Supplementary material: Learning Co-segmentation by Segment Swapping for Retrieval and Discovery

Xi Shen[1], Alexei A. Efros[2], Armand Joulin[3], and Mathieu Aubry[4]

[1,4]LIGM (UMR 8049), École des Ponts ParisTech
[2]University of California, Berkeley
[3]Facebook AI Research

The code is available in the folder *Code_ID4435*. Please refer to the *README.md* for the reproduction of the results. In this supplementary material, we present:

- Section 1: additional retrieval results on Brueghel [1, 53], Tokyo24/7 [60] and Pitts30K [61] similar to Figure.1.
- Section 2: additional discovery results on Brueghel [1, 53].
- Section 3: visual results on the unsupervised saliency detection datasets.
- Section 4: additional training details and examples of training examples.
- Section 5: more details about graphs on the dataset of [48] and Brueghel [1, 53].
- Section 6: more details on our cross-image transformer and ablations on both cross-image transformer and Sparse Nc-Net [45], which includes the ablation study of the score, architecture of the transformer and Sparse Nc-Net[45].
- Section 7: more details on the ArtMiner [53] post-processing, GrabCut [46] as well as more information on Brueghel [1, 53], Tokyo24/7 [60] and Pitts30K [61] datasets.

1. Retrieval results on Brueghel [1, 53], Tokyo24/7 [60] and Pitts30K [61]

We show additional retrieval results on Brueghel [1, 53], Tokyo24/7 [60] and Pitts30K [61] similar to Figure.2. Precisely, the results are organised in the following figures:

- Figure 1 and 2, top-3 retrieved results on Brueghel [1, 53]
- Figure 3 and 4, top-3 retrieved results on Tokyo [60]
- Figure 5 and 6, top-3 retrieved results on Pitts30K [61]

For Tokyo24/7 [60] and Pitts30K [61], we also show the top-3 retrieved images by NetVLAD [2] (the top row for each query), which we use to get top-100 images then apply re-ranking with our approach. Our approach can match to very challenging cases such as large scale difference (e.g., Figure 2 query 2, 3 and 4), large style/ appearance difference (e.g.,Figure 1 query 7 and 8, Figure 2 query 5, 7 and 8, Figure 3 and 4).

2. Additional discovery results on Brueghel [1, 53]

Extracting the most densely connected subgraphs from our full correspondence graph is a difficult problem. Using K-means on the eigenvector decomposition allows us to extract some densely connected regions, but ultimately K-means will cluster all candidate correspondences, including wrong ones. Thus, we compute the following energy associated to each cluster $\mathcal{C}$, and focus on clusters with high energy:

$$E_{\mathcal{C}} = \frac{1}{|\mathcal{C}|} \sum_{i,j \in \mathcal{C}} \mathcal{E}_{i,j} \tag{1}$$

Qualitatively, these clusters indeed correspond to the most interesting ones, examples of such relevant clusters are given in Figure 16 and 17. Inside each cluster, the images are ranked by the total number of correspondences in the cluster. The images that seem to be irrelevant in each cluster are shown with red borders, and typically correspond to images that are the most loosely connected to the rest of the cluster.

Typical failure cases of irrelevant clusters with high energy are shown in Figure 18. They can be clusters associated to textureless regions, such as sky and tree (Figure 18, first and second cluster). Some clusters also include content that is similar but not an exact match (Figure 18, third and fourth cluster), which we believe is because of the similarity between the MOCOV2 [5] feature for these patterns (which is also the reason why our approach can work for more semantic tasks, such as discovery on the internet dataset).

3. Unsupervised segments on ECSSD [54], DUTS [70] and DUT-OMRON [73]

We show unsupervised segments obtained on ECSSD [54], DUTS [70] and DUT-OMRON [73]. For each dataset, we visualize the segments obtained with LOST [55] and adding Bilateral Solver [4].

ECSSD [54] is a test set containing 1 000 images. Images are with complex natural contents. Visual results are provided in Figure 7.

DUTS [70] consists of 10 553 train and 5 019 test images. We only use the test set for evaluation. Visual results are provided in Figure 8.

DUT-OMRON [73] contains 5 168 test images. Visual results are provided in Figure 9.

We can see Bilateral Solver [4] allows refining the boundary and the simple approach can obtain salient segments.

4. Additional training details and examples of training samples

We generate 200k pairs for training and save images both with and without style transfer. Further increasing training size leads to similar performance. 100k pairs are with one object blended, while the other 100k pairs are with two objects blended. During the training, we uniformly sample stylised and non-stylised image to create training pairs.

Some training samples are visualised in Figure 10. The first three rows are pairs with one blended object, the last three rows are pairs with two blended objects. For each pair, we show the images with (column 3 and 4) and without style transfer (column 1 and 2).

5. Details to construct the correspondence graphs

Reducing number of vertices We introduce two strategies to reduce the number of vertices: i) we remove all the correspondences for which the mask prediction is smaller than a threshold τ ($\tau = 0.6$ for Brueghel [1, 53] and $\tau = 0.85$ for Internet dataset [48]) ; ii) for correspondences in a given pair, instead of taking the correspondences from the two directions (source to target and target to source), we only take the direction where the maximum number of correspondences are above the threshold (intuitively it matches a large resolution to a small resolution and keep the direction with the most valid correspondences).

Clustering images with very similar content in Brueghel [1, 53] Sets of images with very similar content have a huge amount of valid correspondences and would thus create a huge amount of vertices in our correspondences graph. To avoid this effect for the Brueghel dataset [1, 53], which contain many very similar images, we first detect images with very similar visual content (which we refer to as *duplicate images*, even if they might have very different style and smalle content difference). We build a graph of images where we connect images if our average mask predictions in both directions is superior to $\delta = 0.65$. We then consider the connected components as duplicate images. The clusters images are shown in Figure 11, 12, 13, 14 and 15. To build the correspondence graph, we only keep one image for each duplicate image cluster, the one with the largest average mask prediction with all the other images of the cluster.

