repository for semantic segmentation. The widely-used stochastic gradient descent algorithm is employed to optimize BSCNet, where the momentum and weight decay are set to 9×10^{-1} and $5 \times e^{-4}$, respectively. For Cityscapes dataset, the cosine learning policy [56] is adopted with initial learning rate 10^{-1} and 16 images per batch for 120K iterations. Additionally, we first randomly crop out image patches with resolution of 1024×1024 from original images as additional data augmentation. The final segmentation performance is evaluated by BSCNet trained from the union of training and validation set. Following [57], the results are also evaluated using multi-scale inputs. To perform ablation studies, BSCNet is only trained from training set, and evaluated in validation set. For CamVid dataset, we set the initial learning rate to 10^{-3}, and train for 140K iterations with batch size 8. Additionally, BSCNet is directly trained using the union of training and validation set with the input resolution of 960×720. Unless special statement, the baseline results are directly borrowed from the corresponding publications.

2) Generation of Boundary Ground Truth: Following [32], the ground truth of object boundaries are produced by performing edge detection based on segmentation ground truth. Concretely, the filtered images that have pyramid resolutions are first produced using multi-scale Laplacian kernels. Thereafter, these filtered images are sequentially upsampled, thresholded, and weighted combined to obtain final binary ground truth.

3) Loss Settings: As shown in Fig. 1, there are three losses used to supervised the entire BSCNet. The first one $\mathcal{L}_f$ is after the final output of our system, while the second one $\mathcal{L}_{aux}$ is before the beginning of HRB, both of which employ cross-entropy loss for semantic segmentation. The last one $\mathcal{L}_b$, defined in (8), is used to supervise boundary predictions. Thus, the total loss $\mathcal{L}$ can be written as:

$$\mathcal{L} = \mathcal{L}_f + \alpha \times \mathcal{L}_{aux} + \beta \times \mathcal{L}_b, \tag{9}$$

where α and β are two non-negative parameters that leverage the trade-off between $\mathcal{L}_f$, $\mathcal{L}_{aux}$, and $\mathcal{L}_b$, respectively, setting as $\alpha = 0.5$ and $\beta = 1$, empirically.

TABLE II
COMPARISON WITH THE SELECTED STATE-OF-THE-ART APPROACHES ON CITYSCAPES DATASET

Method	Year	Resolution	Flo	Par	FPS	mIoU(%)	
						val	test
ESPNetV2 [11]	CVPR19	512 × 1024	5.9	1.3	220	66.4	66.2
ERFNet [6]	TITS18	512 × 1024	27	2.1	46	70.0	68.0
LEDNet [58]	ICIP19	512 × 1024	-	0.9	40	-	70.6
DABNet [7]	BMVC19	512 × 1024	10.5	0.8	28	-	70.1
SGCPNet [59]	TNNLS22	1024×2048	4.5	**0.6**	104	-	70.9
FasterSeg [60]	ICLR20	1024×2048	28	4.4	164	73.1	71.5
MSCFNet [61]	TITS22	512 × 1024	-	1.15	50	-	71.9
PCNet [30]	TITS22	1024×2048	11.8	1.6	72	-	72.7
LETNet [62]	TITS23	512 × 1024	13.6	1.0	150	-	72.8
SFNet [63]	ECCV20	1024×2048	-	9.1	121	-	74.5
LMFFNet [64]	TNNLS22	512 × 1024	16.7	1.4	119	74.9	75.1
STDCNet1 [32]	CVPR21	768 × 1536	-	-	127	74.5	75.3
SwiftNet [65]	CVPR19	1024×2048	52	12.0	40	75.4	75.5
HypSegM [33]	CVPR21	512 × 1024	7.5	10.1	37	76.2	75.8
AFPNet [66]	TSMC22	1024×2048	90	12.2	32	76.7	76.4
STDCNet2 [32]	CVPR21	768 × 1536	64	12.5	97	77.0	76.8
FBSNet [67]	TMM23	512 ×1024	9.5	**0.65**	90	**-**	70.9
DFANet [68]	CVPR19	1024×1024	3.4	7.8	100	-	71.3
MLFNet [41]	TIV23	512 × 1024	15.5	13	72	-	72.1
BiSeNetV1 [8]	ECCV18	640 × 360	2.9	5.8	67	74.8	74.7
LiHRNet [39]	CVPR21	512 × 1024	3.0	1.8	-	76.0	75.3
BiSeNetV2 [9]	IJCV21	1024×2048	21.2	5.8	47	75.8	75.3
DMANet [57]	TITS22	1024×2048	94	14.6	47	-	77.0
PIDNet [42]	CVPR23	1024×2048	46.3	7.6	101	78.3	78.2
Ours	-	512 × 1024	**2.6**	1.5	**380**	76.5	76.2
Ours	-	1024×2048	9.7	1.5	96	77.5	77.2
Ours†	-	1024×2048	9.7	1.5	96	**78.6**	**78.3**

The single-path (first part) and dual-path (Second part) methods are divided for better comparison. 'Flo' and 'Par' stand for FLOPs(G) and parameters(M), respectively. '–' Indicates that the results are not reported. '†' denotes multi-scale evaluation.
The bold values stand for best results.

