Supplementary Material of Dual Contrastive Learning for Unsupervised Image-to-Image Translation

1. Implementation Details

1.1. Architecture of Generator and layers used for PatchNCE loss

Our generator architecture is based on CycleGAN[11] and CUT[9]. We only use ResNet-based[4] generator with 9 residual blocks for training. It contains 2 downsampling blocks, 9 residual blocks, and 2 upsampling blocks. Each downsampling and upsampling block follows two-stride convolution/deconvolution, normalization, ReLU. Each residual block contains convolution, normalization, ReLU, convolution, normalization, and residual connection.

We define the first half of generators G and F as encoder which are represented as G_{enc} and F_{enc}. The patch-based multi-layer PatchNCE loss is computed using features from four layers of the encoder (the first and second downsampling convolution, and the first and the fifth residual block). The patch sizes extracted from these four layers are 9×9, 15×15, 35×35, and 99×99 resolution respectively. Following CUT[9], for each layer's features, we sample 256 random locations and apply the 2-layer MLP (projection head H_X, H_Y) to infer 256-dim final features.

1.2. Architecture of Discriminator

We use the same PatchGAN discriminator architecture as CycleGAN[11] and Pix2Pix[6] which uses local patches of sizes 70x70 and assigns every patch a result. This is equivalent to manually crop one image into 70x70 overlapping patches, run a regular discriminator over each patch, and average the results. For instance, the discriminator takes two images from domain X and domain Y, passes them through five downsampling Convolutional-Normalization-LeakeyReLU layers, and outputs a result matrix of 30x30. Each element corresponds to the classification result of one patch. Following CycleGAN[11] and Pix2Pix[6], in order to improve the stability of adversarial training, we use a buffer to store 50 previously generated images.

1.3. Architecture of four light MLPs

For SimDCL, we use four light MLPs (H_1, H_2, H_3, H_4). These MLPs project the 256-dim features to 64-dim vectors. Each MLP contains one convolutional layer followed by ReLU, average pooling, linear transformation (64-dim to 64-dim), ReLU, and linear transformation (64-dim to 64-dim).

1.4. Additional training details

We presented most training details in the main paper, here, we depict some additional training details. For SimDCL, we use the Adam optimiser[7] with β_1 = 0.5 and β_2 = 0.999. We update the weights of $H_X, H_Y, H_1, H_2, H_3, H_4$ together with learning rate 0.0002. For both DCLGAN and SimDCL, we initialize weights using xavier initialization[2]. We load all images in 286x286 resolution and randomly crop them into 256x256 patches during training and we load test images in 256x256 resolution. All images from the test set are used for evaluation. For all tasks, we train our method and other baselines with a Tesla P100-PCIE-16GB GPU. The GPU driver version is 440.64.00 and the CUDA version is 10.2.

1.5. Additional evaluation details

We list the evaluation details of Fréchet Inception Distance (FID)[5] and Fully convolutional Network (FCN)[8] score. For FID[5] score, we use the official PyTorch implementation[10] with the default setting to match the evaluation protocol of CUT[9]. The link is https://github.com/mseitzer/pytorch-fid.

For FCN[8] score, we use the official PyTorch implementation of CycleGAN[11] and Pix2Pix[6]. The link is https://github.com/junyanz/pytorch-CycleGAN-and-pix2pix. The FCN[8] score is a well-known semantic segmentation metric on the CityScapes dataset. It measures how the algorithm finds correspondences between labels and images. The FCN[8] score is computed using a pretrained FCN-8[8] network that predicts a label map for a photo. We input the generated photos to the pretrained network and measure the predicted labels with ground truth using three semantic segmentation metrics including mean class Intersection over Union (IoU), pixel-wise accuracy (pixAcc), and average class accuracy (classAcc).

