IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received December 16, 2019, accepted March 6, 2020, date of publication March 18, 2020, date of current version March 30, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2981842

Contextual Attention Refinement Network for Real-Time Semantic Segmentation

SHIJIE HAO[1,2], YUAN ZHOU[1,2], YOUMING ZHANG[3], AND YANRONG GUO[1,2]

[1]Key Laboratory of Knowledge Engineering with Big Data, Ministry of Education, Hefei University of Technology, Hefei 230009, China
[2]School of Computer Science and Information Engineering, Hefei University of Technology, Hefei 230009, China
[3]School of Mathematics and Statistics, Northeastern University, Qinhuangdao 066004, China

Corresponding author: Yanrong Guo (yrguo@hfut.edu.cn)

This work was supported in part by the National Key Research and Development Program under Grant 2018YFB0804203, and in part by the National Nature Science Foundation of China under Grant 61772171, Grant 61702156, Grant 61632007, and Grant 61876056.

ABSTRACT Recently, significant progress has been made in pixel-level semantic segmentation using deep neural networks. However, for the current semantic segmentation methods, it is still challenging to achieve the balance between segmentation accuracy and computational cost. To address this issue, we propose the Contextual Attention Refinement Network (CARNet). In this method, we construct the Contextual Attention Refinement Module (CARModule), which learns an attention vector to guide the fusion of low-level and high-level features for obtaining higher segmentation accuracy. The CARModule is lightweight and can be directly equipped with different types of network structures. To better optimize the network, we additionally consider the semantic information, and further introduce the Semantic Context Loss (SCLoss) into the overall loss function. In the experiments, we validate the effectiveness and efficiency of our method on several public datasets for semantic segmentation. The results show that our method achieves a good balance on accuracy and computational costs.

INDEX TERMS Real-time semantic segmentation, contextual attention refinement module, semantic context loss.

I. INTRODUCTION

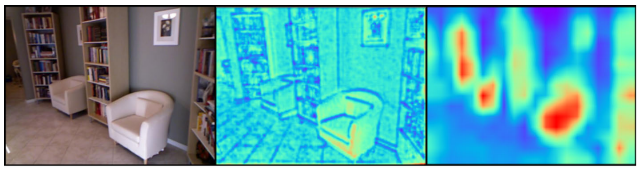
Semantic segmentation is an important task in computer vision because it plays a vital role in many real-world applications, e.g., autonomous driving [1], [2], media generation [3], [4], medical image analysis [5], [6] and multimedia content analysis [7], [8]. Given an image, a semantic segmentation algorithm is expected to accurately partition the target object(s) from the background region. In other words, the algorithm should recognize, locate, and delineate the target object(s) simultaneously. Therefore, compared with other visual information processing tasks, semantic segmentation has its unique requirements.

The emergence of Fully Convolutional Networks (FCN) [9] paves way to the rapid development of semantic segmentation by using deep learning models. FCN achieves pixel-level inference by converting the last fully connected layer of traditional architectures, such as VGG [10] and AlexNet [11], into a fully convolutional layer. A large number of methods choose the architecture of FCN as their baseline such as [12]–[14].

The associate editor coordinating the review of this manuscript and approving it for publication was Eduardo Rosa-Molinar.

Despite the achieved success, the current semantic segmentation methods can hardly satisfy the growing demands of real-world applications. A main challenge is the balance between segmentation accuracy and speed. On the one hand, in the convolutional structure, the operations of pooling and striding inevitably lose the spatial information and thus produce low-resolution network outputs. Various strategies can be used to generate more accurate segmentation results, e.g., designing a dilated convolution [15], combining features at different levels [16], and employing a multi-scale context aggregation strategy [17]. On the other hand, these improvements possibly lower the implementation efficiency. However, achieving a balance between accuracy and speed is critical for the semantic segmentation task in many real-time applications, such as autonomous driving and computer-assisted surgery.

In a typical semantic segmentation model, low-level and high-level features can be both learned along the pipeline. Taking Fig. 1 for example, the feature map extracted from the shallow layers contains more low-level details, such as salient boundaries. In the deep layers, the extracted feature map becomes more aware of semantics, such as the grouped regions with respect to categories and attributes.



(a) Original image (b) shallow-layer feature map (c) deep-layer feature map

FIGURE 1. **Feature maps learned from different layers of our CARNet.**

The above two types of information can be complementary for accurate segmentation. However, their combination is not straightforward. Moreover, although low-level features contain much useful information, their importance to the final inference is not equal. In this context, we propose the CARNet (Fig. 2) that fully leverages these two types of information. In the CARNet, the key component is the CARModule shown in Fig. 3, which fuses features learned from (relatively) shallow-layers and (relatively) deep-layers. We note that the proposed CARModule is lightweight, as it only contains approximately 0.225M parameters and 4.3M FLOPs in a typical application. Our segmentation method equipped with CARModule is also lightweight and achieves good segmentation accuracy in real time. In the experiments, we compare our method with several state-of-the-art methods on three public datasets. The results empirically show that our method achieves a better balance between segmentation accuracy and computational costs. Ablation studies also validate the effectiveness of the key elements in our method.

Our contribution includes the following aspects:

1) We propose the CARModule that effectively improves the semantic segmentation accuracy. This module is able to fuse and enhance the features from neighboring stages of the pipeline under the guidance of the learned attention. Furthermore, the CARModule has low computational costs and is capable of being equipped with different architectures, such as ResNet [18] and MobileNet [19].
2) We introduce a novel Semantic Context Loss (SCLoss) into the overall loss function. In this way, the extended loss function jointly considers pixel-level cross entropy and global-level semantic information, thus better optimizing the whole segmentation model.

Based on the above components, our CARNet achieves competitive performance in terms of accuracy and computational cost on several public datasets.

The remainder of this paper is organized as follows. Section II briefly introduces the related works. Section III describes our proposed method. In Section IV, we demonstrate and analyze our experimental results. Section V concludes the paper and discusses possible future directions.

II. RELATED WORK

In this section, we briefly introduce related research on deep-learning-based semantic segmentation. First, we introduce methods aimed at enhancing segmentation accuracy. Then, we introduce the real-time segmentation methods. Finally, we present the differences between our method and several closely related methods.

A. METHODS FOR ACCURACY

Accuracy is the prime concern for the semantic segmentation task. There are extensive works that aim to enhance segmentation accuracy. Ronneberger *et al.* [14] propose U-Net, which exploits skip connections between the encoder block and the decoder block. This research provides a useful strategy for semantic segmentation, which boosts segmentation accuracy by appropriately combining shallow-layer features (low-level) and deep-layer features (high-level). To pursue higher accuracy, RefineNet [20] also adopts this strategy. It uses a deep residual network [18] as its backbone and adopts a more complicated skip-connection component. Apart from methods using skip connections, [21]–[23] utilize the saved pooling indices and deconvolution to recover the lost details in pooling. However, the information brought by pooling indices is still limited. To reduce the usage of pooling and stride convolution, dilated convolution [15] is a useful technique that enlarges the receptive field without using any downsampling operation. Reference [15] also aggregates a multi-scale context by setting different dilated rates, which validates that leveraging context information is beneficial for gaining higher accuracy. In contrast to the enlarging of receptive field only, [16] utilizes global pooling for extracting the global context, and enhancing the middle-level features. Reference [16] validates that the construction of the global context is important, while the conventional convolution neural network cannot provide a sufficient global context. To fully leverage the multi-scale context, [17] proposes Pyramid Pooling Module (PPM) and [12] introduces Atrous Pyramid Pooling Module (ASPP) as a variant of PPM. Recently, Li *et al.* [24] concentrate on the attention mechanism in semantic segmentation based on the expectation-maximization algorithm. A common limitation of accuracy-oriented methods is their expensive computational costs, which brings challenges for the applications with real-time requirements.

