Noname manuscript No.
(will be inserted by the editor)

BiSeNet V2: Bilateral Network with Guided Aggregation for Real-time Semantic Segmentation

Changqian Yu[1,2] · **Changxin Gao**[1*] · **Jingbo Wang**[3] · **Gang Yu**[4] · **Chunhua Shen**[2] · **Nong Sang**[1]

Received: date / Accepted: date

Abstract The low-level details and high-level semantics are both essential to the semantic segmentation task. However, to speed up the model inference, current approaches almost always sacrifice the low-level details, which leads to a considerable accuracy decrease. We propose to treat these spatial details and categorical semantics separately to achieve high accuracy and high efficiency for real-time semantic segmentation. To this end, we propose an efficient and effective architecture with a good trade-off between speed and accuracy, termed Bilateral Segmentation Network (BiSeNet V2). This architecture involves: (i) a Detail Branch, with wide channels and shallow layers to capture low-level details and generate high-resolution feature representation; (ii) a Semantic Branch, with narrow channels and deep layers to obtain high-level semantic context. The Semantic Branch is lightweight due to reducing the channel capacity and a fast-downsampling strategy. Furthermore, we design a Guided Aggregation Layer to enhance mutual connections and fuse both types of feature representation. Besides, a booster training strategy is designed to improve the segmentation performance without any extra inference cost. Extensive quantitative and qualitative evaluations demonstrate that the proposed architecture performs favourably against a few *state-of-the-art* real-time semantic segmentation approaches. Specifically, for a 2,048×1,024 input, we achieve 72.6% Mean IoU on the Cityscapes test set with a speed of 156 FPS on one NVIDIA GeForce GTX 1080 Ti card, which is significantly faster than existing methods, yet we achieve better segmentation accuracy. Code and trained models will be made publicly available.

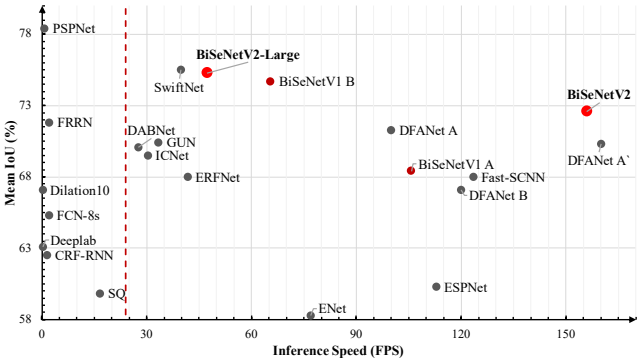


Fig. 1 Speed-accuracy trade-off comparison on the Cityscapes *test* set. Red dots indicate our methods, while grey dots means other methods. The red line represents the real-time speed.

Keywords Semantic Segmentation · Real-time Processing · Deep Learning

1 Introduction

Semantic segmentation is the task of assigning semantic labels to each pixel. It is a fundamental problem in computer vision with extensive applications, including scene understanding (Zhou et al., 2019), autonomous driving (Cordts et al., 2016; Geiger et al., 2012), human-

Changqian Yu, Changxin Gao, Nong Sang
E-mail: {changqian_yu, cgao, nsang}@hust.edu.cn

Chunhua Shen
E-mail: chunhua.shen@adelaide.edu.au

[1]National Key Laboratory of Science and Technology on Multi-spectral Information Processing, School of Artificial Intelligence and Automation, Huazhong University of Science and Technology, Wuhan, China
[2]The University of Adelaide, Australia
[3]The Chinese University of Hong Kong
[4]Tencent
*Corresponding author

machine interaction and video surveillance, just to name a few. In recent years, with the advance of convolutional neural network (Krizhevsky et al., 2012), a series of semantic segmentation methods (Zhao et al., 2017; Chen et al., 2017; Yu et al., 2018b; Chen et al., 2018; Zhang et al., 2018a) based on fully convolutional network (FCN) (Long et al., 2015) have constantly advanced the state-of-the-art performance.

The high accuracy of these methods depends on their backbone networks. There are two main architectures as the backbone networks: (i) *Dilation Backbone*, removing the downsampling operations and upsampling the corresponding filter kernels to maintain high-resolution feature representation (Chen et al., 2015, 2018; Zhao et al., 2017, 2018b; Fu et al., 2019; Yu et al., 2020), as shown in Figure 2 (a). (ii) *Encoder-Decoder Backbone*, with top-down and skip connections to recover the high-resolution feature representation in the decoder part (Lin et al., 2017; Peng et al., 2017; Yu et al., 2018b), as illustrated in Figure 2 (b). However, both architectures are designed for general semantic segmentation tasks with less care about the inference speed and computational cost. In the dilation backbone, the dilation convolution is time-consuming and removing down-sampling operation brings heavy computation complexity and memory footprint. Numerous connections in the encoder-decoder architecture are less friendly to the memory access cost (Ma et al., 2018). However, the real-time semantic segmentation applications demand for an efficient inference speed.

Facing this demand, based on both backbone networks, existing methods (Badrinarayanan et al., 2017; Paszke et al., 2016; Zhao et al., 2018a; Romera et al., 2018; Mazzini, 2018) mainly employ two appraches to accelerate the model: (i) *Input Restricting*. Smaller input resolution results in less computation cost with the same network architecture. To achieve real-time inference speed, many algorithms (Zhao et al., 2018a; Romera et al., 2018; Mazzini, 2018; Romera et al., 2018) attempt to restrict the input size to reduce the whole computation complexity; (ii) *Channel Pruning*. It is a straight-forward acceleration method, especially pruning channels in early stages to boost inference speed (Badrinarayanan et al., 2017; Paszke et al., 2016; Chollet, 2017). Although both manners can improve the inference speed to some extent, they sacrifice the low-level details and spatial capacity leading to a dramatic accuracy decrease. Therefore, to achieve high efficiency and high accuracy simultaneously, it is challenging and of great importance to exploit a specific architecture for the real-time semantic segmentation task.

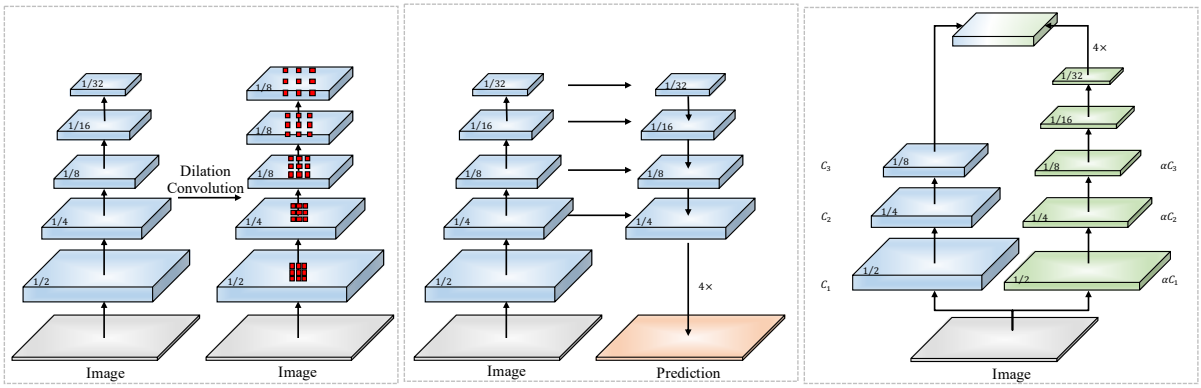
We observe that both of the low-level details and high-level semantics are crucial to the semantic segmentation task. In the general semantic segmentation task, the deep and wide networks encode both information simultaneously. However, in the real-time semantic segmentation task, we can treat spatial details and categorical semantics separately to achieve the trade-off between the accuracy and inference speed.

To this end, we propose a two-pathway architecture, termed **Bi**lateral **Se**gmentation **Net**work (BiSeNet V2), for real-time semantic segmentation. One pathway is designed to capture the spatial details with wide channels and shallow layers, called *Detail Branch*. In contrast, the other pathway is introduced to extract the categorical semantics with narrow channels and deep layers, called *Semantic Branch*. The Semantic Branch simply requires a large receptive field to capture semantic context, while the detail information can be supplied by the Detail Branch. Therefore, the Semantic Branch can be made very lightweight with fewer channels and a fast-downsampling strategy. Both types of feature representation are merged to construct a stronger and more comprehensive feature representation. This conceptual design leads to an efficient and effective architecture for real-time semantic segmentation, as illustrated in Figure 2 (c).

Specifically, in this study, we design a *Guided Aggregation Layer* to merge both types of features effectively. To further improve the performance without increasing the inference complexity, we present a *booster* training strategy with a series of auxiliary prediction heads, which can be discarded in the inference phase. Extensive quantitative and qualitative evaluations demonstrate that the proposed architecture performs favourably against *state-of-the-art* real-time semantic segmentation approaches, as shown in Figure 1.

The main contributions are summarized as follows:

- We propose an efficient and effective two-pathway architecture, termed Bilateral Segmentation Network, for real-time semantic segmentation, which treats the spatial details and categorical semantics separately.
- For the Semantic Branch, we design a new lightweight network based on depth-wise convolutions to enhance the receptive field and capture rich contextual information.
- A booster training strategy is introduced to further improve the segmentation performance without increasing the inference cost.
- Our architecture achieves impressive results on different benchmarks of Cityscapes (Cordts et al., 2016), CamVid (Brostow et al., 2008a), and COCO-Stuff (Caesar et al., 2018). More specifically, we obtain the results of 72.6% mean IoU on the Cityscapes



(a) Dilation Backbone (b) Encoder-Decoder Backbone (c) Bilateral Segmentation Backbone

Fig. 2 Illustration of different backbone architectures. (a) is the dilation backbone network, which removes the downsampling operations and upsampling the corresponding convolution filters. It has heavy computation complexity and memory footprint. (b) is the encoder-decoder backbone network, which adds extra top-down and lateral connections to recover the high-resolution feature map. These connections in the network are less friendly to the memory access cost. To achieve high accuracy and high efficiency simultaneously, we design the (c) Bilateral Segmentation backbone network. This architecture has two pathways, Detail Branch for spatial details and Semantic Branch for categorical semantics. The detail branch has wide channels and shallow layers, while the semantic branch has narrow channels and deep layers, which can be made very lightweight by the factor (λ, e.g., 1/4).

test set with the speed of 156 FPS on one NVIDIA GeForce GTX 1080Ti card.

