Supplementary for "Object-Contextual Representations for Semantic Segmentation"

Yuhui Yuan[1,2,3], Xilin Chen[1,2], and Jingdong Wang[3]

[1] Key Lab of Intelligent Information Processing of Chinese Academy of Sciences (CAS), Institute of Computing Technology, CAS
[2] University of Chinese Academy of Sciences
[3] Microsoft Research Asia
{yuhui.yuan, jingdw}@microsoft.com, xlchen@ict.ac.cn

In Section A, we compare our approach to the existing coarse-to-fine approaches. In Section B, we study the influence of the region numbers and illustrate the qualitative results with Double Attention. In Section C, we report the results of Panoptic-FPN + PPM / ASPP on the COCO val 2017 and the results of Panoptic-FPN / Panoptic-FPN + OCR on the COCO test-dev. In Section D, we apply our OCR on MobileNetV2 to verify the effectiveness of our approach for real-time applications. In Section E, we verify the advantage of our OCR over the conventional DeepLabv3 [1] and DeepLabv3+ [2] based on the recent `MMSegmentation` code base [13]. Last, in Section F, we illustrate some examples of the qualitative improvements based on our OCR scheme.

A. Comparison with Coarse-to-fine Schemes

Many existing studies [4,5,8,11] have exploited various coarse-to-fine schemes to use the coarse segmentation results to boost the final segmentation results. We mainly compare OCR with two popular mechanisms including:
□ label-refinement [5,6]: combine the input image or feature map with a coarse prediction to predict the refined label map. We concatenate the coarse segmentation maps with the feature map output from ResNet-101 Stage 4 and apply the final classifier on the concatenated feature map to predict the refined segmentation maps.
□ label-ensemble [9,10]: ensemble the coarse segmentation maps with the fine segmentation maps directly. We directly use the weighted sum of the coarse segmentation map and the fine segmentation map as the final refined prediction.

Besides, we also report the performance with only the coarse segmentation map (prediction from the ResNet Stage 3) and with only the fine segmentation map (prediction from the ResNet Stage 4). We choose the dilated ResNet-101 as our baseline. According to the results in Table 1, it can be seen that our OCR outperforms all the other coarse-to-fine approaches by a large margin.

B. Ablation Study of Double Attention

Number of Regions We fine-tune the number of regions within Double Attention [3] method and report the results on Cityscapes `val` in Table 2. We

Table 1: Comparison with other coarse-to-fine mechanisms. All the results are evaluated on Cityscapes `val`.

Method	Coarse. seg	Fine. seg	mIoU (%)
baseline	✓	✗	73.90
baseline	✗	✓	75.80
label-ensemble	✓	✓	76.20
label-refinement	✓	✓	77.10
OCR	✓	✓	**79.58**

Table 2: Influence of K within Double Attention. K is the number of regions. K is exact the number of categories for our OCR.

	Double Attention					OCR
# of regions	K=8	K=16	K=32	K=64	K=128	K=19
mIoU	78.52	78.49	78.53	78.65	77.43	**79.58**

choose K=64 if not specified. Besides, it can be seen that the performance with Double Attention is sensitive to the choice of the number of the regions and our approach (with fixed number of regions) consistently outperforms the Double Attention with different region numbers.

Qualitative Results We visualize the predicted regions with Double Attention and the object regions predicted with OCR in Figure 1. It can be seen that the predicted object regions with OCR all correspond to explicit semantic meaning, e.g., road, side-walk and car category separately, while the predicted regions with Double Attention mainly highlight the contour pixels without specific semantic meaning, which might be the main advantages of our approach.

