Supplementary Materials for "Feature Pyramid Transformer"

These materials include the details of the effectiveness of F_{eud} (Section A), an additional study on hyperparameters (Section B), FPT complexity analysis (Section C), more quantitative result comparisons (Section D), and more qualitative results (Section E).

A Effectiveness of F_{eud}

In Section 3.3 of the main paper, we propose to use the negative value of euclidean distance F_{eud} [1] (instead of the conventional F_{sim} [2]) as the similarity function in Grounding Transformer (GT). In this section, we show the effectiveness of F_{eud}. In details, the Mixture of Softmaxes (MoS) [3] is deployed as the normalizing function, and the number of the divided parts $\mathcal{N}$ is set to 4. Table S1 shows that F_{eud} surpasses the classic *softmax*-based F_{sim} in both cases of GT (with and without MoS). In particular, F_{eud} with MoS achieves the best performance. There are at most 3.1% box AP and 3.3% mask AP improvements for object detection and instance segmentation, respectively.

Methods	AP		AP_{50}		AP_{75}		AP_S		AP_M		AP_L	
BFP [4]	31.6	29.9	54.1	50.7	35.9	34.7	16.1	14.2	32.5	31.6	48.8	48.5
+ F_{sim} [2]	32.2	30.5	54.2	50.9	35.6	34.5	16.3	14.5	32.1	31.2	49.0	48.6
+ F_{eud} [1]	33.7	32.1	54.5	52.1	36.0	34.9	16.9	15.6	33.0	31.8	49.4	48.7
+ F_{sim} [2] + MoS [3]	32.6	31.1	54.3	51.2	35.8	34.8	16.4	15.3	32.6	31.5	49.1	48.5
+ F_{eud} [1] + MoS [3]	**34.7**	**33.2**	**55.4**	**53.2**	**37.0**	**36.1**	**17.8**	**16.2**	**33.9**	**32.0**	**50.3**	**49.1**

Table S1. Comparing F_{eud} with F_{sim} on validation set of MS-COCO 2017 [5]. The backbone is ResNet-50 [6]. "BFP" is the bottom-up feature pyramid (BFP) [4]. Results on the left and right of the dashed are respectively from bounding box detection and instance segmentation.

B Hyperparameters

B.1 $\mathcal{N}$ in ST

In Section 3.2, we use MoS [3] as the normalizing function. In this section, we investigate the influence of $\mathcal{N}$ (in MoS) on Self-Transformer (ST). In particular, no MoS [3] means $\mathcal{N}$=1, *i.e.*, the classical *softmax* [7]. In Table S2, we can see that $\mathcal{N}$=2 brings the best performance in all cases.

$\mathcal{N}$	AP		AP_{50}		AP_{75}		AP_S		AP_M		AP_L	
BFP [4]	31.6	29.9	54.1	50.7	35.9	34.7	16.1	14.2	32.5	31.6	48.8	48.5
1 (w/o MoS [3])	31.7	30.0	54.3	50.5	36.0	34.9	16.3	14.3	32.6	31.9	48.5	48.4
2	**31.8**	**30.2**	**54.6**	**51.1**	**36.5**	**35.1**	**16.8**	**14.6**	**33.2**	**32.0**	**49.8**	**49.3**
4	31.1	29.3	54.0	50.5	35.9	34.6	16.4	14.1	32.8	31.4	49.1	48.3
6	30.6	28.7	53.6	49.7	35.7	34.0	16.1	13.7	32.2	30.9	48.8	48.0
8	30.0	28.1	53.1	49.5	35.3	33.7	15.9	13.3	31.8	30.5	48.2	47.5

Table S2. The influence of $\mathcal{N}$ on ST. Experiments are carried out on validation set of MS-COCO 2017 [5]. The backbone is ResNet-50 [6]. "BFP" is the bottom-up feature pyramid (BFP) [4]. "w/o MoS" means that these results are obtained without MoS [3]. Results on the left and right of the dashed are of bounding box detection and instance segmentation.