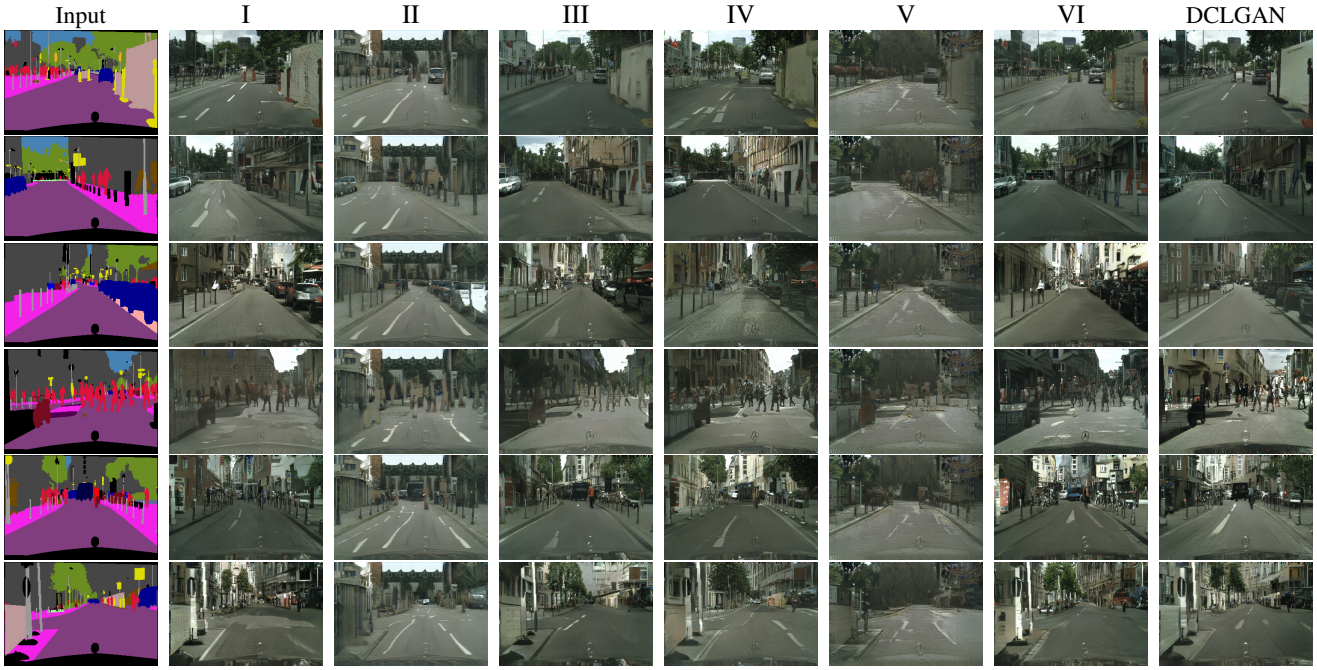

Figure 1. Qualitative results of ablations on CityScapes task.

2. Additional Ablation studies

2.1. Drawing external negatives

In self-supervised representation learning, more negatives (other images than transformed image) lead to a better contrast and therefore a better performance[3, 1]. In contrary, CUT[9] states that Internal negatives (patches from an input image only) are more effective than external negatives (patches from other images). CUT[9] adds negatives from other images with a momentum encoder[3]. We explore this in a different approach, by taking the advantage of the dual setting. DCLGAN produces four different stacks of features at each iteration. Concatenating two stacks of features belonging to the same domain provides more negatives (511 negatives (255 internal and 256 external) for one query, the default DCLGAN uses 255 internal negatives). The FID scores of the default DCLGAN for Horse $\rightarrow$ Zebra, Zebra $\rightarrow$ Horse, and CityScapes are 43.2, **139.5**, and **49.4** respectively, whereas the FID scores of this variant are **41.7**, 149.2, and 49.6. We obverse better quantitative results in Horse $\rightarrow$ Zebra and very close results in CityScapes for this variant. Although the gap of FID score between the default DCLGAN and this variant is very small, the visual quality of this variant is not as good as that of the default DCLGAN, that is, objects in the generated image tend to be merged together, as shown in VI of Figure 1.