σ consistency parameter We take $\sigma = 5$ for discovery on Brueghel [1, 53]. For Internet dataset [48], we set $\sigma = 1.2$ for Horses, $\sigma = 0.8$ for Airplanes and $\sigma = 5$ for Cars.

6. Architecture details and ablation study

Positional encoding in the cross-image transformer We include a 2D positional encoding on top of the feature map before the standard self-attention layer (SA). We use the standard sine and cosine functions of different frequencies similar to

Dataset	Brueghel	Tokyo
w.o Poisson Blending	75.1	60.0
w.o Style Transfer	75.6	57.8
w.o Hard Negative Mining	81.9	74.6
All	**84.4**	**80.0**
Score		
Eqn.3 w.o Trans. Mask	51.2	63.5
Eqn.3 w.o Feat. Similarity	84.1	78.1
Eqn.3	**84.4**	**80.0**
Loss		
$\mathcal{L}_m$	22.4	33.0
$\mathcal{L}_m + \mathcal{L}_{corr}$	79.8	61.3
$\mathcal{L}_{tm} + \mathcal{L}_{corr}$	80.9	67.8
$\mathcal{L}_m + \mathcal{L}_{tm}$	8.5	13.3
$\mathcal{L}_m + \mathcal{L}_{tm} + \mathcal{L}_{corr}$	**84.4**	**80.0**
Architecture		
SA-SA-SA-SA-SA	Diverge	
CA-CA-CA-CA-CA	24.6	27.9
CA-SA-CA-SA-CA	72.9	63.2
SA-CA-SA-CA-SA (proposed)	**84.4**	**80.0**
Positional Encoding (P.E.)		
w.o P.E	56.3	41.3
SA P.E. and CA P.E.	81.4	79.0
Only SA. Pos (proposed)	**84.4**	**80.0**

Table 1. Ablation study of the cross-image transformer.

DeTR [4]:

$$\begin{cases} PE(x, y, 2i) = \sin(\frac{2\pi x}{10000^{\frac{4i}{D}}}) \\ PE(x, y, 2i+1) = \cos(\frac{2\pi x}{10000^{\frac{4i}{D}}}) \\ PE(x, y, 2j+\frac{D}{2}) = \sin(\frac{2\pi y}{10000^{\frac{4j}{D}}}) \\ PE(x, y, 2j+1+\frac{D}{2}) = \cos(\frac{2\pi y}{10000^{\frac{4j}{D}}}) \end{cases} \tag{2}$$

where i,j $\in [0, \frac{D}{4})$ and x, y is the normalised 2d coordinate between 0 and 1.

Ablation study on the cross-image transformer and Sparse Nc-Net [45] The detailed ablation studies for the cross-image transformer and Sparse Nc-Net [45] are in Table 1 and 2 respectively. For the cross-image transformer, we can see: i) the hard negative mining provides consistent improvement on both datasets; ii) as claimed in the paper, the mask term is the key part of the score in Eqn.3; iii) alternating SA and CA layers is important; iv) the positional encoding on the SA is important and further adding positional encoding in CA doesn't lead to better performance. For Sparse Nc-Net [45], we also compare all our modifications to the original model in [45], we notice that: i) similar to the transformer, the poisson blending and style transfer help to generalize better on the test sets; ii) as claimed in the paper, the hard negative mining does not improve the performance on Sparse Nc-Net [45]; iii) similar to the cross-image transformer, the mask term is the key part of the score in Eqn.3; iv) MocoV2 features and learning on the proposed dataset with full supervision boost the performance. Note that for Sparse Nc-Net [45], adding the loss with transported mask ($\mathcal{L}_{tm}$) doesn't improve the performance, which can be related to the fact that the masks predicted by Nc-Net tend to be more localized in discriminative regions.

Dataset			Brueghel	Tokyo
w.o Poisson Blending			81.0	83.4
w.o Style Transfer			81.2	83.6
w.o Hard Negative Mining			**82.7**	**85.4**
All			**82.7**	**85.4**
Score				
Eqn.3 w.o Trans. Mask			35.7	68.9
Eqn.3 w.o Feat. Similarity			82.5	83.5
Eqn.3			**82.7**	**85.4**
Backbone	**Dataset**	**Loss**		
INet-ResNet101*	IVD [45]*	$\mathbf{M}^{neg}$ - $\mathbf{M}^{pos}$ [45]*	34.6	67.0
INet-ResNet101	Ours	$\mathbf{M}^{neg}$ - $\mathbf{M}^{pos}$ [45]	67.8	71.1
		$\mathcal{L}_m + \mathcal{L}_{corr}$	76.5	73.4
MocoV2-ResNet50	Ours	$\mathbf{M}^{neg}$ - $\mathbf{M}^{pos}$ [45]	77.9	83.8
		$\mathcal{L}_m$	81.4	80.0
		$\mathcal{L}_m + \mathcal{L}_{tm}$	80.9	79.7
		$\mathcal{L}_m + \mathcal{L}_{tm} + \mathcal{L}_{corr}$	82.2	85.1
		$\mathcal{L}_{tm} + \mathcal{L}_{corr}$	82.4	84.8
		$\mathcal{L}_m + \mathcal{L}_{corr}$	**82.7**	**85.4**

* We use the official pre-trained model from [45], which is available in https://github.com/ignacio-rocco/sparse-ncnet

Table 2. Ablation study of Sparse Nc-Net [45]. We also compare all the modifications with respect to the original Nc-Net, which employed ResNet101 features pre-trained on ImageNet classification and trained on IVD dataset [45] with weakly supervised loss.