A preliminary version of this work was published in (Yu et al., 2018a). We have extended our conference version as follows. (i) We simplify the original structure to present an efficient and effective architecture for real-time semantic segmentation. We remove the time-consuming cross-layer connections in the original version to obtain a more clear and simpler architecture. (ii) We re-design the overall architecture with more compact network structures and well-designed components. Specifically, we deepen the Detail Path to encode more details. We design light-weight components based on the depth-wise convolutions for the Semantic Path. Meanwhile, we propose an efficient aggregation layer to enhance the mutual connections between both paths. (iii) We conduct comprehensive ablative experiments to elaborate on the effectiveness and efficiency of the proposed method. (iv) We have significantly improved the accuracy and speed of the method in our previous work, i.e., for a 2048×1024 input, achieving 72.6% Mean IoU on the Cityscapes *test* set with a speed of 156 FPS on one NVIDIA GeForce GTX 1080Ti card.

2 Related Work

Recent years have witnessed significant advances in image semantic segmentation. In this section, our discussion mainly focuses on three groups of methods most relevant to our work, i.e., generic semantic segmentation methods, real-time semantic segmentation methods, and light-weight architectures.

2.1 Generic Semantic Segmentation

Traditional segmentation methods based on the threshold selection (Otsu, 1979), the region growing (Vincent and Soille, 1991), the super-pixel (Ren and Malik, 2003; Achanta et al., 2012; Van den Bergh et al., 2012) and the graph (Boykov and Jolly, 2001; Rother et al., 2004) algorithms adopt the hand-crafted features to solve this problem. Recently, a new generation of algorithms based on FCN (Long et al., 2015; Shelhamer et al., 2017) keep improving *state-of-the-art* performance on different benchmarks. Various methods are based on two types of backbone network: (i) *dilation backbone network*; (ii) *encoder-decoder backbone network*.

On one hand, the dilation backbone removes the downsampling operations and upsamples the convolution filter to preserve high-resolution feature representations. Due to the simplicity of the dilation convolution, various methods (Chen et al., 2015, 2018; Zhao et al., 2017; Wang et al., 2018a; Zhang et al., 2018a; Yu et al., 2020) develop different novel and effective components on it. The Deeplabv3 (Chen et al., 2017) devises an atrous spatial pyramid pooling to capture multi-scale context, while the PSPNet (Zhao et al., 2017) adopts a pyramid pooling module on the dilation backbone. Meanwhile, some methods introduce the attention mechanisms, e.g., self-attention (Yuan and Wang, 2018; Fu et al., 2019; Yu et al., 2020), spatial attention (Zhao et al., 2018b) and channel attention (Zhang et al., 2018a), to capture long-range context based on the dilation backbone.

On the other hand, the encoder-decoder backbone network adds extra top-down and lateral connections to recover the high-resolution feature maps in the decoder part. FCN and Hypercolumns (Hariharan et al., 2015) adopt the skip connection to integrate the low-level feature. Meanwhile, U-net (Ronneberger et al., 2015), SegNet with saved pooling indices (Badrinarayanan et al., 2017), RefineNet with multi-path refinement (Lin et al., 2017), LRR with step-wise reconstruction (Ghiasi and Fowlkes, 2016), GCN with "large kernel" convolution (Peng et al., 2017) and DFN with channel attention module (Yu et al., 2018b) incorporate this backbone network to recover the detailed information. HRNet (Wang et al., 2019) adopts multi-branches to maintain the high resolution.

Both types of backbone network encode the low-level details and high-level semantics simultaneously with the wide and deep networks. Although both types of backbone network achieve state-of-the-art performance, most methods run at a slow inference speed. In this study, we propose a novel and efficient architecture to treat the spatial details and categorical semantics separately to achieve a good trad-off between segmentation accuracy and inference speed.

2.2 Real-time Semantic Segmentation

Real-time semantic segmentation algorithms attract increasing attention when a growing practical applications require fast interaction and response. SegNet (Badrinarayanan et al., 2017) uses a small network structure and the skip connection to achieve a fast speed. E-Net (Paszke et al., 2016) devises a lightweight network from scratch and delivers extremely high speed. ICNet (Zhao et al., 2018a) uses the image cascade to speed up the algorithm, while DLC (Li et al., 2017) employs a cascade network structure to reduce the computation in "easy regions". ERFNet (Romera et al., 2018) adopts the residual connection and factorized convolutions to remain efficient and retain accuracy. Meanwhile, ESPNet (Mehta et al., 2018, 2019) devises an efficient spatial pyramid dilated convolution for real-time semantic segmentation. GUN (Mazzini, 2018) employs a guided upsampling module to fuse the information of the multi-resolution input. DFANet (Li et al., 2019b) reuses the feature to enhance the feature representation and reduces the complexity.

Although these methods can achieve a real-time inference speed, they dramatically sacrifice the accuracy to the efficiency with the loss of the low-level details. In this work, we take both of the low-level details and high-level semantics into consideration to achieve high accuracy and high efficiency.

2.3 Light-weight Architecture

Following the pioneering work of group/depth-wise convolution and separable convolution, light-weight architecture design has achieved rapid development, including Xception (Chollet, 2017), MobileNet (Howard et al., 2017; Sandler et al., 2018), ShuffleNet (Zhang et al., 2018b; Ma et al., 2018), to name a few. These methods achieve a valuable trade-off between speed and accuracy for the classification task. In this study, we design a light-weight network given computation complexity, memory access cost and real inference speed for the real-time semantic segmentation.

3 Core Concepts of BiSeNetV2

Our architecture consists of a Detail Branch (Section 3.1) and a Semantic Branch (Section 3.2), which are merged by an Aggregation Layer (Section 3.3). In this section, we demonstrate the core concepts of our architecture, as illustrated in Figure 2(c).

3.1 Detail Branch

The Detail Branch is responsible for the spatial details, which is low-level information. Therefore, this branch requires a rich channel capacity to encode affluent spatial detailed information. Meanwhile, because the Detail Branch simply focuses on the low-level details, we can design a shallow structure with a *small* stride for this branch. Overall, the key concept of the Detail Branch is to use *wide* channels and *shallow* layers for the spatial details. Besides, the feature representation in this branch has a large spatial size and wide channels. Therefore, it is better not to adopt the residual connections, which increases the memory access cost and reduce the speed.

3.2 Semantic Branch

In parallel to the Detail Branch, the Semantic Branch is designed to capture high-level semantics. This branch has *low* channel capacity, while the spatial details can be provided by the Detail Branch. In contrast, in our experiments, the Semantic Branch has a ratio of $\lambda (\lambda < 1)$ channels of the Detail Branch, which makes this branch lightweight. Actually, the Semantic Branch can be any lightweight convolutional model (e.g., (Chollet, 2017; Iandola et al., 2016; Howard et al., 2017; Sandler et al., 2018; Zhang et al., 2018b; Ma et al., 2018)). Meanwhile, the Semantic Branch adopts the

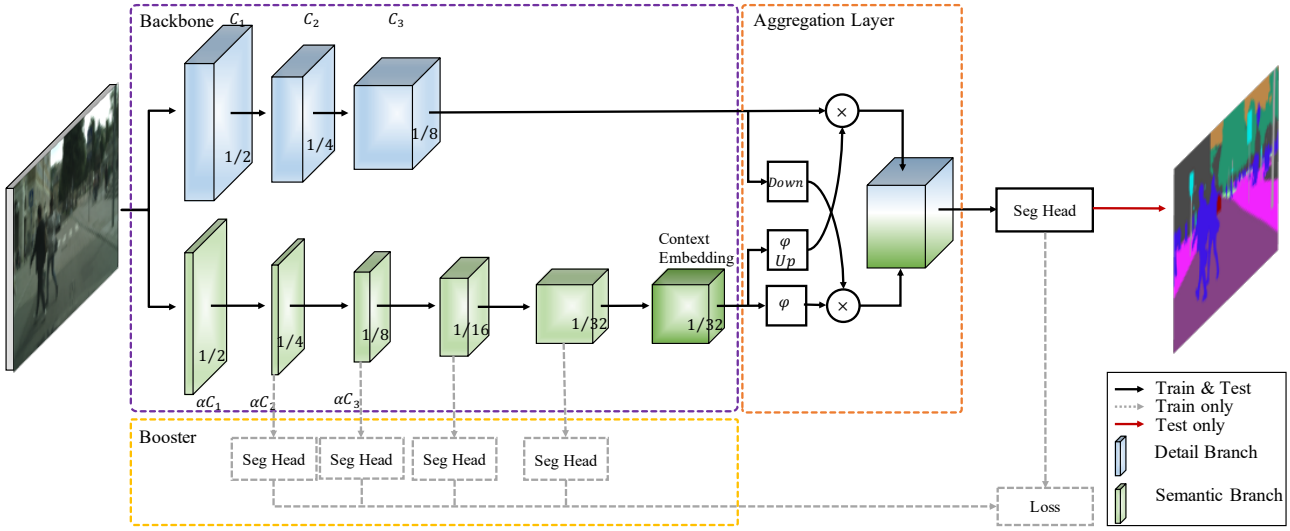


Fig. 3 Overview of the Bilateral Segmentation Network. There are mainly three components: two-pathway backbone in the purple dashed box, the aggregation layer in the orange dashed box, and the booster part in the yellow dashed box. The two-pathway backbone has a Detail Branch (the blue cubes) and a Semantic Branch (the green cubes). The three stages in Detail Branch have C_1, C_2, C_3 channels respectively. The channels of corresponding stages in Semantic Branch can be made lightweight by the factor $\lambda(\lambda < 1)$. The last stage of the Semantic Branch is the output of the Context Embedding Block. Meanwhile, numbers in the cubes are the feature map size ratios to the resolution of the input. In the Aggregation Layer part, we adopt the bilateral aggregation layer. *Down* indicates the downsampling operation, *Up* represents the upsampling operation, φ is the Sigmoid function, and $\bigotimes$ means element-wise product. Besides, in the booster part, we design some auxiliary segmentation heads to improve the segmentation performance without any extra inference cost.

