2020 IEEE International Conference on Robotics and Automation (ICRA)
31 May - 31 August, 2020. Paris, France

FarSee-Net: Real-Time Semantic Segmentation by Efficient Multi-scale Context Aggregation and Feature Space Super-resolution

Zhanpeng Zhang[1], Kaipeng Zhang[2]*
[1]SenseTime Group Limited [2]The University of Tokyo

Abstract— **Real-time semantic segmentation is desirable in many robotic applications with limited computation resources. One challenge of semantic segmentation is to deal with the object scale variations and leverage the context. How to perform multi-scale context aggregation within limited computation budget is important. In this paper, firstly, we introduce a novel and efficient module called Cascaded Factorized Atrous Spatial Pyramid Pooling (CF-ASPP). It is a lightweight cascaded structure for Convolutional Neural Networks (CNNs) to efficiently leverage context information. On the other hand, for runtime efficiency, state-of-the-art methods will quickly decrease the spatial size of the inputs or feature maps in the early network stages. The final high-resolution result is usually obtained by non-parametric up-sampling operation (e.g. bilinear interpolation). Differently, we rethink this pipeline and treat it as a super-resolution process. We use optimized super-resolution operation in the up-sampling step and improve the accuracy, especially in sub-sampled input image scenario for real-time applications. By fusing the above two improvements, our methods provide better latency-accuracy trade-off than the other state-of-the-art methods. In particular, we achieve 68.4% mIoU at 84 fps on the Cityscapes test set with a single Nivida Titan X (Maxwell) GPU card. The proposed module can be plugged into any feature extraction CNN and benefits from the CNN structure development.**

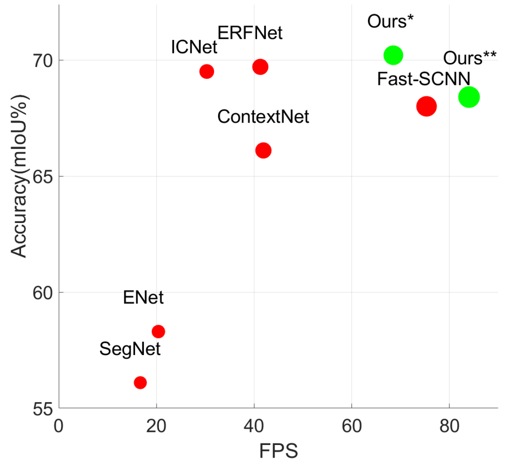

Fig. 1. Illustration of the inference speed (fps) and accuracy (mIoU) on the Cityscapes test set. Bigger dot means faster inference process. The inference speed is evaluated on a single Nvidia Titan X (Maxwell) GPU, for the current real-time semantic segmentation methods, including SegNet [34], ICNet [39], ENet [24], ERFNet [28] and Fast-SCNN [26]. 'Ours*' and 'Ours**' denotes our approach with different input image resolutions (*i.e.*., 512 × 1024 and 512 × 768, respectively). The results of other methods come from the related literature.

I. INTRODUCTION

Semantic segmentation refers to the problem of estimating the class label for all pixels given the input image. It is a fundamental task for many computer vision applications. Thanks to the remarkable development of deep convolutional networks [10], [27], [8], [15], [12], there is substantial progress for this task. However, many algorithms require sophisticated models that come with large memory and computational cost. This is challenging for robotics applications where real-time performance is required and the computation resource is usually limited. A better latency-accuracy trade-off is worth investigation.

Current deep learning based semantic segmentation algorithm usually contains a front-end network and a back-end network. A backbone network pre-trained with large-scale image classification task is usually employed as the front-end for feature extraction. The back-end network, on the other hand, usually contains a module for multi-scale context aggregation (e.g., ASPP [2], [4], PPM [40], RefineNet [17]), as well as some sequential convolution layers to generate the final dense class probability map. Since the front-end network needs to extract high level semantic information, it usually requires deep networks with large numbers of parameters. To accelerate the algorithm, currently real-time segmentation methods employ two and multiple branch architecture [33], [39], [37] for the front-end. In particular, branch with deeper network is fed with a lower resolution version of the image while branch with shallower network deals with the higher resolution version. On the other hand, some approaches achieve real-time performance by designing thin network structure [26], [28]. But accuracy of the segmentation result usually degrades substantially with real-time acceleration techniques. Better trade-off between accuracy and speed stills requires investigation. Fig. 1 shows an overview about the speed and accuracy of some current real-time semantic segmentation methods.

