SimpSON: Simplifying Photo Cleanup with Single-Click Distracting Object Segmentation Network Supplementary Material

Chuong Huynh[1] Yuqian Zhou[2] Zhe Lin[2] Connelly Barnes[2]
Eli Shechtman[2] Sohrab Amirghodsi[2] Abhinav Shrivastava[1]
[1]University of Maryland, College Park [2]Adobe Research
[1]{chuonghm,abhinav}@cs.umd.edu [2]{yuqzhou,zlin,cobarnes,elishe,tamirgho}@adobe.com

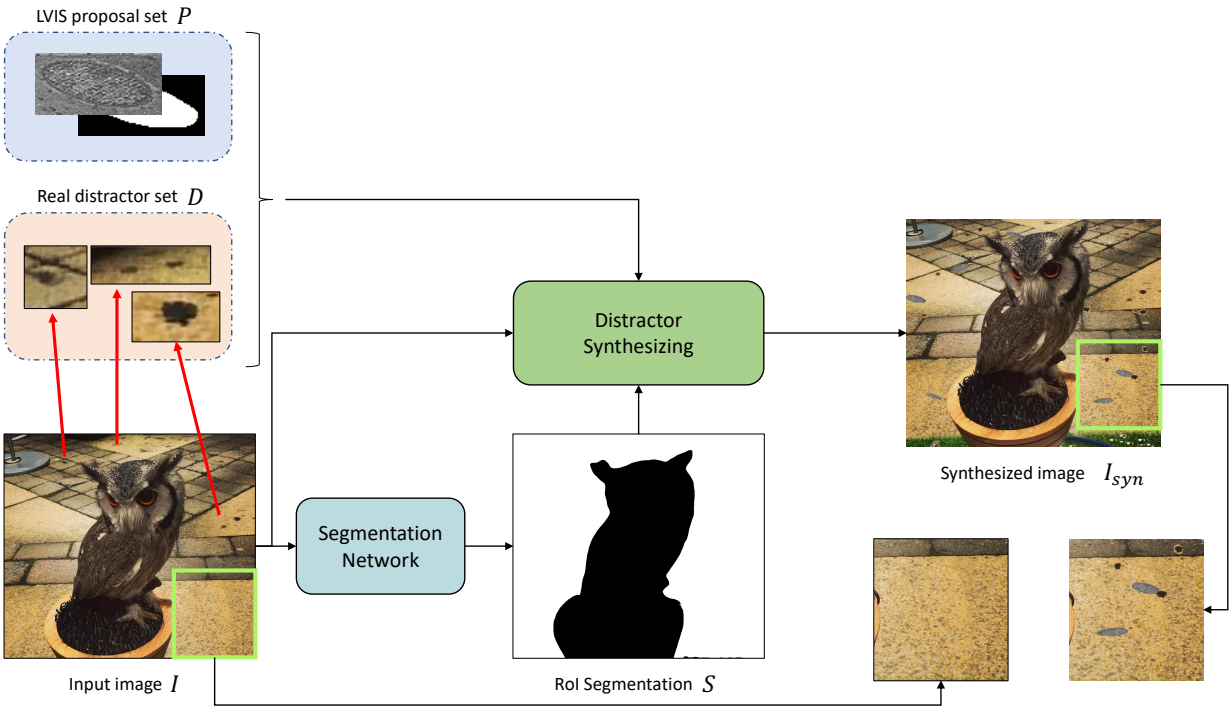


Figure 8. The overview of our distractor synthesis pipeline. The algorithm copies real distractors and objects from LVIS database to target RoI with the help of Segmentation Network. The green box indicates the zoomed-in region before and after the synthesis pipeline. (Best view in digital color image.)

1. Distractor Synthesis Pipeline

We present the algorithm of our dataset synthesis procedure in the Algorithm 2. The "DistractorSyn14k" and "DistractorSyn-Val" datasets are obtained as described in the main paper. For each Region of Interest (RoI), which refers to the segmentation stuff regions where we intent to copy the distractors to, we sample a set of distractor objects from our self-collected real distractor dataset, and also normal objects from LVIS database. We name them "distractor samples". The number of copies n for each sample is constrained by ensuring the total area of the pasted distractors is not exceeding 10% of the RoI. We apply random spatial and color augmentation including random flip, scale, rotate, brightness and contrast adjustment to those distractor samples before copying and pasting them. Gaussian smoothing is also applied while blending them with the image to avoid sharp composition boundary artifacts. It will avoid the model from overfitting the seams. The positions of the

Algorithm 2: Distractor Synthesis Pipeline

Data:
I: Image
$D = \{(p_i, m_i)\}$: Real sample distractor set
$S = \{(r_i, s_i)\}$: RoI segmentations
r_i: RoI mask
s_i :RoI label
$P = \{(p_i, m_i, c_i)\}$: LVIS sample small-object set
p_i: Sample image
m_i: Sample mask
c_i: Sample RoI label
Result: I_{syn}, D_{syn}
$D_{syn} \leftarrow D$;
$I_{syn} \leftarrow I$;
for r_i, s_i *in* S **do**
  $D' \leftarrow$ distractors inside r_i;
  $r'_i \leftarrow r_i \setminus \{m_i | m_i \in D'\}$;
  if $|D'| < 3$ **then**
    $P' \leftarrow$ random from P s.t. $c_i = s_i$;
    $D' \leftarrow D' \cup P'$;
  $\bar{D'} \leftarrow$ total area of distractors in D' ;
  $n \leftarrow \lceil 10\% \times area(r'_i)/\bar{D'} \rceil$;
  for d *in* D' **do**
    $p_j \leftarrow$ image crop in d (real or LVIS);
    $m_j \leftarrow$ mask crop in d (real or LVIS);
    for k *in* $1..n$ **do**
      $\delta_i \leftarrow$ distance map of r'_i;
      $x, y \leftarrow$ center coordinate in δ_i;
      $p'_j, m'_j \leftarrow$augment p_j, m_j ;
      Move p'_j, m'_j to (x, y);
      $I'_{syn} \leftarrow$ blend p'_j to I_{syn};
      if $\|hist(I'_{syn}(x, y)) - hist(I_{syn}(x, y))\| > 0.001$ **then**
        $I_{syn} \leftarrow I'_{syn}$;
        $D_{syn} \leftarrow D_{syn} \cup \{(p'_j, m'_j)\}$;
        $r'_i \leftarrow r_i \setminus m'_j$;
    end
  end
end

distractor placement can be decided by the distance map peak value, and the details are shown in the Algorithm 2.