Table 1 Instantiation of the Detail Branch and Semantic Branch. Each stage S contains one or more operations *opr* (e.g., *Conv2d*, *Stem*, *GE*, *CE*). Each operation has a kernels size k, stride s and output channels c, repeated r times. The expansion factor e is applied to expand the channel number of the operation. Here the channel ratio is $\lambda = 1/4$. The green colors mark *fewer* channels of Semantic Branch in the corresponding stage of the Detail Branch. Notation: *Conv2d* means the convolutional layer, followed by one batch normalization layer and relu activation function. *Stem* indicates the stem block. *GE* represents the gather-and-expansion layer. *CE* is the context embedding block.

Stage	Detail Branch					Semantic Branch						Output Size
	opr	*k*	*c*	*s*	*r*	*opr*	*k*	*c*	*e*	*s*	*r*	
Input												512×1024
S_1	Conv2d	3	64	2	1	Stem	3	16	-	4	1	256×512
	Conv2d	3	64	1	1							256×512
S_2	Conv2d	3	64	2	1							128×256
	Conv2d	3	64	1	2							128×256
S_3	Conv2d	3	128	2	1	GE	3	32	6	2	1	64×128
	Conv2d	3	128	1	2	GE	3	32	6	1	1	64×128
S_4						GE	3	64	6	2	1	32×64
						GE	3	64	6	1	1	32×64
S_5						GE	3	128	6	2	1	16×32
						GE	3	128	6	1	3	16×32
						CE	3	128	-	1	1	16×32

fast-downsampling strategy to promote the level of the feature representation and enlarge the receptive field quickly. High-level semantics require *large* receptive field. Therefore, the Semantic Branch employs the global average pooling (Liu et al., 2016) to embed the global contextual response.

3.3 Aggregation Layer

The feature representation of the Detail Branch and the Semantic Branch is complementary, one of which is unaware of the information of the other one. Thus, an Aggregation Layer is designed to merge both types of feature representation. Due to the fast-downsampling strategy, the spatial dimensions of the Semantic Branch's output are smaller than the Detail Branch. We need

to upsample the output feature map of the Semantic Branch to match the output of the Detail Branch.

There are a few manners to fuse information, e.g., simple *summation*, *concatenation* and some well-designed operations. We have experimented different fusion methods with consideration of accuracy and efficiency. At last, we adopt the bidirectional aggregation method, as shown in Figure 3.

4 Bilateral Segmentation Network

The concept of our BiSeNet is generic, which can be implemented with different convolutional models (He et al., 2016; Huang et al., 2017; Chollet, 2017; Iandola et al., 2016; Howard et al., 2017; Sandler et al., 2018; Zhang et al., 2018b; Ma et al., 2018) and any specific designs. There are mainly three key concepts: (i) The Detail Branch has *high* channel capacity and *shallow* layers with *small* receptive field for the spatial details; (ii)The Semantic Branch has *low* channel capacity and *deep* layers with *large* receptive field for the categorical semantics. (iii)An efficient Aggregation Layer is designed to fuse both types of representation.

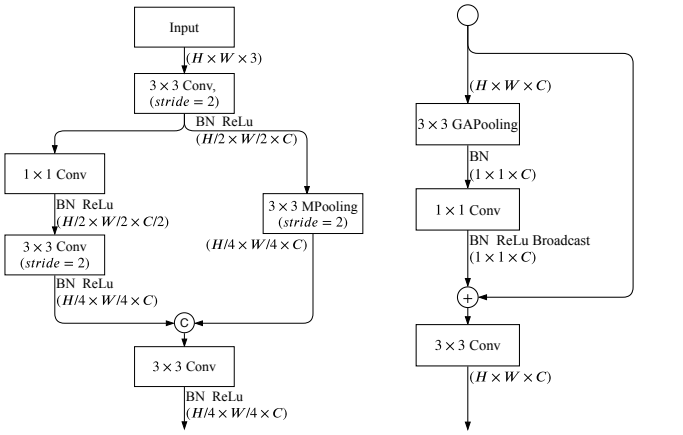
In this subsection, according to the proposed conceptual design, we demonstrate our instantiations of the overall architecture and some other specific designs, as illustrated in Figure 3.

4.1 Detail Branch

The instantiation of the Detail Branch in Table 1 contains three stages, each layer of which is a convolution layer followed by batch normalization (Ioffe and Szegedy, 2015) and activation function (Glorot et al., 2011). The first layer of each stage has a stride $s = 2$, while the other layers in the same stage have the same number of filters and output feature map size. Therefore, this branch extracts the output feature maps that are 1/8 of the original input. This Detail Branch encodes rich spatial details due to the high channel capacity. Because of the high channel capacity and the large spatial dimension, the residual structure (He et al., 2016) will increases the memory access cost (Ma et al., 2018). Therefore, this branch mainly obeys the philosophy of VGG nets (Simonyan and Zisserman, 2015) to stack the layers.

4.2 Semantic Branch

In consideration of the large receptive field and efficient computation simultaneously, we design the Semantic Branch, inspired by the philosophy of the lightweight recognition model, e.g., Xception (Chollet, 2017), MobileNet (Howard et al., 2017; Sandler et al., 2018; Howard et al., 2019), ShuffleNet (Zhang et al., 2018b; Ma et al., 2018). Some of the key features of the Semantic Branch are as follows.



(a) Stem Block (b) Context Embedding Block

Fig. 4 Illustration of Stem Block and Context Embedding Block. (a) is the Stem Block, which adopts a fast-downsampling strategy. This block has two branches with different manners to downsample the feature representation. Then both feature response of two branches is concatenated as the output. (b) is the Context Embedding Block. As demonstrated in Section 3.2, the Semantic Branch requires large receptive field. Therefore, we design a Context Embedding Block with the global average pooling to embed the global contextual information. Notation: *Conv* is convolutional operation. *BN* is the batch normalization. *ReLu* is the ReLu activation function. *Mpooling* is the max pooling. *GPooling* is the global average pooling. and C means concatenation. Meanwhile, $1 \times 1, 3 \times 3$ denote the kernel size, $H \times W \times C$ means the tensor shape (height, width, depth).

Stem Block Inspired from (Szegedy et al., 2017; Shen et al., 2017; Wang et al., 2018b), we adopt the Stem Block as the first stage of the Semantic Branch, as illustrated in Figure 4. It uses two different downsampling manners to shrink the feature representation. And then the output feature of both branches are concatenated as the output. This structure has efficient computation cost and effective feature expression ability.

Context Embedding Block As discussed in Section 3.2, the Semantic Branch requires large receptive field to capture high-level semantics. Inspired from (Yu et al., 2018b; Liu et al., 2016; Zhao et al., 2017; Chen et al., 2017), we design the Context Embedding Block. This block uses the global average pooling and residual connection (He et al., 2016) to embed the global contextual information efficiently, as showed in Figure 4.

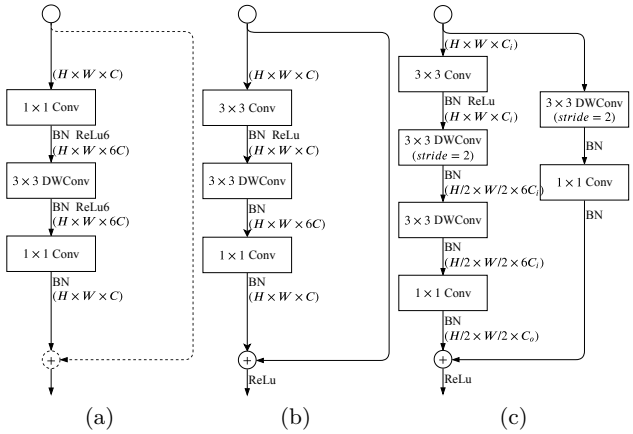


Fig. 5 Illustration of Inverted Bottleneck and Gather-and-Expansion Layer. (a) is the mobile inverted bottleneck Conv proposed in MobileNetv2. The dashed shortcut path and summation circle do not exist with the $stride = 2$. (b)(c) are the proposed Gather-and-Expansion Layer. The bottleneck structure adopts: (i) a 3×3 convolution to gather local feature response and expand to higher-dimensional space;(ii) a 3×3 depth-wise convolution performed independently over each individual output channel of the expansion layer; (iii) a 1×1 convolution as the projection layer to project the output of depth-wise convolution into a low channel capacity space. When the $stride = 2$, we adopt two $kernel\_size = 3$ depth-wise convolutions on the main path and a 3×3 separable convolution as the shortcut. Notation: *Conv* is convolutional operation. *BN* is the batch normalization. *ReLu* is the ReLu activation function. Meanwhile, $1 \times 1, 3 \times 3$ denote the kernel size, $H \times W \times C$ means the tensor shape (height, width, depth).