Since the images may contain a same class in different scales, how to leverage the context information is important for thin and small objects. Multi-scale context aggregation with deep network plays an important role for semantic segmentation. In fact, the back-end network usually needs to accomplish this. Current context aggregation modules, such as Atrous Spatial Pyramid Pooling (ASPP) [2], [4],

*Corresponding author

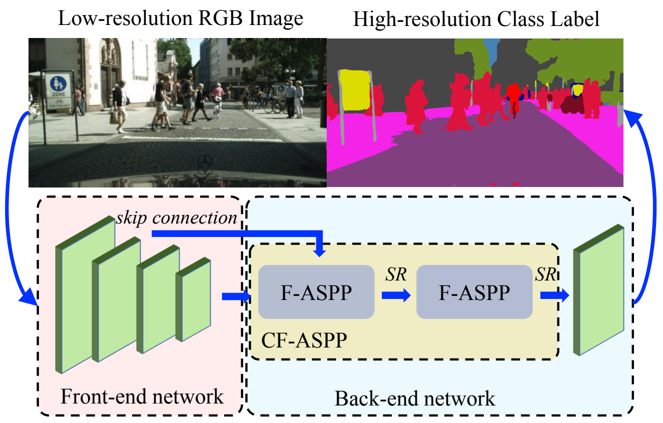

Fig. 2. We divide the semantic segmentation network into front-end and back-end. For the front-end, we can use a generic image representation backbone network. For the back-end, we design a cascaded factorized ASPP (CF-ASPP) module that can perform multi-scale context aggregation efficiently, and combine it with feature space super-resolution (SR) learning, so as the boost the process of generating high-resolution output label map from low-resolution input. The proposed back-end network is easy to implement and can be plugged into other front-end networks.

have shown the effectiveness and rank among the state-of-the-art methods. However, these modules usually work in deep layers of the network where the number of feature map channel is large. In this case, even a convolutional layer with kernel size 3 consumes substantial computation. In this work, we designed a factorized ASPP module for multi-scale context fusion. Moreover, since this module is light-weight, we can repeatedly employ it for stronger context fusion without obvious increase of computation. Here we named this module cascaded factorized ASPP (CF-ASPP).

Another issue of the back-end network for semantic segmentation is that the spatial size of the feature maps decreases substantially after the front-end. In addition, many approaches improve the speed by using lower-resolution images as input directly. This makes it more challenging for the back-end network. Many current back-end networks simply perform up-sample by parameter-free operations such as bilinear interpolation, on the feature map to obtain the result with original resolution. In this case, it is hard to recover the details for the final segmentation result. Mazzini [22] propose Guided Up-sampling Network (GUN) to better recover from the low-resolution feature map. In particular, the network learns to predict a high-resolution guidance offset table of offsets vectors that steer sampling towards the correct semantic class. In this work, we solve this problem from another perspective. In the training process, we use lower-resolution input image (*e.g.*, 512×1024 for Cityscapes dataset instead of 1024×2048) but keeps the high-resolution ground truth for supervision. In fact, this is the problem of recovering high-resolution output from low-resolution input. This problem has been widely studied under the topic of super-resolution [7]. We can leverage the development of super-resolution to solve this problem. This process can be performed by current highly optimized operators, which we fuse with the proposed CF-ASPP.

To this end, in this work, we propose a new back-end network for semantic segmentation. In particular, we propose the CF-ASPP module to efficiently perform multi-scale context aggregation and employ a feature map super-resolution step that can better recover high-resolution result from low-resolution input. Our whole pipeline is shown in Fig. 2. To summarize, our contributions are three-fold:

1) We propose a cascaded factorized ASPP (CF-ASPP) module that is faster and provides more accurate result than the original ASPP [4], [2].
2) We treat the problem of recovering high-resolution segmentation result from low-resolution input as a super-resolution process. The experimental result shows that given lower resolution input image, the performance degrade of our method is lower than other methods. This helps to accelerate the algorithm while keeping reasonable accuracy.
3) We provide a new back-end network for semantic segmentation. The proposed network provides better latency-accuracy trade-off than current state-of-the-art real-time semantic segmentation methods [26], [39], [37]. In addition, the proposed back-end network is easy to implement and can be directly combined with other existing feature extraction network. That means our approach can benefit from the advance of common feature extraction network which is another important computer vision topic [29], [38], [23].

