DABNet: Depth-wise Asymmetric Bottleneck for Real-time Semantic Segmentation

Gen Li
ligen@skku.edu
Joongkyu Kim†
jkkim@skku.edu

Department of Electronic, Electrical and Computer Engineering
Sungkyunkwan University
Suwon, South Korea

Abstract

As a pixel-level prediction task, semantic segmentation needs large computational cost with enormous parameters to obtain high performance. Recently, due to the increasing demand for autonomous systems and robots, it is significant to make a trade-off between accuracy and inference speed. In this paper, we propose a novel Depth-wise Asymmetric Bottleneck (DAB) module to address this dilemma, which efficiently adopts depth-wise asymmetric convolution and dilated convolution to build a bottleneck structure. Based on the DAB module, we design a Depth-wise Asymmetric Bottleneck Network (DABNet) especially for real-time semantic segmentation, which creates sufficient receptive field and densely utilizes the contextual information. Experiments on Cityscapes and CamVid datasets demonstrate that the proposed DABNet achieves a balance between speed and precision. Specifically, without any pretrained model and post-processing, it achieves 70.1% Mean IoU on the Cityscapes test dataset with only 0.76 million parameters and a speed of 104 FPS on a single GTX 1080Ti card. Code is available at `https://github.com/Reagan1311/DABNet`.

1 Introduction

Recently, autonomous systems and augmented reality devices have drawn widespread attention, which are the main application fields of semantic segmentation. Such real-world applications not only demand competitive performance with low energy and memory but also have a strict requirement for inference speed. Existing real-time semantic segmentation [16, 18, 19, 23] models have successfully sped up to real time, but meanwhile, they significantly sacrifice model accuracy. Therefore, how to design an effective real-time semantic segmentation network with fast inference speed and small capacity becomes a challenging problem.

Most of the previous excellent work [3, 4, 5, 12, 31] has already proved the availability of dilated convolution which is able to create sizeable receptive field while maintaining the number of parameters. Thus, many existing semantic segmentation models aiming at real time employ dilated convolution in their network. Another method to effectively reduce the number of parameters is depth-wise separable convolution (ds-Conv). It computes cross-channel and spatial correlations independently, which is widely used in lightweight models

†Corresponding author

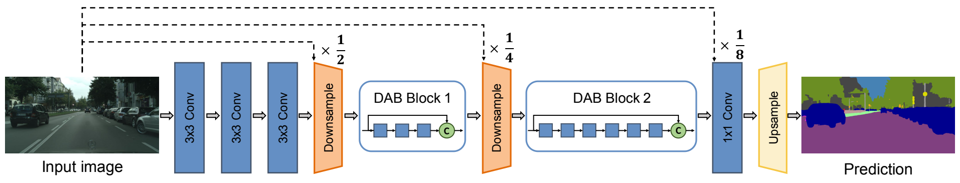

Figure 1: Architecture of proposed Depth-wise Asymmetric Bottleneck Network. "C" means concatenation, dashed lines indicate average pooling operation.

[7, 13, 22]. However, simply replacing the standard convolution with ds-Conv results in large performance degradation, as ds-Conv dramatically reduces the parameters, and it often leads to a sub-optimal problem. Therefore, here we hold the opinion that a well-designed combination of dilated convolution and ds-Conv is especially suitable for real-time semantic segmentation.

Based on the above observation, we propose a novel network architecture specially designated for real-time semantic segmentation, which is presented in Figure 1. More specifically, we design a depth-wise asymmetric bottleneck to extract dense feature under a shallow network, which has common advantages of both dilated convolution and depth-wise separable convolution.

Our main contributions could be summarized as:

- We propose a bottleneck structure which incorporates depth-wise asymmetric convolution with dilated convolution, and it is termed as Depth-wise Asymmetric Bottleneck (DAB). This structure extracts local and contextual information jointly and dramatically reduces the parameters.

- We elaborately design DABNet based on DAB module, it has much fewer parameters than the existing state-of-the-art real-time semantic segmentation methods while providing comparable accuracy and faster inference speed.