$\mathcal{N}$	AP		AP_{50}		AP_{75}		AP_S		AP_M		AP_L	
BFP [4]	31.6	29.9	54.1	50.7	35.9	34.7	16.1	14.2	32.5	31.6	48.8	48.5
1 (w/o MoS [3])	33.7	32.1	54.5	52.1	36.0	34.9	16.9	15.6	33.0	31.8	49.4	48.6
2	34.3	32.7	55.1	52.8	36.5	35.4	17.3	15.9	33.5	31.6	49.9	48.8
4	**34.7**	**33.2**	**55.4**	**53.2**	**37.0**	**36.1**	**17.8**	**16.2**	**33.9**	**32.0**	**50.3**	**49.1**
6	33.4	32.9	54.3	52.7	36.4	35.6	17.3	15.7	33.5	31.5	50.0	48.7
8	32.5	32.3	53.3	52.0	35.9	35.0	17.0	15.2	32.9	30.7	49.5	48.4

Table S3. The influence of $\mathcal{N}$ on GT. Experiments are carried out on validation set of MS-COCO 2017 [5]. The backbone is ResNet-50 [6]. "BFP" is the bottom-up feature pyramid (BFP) [4]. "w/o MoS" means that these results are obtained without MoS [3]. Results on the left and right of the dashed are of bounding box detection and instance segmentation.

B.2 $\mathcal{N}$ in GT

In this section, we investigate the influence of $\mathcal{N}$ (in MoS) on GT. As shown in Table S3, we can see that using $\mathcal{N}$=4 achieves the best performance for both object detection and instance segmentation.

B.3 *square_size* in LGT

In Section 3.3, we introduce LGT for semantic segmentation. In this section, we investigate the influence of the side length *square_size* (of local square area) on LGT. We use MoS [3] with $\mathcal{N}$=4 as the normalizing function. We report the standard mean Intersection of Union (mIoU) on the training set (*i.e.*, Tra.mIoU) as well as the validation set (*i.e.*, Val.mIoU) of Cityscapes [8], in Table S4. We can see that LGT with *square_size*=5 achieves the best performance.

B.4 DropBlock in Instance-level Tasks

In Section 3.5, we apply the DropBlock [10] to each transformed feature map, to alleviate the over-fitting problem. In this section, we investigate the influence of

square_size	Method	Tra.mIoU (%)	Val.mIoU (%)
-	backbone + UFP	86.0	79.1
1	backbone + UFP + LGT	86.1	79.5
3	backbone + UFP + LGT	86.2	79.9
5	backbone + UFP + LGT	**86.3**	**80.0**
7	backbone + UFP + LGT	86.1	79.8
9	backbone + UFP + LGT	85.8	79.6

Table S4. The influence of *square_size* of LGT on the pixel-level semantic segmentation task. The backbone is the dilated ResNet-101 [9]. Experiments are carried out on training set and the validation set of Cityscapes [8]. "UFP" is the unscathed feature pyramid.

Settings	*block_size*=1		*block_size*=3		*block_size*=5		*block_size*=7	
keep_prob=0.1	30.7	29.7	31.4	30.7	31.9	30.1	30.0	29.8
keep_prob=0.3	32.1	30.9	32.9	31.6	33.2	31.1	31.8	30.8
keep_prob=0.5	33.2	31.0	34.2	33.5	35.5	34.7	33.7	33.4
keep_prob=0.7	33.8	32.4	35.6	34.9	36.1	35.7	35.8	34.6
keep_prob=0.9	34.2	33.8	36.6	35.1	**38.0**	**36.8**	36.6	35.3

Table S5. The influence of *block_size* and *keep_prob* of DropBlock [10] on the instance-level tasks (*i.e.*, object detection and instance segmentation). The backbone is ResNet-50 [6]. Results on the left and right of the dashed are AP of bounding box detection and mask AP of instance segmentation.

two hyper-parameters (*i.e.*, the drop block size *block_size* and the keep probability *keep_prob* of each feature position) of DropBlock [10] on instance-level tasks (*i.e.*, object detection and instance segmentation). The MoS [3] is applied in GT with its $\mathcal{N}$=4, and in ST with its $\mathcal{N}$=2. In Table S5, we find that *block_size*=5 and *keep_prob*=0.9 result the best performance.

B.5 DropBlock in Pixel-level Task

In this section, we investigate the influence of two hyper-parameters (*i.e.*, the drop block size *block_size* and the keep probability *keep_prob* of each feature position) of DropBlock [10] on the pixel-level semantic segmentation. The MoS [3] is applied in GT with its $\mathcal{N}$=4, and in ST with its $\mathcal{N}$=2. The *square_size* of LGT is set to 5. We report mIoU on the validation set (*i.e.*, Val.mIoU) of Cityscapes [8] in Table S6. We find that using *block_size*=3 and *keep_prob*=0.9 achieves the best performance.

