High Quality Entity Segmentation
Supplementary File

Lu Qi[1]*, Jason Kuen[2*], Tiancheng Shen[4*], Jiuxiang Gu[2], Wenbo Li[4],
Weidong Guo[3†], Jiaya Jia[4], Zhe Lin[2], Ming-Hsuan Yang[1],

[1]The University of California, Merced [2]Adobe Research
[3]QQ Brower Lab, Tencent, [4]The Chinese University of Hong Kong

This supplementary material document provides more statistical information of and results of ablation study experiments performed on the EntitySeg dataset. Furthermore, we offer more visualization of the EntitySeg dataset and experimental results for the proposed CropFormer. The supplementary material is organized as follows:

- The statistical information of the EntitySeg dataset.
- The experiment setting details left out in the main paper.
- More ablation study experiments on our baseline Mask2Former [3] and the proposed CropFormer.

as well as:

- More visualization results of our proposed CropFormer.
- More visualization examples from EntitySeg dataset.
- More visualization examples in the wild including FSS [9], CAMO [8], OCID [15] and LVIS [5].

1. EntitySeg Dataset

Statistics of category numbers at pixel level Figure 1 shows the class frequency distribution at the pixel level. Compared to the entity-level category numbers in Figure 3 of our paper, the top classes (ranked by pixel-level frequency) mostly belong to stuff in the EntitySeg dataset. This phenomenon is similar to the COCO [10] and ADE20K [16] dataset, where the stuff classes usually have larger areas than thing classes.

Statistics of category numbers at entity level Figure 2 shows the class frequency distribution which follows Zipf's law, resembling existing datasets like COCO [10] and ADE20K [16].

Algorithm 1 The Calculation of Simplicity and Complexity Pseudocode (Python-like)

```
import numpy as np
import cv2

convexity_count = 0
simplicity_count = 0
for mask in masks:
   cnt = cv2.findContours(mask, cv2.RETR_LIST, cv2.
        CHAIN_APPROX_NONE)[0]
   perimeter_inst = 1e-8
   for cnt_ in cnt:
      perimeter_inst += cv2.arcLength(cnt_, True)
   hull = cv2.convexHull(np.transpose(np.nonzero(
        mask)))
   convexhall_inst_area = cv2.contourArea(hull)
   simplicity_inst = math.sqrt(4*math.pi*mask.sum())
        / perimeter_inst
   convexity_inst = mask.sum()/convexhall_inst_area

   convexity_count += np.clip(convexity_inst,0,1)
   simplicity_count += np.clip(simplicity_inst,0,1)
convexity = convexity_count / len(masks)
simplicity = simplicity_count / len(masks)
```

Simplicity and complexity The shape convexity and simplicity of an entity's mask S are measured by Eq. 1 and Eq. 2 following [17, 13]:

$$convexity(S) = \frac{Area(S)}{\text{the}Area(ConvexHull(S))}. \tag{1}$$

$$simplicity(S) = \frac{\sqrt{4\pi * \text{Area(S)}}}{Perimeter(S)}. \tag{2}$$

where large convexity and simplicity values imply that the mask is a simple shape (and both metrics achieve their maximum value of 1.0 for a circle [17]). We attach the pseudo code as follows:

2. Experiments

For the training set, we use the same hyper-parameters of the COCO training except for the training iterations and learning rate steps considering the dataset size difference. Our EntitySeg dataset adopts 1× training schedule as 34,375 iterations and decays learning rate after 30,525 and

*Equal contribution. † indicates corresponding author.

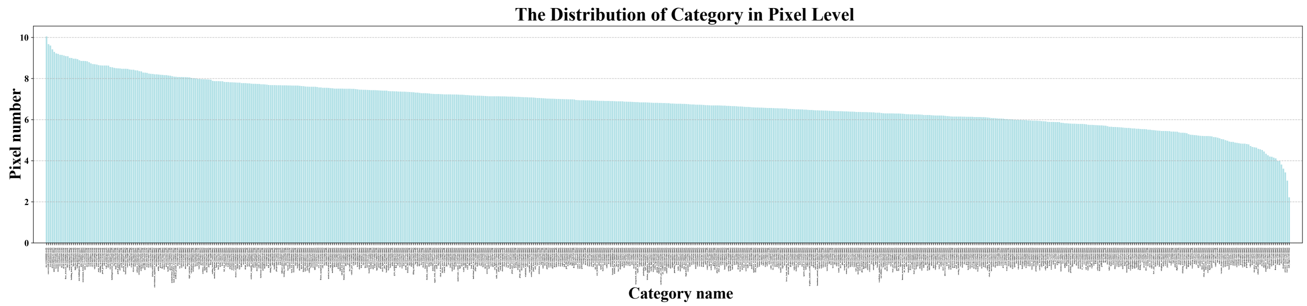

Figure 1: The class distribution of the EntitySeg dataset at pixel level. Please zoom in on the x-axis of the figure to view the class names.

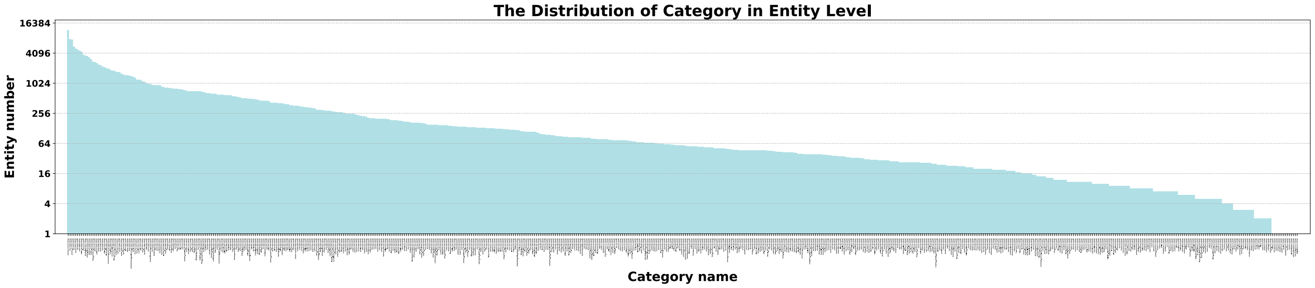

Figure 2: The class distribution of the EntitySeg dataset. Here we mention if a class belongs to *thing* or *stuff*. Please zoom in on the x-axis of the figure to see the class names.

33,138 iterations. N× training schedule indicates that the number of iterations and learning rate decay schedule is adjusted by a factor of N. Given the EntityClass dataset has only one-third image numbers of the EntitySeg dataset, the 1× training schedule is sufficient to obtain the best performance on class-aware segmentation tasks. AdamW [12] is used as the optimizer with a base learning rate of 0.0001 and batch size of 16.

2.1. Class-aware Segmentation Tasks

Entity Segmentation. We split the EntitySeg dataset into train and test sets with 31,913 and 1,314 images. We use class-agnostic metric AP^{e} [14] with a strict non-overlapping mask constraint for evaluation on the entity segmentation task. To reduce the bias of dataset split, we constructed 20 random dataset splits. Fig. 3 shows the ablation study on randomly sampled train/test splits in our EntitySeg dataset for entity segmentation. Overall, we sample the split pairs by 20 times and then train/test Mask2Former [3] with every pair. And the mean and standard variance of AP^{e} is 39.5 and 0.9. Finally, we choose the split whose AP^{e} is closest to 39.5.

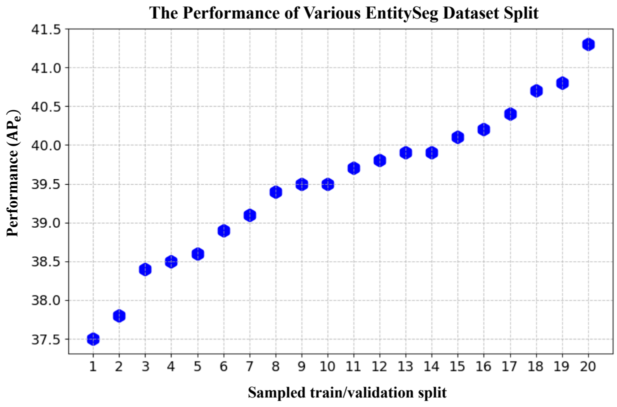

Figure 3: The study on multiple randomly-sampled train/-validation splits for EntitySeg dataset.