- We achieve remarkable results on both of Cityscapes [8] and CamVid [2] benchmarks without any context module, pretrained model, and post-processing scheme. Under an equivalent number of parameters, the proposed DABNet significantly outperforms existing semantic segmentation networks. It can run on high-resolution images (512×1024) at 104 FPS on a single GTX 1080Ti card and 70.1% Mean IoU on the Cityscapes test dataset with merely 0.76 M parameters.

2 Related Work

Recent real-time semantic segmentation models use multiple different techniques, such as dilated convolution and convolution factorization, to enlarge the receptive field and speed up networks. In this section, we briefly introduce recent research progress and describe some approaches that are of great help in real-time semantic segmentation.

Real-time semantic segmentation. Real-time semantic segmentation network requires finding a trade-off between high-quality prediction and high-inference speed. ENet [18] is the first network to be designed in real time, it trims a great number of convolution filters to

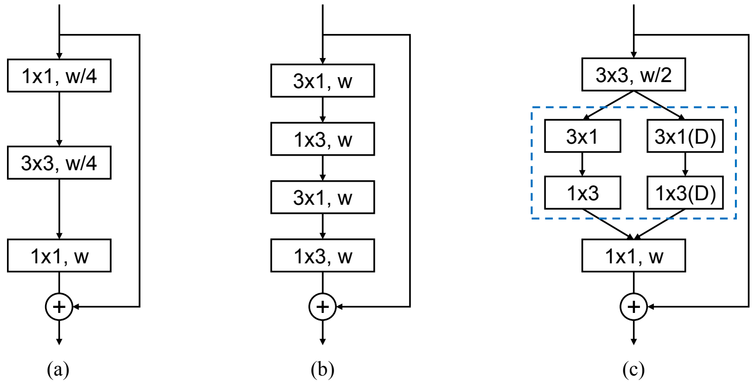

Figure 2: (a) ResNet bottleneck design. (b) ERFNet non-bottlenck-1D module. (c) Our DAB module (Convolutions in the dashed box are depth-wise convolution). "W": The number of input channels. "D": Dilated convolution. For brevity, we omit normalization and activation function.

reduce computation. ICNet [34] proposes an image cascade network that incorporates multi-resolution branches. ERFNet [20] uses residual connections and factorized convolutions to remain efficient while retaining remarkable accuracy. More recently, ESPNet [16] introduces an efficient spatial pyramid (ESP), which brings great improvement in both speed and performance. BiSeNet [30] proposes two paths to combine spatial information and context information. These networks successfully made a trade-off between speed and performance, but there is still sufficient space for further improvement.

Dilated convolution. Dilated convolution [12] inserts zeros between each pixel in a standard convolution, which leads to a large effective receptive field without increasing parameters, hence it is generally used in semantic segmentation models. In DeepLab series [4, 5, 6], an atrous spatial pyramid pooling (ASPP) module is introduced which employs multiple parallel filters with different dilation rates to collect multi-scale information. DenseASPP [29] concatenates a set of dilated convolution layers to generate dense multi-scale feature representation. Most of the state-of-the-art networks in semantic segmentation exploit dilated convolution, which proves its effectiveness in pixel-level prediction task.

Convolution Factorization. Convolution factorization divides a standard convolution operation into several steps to reduce the computational cost and memory, which is extensively adopted in lightweight CNN models. Inception [24, 25, 26] employ several small-sized convolutions to replace the convolution with large kernel size while maintaining the size of the receptive field. Xception [7] and MobileNet [13] use the depth-wise separable convolution to reduce the amount of computation with only a slight drop in performance. MobileNetV2 [22] proposes an inverted residual block and linear bottlenecks to further improve the performance. ShuffleNet [32] applies the point-wise group convolution with channel shuffle operation to enable information communication between different groups of channels.

3 Proposed Network

In this section, we first introduce the DAB module, which is the core component of DABNet. Then based on DAB module, we elaborate design choices of the final model, the overall structure of proposed DABNet is shown in Figure 1.