C. Comparisons With State-of-the-Art Lightweight Networks

1) Experimental Results on Cityscapes: Table II shows the comparative results with some state-of-the-art lightweight networks on the validation and test set of Cityscapes dataset. Compared with the selected baselines, BSCNet achieves best trade-off in terms of segmentation accuracy and running efficiency. With only 9.7 GFLOPs and 1.5 M model size, BSCNet achieves 77.2% mIoU on test set and 96FPS running speed. When multi-scale evaluation is adopted [57], the segmentation performance is improved to 78.3% mIoU score. It is interesting that when the input resolution is reduced to 512×1024, our BSCNet delivers 2.1% mIoU drop, yet obtaining fastest running speed (380FPS), and smallest computing costs (2.6GFLOPs). Even so, our method still outperforms some lightweight networks that have higher input resolution, e.g., 1024×2048, due to the fact that the multi-scale semantic context are well captured by ELPPM and the object boundaries are well explored by BAFM. Specifically, compared with PIDNet [42] that achieves

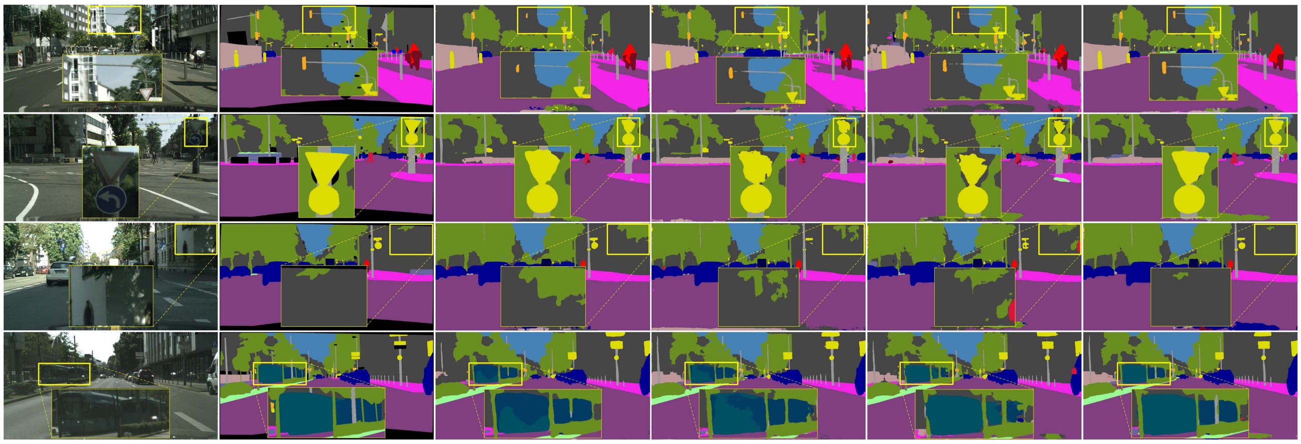


Fig. 6. Comparison of some visual examples of semantic segmentation on Cityscapes validation set. From left to right are input images, corresponding ground truth, segmentation outputs from STDCNet [32], ERFNet [6], BiSeNetV2 [9] and our BSCNet. For more clear comparison of each visual example, the area inside yellow windows are enlarged among all methods. (Best viewed in color).

second-ranked mIoU score, our BSCNet has nearly 5× less parameters and smaller GFLOPs, yet delivering 0.1% mIoU improvement. Among all baselines, although FBSNet [67] has the fewest number of parameters (0.6 M) that is only half size of BSCNet, our method surpasses it by large margins with remarkable 7.4% mIoU improvement, and achieves comparable running speed (96FPS vs 90FPS) and computing costs (9.7GFLOPs vs 9.5GFLOPs).

Fig. 6 shows qualitative results of some visual examples on Cityscapes validation dataset. Each example shows both the original image, corresponding ground truth and the color coded segmentation outputs. For better visual comparison, we also exhibit the segmentation results of three state-of-the-art lightweight models, including STDCNet [32], ERFNet [6], and BiSeNetV2 [9]. It is evident that, compared with state-of-the-art baselines, BSCNet produces smoother outputs and more accurate predictions with delineated object boundaries and shapes, such as the instances of "sign", "tree", and "train" in the second, third, and fourth example. Moreover, our method shows excellent capability to correctly segment tiny object instances, such as "pole" in first and fourth examples, probably due to the auxiliary of detective object boundaries. Finally, it also demonstrates that our approach is very effective for the occlusions between different objects. As shown in the fourth example where the "train" and "bus" are often misclassified by other methods, our BSCNet still obtains accurate segmentation outputs, although they are occluded by the instances of "tree" and "pole", and share very similar appearance.

2) Experimental Results on CamVid: In this section, we carry out experiments on the CamVid dataset to further evaluate the effectiveness of our method. Following [8], [30], we fix the full resolution of 960 × 720 to train BSCNet for fair comparison. To further improve performance, besides training on scratch, we also fine-tune BSCNet pre-trained on Cityscapes. All the results are reported in Table III. It is observed that, without fine-tuning, BSCNet obtains competitive results with respect to PIDNet [42] in terms of mIoU scores (78.4% vs 78.7%), yet BSCNet runs nearly 2× faster, and has only one fifth model size. When the pre-trained model is introduced, the segmentation accuracy improves to 79.8%, without significant grow of model size and GLOPs. In particular, BSCNet achieves smallest computing costs (3GFLOPs), and second-rank running speed (319FPS) among all selected baselines. Moreover, although our BSCNet is only trained from scratch, it still achieves better results (78.4% vs 68.7%) than BiSeNetV1 [8] pre-trained from ImageNet dataset [65]. Similar with the results evaluated on Cityscapes dataset, SGCPNet [59] has approximately half model size of BSCNet, yet delivering 10.8% mIoU score drop, and slightly slower inference speed (278FPS vs 319FPS). Fig. 7 also illustrates some segmentation outputs on CamVid test set. As