B. METHODS FOR SPEED

Real-time semantic segmentation techniques emphasize the balance of speed and accuracy. In other words, they try to reduce parameters and FLOPs of the whole model on the premise that the accuracy loss should be as small as possible. For example, SegNet [22] exploits a small architecture to reduce network parameters and achieves 57% MIoU at 14.7 FPS on 640 $\times$ 360 images of the Cityscapes test set. ENet [13] builds a tight framework to improve efficiency and achieves 58.3% MIoU at 76.9 FPS on 512 $\times$ 1024 images of the Cityscapes test set. ICNet proposed by Zhao *et al.* [25] exploits the strategy of multi-scale inputs and cascaded frameworks. It achieves 67.1% MIoU on 720$\times$960 images for the CamVid [26] test set. BiSeNet [27] constructs a separable spatial path and semantic path to reduce computational costs and achieves 68.4% MIoU at 72.3 FPS on 768 $\times$ 1536 images

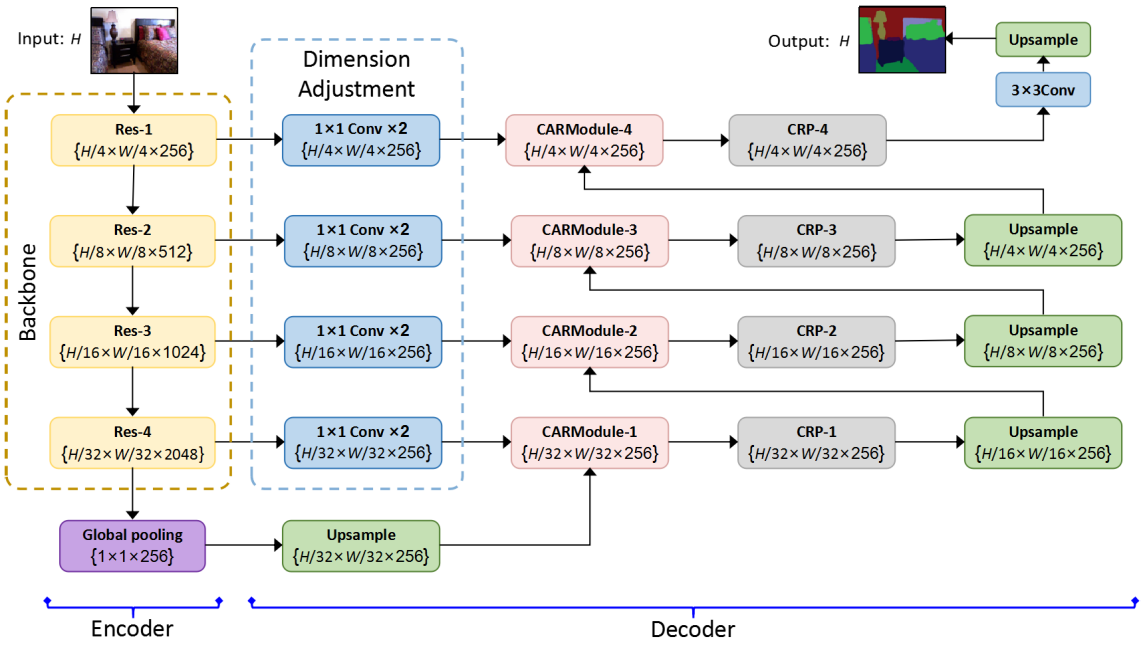


FIGURE 2. The architecture of the Contextual Attention Refinement Network (CARNet). In the figure, "1 × 1Conv ×2" denotes two cascaded 1 × 1 convolutions.

of the Cityscapes test set. In this model, the semantic path aims to extract semantic features, and the spatial path is to maintain spatial details. Recently, Li *et al.* [28] use the strategy of multi-scale feature propagation to construct the real-time segmentation framework, which achieves a good balance between accuracy and speed.

C. DIFFERENCE BETWEEN OUR METHOD AND THE RELATED WORKS

Inspired by the insightful thoughts of the above-mentioned works, we build a network called CARNet for real-time semantic segmentation. The differences between our method and several related ones are described in the following:

1) Compared with [14] and [20], our method does not directly utilize shallow-layer features to refine deep-layer features. Instead, we learn an attention vector to weigh more on the important features and less on the unimportant ones. Although both the RefineNet [20] and our CARNet build an encoder-decoder structure, they still have differences in three aspects. First, the CARNet additionally considers the global features. Second, we use multiple CARMoudules along the different stages of the pipeline. Third, we introduce the Semantic Context Loss (SCLoss) to better optimize the model.
2) Compared with the methods using context information, such as [12], [16], [17], our CARNet fuses and enhances the learned feature maps at all the different resolutions in the pipeline. Furthermore, we build our loss function by simultaneously considering local prediction errors and global semantic context differences.

III. PROPOSED METHOD

In this section, we first present the structure of our CARNet. Then, we introduce the proposed CARModule and the extended loss function.

A. THE ARCHITECTURE OF CARNet

As shown in Fig. 2, our CARNet is an encoder-decoder structure. In the encoder part, the choice of the backbone network structure is open. We choose the ResNet structure [18] to exemplify the encoder part of our CARNet. In the backbone network, the features are gradually condensed into low resolutions. According to different feature resolutions, we divide the backbone network into four parts (i.e., Res-1, Res-2, Res-3, and Res-4), which have the 1/4, 1/8, 1/16 and 1/32 resolutions of the input image. Particularly, we use global pooling to extract an additional global feature ($1 \times 1 \times 256$) from the output of the backbone network.

The aim of the decoder part is to learn more accurate feature maps for pixel inference while keeping the computational costs of this part at a small scale. As shown in Fig. 2, the decoder part has four CARModules, which have connections with the four components at the corresponding resolutions in the encoder part. These CARModules aim to gradually enhance low-resolution feature maps by fusing the feature from different scales. For example, CARModule-1 fuses the feature from the Res-4 layer and the extracted global feature. Moreover, we adopt the Chained Residual Pooling (CRP) module [20] to post-process the fused features. For the details of CRP, we refer readers to [20]. From CARModule-1 to CARModule-4, the feature resolutions are gradually recovered from 1/32 to 1/4. Then, the learned