3.1 Depth-wise Asymmetric Bottleneck

Inspired by bottleneck design in ResNet [10] (See Figure 2(a)) and factorized convolutions in ERFNet [20] (See Figure 2(b)), we design the DAB module with their common advantages.
Bottleneck design. DAB module follows the bottleneck structure similar to ResNet [10]. In order to reduce the parameters and accelerate training process, ResNet-50, 101, 152 modify the original block as a bottleneck design, which is shown in Figure 2(a).

For the same reason, here we also utilize the bottleneck structure in DAB module, as shown in Figure 2(c). Every bottleneck firstly reduces the number of channels by half and restores original channels by a point-wise convolution. Nevertheless, we make some modifications on the original bottleneck structure: (1) We use 3×3 convolution at the beginning of each DAB module. Here we clarify the reason: although a 1×1 convolution has fewer parameters than 3×3 convolution, the intention of ResNet [10] is to make a deep model with more than 100 layers, since deep convolutional neural network can increase the receptive field and capture more complex features. Unfortunately, higher number of layers also bring increasing runtime and memory requirements. Therefore, for the sake of high inference speed, we instead use the 3×3 convolution to avoid making a deep model. (2) We only reduce the number of channels by half after the first convolution, as the maximum number of channels in our model is only 128 compared with thousands of channels in ResNet [10]. Thus, for preserving spatial information, we do not compress the channels too much.
Two-branch structure. For semantic segmentation, previous remarkable networks have already demonstrated the significance of semantic context information, good prediction results usually come from a combination of hierarchical information, consequently how to effectively fuse multi-scale information remains a tough problem. In DAB module, we design a novel two-branch structure and each branch is responsible for extracting corresponding information.

To extract local information, we use a simple 3×3 depth-wise convolution in the first branch. For further reducing the computation, referred by ERFNet [20] non-bottleneck-1D module (Figure 2(b)), we apply convolution factorization to depth-wise convolution. Namely, a standard $n \times n$ depth-wise convolution is substituted for an $n \times 1$ depth-wise convolution followed by a $1 \times n$ depth-wise convolution. For an $N \times N$ kernel, asymmetric convolution reduces the computational complexity per pixel from $O(N^2)$ to $O(N)$.

To extract broader context information, we adopt dilated convolution, which creates a large receptive field without decreasing the resolution of the feature map. However, when dilation rate becomes larger and larger (in most cases, it increases up to 16 or more), we need to implement lots of padding to maintain the size of the feature map. This brings heavy computational cost, which prohibits dilated convolution from being used in a real-time model. In order to resolve the above contradiction, the second branch only applies dilated convolution to the depth-wise asymmetric convolution to reduce computational cost, thus it is called "depth-wise asymmetric dilated convolution".

In summary, all the convolutions we used in the "neck" part of our bottleneck structure are depth-wise convolution. In the two-branch structure, each branch can be termed as local information branch and context information branch on the basis of their respective function. Finally, we add the two branches together and then a 1×1 point-wise convolution is employed at the end of each DAB module, which is used to restore the number of channels and fuse all the channel information.

Layer	Operator	Mode	Channel	Output size
1	3×3 Conv	stride 2	32	256×512
2	3×3 Conv	stride 1	32	256×512
3	3×3 Conv	stride 1	32	256×512
4	Downsampling	-	64	128×256
5-7	$3\times$ DAB module	dilated 2	64	128×256
8	Downsampling	-	128	64×128
9-10	$2\times$ DAB module	dilated 4	128	64×128
11-12	$2\times$ DAB module	dilated 8	128	64×128
13-14	$2\times$ DAB module	dilated 16	128	64×128
15	1×1 Conv	stride 1	19	64×128
16	Bilinear interpolation	$\times8$	19	512×1024

Table 1: Detailed structure of proposed DABNet.