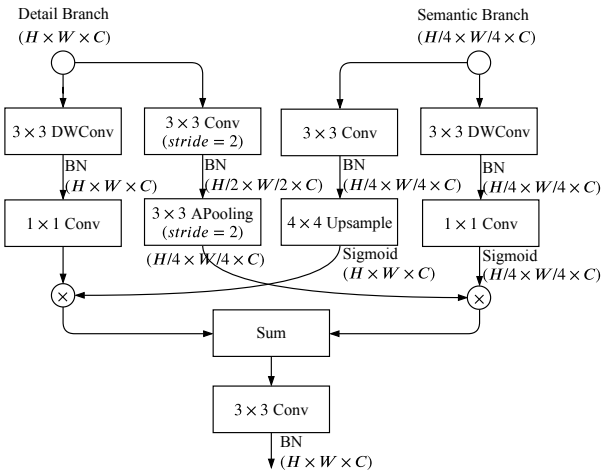


Fig. 6 Detailed design of Bilateral Guided Aggregation Layer. Notation: *Conv* is convolutional operation. *DWConv* is depth-wise convolution. *APooling* is average pooling. *BN* denotes the batch normalization. *Upsample* means bilinear interpolation. *Sigmoid* is the Sigmoid activation function. *Sum* means summation. Meanwhile, $1 \times 1, 3 \times 3$ denote the kernel size, $H \times W \times C$ means the tensor shape (height, width, depth), $\otimes$ represents element-wise product.

Gather-and-Expansion Layer Taking advantage of the benefit of depth-wise convolution, we propose the Gather-and-Expansion Layer, as illustrated in Figure 5. The Gather-and-Expansion Layer consists of: (i) a 3×3 convolution to efficiently aggregate feature responses and expand to a higher-dimensional space; (iii) a 3×3 depth-wise convolution performed independently over each individual output channel of the expansion layer; (iv) a 1×1 convolution as the projection layer to project the output of depth-wise convolution into a low channel capacity space. When $stide = 2$, we adopt two 3×3 depth-wise convolution, which further enlarges the receptive field, and one 3×3 separable convolution as the shortcut. Recent works (Tan et al., 2019; Howard et al., 2019) adopt 5×5 separable convolution heavily to enlarge the receptive field, which has fewer FLOPS than two 3×3 separable convolution in some conditions. In this layer, we replace the 5×5 depth-wise convolution in the separable convolution with two 3×3 depth-wise convolution, which has fewer FLOPS and the same receptive field.

In contrast to the inverted bottleneck in MobileNetv2, the GE Layer has one more 3×3 convolution. However, this layer is also friendly to the computation cost and memory access cost (Ma et al., 2018; Sandler et al., 2018), because the 3×3 convolution is specially optimized in the CUDNN library (Chetlur et al., 2014; Ma et al., 2018). Meanwhile, because of this layer, the GE Layer has higher feature expression ability than the inverted bottleneck.

4.3 Bilateral Guided Aggregation

There are some different manners to merge two types of feature response, i.e., element-wise summation and concatenation. However, the outputs of both branches have different levels of feature representation. The Detail Branch is for the low-level, while the Semantic Branch is for the high-level. Therefore, simple combination ignores the diversity of both types of information, leading to worse performance and hard optimization.

Based on the observation, we propose the Bilateral Guided Aggregation Layer to fuse the complementary information from both branches, as illustrated in Figure 6. This layer employs the contextual information of Semantic Branch to guide the feature response of Detail Branch. With different scale guidance, we can capture different scale feature representation, which inherently encodes the multi-scale information. Meanwhile, this guidance manner enables efficient communication between both branches compared to the simple combination.

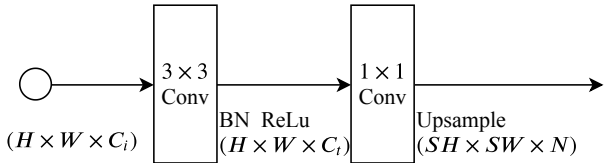


Fig. 7 Detailed design of Segmentation Head in Booster. Notation: *Conv* is convolutional operation. *BN* denotes the batch normalization. *Upsample* means bilinear interpolation. Meanwhile, $1 \times 1, 3 \times 3$ denote the kernel size, $H \times W \times C$ means the tensor shape (height, width, depth), C represents the channel dimension, S denotes the scale ratio of upsampling, and N is the final output dimension.

4.4 Booster Training Strategy

To further improve the segmentation accuracy, we propose a booster training strategy. As the name implies, it is similar to the rocket booster: it can enhance the feature representation in the training phase and can be discarded in the inference phase. Therefore, it increases little computation complexity in the inference phase. As illustrated in Figure 3, we can insert the auxiliary segmentation head to different positions of the Semantic Branch. In Section 5.1, we analyze the effect of different positions to insert. Figure 7 illustrates the details of the segmentation head. We can adjust the computational complexity of auxiliary segmentation head and main segmentation head by controlling the channel dimension C_t.

5 Experimental Results

In this section, we first introduce the datasets and the implementation details. Next, we investigate the effects of each component of our proposed approach on Cityscapes validation set. Finally, we report our final accuracy and speed results on different benchmarks compared with other algorithms.

Datasets. Cityscapes (Cordts et al., 2016) focuses on semantic understanding of urban street scenes from a car perspective. The dataset is split into training, validation and test sets, with $2,975$, 500 and $1,525$ images respectively. In our experiments, we only use the fine annotated images to validate the effectiveness of our proposed method. The annotation includes 30 classes, 19 of which are used for semantic segmentation task. This dataset is challenging for the real-time semantic segmentation because of its high resolution of $2,048 \times 1,024$.

Cambridge-driving Labeled Video Database (CamVid) (Brostow et al., 2008a) is a road scene dataset from the perspective of a driving automobile. It contains 701 images with 960×720 resolution extracted from the video sequences. Following the pioneering work (Brostow et al., 2008b; Sturgess et al., 2009; Badrinarayanan et al., 2017), the images are split into 367 for training, 101 for validation and 233 for testing. We use the subset of 11 classes of the provided 32 candidate categories for the fair comparison with other methods. The pixels do not belong to one of these classes are ignored.

COCO-Stuff (Caesar et al., 2018) augments 10K complex images of the popular COCO (Lin et al., 2014) dataset with dense stuff annotations. This is also a challenging dataset for the real-time semantic segmentation because it has more complex categories, including 91 thing and 91 stuff classes for evaluation. For a fair comparison, we follow the split in (Caesar et al., 2018): 9K images for training and 1K images for testing.

Training. Our models are trained from scratch with the "kaiming normal" initialization manner (He et al., 2015). We use the stochastic gradient descent (SGD) algorithm with 0.9 momentum to train our model. For all datasets, we adopt 16 batch size. For the Cityscapes and CamVid datasets, the weight decay is 0.0005 weight decay while the weight decay is 0.0001 for the COCO-Stuff dataset. We note that the weight decay regularization is only employed on the parameters of the convolution layers. The initial rate is set to $5e^{-2}$ with a "poly" learning rate strategy in which the initial rate is multiplied by $(1 - \frac{iter}{iters_{max}})^{power}$ each iteration with power 0.9. Besides, we train the model for 150K, 10K, 20K iterations for the Cityscapes dataset, CamVid dataset, and COCO-Stuff datasets respectively.

For the augmentation, we randomly horizontally flip, randomly scale, and randomly crop the input images to a fixed size for training. The random scales contain { 0.75, 1, 1.25, 1.5, 1.75, 2.0}. And the cropped resolutions are 2048×1024 for Cityscapes, 960×720 for CamVid, 640×640 for COCO-Stuff respectively. Besides, the augmented input of Cityscapes will be resized into 1024×512 resolution to train our model.

Inference. We do not adopt any evaluation tricks, e.g., sliding-window evaluation and multi-scale testing, which can improve accuracy but are time-consuming. With the input of 2048×1024 resolution, we first resize it to 1024×512 resolution to inference and then resize the prediction to the original size of the input. We measure the inference time with only one GPU card and repeat 5000 iterations to eliminate the error fluctuation. We note that the time of resizing is included in the inference time measurement. In other words, when measuring the inference time, the practical input size is 2048×1024. Meanwhile, we adopt the standard metric of the mean intersection of union (mIoU)

Table 2 Ablations on Cityscapes. We validate the effectiveness of each component step by step. We show segmentation accuracy (mIoU%), and computational complexity measured in GFLOPs with the input of spatial size 2048 × 1024. Notation: *Detail* is the Detail Branch. *Semantic* is the Semantic Branch. *BGA* represents the Bilateral Guided Aggregation Layer. *Booster* means the booster training strategy. *OHEM* is the online hard example mining.

Detail	Semantic	Aggregation			Booster	OHEM	mIoU(%)	GFLOPs
		Sum	Concate	BGA				
✓							62.35	15.26
	✓						64.68	7.63
✓	✓	✓					68.60	20.77
✓	✓		✓				68.93	21.98
✓	✓			✓			69.67	21.15
✓	✓			✓	✓		73.19	21.15
✓	✓			✓	✓	✓	**73.36**	21.15

Table 3 Ablations on the Semantic Branch design on Cityscapes. We conduct experiments about the channel capacity, the block design, and the expansion ratio of the Semantic Branch. Notation: *GLayer* indicates the Gather Layer, the first 3 × 3 convolution in GE Layer. *DDWConv* is double depth-wise convolution layer.

	mIoU(%)	GFLOPs
Detail-only	62.35	15.26
$\lambda = 1/2$	69.66	25.84
1/4	**69.67**	21.15
1/8	69.26	19.93
1/16	68.27	19.61

(a) **Channel capacity ratio**: Varying values of λ can control the channel capacity of the first two stages in the Semantic Branch. The channel dimensions of the last two stages are still 64 and 128. Here, we choose $\lambda = 1/4$.