C FPT Complexity

In Section 4.1.1, we report the model efficiency. In this section, we supplement the details of model Parameters (Params) and FLOPs using Mask R-CNN [11]

Settings	*block_size*=1	*block_size*=3	*block_size*=5	*block_size*=7
keep_prob=0.1	80.8	81.0	80.8	80.4
keep_prob=0.3	81.0	81.1	81.0	80.7
keep_prob=0.5	81.2	81.4	81.2	80.9
keep_prob=0.7	81.4	81.6	81.3	81.1
keep_prob=0.9	81.3	**81.7**	81.5	81.4

Table S6. The influence of *block_size* and *keep_prob* of DropBlock [10] on the pixel-level semantic segmentation. The backbone is the dilated ResNet-101 [9]. Experiments are carried out on validation set of Cityscapes [8]. Results in this table refer to the mIoU on the validation set (*i.e.*, Val.mIoU).

Methods	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L	Params	FLOPs
BFP [4]	29.9	50.7	34.7	14.2	31.6	48.5	1×	1×
+ non-local [7]	30.8 (↑ **0.9**)	52.4 (↑ **1.7**)	35.5 (↑ **0.8**)	15.2 (↑ **1.0**)	32.5 (↑ **0.9**)	49.5 (↑ **1.0**)	1.24×	1.24×
+ ST	30.6 (↑ **0.7**)	51.4 (↑ **0.7**)	35.5 (↑ **0.8**)	15.1 (↑ **0.9**)	32.1 (↑ **0.5**)	49.7 (↑ **1.2**)	1.59×	1.44×
+ GT	33.9 (↑ **4.0**)	52.4 (↑ **1.7**)	37.7 (↑ **3.0**)	16.9 (↑ **2.7**)	33.3 (↑ **1.7**)	51.7 (↑ **3.2**)	1.85×	1.54×
+ RT	33.1 (↑ **3.2**)	52.0 (↑ **1.3**)	37.7 (↑ **3.0**)	15.3 (↑ **1.1**)	34.9 (↑ **3.3**)	52.1 (↑ **3.6**)	1.15×	1.09×
+ FPT	36.8 (↑ **6.9**)	55.9 (↑ **5.2**)	38.6 (↑ **3.9**)	18.8 (↑ **4.6**)	35.3 (↑ **3.7**)	54.2 (↑ **5.7**)	2.54×	2.01×

Table S7. Model complexity analysis on validation set of MS-COCO 2017 [5] for instance segmentation. The backbone is ResNet-50 [6]. "BFP" is the bottom-up feature pyramid (BFP) [4].

head. In Table S7, we compare our FPT and its components (*i.e.*, ST, GR, and RT) to the non-local operation on the validation set of MS-COCO 2017 for instance segmentation [5]. The implementation detail of the non-local operation is the same as that in [7].

From Table S7, we can find that for the non-local operation the average increases of the model Params and FLOPs required by AP at per improved point are 0.27× and 0.27×, respectively. In contrast, the average increases in our FPT are lower (better) as 0.21× and 0.15×, respectively.

D More Quantitative Result Comparisons

D.1 Results on Stronger Backbones

In addition to ResNet [6], we also employ the Non-local ResNet [7], the Global Context Network (GC-ResNet) [12], and the Attention Augmented Network (AA-ResNet) [13] as backbone networks in the instance-level recognition. In this section, we report more quantitative results on these backbones in Table S8.

In Table S8, we can observe that BFP+FPT still achieves the better performance than BFP+FPN, BFP+BPA and BFP+BFI on the stronger backbone networks (*i.e.*, NL-, GC-, and AA-ResNet). In particular, BFP+FPT achieves up to 40.8% bounding box AP (and 38.7% mask AP), while BFP+FPN, BFP+BPA and BFP+BFI can achieve 37.9% bounding box AP (and 36.8% mask AP),