II. RELATED WORK

A. *Quality Driven Semantic Segmentation*

Since the introduction of FCN [19], there are extensive research works on deep learning for semantic segmentation. How to model the spatial relationship between pixels or leverage the context for inference is the main concern of current methods. For example, the skip-connection [10] is widely employed to fuse the high level semantic feature and low level spatial cues; ASPP [2], [3], [4] and PPM [40] is the component applied on the extracted feature from front-end to fuse context of multiple scales; CRF [1] and MRF [18] is used to model the spatial relationship between pixels or regions. Recently, HRNet [33] is proposed for semantic segmentation. The whole network maintains multiple branches with different resolutions of the image. The representation of different resolutions is densely fused. These methods achieve high quality results at the cost of heavy computation.

B. *Real-Time Semantic Segmentation*

Recently, real time semantic segmentation attracts more and more researchers. Research works among this line aims to improve the model inference time while keeping decent accuracy. ENet [24] is one of the pioneers in this line. The authors design a light-weight network structure and the input image is heavily down-sampled at the early stage of the network to reduce processing time. ERFNet [28] takes another approach, where the network contains multiple factorized convolution blocks. In particular, the combinations of $n \times 1$ and $1 \times n$ convolution is used to reduce the computation of original $n \times n$ convolution. Similar structure is also used in [35]. Two-branch and multi-branch network design is also proposed. ICNet [39], ContextNet [25], BiSeNet [37] and Fast-SCNN [26] learn global image context with lower resolution input with a deeper network branch, which is then combined with the feature from a shallower branch that describes the boundary information using higher resolution input, or feature map obtained from the early stage of the deeper branch. Reusing feature is another direction to reduce the computation cost. Most recently, Li *et al.* [16] design a network that contains multiple sub-modules. Features from early module is reused for the following module. Different from these approaches, we treat the whole segmentation network as a combination of front-end and back-end, and our work does not need specific backbone/feature-extraction network design for the front-end network. We focus on how to design an efficient back-end network. In fact, the module proposed in this work can be plugged into other feature-extraction networks, such as MobileNet [29] and ShuffleNet [38].

C. General Deep Network Acceleration

In addition to the real time network for semantic segmentation research, there are extensive studies on general deep network acceleration for applications such as object detection and image classification. For example, network quantization [14] is applied for convolution parameters for better inference speed than the floating point computation. On the other hand, network compression technique either uses a pruning operation [20] to reduce the network structure or use a bigger network to guide the training of a smaller network [11]. Recently, many light-weight backbone networks, such as MobileNet [29] and ShuffleNet [38], are also proposed for efficient feature extraction with light-weight building blocks. Note that these research works are orthogonal to our work. The proposed network can benefit from the advance of these directions.

III. Approach

A. Approach Overview

Given an input RGB image $I \in \mathbb{R}^{H' \times W' \times 3}$, deep learning based semantic segmentation method usually consists of successive convolution layers and the output is $L \in \mathbb{R}^{H \times W \times N}$, where N denotes the number of class for the segmentation task, and L is the class probability for each pixel. H and W denotes the height and width of the image, respectively. Differently, in our formulation we allow low-resolution input and high-resolution output to accelerate the process so we have $H' \leqslant H$ and $W' \leqslant W$. The network contains the front-end and back-end as shown in Fig. 2. The front-end is for feature extraction, where we can employ existing deep models, such as VGG [32], ResNet [10], and MobileNet [29]. We can extract the feature from one or more intermediate layers from these networks. Since this part is not the main focus for this paper, we do not explain it in detail. For the back-end network, the input feature map is extracted from the front-end and the output is the probability map L. Here we employ the proposed cascaded factorized ASPP (see Sec. III-B) that is fused with feature space super-resolution (see Sec. III-C).

B. Cascaded Factorized ASPP

Due to the variations of the object size, how to capture and fuse image feature in different scales is important for semantic segmentation. Atrous convolution [3] has been employed extensively for semantic segmentation. Different from conventional convolution filters, atrous convolution filters can be treated as inserting $r - 1$ zeros between two neighboring filter values along each spatial dimension, where r is the atrous rate. We can see that atrous convolution allow us to enlarge the filter's *field-of-view* without increasing the number of model parameters and computation cost. On the other hand, inspired by spatial pyramid pooling method of [9], Chen *et al.* [2] propose atrous spatial pyramid pooling (ASPP) for semantic segmentation. In particular, in ASPP, there are multiple parallel 3×3 atrous convolutions with different atrous rates. Those parallel atrous convolutions are applied on top of the feature map from a feature extraction network. We can see that the multiple atrous rates can help the network to capture multi-scale context information. However, the ASPP is applied on the feature map obtained from a deep network. The channel number of the feature map is usually large (*e.g.*., 512 for ResNet 18 [10]). Even with the kernel size of 3×3, ASPP still consumes substantial computation cost. To reduce this problem and further increase the effectiveness of ASPP, we do the following modifications.