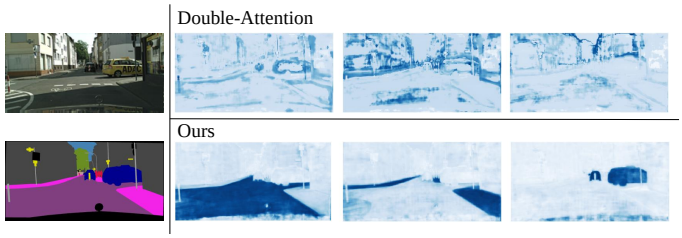

Fig. 1: We randomly choose an image and its ground-truth segmentation map from Cityscapes `val`. The first row illustrates 3 regions predicted with Double Attention and the second row illustrates 3 object regions generated with our OCR. It can be seen that OCR based object regions are more reliable compared to the Double Attention.

C. More Panoptic Segmentation Results

First, we directly apply the PPM or ASPP head before the semantic segmentation head within Panoptic-FPN without any other modifications. In Table 3, we report the results of both methods and we can find our OCR outperforms both the PPM head and the APP head based on Panoptic-FPN. Notably, as illustrated in the paper, our OCR is also more efficient than both PPM and ASPP. Second, we also report the results on the COCO test-dev based on our

Table 3: **Panoptic segmentation results on COCO val 2017.** The performance of Panoptic-FPN [7] is reproduced based on the official open-source Detectron2 [12] and we use the 3× learning rate schedule by default. Our OCR consistently outperforms both PPM and ASPP under the fair comparisons.

Backbone	Method	AP	PQ^{Th}	mIoU	PQ^{St}	PQ
ResNet-50	Panoptic-FPN	40.0	48.3	42.9	31.2	41.5
	Panoptic-FPN + PPM	40.3 (*+0.3*)	48.3 (*+0.0*)	43.2 (*+0.3*)	31.7 (*+0.5*)	41.7 (*+0.2*)
	Panoptic-FPN + ASPP	40.2 (*+0.2*)	48.4 (*+0.1*)	43.3 (*+0.4*)	31.8 (*+0.6*)	41.8 (*+0.3*)
	Panoptic-FPN + OCR	40.4 (*+0.4*)	48.6 (*+0.3*)	44.3 (*+1.4*)	33.9 (*+2.7*)	42.7 (*+1.2*)
ResNet-101	Panoptic-FPN	42.4	49.7	44.5	32.9	43.0
	Panoptic-FPN + PPM	42.5 (*+0.1*)	50.1 (*+0.4*)	44.2 (*−0.3*)	32.8 (*−0.1*)	43.2 (*+0.2*)
	Panoptic-FPN + ASPP	42.3 (*−0.1*)	49.8 (*+0.1*)	44.4 (*−0.1*)	33.0 (*+0.1*)	43.1 (*+0.1*)
	Panoptic-FPN + OCR	42.7 (*+0.3*)	50.2 (*+0.5*)	45.5 (*+1.0*)	35.2 (*+2.3*)	44.2 (*+1.2*)

Table 4: **Panoptic segmentation results on COCO test-dev.** We submit the results based on Panoptic-FPN / Panoptic-FPN + OCR based on ResNet-101 to the COCO test-dev leaderboard. We also report the original results reported in [7]. Our OCR consistently improves the performance on the COCO test-dev.

Method	PQ^{Th}	PQ^{St}	PQ
Panoptic-FPN [7]	48.3	29.7	40.9
Panoptic-FPN	50.8	32.4	43.5
Panoptic-FPN + OCR	50.7 (*−0.1*)	35.1 (*+2.7*)	44.5 (*+1.0*)

OCR in Table 4. We can see that our OCR consistently improves the results on both the COCO val set and test-dev set.

D. Application to MobileNetV2

We apply the OCR on MobileNetV2 and report the performance in Table 2. Specifically, we train the MobileNetV2 following the same training settings expect changing the batch size as 16 and the training iterations as 100K. It can be seen that our OCR significantly improves the segmentation performance on the Cityscapes `val` while slightly increases the inference time (or smaller FPS).