Table 1 shows the benchmark of our EntitySeg dataset with Mask-RCNN [6], EntityFramework [14], Mask2Former [4]. The first two methods are convolution-based dense prediction methods, and the last is a Transformer-based query prediction method. In Table 1, the Transformer-based Mask2Former performs better than the other two convolution-based methods on all metrics, demonstrating the advantage of transform-based methods on high-quality mask generation. We witness that the COCO-E pretrained weights can further boost entity segmentation performance. We also explored the optimal training iterations needed by Mask2Former and found that it performs the best with 3× training schedule.

Semantic Segmentation. In Table 2, we evaluate the two popular semantic segmentation methods DeeplabV3 [1] and Mask2Former [4] on EntitySem. Specifically, we choose

Model	Backbone	Iteration	Pretrain	AP^e	AP^e_{50}	AP^e_{75}
Mask-RCNN [6]	Swin-T	1×	ImageNet	24.9	45.8	24.1
			COCO-E	28.4	49.2	28.1
EntityFramework [14]	Swin-T	1×	ImageNet	26.0	42.8	25.7
			COCO-E	29.9	47.6	30.1
Mask2Former [4]	Swin-T	1×	ImageNet	33.2	50.2	33.1
			COCO-E	39.5	56.9	40.2
		2×		40.2	57.6	41.1
		3×		40.9	58.1	41.6
		4×		40.9	57.9	41.9
	Swin-L	3×		46.2	63.7	47.5

Table 1: Entity segmentation benchmark in Entity Dataset. The column of ‘Pretrain’ indicates the pretraining weights we used where the ‘ImageNet’ is ImageNet pretraining and ‘COCO-E’ refers to pretraining on COCO dataset that has been converted to class-agnostic entity segmentation format [14].

Model	Backbone	Pretrain	mIoU
DeeplabV3 [2]	R-50	ImageNet	27.9
Mask2Former [4]	R-50	ImageNet	37.8
		COCO-P	43.3
	Swin-T	COCO-E	43.0
		COCO-P	45.0
	Swin-L	COCO-E	50.7
		COCO-P	50.5

Table 2: Benchmark on class-aware semantic segmentation in EntitySem Dataset. The ‘COCO-P’ and ‘COCO-E’ indicate weights trained in the COCO datasets with panoptic and entity segmentation tasks.

Model	Backbone	Pretrain	AP	AP_{50}	AP_{75}
Mask-RCNN [6]	R-50	ImageNet	5.0	9.3	4.9
		COCO-I	11.9	18.9	12.4
Mask2Former [4]	R-50	ImageNet	13.0	19.6	13.3
		COCO-I	20.3	29.2	21.0
	Swin-T	COCO-E	20.0	28.8	20.7
		COCO-I	22.5	32.4	23.5
		COCO-P	22.7	32.7	23.5
	Swin-L	COCO-E	28.0	39.3	29.4
		COCO-P	30.3	42.3	31.6

Table 3: Benchmark on class-aware instance segmentation in Entity Dataset. The ‘COCO-I’ indicates weights trained in the COCO datasets with instance segmentation tasks.

Model	Backbone	Pretrain	PQ	SQ	RQ
PanopticFPN [7]	R-50	ImageNet	3.6	25.8	5.5
		COCO-P	6.7	36.4	10.1
Mask2Former [4]	R-50	ImageNet	5.5	26.3	8.2
		COCO-I	9.6	39.0	14.1
	Swin-T	COCO-E	7.4	32.6	11.0
		COCO-P	9.8	38.5	14.6
	Swin-L	COCO-E	11.7	43.2	17.3
		COCO-P	13.4	48.7	19.9

Table 4: Benchmark on class-aware panoptic segmentation on the Entity Dataset.

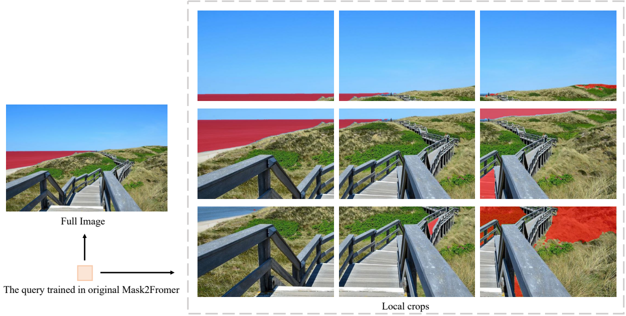

Figure 4: Preliminary study on robustness of queries trained in an image-level Mask2Former.

150 categories with the highest pixel-level frequency as EntitySem for semantic segmentation. EntitySem has 9,729 and 1,444 images for training and testing, respectively. We can see that semantic segmentation performance is still related to the pretraining weights and network structure. Mask2Former with Swin-L backbone and COCO-E pretrained weights obtains the mIoU of 50.5 on EntitySem.

Instance Segmentation. In Table 3, we ablate two popular instance segmentation methods including Mask-RCNN [6] and Mask2Former [4] on EntityIns. We select 206 thing categories with the highest object-level frequency in the EntityClass dataset to benchmark instance segmentation. In the EntityIns, 8,993 and 1,498 images for training and testing, respectively. We can see that Mask2Former with Swin-L backbone and COCO-P pretrained weights the best AP of 30.3 on EntityIns.

Panoptic Segmentation. Table 4 shows the performance of two popular panoptic segmentation methods, including PanopticFPN [7] and Mask2Former [4]. Similar to EntityIns, we select 345 categories, including 274 things and 71 stuffs, with the highest entity-level frequency to construct EntityPan. There are 9,968 and 1,481 images for training and testing. We find that the task becomes more challenging with the greater variety of class labels. *E.g.*, in Table 4, the PQs are much lower than those of existing panoptic datasets. In addition, current methods perform worse on EntityPan compared to existing datasets, especially on recognition quality (RQ), which adversely impacts PQ.

2.2. Entity Segmentation

The robustness of queries in image-level Mask2Former Figure 4 shows the preliminary investigation on the robustness of queries in an image-level Mask2Former. We find that the same queries are not robust in representing the same entities across different image crops. Thus we are not able to utilize such queries to ensemble inference results of same entities across multiple crops. That provides us with a strong motivation to design CropFormer to solve the above-mentioned limitation of Mask2Former.

The impact of input resolution We ablate the impact of input resolution on the final performance. In Table 5, we use the bilinear and nearest interpolation in OpenCV to resize the original image and annotation masks in EntitySeg Dataset to the three kinds of sizes. We train Mask2Former with such different size kinds and find that the perfor-

Resolution	AP^{e}
(400, 677)	37.5
(800, 1333)	38.4
(1600, 2666)	39.1
original	**39.5**

Table 5: The study on the impact of the image and annotation resolutions on Mask2Former training. '(x, y)' indicates the shorter side's size and longer side's maximum size, respectively

Datloader Style	Details	AP^{e}
Mask2Former	Scale(0.1, 2.0)→Crop(1024, 1024)	39.5
Mask-RCNN	Scale to ((640, 800), 1333)	39.5

Table 6: The ablation study on the dataloader choice for Mask2Former training. The column 'Details' describes some details of the dataloader style.

Layers	AP^{e}
3	40.6
6	40.8
9	**41.0**

Table 7: The ablation study on the number of Transformer layers in the batch-level decoder in CropFormer.

mance degrades severely with input images and mask annotations that are low in resolutions. This somehow suggests that fine-grained entity segmentation can benefit greatly from high-resolution training images, which are one of the unique characteristics of our EntitySeg dataset.

Dataloader used in Mask2Former There are two kinds of dataloaders are applicable to entity-level segmentation tasks (*e.g.,* panoptic segmentation, instance segmentation). The first is the Mask2Former style, which scales the original image with a random scale ratio (from 0.1 to 2.0) before cropping a region of 1024×1024. The second is the Mask-RCNN style, which directly scales the original image to $\{640, ..., 800\}$px for the shorter side and maximum of 1,333px for the longer side. In Table 6, we compare these two styles and find no difference between them when applied to our baseline Mask2Former. To avoid further cropping on the corner crops of CropFormer, we apply the Mask-RCNN-style dataloader to CropFormer, as well as to other experimented models for the sake of consistency.