Methods	Backbone	AP		AP_{50}		AP_{75}		AP_S		AP_M		AP_L	
BFP+FPN [4]	NL-ResNet	37.2	36.4	60.1	59.2	40.0	38.5	19.0	16.7	37.8	37.1	51.1	49.9
BFP+BPA [14]	NL-ResNet	38.5	37.6	60.9	59.5	41.6	39.2	20.5	18.1	39.5	38.7	51.9	51.0
BFP+BFI [15]	NL-ResNet	38.9	37.8	61.2	59.7	41.5	39.5	20.2	18.6	39.7	38.9	51.5	50.5
BFP+FPT	NL-ResNet	40.1	38.0	62.9	60.7	42.0	40.6	21.4	19.1	40.8	39.9	53.0	51.8
BFP+FPN [4]	GC-ResNet	37.7	36.8	60.4	59.5	40.1	38.8	19.2	17.2	38.5	37.5	51.3	50.5
BFP+BPA [14]	GC-ResNet	38.8	37.4	61.2	59.8	41.9	40.3	20.8	18.5	39.7	38.9	52.2	51.5
BFP+BFI [15]	GC-ResNet	39.0	37.7	62.0	60.2	42.3	40.7	21.1	18.9	40.2	39.1	52.0	51.8
BFP+FPT	GC-ResNet	40.4	38.5	63.3	61.0	43.5	41.9	22.6	19.7	41.1	40.5	53.4	52.3
BFP+FPN [4]	AA-ResNet	37.9	36.7	60.7	59.6	40.3	38.4	19.6	17.5	38.6	37.3	51.8	50.1
BFP+BPA [14]	AA-ResNet	38.5	37.5	61.7	59.3	41.8	40.1	20.4	18.2	39.8	38.5	52.7	51.7
BFP+BFI [15]	AA-ResNet	38.9	37.9	62.1	60.1	42.3	40.7	21.0	18.5	39.5	38.9	52.2	51.3
BFP+FPT	AA-ResNet	40.8	38.7	63.8	61.3	43.7	41.5	22.7	19.4	41.5	40.8	53.3	52.0

Table S8. Combining FPN/BPA/BFI and NL-ResNet/GC-ResNet/AA-ResNet on validation set of MS-COCO 2017 [5]. The base is ResNet-50. Results on the left and right of the dashed are respectively from bounding box detection and instance segmentation.

Methods	Backbone	AP		AP_{50}		AP_{75}		AP_S		AP_M		AP_L		Params
BFP+ FPT	ResNet-50	38.0	36.8	57.1	55.9	38.9	38.6	20.5	18.8	38.1	35.3	55.7	54.2	88.2 M
BFP+ FPN [4]	ResNet-101	36.2	35.7	59.1	58.0	39.0	37.8	18.2	15.5	39.0	38.1	52.4	49.2	88.0 M
BFP+ BPA [14]	ResNet-101	37.3	36.3	60.4	58.7	39.9	38.3	18.9	16.3	39.7	39.0	53.0	50.5	88.4 M
BFP [4]	ResNet-152	35.8	34.6	55.7	53.8	37.8	35.6	15.3	14.3	35.2	33.2	51.5	45.8	89.3 M
BFP+ FPN [4]	ResNet-152	38.3	37.1	60.2	58.5	39.7	38.0	19.0	16.1	39.6	38.9	53.0	50.1	91.2 M

Table S9. Result comparisons on different backbones. Experiments are carried out on validation set of MS-COCO 2017 [5]. "BFP" is the bottom-up feature pyramid (BFP) [4]. Results on the left and right of the dashed are of bounding box detection and instance segmentation.

38.8% bounding box AP (and 37.6% mask AP), and 39.0% bounding box AP (and 37.9% mask AP), respectively.

D.2 Results on Deeper Backbones

In this section, we report more result comparisons on the deeper backbone network (*i.e.*, ResNet-152) in Table S9. We can observe that BFP + FPT on **ResNet-50** achieves 38.0%/36.8% AP, which surpasses BFP + FPN [4] and BFP + BPA [14] (*i.e.*, 36.2%/35.7% AP and 37.3% /36.3% AP) on **ResNet-101** under the similar number of parameters (88 M). Compared to results on **ResNet-152**, FPT can still surpass BFP (35.8%/34.6% AP) with fewer parameters. Although BFP+FPN on **ResNet-152** can slightly outperform FPT on **ResNet-50**, it has more parameters.

E More Qualitative Results

This section supplements to the visualization results given in Section 4.1.1 and Section 4.2.2 (of the main paper). The results of object detection, instance segmentation and semantic segmentation are visualized respectively in Fig. S1, Fig. S2 and Fig. S4. The samples for object detection and instance segmentation are from the test set of MS-COCO 2017 [5]. As we can see in Fig. S1 and Fig. S2 that most of our predictions are of high quality, *e.g.*, small objects such as persons and sheep in the distance are correctly detected. The samples for semantic segmentation are from the validation set of PSACAL VOC2012 [16]. The demonstration in Fig. S4 validates that FPT achieves precise segmentation of the thinner objects, *e.g.*, the biker's foot, the cat's tail, the man in the distance and the woman's arm. Moreover, FPT enhances the segmentation quality of larger objects, *e.g.*, the sofa, the bottle, and the dining-table.