Activation function. In our DAB module, we adopt pre-activation scheme [11] and batch normalization [14] is used before every non-linear function. Referring to ENet [18], we use PReLU [9] as nonlinearity function, as PReLU achieves slightly better performance than ReLU due to the shallow network model. And as mentioned in [22], using non-linear layers in bottlenecks hurts the performance, therefore non-linearity is removed after the final 1×1 point-wise convolution.

3.2 Network Architecture Design

Based on the DAB module, we elaborately design the architecture of DABNet as shown in Figure 1. In this subsection, we discuss the design choices on the final model of DABNet. Since our research aims at the lightweight model with fast inference speed and comparable results, the essence is to build a shallow model with fewer parameters and remove redundant components. The detailed architecture of DABNet is presented in Table 1.

Downsampling. We first use three 3×3 convolutions to extract initial features, then we employ the same downsampling block with ENet [18] initial block, which is a concatenation of a 3×3 convolution with stride 2 and a 2×2 max-pooling. Similarly, we only use the initial block in the first downsampling, the second downsampling block is a single 3×3 convolution with stride 2.

Downsampling operation reduces the size of the feature map and enlarges the receptive field for extracting more contextual information. However, a reduction in the resolution of the feature map often results in information loss, which has a severe effect on the final prediction result. Therefore, to preserve the spatial information and details, we only employ three downsampling operations in our model and finally obtain $1/8$ feature map resolution, while most of existing semantic segmentation models employ five downsampling operations and get $1/32$ feature map resolution. Furthermore, inspired by ESPNetv2 [17], we build a long-range shortcut connection between the input image and each downsampling block as well as the last convolution layer (see Figure 1), which facilitates feature reuse and compensates information loss. This connection first downsamples the original image to the same size as the feature map and then concatenates them together.

DAB block. We design two DAB blocks in the DABNet, which include several consecutive DAB modules for dense feature extraction. The first and the second DAB block consist of 3 and 6 DAB modules, respectively.

To better strengthen spatial relationships and feature propagation, we introduce inter-block concatenation to combine high-level feature with low-level feature, which means stacking first DAB module with last DAB module in each DAB block. Besides, we employ dilated convolution in every DAB module as mentioned in Section 3.1. Specifically, all the DAB module in DAB block 1 include a depth-wise asymmetric dilated convolution with dilation rate 2, and dilation rates in DAB block 2 are 4, 4, 8, 8, 16, 16, respectively. As with [31], we choose this sequential scheme to enlarge the receptive field gradually.

Design choices. Note that DABNet is lack of decoder structure, which is much different from popular encoder-decoder structure utilized in most of the semantic segmentation models. In pursuit of faster inference speed, we discard the decoder in DABNet. Although the decoder can lead to an effective increase in accuracy, even a small decoder would slow down the network and bring extra computation. Besides, as we only downsample the input image three times, decoder is not essential in our network.

To make an end-to-end deep learning architecture, we do not use any post-processing technique to improve the final result. Also, our model does not depend on any backbone, we design our model from scratch. It is worth noting that the capacity of DABNet is extremely low, and we employ less than 0.76 million parameters.

4 Experiments

In this section, we evaluate our proposed network on two challenging datasets: Cityscapes [8] and Camvid [2], which are widely used in semantic segmentation. Firstly, we introduce the datasets and implementation protocol. Then, we conduct several experiments on Cityscapes validation set to prove the effectiveness of our network. Finally, we report the comparisons with state-of-the-art systems, all the performances are measured using mean intersection-over-union (mIoU).

4.1 Experimental Settings

Cityscapes. The Cityscapes is a large urban street scene dataset. It contains a train set of 2975 images, a validation set of 500 images and a test set of 1525 images. There is another set of 19,998 images with coarse annotation, but we only use the fine annotated images for all experiments. The images have a resolution of 1024×2048 and 19 semantic categories.

CamVid. The CamVid is another street scene dataset for autonomous driving applications. It includes 701 images in total, 367 for training, 101 for validation and 233 for testing. The images have a resolution of 360×480 and 11 semantic categories.

