VNet: A Versatile Network for Efficient Real-Time Semantic Segmentation

Ning Lin[1,2], Hang Lu[1,2], Jingliang Gao[1,2], Shunjie Qiao[3] and Xiaowei Li[1,2]
State Key Laboratory of Computer Architecture, Institute of Computing Technology, Chinese Academy of Sciences[1]
University of Chinese Academy of Sciences[2]
Department of Computer Science, University of Hong Kong[3]
{linning19b, luhang, gaojingliang, lxw}@ict.ac.cn, {qsj1024}@connect.hku.hk

Abstract—Many recent excellent methods for efficient real-time semantic segmentation are of low precision and heavily rely on multiple GPUs for training. In this paper, we rethink the critical factors affecting the accuracy of efficient segmentation models. The previous works usually reduce the input resolution prior to training the parameters of models by cropping or resizing the images. On the contrary, our empirical study shows that the reduced images lose the important content information and details, which are vital to the high precision. However, the previous methods are unable to train the original high-resolution images due to the memory-limited GPUs.

To tackle this problem, we propose a novel versatile network (VNet), which employs reversible mechanism and asymmetric convolution to achieve highly efficient and extremely low memory consumption in backward propagation. In particular, we keep all the detailed spatial information of the input images without cropping or resizing to pursue decent prediction accuracy. It is worth noting that VNet can train multiple 1024×2048 high-resolution images on only one standard GPU card. Under the same conditions, our model achieves a new state-of-the-art result on Cityscapes datasets. Specifically, it can process the 1024×2048 high-resolution inputs at a rate of 37.4 and 15.5 frames per second (fps) on a standard GPU and an edge device, respectively, with only 0.16 million parameters.

I. Introduction

Modern semantic segmentation task, the important applications including medical image analysis, image editing and video surveillance, has made remarkable progress by using deep convolution neural network models (DCNN). The accurate DCNN models largely depend on deeper and wider convolution layers with myriads of parameters and operations, which are not suitable for resource-constrained edge devices like cellphones, drones and self-driving cars.

Efficient real-time semantic segmentation method has recently drawn much attention, as intelligence edge devices not only have faster inference speed requirement for semantic segmentation models but also cannot rely on the cloud services of data center devices. There are two feasible approaches to obtaining the efficient semantic segmentation model. One approach is by designing the efficient models, which fabricates the model architecture from scratch (e.g., ENet [5]). Another less common, but increasingly popular method is network compression, which can obtain the light-weight model (e.g. ICNet [4]) with pruning methods [7] widely used in image classification tasks. However, both of them are challenging to make semantic segmentation model light and fast without sacrificing too much accuracy.

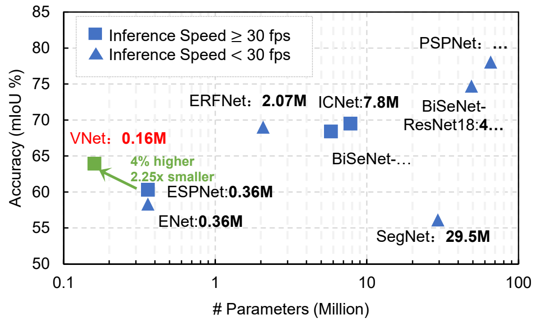


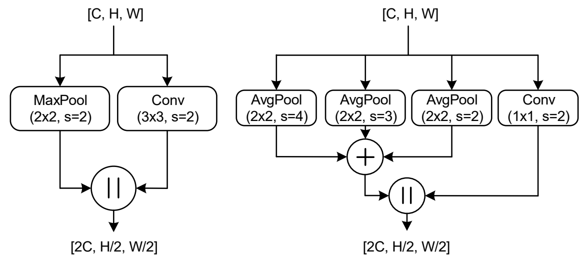
Figure 1 The number of parameters, inference speed (fps) and mIoU (%) performance on Cityscapes test set. The inference speed is tested using 1024×2048 high-resolution images on NVIDIA TITAN Xp.

In this paper, we investigate the central question: whether there are other important factors that can achieve better accuracy and why previous works ignore them. In previous work, it is common to reduce the high-resolution images before training the model using GPUs. For instance, the 1024×2048 resolution Cityscapes dataset [8] is usually cropped in random or resized to the half (512×1024) or even the quarter (256×512) of the original resolution. Whereas our empirical study shows that the cropped images lose content information and the resized images damage the details, and both of content information and details are useful for the accuracy of small models. It is critical for efficient semantic segmentation models to keep the input content information intact. However, due to GPU memory limitation, previous work cannot use the original high-resolution images to train the model. Though it is possible to train the original resolution images on multiple GPUs (one GPU card per im age), some larger models, which have higher accuracy and more parameters than ENet, are even unable to train a single original resolution image on one GPU card. Therefore, larger GPU memory devices have to be adopted, which are more expensive.