2.2. Qualitative results of ablations

For different ablations including (I) Adding the first RGB pixels back, (II) Using dot product instead of cosine similarity, (III) Using the same encoder and MLP for one mapping instead of two, (IV) Adding cycle-consistency loss, (V) Removing the dual setting, and (VI) Drawing external negatives. We show the randomly picked qualitative results in Figure 1 and Figure 2. The qualitative results suggest that DCLGAN generates more realistic images than other variants, while each of our contribution has shown its efficiency.

3. Additional Results

We evaluated our proposed method and baselines among nine tasks. We choose the best four methods and show more randomly selected qualitative results among all tasks except for Facade $\rightarrow$ Label. This is an extension of Figure 2 and Figure 3 in the main paper. Figure 3 shows some qualitative results of Horse $\leftrightarrow$ Zebra, Figure 4 shows the results of Cat $\leftrightarrow$ Dog. The results of CityScapes and Label $\rightarrow$ Facade are shown in Figure 5, Figure 6 shows the results of Van Gogh $\rightarrow$ Photo and Orange $\rightarrow$ Apple.

Cat $\leftrightarrow$ Dog task requires geometry changes to match the distribution. As shown in Figure 4, DCLGAN performs the best in geometry changes and generates realistic cats/dogs with reasonable structure while CycleGAN[11] fails to perform any geometry change. For texture changes, DCLGAN consistently outperforms all other methods on the whole. This is shown on the third row of Figure 6 when CUT[9]

fails to modify the color of whole oranges.

References

[1] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International Conference on Machine Learning (ICML)*, 2020. 2

[2] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 249–256, 2010. 1

[3] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9729–9738, 2020. 2

[4] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer vision and pattern recognitio (CVPR)*, pages 770–778, 2016. 1

[5] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *Advances in neural information processing systems (NIPS)*, pages 6626–6637, 2017. 1

[6] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 1

[7] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *International Conference on Learning Representations (ICLR)*, 2014. 1

[8] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *IEEE conference on computer vision and pattern recognition (CVPR)*, pages 3431–3440, 2015. 1

[9] Taesung Park, Alexei A. Efros, Richard Zhang, and Jun-Yan Zhu. Contrastive learning for unpaired image-to-image translation. In *European Conference on Computer Vision (ECCV)*, 2020. 1, 2

[10] Maximilian Seitzer. pytorch-fid: FID Score for PyTorch. https://github.com/mseitzer/pytorch-fid, August 2020. Version 0.1.1. 1

[11] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *IEEE International Conference on Computer Vision (ICCV)*, 2017. 1, 2

