Extreme Point Supervised Instance Segmentation

Supplementary Material

In this supplementary material, we provide the following contents omitted from the main paper due to the space limit.

- Details of pseudo label generator architecture (Sec. A)
- More experimental details (Sec. B)
- Analysis on propagation (Sec. C)
- Impact of hyperparameters (Sec. D)
- Analysis on similarity extractor (Sec. E)
- Time and memory complexity of EXITS (Sec. F)
- More qualitative results (Sec. G)
- Limitation of the proposed method (Sec. H)

A. Details of Pseudo Label Generator Architecture

The network of pseudo label generator consists of the ViT encoder and the mask decoder. The architecture of ViT encoder follows the standard vision transformer design, which consists of 12 transformer layers. We do not use a class token, only the output features are fed into the mask decoder. The ViT encoder produces the image features $F \in \mathbb{R}^{N \times N \times D}$ from the cropped input image.

The mask decoder architecture consists of two heads, a pixel-wise head, and a prototype head, a design inspired by YOLACT [1]. The pixel-wise head comprises four convolutional layers, with bilinear interpolation used to upscale the feature resolution between the second and third convolutional layers. The feature map F goes through the pixel-wise head and resulting $F_{\text{pixel}} \in \mathbb{R}^{H \times W \times D/3}$. The prototype head consists of two fully connected layers with ReLU activation function and $D/3$ hidden dimensions. We use average pooling along spatial dimension of F, and it go through the prototype head and resulting $F_{\text{proto}} \in \mathbb{R}^{D/3}$

We produce mask feature map by inner product between F_{pixel} and F_{proto}, and the mask probability map $\mathbf{M}$ is given by

$$\mathbf{M} = \sigma(F_{\text{pixel}}\ F_{\text{proto}}) \tag{a1}$$

where σ denotes sigmoid function.

B. More Experimental Details

The hyperparameter δ, which is a small margin to push extreme points toward the center of the object, is set as follows: 24 for COCO [11], 16 for LVIS v1.0 [6], and 12 for PASCAL VOC [5]. Note that we push the extreme points with these margin on the resized image space, which is 512×512. The hyperparameters $\lambda_{mil}, \lambda_{point}, \lambda_{crf}$, which balance each loss term, are set as follows: 10, 0.5, 0.5 for COCO and LVIS v1.0, and 10, 0.05, 0.5 for PASCAL VOC. Note that MIL loss is applied only to samples where pseudo point supervision within the bounding box could not be provided using the point retrieval algorithm, *i.e.* $|\hat{\mathcal{P}}_{\text{FG}} \cup \hat{\mathcal{P}}_{\text{BG}}| = 0$. This accounts for only about 7% of the total images.

Index of layer	AP	AP_{50}	AP_{75}
all	39.5	66.7	40.4
#8	39.6	66.8	41.5
#10	**40.4**	**67.4**	41.4
#12	40.4	66.8	**41.8**

Table a1. Index of the transformer layer used for extracting similarity matrix. *all* refers to the results obtained by averaging the similarity matrices from all the transformer layers. The rows with gray background represent the values used in our model.

α	AP	AP_{50}	AP_{75}
1	34.0	63.0	32.4
2	36.1	64.3	35.0
3	**40.4**	**67.4**	**41.4**
4	39.8	66.4	40.0
∞	39.6	66.3	40.1

Table a2. Effect of α in propagation process. The rows with gray background represent the values used in our model.

C. Analysis on Propagation

Similarity matrix. We extract the semantic similarity between points from the multi-head self-attention of the transformer in the similarity extractor. Table a1 shows the impact of using different transformer layers for the extraction of the similarity matrix. Since earlier layers easily miss high-level semantics, averaging similarity matrices across all layers does not yield the best results. Therefore, we empirically choose to use 10^{th} layer for extracting the similarity matrix.