The goal of this paper is to find an effective solution for training the high-resolution images on only one GPU. We propose a novel versatile network (VNet), which is mainly composed of Contextual Pyramid Pooling (CPP) modules and

Table 1 The detailed VNet architecture. “Conv” stands for Conv-BN-PReLU. “D” represents the didated Convolution. The input resolution is 1024×2048.

Name	Type	Channel	Output
Stage 1	3x3 Conv (stride =2)	8	512×1024
	3x3 Conv (stride =1)	8	512×1024
	3x3 Conv (stride =1)	8	512×1024
Stage 2	CPP module	32	256×512
	V module × α	32	256×512
Stage 3	CPP module	64	128×256
	V module (D) × β	64	128×256



(a) Downsampler (b) CPP

Figure 2 Depiction of downsampler module and our proposed contextual pyramid pooling module (CPP).

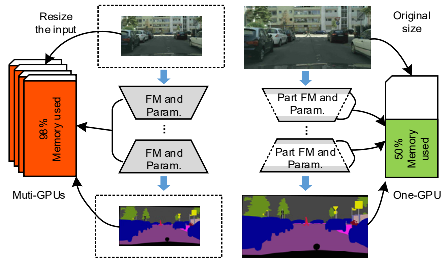
Versatile modules (V modules). In summary, our paper makes the following two contributions:

- *We propose a versatile semantic segmentation network that has smaller parameters, faster inference and decent accuracy compared to previous models.* VNet can process the 1024×2048 high-resolution images at a rate of 37.4 fps and a 15.5 fps on a NVIDIA TITAN Xp and a NVIDIA Jetson Tx2 with only 0.16M parameters. Benefiting from the high-resolution training, the accuracy of VNet is higher than previous small models. In Figure 1, the accuracy of VNet is ~4% higher than ESPNet, while the parameter of VNet is 2.25x smaller than that of ESPNet .
- *VNet can be trained using multiple high-resolution images on only one GPU.* The versatile model of VNet employs reversible mechanism to achieve extremely low memory consumption during the backward propagation. Taking four 1024×2048 images as an example, traditional methods require at least two GPUs with each one 12 GB memory. However, VNet can be trained on one GPU with 9345M memory as shown in Table 3.

II. Background and Related Work

A. Semantic Segmentation

ENet [5] is the first light-weight network structure. It employs ResNet block [10] and fewer convolution filters to illustrate that highly efficient semantic segmentation is feasible on edge devices. Recently, ESPNet [9] introduces an efficient spatial pyramid module composed of point-wise convolution and spatial pyramid of dilated convolution to reduce computation and preserve a large receptive field. Although the two neural networks above are light-weight, the prediction accuracy is significantly sacrificed. Therefore, ERFNet [6] uses more deep architecture that consists of residual connections and factorized convolutions to compensate the loss



(a) (b)

Figure 3 Illustration of the conventional architecture and our proposed method in the training procedure. (a) use the cropped or resized image as input and save all feature maps (FM) on Multi-GPUs. (b) just use the original resolution image as input and save parts of feature maps on one GPU.

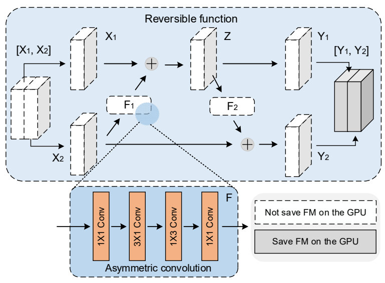


Figure 4 The proposed V Module consists of two components: reversible function and asymmetric convolution. Only the output feature maps (FM) of V module are needed to be saved on GPU.

of precision, and it can process the 360×640 images at a speed of 7 fps on a NVIDIA Jetson Tx1 (embedded GPU) . ICNet [4] is based on the high accuracy method – PSPNet [1]. Originally, the parameters of PSPNet are compressed with the method of pruning filters [7]. Despite decent accuracy achieved in both of these two networks, they are unable to meet the requirement of real-time inference speed on edge devices. Therefore, there is still a big room for the improvement of semantic segmentation efficiency.