7. ArtMiner [53] post-processing, GrabCut [46], Brueghel [1, 53], Tokyo 24/7 [60] and Pitts30k [61]

ArtMiner [53] post-processing All the images are initially resized to 640 $\times$ 640. To compare with the detection results in ArtMiner [53], we first compute a coarse estimation of the center of the matching object by computing the average of the predicted correspondences weighted by the predicted mask. We then crop a 320 $\times$ 320 patch around this point, which intuitively should contain the matching bounding box. The cropped patch is resized to five scales: 160, 224, 320, 448, 640 and the query patch is such that the maximum dimension is 128. We finally perform one-shot detection of the resized query patch on the different scales of the cropped target patch with cosine similarity similar to the baseline in ArtMiner [53] to obtain bounding box. This post-processing takes approximately 1 hour for the full dataset which is more than 40 times faster than discovery in ArtMiner [53] and 7 times faster than cosine similarity in ArtMiner [53].

GrabCut [46] As explained in the paper, we compute the eigenvector corresponding to the maximum eigenvalue as the seed for GrabCut [46]. For each image, we compute the potential by associating to each position the sum of the eigenvector values for the correspondences at this position. For GrabCut [46], we use the implementation available in OpenCV [1]. The pixels with potential larger than 60% of the maximum potential are set to be foreground pixels.

Brueghel [1, 53] The dataset is proposed in ArtMiner [1, 53], and contains 1, 587 artworks crafted with different media (e.g. oil, ink, chalk, watercolor) and on different materials (e.g. paper, panel, copper), describing a wide variety of scenes (e.g. landscape, religious, still life). It includes annotations of 10 of the most commonly repeated details which results in 273 annotated instances with bounding boxes, with a minimum of 11 and a maximum of 57 annotations per pattern.

[1] https://docs.opencv.org/3.4/d8/d83/tutorial_py_grabcut.html

Tokyo 24/7 [60] and Pitts30k [61] Pitts30k [61] contains 30k database images downloaded from Google Street View and 24k test queries generated from Street View but taken at different times, years apart. This dataset is divided into three roughly equal parts for training, validation, and testing, each containing around 10k database images and 8k queries, where the division was done geographically to ensure the sets contain independent images. Similar to Patch-NetVLAD [19], we evaluate its test set containing 8, 280 queries and 83, 952 database images. Tokyo 24/7 [60] consists of 76k database images and 315 query images taken using mobile phone cameras. This is an extremely challenging dataset where the queries were taken at daytime, sunset, and night, while the database images were only taken at daytime as they originate from Google Street View as described above.

Figure 1. Additional visual results on Brueghel [1, 53] (1st Part).

Figure 2. Additional visual results on Brueghel [1, 53] (2nd Part).

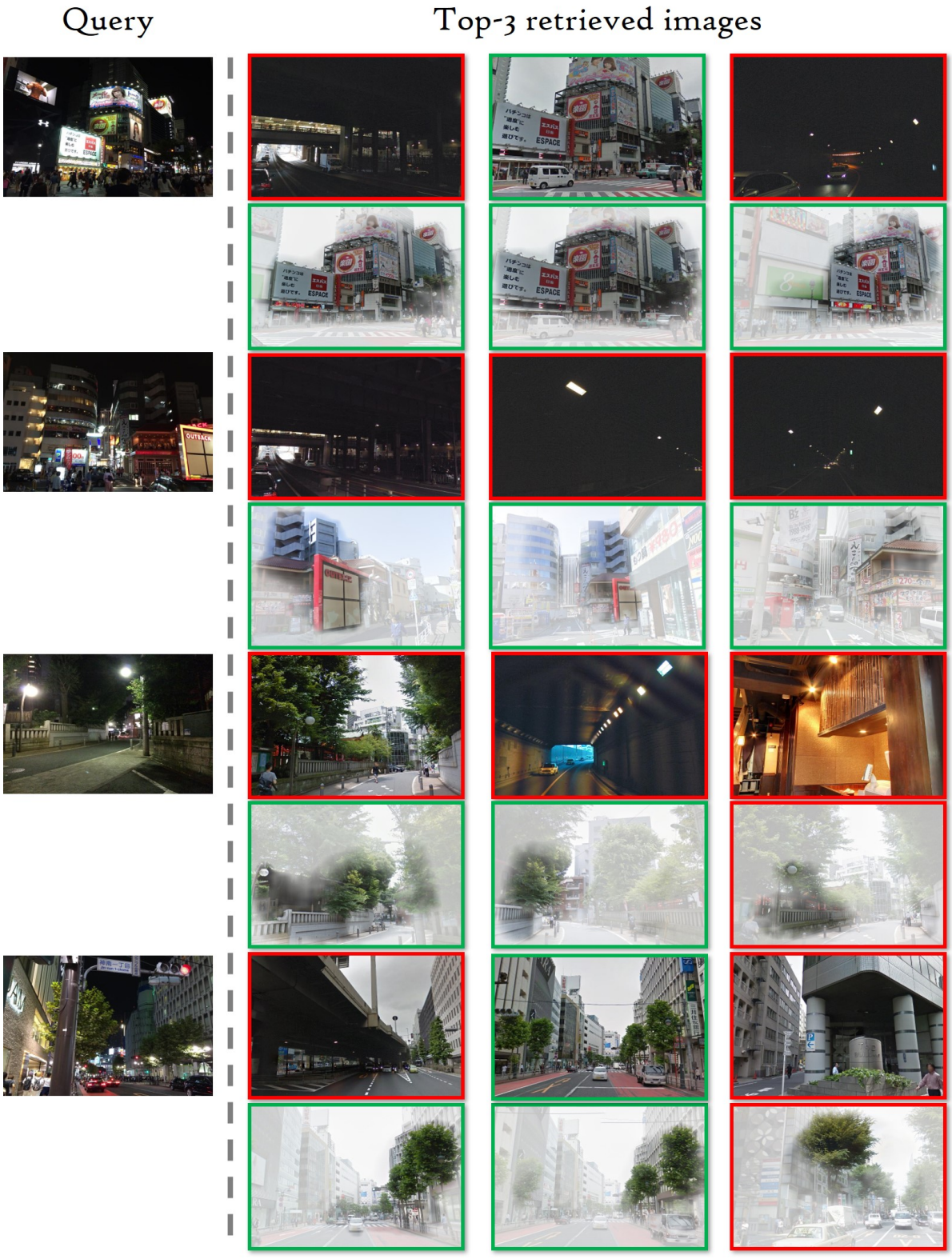


Figure 3. Additional visual results on Tokyo24/7 [60] (1st Part). Top row: top-3 retrieved results with NetVLAD [2]. Bottom row: top-3 results after re-ranking with our approach.