Implementation protocol. All the experiments are performed with one 1080Ti GPU, CUDA 9.0 and cuDNN V7 on the Pytorch platform. Runtime evaluation is performed on a single 1080Ti card, we report an average of 100 frames for the frames per second (FPS) measurement.

We use mini-batch stochastic gradient descent (SGD) with batch size 8, momentum 0.9 and weight decay $1e^{-4}$ in training. We employ the "poly" learning rate policy [5] and the initial learning rate is set to $4.5e^{-2}$ with power 0.9. As no pre-training is used, here we set the maximum number of epochs to 1000. For data augmentation, we employ random horizontal flip, the mean subtraction, and random scale on the input images during training. The random scale contains {0.75, 1.0, 1.25, 1.5, 1.75, 2.0}. Finally, we randomly crop the image into fixed size for training.

Model	mIoU (%)	FPS	Parameters (M)
DABNet-baseline	69.1	104.2	0.76
(a) Accuracy			
DABNet-(r=4)	66.8	104.3	0.76
DABNet-(r=3, 3, 7, 7, 13, 13)	68.4	104.2	0.76
DABNet-ERFdecoder	69.4	58.6	1.02
DABNet-SPP	68.6	72.2	3.22
(b) Speed			
DABNet-w/o dilation	-	104.5	0.76
DABNet-First 3x3 conv(r=2)	-	85.6	0.76

Table 2: Ablation study results on Cityscapes validation set. FPS are estimated for an input of 512×1024 on a single GTX 1080Ti.

4.2 Ablation Study

In this subsection, we design a series of experiments to demonstrate the effectiveness of our network. We adopt Cityscapes dataset [8] to conduct quantitative and qualitative analysis. All the ablation studies are trained from Cityscapes training set and evaluated on Cityscapes validation set.

Dilation rates. We adopt gradually increasing sequence of dilation rates in the DAB block 2, which is {4, 4, 8, 8, 16, 16}. To investigate the effectiveness of this scheme, we set all the dilation rate as 4 in the DAB block 2 for comparison. In addition, previous work [28] proposed that coprime dilation rates yield better results, therefore we set another varying dilation rates list, which is {3, 3, 7, 7, 13, 13}. In Table 2(a), DABNet-(r=4) performs 2.3% lower accuracy than DABNet, which indicates the importance of increasing dilation rates. Also, coprime dilation rates perform worse than DABNet, it seems large dilation rates are more effective in our network.

Decoder. For striking a balance between the speed and prediction performance, we discard the decoder module in DABNet. Here we build a network with the decoder of ERFNet for comparison and use deconvolution layer to restore the resolution of the feature map. As can be seen from Table 2(a), adding the ERF decoder can bring 0.3% slightly better accuracy, while resulting in increased runtime with only 58.6 FPS. Apparently, the decoder is not essential in DABNet, which heavily slows down the model.

Context module. PSPNet [33] is a classic model which employs a spatial pyramid pooling (SPP) module to improve performance by capturing global and local context at different feature resolutions. To explore the ability to capture context information, we construct a DABNet variant with an SPP head in the end, which is termed as "DABNet-SPP". Table 2(a) shows that DABNet-SPP even decreases 0.5% accuracy, also it has 4.2% times more parameters and slows down the model by 32 FPS. Therefore, we conclude that the DAB module can better capture context information than a heavy SPP context module.

Inference speed. As mentioned in Section 3.1, dilated convolution brings heavy computational cost and slows down the model. In order to explore the effect of dilated convolution on inference speed, we design two experiments for comparison in FPS. In the first experiment we remove all the dilated convolutions in DABNet, and in the other experiment we further adopt a dilation rate 2 in the first 3×3 standard convolution of our DAB module. As presented in both Table 2 (a) and (b), even if we decrease the dilation rate or remove