Fig. 8 illustrates the entire process and intermediate results of our data synthesis pipeline. We tend to make the synthesized images look realistic, though the compositional artifacts still exist. The resulting images still look natural enough since the distractor samples are from real distractor datasets and real small objects. According to our experiments, using synthetic data will not greatly influence the generalization ability of the model to real images. More image harmonization and deep composition techniques can be further explored in the future work. Some other results are shown in Fig. 9. Our synthetic images are comparable with real distractor ones which can be used to train and evaluate the CPN module. Our images contains many repeated distractors with diversity in appearances and categories, simulating the properties of distractors in the real-world.

2. More details about the Distractor20K dataset

Our dataset Distractor20K is collected to have 107 different categories belonging to 28 super categories. There exist known and unknown categories that label unrecognizable regions or not in the defined category set. Both stuff and things are considered distractors in our dataset; in detail, there are 79 object categories and 28 non-object categories. If we follow the LVIS dataset to split the categories based on their frequencies, there are 13 rare, 25 common, and 69 frequent categories in our collected dataset. The number of instances and images for each category can be seen in Fig. 10. Rare categories appear in a maximum of ten images in the entire dataset, while common categories have less than 100 images. All the category names are hidden for commercial use.

The Fig. 11 illustrates the histogram of a number of distractor categories for each image. An image can have up to 25 categories of 15 super categories, and the average contains 3-5 categories.

Figure 12 shows the histogram of the ratio of the instance mask size over the image size in our Distractor20K. According to the statistics, we found that a significant amount of distracting instances are medium and small, as tiny as only occupying 0.01 of the image. Those distractors can be stones on the ground, graffiti on the wall, leaves on the water's surface, fire valves on the ceiling, etc. Photographers have the requests to clean those things from the photos, while existing segmentation models do not help segment them automatically.

3. More Results of 1C-DSN

3.1. Existence of Negative Clicks

Additional experiments are executed with FocalClick and RiTM on the LVIS dataset. To validate the capability of those models with the one-click procedure, we finetuned the models on LVIS dataset with only one positive click as input. In testing, the click generator is customized to produce positive clicks only. No additional clicks are added when there are only a few false negatives at the boundary because clicking at the boundary can cause severe accuracy degrading due to the precision of click positions.

The Fig. 13 shows the performance of Interactive Segmentation models in two different clicking strategies. The public weights are used in the positive-negative strategy,

Synthesized Distractors
(Top to Bottom: Origin image, Synthesized image, Zoomed Patches)

Real Distractors

Figure 9. Examples of synthesized and real distractors. Similar to real photos, our synthesized database contains distractors with different appearances and categories. (Best view in digital color image.)

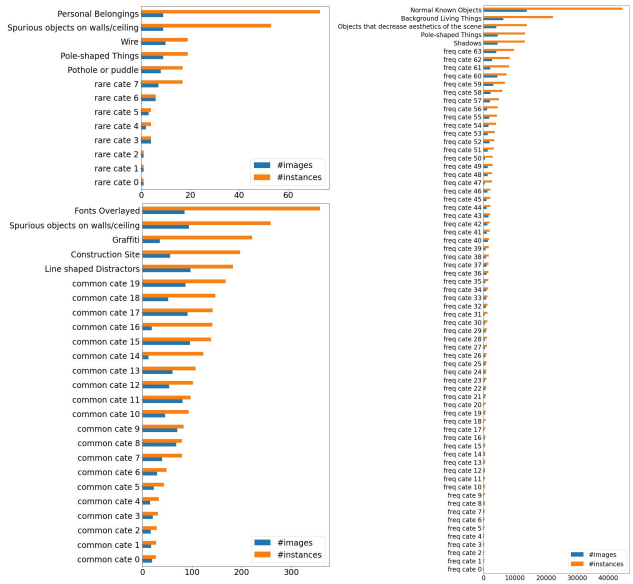


Figure 10. Frequencies of categories in the Distractor20K

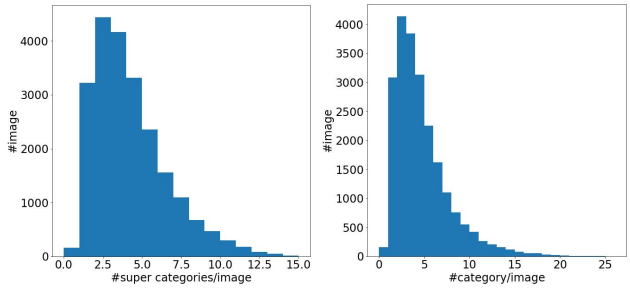


Figure 11. The number of categories in each image in the Distractor20K

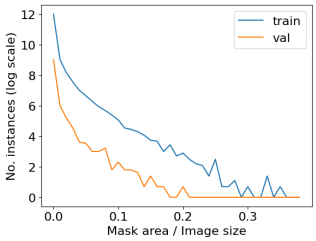


Figure 12. The number of instances regards to the object size in our distractor datasets.

while our finetuned models are tested with positive clicks only. All frameworks using the positive-click strategy, including ours, do not receive large improvements without negative clicks to refine the boundary. In contrast, the

performance of RiTM and FocalClick using the positive-negative strategy increases by adding more negative clicks. It demonstrates the existence of negative clicks indeed helps to improve the overall masking performance with multiple clicks from the users.