Firstly, we decompose the 3×3 atrous convolution layer into two layers: 1) point-wise convolution layer (*i.e.*, kernel size is 1×1) that linearly combines the input channels and reduce the dimension of the output channel dimension; This layer is to perform channel-wise information interaction; 2) depth-wise and atrous convolution with the same kernel size (*i.e.*, 3×3) and atrous rate as the original atrous convolution. The depth-wise convolution here is to reduce the computation cost. This layer is to enable the model to capture the feature of the neighboring area. Similar factorization is also used in [29], [10]. Differently, depth-wise convolution is employed here instead of conventional convolution. To this end, we have the factorized ASPP (F-ASPP) module. Let F_i and F_o be the input and output channel number, we can see that for the original 3×3 atrous convolution, the computation complexity for a feature map $M_i \in \mathbb{R}^{h \times w \times F_i}$ is $h \times w \times 3 \times 3 \times F_i \times F_o$, while the computation complexity of the factorized one is $h \times w \times F_i \times F_o$+$h \times w \times 3 \times 3 \times F_o$. Given that F_i and F_o is 512, and 256 respectively in our implementation, we can see that the computation cost is about 8.8x reduced.

Secondly, instead of applying the ASPP module only once, we cascade two factorized ASPPs in our network. The motivation here is that we can perform extensive multi-scale context aggregation in this case. On the other hand, the factorize ASPP already reduce the computation a lot compared to the original ASPP and we use less channels in the second F-ASPP, cascading this component does not bring

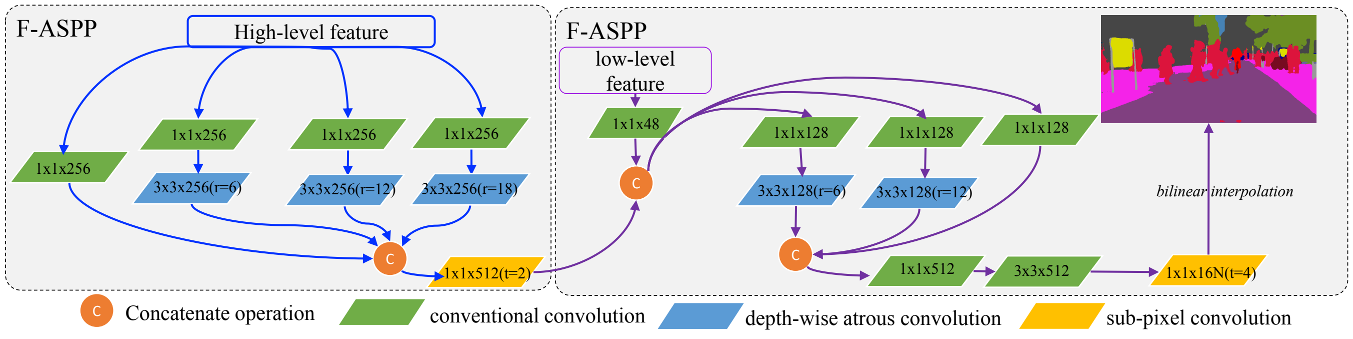

Fig. 3. Illustration of the proposed back-end network for semantic segmentation. The high-level and low-level feature is obtained from the deeper and shallower layers of the front-end network, respectively, to capture the image description in different levels. The numbers on the convolution layer denotes the according parameters (*e.g.*., '$3 \times 3 \times 256(r = 18)$' means that the kernel size is 3×3 with output channel 256. r is the atrous rate and t is the up-sampling factor.) N is the class number of the segmentation task. For example, the last sub-pixel convolution layer with up-sampling factor $t = 4$ will generate a feature map with N channels. The spatial size is 4x larger than that of the previous feature map. Then each channel describes the probability of the according class. The class label can be obtained by an 'argmax' operator. Each convolution layer is followed by batch normalization [12] and Relu nonlinear activation [15].

much additional computation cost but improve the accuracy significantly (see our experiment). The detailed structure of the CF-ASPP is illustrated in Fig. 3.