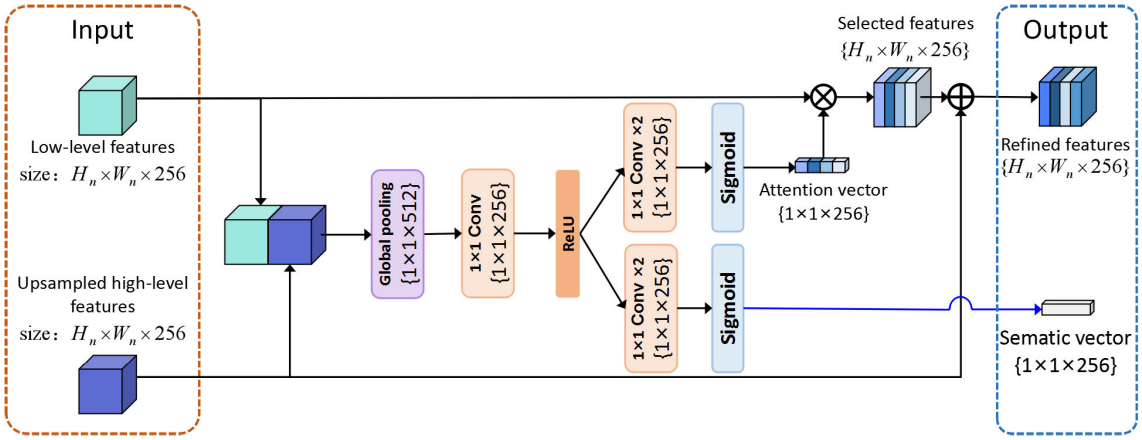


FIGURE 3. The architecture of the Contextual Attention Refinement Module (CARModule). H_n and W_n denote the height and width of input features in the n^{th} CARModule. C denotes the number of categories.

feature map is sent into the 3 × 3 convolutional classifier to produce the final prediction.

Remarks: The CARNet achieves a balance between accuracy and speed. As for the speed, in the encoder, we first gradually downsample features into low-resolution ones. This strategy is different from the methods maintaining high-resolution feature maps in the pipeline (e.g., deeplab [12]). Second, the decoder is computationally modest because we adopt 1 × 1 convolutions to construct CARModule. The motivations lie in two aspects. On the one hand, convolutional operators in the fusion block aim to 1) transform the features from different stages into a common space or 2) make dimension adjustment. Therefore, utilizing convolutions with large kernels is not necessary. On the other hand, using 1 × 1 convolutions can further decrease the model's computational costs. Therefore, the CARModule only contains approximately 0.225 M parameters and the decoder component totally includes 5.5 M parameters.

As for the accuracy, first, appropriate cooperation between shallow-layer and deep-layer features enhances the feature representations. Second, the extended loss function by additionally considering global semantics can better regularize the network to perceive context information, and further preserves the model's accuracy. As stated above, the feature fusion and the extended loss function can be implemented with lightweight computational costs.

B. CONTEXTUAL ATTENTION REFINEMENT MODULE

1) MODULE STRUCTURE

The proposed CARModule structure is shown in Fig. 3. Specifically, the inputs of the CARModule are from two neighboring components in the encoder stage. One is from the current scale of features, and the other is the up-sampled features from its succedent components. It is known that the features learned from deeper layers contain relatively more semantic information. Thus, we call the two inputs low-level features and high-level features. After concatenating them together, we send the combined features into a pipeline containing global pooling, 1 × 1 convolution (dimension changing) and ReLU (nonlinear mapping). This output is sent into two sub-paths. The first is to generate an attention vector, and the other is to generate a semantic vector. The attention vector is used as the fusion weights between the low-level features and the high-level features. The semantic vector is used to build the Semantic Context Loss (SCLoss) for regularizing the training process.

2) FEATURE FUSION

For each CARModule, the feature fusion is realized via a weighted combination of the two fusion sources (Eq. 1).

$$\mathbf{F} = \boldsymbol{\sigma}_a \otimes \mathbf{F}_1 + \mathbf{F}_2 \tag{1}$$

Here, $\mathbf{F}_1$ and $\mathbf{F}_2$ are the low-level features and the high-level features, respectively. $\otimes$ is the channel-wise multiplier. $\boldsymbol{\sigma}_a$ is the learned attention vector, which is obtained from the following process:

$$\boldsymbol{\sigma}_a = Sigmoid(\mathbf{W}_2 \cdot (\mathbf{W}_1 \cdot ReLU(\mathbf{W} \cdot G(\mathbf{F}_1 \oplus \mathbf{F}_2)))) \tag{2}$$

Here, $\mathbf{W}$, $\mathbf{W}_1$ and $\mathbf{W}_2$ both denote 1 × 1 convolutions. $\oplus$ denotes concatenation operation, and $G(\cdot)$ denotes global pooling.

Remarks: First, the learned attention vector $\boldsymbol{\sigma}_a$ is used to emphasize important details of low-level features. Therefore, in each component of the CARNet decoder part, the fused features $\mathbf{F}$ can be strengthened by carefully combining various features. In other words, the low-level (or shallow-layer) features are effectively selected under the guidance of a high-level context before fusion. Second, we note that the CARModule can be efficiently implemented, as all its components are computationally inexpensive. Third, the CARModule has a good portability because it only requires features extracted from different network layers. For example, we can conveniently equip the CARModule into the MobileNet, of which the process is slightly different from "ResNet+CARModule".

C. LOSS FUNCTION

By re-examining the CARNet structure, the cooperation between the encoder part and the decoder part exists in all different resolutions. To better control the training process, a loss function covering the whole CARNet should be designed.

To this end, we first present our loss function in Eq. 3.

$$\ell_{total} = \ell_p + \lambda \sum_{n=1}^{N} \ell_s^n \tag{3}$$

In the function, ℓ_p is the traditional cross-entropy loss extensively adopted in semantic segmentation tasks. ℓ_s^n is the proposed Semantic Context Loss (SCLoss) function applied to the n^{th} CARModule. Here N denotes the total number of adopted CARModules. λ is a hyperparameter weighing between the traditional loss and the SCLoss.

Next, we introduce the detailed construction of these two types of loss. For ℓ_p, we follow the conventional definition:

$$\ell_p = -\sum_{k=1}^{H}\sum_{j=1}^{W}\sum_{i=1}^{C} y_{(i,j,k)} log(x_{(i,j,k)}) \tag{4}$$

Here, C denotes the total number of categories. H and W denote the height and width of one-hot encoded ground truth. By using the bilinear interpolation operator, the network output is kept in the same size as the ground truth map ($H \times W \times C$). $x_{i,j,k}$ or $y_{i,j,k}$ denotes the value of the element from the prediction or ground truth map located in (i, j, k).

Our SCLoss is defined as follows:

$$\ell_s = -\sum_{i=1}^{C} v_{(i)} \log(\sigma_{s(i)}) \tag{5}$$

In SCLoss, $\mathbf{V} = \{v_{(1)}, \ldots, v_{(i)}, \ldots, v_{(C)}\}$ denotes the ground-truth semantic one-hot vector ($1 \times 1 \times C$), revealing the categories included in the image. $\boldsymbol{\sigma}_s = \{\sigma_{s(1)}, \ldots, \sigma_{s(i)}, \ldots, \sigma_{s(C)}\}$ is the learned semantic vector ($1 \times 1 \times C$):

$$\boldsymbol{\sigma}_s = Sigmoid(\mathbf{W}_4 \cdot (\mathbf{W}_3 \cdot ReLU(\mathbf{W} \cdot G(\mathbf{F}_1 \oplus \mathbf{F}_2)))) \tag{6}$$

Here, $\mathbf{W}_3$ and $\mathbf{W}_4$ denote 1×1 convolutions.