TABLE III
COMPARISON WITH THE SELECTED STATE-OF-THE-ART APPROACHES ON CAMVID TEST SET

Method	Year	Backbone	Flo	Par	FPS	mIoU(%)
DABNet [7]	BMVC19	No	13.8	0.8	74	66.4
LEDNet [58]	ICIP19	No	3.8	0.9	61	66.6
SGCPNet [59]	TNNLS22	MobileNet-V3	-	**0.6**	278	69.0
MSCFNet [61]	TITS22	No	-	1.15	38	69.3
LETNet [62]	TITS23	No	-	1.0	200	70.5
FasterSeg [60]	ICLR20	No	-	-	**398**	71.1
LMFFNet [64]	TNNLS22	No	22.0	1.4	116	72.0
PCNet [30]	TITS22	No	11.8	1.5	79	72.9
SFNet [63]	ECCV20	ResNet18	-	9.1	36	73.8
SwiftNet [65]	CVPR19	ResNet18	36	12.0	67	73.9
STDCNet [32]	CVPR21	STDC2	84	12.5	152	73.9
HypSegM [33]	CVPR21	EfficientNet-B1	-	9.9	38	78.4
DFANet [67]	CVPR19	XceptionA	-	7.8	120	64.7
BiSeNetV1 [8]	ECCV18	ResNet18	3.7	5.8	116	68.7
MLFNet [41]	TIV23	MobileNet-V2	19.8	13.0	68	71.8
DMANet [57]	TITS22	ResNet18	-	-	120	76.2
BiSeNetV2 [9]	IJCV21	No	-	5.8	33	78.5
PIDNet [42]	CVPR23	PIDNet-S	-	7.6	154	78.7
Ours	-	No	3.0	1.5	319	78.4
Ours†	-	No	**3.0**	1.5	319	**79.8**

The single-path (first part) and dual-path (second part) methods are divided for better comparison. 'Flo' and 'Par' stand for FLOPs(G) and parameters(M), respectively. '–' denotes that the results are not reported. '†' indicates pre-trained on cityscapes dataset. The bold values stand for best results.

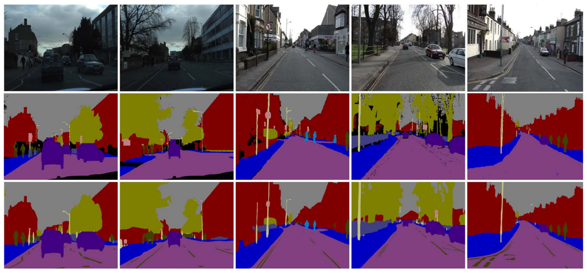


Fig. 7. Some visual examples of segmentation outputs on Camvid test dataset. From top to bottom are input images, corresponding ground truth, and segmentation results using our BSCNet. (Best viewed in color).

TABLE IV
Comparison With the Selected State-of-the-Art Approaches on KITTI Dataset

Method	Year	FLOPs (G)	Params (M)	FPS	mIoU(%)
CGNet [12]	TIP20	7.0	**0.5**	71	36.9
DABNet [7]	BMVC19	10.5	0.8	98	37.2
LEDNet [58]	ICIP19	11.5	1.0	61	40.3
ERFNet [6]	TITS18	26.9	2.1	105	42.9
LETNet [62]	TITS23	13.6	1.0	150	43.5
ESPNetV2 [11]	CVPR19	5.9	1.3	220	45.8
LMFFNet [64]	TNNLS22	16.7	1.4	119	49.3
DMANet [57]	TITS22	23.5	14.6	190	47.2
BiSeNetV2 [9]	IJCV21	5.3	5.8	188	50.7
Ours	-	**2.6**	1.5	**380**	**52.4**

The single-path (first part) and dual-path (second part) methods are divided for better comparison. For fair comparison, the input resolutions of all methods are fixed to 512 × 1024.
The bold values stand for best results.

TABLE V
The Effectiveness of Each Component on Cityscapes Val Set With the Input Resolution of 1024 × 2048

baseline	BFM$_1$	BFM$_2$	ELPPM	BAFM	Params (M)	Flops (G)	FPS	mIoU (%)
✓					1.25	8.826	130	73.74
✓	✓				1.31	8.839	126	74.38
✓		✓			1.31	8.832	127	73.96
✓	✓	✓			1.36	8.851	123	74.64
✓	✓	✓	✓		1.45	8.894	114	77.01
✓	✓	✓	✓	✓	1.49	9.712	96	78.63

can be seen, BSCNet still obtains visually-pleasing segmenting results, consistenting with the outputs as shown in Fig. 6.

3) Experimental Results on KITTI: Following [64], we also evaluate the generalization capability of BSCNet. Specifically, we directly predict the segmentation outputs of training images in KITTI using BSCNet trained from Cityscapes. The results are reported in Table IV. Among all selected state-of-the-art networks, BSCNet achieves the highest segmentation accuracy (52.4% mIoU score), together with fewest computing budgets (2.6GFLOPs) and fastest running speed (380FPS). In spit of only having one third model size of BSCNet, CGNet [12] requires more than 2.8× computation amount, approximate one fifth running speed, and 15.5% accuracy loss.

D. Ablation Studies

In order to understand the underlying behavior of BSCNet, this section reports the results of a series of ablation studies.

1) Ablation Studies of Different Components: Table V reports some ablation studies on Cityscapes validation set, which quantify the individual contribution of three main components: BFM, ELPPM, and BAFM, respectively. We first construct baseline only using LRB and HRB, and then three components are sequentially added. The results show that each component continuously improves the performance, yet with small amount of increase in terms of GFLOPs and model size. Among all components, BFM obtains smallest mIoU gain (0.9%). However, BAFM improves 1.62% mIoU scores, with only increase of 0.04 M model size and 0.818GFLOPs, indicating that the learned object boundaries are indeed helpful for semantic segmentation. Particularly, ELPPM achieves remarkable mIoU improvement (2.37%) with slightly increase of model parameters and computing overheads (0.09 M and 0.043GFLOPs), benefiting from the powerful capability of learning multi-scale context. We further evaluate the importance of two BFMs by sequentially plugging them into our backbone. From Table V, exchanging information with high-resolution features obtains better results, probably because high-resolution features keep more image details for interaction. Moreover, when two BFMs are both employed, our method achieves the best results, indicating that using feature interaction as many as possible is beneficial for improving performance. Fig. 8 illustrates the segmentation outputs that evaluate the contributions of different components. Consistenting with Table V, the visual results are gradually closed to ground truth when individual component is introduced step-by-step, whether correctly identifying different object instance (e.g., "terrain" in the final example) or accurately delineating object boundaries (e.g, "sign" and "person" in the first two examples).