GLayer	DDWConv	Context	mIoU(%)	GFLOPs
✓	✓	✓	**69.67**	21.15
✓	✓		69.01	21.07
✓		✓	68.98	21.15
	✓	✓	66.62	15.78

(b) **Block Analysis**: We specifically design the GE Layer and adopt double depth-wise convolutions when *stride* = 2. The second row means we use one 5 × 5 depth-wise convolution instead of two 3 × 3 depth-wise convolution. The third row represents we replace the first 3 × 3 convolution layer of GE Layer with the 1 × 1 convolution.

	mIoU(%)	GFLOPs
Detail-only	62.35	15.26
$\epsilon = 1$	67.48	17.78
2	68.41	18.45
4	68.78	19.8
6	**69.67**	21.15
8	68.99	22.49

(c) **Expansion ratio**: Varying values of ϵ can affect the representative ability of the Semantic Branch. We choose the $\epsilon = 6$ to make the trade-off between accuracy and computation complexity.

for the Cityscapes dataset and CamVid dataset, while the mIoU and pixel accuracy (pixAcc) for the COCO-Stuff dataset.

Setup. We conduct experiments based on PyTorch 1.0. The measurement of inference time is executed on one NVIDIA GeForce GTX 1080Ti with the CUDA 9.0, CUDNN 7.0 and TensorRT v5.1.5[1].

5.1 Ablative Evaluation on Cityscapes

This section introduces the ablation experiments to validate the effectiveness of each component in our method. In the following experiments, we train our models on Cityscapes (Cordts et al., 2016) training set and evaluate on the Cityscapes validation set.

Individual pathways. We first explore the effect of individual pathways specifically. The first two rows in Table 2 illustrates the segmentation accuracy and computational complexity of using only one pathway alone. The Detail Branch lacks sufficient high-level semantics, while the Semantic Branch suffers from a lack of low-level spatial details, which leads to unsatisfactory results. Figure 8 illustrates the gradual attention on the spatial details of Detail Branch. The second group in Table 2 shows that the different combinations of both branches are all better than the only one pathway models. Both branches can provide a complementary representation to achieve better segmentation performance. The Semantic Branch and Detail Branch alone only achieve 64.68% and 62.35% mean IoU. However, with the simple summation, the Semantic Branch can bring in over 6% improvement to the Detail Branch, while the Detail Branch can acquire 4% gain for the Semantic Branch. This observation shows that the representations of both branches are complementary.

Aggregation methods. We also investigate the aggregation methods of two branches, as illustrated in Table 2. For an effective and efficient aggregation, we design the Bilateral Guided Aggregation Layer, which adopts the high-level semantics as the guidance to aggregate the multi-scale low-level details. We also show two variants without Bilateral Guided Aggregation Layer as the naive aggregation baseline: *summation* and *concatenation* of the outputs of both branches. For a fair

[1] We use FP32 data precision.

(a) Stage$_1$ (b) Stage$_2$ (c) Stage$_3$

Fig. 8 Examples showing visual explanations for the different stages of the Detail Branch. Following the Grad-CAM (Selvaraju et al., 2017), we visualize the Grad-CAMs of Detail Branch. The visualization shows that Detail Branch can focus on the spatial details, e.g., boundary, gradually.

comparison, the inputs of the *summation* and *concatenation* are through one separable layer respectively. Figure 9 demonstrates the visualization outputs of Detail Branch, Semantic Branch and the aggregation of both branches. This illustrates that Detail Branch can provide sufficient spatial details, while Semantic Branch captures the semantic context.

Table 3 illustrates a series of analysis experiments on the Semantic Branch design.

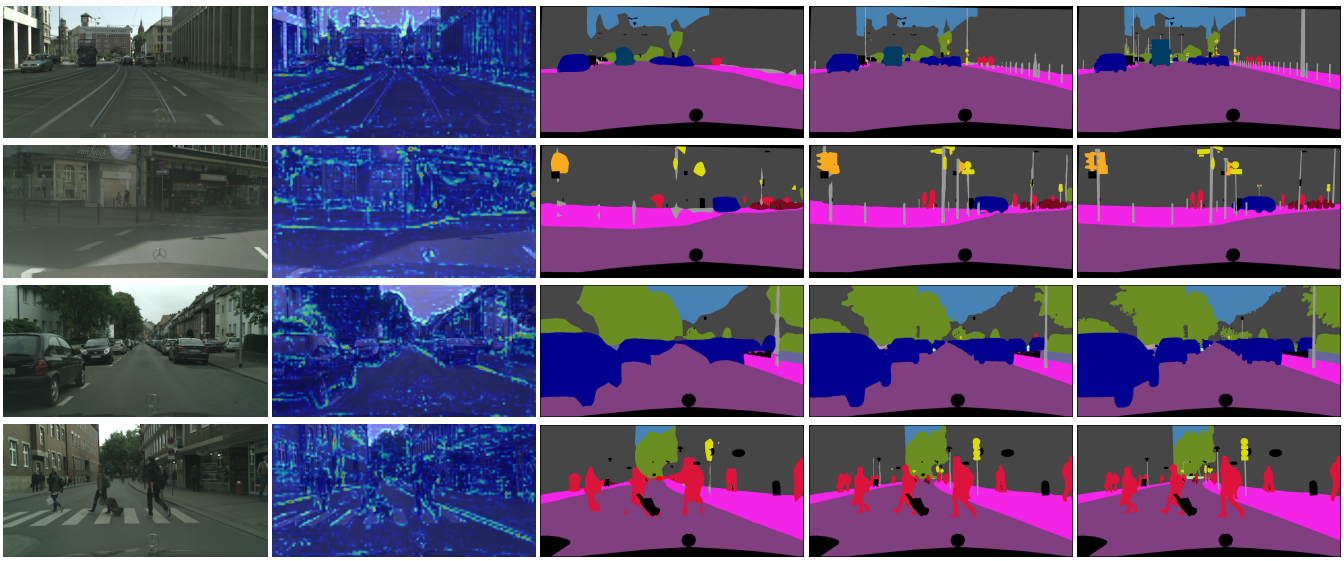
Channel capacity of Semantic Branch. As discussed in Section 3.2 and Section 4.2, the Semantic Branch is responsible for the high-level semantics, without caring the spatial details. Therefore, the Semantic Branch can be made very lightweight with low channel capacity, which is adapted by the channel capacity ratio of λ. Table 4a illustrates the detailed comparison experiments of varying λ.

Different λ brings in different extents of improvement to the Detail-only baseline. Even when $\lambda = 1/16$, the first layer of the Semantic Branch has only 4 channel dimension, which also brings in 6% (62.35% $\rightarrow$ 68.27%) improvement to the baseline. Here, we employ $\lambda = 1/4$ as our default.

Table 4 Booster position. We can add the auxiliary segmentation head to different positions as the booster of the Semantic Branch. Here, stage$_s$ represents the auxiliary segmentation head can be added after s stage. stage$_{5_4}$ and stage$_{5_5}$ means the position before or after the Context Embedding Block respectively. OHEM represents the online bootstrapping strategy.

stage$_2$	stage$_3$	stage$_4$	stage$_{5_4}$	stage$_{5_5}$	OHEM	mIoU(%)
						69.67
✓	✓	✓	✓	✓		73.04
✓	✓	✓	✓			<u>73.19</u>
✓	✓	✓				71.62
	✓	✓	✓	✓		72.84
		✓	✓	✓		72.68
			✓	✓		72.03
✓	✓	✓	✓		✓	**73.36**

Block design of of Semantic Branch. Following the pioneer work (Sandler et al., 2018; Howard et al., 2019), we design a Gather-and-Expansion Layer, as discussed in Section 4.2 and illustrated in Figure 5. The main improvements consist of two-fold: (i) we adopt one 3×3 convolution as the Gather Layer instead of one



(a) Input (b) Detail Branch (c) Semantic Branch (d) Prediction (e) Groundtruth

Fig. 9 Visual improvement of the Bilateral Guided Aggregation layer on the Cityscapes *val* set.

Table 5 Generalization to large models. We enlarge our models from two aspects: (i) wider models; (ii) deeper models.

Wider	mIoU(%)	GFLOPs
$\alpha = 1.0$	73.36	21.15
1.25	73.61	34.98
1.50	74.67	49.46
1.75	74.04	66.45
2.0	**75.11**	85.94

(a) **Wider models**: Varying values of α can control the channel capacity of our architecture.

Deeper	mIoU(%)	GFLOPs
$d = 1.0$	73.36	21.15
2.0	74.10	25.26
3.0	**74.28**	29.38
4.0	74.02	33.5

(b) **Deeper models**: Varying values of d represents the layer number of the model.

Table 6 Compatibility with other models. We empoly different light-weight models as the Semantic Branch to explore the compatibility of our architecture.

Semantic Branch	Pretrained	mIoU(%)	GFLOPs
ShuffleNetV2 1.5×	ImageNet	74.07	128.71
MobileNetV2	ImageNet	72.95	129.45
ResNet-18	ImageNet	75.22	143.34
Ours($\alpha = 1.0, d = 1.0$)	no	73.36	21.15
Ours($\alpha = 2.0, d = 3.0$)	no	**75.80**	118.51

point-wise convolution in the inverted bottleneck of MobileNetV2 (Sandler et al., 2018); (ii) when *stride* $= 2$, we employs two 3×3 depth-wise convolution to substitute a 5×5 depth-wise convolution.

Table 4b shows the improvement of our block design. The Gather-and-Expansion Layer can enlarge the receptive field to capture high-level semantics efficiently.

Expansion ratio of GE layer. The first 3×3 convolution layer in GE Layer is also an expansion layer, which can project the input to a high-dimensional space. It has an advantage in memory access cost (Sandler et al., 2018; Howard et al., 2019). The expansion ratio of ϵ can control the output dimension of this layer. Table 4c investigates the effect of varying ϵ. It is surprising to see that even with $\epsilon = 1$, the Semantic Branch can also improve the baseline by 4% (62.35% $\rightarrow$ 67.48%) mean IoU, which validates the lightweight Semantic Branch is efficient and effective.