C. Feature Space Super-resolution

In addition to kernel factorization, a simple way to reduce the computation cost is to reduce the input image resolution and up-sample the low-resolution result to the high-resolution one. However, recovering high-resolution result from low-resolution is challenging. Another issue is that after several stages of the front-end network, the feature map spatial size also decreases a lot (*e.g.*., 1/8, 1/16 of the input size). This is mainly because 1) the front-end network contains sub-sampling layers (*e.g.*., max-pooling) or layers with stride lager than 1; 2) the channel number in deep layer is usually large (*e.g.*, 512, 1024)) and keeping high-resolution feature map in deep layers will bring too much computation cost. So we need to generate a high-resolution segmentation map using the low-resolution input. A parameter-free interpolation operation (*e.g.*., bilinear up-sampling) may not be a good solution for this.

In this work, we treat this problem as a super-resolution process in the feature space. Firstly, in the training process, we use down-sampled RGB image as input, and the original high-resolution class label map as the ground truth. Secondly, for the network structure design, we gradually up-sample the feature map in the back-end network. The up-sample operation is performed by sub-pixel convolution [30] that is widely used for the image super-resolution task. Here we explain how we apply sub-pixel convolution in our work. Given the input feature map $M_i \in \mathbb{R}^{h \times w \times F_i}$ and the up-sampling factor t, we need to generate an output feature map $M_o \in \mathbb{R}^{tw \times th \times F_i}$. Firstly, we apply a pixel-wise convolution layer to M_i and generate a feature map $M_m \in \mathbb{R}^{h \times w \times F_i t^2}$. Secondly, we apply a periodic shuffling operator that rearranges the elements of the feature map $M_m \in \mathbb{R}^{h \times w \times F_i t^2}$ to a feature map $M_o \in \mathbb{R}^{tw \times th \times F_i}$. Detailed definition of periodic shuffling can be found in [30]. It has been explained in [31] that compared to the deconvolution operator with the same computation budget, sub-pixel convolution has more representation power. Detailed explanation for this is out of the scope of this paper.

To this end, we have the sub-pixel convolution for feature map super-resolution. Here we apply this operation twice in the proposed CF-ASPP. Fig. 3 illustrates the network details.

IV. Experiments

A. Evaluation Dataset

We perform the evaluation on the cityscapes dataset [5] since it is a popular and standard benchmark for semantic segmentation research. The high resolution input (i.e., 1024×2048 RGB image) makes it challenging for real time segmentation algorithms. We follow the official evaluation protocol of this benchmark. In particular, it contains an image collection with fine annotations of 30 common classes in urban street scenes (e.g., road, car, sky, person). The collection is divided into training, validation, and test split, with 2,975, 500, and 1,525 images, respectively. Images in this dataset are illustrated in Fig. 4. Among the 30 annotated classes, 19 classes are used for evaluation. According to the dataset, the official accuracy criterion is the mean intersection-over-union (IoU) metric. In particular, we have $IoU = TP/(TP + FP + FN)$, where TP, FP, and FN denotes the size of true positive, false positive, and false negative, respectively. For the inference speed criterion, we use the direct metric–inference time. We do not use FLOPs because it is an indirect metric. It is usually not equivalent to the direct metric because it cannot reflect some hardware related factors such as memory access cost and degree of parallelism. This has been discussed in [21].

B. Implementation Details

For the software platform, the proposed algorithm is implemented by Pytorch 1.1 with CuDNN v7.0 (no TensorRT or other inference optimizers to avoid external influence). For the hardware, the program runs on a PC with Nvidia Titan X (Maxwell) GPU. This is to keep the same platform with

TABLE I
ABLATION STUDY FOR THE PROPOSED METHOD ON THE CITYSCAPES VALIDATION SET WITH NIVIDA TITAN X (MAXWELL)

Test ID	Method ID	input size	mIoU (%)	running time (ms)
1	1	1024 × 2048	72.01	42.0
2	4	1024 × 2048	73.52	46.1
3	1	768 × 1536	70.40	25.4
4	4	768 × 1536	72.28	26.8
5	1	512 × 1024	68.33	15.8
6	2	512 × 1024	68.61	13.3
7	3	512 × 1024	70.05	13.5
8	4	512 × 1024	70.29	14.6
9	4	512 × 768	69.84	11.9