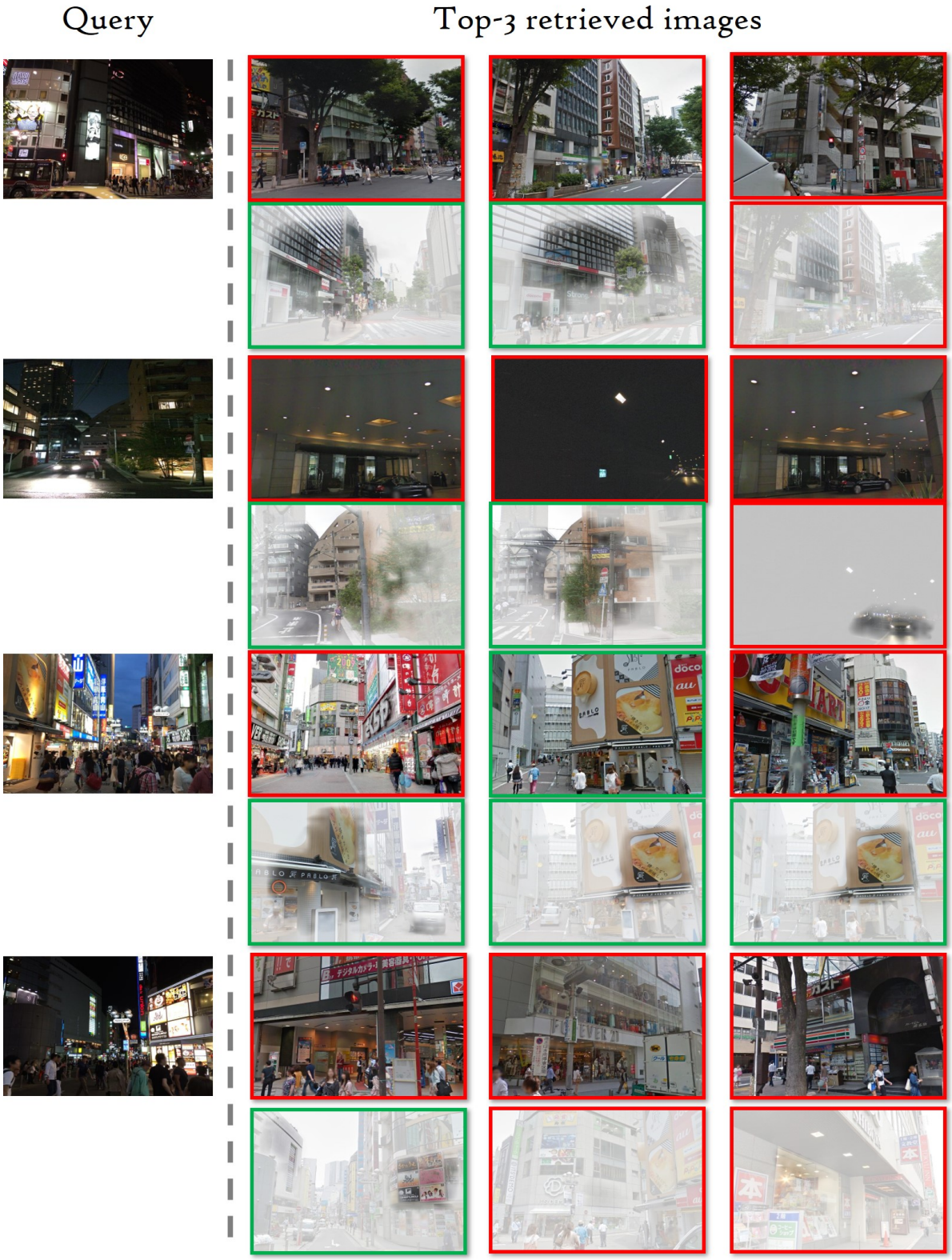


Figure 4. Additional visual results on Tokyo24/7 [60] (2nd Part). Top row: top-3 retrieved results with NetVLAD [2]. Bottom row: top-3 results after re-ranking with our approach.

Figure 5. Additional visual results on Pitts30K [61] (1st Part). Top row: top-3 retrieved results with NetVLAD [2]. Bottom row: top-3 results after re-ranking with our approach.

Figure 6. Additional visual results on Pitts30K [61] (2nd Part). Top row: top-3 retrieved results with NetVLAD [2]. Bottom row: top-3 results after re-ranking with our approach.