To further demonstrate BAFM, Fig. 9 exhibits some estimated binary boundaries, and its influence to final segmentation outputs. It is observed that BAFM is very effective for clutter area and tiny objects. For instance, the misclassified "fence" are rectified to "tree" and "building" in first example, and the incompletely slender "pole" and discontinuous "sidewalk" broken by "pole" and "road" are correctly identified in last two examples. On the other hand, feature propagation in BAFM heavily depends on the quality of estimated binary boundaries. In the first example, some instance of "pole" are incorrectly classified to "building" or "grass", mainly due to the fact that their boundaries are not well predicted. The edge detectors may consider that these poles are texture part of "building" or "grass".

Fig. 10 also shows the validity of feature propagation in BAFM. Before the features are propagated, the high feature response always scattered over different parts of an object area (e.g., windows of "car" in first example, and heads of "person" in second one), which is unfavourable for classification. After feature propagation is conducted, due to the uniform assign of feature responses, pixel features are consistently enhanced within the same segment (e.g., "car" in the first example, "person" in the second example, and "tree" in two examples), resulting in the improvement of discriminative power that is beneficial for correctly recognizing the associated object area.

2) Ablation Studies for ELPPM and BAFM: As plug-and-play modules, ELPPM and BAFM play an essential role for capturing multi-scale context and utilizing boundary guidance.

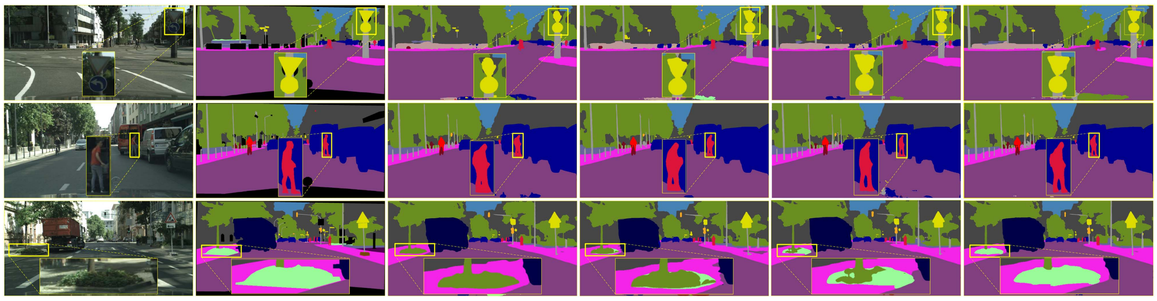


Fig. 8. Some visual examples for qualitative evaluation of different components. From left to right are input images, corresponding ground truth, segmentation results from baseline, baseline + BFM, baseline + BFM + ELPPM, and baseline + BFM + ELPPM + BAFM. For more clear comparison of each visual example, the area inside yellow windows are enlarged among all methods. (Best viewed in color).

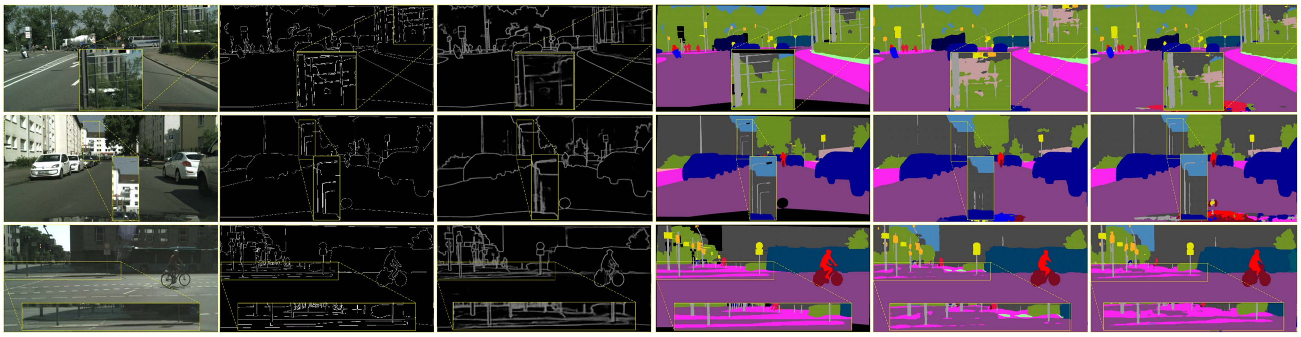


Fig. 9. Some visual examples using BAFM on Cityscapes validation dataset. From left to right are input images, boundary ground truth, estimated boundaries, segmentation ground truth, segmentation results without BAFM, and with BAFM. For more clear comparison, the area inside yellow windows are enlarged for all visual examples. (Best viewed in color).

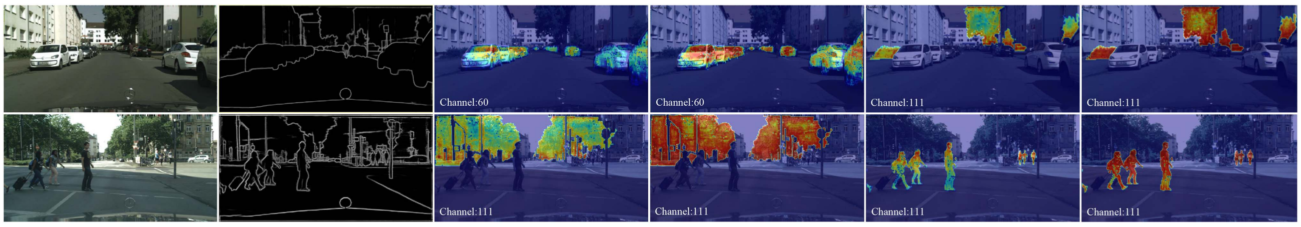


Fig. 10. Feature propagation in BAFM. From left to right are input images, estimated boundaries, and feature heatmaps before and after propagation. Red color indicates high responses, while blue color denotes low ones. For clarity, the channel numbers are superimposed on heatmaps. (Best viewed in color).