TABLE II
COMPARISON WITH OTHER METHODS ON CITYSCAPES TEST SET (THE RUNNING TIME IS MEASURED WITH NIVIDA TITAN X (MAXWELL)[1]

Method	mIoU (%)	running time (fps)
SegNet [34]	56.1	16.7
ENet [24]	58.3	20.4
ICNet [39]	69.5	30.3
ERFNet [28]	69.7	41.7
ContextNet [25]	66.1	41.9
BiSeNet [37]	68.4	105.8*
GUN [22]	70.4	33.3*
Fast-SCNN [26]	68.0	75.3
ESPNetV2 [23]	66.4	-
ThunderNet [36]	64.0	96.2*
LEDNet [35]	70.6	71.0**
Ours (512 × 1024 input)	70.2	68.5
Ours (512 × 768 input)	68.4	84.0

many other current methods [39], [26], [25] for fair comparison. When we evaluate the running time performance, we use a single CPU thread and a single GPU and we measure the average model inference time one the 500 cityscapes validation images.

In the training process, for the front-end network, we use ResNet-18 [10] pre-trained with the ImageNet classification task [6]. For the high-level feature from the front-end network, we use the feature from the last convolution layer before the global average pooling layer [10]. For the low-level feature, we use the feature from the layer 'conv3_x' of the network. To train the whole network, we use the 2,975 training images with fine annotations from cityscapes dataset. Batch normalization [12] and Relu activation [15] is used after every convolution layer. Stochastic gradient decent (SGD) with momentum 0.9 and batch-size 16 is used in the training process. The initial learning rate is set 0.1 and decayed by a factor of 0.9 every 50 epochs. All the experiments of the proposed method are trained for 400 epochs. We also perform extensive data augmentation as other current methods, including random flipping, rotation, color channel noise, and resizing. Similar to other methods, we use the cross-entropy loss.

C. Ablation Study on Network Structure

To evaluate the proposed method, firstly we perform ablation study using the following baseline methods:

1) Front-end network with ResNet-18 and back-end network with original ASPP and the decoder from DeeplabV3+ [4].
2) Front-end network with ResNet-18. The back-end network contains one F-ASPP (without feature space resolution) and the decoder from DeeplabV3+ [4].
3) Front-end network with ResNet-18. The back-end network contains CF-ASPP (without feature space resolution) and no decoder from DeeplabV3+ [4].
4) The full proposed method.

From Tab. I, we can have the following observations by comparing different test cases. By comparing Test 1, 2, 3 and 4, we can see that the accuracy of proposed back-end network is better than the original one from DeeplabV3+ [4]. In addition, the accuracy decreases when the input resolution decreases. But the accuracy degradation is smaller for the proposed method. This demonstrates the effectiveness of the proposed feature space super-resolution approach. Although our running time consumption is larger than baseline method 1 for high resolution input, we can see that the situation reverses when given lower resolution input (see Test 5 and 8). This is mainly because the additional computation of the proposed method is caused by the increased number of layers. Given high resolution input, the feature map spatial size is large, memory transfer between layers consumes a large part. Given lower resolution input, the situation reverses. This also shows that our method fits the cases with smaller input size.

By comparing Test 5, 6, 7, and 8, we can see that: 1) By incorporating F-ASPP, the performance can be improved a little, and the improvement can be larger if we employ CF-ASPP. This demonstrates the effectiveness of F-ASPP and cascading F-ASPP. 2) F-ASPP is faster than the original ASPP (Test 5 and 6). 3) With CF-ASPP, both the running time efficiency and accuracy is better than the original one (Test 5 and 7). 4) By feature space super-resolution, we can obtain better result (Test 7 and 8).

D. Comparison with Other Methods

We also compare the proposed the method with other state-of-the-art real-time semantic segmentation algorithms including: ENet [24], ICNet [39], ContextNet [25], ESPNetV2 [23], GUN [22], Fast-SCNN [26], ERFNet [28], BiSeNet [37], LEDNet [35] and ThunderNet [36]. The results are listed in Tab. II. The results of other method are from related literatures. We can see that we almost achieve the fastest speed compared to other methods. Although some methods have faster FPS records, but they are evaluated on other GPUs that have higher performance. For example, a same network of [26] runs 75.3 fps and 123.5 fps on Titan X (Maxwell) and Titan Xp, respectively [26] (*i.e.*., 68% faster).

[1]For the running time, '*' means that the time is measured using Nvidia Titan Xp and '**' means Nvidia GTX1080Ti. It is benchmarked in [13] and [26] that these two GPUs are about 60% faster than Nvidia Titan X (Maxwell) for the network forward time.