Method	FPS	Cityscapes `val` mIoU
MobileNetV2	**31**	69.50%
MobileNetV2 + OCR	28	**74.18**%

Fig. 2: MobileNetV2 + OCR: Speed (measured by FPS) is tested on P40 GPU with input image of size 1024 × 512

E. `MMSegmentation` Results

To verify that our OCR method generalizes well across different code bases, we further compare the segmentation results of OCR, DeepLabv3 and DeepLabv3+

Table 5: **Cityscapes results based on `MMSegmentation` code base.** The GPU memory consumption is the smaller the better and both FPS and mIoU are the larger the better. The mIoU is evaluated on Cityscapes validation set w/o using flip and multi-scale testing. The GPU Memory consumption is tested with 2 images on each GPU during training. The FPS is tested based on processing 1 image with resolution 1024×2048 on a single GPU. We all use Tesla V100 GPU and Pytorch 1.5.1 for experiments.

Method	iterations	GPU Mem (GB)	FPS	mIoU
DeepLabv3	40K	9.6	2	79.69
DeepLabv3	80K			80.43
DeepLabv3+	40K	11	2.64	80.13
DeepLabv3+	80K			**80.86**
OCR	40K	**8.8**	**3.02**	80.30
OCR	80K			80.81

based on a very recent code base `MMSegmentation` [13]. Specifically speaking, we evaluate different methods under two different training iteration schedules: (i) 40K iterations, (ii) 80K iterations. We set the initial learning rate 0.02 and the batch size 16 for both training schedules. We choose the crop size as 1024×512 and the backbone as dilated ResNet-101 with output stride 8 for all methods by default to ensure the fairness of the comparison.

We report the GPU memory consumption (for training), inference speed (for testing) and mIoUs (on Cityscapes validation set) in Table 5. We can see that OCR achieves better or comparable performance compared to DeepLabv3 and DeepLabv3+ under both kinds of training settings. Especially, our OCR requires less GPU memory consumption and achieve higher FPS on Cityscapes benchmark.

F. Qualitative Improvements

We illustrate the qualitative improvements of our method in Fig. 3 on different benchmarks. We use white dashed boxes to mark the hard regions that are well-classified by our approach but mis-classified by the baseline.

References

1. Chen, L.C., Papandreou, G., Schroff, F., Adam, H.: Rethinking atrous convolution for semantic image segmentation. arXiv:1706.05587 (2017)
2. Chen, L.C., Zhu, Y., Papandreou, G., Schroff, F., Adam, H.: Encoder-decoder with atrous separable convolution for semantic image segmentation. In: ECCV (2018)
3. Chen, Y., Kalantidis, Y., Li, J., Yan, S., Feng, J.: A$\hat{2}$-nets: Double attention networks. In: NIPS (2018)
4. Fieraru, M., Khoreva, A., Pishchulin, L., Schiele, B.: Learning to refine human pose estimation. In: CVPRW (2018)