However, for both RiTM and FocalClick, the existence of negative clicks also does harm to the performance when there is only one single positive click. As shown in Fig. 13, in the first positive click, the new finetuned models achieve higher results than the ones using negative clicks in training. As we mentioned in the main paper, distractors are mostly medium and small objects, and users prefer to use fewer clicks for them. Therefore, the positive-click strategy is more suitable for distractor removal and photo-cleaning applications. Following this core idea and under fair comparison, our framework achieves reasonable performance with one positive click than the other two interactive segmentation models.

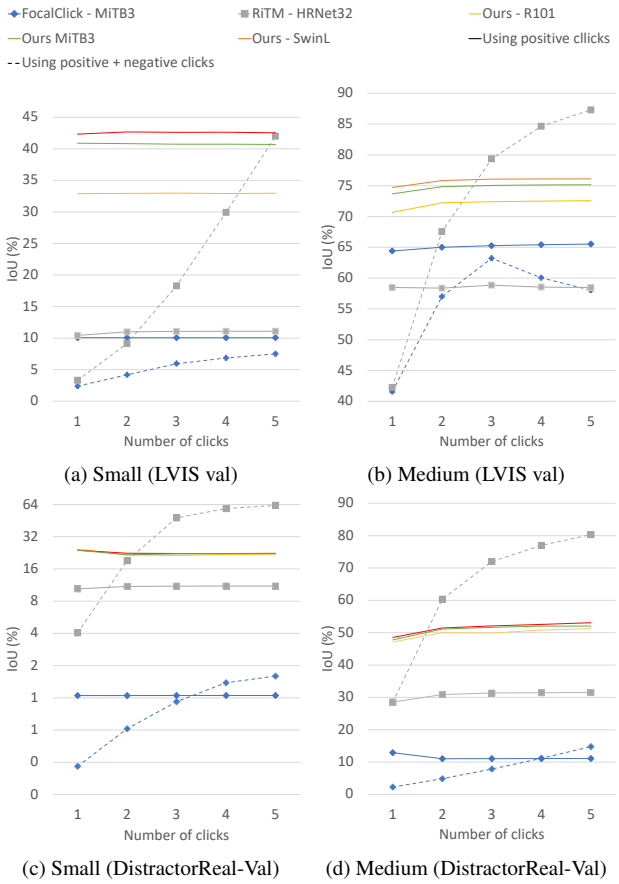
Some qualitative results of different click samplers on LVIS val set are shown in Fig. 14. The backbone used is MiT-B3. Other models with more negative clicks help to improve the detailed segmentation boundary and achieve overall better results. However, our model obtains better masking quality than one-positive-click finetuned FocalClick and RiTM, and requires less user effort to select. The one-click system also helps with group selection scenarios later in the pipeline.

3.2. Randomness of Clicks

To evaluate the robustness of models with different click positions, we increase the randomness of clicks surrounding the object's center. Let d_{max} be the peak value in the distance map Δ, which localizes in the center of the object. The randomness level r defines a threshold such that the clicks are placed among all positions with $d_i \geq (1.0 - r) \times d_{max}$. When clicks are always at the object center, the randomness r is zero. Otherwise, when the click can be anywhere in the mask, the randomness $r = 1.0$. The Fig. 15 presents the decrease in performance when increasing the randomness of click positions. All models are finetuned on LVIS dataset with one positive click procedure. With small objects, the randomness level does not affect the IoU significantly. However, the performance goes down quickly with medium objects when increasing the randomness from 50% to 80%. Our performance still remains high with a large randomness level, indicating the model's robustness to click randomness.

Click Embedding	IDS	PVM	AP (%)	AR (%)
			28.9	39.2
		✓	29.9	39.0
	✓		23.0	42.2
	✓	✓	26.7	42.0
✓			34.1	41.0
✓		✓	33.7	39.0
✓	✓		34.4	47.0
✓	✓	✓	42.4	49.7

Table 6. Performance of EntitySeg (SwinL) and 1C-DSN (SwinL) on DistractorSyn-Val set. The click embedding helps producing better exemplar masks then improve the performance in finding similar distractors.



(a) Small (LVIS val) (b) Medium (LVIS val)

(c) Small (DistractorReal-Val) (d) Medium (DistractorReal-Val)

Figure 13. Performance of Interactive Segmentation with different click procedures. Without negative clicks, the current Interactive Segmentation models cannot achieve higher performance when increasing the number of clicks. Our models outperform the model with only one positive click, which better suits the distractor selection task.

3.3. Similarity Findings without Click Embedding

We show in Table 6 the performance of EntitySeg [10] model in similarity finding and group selection. This experiment aims to ensure that our one-click-based segmentation model is necessary for the group selection scheme.