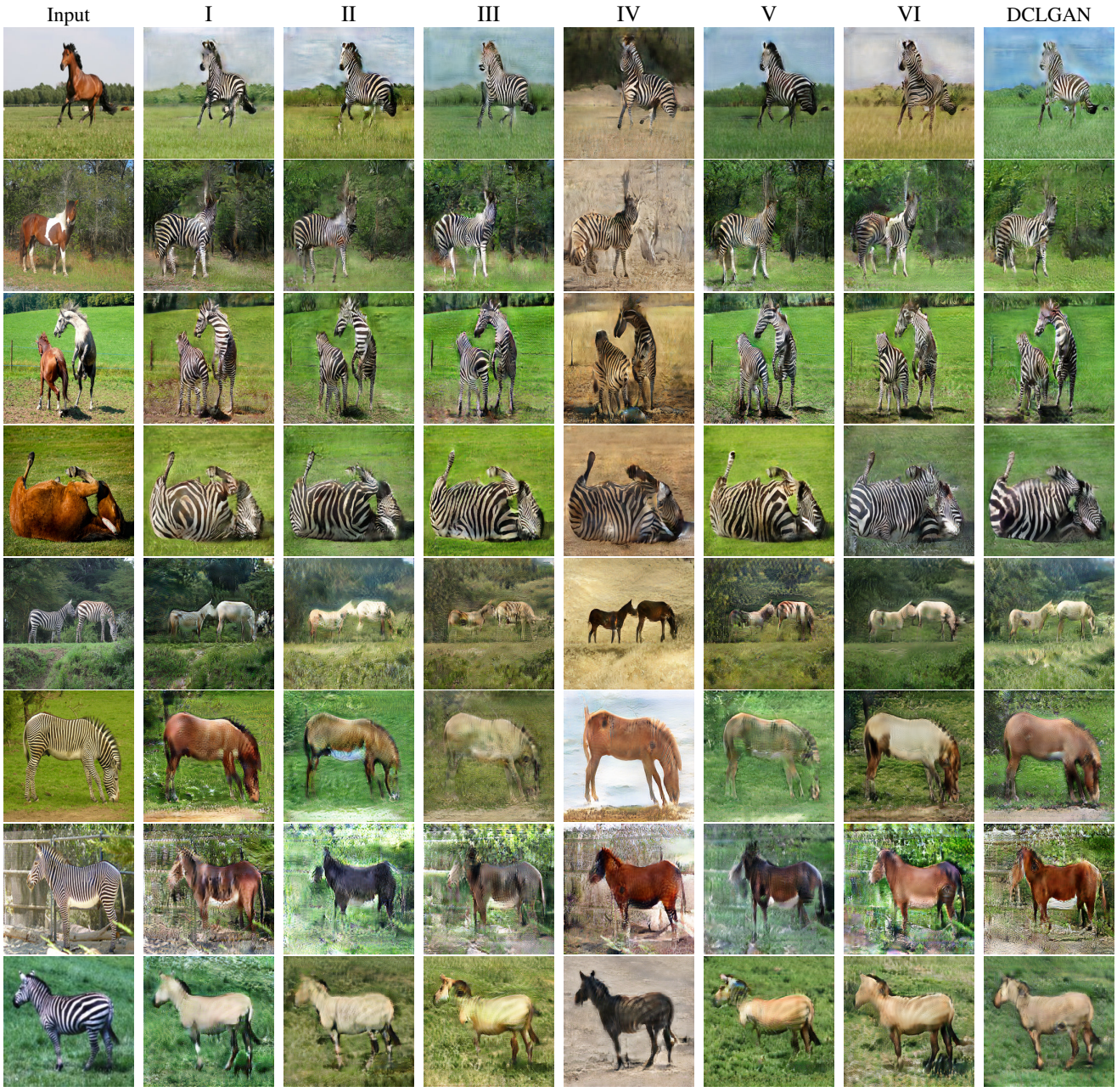

Figure 2. Qualitative results of ablations on Horse → Zebra and Zebra → Horse tasks.