Effect of number of hops (α). Table a2 shows the effect of α in propagation process. when $\alpha = 1$, equivalent to generating pseudo point labels directly from the similarity matrix, there is a substantial drop in performance. This indicates that the propagation process is crucial for generating accurate pseudo point labels. We also measured the performance when the random walk propagation was continued until convergence after an unlimited number of steps, which is also known as the Absorbing Markov Chain [9, 14, 15]. It is calculated by

$$\mathbf{T}^{\infty} = (1-\beta)(\mathbf{I} - \beta\mathbf{T})^{-1} \tag{a2}$$

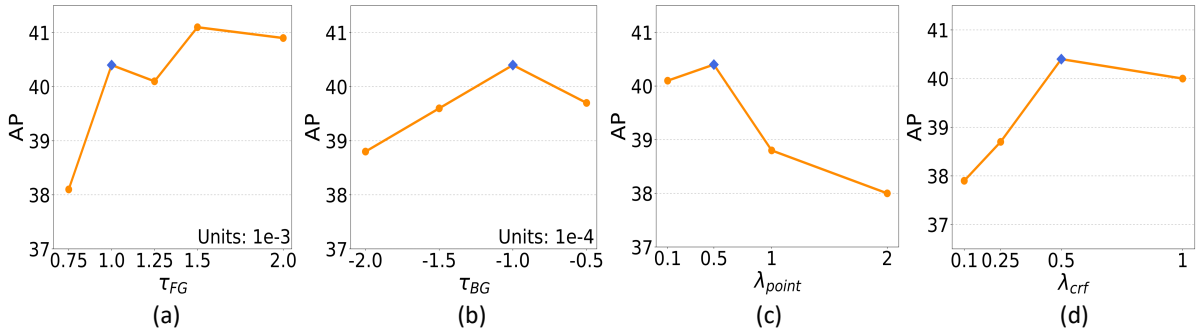

Figure a1. Average Precision (AP) of our second stage model varying hyperparameters. The model is evaluated on Pascal VOC using SOLOv2 [13] and ResNet50 [7] backbone. (a) The foreground point threshold τ_{FG}. (b) The background point threshold τ_{BG}. (c) Loss balancing term λ_{point}. (d) Loss balancing term λ_{crf}. The blue diamond marker indicates the value selected for our final model.

where $\mathbf{I}$ denotes identity matrix and $\beta \in [0, 1]$ denotes blending coefficient between propagated scores and initial scores. In cases where the random walk process converged, we observed the best performance at $\beta = 0.25$; however, it still did not outperform the results obtained after three propagation steps. Furthermore, considering the increased computational cost needed to compute Eq. (a2), we set the optimal value of α to 3.

D. Impact of Hyperparameters

Effect of τ_{FG} and τ_{BG}. In Fig. a1 (a) and (b), we demonstrate the effect of two thresholds, τ_{FG} and τ_{BG}, respectively. In the case of τ_{FG}, we observe that the hyperparameter value we selected are not optimal and there is potential for further performance improvement. This indicates that we did not exhaustively tune these parameters using the validation set.
Effect of loss balancing terms. In Fig. a1 (c) and (d), we show the instance segmentation results by using different loss coefficients, λ_{point} and λ_{crf}. Our model demonstrates robustness to these hyperparameter changes, surpassing the baseline [10] in every setting.

E. Analysis on Similarity Extractor

Effect of warm-up training epochs.[1] As shown in Table a3, more warm-up training leads to better performance by improving background-foreground discrimination of the similarity extractor.

	w/o warm-up	4 epochs warm-up	8 epochs warm-up (Ours)
mask AP	36.0	37.0	**37.3**

Table a3. Effect of warm-up training for similarity extractor.

Impact of pretrained weights.[1] In Table a4, we investigate the impact of pretrained weights for the similarity extractor, as it can have a significant impact on label propagation. In our experiments, using Masked Auto Encoder (MAE) [8] pretrained weights shows the best result. We hypothesize that the pixel-wise reconstruction training approach enhances the similarity extractor's ability to learn pixel-level relationships.

Pretrained weights	AP	AP_{50}	AP_{75}
MAE [8]	**37.3**	**64.4**	**37.8**
ImageNet 22k [4]	34.2	62.3	33.3
ImageNet 1k with DeiT [12]	35.8	62.4	35.9
DINO [2]	34.7	62.0	33.9

Table a4. Impact of pretrained weights for similarity extractor.

F. Time and Memory Complexity of EXITS

We compare the training time and number of parameters of EXITS with those of MAL, which is our strong baseline model. As shown in Table a5, EXITS shows a 20% increase in training time over MAL due to the warm-up of the similarity extractor and point retrieval process in Stage 1, with an increase in the number of parameters by 86M due to the similarity extractor module. Although EXITS requires an additional step for warm-up, the consequent increase in training time is only 5% of the total training time, and the similarity extractor does not affect the space-time complexity of inference.