Remarks: We build the loss function in Eq. 3 based on the following considerations. First, the SCLoss ℓ_s involves in the connections between the encoder and the decoder at different resolutions, which fully leverages the proposed CARNet. Second, the traditional loss ℓ_p and the SCLoss ℓ_s are complimentary to each other. On the one hand, ℓ_p measures local pixel-wise training errors. On the other hand, with $\boldsymbol{\sigma}_s$ as an overall semantic descriptor on input images, ℓ_s measures a more global loss. By leveraging these two types of loss, the training process is expected to be better regularized. The ablation study in the next section empirically validates our extension on the loss function.

IV. EXPERIMENTS

We conduct extensive experiments to evaluate our method. The experimental data include three public datasets: Cityscapes [34], NYUDv2 [35], [36], and ADE20K [37]. In this section, we first compare our method with several off-the-shelf semantic segmentation methods in the aspects of accuracy and computational costs. Then, we conduct ablation study to validate the contributions of our method. In these experiments, we adopt deep residual network (ResNet) [18] as the backbone. To demonstrate the flexibility of our CARModule, we further conduct experiments on the MobileNet [19] and provide corresponding results.

A. IMPLEMENTATION DETAILS

The implementation details are described in the following. The codes were implemented under the Pytorch platform.[1] The experiments on Cityscapes were conducted on the NVIDIA TITAN X GPU card. The experiments on NYUDv2 and ADE20K were conducted on the GTX 1080 Ti GPU card. For the three datasets, we adopted slightly different training strategies. For the NYUDv2 dataset (relatively small data size), we set the initial learning rate $5e^{-4}$ for the encoder part and $5e^{-3}$ for the decoder part. We halve the learning rate for every 100 epochs until a total of 300 epochs, or the accuracy stops improving. For the Cityscapes and the ADE20K dataset (relatively large data size), we set the initial learning rate $1e^{-3}$ for the encoder part and $1e^{-2}$ for the decoder part. We reduce the learning rate by one-fifth for every 100 epochs until a total of 300 epochs, or the accuracy on validation set stops improving. The commonly used data augmentation strategies were adopted. For example, we randomly flip and scale the images from 0.5 to 2. We set 0.9 for momentum and $1e^{-5}$ for weight decay. In particular, we set different crop sizes for the three datasets, i.e., 500×500 for the NYUDv2 dataset, 580×580 for the ADE20K dataset, and 800×800 for the Cityscapes dataset. Due to the limited GPU memory, we set the mini-batch size as 6 during the entire training process. In our method, without special explanations, we set the hyperparameter $\lambda = 0.2$ and adopt four CARModules for our CARNet, in which each CARModule is equipped with SCLoss. In the part E of this section, we validate our parameter setting.

As for the quantitative evaluation, we use Mean Intersection over Union (MIoU), described in Eq.7, as the metric for measuring accuracy. In Eq.7, p_{ii}, p_{ij}, and p_{ji} denote the number of true positives, false positives and false neg- atives, respectively. $k + 1$ denotes the category number. We adopt four metrics for evaluating computational costs, including floating-point operations (FLOPs), parameter number (Params), implementation time (Time) and frames per second (FPS).

$$MIoU = \frac{1}{k+1} \sum_{i=0}^{k} \frac{p_{ii}}{\sum_{j=0}^{k} p_{ij} + \sum_{j=0}^{k} p_{ji} - p_{ii}} \tag{7}$$

B. CITYSCAPES

The dataset of Cityscapes is widely used in semantic segmentation, especially for evaluating real-time methods. It contains

[1] https://pytorch.org

a large number of street scenes collected from 50 different European cities. This dataset provides 5000 fine-annotated images and 20000 coarse-annotated images. Our method is trained with fine-annotated images in all experiments. There are 2975 images for training, 500 images for validation, and 1525 images for testing. The ground truth annotations of the test set are not released yet. We obtained the accuracy on this dataset by submitting our results to the official evaluation server.[2]

We compare our method with several state-of-the-art methods in terms of accuracy and efficiency. Of note, our method has four versions, i.e., CARNet-ResNet50, CARNet-ResNet101, CARNet-ResNet152, and CARNet-Mobile. The first three ones use ResNet with different layers as the backbone, and the last one uses MobileNet as its backbone. We have the following observations from the results listed in Table. 1. Compared to RefineNet-ResNet101 [20], CARNet-ResNet101 has better performance in terms of MIoU, FLOPs, and Params. With only one-eighth FLOPs and one-fourth Params, CARNet-ResNet50 achieves slightly worse accuracy (0.4%) than RefineNet-ResNet101. Even for CARNet-ResNet152, which is our most computationally expensive versions, it still acquires less computational costs but higher accuracy than [20]. Compared to the PSPNet [17] trained with extra coarse-annotated images, our CARNet-ResNet152 has a much lower accuracy (approximately 6% lower) but much higher efficiency (more than 37 times higher in FPS). Compared to the ICNet [25] trained with coarse-annotated images, our CARNet-Mobile version achieves comparable accuracy but with much less computational costs (15% Params and 60% FLOPs of the ICNet). Compared to the two versions of BiSeNet [27] (BiSeNet1 and BiSeNet2), our method also achieves comparable or better results. For example, CARNet-Mobile has better performance on both accuracy and speed than BiSeNet1. The performance of CARNet-ResNet101 is roughly comparable to that of BiSeNet2. We also provide a visualized parameter comparison in Fig. 4. From the figure, we can see that our four versions (the four red dots) form a Pareto-like boundary among all the methods. The enveloping line of the four versions presents a clear trend of balance between accuracy and speed. Finally, in Fig. 5, we demonstrate some segmentation results of our method based on the Cityscapes validated set. By comparing the four versions of our method, we can observe that CARNet-ResNet152 achieves the best performance on accuracy, which is consistent with the trend in Table. 1.

C. NYUDv2

NYUDv2 is a standard semantic segmentation dataset containing a total of 1449 indoor RGBD images with 40 different categories. Among them, 795 images are for training, and 654 images are for testing. Of note, we do not use any depth information in NYUDv2.

[2]https://www.cityscapes-dataset.com/submit

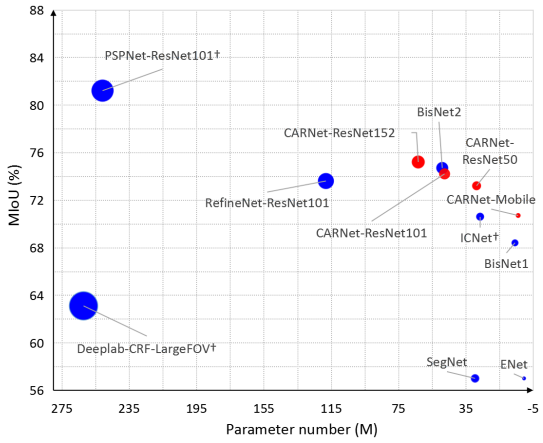


FIGURE 4. Visualized comparison between our method and other state-of-the-art methods. Red marks denote different versions of our method. Blue dots denote the methods for comparison. '†' indicates that the method uses coarse annotations of the Cityscapes dataset during the training process.

First, we present some segmentation results based on the four versions of our method in Fig. 6. Similarly, we can see that our method achieves a more accurate segmentation result when a larger backbone is adopted. Second, we compare our method with two closely related works [20] and [38] in terms of both accuracy and computational cost. Table. 2 reports the experimental results. Compared with the LWRF method [38], our CARNet has consistently better accuracies under different baseline structures (ResNet50, ResNet101, ResNet152) The FLOPs of our three CARNet-ResNet versions are comparable to their corresponding LWRF-ResNet counterparts. Larger superiority on both accuracy and computational costs can be seen in the comparison between RefineNet [20] and CARNet. We also test the CARNet-Mobile version on this dataset. Compared with LWRF-ResNet50, CARNet-Mobile has only 29% FLOPs of LWRF-ResNet50 with 1.9 % accuracy loss.