We simply apply our CPN and PVM modules to an En-

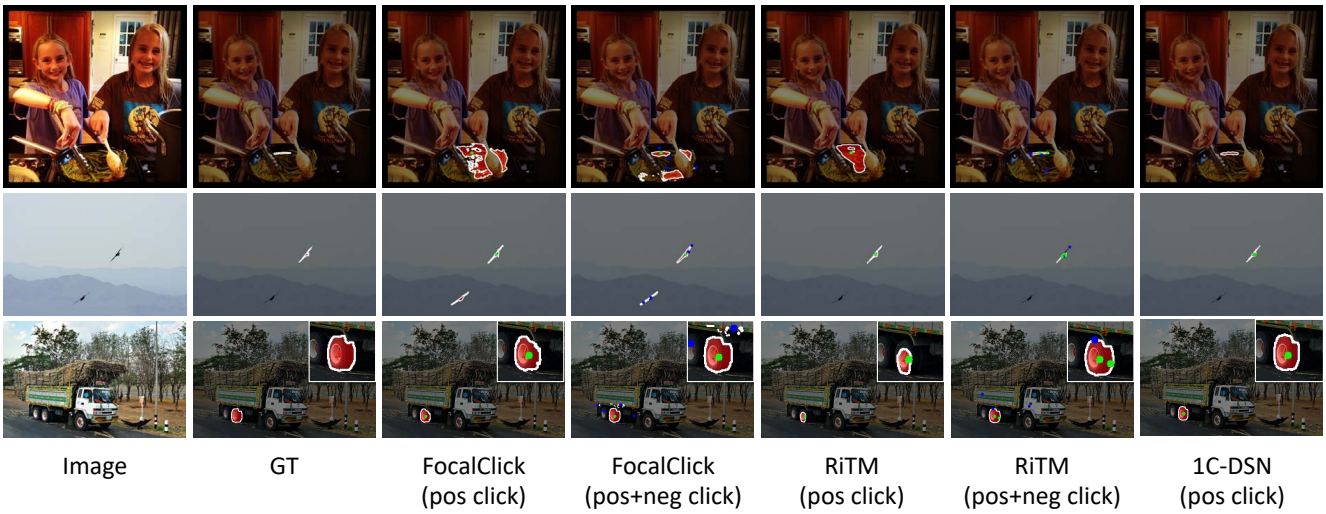
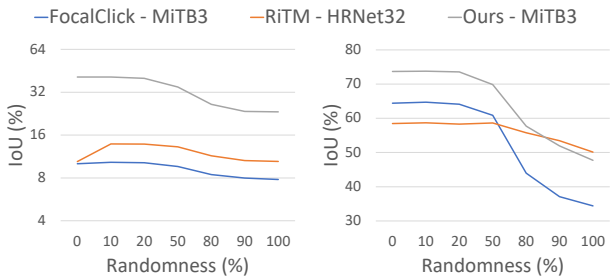


Figure 14. Qualitative results of different click samplers and frameworks on LVIS val set. With only one positive click, our model 1C-DSN can select perfect masks of objects while other frameworks require negative clicks. Except for RiTM using HRNet32, other frameworks use MiT-B3 as backbones. (Green: positive click, Blue: negative click. Best view in digital color)



(a) Small (LVIS val) (b) Medium (LVIS val)

Figure 15. The performance of one-positive-click models drops when increasing the randomness of click locations.

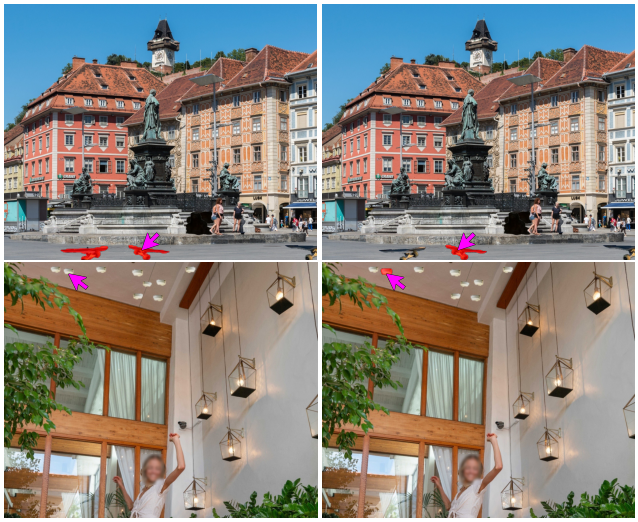


Figure 16. Typical failure cases of EntitySeg (left) without click embedding on DistractorSyn-Val set. EntitySeg baseline may over-predict or under-predict the masks making the self-similarity mining not reliable enough. Our framework (right) still has good predicted mask. Both models using Swin-Large backbone and are trained on Distractor20K. (Best view in color)

titySeg model (without click embedding inputs) finetuned on Distractor20K dataset. We use the model predictions on one image to extract exemplar masks for the CPN and PVM modules. For the clicks which do not have corresponding masks, no further steps are executed and then the prediction output will be empty. Without running IDS or PVM, the EntitySeg model provides lower recall and precision than our models. The Fig. 16 gives typical failure cases of EntitySeg models. Masks can be over-segmented or wrongly detected by EntitySeg without click inputs. Therefore, adding either IDS or PVM steps does not improve the final results for EntitySeg baseline model.

4. Compare CPN with Other Self-similarity Methods

Related Works. Self-similarity is a commonly used technique in vision tasks to find repeating patterns and learn better globally consistent features. It has been used in various models, including non-local network [22], contextual attention [24], self-attention [15, 19], and transformer-related models [3, 6, 21, 23]. Attention has also shown to benefit almost all vision models, including image super-resolution [7], object detection [20, 26], and image synthesis [2, 4, 5], among others [8, 9, 13, 14, 18]. The most similar work re-