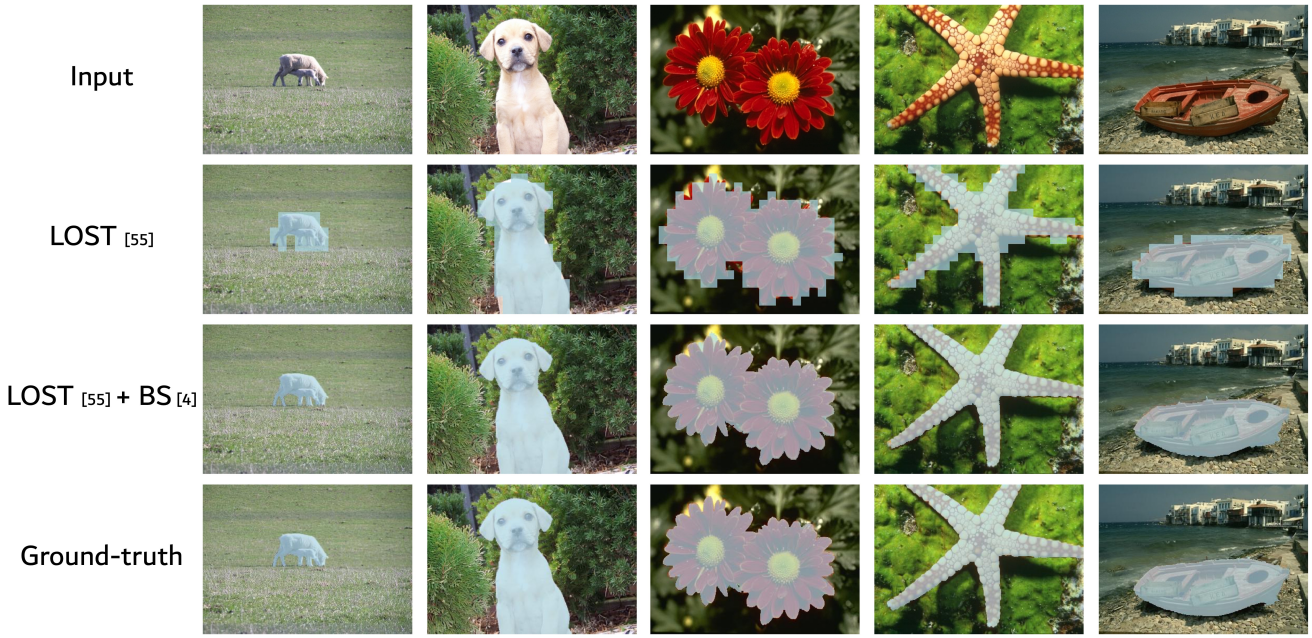


Figure 7. Unsupervised segments on ECSSD [54]

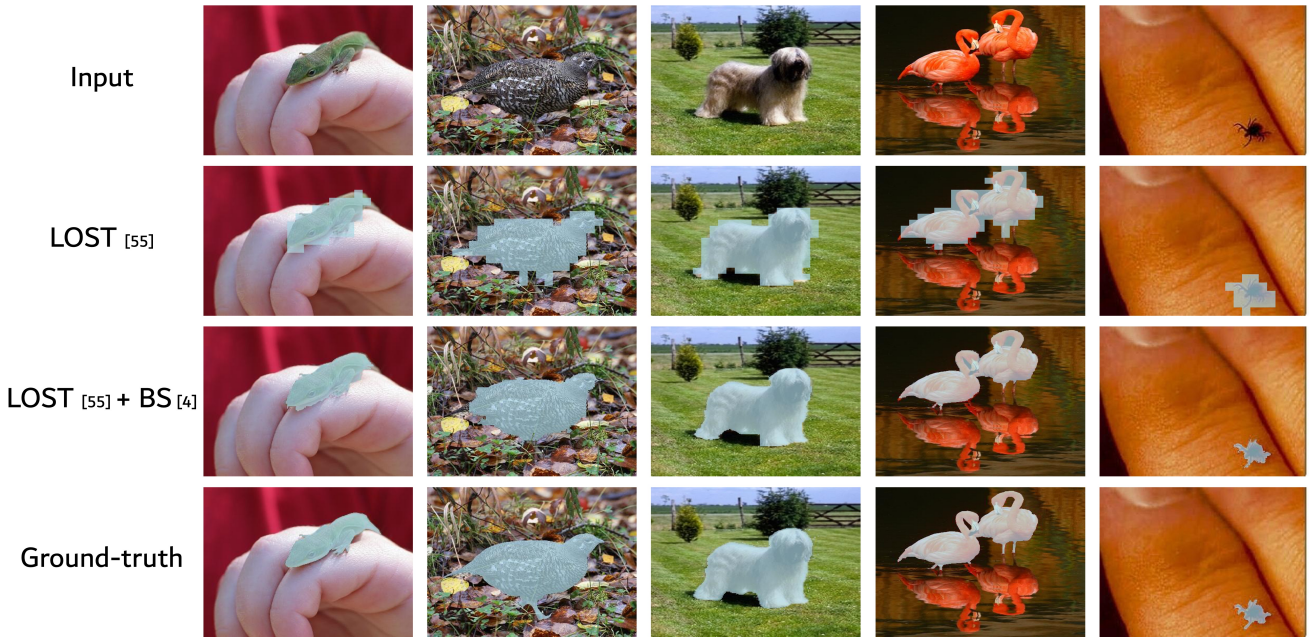


Figure 8. Unsupervised segments on DUTS [70]