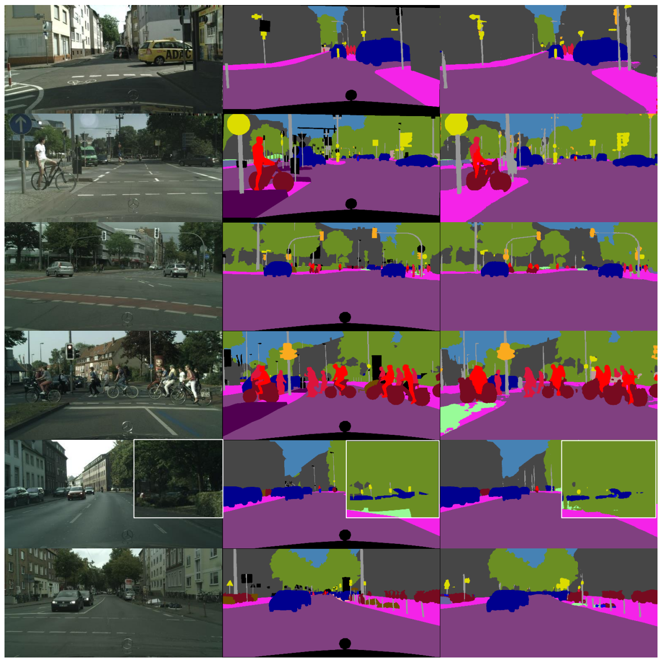

Fig. 4. Illustration of our result on the cityscapes validation set.

For the accuracy, our method is also among the leading competitors. This shows that our method achieves the state-of-the-art latency-accuracy trade-off.

In addition, we test the inference time of the proposed network in a Jetson AGX Xavier embedded system and obtain 21.2fps using the input image size of 512 × 768. Here we simply test it with an unoptimized implementation (we only use the python interface of Pytorch and do not use the TensorRT library or NVDLA engine for inference acceleration), thus the running time can be further reduced by other engineering adaptation.

E. *Qualitative Analysis*

Fig. 1 shows some of our result on the cityscapes validation set. We can see that the proposed method can deal with some of the small and thin objects. From the 5th row, we can see that even some cars are occluded, they can still be identified to some extent. This figure is best viewed with zoomed in.

V. Conclusion

This paper proposes a back-end network component for real time semantic segmentation. On one hand, we propose a cascaded factorized ASPP module for efficient multi-scale context aggregation. On the other hand, to allow the model to use down-sampled lower-resolution image as input and generate high-resolution segmentation output, we employ the super-resolution approach in the feature space. We conduct extensive experiments and the effectiveness of these two aspects have been demonstrated. The latency-accuracy trade-off outperforms many existing state-of-the-art methods. Future research direction can be how to combine the proposed network with other network acceleration techniques such as network compression and quantization.

References

[1] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L. Yuille. Semantic image segmentation with deep convolutional nets and fully connected CRFs. In *International Conference on Learning Representations*, 2015.

[2] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L. Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(4):834–848, 2016.

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

[4] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *European Conference on Computer Vision*, pages 808–818, 2018.

[5] 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. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3213–3223, 2016.

[6] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *IEEE conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009.

[7] Chao Dong, Chen Change Loy, Kaiming He, and Xiaoou Tang. Image super-resolution using deep convolutional networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(2):295–307, 2016.

[8] Kaiming He, Georgia Gkioxari, Piotr Dollr, and Ross B. Girshick. Mask R-CNN. In *International Conference on Computer Vision*, pages 2961–2969, 2017.

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. In *European Conference on Computer Vision*, pages 1094–1916, 2014.

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

[11] Geoffrey E. Hinton, Oriol Vinyals, and Jeffrey Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

[12] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning*, pages 448–456, 2015.

[13] Justin Johnson. cnn-benchmarks. `https://github.com/jcjohnson/cnn-benchmarks`.

[14] Sangil Jung, Changyong Son, Seohyung Lee, Jinwoo Son, Jae-Joon Han, Youngjun Kwak, Sung Ju Hwang, and Changkyu Choi. Learning to quantize deep networks by optimizing quantization intervals with task loss. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 4350–4359, 2019.

[15] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, pages 1097–1105, 2012.

[16] Hanchao Li, Pengfei Xiong, Haoqiang Fan, and Jian Sun. DFANet: Deep feature aggregation for real-time semantic segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 9522–9531, 2019.