As a result, this section evaluates two modules by alternatively plugging them into different recent state-of-the-art lightweight backbones. In order to increase diversity of backbones, single-path [6], [51], [69], [70] and dual-path [9], [32] architectures are both adopted. The results are reported in Table VI. Regardless of which lightweight backbone is utilized, ELPPM and BAFM are able to consistently boost segmentation performance, yet with slight increase of model size and GFLOPs. Particularly, there are barely any changes of model parameters when BAFM is adopted. It is also intriguing that the gains brought by two modules gradually decrease, as more and more powerful lightweight backbones are employed. Consistenting with Table V, ELPPM always brings more improvement than BAFM.

3) Ablation Studies for Capturing Multi-Scale Semantic Context: To evaluate the ability for capturing multi-scale semantic context, this section compares ELPPM with Res2Net [20] and PSPNet [2]. The results are reported in Table VII. It can be seem that there is a slightly performance improvement (0.33% mIoU score), yet with significant increase of model size (0.72 M), probably due to the standard convolutions used in Res2Net, instead of depth-wise convolution employed in ELPPM. When ELPPM is replaced by PSPNet [2], it requires less model parameters (1.41 M) and fewer computing overheads (9.677GFLOPs) due to its pooling operations to investigate multi-scale global context. However, PSPNet is too weak to extract semantic context, leading to the poor segmentation mIoU scores (76.58% vs 78.63%).

4) Ablation Studies for the Location of BFM: This section evaluates the influce by changing the positions of BFM in our backbone. However, it is impractical to enumerate all possible positions. We thus fix one BFM, and change the location of another. The results are reported in Table VIII. It is observed that moving BFMs behind the stride version of ECCM often delivers to poor performance, probably because image details have been

TABLE VI
ABLATION STUDIES FOR ELPPM AND BAFM AS PLUG-AND-PLAY MODULES ON CITYSCAPES VAL SET

Method	Backbone	ELPPM	BAFM	Par	Flo	FPS	mIoU (%)
MobileNet [68]	MobileNet-V3			0.43	3.17	69	58.9
		✓		0.52	3.21	68	66.4 (↑ 7.5)
			✓	0.47	3.99	65	65.2 (↑ 6.3)
ShuffleNet [51]	ShuffleNet-V2			6.04	4.73	57	71.3
		✓		6.13	4.77	54	75.8 (↑ 4.5)
			✓	6.08	5.54	53	73.5 (↑ 2.2)
ERFNet [6]	ERFNet			2.17	27.00	46	70.0
		✓		2.26	27.04	44	72.7 (↑ 2.7)
			✓	2.21	27.82	41	71.6 (↑ 1.6)
EfficientNet [69]	EfficientNet-B1			7.81	5.79	29	74.1
		✓		7.90	5.83	26	76.1 (↑ 2.0)
			✓	7.85	6.61	22	75.9 (↑ 1.8)
BiseNetV2 [9]	XCeption			5.85	21.21	47	75.8
		✓		5.94	21.25	45	77.3 (↑ 1.5)
			✓	5.89	22.03	42	76.5 (↑ 0.7)
STDCNet [32]	STDC2			12.57	64.08	97	77.0
		✓		12.66	64.12	94	78.1 (↑ 1.1)
			✓	12.61	64.90	90	77.3 (↑ 0.3)

Par' and 'Flo' stand for parameters(M) and FLOPs(G), respectively.

TABLE VII
PERFORMANCE COMPARISON BETWEEN ELPPM, RES2NET, AND PSPNET IN TERMS OF EFFICIENCY AND ACCURACY

ELPPM	Res2Net	PSPNet	Params (M)	Flops (G)	FPS	mIoU (%)
✓			1.49	9.712	96	78.63
	✓		2.21	10.044	81	78.96
		✓	1.41	9.677	105	76.58

TABLE VIII
ABLATION STUDIES BY CHANGING THE POSITIONS OF BFMS

Layer Number	6	9	10	12
BFM_1	75.7%	76.3%	**78.6%**	77.1%
Layer Number	6	7	9	12
BFM_2	77.7%	**78.6%**	75.%3	74.6%

"Layer number" means moving BFM after that layer.
The bold values stand for best results.

discarded in low-resolution features, which are unsuitable for directly interacting with high-resolution features.

E. Analysis of Parameter Settings

1) Effect of Pooling Path Number in ELPPM: The number of pooling paths determines how many scales of context cues are explored, significantly influencing the trade-off between the capability of context representation and computational efficiency in ELPPM. We thus evaluate the performance variance along with the changes of path number i, ranged from 2 to 8 with updated step 2. The results are reported in Table IX. Note we employ the mIoU improvement brought by ELPPM only as baseline. From Table IX, the mIoU score boosts along with the increase of pooling path, yet requiring more and more model size and computing costs. The best trade-off is achieved when $i = 4$, thus chosen as default setting in ELPPM.

2) Effect of Different Weight Combinations for Loss Function: This section evaluates the effect of auxiliary loss $\mathcal{L}_{aux}$ and $\mathcal{L}_b$, which is helpful to optimize the whole training process, and has no interference with learning the master branch loss $\mathcal{L}_f$. By adjusting and combining hyper-parameters α and β in range [0, 1] in steps 0.5, we carry on a set of experiments for optimization. Note $\mathcal{L}_b = 0$ indicates the entire training process is dominant by segmentation loss, without any guidance from object boundaries. The results are reported in Table X. It is observed that the mIoU scores peak at $\alpha = 0.5$ and $\beta = 1$, which are also opt to default settings in BSCNet.