The layers of transformer decoder in batch decoder We ablate the number of layers used in the transformer decoder of the batch-level decoder in Table 7. We find that using all nine layers could obtain the best performance.

Improvement with a stronger backbone Table 8 shows that our proposed CropFormer with Swin-L [11] backbone can still achieve a remarkable improvement of 1.8 AP^{e}, which approximates the AP^{e} gain from training it with a less strong Swin-T backbone. This suggests that the effects of CropFormer on fine-grained entity segmentation are highly complementary with the benefits introduced by more advanced backbones.

Backbone	Method	AP^{e}	$\mathrm{AP}^{\mathrm{e}}_{50}$	$\mathrm{AP}^{\mathrm{e}}_{75}$
Swin-Large	Mask2Former	46.2	63.7	47.5
	CropFormer	48.0	65.3	49.3

Table 8: The results of from training CropFormer with a stronger backbone.

Pretrain	Train	Pretrain-Train	Pretrain-Train
(COCO-E)	(EntitySeg)	(Test on COCO-E)	(Test on EntitySeg)
○	○	30.7→30.5	22.8→39.5
○	✓	30.7→30.4	22.8→41.0
✓	✓	32.3→30.7	21.4→41.4

Table 9: The ablation study on using CropFormer in the COCO-Pretraining stage. ○ and ✓ respectively indicate that CropFormer is used or not used in a particular stage. The left side of → respectively. indicates the AP^{e} obtained by the model pretrained on COCO-E, while the right side of → indicates the AP^{e} obtained by the model trained on EntitySeg after COCO-E pretraining.

CropFormer used in COCO-E pretraining In Table 9, we show the ablation study on whether using CropFormer in the COCO-E pretraining stage. And we see that using CropFormer in the COCO-E pretraining stage improves the final performance slightly on the EntitySeg dataset. It is likely because CropFormer is designed specifically to take advantage of the high-resolution images and mask annotations from EntitySeg dataset, instead of the low-resolution ones from COCO dataset.

3. Visualization

3.1. Inference results of CropFormer

In Figure 5, 6 and 7, we show the qualitative results of our CropFormer on EntitySeg test set. Based on the left-to-right order, the five sub-figures are the original image, the result of the full image (Batch-O), the result of ensembling four crops (Batch-C), the result of ensembling both the full image and four crops (Batch-OC), and the ground truth, respectively. The same colors across the 2nd, 3rd, and 4th subfigures correspond to the same respective queries, which demonstrate the effectiveness of batch-level queries enabled by our proposed CropFormer which has the association ability to connect the same entities across different image crops and the full image.

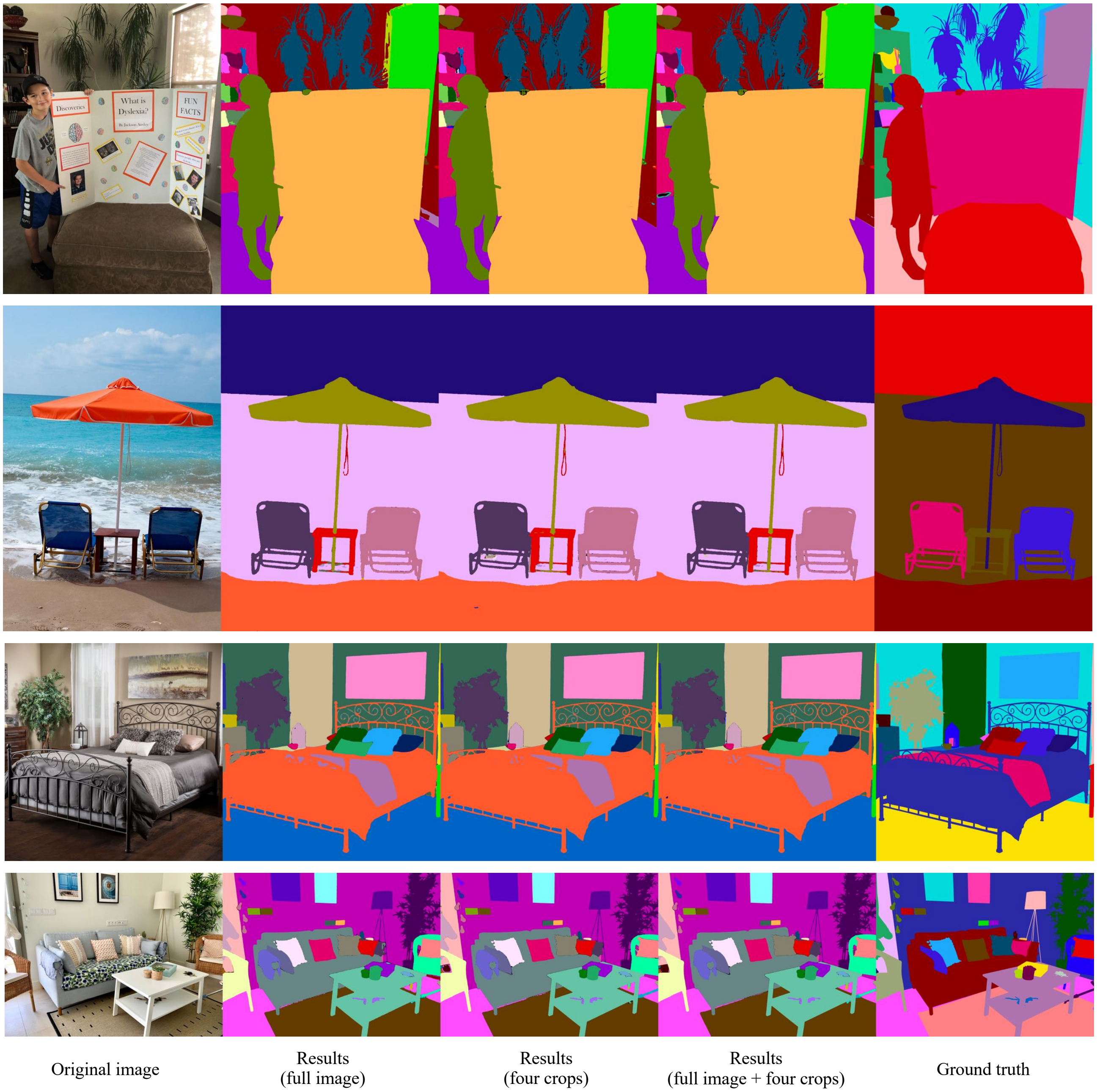

Figure 5: The visualization results from our CropFormer with Swin-L backbone which has 48.0 AP^e on EntitySeg test set.

3.2. Ground Truth of EntitySeg Dataset

In Figure 8, 9, 10, 11 and 16, we show more visualization examples from our EntitySeg Dataset to better demonstrate the high-quality nature of the dataset.