B. High-resolution Training

Traditionally, during the training procedure, not only the training samples and parameters of DNNs, but also the feature maps are saved on graphics processing units (GPUs). Due to the limited memory capacity of GPUs (e.g., NVIDIA TITAN Xp), many CNN-based models [1, 3, 4] have to be trained with a mini-batch size of one, or with the resized images. Some methods [11, 12] have proved that the appropriate mini-batch is a benefit for the improvement of network performance. Moreover, the cropp ed images miss part of the image contextual information and the resized images lose the details of the information. Recently, reversible mechanism methods are proposed to achieve competing accuracy of image classification task on cifar-10, cifar-100 and ImageNet dataset. For instance, RevNet [13] is a variant of reversible mechanism where the feature maps of each layer can be reconstructed from that of the next layer. This can save a lot of memory footprint during training process. However, to the best of our knowledge, the reversible mechanism has never been used in the semantic segmentation as a technical means to train the high-resolution images, which largely improves the accuracy performance for light-weight semantic models.

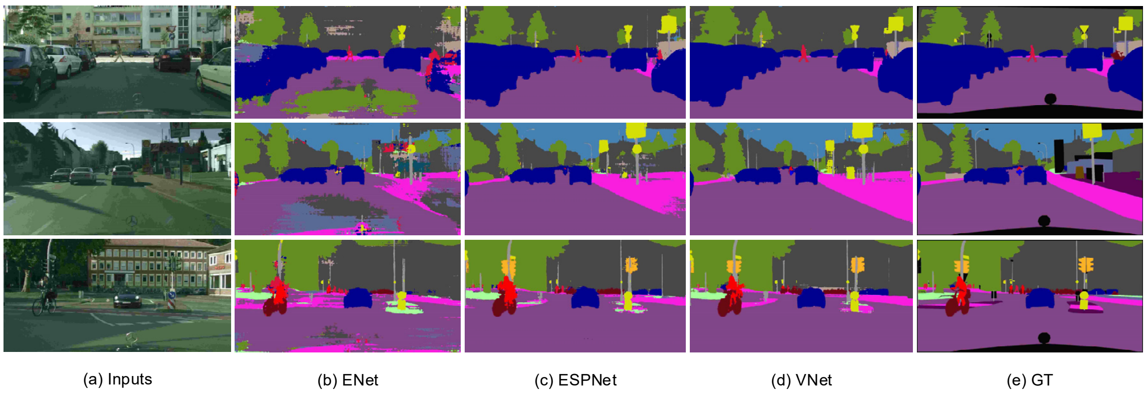


Figure 5 Sample results of ENet, ESPNet and VNet on Cityscapes validation set. From left to right: (a) input images; (b) ENet results; (c) ESPNet results; (d) VNet results; (e) the ground truth (GT).

III. Versatile Network Architecture

In Table 2, VNet consists of three stages: In stage 1, we use the three 3×3 convolution layers to obtain low-level 1/2 resolution feature maps and this stage has only 8 channels. In stage 2, CPP module is used to replace the parameter-consuming downsampler module and get the 1/4 resolution feature maps. On top of CPP, we stack α V modules, which is a versatile module with light weight, small calculation and low memory consumption. In stage 3, the CPP module is also used to get 1/8 resolution feature maps and then β V modules.

1) Contextual Pyramid Pooling (CPP)

Due to the extremely small number of parameters, light-weight networks are always not easy to meet the requirement of high precision. Simply using maxpool modules in light-weight segmentation architecture will lose spatial contextual information, and inevitably reduce the network precision. Thus, the downsampler module is created to compensate for the loss of accuracy in ERFNet [6]. In Figure 2 (a), the downsampler module concatenates the parallel outputs of 3×3 dilated convolution layers [14] with stride 2 and maxpool modules. However, the increased 3×3 convolution layers will result in significant computational complexity and storage overhead.

we propose the CPP module, which concatenates the parallel outputs of point-wise convolutions with stride 2 and the sum of pyramid pooling modules. For the point-wise convolution, it is a 1×1 convolution designed to utilize the channel level contextual information, which decreases the number of parameters and reduces computation overhead. In respect of the pyramid pooling modules, we use it to aggregate more spatial level contextual information and improve the accuracy. Figure 2 (b) presents the details of the CPP module structure.

2) Versatile Module (V Module)

Figure 3 illustrates conventional training paradigm and our proposed method. We propose the versatile module with two key components, reversible function and asymmetric convolution layers. As the name of versatile module implies, the V module not only has small amount of parameters and calculation, but also small memory footprint in the training process. Figure 4 shows the structure of versatile module.

