Project Research Report: Photo Animation and Animation Style Image Inpainting

Hanpei Fang
School of Fundamental Science and Engineering, Waseda University
Shinjuku, Tokyo, Japan
hanpeifang@ruri.waseda.jp

ABSTRACT

In this laboratory report, I will discuss the transformation from real-world photos to animation-style pictures using generative adversarial network (GAN). Specifically, I created a dataset from a famous Japanese animation, Yurucamp, and compared the results of different model structures trained by this dataset. The application of image inpainting in animation-style images will also be explored.

KEYWORDS

Style Transformation, Image Inpainting, Generative Adversarial Networks, Animation

ACM Reference Format:
Hanpei Fang. 2022. Project Research Report: Photo Animation and Animation Style Image Inpainting. In *Proceedings of Conference Name.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/8888888.7777777

1 INTRODUCTION

Animation has become a common art form in recent years, and it has accumulated a substantial amount of popularity from all over the world. However, creating animation is very laborious and requires advanced artistic skills. For animation artists, creating high-quality animation works requires careful consideration of lines, colors, textures, ratios, and other factors, which prevents many people from creating their animation work.

Like other forms of artwork, lots of animation scenes were created based on real-world scenes. As a common artistic form, not only commercial companies and professional illustrators, enthusiasts and ordinary people may also want to turn their photos into animation-style. Moreover, for professionals, having an apparent impression before doing the work can be helpful and save much time. Therefore, the technique of transferring real-world photos to animation-style pictures automatically can be very valuable and necessary.

For another, image inpainting has been an attractive topic for many years. Lots of researchers have achieved impressive outcome on real-world images, such as [Liu et al. 2018], but for animation pictures, there are not many researches.

2 RELATED WORK

2.1 Style Transformation using GAN

Creating a satisfactory dataset of paired data is not easy, so generative adversarial network has been widely used in many image generation tasks. In CartoonGAN [Chen et al. 2018], the researchers proposed a solution to transforming photos of real-world scenes into animation-style images by proposing two novel losses: (1) a semantic content loss, which is formulated as a sparse regularization in the high-level feature maps of the VGG network to cope with substantial style variation between photos and cartoons, and (2) an edge-promoting adversarial loss for preserving clear edges. In their results, they get better generated animation images than using CycleGAN [Zhu et al. 2017].

In AnimeGAN [Chen et al. 2019], based on CartoonGAN, the authors proposed the grayscale style loss, the color reconstruction loss and the grayscale adversarial loss.

2.2 Generating Dataset from Video

For photo animation, extracting pictures from animation videos is the simplest way to generate a dataset. Simply extracting every frame from the videos is a solution, but it may include lots of repeated data, and some critical information may only be a small part of the entire dataset. In Comixify [Pęśko et al. 2018], the authors proposed a method to extract keyframes from video, which allows us to extract the critical information from the video.

2.3 Animation Style Image Inpainting

As mentioned before, image inpainting is an important and popular topic in computer vision. In EdgeConnect [Nazeri et al. 2019], the authors proposed a two-stage adversarial model EdgeConnect that comprises an edge generator followed by an image completion network, which can be used in animation-style image inpainting.

3 RESEARCH PROCESS AND RESULTS

In this section, I will discuss the research process, results, and the tools used. The code used in the experiments are implemented in PyTorch, TensorFlow, and Caffe, depending on different models.

3.1 Animation Inpainting

At the beginning of this semester, I try to use EdgeConnect to conduct animation inpainting tasks. Thanks to Sheng You's work in [You 2019], he provides the training data and test tool for this project. Unfortunately, I lost the data of this part. The author of Anime-InPainting also made another project, PI-REC [You et al.

2019], by extending EdgeConnect's idea. PI-REC is a very interesting project, in which user can generate animation character (or other things) by simply inputing a rough sketch and color blocks.

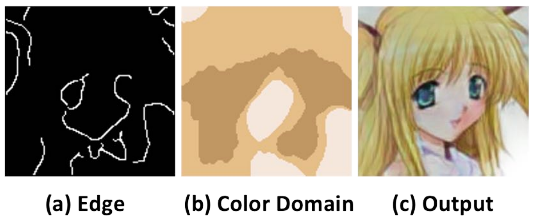


Figure 1: Input and Output of PI-REC.