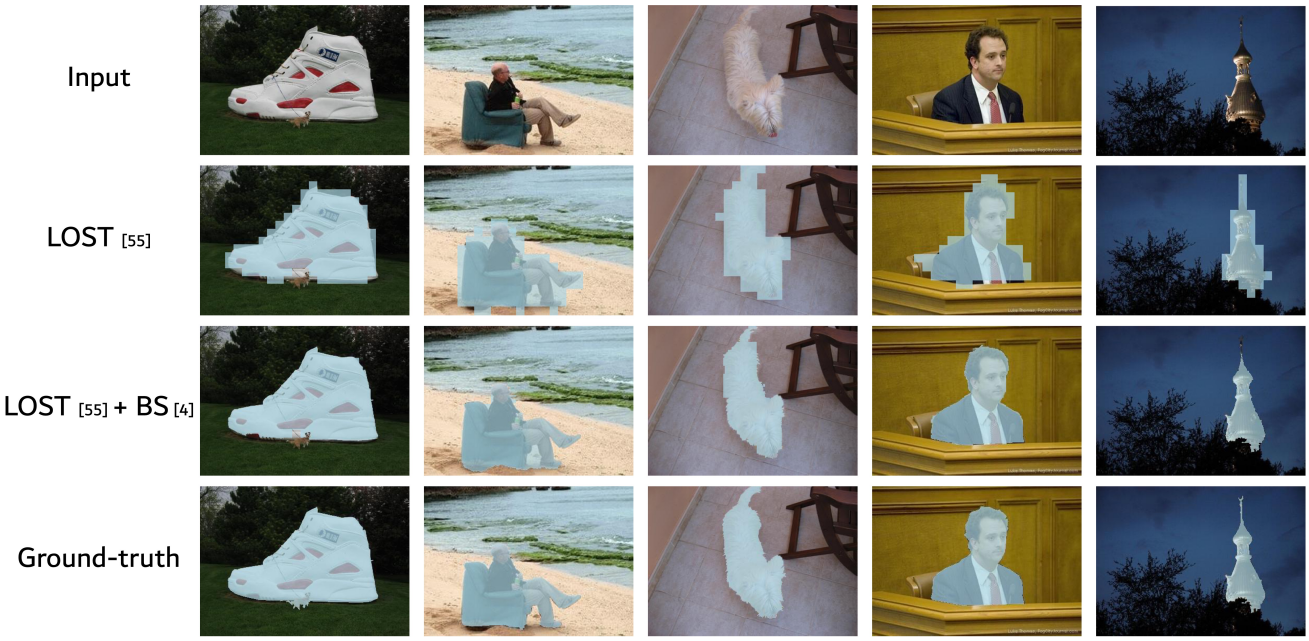
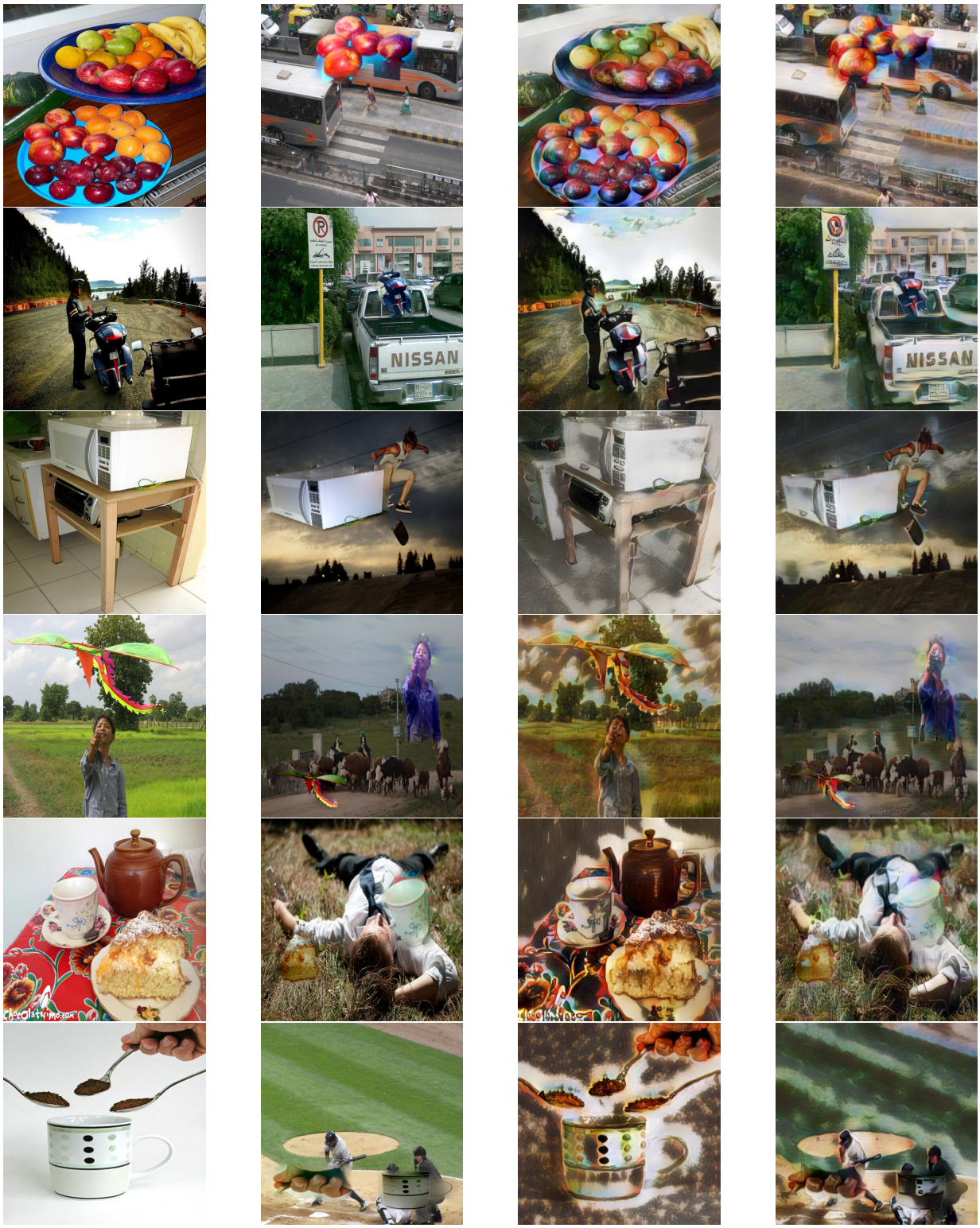


Figure 9. Unsupervised segments on DUT-OMRON [73]



(a) Source Image (b) Target Image (c) Stylised Source Image (d) Stylised Target Image

Figure 10. Image pairs generated with “segment augmentation” in addition to Figure 1 in the main paper.