Booster training strategy. We propose a booster training strategy to further improve segmentation accuracy, as discussed in Section 4.4. We insert the segmentation head illustrated in Figure 7 to different positions of Semantic Branch in the training phase, which are discarded in the inference phase. Therefore, they increase little computation complexity in the inference phase, which is similar to the booster of the rocket. Table 4 shows the effect of different positions to insert segmentation head. As we can see, the booster training strategy can obviously improve segmentation accuracy. We choose the configuration of the third row of Table 4, which further improves the mean IoU by over 3% (69.67% $\rightarrow$ 73.19%), without sacrificing the inference speed. Based on this configuration, we adopt the online bootstrapping strategy (Wu et al., 2016) to improve the performance further.

5.2 Generalization Capability

In this section, we mainly explore the generalization capability of our proposed architecture. First, we investigate the performance of a wider model and deeper

Table 7 Comparison with state-of-the-art on Cityscapes. We train and evaluate our models with 2048×1024 resolution input, which is resized into 1024×512 in the model. The inference time is measured on one NVIDIA GeForce 1080Ti card. Notation: γ is the downsampling ratio corresponding to the original 2048×1024 resolution. *backbone* indicates the backbone models pre-trained on the ImageNet dataset. "-" represents that the methods do not report the corresponding result. The DFANet A and DFANet B adopt the 1024×1024 input size and use the optimized depth-wise convolutions to accelerate speed.

method	ref.	γ	backbone	mIoU(%)		FPS
				val	*test*	
large models						
CRF-RNN*	ICCV2015	0.5	VGG16	-	62.5	1.4
DeepLab*	ICLR2015	0.5	VGG16	-	63.1	0.25
FCN-8S*	CVPR2015	1.0	VGG16	-	65.3	2
Dilation10	ICLR2016	1.0	VGG16	68.7	67.1	0.25
LRR	ECCV2016	1.0	VGG16	70.0	69.7	-
Deeplabv2	ICLR2016	1.0	ResNet101	71.4	70.4	-
FRRN	CVPR2017	0.5	no	-	71.8	2.1
RefineNet	CVPR2017	1.0	ResNet101	-	73.6	-
DUC	WACV2018	1.0	ResNet101	76.7	76.1	-
PSPNet	CVPR2017	1.0	ResNet101	-	**78.4**	0.78
small models						
ENet	arXiv2016	0.5	no	-	58.3	76.9
SQ	NIPSW2016	1.0	SqueezeNet	-	59.8	16.7
ESPNet	ECCV2018	0.5	ESPNet	-	60.3	112.9
ESPNetV2	CVPR2019	0.5	ESPNetV2	66.4	66.2	-
ERFNet	TITS2018	0.5	no	70.0	68.0	41.7
Fast-SCNN	BMVC2019	1.0	no	68.6	68.0	123.5
ICNet	ECCV2018	1.0	PSPNet50	-	69.5	30.3
DABNet	BMVC2019	1.0	no	-	70.1	27.7
DFANet B	CVPR2019	0.5*	Xception B	-	67.1	120
DFANet A′	CVPR2019	0.5	Xception A	-	70.3	**160**
DFANet A	CVPR2019	0.5*	Xception A	-	71.3	100
GUN	BMVC2018	0.5	DRN-D-22	69.6	70.4	33.3
SwiftNet	CVPR2019	1.0	ResNet18	75.4	75.5	39.9
BiSeNetV1	ECCV2018	0.75	Xception39	69.0	68.4	105.8
BiSeNetV1	ECCV2018	0.75	ResNet18	74.8	74.7	65.5
BiSeNetV2	—	0.5	no	73.4	72.6	156
BiSeNetV2-L	—	0.5	no	**75.8**	75.3	47.3

model in Table 5. Next, we replace the Semantic Branch with some other general light-weight models to explore the compatibility ability in Table 6.

Generalization to large models. Although our architecture is designed mainly for the light-weight task, e.g., real-time semantic segmentation, BiSeNet V2 can be also generalized to large models. We mainly enlarge the architecture from two aspects: (i) wider models, controlled by the width multiplier α; (ii) deeper models, controlled by the depth multiplier d. Table 5 shows the segmentation accuracy and computational complexity of wider models with the different width multiplier α and different depth multiplier d. According to the experiments, we choose $\alpha = 2.0$ and $d = 3.0$ to build our large architecture, termed BiSeNetV2-Large, which achieves 75.8% mIoU and GFLOPs.

Table 8 Comparison with state-of-the-art on CamVid. With 960×720 input, we evaluate the segmentation accuracy and corresponding inference speed. Notation: *backbone* means the backbone models pre-trained on extra datasets, e.g., ImageNet dataset and Cityscapes dataset. $*$ indicates the models are pre-trained on Cityscapes. † represents the models are trained from scratch.

method	ref.	backbone	mIoU(%)	FPS
large models				
SegNet	TPAMI2017	VGG16	60.1	4.6
DPN	ICCV2015	VGG16	60.1	1.2
Deeplab	ICLR2015	VGG16	61.6	4.9
RTA	ECCV2018	VGG16	62.5	0.2
Dilation8	ICLR2016	VGG16	65.3	4.4
PSPNet	CVPR2017	ResNet50	69.1	5.4
DenseDecoder	CVPR2018	ResNeXt101	70.9	-
VideoGCRF*	CVPR2018	ResNet101	75.2	-
small models				
ENet	arXiv2016	no	51.3	61.2
DFANet B	CVPR2019	Xception B	59.3	160
DFANet A	CVPR2019	Xception A	64.7	120
ICNet	ECCV2018	PSPNet50	67.1	27.8
SwiftNet	CVPR2019	ResNet18†	63.33	-
SwiftNet	CVPR2019	ResNet18	72.58	-
BiSeNetV1	ECCV2018	Xception 39	65.6	**175**
BiSeNetV1	ECCV2018	ResNet18	68.7	116.25
BiSeNetV2	—	no	72.4	124.5
BiSeNetV2-L	—	no	73.2	32.7
BiSeNetV2*	—	no	76.7	124.5
BiSeNetV2-L*	—	no	**78.5**	32.7

Compatibility with other models. The BiSeNetV2 is a generic architecture with two branches. In this work, we design some specific blocks for the Semantic Branch. The Semantic Branch can be any light-weight convolutional models (He et al., 2016; Howard et al., 2017). Therefore, to explore the compatibility ability of our architecture, we conduct a series of experiments with different general light-weight models. Table 6 shows the results of the combination with different models.

5.3 Performance Evaluation

In this section, we compare our best model (BiSeNetV2 and BiSeNetV2-Large) with other *state-of-the-art* methods on three benchmark datasets: Cityscapes, CamVid and COCO-Stuff.

Cityscapes. We present the segmentation accuracy and inference speed of the proposed BiSeNetV2 on Cityscapes *test* set. We use the training set and validation set with 2048×1024 input to train our models, which is resized into 1024×512 resolution at first in the models. Then we evaluate the segmentation accuracy on the test set. The measurement of inference time is conducted on one NVIDIA GeForce 1080Ti card. Table 7 reports the comparison results of our method and state-of-the-art methods. The first group is non-real-time methods, containing CRF-RNN (Zheng et al., 2015), Deeplab-CRF (Chen et al., 2015), FCN-8S (Long et al., 2015), Dilation10 (Yu and Koltun, 2016), LRR (Ghiasi and Fowlkes, 2016), Deeplabv2-CRF (Chen et al., 2016), FRRN (Pohlen et al., 2017), RefineNet (Lin et al., 2017), DUC (Wang et al., 2018a), PSPNet (Zhao et al., 2017). The real-time semantic segmentation algorithms are listed in the second group, including ENet (Paszke et al., 2016), SQ (Treml et al., 2016), ESPNet (Mehta et al., 2018), ESPNetV2 (Mehta et al., 2019), ERFNet (Romera et al., 2018), Fast-SCNN (Poudel et al., 2019), ICNet (Zhao et al., 2018a), DABNet (Li et al., 2019a), DFANet (Li et al., 2019b), GUN (Mazzini, 2018), SwiftNet (Orsic et al., 2019), BiSeNetV1 (Yu et al., 2018a). The third group is our methods with different levels of complexities. As shown in Table 7, our method achieves 72.6% mean IoU with 156 FPS and yields 75.3% mean IoU with 47.3 FPS, which are *state-of-the-art* trade-off between accuracy and speed. These results are even better than several non-real-time algorithms in the second group of Table 7. We note that many non-real-time methods may adopt some evaluation tricks, e.g., multi-scale testing and multi-crop evaluation, which can improve the accuracy but are time-consuming. Therefore, we do not adopt this strategy with the consideration of speed. To better view, we illustrate the trade-off between performance and speed in Figure 1. To highlight the effectiveness of our method, we also present some visual examples of BiSeNetV2 on Cityscapes in Figure 10.

CamVid. Table 8 shows the statistic accuracy and speed results on the CamVid dataset. In the inference phase, we use the training dataset and validation dataset to train our model with 960×720 resolution input. Our models are compared to some non-real-time algorithms, including SegNet (Badrinarayanan et al., 2017), Deeplab (Chen et al., 2015), RTA (Huang et al., 2018), Dilate8 (Yu and Koltun, 2016), PSPNet (Zhao et al., 2017), VideoGCRF (Chandra et al., 2018), and DenseDecoder (Bilinski and Prisacariu, 2018), and real-time algorithms, containing ENet (Paszke et al., 2016), ICNet (Zhao et al., 2018a), DABNet (Li et al., 2019a), DFANet (Li et al., 2019b), SwiftNet (Orsic et al., 2019), BiSeNetV1 (Yu et al., 2018a). BiSeNetV2 achieves much faster inference speed than other methods. Apart from the efficiency, our accuracy results also outperform these work. Besides, we investigate the effect of the pre-training datasets on CamVid. The last two rows of Table 8 show that pre-training on Cityscapes can greatly improve the mean IoU over 6% on the CamVid test set.