D. ADE20K

ADE20K is a large-scale scene parsing dataset containing more than 20 K scene images with 150 object categories. There are 20100 images for training and 2000 images for validating (the test set is temporarily unavailable). This dataset is considered challenging for the semantic segmentation task.

The results for comparison are listed in Table. 3. We can see that the two versions of our method (CARNet-ResNet101 and CARNet-ResNet152) do not achieve the best accuracy. Nevertheless, the accuracy gap between the best two methods (PSPNet-ResNet101 and PSPNet-ResNet152) [17] and ours is not large (approximately 1% for the same baseline structure), while our computational costs are much less than PSPNet based methods. These results again validate that our method achieves a good balance between accuracy and speed. In addition, we also test the performance by using the CARNet-Mobile on the ADE20K dataset.and the accuracy

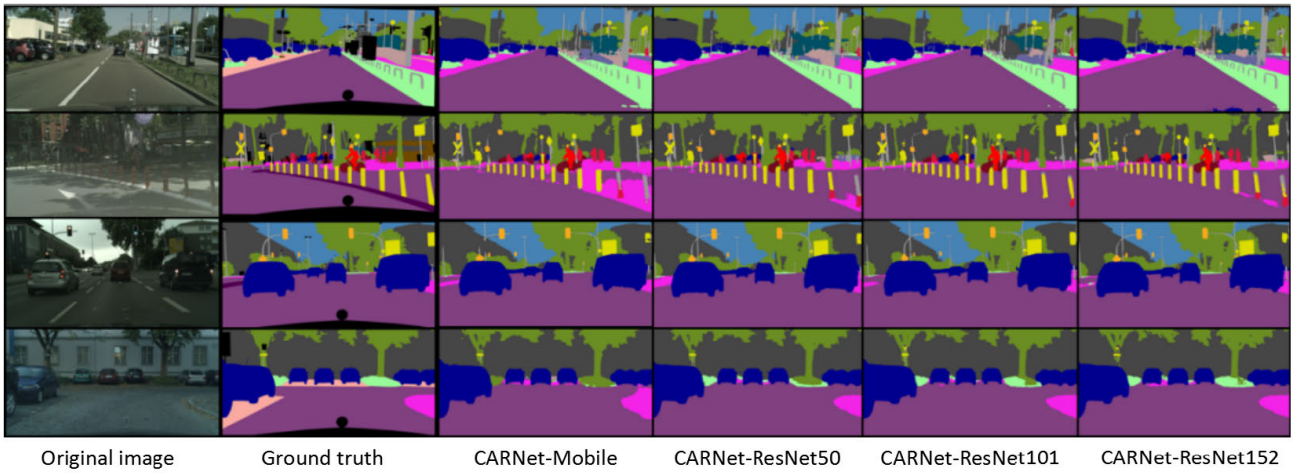


FIGURE 5. Some examples of our method's segmentation results and the ground-truth annotations on the Cityscapes validation set.

TABLE 1. The results on Cityscapes test set. "†" indicates the methods using extra coarse-annotated images of Cityscapes dataset. "msc" indicates that the method adopts the multi-scale strategy. Time (ms) and Frame (fps) results are from a single NVIDIA TITAN X GPU card. "–" means the result is unavailable.

Method	Input size	FLOPs (G)	Params (M)	Time (ms)	Frame (fps)	MIoU (%)
SegNet [29]	640 × 360	286	29.5	60	16.7	57
ENet [13]	640 × 360	**3.8**	**0.4**	**7**	**135.4**	57
SQ [30]	1024 × 2048	270	–	60	16.7	59.8
CRF-RNN [31]	512 × 1024	–	–	700	1.4	62.5
DeepLab-CRF-LargeFOV† [12]	512 × 1024	457.8	262.1	4000	0.25	63.1
BiSeNet1 [27]	768 × 1536	14.8	5.8	13	72.3	68.4
ICNet† [25]	1024 × 2048	28.3	26.5	33	30.3	70.6
FRRN [32]	512 × 1024	235	–	469	2.13	71.8
TwoColumn [33]	512 × 1024	57.2	–	68	14.7	72.9
RefineNet-ResNet101 [20]	512 × 1024	541.4	118.1	–	–	73.6 (msc)
BiSeNet2 [27]	768 × 1536	55.3	49	21	45.7	74.7
PSPNet-ResNet101† [17]	713 × 713	412.2	250.8	1288	0.78	**81.2**
CARNet-Mobile	512 × 1024	17.6	3.9	12.6	79.4	70.7
CARNet-ResNet50	512 × 1024	64.8	28.6	15	66.7	73.2
CARNet-ResNet101	512 × 1024	103.8	47.6	25	40	74.2
CARNet-ResNet152	512 × 1024	142.9	63.3	34	29.4	75.2

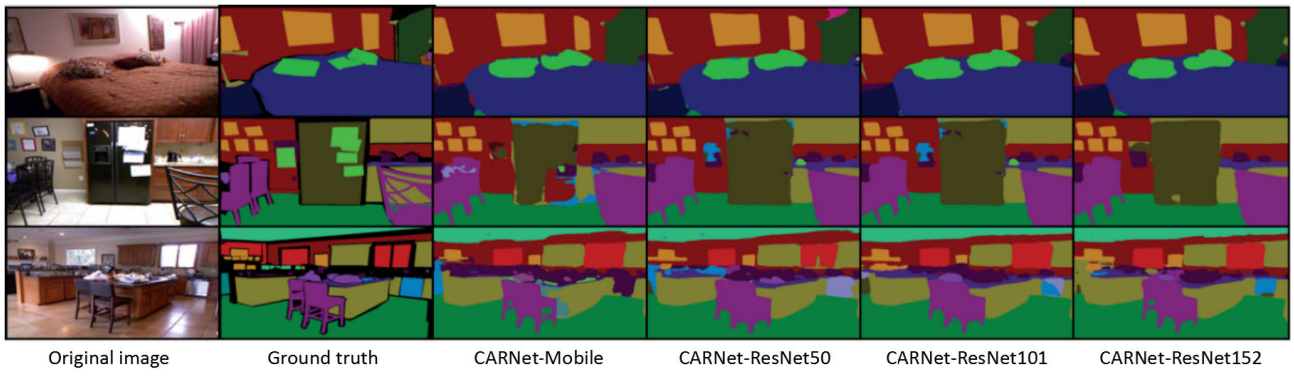


FIGURE 6. Some examples of our method's segmentation results and the ground-truth annotations of the NYUDv2 test set.

is only comparable to Cascaded-DilatedNet [37]. However, it has a very small number of parameters (4.2M), which is much lower than all the other competitors.

E. ABLATION STUDY

In this section, we conduct ablation studies to further validate our method. We organize this subsection into four parts:

TABLE 2. The results on the NYUDv2 test set. Time (*mean* ± *std*) results are from a single NVIDIA GTX 1080 Ti GPU card at a 625 × 468 input image. "msc" indicates that the method adopts the multi-scale strategy.