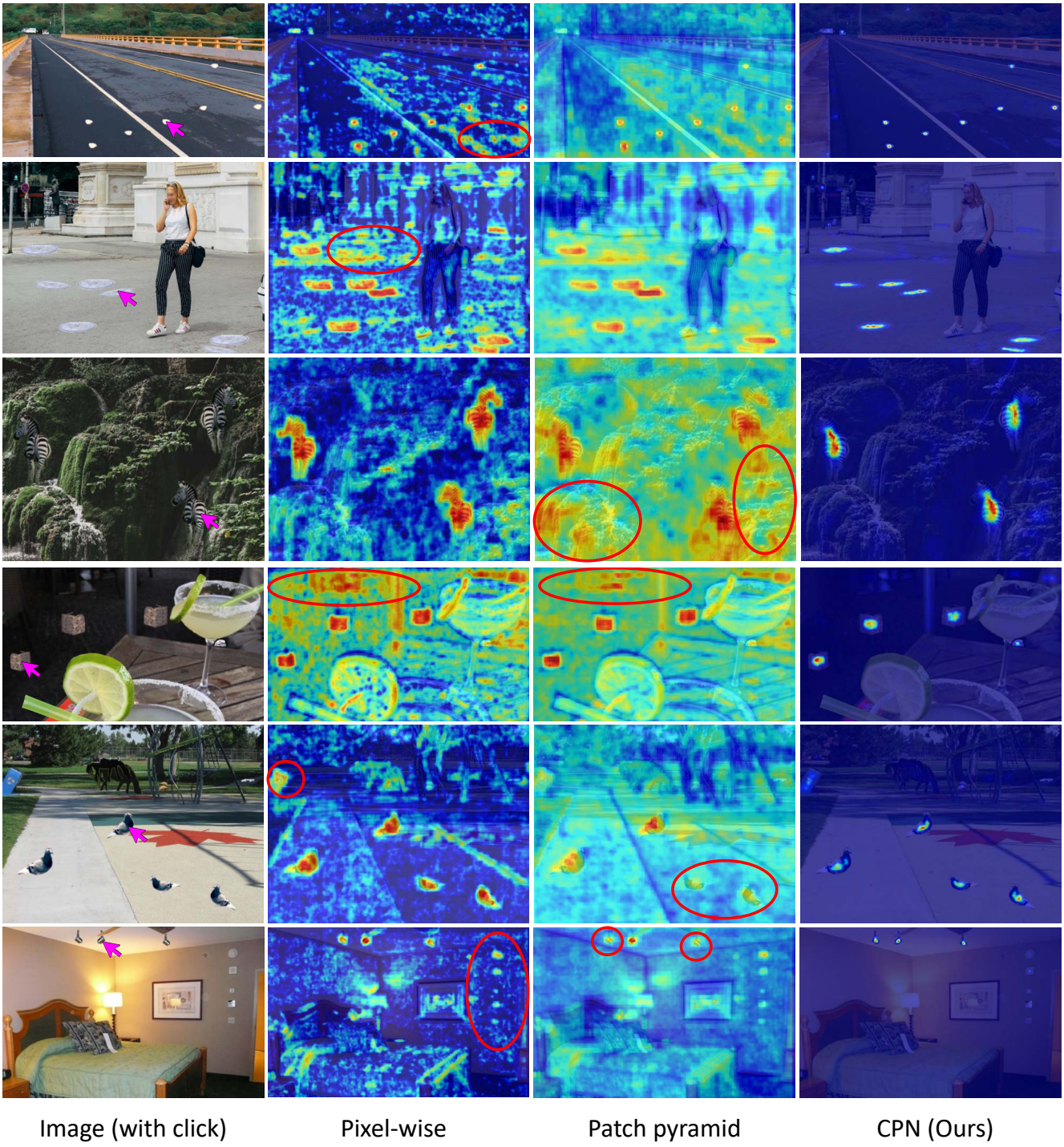


Figure 17. Our framework produces cleaner heatmaps than other naive self-similarity methods. Red eclipses indicate the false or missed detection regions. Images are from DistractorSyn-Val dataset. (Best view in color)

lated to our task is visual counting [1, 11, 12, 16, 17], which aims to localize all similar objects within the same images by actively sampling a few of them. However, visual counting works do not require masking the objects, and the targets of visual counting are usually the main subjects of the photos. In our task, we may face more complicated and challenging image contexts, where diverse context yields a high false positive detection rate. To address this issue, we

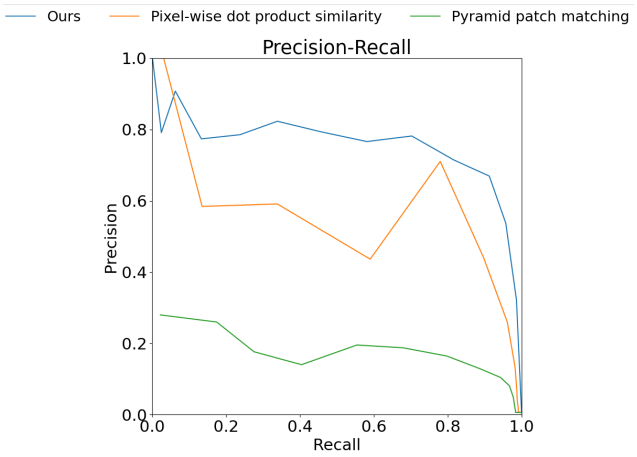
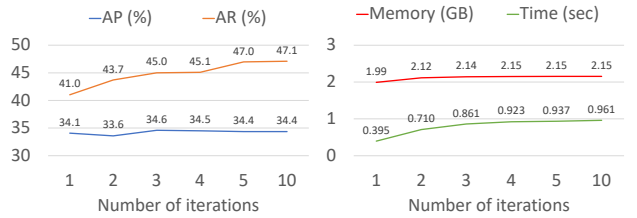


Figure 18. Precision-Recall of heatmaps predicted by different similarity finding approaches on DistractorSyn-Val. Our models has higher precision than other methods.

leverage a transformer decoder to learn cross-scale attention and generate the attention heatmap, along with an additional verification scheme to remove false positives.

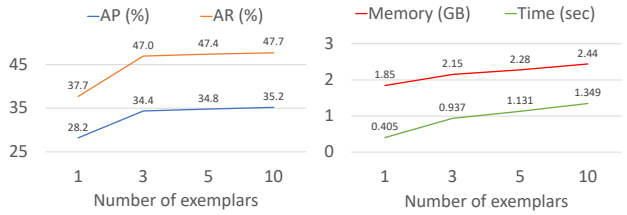
Point Detection Precision and Recall. To evaluate the performance of the CPN module in similarity heatmap generation, we use Area Under the Curve of Precision-Recall (AUC-PR) on the similarity heatmap. The metric is used in Table 4. A click located at the ground-truth mask region is counted as a true positive; otherwise, it is counted as a false positive. The precision is the proportion of true positive clicks and the total predicted clicks. The recall is the ratio between the number of masks having predicted clicks over the total of masks. We compute precision and recall in different thresholds to get the curve between them.

Pixel-wise Dot Product Similarity. Since there are no previous works on distractor similarity findings and it is not fair enough to directly compare with visual counting works, we can only compare our CPN with some naive baselines. With the finer feature map $X_1 \in \mathbb{R}^{h \times w \times d}$ and the mask M,



(a) Average Precision Recall (b) Computational cost

Figure 19. The increasing number of iterations improves the Average Recall while maintaining the Average Precision. The computational cost is also increased. The performance is saturated after about five iterations.