Table 9 Comparison with state-of-the-art on COCO-Stuff. Our models are trained and evaluated with the input of 640×640 resolution. Notation: *backbone* is the backbone models pre-trained on ImageNet dataset.

method	ref.	backbone	mIoU(%)	pixAcc(%)	FPS
large models					
FCN-16s	CVPR2017	VGG16	22.7	52.0	5.9
Deeplab	ICLR2015	VGG16	26.9	57.8	8.1
FCN-8S	CVPR2015	VGG16	27.2	60.4	-
PSPNet50	CVPR2017	ResNet50	**32.6**	-	6.6
small models					
ICNet	ECCV2018	PSPNet50	29.1	-	35.7
BiSeNetV2	—	no	25.2	60.5	**87.9**
BiSeNetV2-L	—	no	28.7	**63.5**	42.5

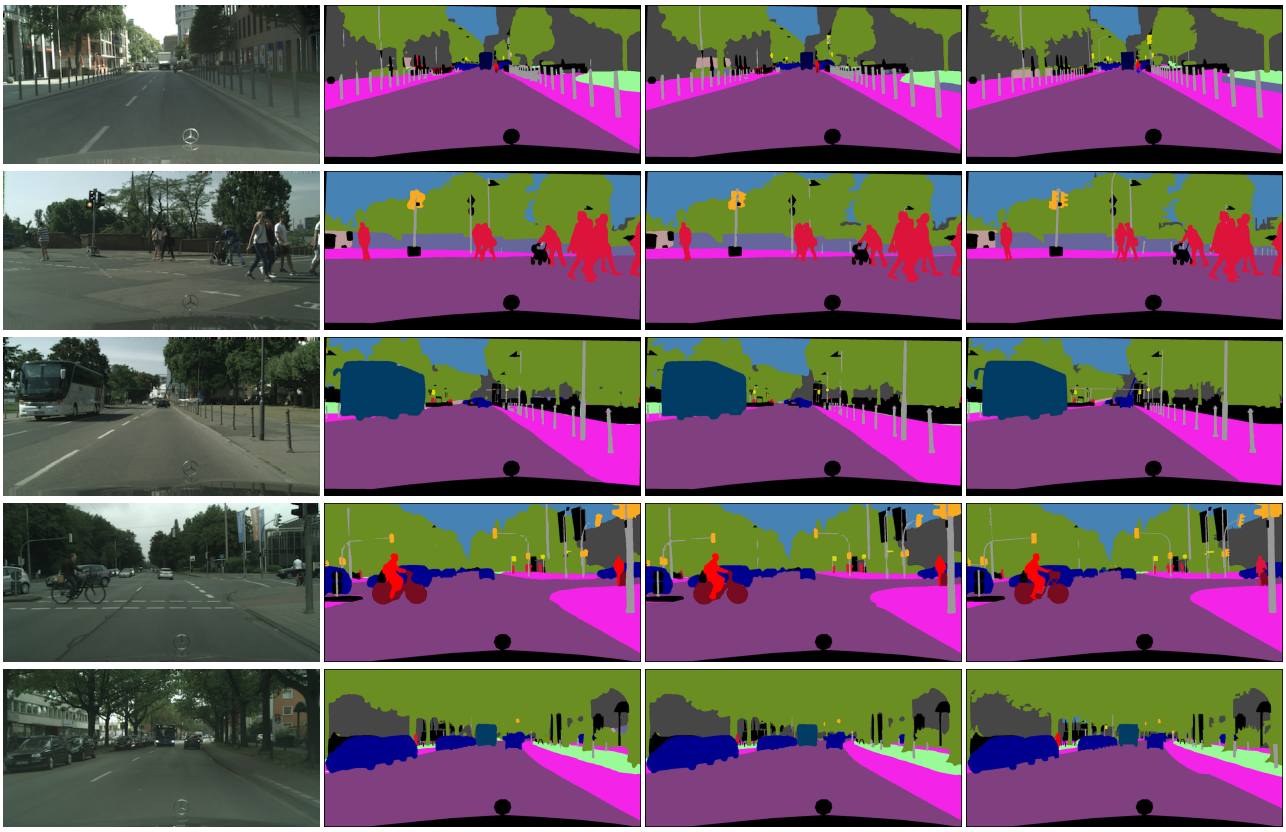
COCO-Stuff. We also report our accuracy and speed results on COCO-Stuff validation dataset in Table 9. In the inference phase, we pad the input into 640×640 resolution. For a fair comparison (Long et al., 2015; Chen et al., 2016; Zhao et al., 2017, 2018a), we do not adopt any time-consuming testing tricks, such as multi-scale and flipping testing. Even With more complex categories in this dataset, compared to pioneer work, our BiSeNetV2 still performs more efficient and achieve comparable accuracy.

6 Concluding Remarks

We observe that the semantic segmentation task requires both low-level details and high-level semantics. We propose a new architecture to treat the spatial details and categorical semantics separately, termed Bilateral Segmentation Network (BiSeNetV2). The BiSeNetV2 framework is a generic architecture, which can be implemented by most convolutional models. Our instantiations of BiSeNetV2 achieve a good trade-off between segmentation accuracy and inference speed. We hope that this generic architecture BiSeNetV2 will foster further research in semantic segmentation.

Acknowledgment

This work is supported by the National Natural Science Foundation of China (No. 61433007 and 61876210).



(a) Input (b) BiSeNetV2 (c) BiSeNetV2-Large (d) Groundtruth

Fig. 10 Visualization examples on the Cityscapes *val* set produced from BiSeNetV2 and BiSeNetV2-Large. The first row shows that our architecture can focus on the details, e.g., fence. The bus in the third row demonstrates the architecture can capture the large object. The bus in the last row illustrates the architecture can encode the spatial context to reason it.

References

Achanta R, Shaji A, Smith K, Lucchi A, Fua P, Süsstrunk S (2012) Slic superpixels compared to state-of-the-art superpixel methods. IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) 34(11):2274–2282 3

Badrinarayanan V, Kendall A, Cipolla R (2017) SegNet: A deep convolutional encoder-decoder architecture for image segmentation. IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) 39(12):2481–2495 2, 4, 8, 13

Van den Bergh M, Boix X, Roig G, de Capitani B, Van Gool L (2012) Seeds: Superpixels extracted via energy-driven sampling. In: Proc. European Conference on Computer Vision (ECCV), pp 13–26 3

Bilinski P, Prisacariu V (2018) Dense decoder shortcut connections for single-pass semantic segmentation. In: Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp 6596–6605 13

Boykov YY, Jolly MP (2001) Interactive graph cuts for optimal boundary & region segmentation of objects in nd images. In: Proc. IEEE International Conference on Computer Vision (ICCV), vol 1, pp 105–112 3

Brostow GJ, Shotton J, Fauqueur J, Cipolla R (2008a) Segmentation and recognition using structure from motion point clouds. In: Proc. European Conference on Computer Vision (ECCV), pp 44–57 2, 8

Brostow GJ, Shotton J, Fauqueur J, Cipolla R (2008b) Segmentation and recognition using structure from motion point clouds. In: Proc. European Conference on Computer Vision (ECCV) 8

Caesar H, Uijlings J, Ferrari V (2018) Coco-stuff: Thing and stuff classes in context. In: Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR) 2, 8

Chandra S, Couprie C, Kokkinos I (2018) Deep spatio-temporal random fields for efficient video segmentation. In: Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp 8915–8924 13

Chen LC, Papandreou G, Kokkinos I, Murphy K, Yuille AL (2015) Semantic image segmentation with deep convolutional nets and fully connected crfs. In: Proc. International Conference on Learning Representations (ICLR) 2, 3, 13

Chen LC, Papandreou G, Kokkinos I, Murphy K, Yuille AL (2016) Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. arXiv 13

Chen LC, Papandreou G, Schroff F, Adam H (2017) Rethinking atrous convolution for semantic image segmentation. arXiv 2, 3, 6

Chen LC, Zhu Y, Papandreou G, Schroff F, Adam H (2018) Encoder-decoder with atrous separable convolution for semantic image segmentation. In: Proc. European Conference on Computer Vision (ECCV), pp 801–818 2, 3

Chetlur S, Woolley C, Vandermersch P, Cohen J, Tran J, Catanzaro B, Shelhamer E (2014) cudnn: Efficient primitives