Reversible Function. For reversible function [13], its input feature map can be computed from its output. When we stack more reversible blocks, only the final feature map needs to be cached. Thus in backpropagation procedure, we can calculate the intermediate feature map by the inversion property of Reversible block. The specific calculation formulation for the forward and backward of reverse is:

Forward:	**Reverse:**
$X = [X_1, X_2]$	$Y = [Y_1, Y_2]$
$Z = X_1 + F_1(X_2)$	$Z = Y_1$
$Y_1 = Z$	$X_2 = Y_2 - Z$
$Y_2 = X_2 + Z$	$X_1 = Z - F_1(X_2)$
$Y = [Y_1, Y_2]$	$X = [X_1, X_2]$

Asymmetric Convolution. Asymmetric convolution [15] has been introduced that a n×n convolution kernel can be factorized to n×1 convolution followed by 1×n convolution and this factorization works well on medium layers. These decomposed layers have three benefits: light weight, low computational cost and deeper structures. For light weight and low computational cost, we replace the 3×3 convolution kernel with a 3×1 convolution followed by a 1×3 convolution. So, we can save 33% of parameters and computational cost when the numbers of input and output filters of them are identical.

IV. Evaluation

A. Experimental Setup

We use dataset *Cityscapes* [8], which contains 5000 fine annotated 1024×2048 high-resolution images collected from 50 different urban streets. It is comprised of 2,975 training images, 500 validation images and 1,525 test images. Each pixel of images is annotated to one of 19 classes. Unless and otherwise stated explicitly, all experiments results, such as inference speed and GFLOPs are reported for 1024×2048 resolution RGB images. We use ADAM with 4 images of one batch, betas=(0.9, 0.999), and the weight decay is 0.0001 for optimization. For learning strategy, we choose "poly" learning policy, and the initial rate is multiplied by $(1 - iter/maxiter)^{power}$. The initial learning rate is 0.0005 and power is set to 0.9, together with the maximum epoch number 300 for Cityscapes.

Table 2 Evaluation results on the Cityscapes test set. The inference speed is evaluated on the NVIDIA TITAN Xp GPU. "

Network	Param. (M)	Speed (fps)	mIoU (%)
PSPNet [1]	65.7	1.9	**78.4**
BiSeNet-ResNet18 [2]	49	27	74.7
SegNet [3]	29.5	5	56.1
ICNet [4]	7.8	30.3	69.5
BiSeNet-Xception [2]	5.8	30	68.4
ERFNet [6]	2.07	17.9	68.0
ENet [5]	0.36	22	58.3
ESPNet [9]	0.36	**47.6**	60.3
VNet ($\alpha = 1, \beta = 2$)	**0.16**	37.4	63.93

Table 3 The effectiveness of V module. "TMem." indicates the GPU memory usage of training data. "TVMem." is the GPU memory usage of training and validation data. "False" stands for the network cannot be trained due to out of memory.

Network	Batch Size	TMem. (MB)	TVMem. (MB)	Train. Cost (GPU days)
PSPNet	1	>12189	>12189	False
ESPNet	4	7431	>12189	False
ENet	4	8151	>12189	False
VNet	4	**5021**	**9345**	**2.9**

B. *Accuracy and Speed*

We report the performance comparison of our proposed VNet ($\alpha = 1, \beta = 2$) and other state-of-the-art methods on Cityscapes test set in Table 2. The inference speed is measured on the original 1024×2048 high-resolution images, and we do not employ any trick, such cropping the image to smaller size. Compared with small memory footprint model, e.g. SegNet, ENet, ICNet, ESPNet and BiSeNet, our proposed VNet can achieve a relatively high mIoU and much faster speed. For example, our VNet is even 15.4 fps faster than recently state-of-the-art ENet, and the accuracy of VNet is 5.63% and 3.63% higher than ENet and ESPNet, respectively. More notably, the parameter of VNet is only 0.18M. Besides, the visual results are shown in Figure 5, and we can find that the results of VNet is closer to the ground truth compared to ENet and ESPNet.

C. *Performance Analysis on Edge Device*

Figure 6 compares the inference speed of VNet with ERFNet, ENe t and ESPNet. Our model has the fastest inference speed under the condition of same GPU frequency, because it has smaller channel number than the frameworks transferred from the ImageNet classification networks. For example, when we set GPU frequency to 1300, VNet is 6.5 fps, 6.3 fps, 11.9 fps faster than ESPNet, ENet and ERFNet, respectively. Thus, VNet is currently the fastest segmentation model, and it is able to meet the requirement of real-time inference for edge devices.