TABLE IX
PARAMETER ANALYSIS FOR POOLING PATH NUMBER i IN ELPPM ON CITYSCAPES VAL SET

Pooling path number i	Params (M)	Flops (G)	mIoU (%)
2	0.05	0.02	1.98
4	0.09	0.04	2.37
6	0.12	0.07	2.41
8	0.17	0.09	2.43

TABLE X
PARAMETER ANALYSIS FOR DIFFERENT WEIGHT COMBINATIONS OF LOSS FUNCTION ON CITYSCAPES VAL SET

mIoU (%)	73.7	74.1	75.8	75.5	76.9	**78.6**	77.0	77.4	77.9
α	0	0	0	0.5	0.5	0.5	1	1	1
β	0	0.5	1	0	0.5	1	0	0.5	1

The bold values stand for best results.

V. CONCLUDING REMARKS AND FUTURE WORK

This paper has presented a BSCNet for lightweight semantic segmentation that leverages boundary auxiliary and multi-scale semantic context. From resolution perspective, ELPPM adopts pyramid pooling structure to achieve powerful representation for capturing multi-scale context clues. Additionally, the proposed BAFM employs estimated object boundaries as high-level guidance to propagate convolution features, improving discrimination power for identifying each pixel. Both ELPPM and BAFM are computationally efficient as they require very few model size and computational costs. Finally, we introduce the detailed architecture of entire BSCNet. The experimental results show that BSCNet achieves state-of-the-art trade-off in terms of segmentation accuracy and implementing efficiency on three datasets: Cityscapes, Camvid, and KITTI. Moreover, the ablation studies demonstrate that ELPPM and BAFM are very simple and effective plug-and-play modules by varying recent state-of-the-art lightweight backbone networks.

In spite to achieving impressive performance for lightweight semantic segmentation, in the future, we believe that BSCNet can be easily transferred to other real-time multimedia tasks, such as pose estimation [71] and character recognition [72].

REFERENCES

[1] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2015, pp. 3431–3440.
[2] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, "Pyramid scene parsing network," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 2881–2890.

[3] M. Lingfeng, X. Hongtao, C. Liu, and Y. Zhang, "Learning cross-channel representations for semantic segmentation," *IEEE Trans. Multimedia*, vol. 25, pp. 2774–2787, 2023.

[4] C. Yin, J. Tang, T. Yuan, Z. Xu, and Y. Wang, "Bridging the gap between semantic segmentation and instance segmentation," *IEEE Trans. Multimedia*, vol. 24, pp. 4183–4196, 2022.

[5] Z. Kaipeng and S. Yoichi, "Semantic image segmentation by dynamic discriminative prototypes," *IEEE Trans. Multimedia*, vol. 26, pp. 737–749, 2024.

[6] E. Romera, J. M. Alvarez, L. M. Bergasa, and R. Arroyo, "ERFNet: Efficient residual factorized convnet for real-time semantic segmentation," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 1, pp. 263–272, Jan. 2018.

[7] G. Li and J. Kim, "DABNet: Depth-wise asymmetric bottleneck for real-time semantic segmentation," in *Proc. Brit. Mach. Vis. Conf.*, 2019, pp. 1–12.

[8] C. Yu et al., "BiSeNet: Bilateral segmentation network for real-time semantic segmentation," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 325–341.

[9] C. Yu et al., "BiSeNet V2: Bilateral network with guided aggregation for real-time semantic segmentation," *Int. J. Comput. Vis.*, vol. 129, no. 11, pp. 3051–3068, 2021.

[10] S. N. Xie, R. Girshick, P. Dollar, Z. Tu, and K. He, "Aggregated residual transformations for deep neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 5987–5995.

[11] S. Mehta, M. Rastegari, L. Shapiro, and H. Hajishirzi, "Espnetv2: A light-weight, power efficient, and general purpose convolutional neural network," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 9190–9200.

[12] 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, 2020.

[13] H. Zhao, X. Qi, X. Shen, J. Shi, and J. Jia, "ICNet for real-time semantic segmentation on high-resolution images," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 405–420.

[14] X. Wang, R. Girshick, A. Gupta, and K. He, "Non-local neural networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 7794–7803.

[15] J. Fu et al., "Dual attention network for scene segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 3146–3154.

[16] T. Takikawa, D. Acuna, V. Jampani, and S. Fidler, "Gated-SCNN: Gated shape CNNs for semantic segmentation," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2019, pp. 5229–5238.

[17] M. Zhen et al., "Joint semantic segmentation and boundary detection using iterative pyramid contexts," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 13666–13675.

[18] H. J. Lee, J. U. Kim, S. Lee, H. G. Kim, and Y. M. Ro, "Structure boundary preserving segmentation for medical image with ambiguous boundary," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 4817–4826.

[19] X. Yin, D. Min, Y. Huo, and S.-E. Yoon, "Contour-aware equipotential earning for semantic segmentation," *IEEE Trans. Multimedia*, vol. 25, pp. 6146–6156, 2023.

[20] S.-H. Gao et al., "Res2net: A new multi-scale backbone architecture," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 43, no. 2, pp. 652–662, Feb. 2021.

[21] Q. Huang et al., "Object boundary guided semantic segmentation," in *Proc. Asian Conf. Comput. Vis.*, 2016, pp. 197–212.

[22] W. Shen, B. Wang, Y. Jiang, Y. Wang, and A. Yuille, "Multi-stage multi-recursive-input fully convolutional networks for neuronal boundary detection," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2017, pp. 2391–2400.

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

[24] G. Brostow, J. Shotton, J. Fauqueur, and R. Cipolla, "Segmentation and recognition using structure from motion point clouds," in *Proc. Eur. Conf. Comput. Vis.*, 2008, pp. 44–57.

[25] 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.