for deep learning. arXiv 7
Chollet F (2017) Xception: Deep learning with depthwise separable convolutions. In: Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR) 2, 4, 6
Cordts M, Omran M, Ramos S, Rehfeld T, Enzweiler M, Benenson R, Franke U, Roth S, Schiele B (2016) The cityscapes dataset for semantic urban scene understanding. In: Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR) 1, 2, 8, 9
Fu J, Liu J, Tian H, Fang Z, Lu H (2019) Dual attention network for scene segmentation. In: Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR) 2, 3
Geiger A, Lenz P, Urtasun R (2012) Are we ready for autonomous driving? the kitti vision benchmark suite. In: Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp 3354–3361 1
Ghiasi G, Fowlkes CC (2016) Laplacian pyramid reconstruction and refinement for semantic segmentation. In: Proc. European Conference on Computer Vision (ECCV) 4, 13
Glorot X, Bordes A, Bengio Y (2011) Deep sparse rectifier neural networks. In: Proc. International conference on artificial intelligence and statistics, pp 315–323 6
Hariharan B, Arbeláez P, Girshick R, Malik J (2015) Hypercolumns for object segmentation and fine-grained localization. In: Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp 447–456 4
He K, Zhang X, Ren S, Sun J (2015) Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In: Proc. IEEE International Conference on Computer Vision (ICCV), pp 1026–1034 8
He K, Zhang X, Ren S, Sun J (2016) Deep residual learning for image recognition. In: Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR) 6, 12
Howard A, Sandler M, Chu G, Chen LC, Chen B, Tan M, Wang W, Zhu Y, Pang R, Vasudevan V, et al. (2019) Searching for mobilenetv3. In: Proc. IEEE International Conference on Computer Vision (ICCV) 6, 7, 10, 11
Howard AG, Zhu M, Chen B, Kalenichenko D, Wang W, Weyand T, Andreetto M, Adam H (2017) Mobilenets: Efficient convolutional neural networks for mobile vision applications. arXiv 4, 6, 12
Huang G, Liu Z, van der Maaten L, Weinberger KQ (2017) Densely connected convolutional networks. In: Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp 2261–2269 6
Huang PY, Hsu WT, Chiu CY, Wu TF, Sun M (2018) Efficient uncertainty estimation for semantic segmentation in videos. In: Proc. European Conference on Computer Vision (ECCV), pp 520–535 13
Iandola FN, Moskewicz MW, Ashraf K, Han S, Dally WJ, Keutzer K (2016) Squeezenet: Alexnet-level accuracy with 50x fewer parameters and ¡1mb model size. arXiv abs/1602.07360 4, 6
Ioffe S, Szegedy C (2015) Batch normalization: Accelerating deep network training by reducing internal covariate shift. In: Proc. International Conference on Machine Learning (ICML), pp 448–456 6
Krizhevsky A, Sutskever I, Hinton GE (2012) Imagenet classification with deep convolutional neural networks. In: Proc. Neural Information Processing Systems (NeurIPS) 2
Li G, Yun I, Kim J, Kim J (2019a) Dabnet: Depth-wise asymmetric bottleneck for real-time semantic segmentation. In: Proc. British Machine Vision Conference (BMVC) 13
Li H, Xiong P, Fan H, Sun J (2019b) Dfanet: Deep feature aggregation for real-time semantic segmentation. In: Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR) 4, 13
Li X, Liu Z, Luo P, Loy CC, Tang X (2017) Not all pixels are equal: difficulty-aware semantic segmentation via deep layer cascade. In: Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR) 4
Lin G, Milan A, Shen C, Reid I (2017) Refinenet: Multi-path refinement networks with identity mappings for high-resolution semantic segmentation. In: Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR) 2, 4, 13
Lin TY, Maire M, Belongie S, Hays J, Perona P, Ramanan D, Dollár P, Zitnick CL (2014) Microsoft coco: Common objects in context. In: Proc. European Conference on Computer Vision (ECCV) 8
Liu W, Rabinovich A, Berg AC (2016) Parsenet: Looking wider to see better. arXiv 5, 6
Long J, Shelhamer E, Darrell T (2015) Fully convolutional networks for semantic segmentation. In: Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR) 2, 3, 13
Ma N, Zhang X, Zheng HT, Sun J (2018) Shufflenet v2: Practical guidelines for efficient cnn architecture design. In: Proc. European Conference on Computer Vision (ECCV), pp 116–131 2, 4, 6, 7
Mazzini D (2018) Guided upsampling network for real-time semantic segmentation. In: Proc. British Machine Vision Conference (BMVC) 2, 4, 13
Mehta S, Rastegari M, Caspi A, Shapiro L, Hajishirzi H (2018) Espnet: Efficient spatial pyramid of dilated convolutions for semantic segmentation. In: Proc. European Conference on Computer Vision (ECCV), pp 552–568 4, 13
Mehta S, Rastegari M, Shapiro LG, Hajishirzi H (2019) Espnetv2: A light-weight, power efficient, and general purpose convolutional neural network. In: Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR) 4, 13
Orsic M, Kreso I, Bevandic P, Segvic S (2019) In defense of pre-trained imagenet architectures for real-time semantic segmentation of road-driving images. In: Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp 12607–12616 13
Otsu N (1979) A threshold selection method from gray-level histograms. IEEE transactions on systems, man, and cybernetics 9(1):62–66 3
Paszke A, Chaurasia A, Kim S, Culurciello E (2016) Enet: A deep neural network architecture for real-time semantic segmentation. arXiv 2, 4, 13
Peng C, Zhang X, Yu G, Luo G, Sun J (2017) Large kernel matters–improve semantic segmentation by global convolutional network. In: Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR) 2, 4
Pohlen T, Hermans A, Mathias M, Leibe B (2017) Full-resolution residual networks for semantic segmentation in street scenes. In: Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp 4151–4160 13
Poudel RP, Liwicki S, Cipolla R (2019) Fast-scnn: fast semantic segmentation network. arXiv 13
Ren X, Malik J (2003) Learning a classification model for segmentation. In: Proc. IEEE International Conference on Computer Vision (ICCV), p 10 3
Romera E, Alvarez JM, Bergasa LM, Arroyo R (2018) Erfnet: Efficient residual factorized convnet for real-time semantic segmentation. IEEE Transactions on Intelligent Transportation Systems 19(1):263–272 2, 4, 13
Ronneberger O, Fischer P, Brox T (2015) U-net: Convolutional networks for biomedical image segmentation. In: Proc. International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI) 4

Rother C, Kolmogorov V, Blake A (2004) Grabcut: Interactive foreground extraction using iterated graph cuts. In: ACM Transactions on Graphics, vol 23, pp 309–314 3

Sandler M, Howard A, Zhu M, Zhmoginov A, Chen LC (2018) Mobilenetv2: Inverted residuals and linear bottlenecks. In: Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp 4510–4520 4, 6, 7, 10, 11

Selvaraju RR, Cogswell M, Das A, Vedantam R, Parikh D, Batra D (2017) Grad-cam: Visual explanations from deep networks via gradient-based localization. In: Proc. IEEE International Conference on Computer Vision (ICCV), pp 618–626 10

Shelhamer E, Long J, Darrell T (2017) Fully convolutional networks for semantic segmentation. IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) (4):640–651 3

Shen Z, Liu Z, Li J, Jiang YG, Chen Y, Xue X (2017) Dsod: Learning deeply supervised object detectors from scratch. In: Proc. IEEE International Conference on Computer Vision (ICCV) 6

Simonyan K, Zisserman A (2015) Very deep convolutional networks for large-scale image recognition. In: Proc. International Conference on Learning Representations (ICLR) 6

Sturgess P, Alahari K, Ladicky L, Torr PHS (2009) Combining Appearance and Structure from Motion Features for Road Scene Understanding. In: Proc. British Machine Vision Conference (BMVC) 8

Szegedy C, Ioffe S, Vanhoucke V, Alemi AA (2017) Inception-v4, inception-resnet and the impact of residual connections on learning. In: Proc. Thirty-first AAAI conference on artificial intelligence 6

Tan M, Chen B, Pang R, Vasudevan V, Sandler M, Howard A, Le QV (2019) Mnasnet: Platform-aware neural architecture search for mobile. In: Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp 2820–2828 7

Treml M, Arjona-Medina J, Unterthiner T, Durgesh R, Friedmann F, Schuberth P, Mayr A, Heusel M, Hofmarcher M, Widrich M, et al. (2016) Speeding up semantic segmentation for autonomous driving. In: Proc. Neural Information Processing Systems Workshops 13

Vincent L, Soille P (1991) Watersheds in digital spaces: an efficient algorithm based on immersion simulations. IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) (6):583–598 3

Wang J, Sun K, Cheng T, Jiang B, Deng C, Zhao Y, Liu D, Mu Y, Tan M, Wang X, Liu W, Xiao B (2019) Deep high-resolution representation learning for visual recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) 4

Wang P, Chen P, Yuan Y, Liu D, Huang Z, Hou X, Cottrell G (2018a) Understanding convolution for semantic segmentation. In: Proc. IEEE Winter Conference on Applications of Computer Vision (WACV) 3, 13

Wang RJ, Li X, Ling CX (2018b) Pelee: A real-time object detection system on mobile devices. In: Proc. Neural Information Processing Systems (NeurIPS) 6

Wu Z, Shen C, Hengel Avd (2016) High-performance semantic segmentation using very deep fully convolutional networks. arXiv 11

Yu C, Wang J, Peng C, Gao C, Yu G, Sang N (2018a) Bisenet: Bilateral segmentation network for real-time semantic segmentation. In: Proc. European Conference on Computer Vision (ECCV), pp 325–341 3, 13

Yu C, Wang J, Peng C, Gao C, Yu G, Sang N (2018b) Learning a discriminative feature network for semantic segmentation. In: Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR) 2, 4, 6

Yu C, Wang J, Gao C, Yu G, Shen C, Sang N (2020) Context prior for scene segmentation. In: Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR) 2, 3

Yu F, Koltun V (2016) Multi-scale context aggregation by dilated convolutions. In: Proc. International Conference on Learning Representations (ICLR) 13

Yuan Y, Wang J (2018) Ocnet: Object context network for scene parsing. arXiv 3

Zhang H, Dana K, Shi J, Zhang Z, Wang X, Tyagi A, Agrawal A (2018a) Context encoding for semantic segmentation. In: Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp 7151–7160 2, 3

Zhang X, Zhou X, Lin M, Sun J (2018b) Shufflenet: An extremely efficient convolutional neural network for mobile devices. In: Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp 6848–6856 4, 6

Zhao H, Shi J, Qi X, Wang X, Jia J (2017) Pyramid scene parsing network. In: Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR) 2, 3, 6, 13

Zhao H, Qi X, Shen X, Shi J, Jia J (2018a) Icnet for real-time semantic segmentation on high-resolution images. In: Proc. European Conference on Computer Vision (ECCV), pp 405–420 2, 4, 13

Zhao H, Zhang Y, Liu S, Shi J, Loy CC, Lin D, Jia J (2018b) PSANet: Point-wise spatial attention network for scene parsing. In: Proc. European Conference on Computer Vision (ECCV) 2, 3

Zheng S, Jayasumana S, Romera-Paredes B, Vineet V, Su Z, Du D, Huang C, Torr PH (2015) Conditional random fields as recurrent neural networks. In: Proc. IEEE International Conference on Computer Vision (ICCV) 13

Zhou B, Zhao H, Puig X, Xiao T, Fidler S, Barriuso A, Torralba A (2019) Semantic understanding of scenes through the ade20k dataset. International Journal of Computer Vision (IJCV) 127(3):302–321 1