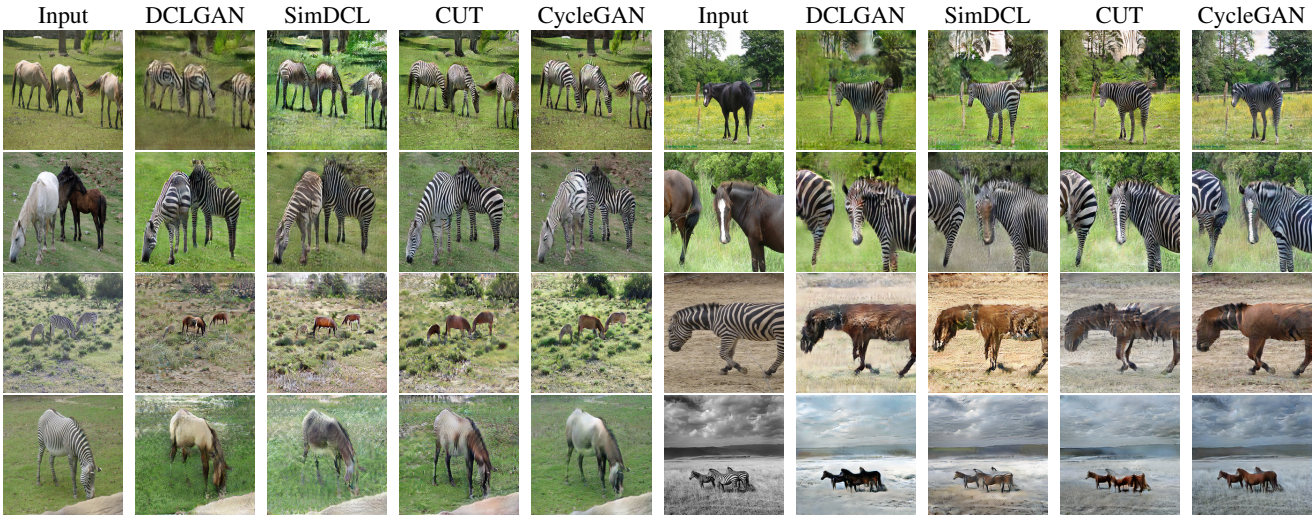

Figure 3. Additional results of Horse ↔ Zebra.

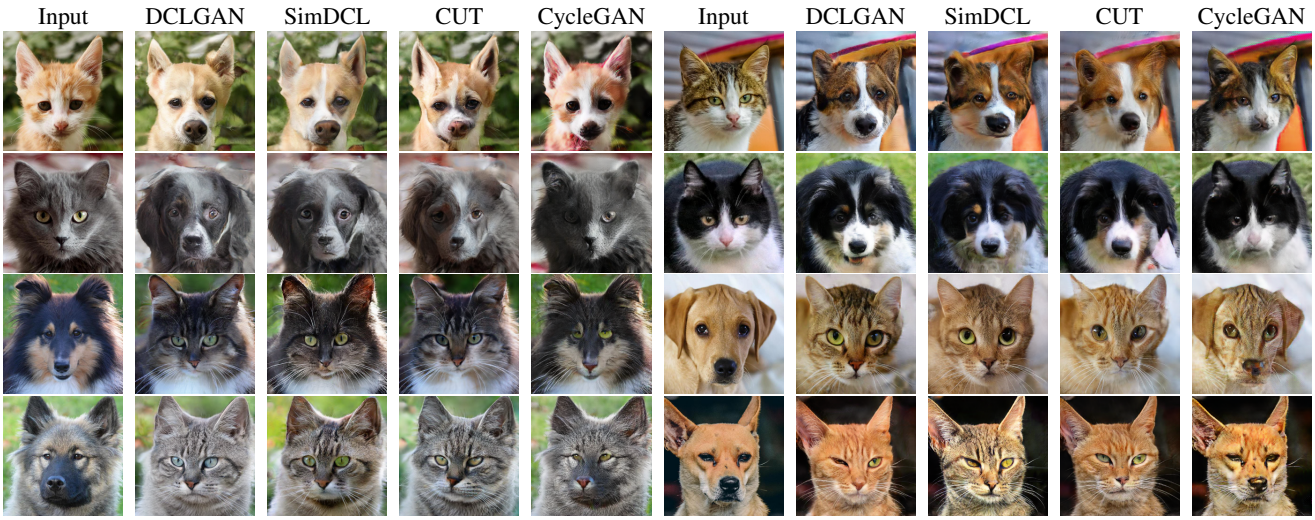

Figure 4. Additional results of Cat ↔ Dog.