In Fig. S3, we additionally present the failure examples for object detection and instance segmentation. One possible reason for these failure results is that the background of these objects is not annotated in the ground truth, for example there is no category information for "mirror" and "painting" in the MS-COCO 2017 [5] dataset. Hence, objects in these backgrounds can easily be recognized as the real ones, *e.g.*, the cat in the painting, the bike on the wall, and the man in the mirror.

References

1. Zhang, Y., Hare, J., Prügel-Bennett, A.: Learning to count objects in natural images for visual question answering. In: ICLR. (2018)
2. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. In: NeurIPS. (2017)
3. Yang, Z., Dai, Z., Salakhutdinov, R., Cohen, W.W.: Breaking the softmax bottleneck: A high-rank rnn language model. In: ICLR. (2018)
4. Lin, T.Y., Dollár, P., Girshick, R., He, K., Hariharan, B., Belongie, S.: Feature pyramid networks for object detection. In: CVPR. (2017)
5. Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft coco: Common objects in context. In: ECCV. (2014)
6. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR. (2016)
7. Wang, X., Girshick, R., Gupta, A., He, K.: Non-local neural networks. In: CVPR. (2018)
8. Cordts, M., Omran, M., Ramos, S., Rehfeld, T., Enzweiler, M., Benenson, R., Franke, U., Roth, S., Schiele, B.: The cityscapes dataset for semantic urban scene understanding. In: CVPR. (2016)
9. Yu, F., Koltun, V.: Multi-scale context aggregation by dilated convolutions. In: ICLR. (2016)
10. Ghiasi, G., Lin, T.Y., Le, Q.V.: Dropblock: A regularization method for convolutional networks. In: NeurIPS. (2018)
11. He, K., Gkioxari, G., Dollár, P., Girshick, R.: Mask r-cnn. In: ICCV. (2017)
12. Cao, Y., Xu, J., Lin, S., Wei, F., Hu, H.: Gcnet: Non-local networks meet squeeze-excitation networks and beyond. In: ICCV. (2019)

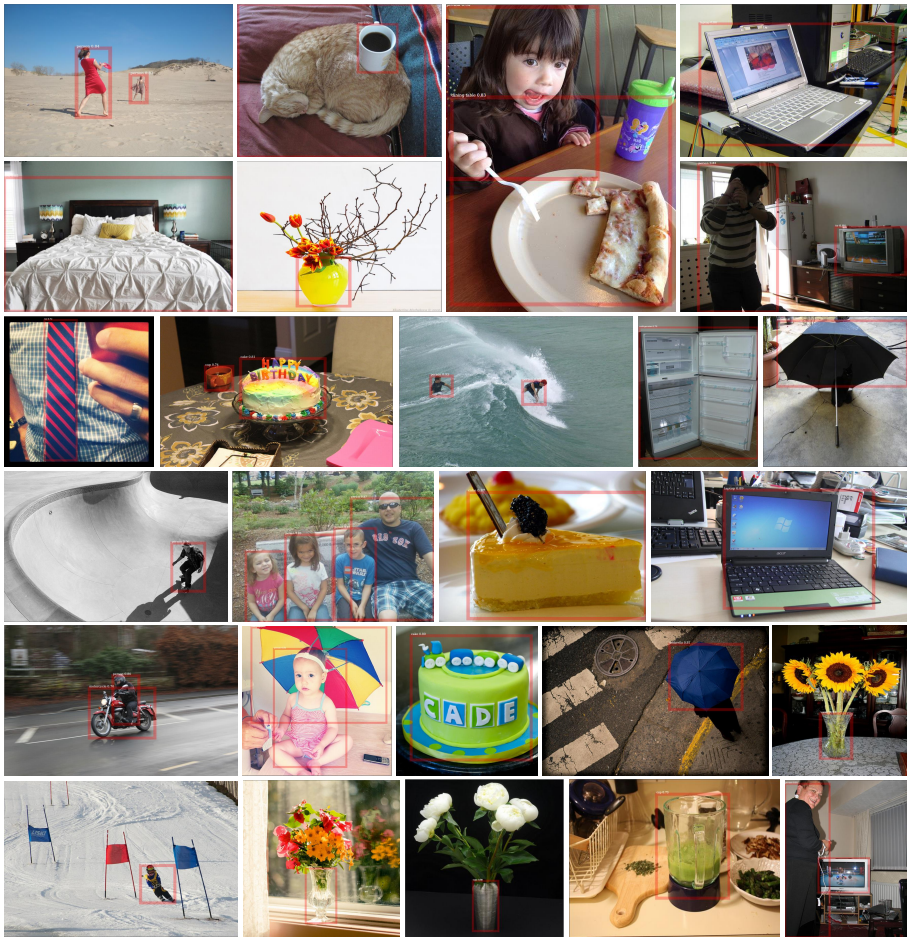

Fig. S1. More object detection results. Samples are from test set of MS-COCO 2017 [5].