References

[1] 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. *PAMI*, 2017. 2

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

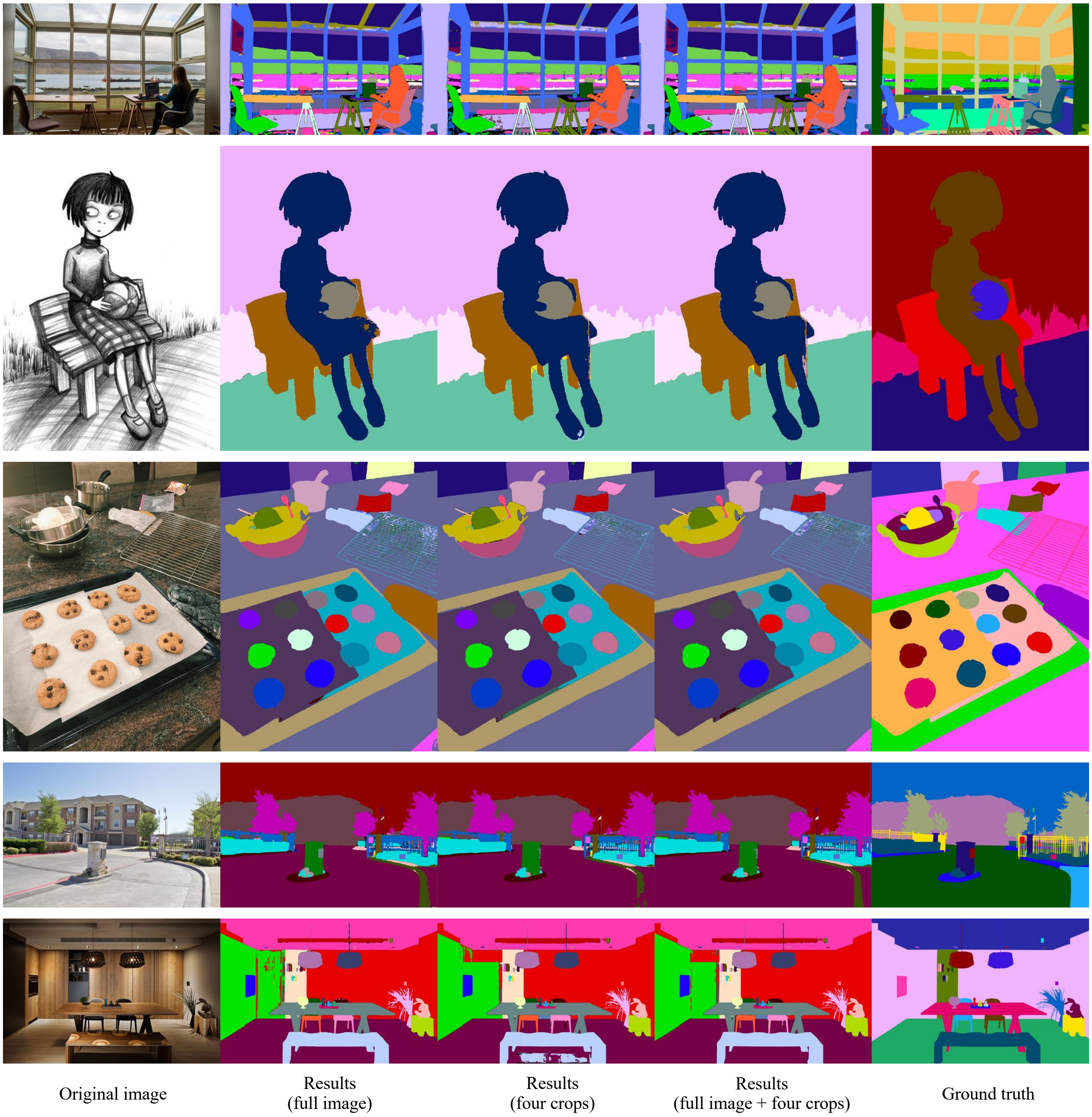

Figure 6: The visualization results from our CropFormer with Swin-L backbone which has 48.0 AP^e on EntitySeg test set.

[3] Bowen Cheng, Anwesa Choudhuri, Ishan Misra, Alexander Kirillov, Rohit Girdhar, and Alexander G Schwing. Mask2former for video instance segmentation. *arXiv preprint arXiv:2112.10764*, 2021. 1, 2

[4] Bowen Cheng, Ishan Misra, Alexander G Schwing, Alexander Kirillov, and Rohit Girdhar. Masked-attention mask transformer for universal image segmentation. In *CVPR*, 2022. 2, 3

[5] Agrim Gupta, Piotr Dollar, and Ross Girshick. Lvis: A dataset for large vocabulary instance segmentation. In *CVPR*, 2019. 1, 17

[6] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *ICCV*, 2017. 2, 3

[7] Alexander Kirillov, Ross Girshick, Kaiming He, and Piotr Dollár. Panoptic feature pyramid networks. In *CVPR*, 2019. 3

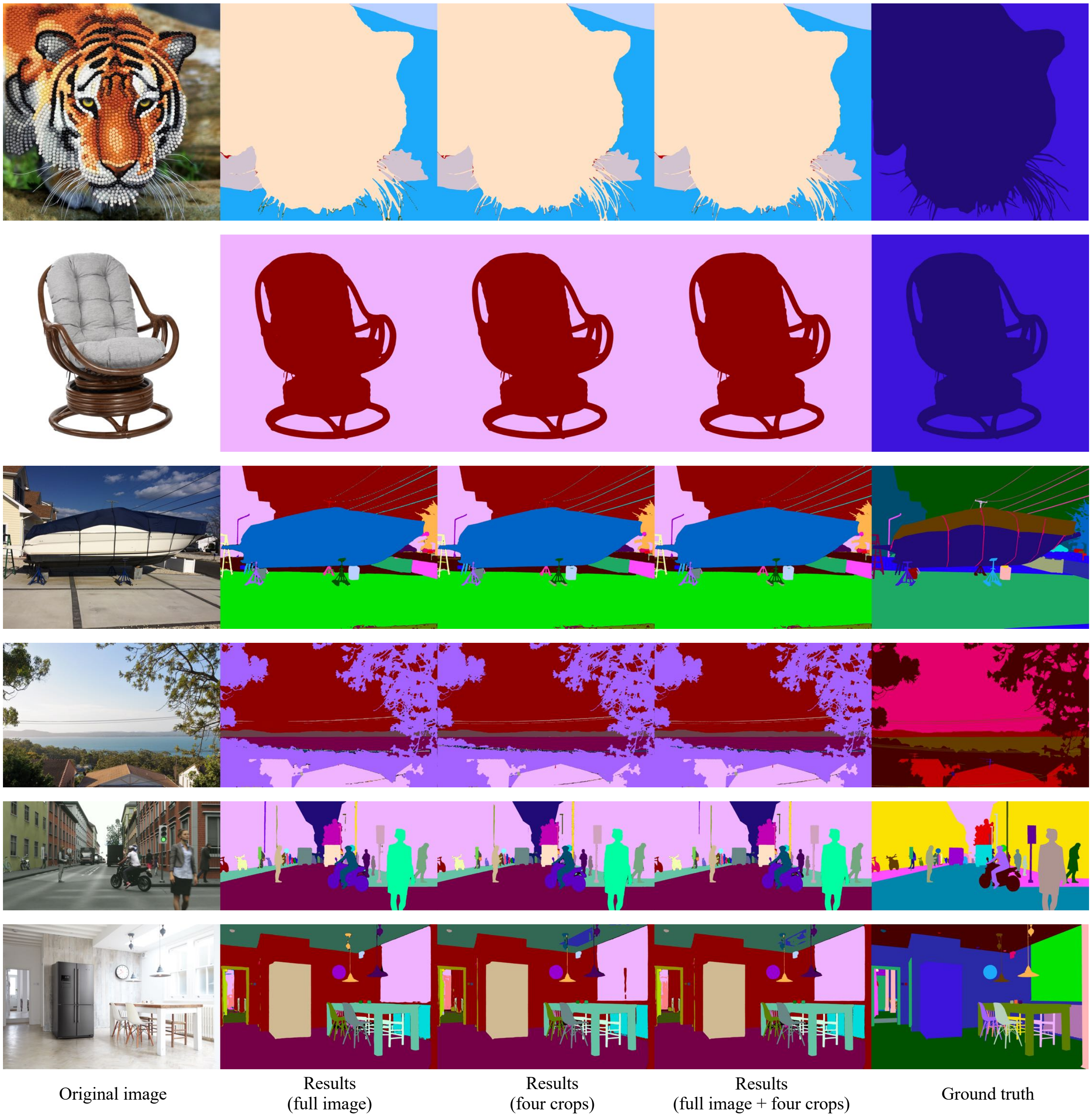

Figure 7: The visualization results from our CropFormer with Swin-L backbone which has 48.0 AP^e on EntitySeg test set.