Method	Pretrain	InputSize	mIoU (%)	FPS	GPU	Parameters (M)
DeepLab-v2 [6]	ImageNet [21]	512×1024	70.4	<1	TitanX	44
PSPNet [33]	ImageNet	713×713	78.4	<1	TitanX	65.7
SegNet [1]	ImageNet	360×640	56.1	14.6	TitanX	29.5
ENet [18]	No	512×1024	58.3	76.9	TitanX	**0.36**
SQ [27]	ImageNet	1024×2048	59.8	16.7	TitanX	-
ESPNet [16]	No	512×1024	60.3	**112**	TitanX-P	**0.36**
ContextNet [19]	No	512×1024	66.1	65.5	TitanX	0.85
ERFNet [20]	No	512×1024	68.0	41.7	TitanX	2.1
BiSeNet [30]	ImageNet	768×1536	68.4	105.8	TitanXp	5.8
ICNet [34]	ImageNet	1024×2048	69.5	30.3	TitanX	26.5
DABNet (Ours)	No	512×1024	**70.1**	104.2	1080Ti	0.76

Table 3: Evaluation results on the Cityscapes test set. TitanX represents the TitanX Maxwell, TitanX-P represents the TitanX Pascal. (The GPU computational efficiency: TitanX Maxwell < Titan X Pascal ≈ 1080Ti < TitanXp)

Method	mIoU (%)	Parameters (M)
ENet [18]	51.3	**0.36**
SegNet [1]	55.6	29.5
FCN-8s [15]	57.0	134.5
Dilation8 [31]	65.3	140.8
BiSeNet [30]	65.6	5.8
ICNet [34]	**67.1**	26.5
DABNet(Ours)	66.4	0.76

Table 4: Evaluation results on the CamVid test set.

all the dilated convolutions, the FPS is hardly changed (range from 104.2 to 104.5). Nevertheless, when we apply dilated convolution (with only a dilation rate 2) to the standard convolution, there is an obvious decrease in speed, from 104.2 to 85.6. The experimental result proves that dilated convolution has a significant effect on inference speed, but when applied to depth-wise convolution, it has nearly no harmful effect.

4.3 Comparison with state-of-the-arts

In this subsection, we compare our algorithm with state-of-the-art models. We first report the final results on Cityscapes [8] and CamVid [2] benchmarks, then we analyze the speed of our model and compute the FPS of other state-of-the-art methods under the same status for fair comparison. During evaluation, we do not adopt any testing tricks such as multi-crop and multi-scale testing.

Accuracy and parameter comparisons. With only 0.76 million parameters, our DABNet achieves impressive results. Here we show the results on Cityscapes and CamVid benchmarks, respectively. As shown in Table 3, DABNet achieves 70.1% mIoU on Cityscapes test set, which significantly outperforms existing real-time segmentation work.

In general, the more parameters, the more complex the function that can be fitted. However, most of the large models have massive redundant parameters. Table 3 shows that DABNet only uses 3% of ICNet's parameters, but achieves a better result. Moreover, as can be observed, compared with the offline methods—PSPNet [33] and DeepLabV2 [6], we only

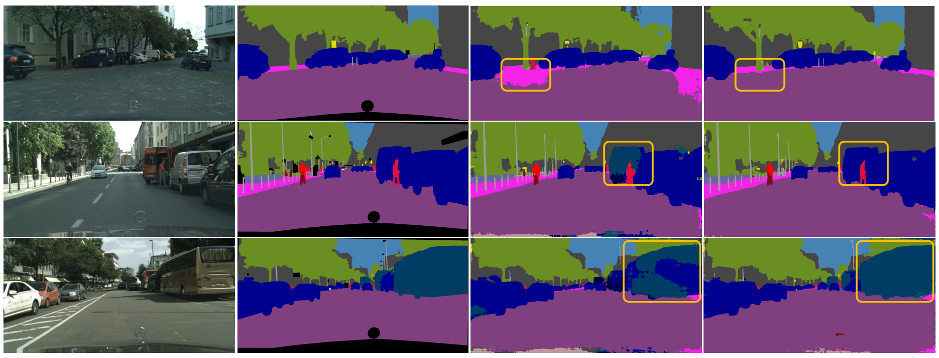

Figure 3: Qualitative examples of the segmentation on Cityscapes validation set. From left to right: Input image, ground-truth, prediction of ENet, and prediction of DABNet.