(a) Average Precision Recall (b) Computational cost

Figure 20. The performance of IDS increases proportionally with the number of exemplars and the computational cost.

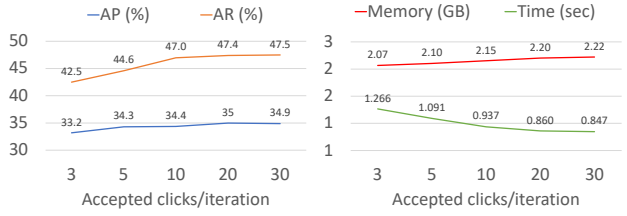
we compute the query vector $q \in \mathbb{R}^d$ by Masked Global Average Pooling [25]. The dot product similarity is then computed between X_1 and q to get the heatmap $H \in \mathbb{R}^{h \times w}$.

Pyramid Patch Matching. We firstly build the 3-level pyramid features of X_1 with the scales $\frac{1}{4}$, $\frac{1}{8}$, and $\frac{1}{16}$ of the original image. With the query mask, the query patch feature $q \in \mathbb{R}^{3 \times 3 \times d}$ is extracted by RoI-Align. By sliding the query patch feature on the feature pyramid, we can compute the similarity at each location to the query patch with Sum Squared Distance (SSD). The final heatmap is the average of responses of all pyramid levels.

Comparing with our CPN. Fig. 17 shows the differences in heatmaps produced by different methods. Pixel-wise similarity can cause many false positives, while the patch pyramid approach is not robust to objects with variant appearances. Our proposed method generates cleaner heatmaps with high precision. The precision-recall curve of three methods on DistractorSyn-Val is shown in Fig. 18. Our method outperforms other baselines with the balance between precision and recall rate.

5. Hyper-parameters of IDS

The following sections evaluate the performance of our proposed IDS with different hyper-parameters on DistractorSyn-Val. All experiments are with the SwinL



(a) Average Precision Recall (b) Computational cost

Figure 21. Accepting more clicks in each iteration improves the performance and the speed. However, memory usage also grows as a result.

Figure 22. Our model also performs well with large objects.

Figure 23. The robustness of our model when clicking on different objects.

backbone and trained on DistractorSyn14K. The pretrained weights on DistractorReal20K are used for feature extraction and mask generation modules. If not explicitly stated, all experiments have default hyper-parameters: the number of iterations $N = 5$, the number of exemplars $m = 3$, and the number of accepted clicks for each iteration $k = 10$. To make the comparison consistent, the PVM does not validate the outputs at each iteration.

Besides the Average Precision (AP) and Average Recall (AR), we also compute the time and GPU memory complexity of different hyper-parameters. While the time is measured before starting IDS until the last iteration, the GPU memory is the additional cost raised by the IDS, not by the whole network. The memory amount is computed with PyTorch API.

5.1. Number of Iterations

The Fig. 19 illustrates the performance of IDS with different numbers of iterations N. When $N = 1$, all proposal clicks are accepted, which are equivalent to the non-IDS experiment. Other experiments use the default value of $k = 10$ by default. There is a trade-off between computational cost and the performance of the framework when increasing the number of iterations. The AR increases proportionally to N until the fifth iteration. Because almost all clicks have been accepted after five iterations, continuously running the CPN after that only yields incremental improvements.

5.2. Number of Accepted Clicks

The results of different accepted clicks for each iteration are shown in Fig. 21. The performance and speed of the entire IDS process increase When accepting more clicks for each round. However, it also consumes more memory for mask generation. In practice, depending on the occurrence of distractors in the image, we can balance between the number of accepted clicks and the number of iterations to achieve the best results.

5.3. Number of Exemplars

We change the number of exemplars used for querying similar objects in each iteration of IDS process. The results are shown in Fig. 20. An increasing in the number of exemplars significantly rises the time and memory complexity. Additionally, the AP and AR are also improved with more exemplars.

6. Additional Qualitative Results

Our framework not only works with tiny distractors but also yield good results on large objects. Fig. 22 shows some examples where the selecting objects are larger than 10% of the image. Besides, Fig. 23 illustrates the robustness of our model in which the results are consistent while different