[8] Trung-Nghia Le, Yubo Cao, Tan-Cong Nguyen, Minh-Quan Le, Khanh-Duy Nguyen, Thanh-Toan Do, Minh-Triet Tran, and Tam V Nguyen. Camouflaged instance segmentation in-the-wild: Dataset, method, and benchmark suite. *TIP*, 2021. 1, 14

[9] Xiang Li, Tianhan Wei, Yau Pun Chen, Yu-Wing Tai, and Chi-Keung Tang. Fss-1000: A 1000-class dataset for few-shot segmentation. In *CVPR*, 2020. 1, 15

[10] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *ECCV*, 2014. 1

[11] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF International Conference on*

Computer Vision, 2021. 4

[12] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *ICLR*, 2017. 2

[13] Lu Qi, Li Jiang, Shu Liu, Xiaoyong Shen, and Jiaya Jia. Amodal instance segmentation with kins dataset. In *CVPR*, 2019. 1

[14] Lu Qi, Jason Kuen, Yi Wang, Jiuxiang Gu, Hengshuang Zhao, Zhe Lin, Philip Torr, and Jiaya Jia. Open-world entity segmentation. *TPAMI*, 2022. 2, 3

[15] Markus Suchi, Timothy Patten, David Fischinger, and Markus Vincze. Easylabel: A semi-automatic pixel-wise object annotation tool for creating robotic RGB-D datasets. In *ICRA*, 2019. 1, 16

[16] Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Scene parsing through ade20k dataset. In *CVPR*, 2017. 1

[17] Yan Zhu, Yuandong Tian, Dimitris Metaxas, and Piotr Dollár. Semantic amodal segmentation. In *CVPR*, 2017. 1

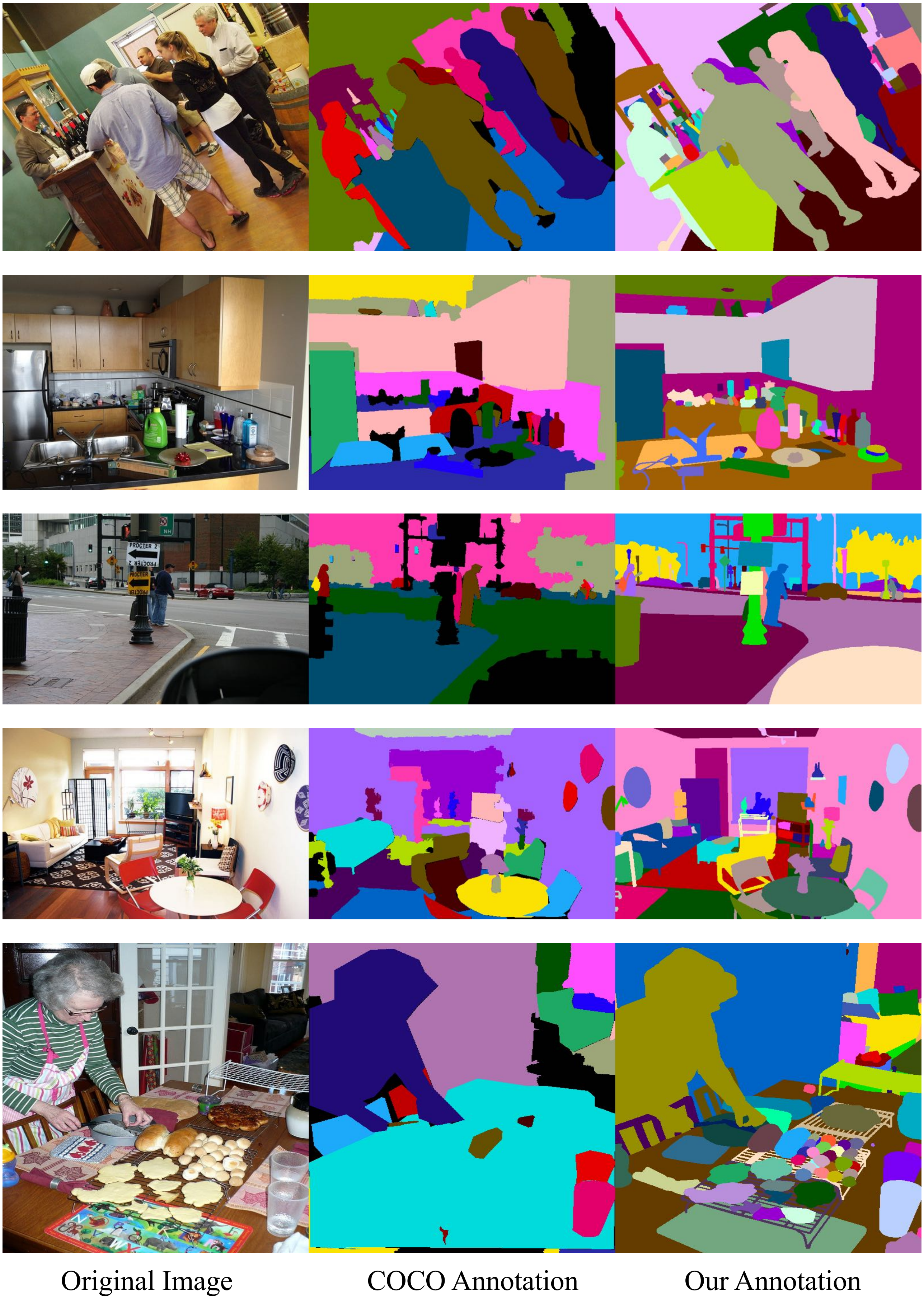

Figure 8: More visualization examples for comparison between COCO annotations and ours.

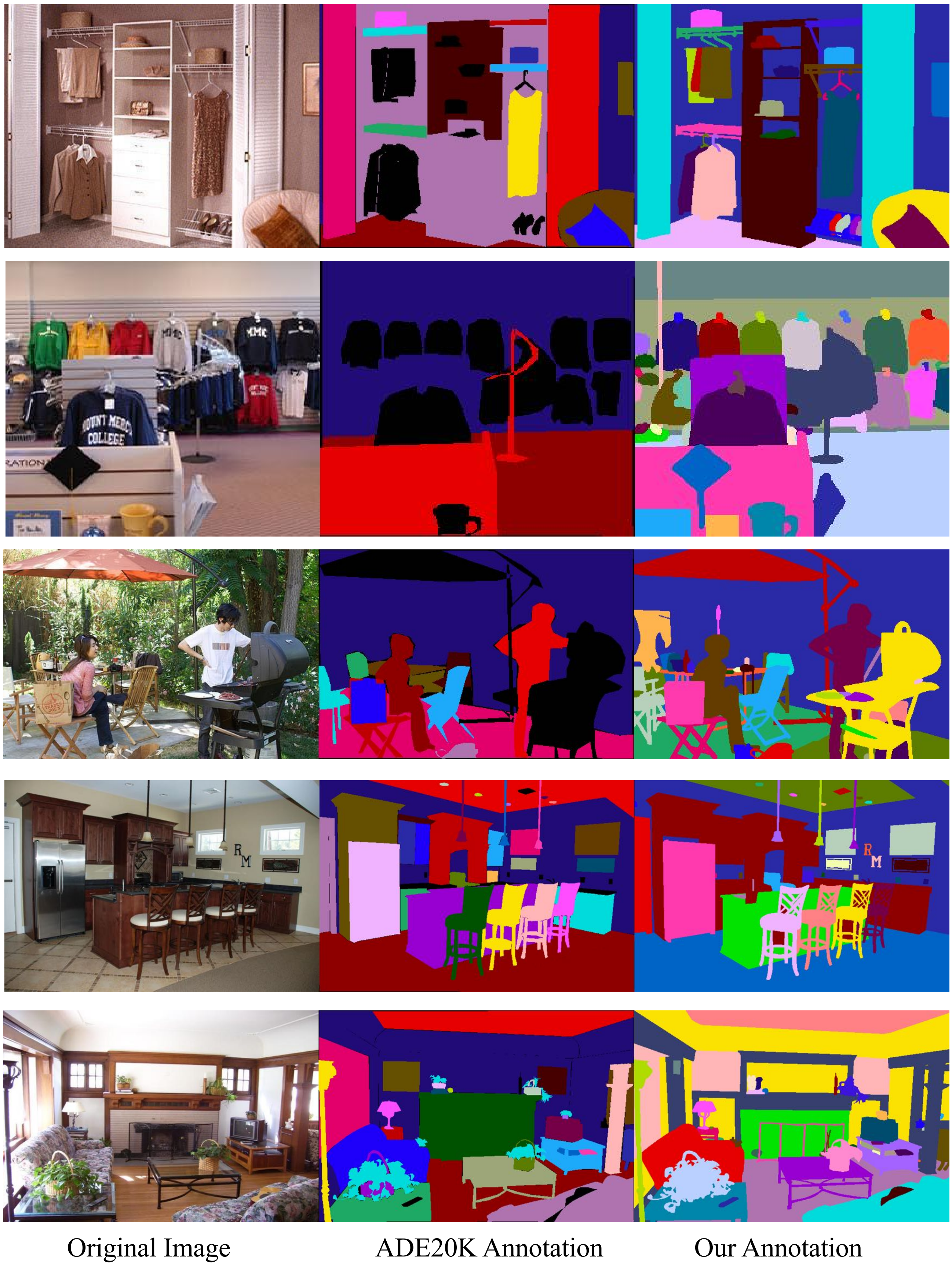

Figure 9: More visualization examples for comparison between ADE20K annotations and ours.