[26] J. Wu, C. Leng, Y. Wang, Q. Hu, and J. Cheng, "Quantized convolutional neural networks for mobile devices," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 4820–4828.

[27] S. Han, H. Mao, and W. J. Dally, "Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding," in *Proc. Int. Conf. Learn. Representation*, 2016, pp. 1–14.

[28] A. Shumin, L. Qingmin, L. Zongqing, and J. Xue, "Efficient semantic segmentation via self-attention and self-distillation," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 9, pp. 15256–15266, Sep. 2022.

[29] 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.

[30] L. Qingxuan, S. Xin, C. Changrui, D. Junyu, and H. Zhou, "Parallel complement network for real-time semantic segmentation of road scenes," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 5, pp. 4432–4444, May 2022.

[31] S. Mehta, M. Rastegari, A. Caspi, L. Shapiro, and H. Hajishirzi, "ESPNet: Efficient spatial pyramid of dilated convolutions for semantic segmentation," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 552–568.

[32] M. Fan et al., "Rethinking BiSeNet for real-time semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 9716–9725.

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

[34] W. Yuhuan, L. Yun, Z. Xin, and C. Mingming, "P2T: Pyramid pooling transformer for scene understanding," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 45, no. 11, pp. 12760–12771, Nov. 2023.

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

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

[37] W. Haiping et al., "CvT: Introducing convolutions to vision transformers," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2021, pp. 22–31.

[38] X. Guoan et al., "Lightweight real-time semantic segmentation network with efficient transformer and CNN," *IEEE Trans. Intell. Transp. Syst.*, vol. 24, no. 12, pp. 15897–15906, Dec. 2023.

[39] C. Yu et al., "Lite-HRNet: A lightweight high-resolution network," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 10435–10445.

[40] J. Wang et al., "Deep high-resolution representation learning for visual recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 43, no. 10, pp. 3349–3364, Oct. 2021.

[41] J. Fan et al., "MLFNet: Multi-level fusion network for real-time semantic segmentation of autonomous driving," *IEEE Trans. Intell. Veh.*, vol. 8, no. 1, pp. 756–767, Jan. 2023.

[42] X. Jiacong, X. Zixiang, and S. P. Bhattacharyya, "PIDNet: A real-time semantic segmentation network inspired by PID controllers," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2023, pp. 19529–19539.

[43] M. Yang, K. Yu, C. Zhang, Z. Li, and K. Yang, "DenseASPP for semantic segmentation in street scenes," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 3684–3692.

[44] X. Ding et al., "Scaling up your kernels to 31x31: Revisiting large kernel design in CNNs," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 11963–11975.

[45] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "DeepLab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 4, pp. 834–848, Apr. 2018.

[46] Z. Zhu, M. Xu, S. Bai, T. Huang, and X. Bai, "Asymmetric non-local neural networks for semantic segmentation," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2019, pp. 593–602.

[47] S. N. Xie and Z. W. Tu, "Holistically-nested edge detection," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2015, pp. 1395–1403.

[48] R. X. Deng and S. J. Liu, "Deep structural contour detection," in *Proc. ACM Int. Conf. Multimedia*, 2020, pp. 304–312.

[49] W. J. Xuan, S. L. Huang, J. H. Liu, and B. Du, "FCL-Net: Towards accurate edge detection via fine-scale corrective learning," *Neural Netw.*, vol. 145, pp. 248–259, 2022.

[50] W. J. Xuan et al., "PNT-Edge: Towards robust edge detection with noisy labels by learning pixel-level noise transitions," in *Proc. ACM Int. Conf. Multimedia*, 2023, pp. 1924–1932.

[51] 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.*, 2018, pp. 116–131.

[52] H. Ding, X. Jiang, A. Q. Liu, N. M. Thalmann, and G. Wang, "Boundary-aware feature propagation for scene segmentation," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2019, pp. 6818–6828.

[53] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 7132–7141.

[54] S. Bing, Z. Zhen, W. Bing, and G. Wang, "Scene segmentation with dag-recurrent neural networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 6, pp. 1480–1493, Jun. 2018.

[55] M. Contributors, "MMSegmentation: Openmmlab semantic segmentation toolbox and benchmark," 2020. [Online]. Available: https://github.com/open-mmlab/mmsegmentation

[56] I. Loshchilov and F. Hutter, "SGDR: Stochastic gradient descent with warm restarts," in *Proc. Int. Conf. Learn. Representation*, 2017, pp. 1–14.

[57] W. Xi et al., "Deep multi-branch aggregation network for real-time semantic segmentation in street scenes," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 10, pp. 17224–17240, Oct. 2022.

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

[59] S. Hao et al., "Real-time semantic segmentation via spatial-detail guided context propagation," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, Mar.8, 2022, doi: 10.1109/TNNLS.2022.3154443.

[60] W. Chen et al., "FasterSeg: Searching for faster real-time semantic segmentation," in *Proc. Int. Conf. Learn. Representation*, 2020, pp. 1–14.

[61] G. Guangwei et al., "MSCFNet: A lightweight network with multi-scale context fusion for real-time semantic segmentation," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 12, pp. 25489–25499, Dec. 2022.

[62] X. Guoan et al., "Lightweight real-time semantic segmentation network with efficient transformer and CNN," *IEEE Trans. Intell. Transp. Syst.*, vol. 24, no. 12, pp. 15897–15906, Dec. 2023.

[63] X. Li et al., "Semantic flow for fast and accurate scene parsing," in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 775–793.

[64] M. Shi et al., "LMFFNet: A well-balanced lightweight network for fast and accurate semantic segmentation," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 34, no. 6, pp. 3205–3219, Jun. 2023.

[65] 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 Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 12607–12616.