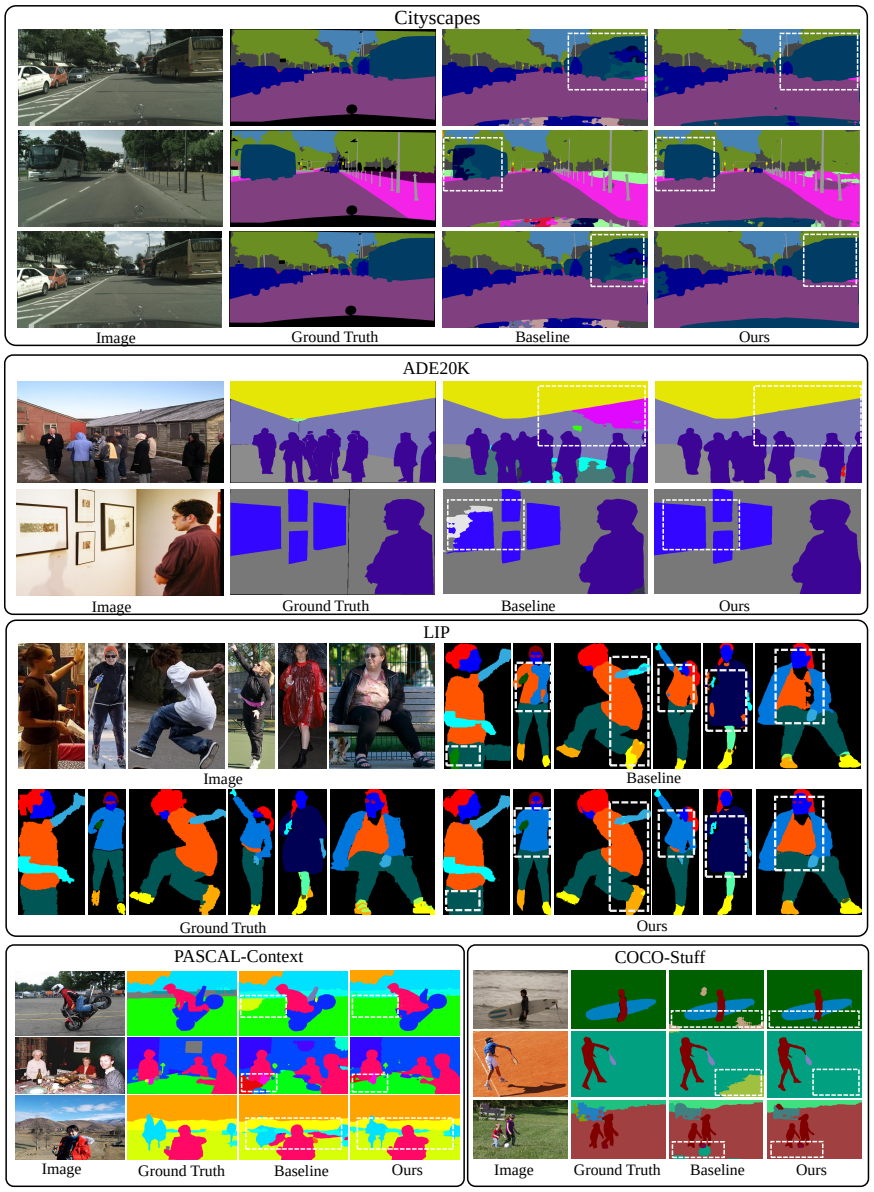

Fig. 3: **Qualitative comparisons.** We compare the segmentation results with dilated ResNet-101 (baseline) and dilated ResNet-101 + OCR (ours) on the 5 benchmarks. We mark the improved regions with white dashed boxes.

5. Gidaris, S., Komodakis, N.: Detect, replace, refine: Deep structured prediction for pixel wise labeling. In: CVPR (2017)

6. Huang, Y.H., Jia, X., Georgoulis, S., Tuytelaars, T., Van Gool, L.: Error correction for dense semantic image labeling. In: CVPRW (2018)
7. Kirillov, A., Girshick, R., He, K., Dollár, P.: Panoptic feature pyramid networks. In: CVPR (2019)
8. Li, K., Hariharan, B., Malik, J.: Iterative instance segmentation. In: CVPR (2016)
9. Li, X., Liu, Z., Luo, P., Change Loy, C., Tang, X.: Not all pixels are equal: Difficulty-aware semantic segmentation via deep layer cascade. In: CVPR (2017)
10. Nigam, I., Huang, C., Ramanan, D.: Ensemble knowledge transfer for semantic segmentation. In: WACV (2018)
11. Tu, Z., Bai, X.: Auto-context and its application to high-level vision tasks and 3d brain image segmentation. PAMI (2010)
12. Wu, Y., Kirillov, A., Massa, F., Lo, W.Y., Girshick, R.: Detectron2. https://github.com/facebookresearch/detectron2 (2019)
13. Xu, J., Chen, K., Lin, D.: MMSegmenation. https://github.com/open-mmlab/mmsegmentation (2020)