Fig. S2. More instance segmentation results. Samples are from test set of MS-COCO 2017 [5]. The red rectangle highlights the better predicted areas of FPT.

Fig. S3. Failure examples of object detection and instance segmentation. Samples are from test set of MS-COCO 2017 [5].

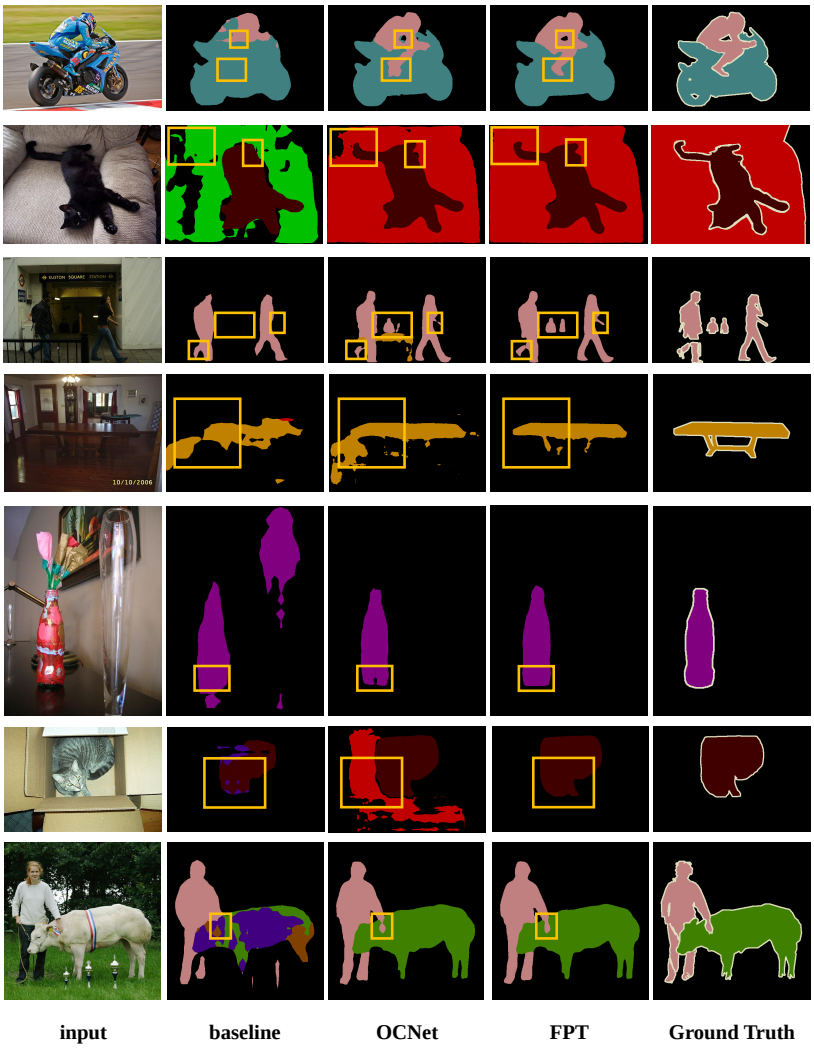

Fig. S4. Semantic segmentation results. Samples are from val set of PSACAL VOC 2012 [16]. The yellow rectangle highlights the better predicted areas of FPT.

13. Irwan, B., Barret, Z., Ashish, V., Jonathon, S., Quoc, V.L.: Attention augmented convolutional networks. In: ICCV. (2019)
14. Liu, S., Qi, L., Qin, H., Shi, J., Jia, J.: Path aggregation network for instance segmentation. In: CVPR. (2018)
15. Lin, D., Shen, D., Shen, S., Ji, Y., Lischinski, D., Cohen-Or, D., Huang, H.: Zigzagnet: Fusing top-down and bottom-up context for object segmentation. In: CVPR. (2019)
16. Everingham, M., Eslami, S.A., Van Gool, L., Williams, C.K., Winn, J., Zisserman, A.: The pascal visual object classes challenge: A retrospective. IJCV **111**(1) (2015) 98–136