In Figure 1, the edge and color domain is the input of this model, and the output is the generated image.

3.2 Dataset

The dataset used in the following experiments is created from a famous Japanese animation, Yurucamp. The method introduced in Comixify [Pęśko et al. 2018] is used but the last aesthetis estimation procedure is omitted to obtain more frames.

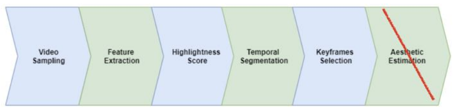


Figure 2: Keyframes Extraction Method in Comixify[Pęśko et al. 2018].

By introducing this method, I created a dataset containing 2,828 images.

3.3 CartoonGAN

The code used in this experiment [Chen et al. 2018] is provided by mnicnc404 in [mnicnc404 2021]. In this experiment, the animation dataset is pre-processed into 62,243 256 × 256 images by scaling and croping, and the real world photo dataset contains 6,657 images. The result of this experiment is shown in Figure 3.

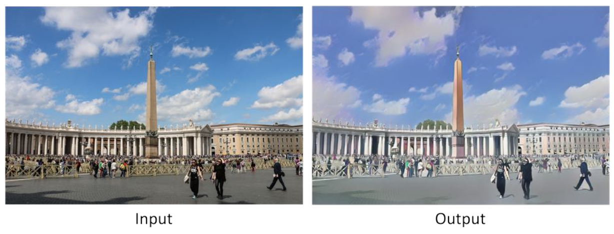


Figure 3a: Experiment Result of CartoonGAN.

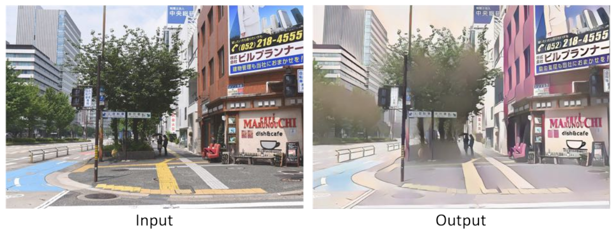


Figure 3b: Experiment Result of CartoonGAN.

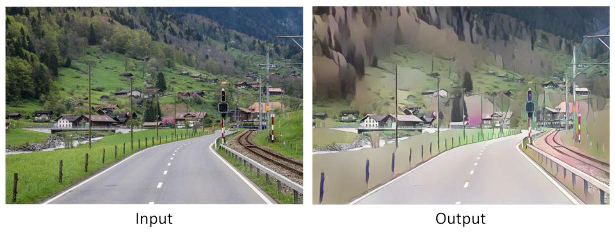


Figure 3c: Experiment Result of CartoonGAN.

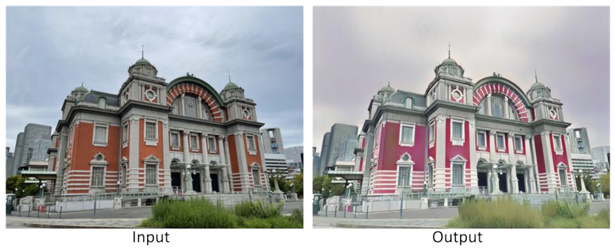


Figure 3d: Experiment Result of CartoonGAN.

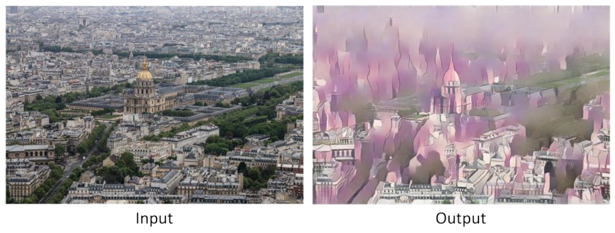


Figure 3e: Experiment Result of CartoonGAN.

As the results shown in the figures, the model can produce a good result in some cases, such as Figures 3a and 3d tend to over-process complex patterns, like tree and grass in Figures 3b, 3c, and 3e.