Figure 11. Duplicate images clusters (1)

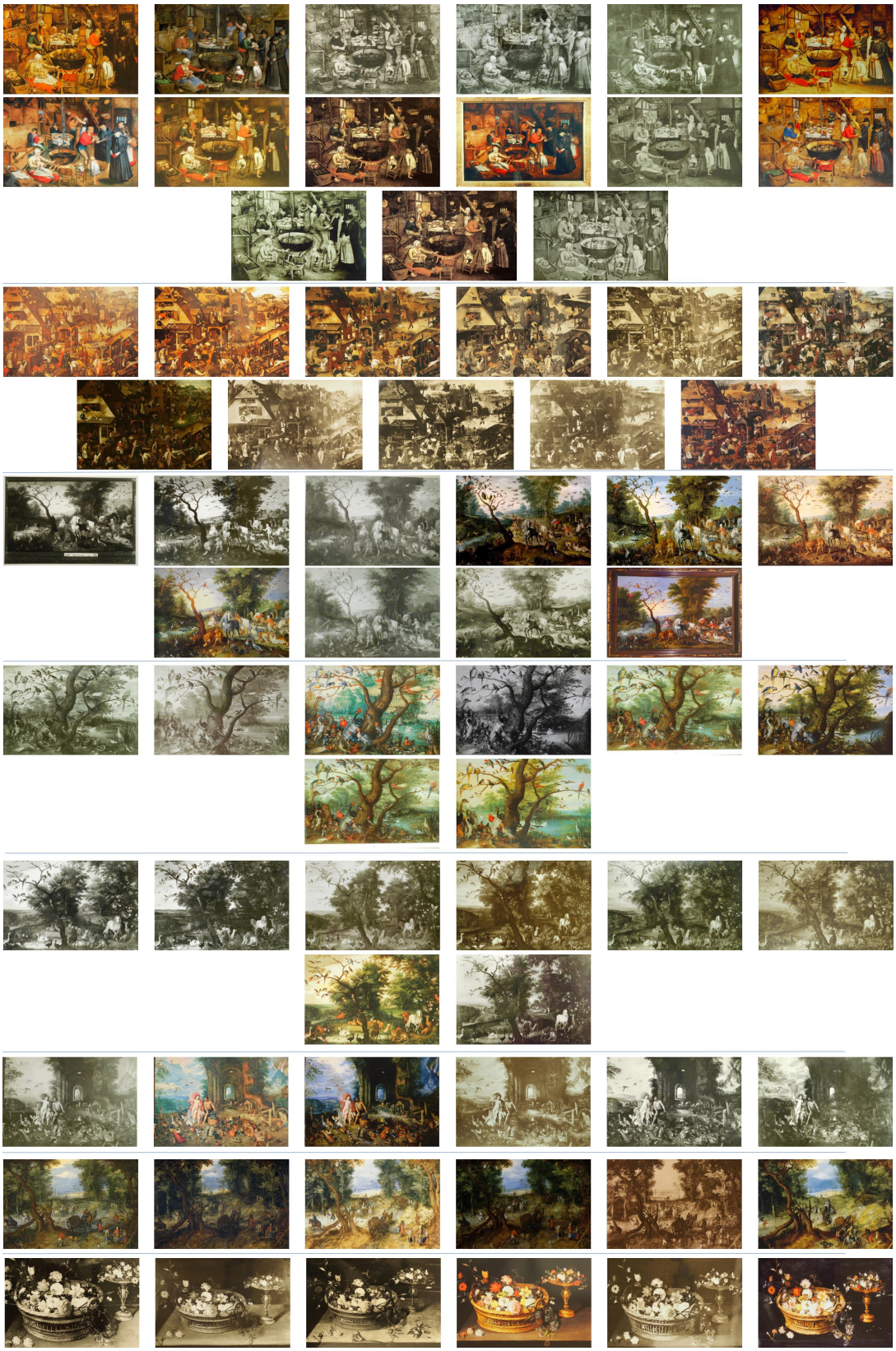


Figure 12. Duplicate images clusters (2)

Figure 13. Duplicate images clusters (3)

Figure 14. Duplicate images clusters (4)

Figure 15. Duplicate images clusters (5)