[66] H. Junhyuk, S. Hongje, K. Sangki, and K. Euntai, "Adjacent feature propagation network (AFPNet) for real-time semantic segmentation," *IEEE Trans. Syst., Man, Cybern.: Syst.*, vol. 52, no. 9, pp. 5877–5888, Sep. 2022.

[67] G. Gao et al., "FBSNet: A fast bilateral symmetrical network for real-time semantic segmentation," *IEEE Trans. Multimedia*, vol. 25, pp. 3273–3283, 2023.

[68] H. Li, P. Xiong, H. Fan, and J. Sun, "DFANet: Deep feature aggregation for real-time semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 9522–9531.

[69] H. Andrew et al., "Searching for mobilenetv3," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2019, pp. 1314–1324.

[70] T. Mingxing and V. L. Quoc, "Efficientnet: Rethinking model scaling for convolutional neural networks," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 6105–6114.

[71] L. Huan et al., "Fast human pose estimation in compressed videos," *IEEE Trans. Multimedia*, vol. 25, pp. 1390–1400, 2023.

[72] W. Xiangping et al., "LCSegNet: An efficient semantic segmentation network for large-scale complex Chinese character recognition," *IEEE Trans. Multimedia*, vol. 23, pp. 3427–3440, 2021.

Quan Zhou (Senior Member, IEEE) received the B.S. degree in electronics and information engineering from the China University of Geosciences, Wuhan, China, in 2002, and the M.S and Ph.D. degrees in electronics and information engineering from the Huazhong University of Science and Technology, Wuhan, in 2006 and 2013, respectively. From 2019 to 2020, he was a Visiting Scholar with Temple University, Philadelphia, PA, USA. He is currently a Full Professor with the National Engineering Research Center of Communication and Network Technology, Nanjing University of Posts and Telecommunications, Nanjing, China. He has authored or coauthored more than 70 academic articles, including IEEE TRANSACTIONS ON IMAGE PROCESSING, IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS, IEEE TRANSACTIONS ON MEDICAL IMAGING, IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS, and *Pattern Recognition*. His research interests include deep learning, pattern recognition, and computer vision. He was the Area Chairs of IEEE ICME2019 and PRCV2022, and TPC members of ISAIR and WCSP. He was a leading Guest Editor of IEEE TRANSACTIONS ON MULTIMEDIA, IEEE TRANSACTIONS ON FUZZY SYSTEMS, *Pattern Recognition*, *Computers and Electrical Engineering*, and *Multimedia Tools and Applications*, and reviewer for more than 70 SCI journals. He is Senior Member of the International Association of Pattern Recognition (IAPR).

Linjie Wang received the B.S. degree in communication engineering from the North China Institute of Science and Technology, Hebei, China, in 2019, and the M.S. degree in telecommunications engineering from the Nanjing University of Posts and Telecommunications, Nanjing, China, in 2023. She is currently a Software Engineer with New H3C Technologies Company Ltd., Hangzhou, China. Her research interests include semantic segmentation, object detection, and image understanding.

Guangwei Gao (Senior Member, IEEE) received the Ph.D. degree in pattern recognition and intelligence systems from the Nanjing University of Science and Technology, Nanjing, China, in 2014. He was a Visiting Student with the Department of Computing, Hong Kong Polytechnic University, Hong Kong, in 2011 and 2013, respectively. From 2019 to 2021, he was a Project Researcher with the National Institute of Informatics, Japan. He is currently an Associate Professor with the Nanjing University of Posts and Telecommunications. He has authored or coauthored more than 70 scientific papers in IEEE TRANSACTIONS ON IMAGE PROCESSING/TCSVT/TITS/TMM/TIFS, *ACM Transactions on Internet Technology*/*ACM Transactions on Multimedia Computing, Communications, and Applications*, *PR*, AAAI, and IJCAI. His research interests include pattern recognition and computer vision. Personal website: *https://guangweigao.github.io*.

Bin Kang (Member, IEEE) received the Ph.D. degree in electrical engineering from the Nanjing University of Posts and Telecommunications, Nanjing, China, in 2016. He is an Associate Professor with the Nanjing University of Posts and Telecommunications. His research interests include the area of computer vision and pattern recognition.

Weihua Ou received the Ph.D. degree in information and communication engineering from the Huazhong University of Science and Technology, Wuhan, China. He is currently a Full Professor with the School of Big data and Computer Science, Guizhou Normal University, Guiyang, China. His research results have published more 80 paper at prominent journals and conferences, such as IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS, IEEE TRANSACTIONS ON MULTIMEDIA, IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY, *PR*, ICPR, and ICME. His research interests include graph learning, LLM, and multimodal machine learning. His publications have been cited in Google Scholar more than 1800 times, his H-Index is 23.

Huimin Lu (Senior Member, IEEE) received the Ph.D. degree in electrical engineering from the Kyushu Institute of Technology, Kitakyushu, Japan, in 2014. From 2013 to 2016, he was a JSPS Research Fellow with the Kyushu Institute of Technology. From 2016 to 2024, he was an Associate Professor with the Kyushu Institute of Technology and an Excellent Young Researcher of Ministry of Education, Culture, Sports, Science and Technology. He is currently a Professor with Southeast University, Nanjing, China. His research interests include artificial intelligence, computer vision, and robotics. He is the Editor-in-Chief of *Computers & Electrical Engineering* and *Cognitive Robotics Journal*, and the Editor of *Wireless Networks*, *Applied Soft Computing*, IEEE TRANSACTIONS ON ARTIFICIAL INTELLIGENCE, *IEICE Transactions on Information and Systems*, and *Pattern Recognition*. He is the Guest Editor for many journals, such as IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY, IEEE TRANSACTIONS ON NETWORK SCIENCE AND ENGINEERING, *ACM Transactions on Internet Technology*, IEEE/CAA JOURNAL OF AUTOMATICA SINICA, IEEE INTERNET OF THINGS JOURNAL, and IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS. He is the Chair of IEEE Computer Society Technical Committee on Big Data.