D. *Effectiveness of Versatile Module*

From Table 3, we can find that the PSPNet's memory usage exceeds other networks, although the batch size equals one. Compared with ESPNet and ENet, our proposed VNet has the smallest memory overhead and can save 2410MB and 3130MB for training dataset, respectively. If the accuracy of the validation dataset needs to be verified during the training procedure, only VNet can be trained for 1024×2048 images. In addition, VNet takes 2.9 GPU days to train four high-resolution images, while other light-weight semantic models cannot train them due to the out of GPU memory. Moreover, the high-accuracy semantic model even cannot train one high-resolution image on a NVIDIA TITAN Xp GPU. Overall, our proposed V module can train the VNet on only one GPU.

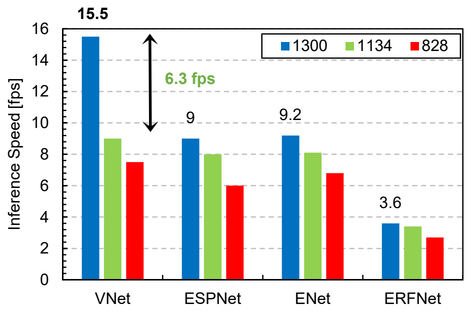


Figure 6 The inference speeds of VNet, ENet, ERFNet and ESPNet measured on NVIDIA Jetson Tx2 with three GPU frequencies (MHz): 1300,1134 and 828.

V. Conclusion

We propose a versatile network (VNet) based on V module and CPP module. VNet has the ability to train many high-resolution images on only one standard GPU (12GB) without the need of images cropping or resizing. The experimental results show that VNet can process the high-resolution images at a rate of 37.4 and 15.5 fps on a standard GPU and an edge device, respectively, with only 0.16 million parameters.

Acknowledgement

This work is supported in part by the national Natural Science Foundation of China (NSFC) under grant No. (61432017, 61602442, 61876173), and in part by the National Key Research and Development Project under grant No. 2018AAA0102700. Corresponding authors are Hang Lu and Xiaowei Li.

References

[1] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, "Pyramid scene parsing network," in CVPR, 2017.
[2] C. Yu, J. Wang, C. Peng, C. Gao, G. Yu, and N. Sang, "Bisenet: Bilateral segmentation network for real-time semantic segmentation," in ECCV, 2018.
[3] V. Badrinarayanan, A. Kendall, and R. Cipolla, "Segnet: A deep convolutional encoder-decoder architecture for image segmentation," in TPAMI, 2017.
[4] H. Zhao, X. Qi, X. Shen, J. Shi, and J. Jia, "Icnet for real-time semantic segmentation on high-resolution images," in ECCV, 2018.
[5] A. Paszke, A. Chaurasia, S. Kim, and E. Culurciello, "Enet: A deep neural network architecture for real-time semantic segmentation," arXiv preprint arXiv:1606.02147, 2016.
[6] E. Romera, J. M. Alvarez, L. M. Bergasa, and R. Arroyo, "ERFNet: Efficient residual factorized convnet for real-time semantic segmentation," IEEE Transactions on Intelligent Transportation Systems 2018.
[7] H. Li, A. Kadav, I. Durdanovic, H. Samet, and H. P. Graf, "Pruning filters for efficient convnets," arXiv preprint arXiv:1608.08710, 2016.
[8] M. Cordts et al., "The cityscapes dataset for semantic urban scene understanding," in CVPR, 2016.
[9] S. Mehta, M. Rastegari, A. Caspi, L. Shapiro, and H. Hajishirzi, "ESPNet: Efficient Spatial Pyramid of Dilated Convolutions for Semantic Segmentation," in ECCV, 2018.
[10] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in CVPR, 2016.
[11] P. Goyal et al., "Accurate, large minibatch sgd: Training imagenet in 1 hour," arXiv preprint arXiv:1706.02677, 2017.
[12] Y. You, I. Gitman, and B. Ginsburg, "Scaling SGD Batch Size to 32K for ImageNet Training," in CVPR, 2017.
[13] A. N. Gomez, M. Ren, R. Urtasun, and R. B. Grosse, "The reversible residual network: Backpropagation without storing activations," in NIPS, 2017.
[14] F. Yu and V. Koltun, "Multi-scale context aggregation by dilated convolutions," in ICLR, 2015.
[15] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam, "Rethinking atrous convolution for semantic image segmentation," arXiv preprint arXiv:1706.05587, 2017.