Method	MIoU (%)	FLOPs (G)	Time (ms)
LWRF-ResNet50 [38]	41.7	37.9	19.56 ± 0.29
LWRF-ResNet101 [38]	43.6	60.8	27.16 ± 0.19
LWRF-ResNet152 [38]	44.4	83.5	35.82 ± 0.23
RefineNet-ResNet50 [20]	42.5	285.2	54.18 ± 0.46
RefineNet-ResNet101 [20]	43.6	308.1	60.25 ± 0.53
RefineNet-ResNet152 [20]	46.5 (msc)	330.9	69.37 ± 0.78
CARNet-Mobile	39.8	**11.0**	$\mathbf{8.52 \pm 0.77}$
CARNet-ResNet50	43.1	38.1	9.91 ± 1.71
CARNet-ResNet101	45.3	61.0	15.06 ± 2.96
CARNet-ResNet152	**46.6**	83.8	20.15 ± 3.02

TABLE 3. The results on the ADE20K validation set. "msc" indicates that the method adopts the multi-scale strategy. "–" means the result is unavailable.

Method	Params(M)	MIoU(%)
SegNet [29]	29.6	21.6
Cascaded-SegNet [37]	–	27.5
FCN-8S [9]	135.0	29.4
DilatedNet [15], [39]	–	32.3
Cascaded-DilatedNet [37]	–	34.9
RefineNet-ResNet101 [20]	118.3	40.2 (msc)
RefineNet-ResNet152 [20]	134.0	40.7 (msc)
PSPNet-ResNet101 [17]	250.9	42.0
PSPNet-ResNet152 [17]	–	**42.6**
CARNet-Mobile	**4.2**	34.5
CARNet-ResNet101	48.0	40.9
CARNet-ResNet152	63.7	41.7

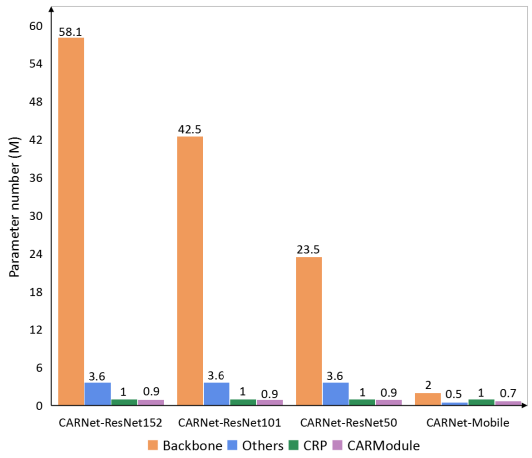


FIGURE 7. The detailed parameter number of several key components in different CARNet versions. "Others" indicates the 3 × 3 convolutional classifier and 1 × 1 convolutions used in dimensional adjustment.

parametric analysis, FLOPs analysis accuracy analysis, hyperparameter setting analysis, and visualized analysis.

1) PARAMETRIC ANALYSIS

In this part, we investigate the parameter number of several components of our method, which is shown in Fig. 7. As for the three ResNet-based versions, the baseline structure contains the largest number of parameters (approximately 58 M to 23 M). The "Others" component, which includes 3 × 3 and 1 × 1 convolutions, contains 3.6 M parameters. The CRP component contains 1 M parameters. Our four CARModules only contain 0.9 M parameters, which is almost negligible compared to the baseline structure. As for the CARNet-Mobile version, the baseline parameter number substantially decreases to 2 M, and the parameter of the CARModule part also slightly decreases to 0.7 M. Thus, the CARNet-Mobile is always the most computationally efficient among all four versions of our method. Based on these results, we can see that our CARModule always keeps lightweight with different backbone structures.

TABLE 4. The detailed FLOPs of several key components in our different CARNet versions. "Others" indicates the 3 × 3 convolutional classifier and 1 × 1 convolutions used in dimensional adjustment.

Method	BS	CARModule	CRP	Others
CARNet-ResNet152	121.379 G	0.024 G	11.465 G	10.074 G
CARNet-ResNet101	82.281 G	0.024 G	11.464 G	10.076 G
CARNet-ResNet50	43.243 G	0.024 G	11.464 G	10.075 G
CARNet-Mobile	3.41 G	0.023 G	11.464 G	2.727 G

2) FLOPs ANALYSIS

In this part, we study the FLOPs of several components of our method, which is shown in Table. 4. As for the three ResNet-based versions, the baseline structure contains the largest FLOPs (approximately 43.243 G to 121.379 G). The "Others" component contains about 10.075 G FLOPs, and the CRP component contains about 11.465 G FLOPs. However, our four CARModules only contain 0.024 G FLOPs. Compared to the baseline, CRP and "Others" components, our CARModules are negligible. As for the CARNet-Mobile version, the baseline FLOPs substantially decrease to 3.41 G. The "Others" component and CRP component contain 2.727 G and 11.464 G FLOPs respectively. But our CARModules are still negligible, which totally contain 0.023 G FLOPs. Based on these results, we can see that our CARModule is the most computationally modest among the components of our CARNet.

3) ACCURACY ANALYSIS

In this part, we first study each component of our method in terms of their contribution to the final accuracy. We adopt ResNet50 and MobileNet as two baselines. In Table. 5, taking ResNet50-based models for example, "BS" denotes we just utilize the baseline ResNet50 network to realize semantic segmentation. "BS+CARModule" denotes that we introduce four CARModules into the baseline. "BS+CARModule+SCLoss" denotes we additionally introduce the full SCLoss into the overall loss function. "BS+CARModule+SCLoss+CRP" denotes the full version of our method, in which the CRP module is also adopted. From the results in Table. 5, we can see that our two key components have solid support for boosting the segmentation accuracy in both two experiment sets (ResNet50-based

TABLE 5. The ablation study for contributions of each component on the NYUDv2 test set.

Method	MIoU(%)	improvement(%)
ResNet50-based model:		
BS	35.0	/
BS+CARModule	40.1	**+5.1**
BS+CARModule+SCLoss	42.0	+1.9
BS+CARModule+SCLoss+CRP	**43.1**	+1.1
MobileNet-based model:		
BS	30.7	/
BS+CARModule	34.6	**+3.9**
BS+CARModule+SCLoss	37.0	+2.4
BS+CARModule+SCLoss+CRP	**39.8**	+2.8

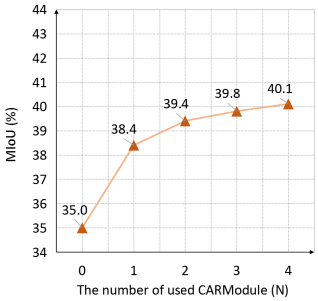


FIGURE 8. The results for the influence of the different CARModule number (N) on the NYUDv2 test set. The experiments are conducted on the "ResNet50+CARModule" framework. Since four different feature resolutions are included in the backbone network, we set $N = \{0, 1, 2, 3, 4\}$. By increasing N, the modules from CARModule-1 to CARModule-4 are successively applied.

and MobileNet-based). On the one hand, the CARModule obtains the largest accuracy gain (5.1% and 3.9%) in both experiments. On the other hand, using the SCLoss also has a clear enhancement of the segmentation accuracy (1.9% and 2.4%). In addition, we observe that the CRP component also contributes to the accuracy enhancement (1.1% and 2.8%).