[17] Guosheng Lin, Anton Milan, Chunhua Shen, and Ian D. Reid. RefineNet: Multi-path refinement networks for high-resolution semantic segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1925–1934, 2016.

[18] Ziwei Liu, Xiaoxiao Li, Ping Luo, Chen Change Loy, and Xiaoou Tang. Semantic image segmentation via deep parsing network. In *International Conference on Computer Vision*, pages 1377–1385, 2015.

[19] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3431–3440, 2015.

[20] Jian-Hao Luo, Jianxin Wu, and Weiyao Lin. ThiNet: A filter level pruning method for deep neural network compression. In *International Conference on Computer Vision*, pages 5058–5066, 2017.

[21] Ningning Ma, Xiangyu Zhang, Hai-Tao Zheng, and Jian Sun. Shufflenet V2: practical guidelines for efficient CNN architecture design. *CoRR*, abs/1807.11164, 2018.

[22] Davide Mazzini. Guided upsampling network for real-time semantic segmentation. In *British Machine Vision Conference*, 2018.

[23] Sachin Mehta, Mohammad Rastegari, Linda Shapiro, and Hannaneh Hajishirzi. ESPNetv2: A light-weight, power efficient, and general purpose convolutional neural network. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 9190–9200, 2019.

[24] Adam Paszke, Abhishek Chaurasia, Sangpil Kim, and Eugenio Culurciello. ENet: A deep neural network architecture for real-time semantic segmentation. *CoRR*, abs/1606.02147, 2016.

[25] Rudra P. K. Poudel, Ujwal Bonde, Stephan Liwicki, and Christopher Zach. ContextNet: Exploring context and detail for semantic segmentation in real-time. In *British Machine Vision Conference*, 2018.

[26] Rudra P K Poudel, Stephan Liwicki, and Roberto Cipolla. Fast-SCNN: Fast semantic segmentation network. In *British Machine Vision Conference*, 2019.

[27] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems*, pages 91–99, 2015.

[28] Eduardo Romera, Jose M. Alvarez, Luis Miguel Bergasa, and Roberto Arroyo. ERFNet: Efficient residual factorized convnet for real-time semantic segmentation. *IEEE Transactions on Intelligent Transportation Systems*, 19(1):263–272, 2018.

[29] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. MobileNetV2: Inverted residuals and linear bottlenecks. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 4510–4520, 2018.

[30] W. Shi, J. Caballero, F. Huszr, J. Totz, A. P. Aitken, R. Bishop, D. Rueckert, and Z. Wang. Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1874–1883, 2016.

[31] Wenzhe Shi, Jose Caballero, Lucas Theis, Ferenc Huszar, Andrew P. Aitken, Christian Ledig, and Zehan Wang. Is the deconvolution layer the same as a convolutional layer? *CoRR*, abs/1609.07009, 2016.

[32] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations*, 2015.

[33] Ke Sun, Yang Zhao, Borui Jiang, Tianheng Cheng, Bin Xiao, Dong Liu, Yadong Mu, Xinggang Wang, Wenyu Liu, and Jingdong Wang. High-resolution representations for labeling pixels and regions. *CoRR*, abs/1904.04514, 2019.

[34] 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, 2015.

[35] Yu Wang, Quan Zhou, Jia Liu, Jian Xiong, Guangwei Gao, Xiaofu Wu, and Longin Jan Latecki. LEDNet: A lightweight encoder-decoder network for real-time semantic segmentation. In *IEEE International Conference on Image Processing*, pages 1860–1864, 2019.

[36] Wei Xiang, Hongda Mao, and Vassilis Athitsos. ThunderNet: A turbo unified network for real-time semantic segmentation. In *IEEE Winter Conference on Applications of Computer Vision*, pages 1789–1796, 2019.

[37] Changqian Yu, Jingbo Wang, Chao Peng, Changxin Gao, Gang Yu, and Nong Sang. BiSeNet: Bilateral segmentation network for real-time semantic segmentation. In *European Conference on Computer Vision*, pages 334–349, 2018.

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

[39] Hengshuang Zhao, Xiaojuan Qi, Xiaoyong Shen, Jianping Shi, and Jiaya Jia. ICNet for real-time semantic segmentation on high-resolution images. In *European Conference on Computer Vision*, pages 405–420, 2018.

[40] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. Pyramid scene parsing network. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2881–2890, 2016.