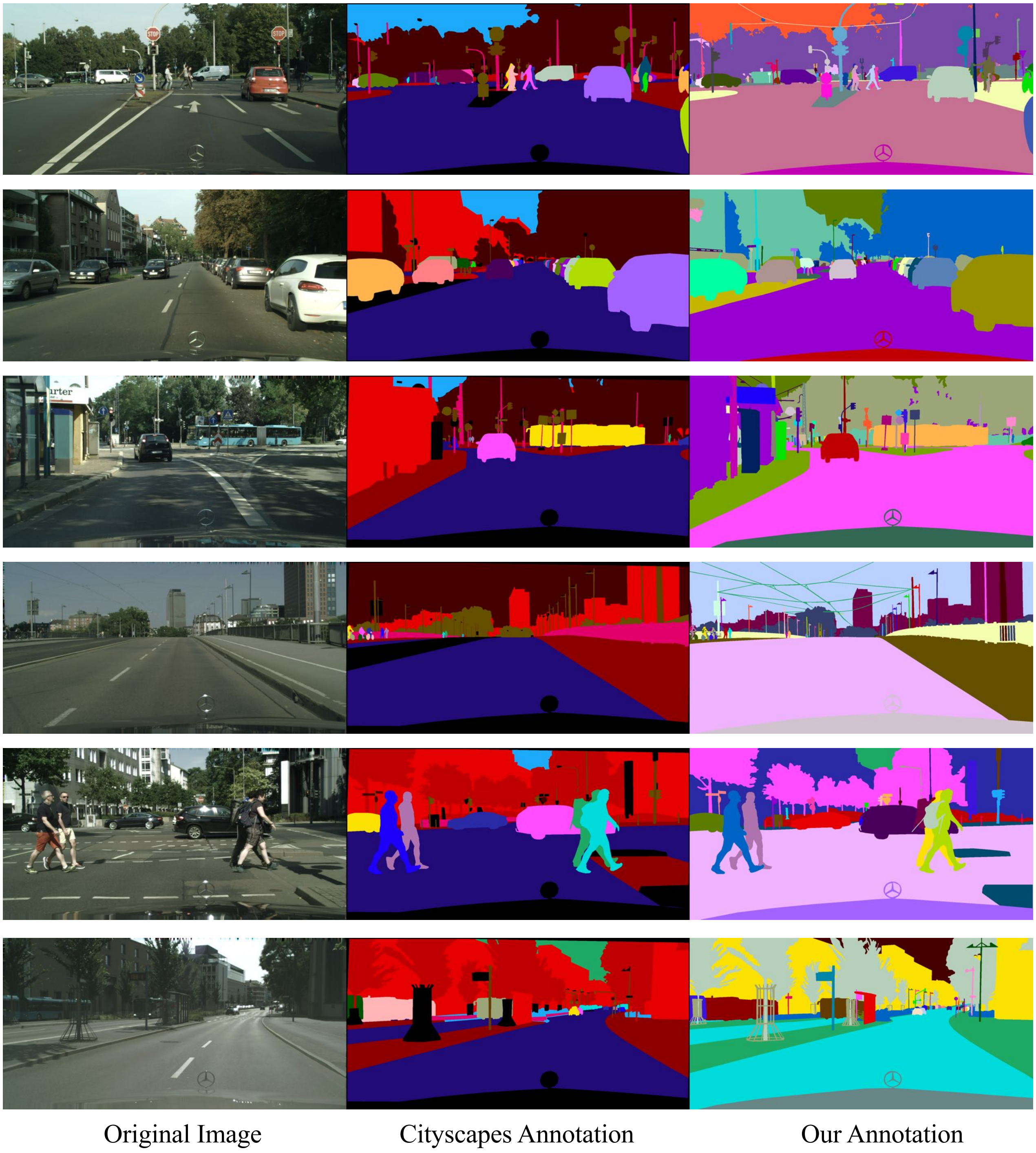

Figure 10: More visualization examples for comparison between Cityscapes annotations and ours.

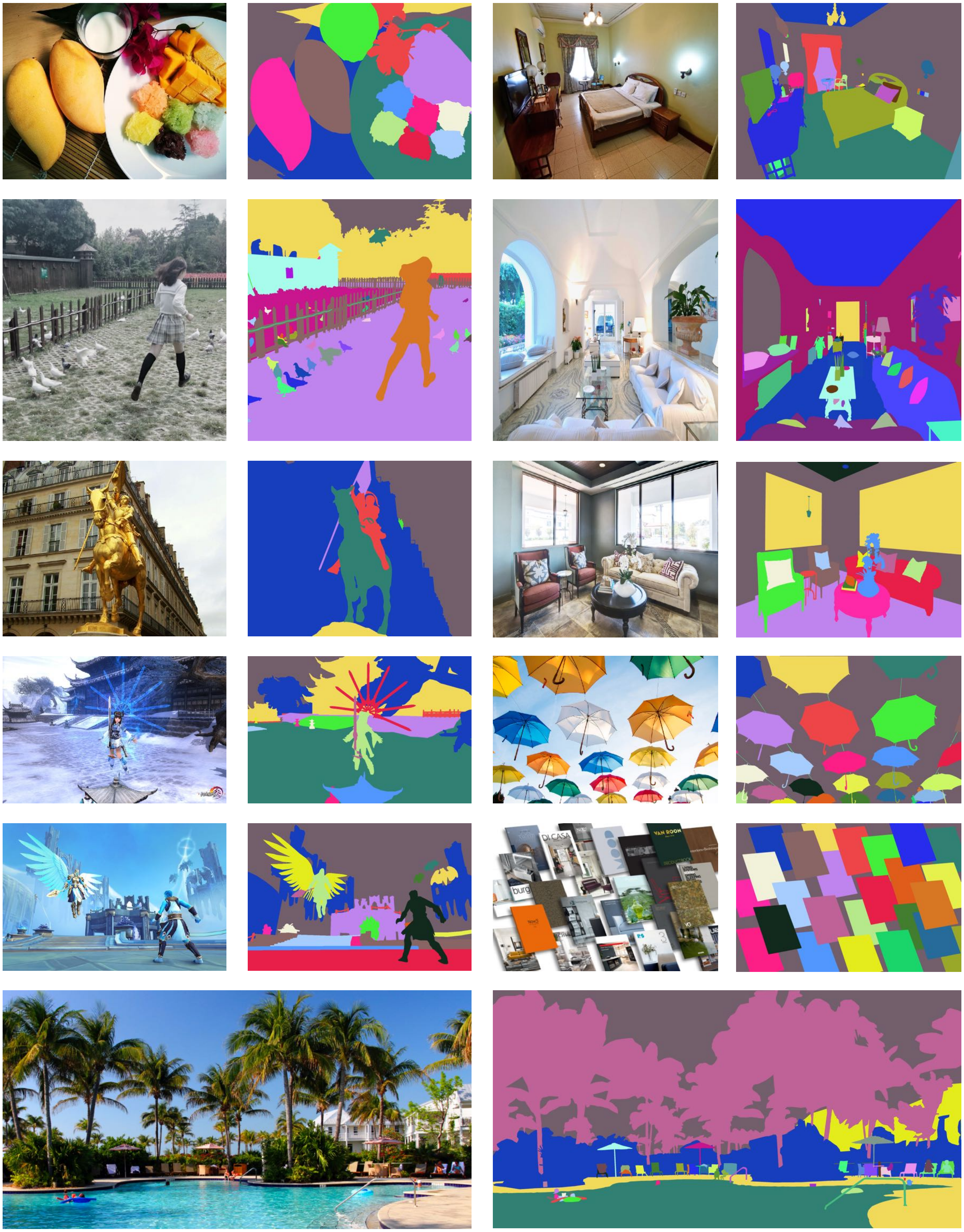

Figure 11: Visualization examples from our EntitySeg dataset.