Then, we further investigate the influence of the different CARModule number. To eliminate the interference brought by other components (e.g., SCLoss and CRP module), we adopt a degraded CARNet, i.e. "ResNet50+CARModule". The results in Fig. 8 demonstrate when $N = 4$(feature fusion is gradually conducted at all the resolutions) the best performance is achieved, and the effectiveness of our CARModule is validated. In addition, we use the full version of CARNet (i.e., "ResNet50+CARModule+SCLoss+CRP") to evaluate the different number of SCLoss terms used in the total loss function. Based on the results shown in Fig. 9, we find that, using four SCLoss terms by equipping each CARModule with a SCLoss term yields the best performance. These results again validate the effectiveness of combining the global SCLoss with the local cross-entropy loss.

4) HYPERPARAMETER SETTING ANALYSIS

In this part, we provide experiments to validate the hyperparameter setting by conducting the experiments on the Cityscapes validation set with $\lambda = \{0, 0.2, 0.4, 0.6, 0.8, 1\}$. To speed up the experiments and appropriately decrease GPU

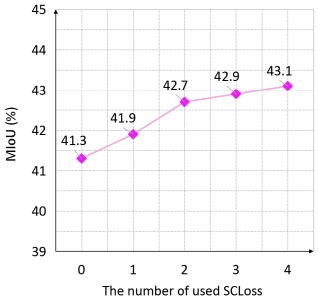


FIGURE 9. The results for the influence of the different number of SCLoss on the NYUDv2 test set. The experiments are conducted on the "ResNet50+CARModule+SCLoss+CRP" framework. With the number increased, the modules from CARModule-1 to CARModule-4 are successively equipped with SCLoss.

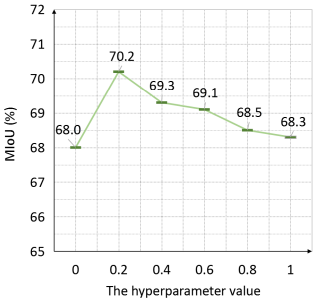


FIGURE 10. The study for the hyperparameter λ on the Cityscapes validation set. The experiments are conducted on the "ResNet50+CARModule+SCLoss+CRP" framework. We find that $\lambda = 0.2$ yields the best performance.

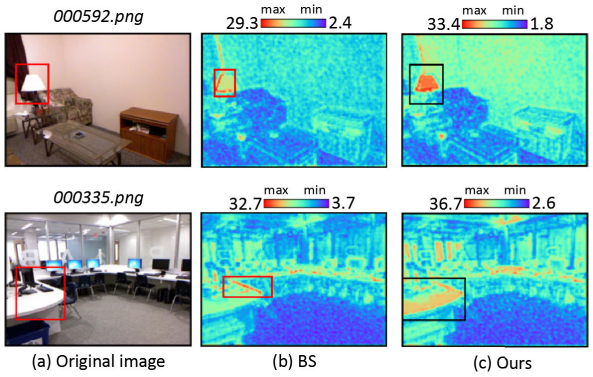


FIGURE 11. Some examples of the learned feature maps from Res-1 of our method and the baseline (BS). The color bar implies the value range of each feature map.

memory costs, we adopt 500 × 500 crop size in this part. Particularly, the full version of the CARNet framework (i.e, "ResNet50+CARModule+SCLoss+CRP") is used. The experimental results are shown in Fig. 10. We find that $\lambda = 0.2$ yields the best performance. Based on this validation, we apply $\lambda = 0.2$ to all the experiments mentioned above.

5) VISUALIZED ANALYSIS

We also conduct some visualized analysis of our method. First, we show an example of a visualized shallow-layer (Res-1) feature map in Fig. 11, in which ResNet50 is used

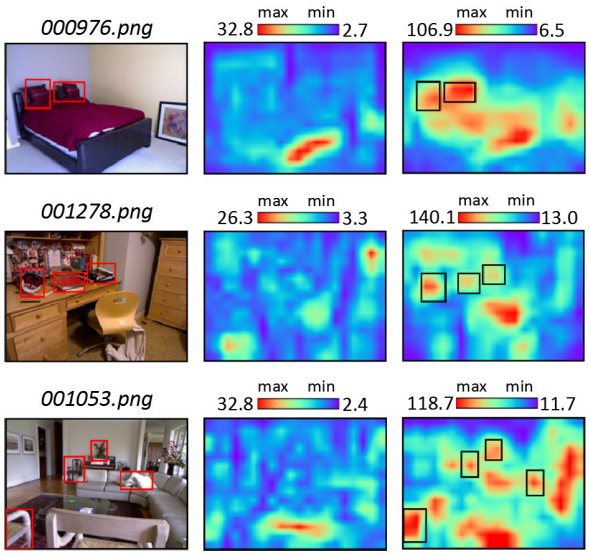


FIGURE 12. Some examples of the learned feature maps from Res-4 of our method and the baseline (BS). The color bar implies the value range of each feature map.

as the baseline. From the rectangles, we can see that our method learns a more semantically consistent feature map than the baseline version. We also visualize the feature maps learned from deep layers (Res-4) in Fig. 12. By comparing Fig. 12 (b) with (c), we further observe the improvements over the baseline. First, as exemplified in the small rectangles, our method becomes more aware of the small objects than the baseline method. Second, the heated areas in Fig. 12 (c) better mimic the visual saliency to some extent. Third, from the color bars, we observe the value range in Fig. 12 (c) is much larger than that in Fig.12 (b), which implies the features learned from our method are more discriminative. The above observations help to validate that our method learns better features for accurate segmentation.

V. CONCLUSION

In this paper, we propose the CARNet to achieve a good balance between segmentation accuracy and speed in semantic segmentation. In our method, we build the CARModule that effectively fuses the features from different levels. We also introduce the SCLoss to better regularize the training process. We validate our method on several public datasets in the experiments and obtain promising qualitative and quantitative results. As for future research, we have the following considerations. First, we aim to apply our CARModule in other fine-grained segmentation applications, such as instance segmentation and panoptic segmentation. Second, for the CARModule, we try to improve the attention mechanism for feature fusion.

REFERENCES

[1] M. Siam, S. Elkerdawy, M. Jagersand, and S. Yogamani, "Deep semantic segmentation for automated driving: Taxonomy, roadmap and challenges," in *Proc. IEEE 20th Int. Conf. Intell. Transp. Syst. (ITSC)*, Oct. 2017, pp. 1–8.

[2] M. Siam, M. Gamal, M. Abdel-Razek, S. Yogamani, M. Jagersand, and H. Zhang, "A comparative study of real-time semantic segmentation for autonomous driving," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2018, pp. 1–10.

[3] T. Park, M.-Y. Liu, T.-C. Wang, and J.-Y. Zhu, "Semantic image synthesis with spatially-adaptive normalization," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 2337–2346.

[4] T.-C. Wang, M.-Y. Liu, J.-Y. Zhu, A. Tao, J. Kautz, and B. Catanzaro, "High-resolution image synthesis and semantic manipulation with conditional GANs," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 8798–8807.

[5] Y. Guo, Y. Gao, and D. Shen, "Deformable MR prostate segmentation via deep feature learning and sparse patch matching," *IEEE Trans. Med. Imag.*, vol. 35, no. 4, pp. 1077–1089, Dec. 2016.