8 NVIDIA A100 SXM4, COCO dataset, Mask2Former with Swin-Small					
		Stage 1			Stage 2
	Method	Warm-up SE	Pseudo label generator training	Pseudo label generation	Final model training
Training time	MAL [10]	-	23 hrs	1 hrs 10mins	71 hrs
	EXITS (Ours)	2 hrs	26 hrs	1 hrs 10mins	71 hrs
# params	MAL [10]	-	93M	-	69M
	EXITS (Ours)	86M	93M	-	69M

Table a5. Time and memory complexity comparison.

[1]For an experimental setup, we use the PASCAL VOC and SOLOv2 with ResNet50 backbone as final model. Also we follow $1\times$ schedule for the training configuration of mmdetection.

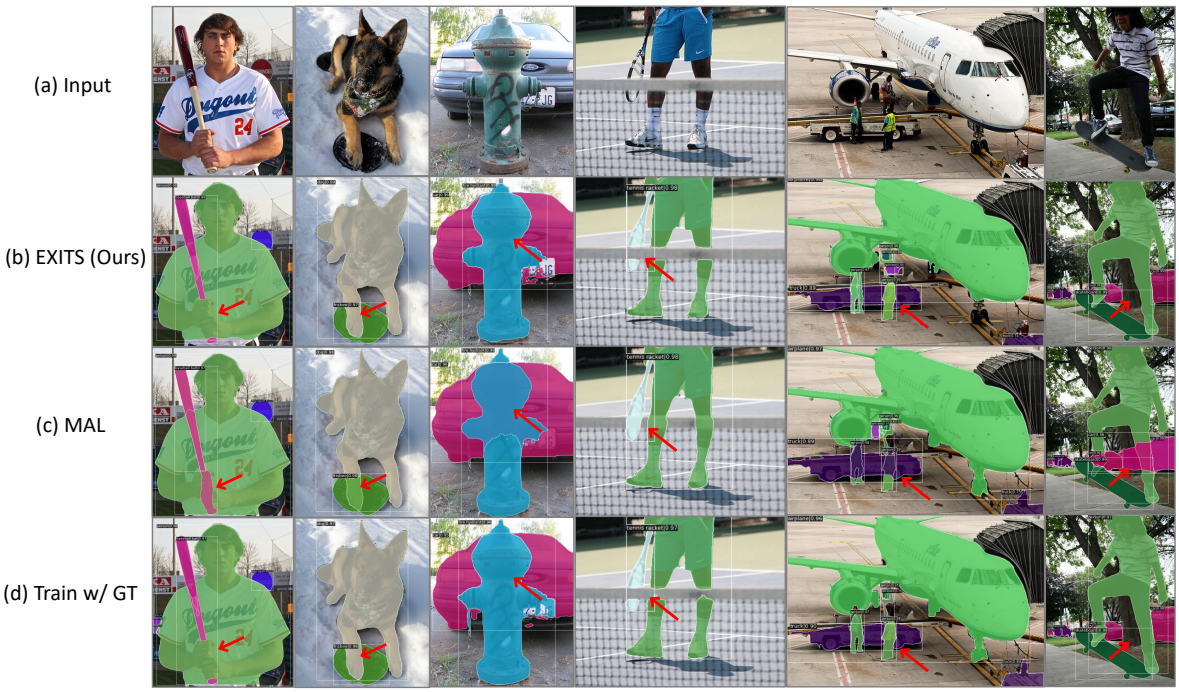

Figure a2. Qualitative comparison of instance segmentation results, especially for separated objects due to occlusions. (a) inputs (b) EXITS (ours), (c) model train with pseudo labels from MAL [10], (d) model train with ground-truth label. The red arrow points to the area where occlusion occurs.

Figure a3. Qualitative comparison of instance segmentation results in complex scenes. (a) inputs (b) EXITS (ours), (c) model train with pseudo labels from MAL [10], (d) model train with the ground-truth label.