Model	GTX 1080Ti					
	256 × 512		512 × 1024		1024 × 2048	
	ms	fps	ms	fps	ms	fps
SegNet [1]	16	64.2	56	17.9	-	-
ENet [18]	10	99.8	13	74.9	44	22.9
ICNet [34]	9	107.9	15	67.2	40	25.1
ESPNet [16]	5	182.5	9	115.2	30	33.3
DABNet (Ours)	6	170.2	10	104.2	36	27.7

Table 5: Speed comparison of our method against other state-of-the-art methods. All the run-time is computed on a single GTX 1080Ti. ("-" indicates the model exceeds the maximum memory of a single GTX 1080Ti.)

use 1% of their parameters to keep a comparable performance. The qualitative results on Cityscapes validation set are presented in Figure 3. On CamVid test set, as reported in Table 4, DABNet again achieves outstanding performance with small capacity, and it can process a 360 × 480 CamVid image at the speed of 146 FPS.

Speed comparison. Since different methods employ various devices and diverse input sizes, for fair comparison of speed, all the speed experiments are performed with a single 1080Ti GPU on the Pytorch platform. Additionally, we also list the FPS and their test device from existing literature in Table 3 for reference. Table 5 presents the speed comparison between our method with other state-of-the-art approaches at full, half and quarter resolution on images of Cityscapes. DABNet is able to process a 1024 × 2048 image at the speed of 27.7 FPS. ESPNet [16] is one of the fastest real-time networks, and it has slightly quicker speed than DABNet. However it only achieves 60.3% mIoU which is 9.8% less than our network. The speed comparison represents that DABNet is able to process a high resolution image with fast inference speed while maintaining high performance.

5 Conclusions

In this paper, we propose a novel Depth-wise Asymmetric Bottleneck module to extract local and contextual features jointly. Based on the DAB module, we elaborately design the Depth-wise Asymmetric Bottleneck Network, a lightweight model with fast inference speed and competitive results. Through analysis and quantitative experimental results, we demonstrate the effectiveness of our method. The proposed DABNet achieves 70.1% Mean IoU on Cityscapes test set with only 0.76 million parameters, and can run at 104 fps on 512×1024 high-resolution images. In conclusion, our network shows significant improvements in all the aspects of accuracy, speed, and capacity compared with the state-of-the-art methods.

Acknowledgement

This work was supported by the National Research Foundation of the Korea (No.NRF-2017R1D1A1B03034288) and the MSIT (Ministry of Science, ICT), Korea, under the High-Potential Individuals Global Training Program (2019-0-01609) supervised by the IITP (Institute for Information & Communications Technology Planning & Evaluation).

References

[1] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 39(12):2481–2495, 2017.

[2] Gabriel J. Brostow, Jamie Shotton, Julien Fauqueur, and Roberto Cipolla. Segmentation and recognition using structure from motion point clouds. In *ECCV*, 2008.

[3] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan Loddon Yuille. Semantic image segmentation with deep convolutional nets and fully connected crfs. *CoRR*, abs/1412.7062, 2015.

[4] Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam. Rethinking atrous convolution for semantic image segmentation. *CoRR*, abs/1706.05587, 2017.

[5] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan Loddon Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40:834–848, 2018.

[6] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *ECCV*, 2018.

[7] FranÃğois Chollet. Xception: Deep learning with depthwise separable convolutions. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1800–1807, 2017.

[8] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes

dataset for semantic urban scene understanding. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3213–3223, 2016.

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pages 1026–1034, 2015.

[10] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.

[11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. In *ECCV*, 2016.

[12] Matthias Holschneider, Richard Kronland-Martinet, Jean Morlet, and Ph Tchamitchian. A real-time algorithm for signal analysis with the help of the wavelet transform. In *Wavelets*, pages 286–297. Springer, 1990.