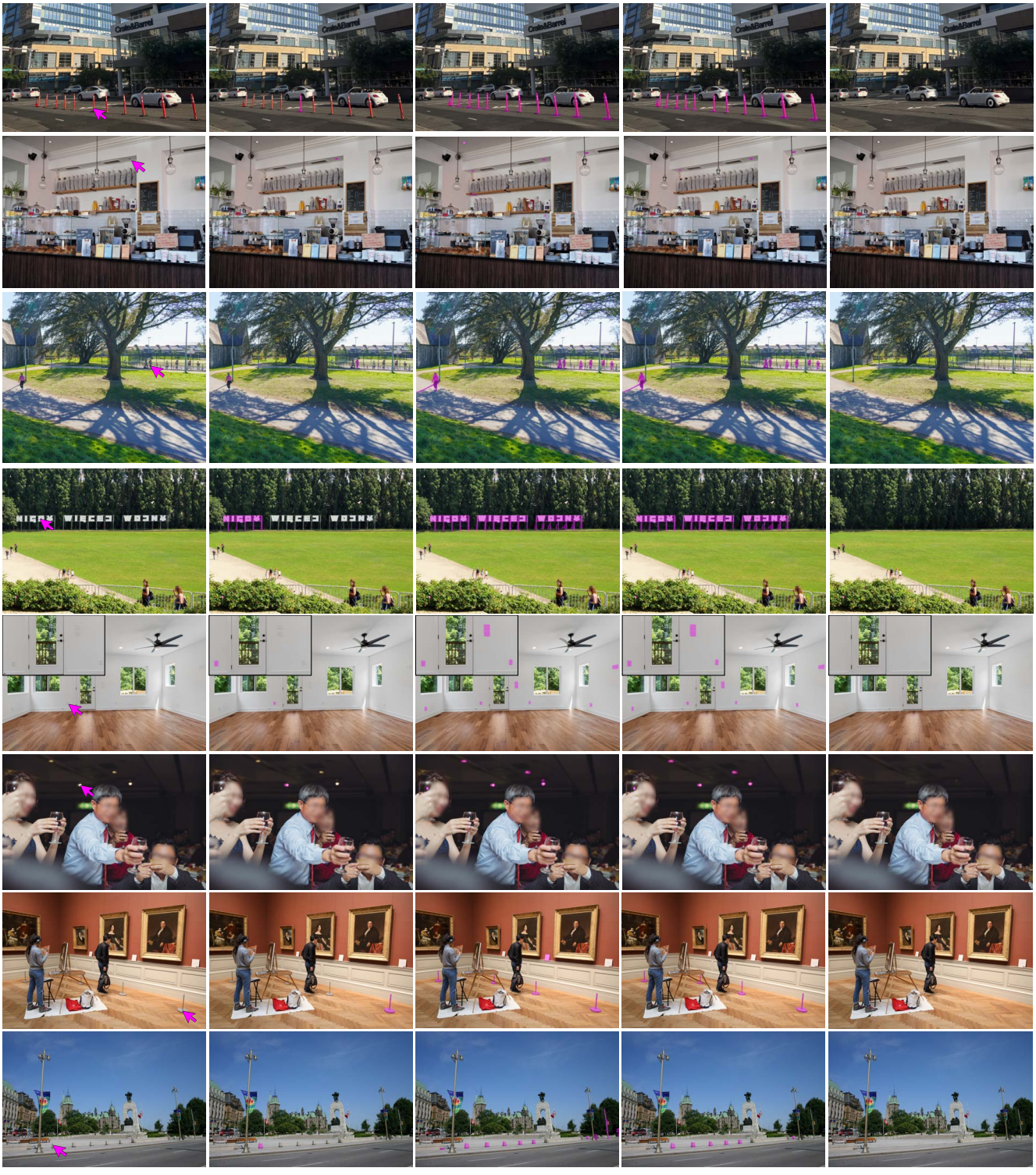


Figure 24. Some intermediate results without IDS process. Our CPN successfully finds similar objects with a high recall rate, and PVM correctly removes false positives to clean. (Best view in color and zoom-in)

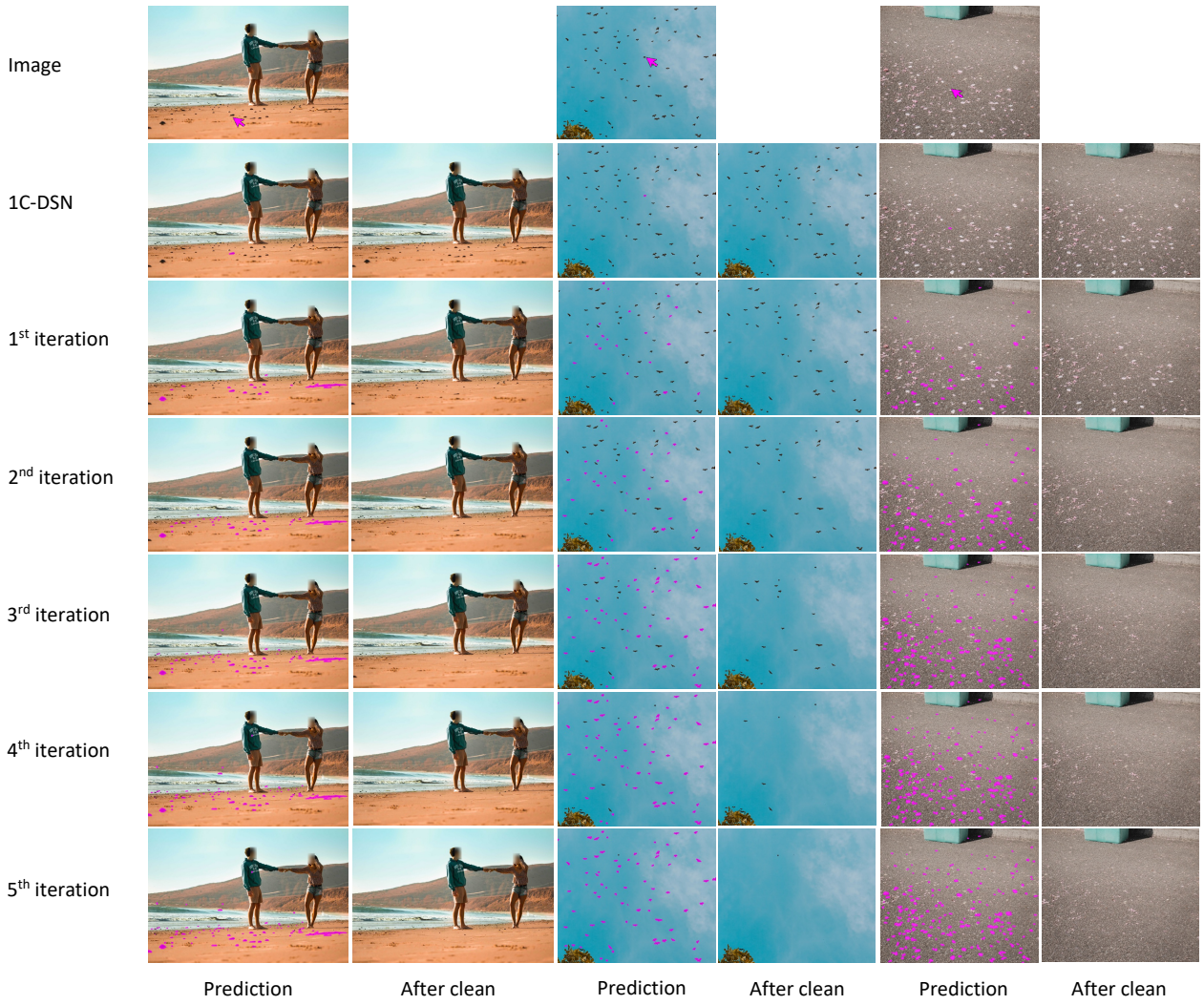


Figure 25. The progress of cleaning photos with IDS. For each iteration, some new similar distractors are detected, and the photos become cleaner than in the previous step. In the end, all distractors are removed from the image. (Best view in color and zoom-in)

objects are clicked.