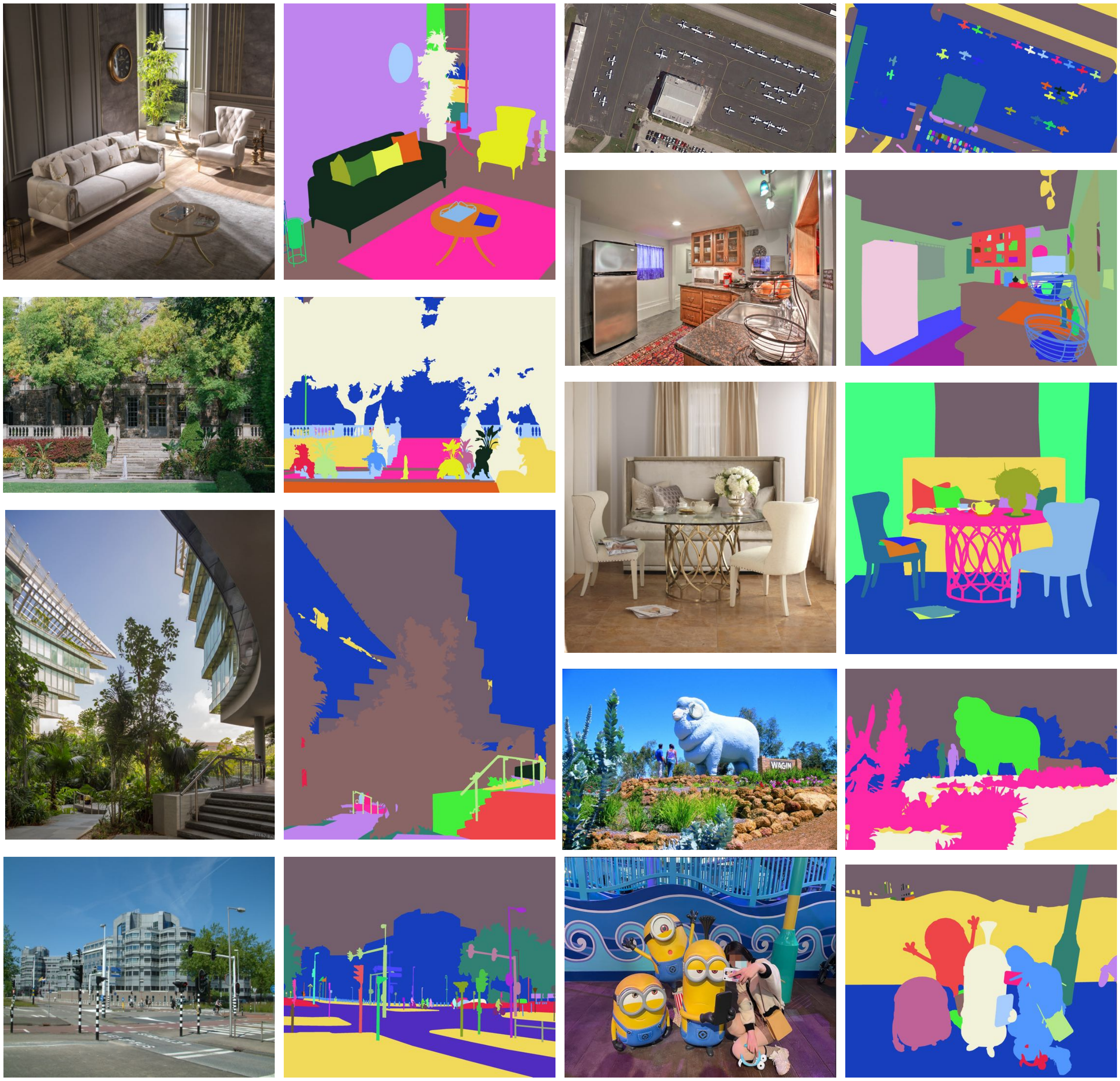

Figure 12: Visualization examples from our EntitySeg dataset.

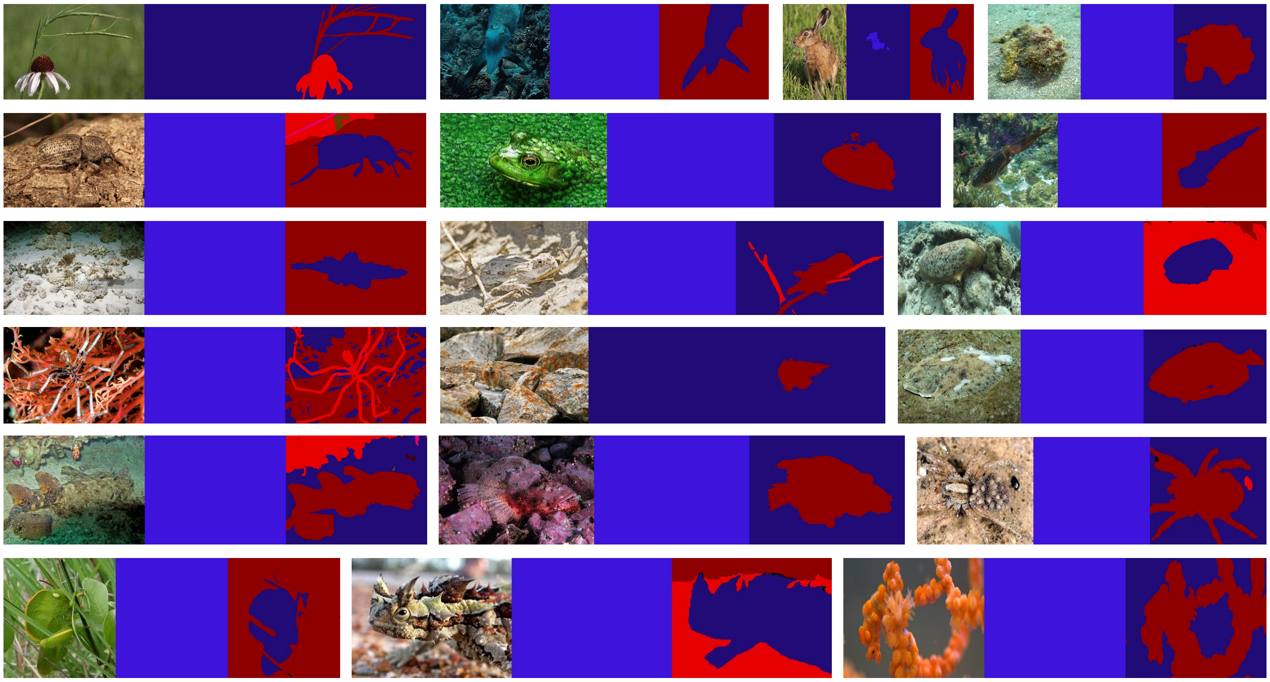

Figure 13: Visualization examples in CAMO [8] dataset. The left to right sub-figures is the original image, the visualization results of Mask2Former in COCO panoptic segmentation, and the visualization results of CropFormer in EntitySeg entity segmentation.