[13] Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *CoRR*, abs/1704.04861, 2017.

[14] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, 2015.

[15] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *CVPR*, 2015.

[16] Sachin Mehta, Mohammad Rastegari, Anat Caspi, Linda G. Shapiro, and Hannaneh Hajishirzi. Espnet: Efficient spatial pyramid of dilated convolutions for semantic segmentation. In *ECCV*, 2018.

[17] Sachin Mehta, Mohammad Rastegari, Linda G. Shapiro, and Hannaneh Hajishirzi. Espnetv2: A light-weight, power efficient, and general purpose convolutional neural network. *CoRR*, abs/1811.11431, 2018.

[18] Adam Paszke, Abhishek Chaurasia, Sangpil Kim, and Eugenio Culurciello. Enet: A deep neural network architecture for real-time semantic segmentation. *CoRR*, abs/1606.02147, 2017.

[19] Rudra P. K. Poudel, Ujwal Bonde, Stephan Liwicki, and Christopher Zach. Contextnet: Exploring context and detail for semantic segmentation in real-time. In *BMVC*, 2018.

[20] Eduardo Romera, José Manuel Álvarez, Luis Miguel Bergasa, and Roberto Martín Arroyo. Erfnet: Efficient residual factorized convnet for real-time semantic segmentation. *IEEE Transactions on Intelligent Transportation Systems*, 19:263–272, 2018.

[21] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael S. Bernstein, Alexander C. Berg, and Li Fei-Fei. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115:211–252, 2015.

[22] Mark B. Sandler, Andrew G. Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4510–4520, 2018.

[23] Mennatullah Siam, Mostafa Gamal, Moemen Abdel-Razek, Senthil Yogamani, and Martin Jägersand. Rtseg: Real-time semantic segmentation comparative study. *2018 25th IEEE International Conference on Image Processing (ICIP)*, pages 1603–1607, 2018.

[24] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott E. Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–9, 2015.

[25] Christian Szegedy, Sergey Ioffe, and Vincent Vanhoucke. Inception-v4, inception-resnet and the impact of residual connections on learning. In *AAAI*, 2016.

[26] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2818–2826, 2016.

[27] Michael Treml, José Arjona-Medina, Thomas Unterthiner, Rupesh Durgesh, Felix Friedmann, Peter Schuberth, Andreas Mayr, Martin Heusel, Markus Hofmarcher, Michael Widrich, et al. Speeding up semantic segmentation for autonomous driving. In *MLITS, NIPS Workshop*, volume 2, page 7, 2016.

[28] Panqu Wang, Pengfei Chen, Ye Yuan, Ding Liu, Zehua Huang, Xiaodi Hou, and Garrison W. Cottrell. Understanding convolution for semantic segmentation. *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1451–1460, 2018.

[29] Mingxiang Yang, Kun Yu, Chi Zhang, Zhongqiao Li, Kuiyuan Yang, and DeepMotion. Denseaspp for semantic segmentation in street scenes. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3684–3692, 2018.

[30] Changqian Yu, Jingbo Wang, Guo Chao Alex Peng, Changxin Gao, Gang Yu, and Nong Sang. Bisenet: Bilateral segmentation network for real-time semantic segmentation. In *ECCV*, 2018.

[31] Fisher Yu and Vladlen Koltun. Multi-scale context aggregation by dilated convolutions. *CoRR*, abs/1511.07122, 2016.

[32] Xiangyu Zhang, Xinyu Zhou, Mengxiao Lin, and Jian Sun. Shufflenet: An extremely efficient convolutional neural network for mobile devices. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6848–6856, 2018.

[33] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. Pyramid scene parsing network. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6230–6239, 2017.

[34] Hengshuang Zhao, Xiaojuan Qi, Xiaoyong Shen, Jianping Shi, and Jiaya Jia. Icnet for real-time semantic segmentation on high-resolution images. In *ECCV*, 2018.