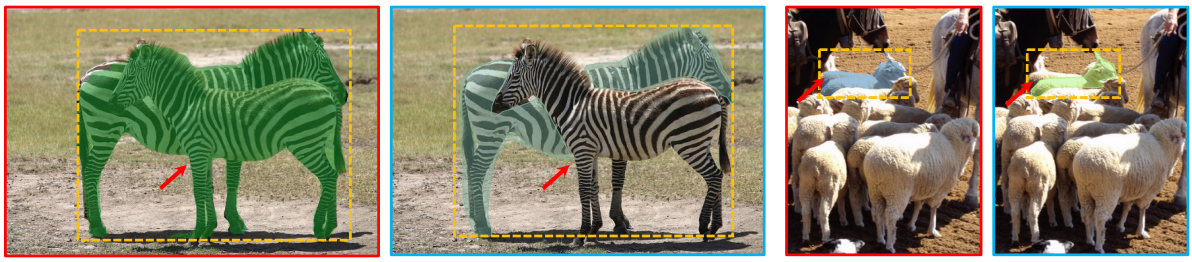

Figure a4. Failure cases of pseudo labels. Our pseudo label generator sometimes fails to predict when instances of the same class are encompassed by the same bounding box. Red box indicates generated pseudo label from the first stage of EXITS and blue box indicates ground-truth label.

G. More Qualitative Results

We visualize the final prediction results produced by Mask2Former [3] trained with pseudo labels from the pseudo label generator of EXITS using COCO `test-dev` set. We also visualized the results of the state-of-the-art box-supervised instance segmentation method, MAL [10], and the upper-bound model trained with the ground-truth labels as a comparison group. As can be seen in Fig. a2, the instance segmentation model trained with our method is capable of generating masks for separated objects, excluding the occluder. This demonstrates almost no difference compared to the results trained with ground-truth labels, while the model trained using pseudo labels generated by MAL struggles in these cases. Additionally, as illustrated in Fig. a3, the model trained with our pseudo labels thoroughly predicts even in complex scenes with numerous instances, in contrast to models trained using pseudo labels generated by MAL, which often fail in these scenarios.

H. Limitation

As observed in Fig. a4, our pseudo label generator often mispredicts when multiple objects of the same class are encompassed by the same bounding box. This issue arises as our point retrieval algorithm assigns pseudo point labels based on the results of the propagation difference between points outside of the bounding box and extreme points. One potential clue to solve this issue is to utilize the fact that even objects within the same bounding box have different extreme point annotations. However, this is beyond the scope of this work and will be left for future research.

References

[1] Daniel Bolya, Chong Zhou, Fanyi Xiao, and Yong Jae Lee. Yolact: Real-time instance segmentation. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9157–9166, 2019. 1

[2] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 9650–9660, 2021. 2

[3] Bowen Cheng, Ishan Misra, Alexander G Schwing, Alexander Kirillov, and Rohit Girdhar. Masked-attention mask transformer for universal image segmentation. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1290–1299, 2022. 4

[4] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *Proc. International Conference on Learning Representations (ICLR)*, 2021. 2

[5] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The Pascal Visual Object Classes (VOC) Challenge. *International Journal of Computer Vision (IJCV)*, 2010. 1

[6] Agrim Gupta, Piotr Dollar, and Ross Girshick. Lvis: A dataset for large vocabulary instance segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5356–5364, 2019. 1

[7] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 2

[8] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 16000–16009, 2022. 2

[9] Bowen Jiang, Lihe Zhang, Huchuan Lu, Chuan Yang, and Ming-Hsuan Yang. Saliency detection via absorbing markov chain. In *Proceedings of the IEEE international conference on computer vision*, pages 1665–1672, 2013. 1

[10] Shiyi Lan, Xitong Yang, Zhiding Yu, Zuxuan Wu, Jose M. Alvarez, and Anima Anandkumar. Vision transformers are good mask auto-labelers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 23745–23755, 2023. 2, 3, 4

[11] 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 *Proc. European Conference on Computer Vision (ECCV)*, 2014. 1

[12] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *International conference on machine learning*, pages 10347–10357. PMLR, 2021. 2

[13] Xinlong Wang, Rufeng Zhang, Tao Kong, Lei Li, and Chunhua Shen. Solov2: Dynamic and fast instance segmentation. *Advances in Neural information processing systems*, 33: 17721–17732, 2020. 2

[14] Donghun Yeo, Bohyung Han, and Joon Hee Han. Unsupervised co-activity detection from multiple videos using absorbing markov chain. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2016. 1

[15] Donghun Yeo, Jeany Son, Bohyung Han, and Joon Hee Han. Superpixel-based tracking-by-segmentation using markov chains. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1812–1821, 2017. 1