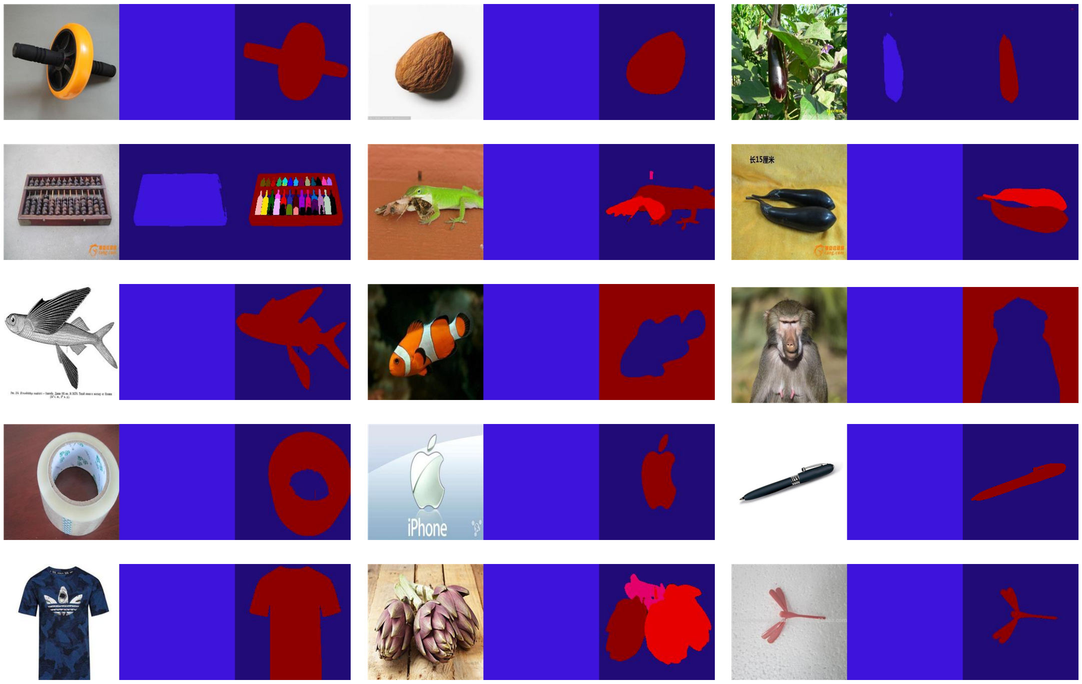

Figure 14: Visualization examples in FSS [9] dataset. The left to right sub-figures is the original image, the visualization results of Mask2Former in COCO panoptic segmentation, and the visualization results of CropFormer in EntitySeg entity segmentation.

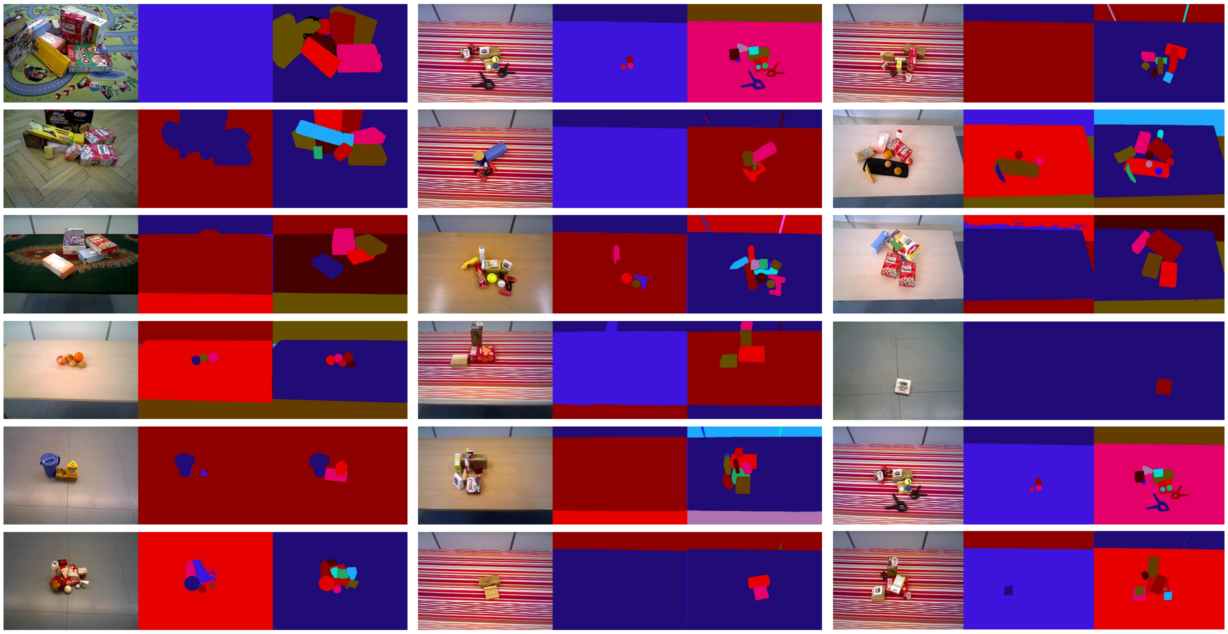

Figure 15: Visualization examples in OCID [15] dataset. The left to right sub-figures is the original image, the visualization results of Mask2Former in COCO panoptic segmentation, and the visualization results of CropFormer in EntitySeg entity segmentation.

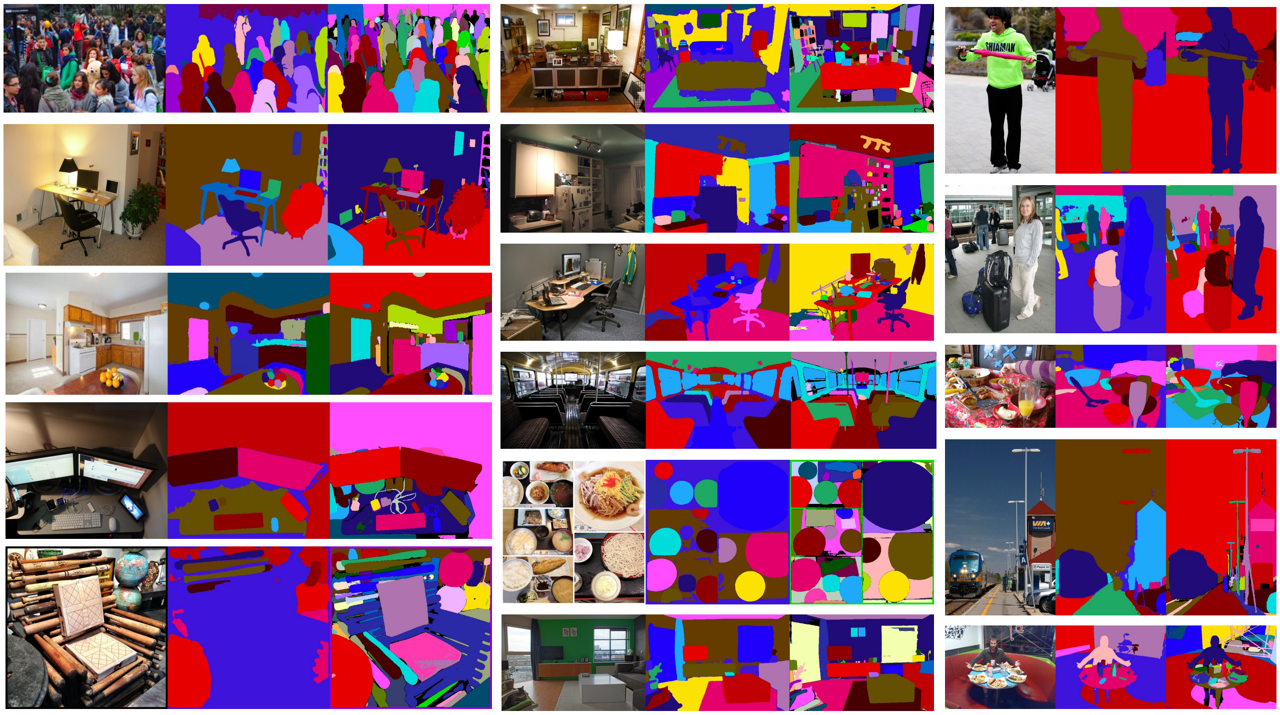

Figure 16: Visualization examples in LVIS [5] dataset. The left to right sub-figures is the original image, the visualization results of Mask2Former in COCO panoptic segmentation, and the visualization results of CropFormer in EntitySeg entity segmentation.