[6] X. Zhu, H.-I. Suk, H. Huang, and D. Shen, "Low-rank graph-regularized structured sparse regression for identifying genetic biomarkers," *IEEE Trans. Big Data*, vol. 3, no. 4, pp. 405–414, Dec. 2017.

[7] M. Wang, H. Li, D. Tao, K. Lu, and X. Wu, "Multimodal graph-based reranking for Web image search," *IEEE Trans. Image Process.*, vol. 21, no. 11, pp. 4649–4661, Nov. 2012.

[8] R. Hong, M. Wang, Y. Gao, D. Tao, X. Li, and X. Wu, "Image annotation by multiple-instance learning with discriminative feature mapping and selection," *IEEE Trans. Cybern.*, vol. 44, no. 5, pp. 669–680, May 2014.

[9] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 3431–3440.

[10] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, *arXiv:1409.1556*. [Online]. Available: http://arxiv.org/abs/1409.1556

[11] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2012, pp. 1097–1105.

[12] 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.

[13] A. Paszke, A. Chaurasia, S. Kim, and E. Culurciello, "ENet: A deep neural network architecture for real-time semantic segmentation," 2016, *arXiv:1606.02147*. [Online]. Available: http://arxiv.org/abs/1606.02147

[14] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Proc. Int. Conf. Med. Image Comput. Comput.-Assist. Intervent.* Berlin, Germany: Springer, 2015, pp. 234–241.

[15] F. Yu and V. Koltun, "Multi-scale context aggregation by dilated convolutions," 2015, *arXiv:1511.07122*. [Online]. Available: http://arxiv.org/abs/1511.07122

[16] W. Liu, A. Rabinovich, and A. C. Berg, "ParseNet: Looking wider to see better," 2015, *arXiv:1506.04579*. [Online]. Available: http://arxiv.org/abs/1506.04579

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

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

[19] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "MobileNetV2: Inverted residuals and linear bottlenecks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 4510–4520.

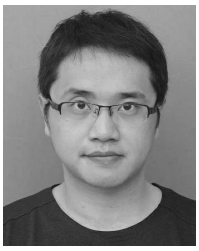
[20] G. Lin, A. Milan, C. Shen, and I. Reid, "RefineNet: Multi-path refinement networks for high-resolution semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 1925–1934.

[21] H. Noh, S. Hong, and B. Han, "Learning deconvolution network for semantic segmentation," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 1520–1528.

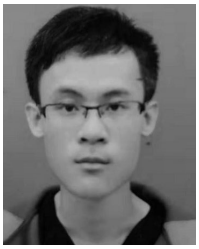
[22] V. Badrinarayanan, A. Kendall, and R. Cipolla, "SegNet: A deep convolutional encoder-decoder architecture for image segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 12, pp. 2481–2495, Dec. 2017.

[23] A. Kendall, V. Badrinarayanan, and R. Cipolla, "Bayesian SegNet: Model uncertainty in deep convolutional encoder-decoder architectures for scene understanding," 2015, *arXiv:1511.02680*. [Online]. Available: http://arxiv.org/abs/1511.02680

[24] X. Li, Z. Zhong, J. Wu, Y. Yang, Z. Lin, and H. Liu, "Expectation-maximization attention networks for semantic segmentation," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 9167–9176.
[25] 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.
[26] G. J. Brostow, J. Fauqueur, and R. Cipolla, "Semantic object classes in video: A high-definition ground truth database," *Pattern Recognit. Lett.*, vol. 30, no. 2, pp. 88–97, Jan. 2009.
[27] C. Yu, J. Wang, C. Peng, C. Gao, G. Yu, and N. Sang, "Bisenet: Bilateral segmentation network for real-time semantic segmentation," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 325–341.
[28] H. Li, P. Xiong, H. Fan, and J. Sun, "DFANet: Deep feature aggregation for real-time semantic segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 9522–9531.
[29] V. Badrinarayanan, A. Handa, and R. Cipolla, "SegNet: A deep convolutional encoder-decoder architecture for robust semantic pixel-wise labelling," 2015, *arXiv:1505.07293*. [Online]. Available: http://arxiv.org/abs/1505.07293
[30] M. Treml, J. Arjona-Medina, T. Unterthiner, R. Durgesh, F. Friedmann, P. Schuberth, A. Mayr, M. Heusel, M. Hofmarcher, and M. Widrich, "Speeding up semantic segmentation for autonomous driving," in *Proc. Neural Inf. Process. Syst. Workshop Mach. Learn. Intell. Transp. Syst.*, vol. 1, 2016, p. 5.
[31] S. Zheng, S. Jayasumana, B. Romera-Paredes, V. Vineet, Z. Su, D. Du, C. Huang, and P. H. S. Torr, "Conditional random fields as recurrent neural networks," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 1529–1537.
[32] T. Pohlen, A. Hermans, M. Mathias, and B. Leibe, "Full-resolution residual networks for semantic segmentation in street scenes," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 4151–4160.
[33] Z. Wu, C. Shen, and A. van den Hengel, "Real-time semantic image segmentation via spatial sparsity," 2017, *arXiv:1712.00213*. [Online]. Available: http://arxiv.org/abs/1712.00213
[34] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The cityscapes dataset for semantic urban scene understanding," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 3213–3223.
[35] S. Gupta, P. Arbelaez, and J. Malik, "Perceptual organization and recognition of indoor scenes from RGB-D images," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2013, pp. 564–571.
[36] N. Silberman, D. Hoiem, P. Kohli, and R. Fergus, "Indoor segmentation and support inference from RGBD images," in *Proc. Eur. Conf. Comput. Vis.* Berlin, Germany: Springer, 2012, pp. 746–760.
[37] B. Zhou, H. Zhao, X. Puig, S. Fidler, A. Barriuso, and A. Torralba, "Scene parsing through ADE20K dataset," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 633–641.
[38] V. Nekrasov, C. Shen, and I. Reid, "Light-weight RefineNet for real-time semantic segmentation," 2018, *arXiv:1810.03272*. [Online]. Available: http://arxiv.org/abs/1810.03272
[39] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Semantic image segmentation with deep convolutional nets and fully connected CRFs," 2014, *arXiv:1412.7062*. [Online]. Available: http://arxiv.org/abs/1412.7062



SHIJIE HAO received the Ph.D. degree from the Hefei University of Technology (HFUT), in 2012. He is currently an Associate Professor with the School of Computer Science and Information Engineering, HFUT. He is also with the Key Laboratory of Knowledge Engineering with Big Data, Ministry of Education, HFUT. His research interests include image processing and multimedia content analysis.



YUAN ZHOU is currently pursuing the master's degree with the School of Computer and Information, Hefei University of Technology. He is also with the Key Laboratory of Knowledge Engineering with Big Data, Ministry of Education, Hefei University of Technology. His research interest includes image segmentation.

YOUMING ZHANG is currently pursuing the bachelor's degree with the School of Mathematics and Statistics, Northeastern University, Qinhuangdao. His research interest includes digital image processing and analysis.

YANRONG GUO received the Ph.D. degree from the Hefei University of Technology (HFUT), in 2013. She was a Postdoctoral Researcher with the University of North Carolina at Chapel Hill (UNC), from 2013 to 2016. She is currently an Associate Professor with the School of Computer and Information, HFUT. She is also with the Key Laboratory of Knowledge Engineering with Big Data, Ministry of Education, HFUT. Her research interests include image processing and pattern recognition.