Some additional results on real photos are shown in the Fig. 24 and Fig. 25. In some simple cases where there are not many repeated distractors, the CPN and PVM frameworks can work perfectly without IDS process. The CPN tries to achieve a high recall rate, and then the PVM helps increase precision by removing outliers.

The Fig. 25 shows some extreme cases where many similar distractors appeared, and the IDS joins in selecting all similar distractors with only one click. The photos get cleaner after each iteration because more distractors are selected and removed.

References

[1] Luca Ciampi. Deep learning techniques for visual counting. *arXiv preprint arXiv:2206.03033*, 2022. 6

[2] Tan M. Dinh, Anh Tuan Tran, Rang Nguyen, and Binh-Son Hua. Hyperinverter: Improving stylegan inversion via hypernetwork. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11389–11398, June 2022. 5

[3] 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. *arXiv preprint arXiv:2010.11929*, 2020. 5

[4] Patrick Esser, Robin Rombach, and Bjorn Ommer. Taming transformers for high-resolution image synthesis. In *Pro-*

ceedings of the IEEE/CVF conference on computer vision and pattern recognition, pages 12873–12883, 2021. 5

[5] Long-Nhat Ho, Anh Tuan Tran, Quynh Phung, and Minh Hoai. Toward realistic single-view 3d object reconstruction with unsupervised learning from multiple images. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 12600–12610, October 2021. 5

[6] 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*, pages 10012–10022, 2021. 5

[7] Yiqun Mei, Yuchen Fan, Yuqian Zhou, Lichao Huang, Thomas S Huang, and Honghui Shi. Image super-resolution with cross-scale non-local attention and exhaustive self-exemplars mining. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5690–5699, 2020. 5

[8] Khoi Pham, Kushal Kafle, Zhe Lin, Zhihong Ding, Scott Cohen, Quan Tran, and Abhinav Shrivastava. Learning to predict visual attributes in the wild. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 13018–13028, June 2021. 5

[9] Khoi Pham, Kushal Kafle, Zhe Lin, Zhihong Ding, Scott Cohen, Quan Tran, and Abhinav Shrivastava. Improving closed and open-vocabulary attribute prediction using transformers. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXV*, pages 201–219. Springer, 2022. 5

[10] Lu Qi, Jason Kuen, Yi Wang, Jiuxiang Gu, Hengshuang Zhao, Zhe Lin, Philip Torr, and Jiaya Jia. Open-world entity segmentation. *arXiv preprint arXiv:2107.14228*, 2021. 4

[11] Viresh Ranjan, Hieu Le, and Minh Hoai. Iterative crowd counting. In *Proceedings of the European Conference on Computer Vision (ECCV)*, September 2018. 6

[12] Viresh Ranjan and Minh Hoai Nguyen. Exemplar free class agnostic counting. In *Proceedings of the Asian Conference on Computer Vision (ACCV)*, pages 3121–3137, December 2022. 6

[13] Nirat Saini, Bo He, Gaurav Shrivastava, Sai Saketh Rambhatla, and Abhinav Shrivastava. Recognizing actions using object states. In *ICLR2022 Workshop on the Elements of Reasoning: Objects, Structure and Causality*. 5

[14] Nirat Saini, Khoi Pham, and Abhinav Shrivastava. Disentangling visual embeddings for attributes and objects. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 13658–13667, June 2022. 5

[15] Peter Shaw, Jakob Uszkoreit, and Ashish Vaswani. Self-attention with relative position representations. *arXiv preprint arXiv:1803.02155*, 2018. 5

[16] Min Shi, Hao Lu, Chen Feng, Chengxin Liu, and Zhiguo Cao. Represent, compare, and learn: A similarity-aware framework for class-agnostic counting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9529–9538, 2022. 6

[17] Weibo Shu, Jia Wan, Kay Chen Tan, Sam Kwong, and Antoni B Chan. Crowd counting in the frequency domain. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19618–19627, 2022. 6

[18] Phong Tran, Anh Tuan Tran, Quynh Phung, and Minh Hoai. Explore image deblurring via encoded blur kernel space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11956–11965, June 2021. 5

[19] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017. 5

[20] Junke Wang, Zuxuan Wu, Jingjing Chen, Xintong Han, Abhinav Shrivastava, Ser-Nam Lim, and Yu-Gang Jiang. Objectformer for image manipulation detection and localization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2364–2373, June 2022. 5

[21] Wenhai Wang, Enze Xie, Xiang Li, Deng-Ping Fan, Kaitao Song, Ding Liang, Tong Lu, Ping Luo, and Ling Shao. Pyramid vision transformer: A versatile backbone for dense prediction without convolutions. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 568–578, 2021. 5

[22] Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7794–7803, 2018. 5

[23] Enze Xie, Wenhai Wang, Zhiding Yu, Anima Anandkumar, Jose M Alvarez, and Ping Luo. Segformer: Simple and efficient design for semantic segmentation with transformers. *Advances in Neural Information Processing Systems*, 34:12077–12090, 2021. 5

[24] Jiahui Yu, Zhe Lin, Jimei Yang, Xiaohui Shen, Xin Lu, and Thomas S Huang. Generative image inpainting with contextual attention. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5505–5514, 2018. 5

[25] Xiaolin Zhang, Yunchao Wei, Yi Yang, and Thomas S Huang. Sg-one: Similarity guidance network for one-shot semantic segmentation. *IEEE transactions on cybernetics*, 50(9):3855–3865, 2020. 7

[26] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable detr: Deformable transformers for end-to-end object detection. *arXiv preprint arXiv:2010.04159*, 2020. 5