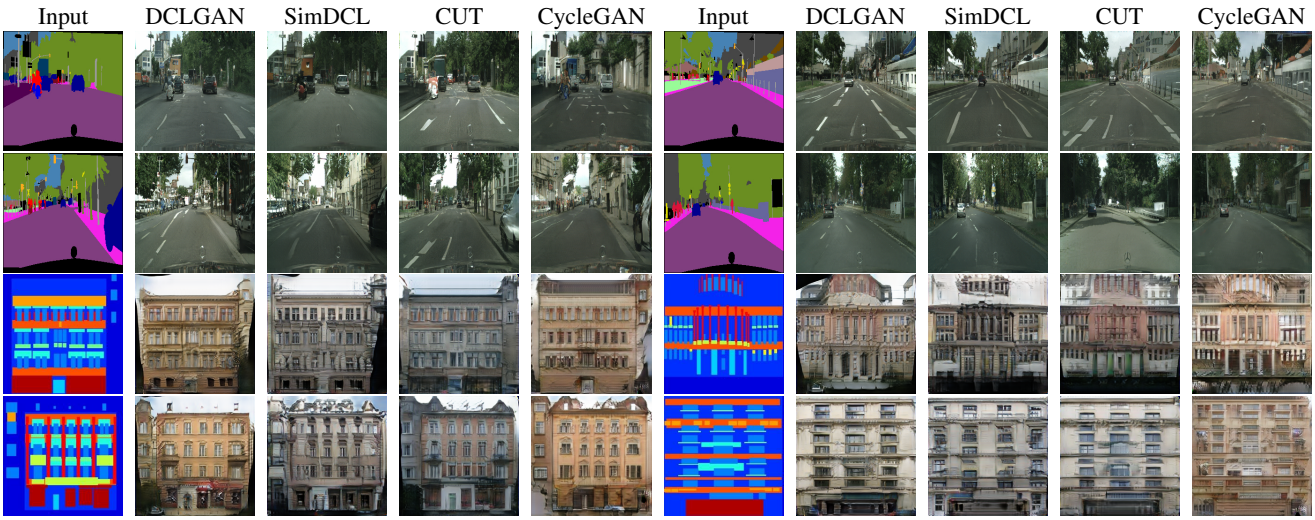

Figure 5. Additional results of CityScapes and Label → Facade.

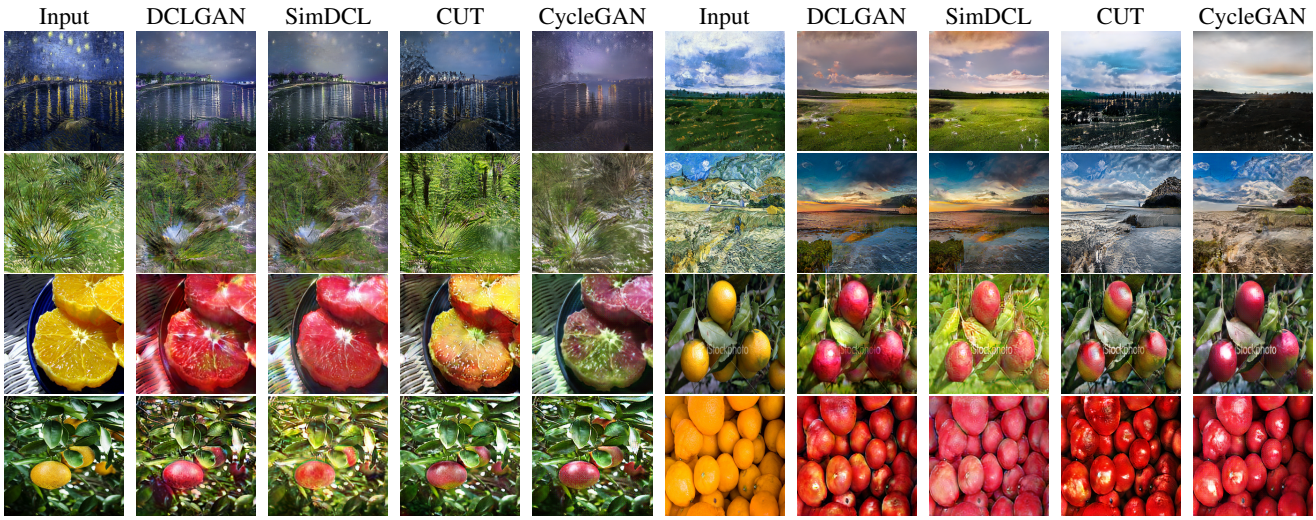

Figure 6. Additional results of Van Gogh → Photo and Orange → Apple.