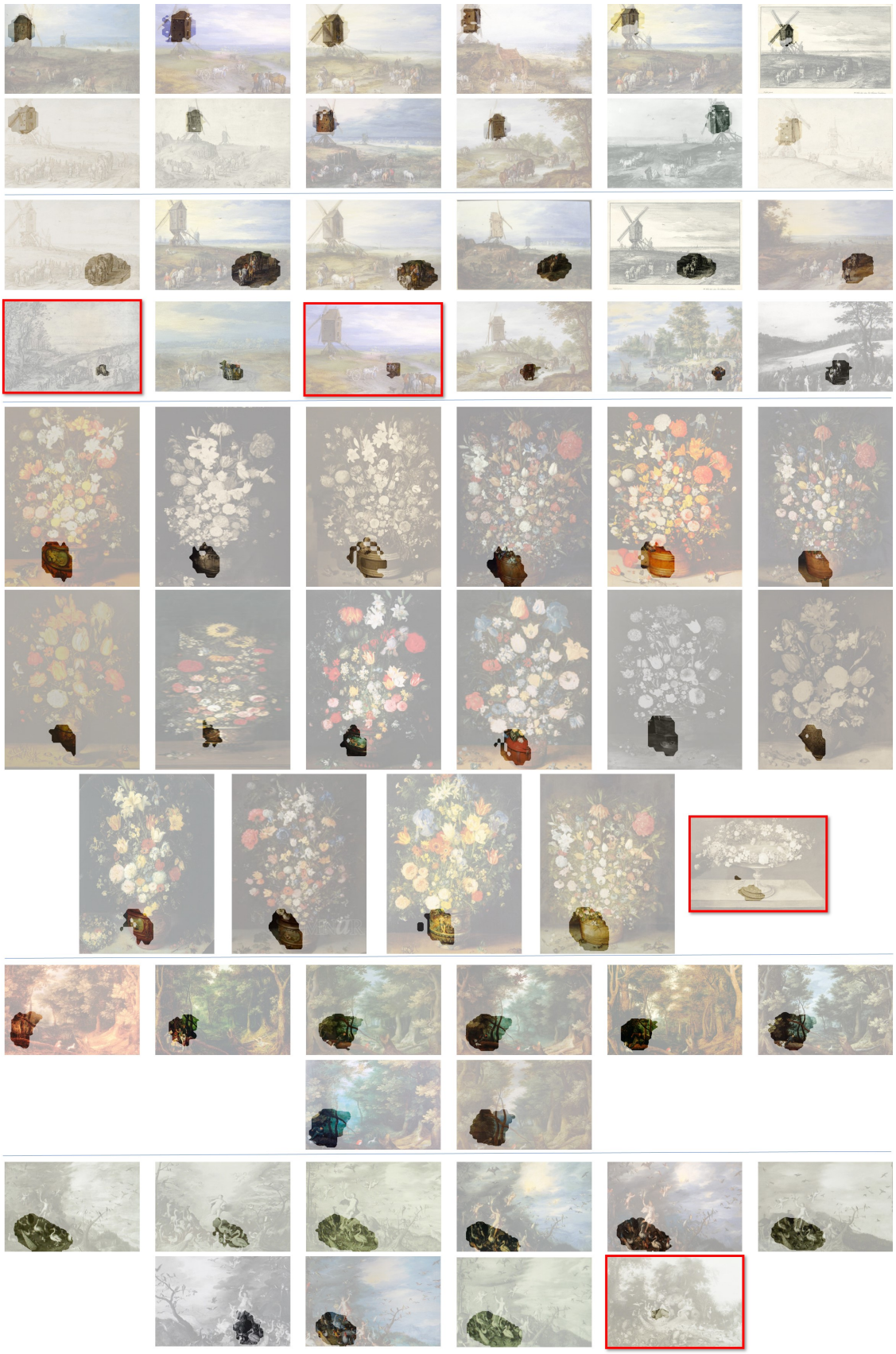


Figure 16. Discovered clusters on Brueghel [1, 53] (1). Images that seem to be irrelevant in each cluster are shown with red borders.

Figure 17. Discovered clusters on Brueghel [1, 53] (2). Images that seem to be irrelevant in each cluster are shown with red borders.

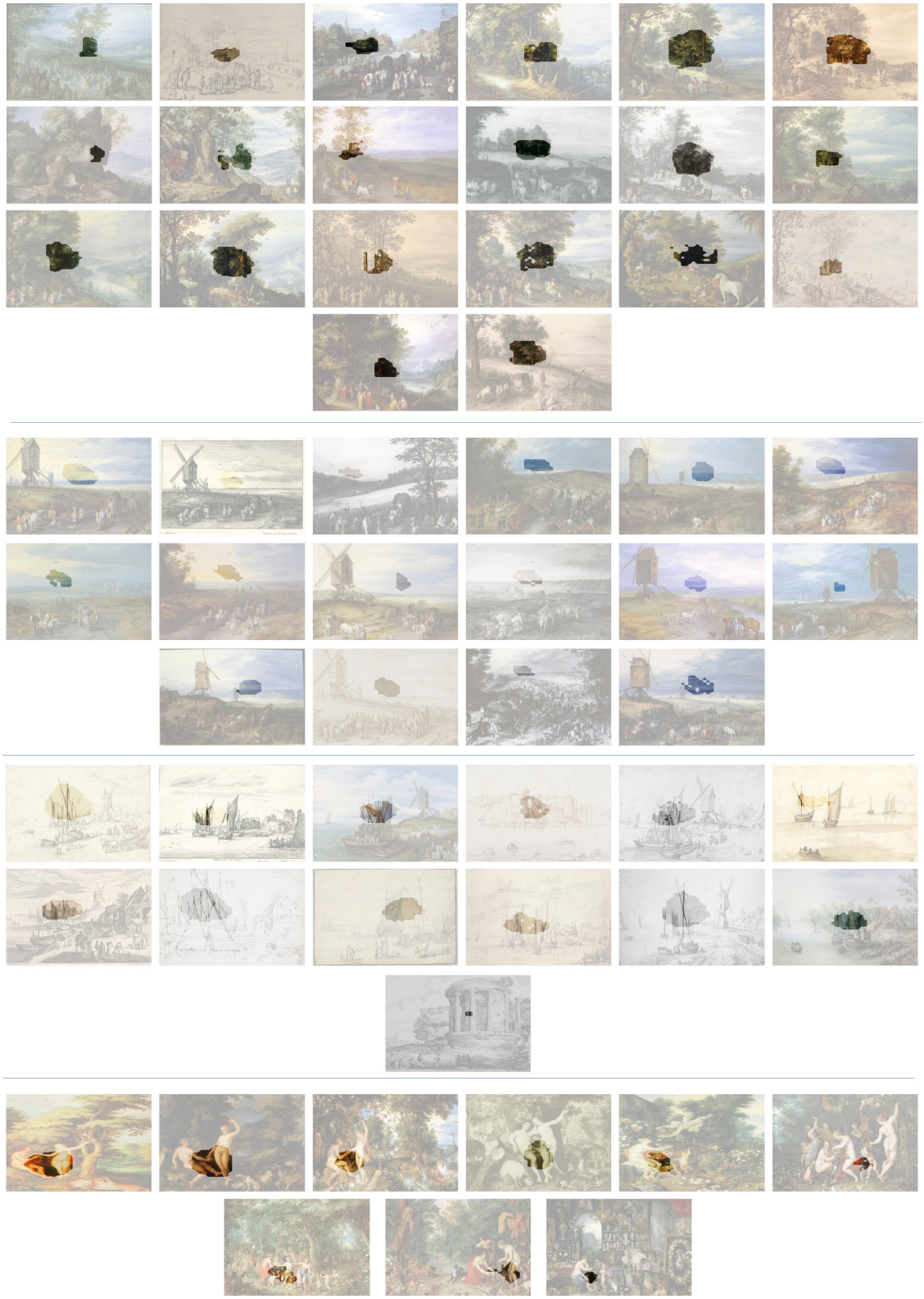


Figure 18. Discovered clusters on Brueghel [1, 53]. Typical failure cases.