3.4 AnimeGAN

The code used in the first part of this experiment is the v2 version provided by the author in [Chen 2020]. In the first part of this experiment, the same animation and real-world photo datasets used are the same as the ones in the experiment of CartoonGAN.

In addition, a smooth dataset including the smoothed images of the animation dataset is used. The result is shown in Figure 4.

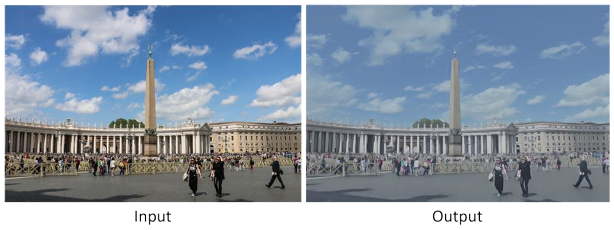


Figure 4a: Experiment Result of AnimeGANv2.

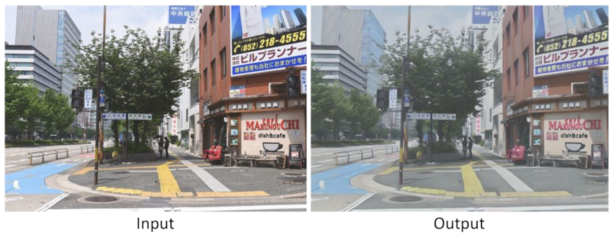


Figure 4b: Experiment Result of AnimeGANv2.

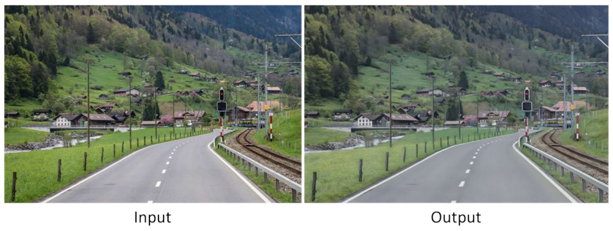


Figure 4c: Experiment Result of AnimeGANv2.

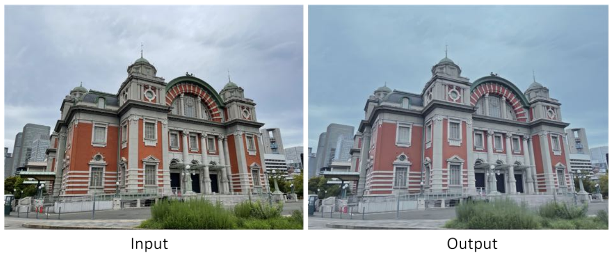


Figure 4d: Experiment Result of AnimeGANv2.

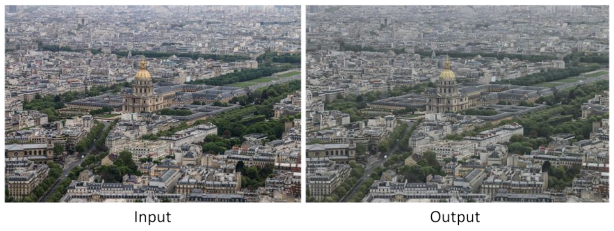


Figure 4e: Experiment Result of AnimeGANv2.

From the results shown in Figure 4, AnimeGANv2 does prevent the over-processing problem in CartoonGAN and keep most of the features in the original photos. However, it is exactly the problem. Compared with CartoonGAN, it is hard to say the outputs are animation-style images. For another, all of the generated images are unnaturally gray, which makes the images weird.

3.5 Modified Model

To improve the outcome, I try to generate the dataset online. The size of the animation dataset is 2,828, and the images will be scaled and cropped randomly in each iteration. The real-world datasets used in this experiment include a small one which is the same as the dataset used previously, and a larger one, the Places dataset [Zhou et al. 2017]. I also try to support multi-GPU training, but it is not easy with TensorFlow. Therefore, I implement a PyTorch version (source code is available at https://github.com/Allen931/AnimeGAN-torch). Since byyandlee has implemented the generator in [bryandlee 2021], I use the implementation directly and implement the remaining part.

In this part, I tried different losses, generator structures and discriminator structures. After dozens of experiments on different combinations, the generator structure of the currently best combination is shown as Figure 5, and the discriminator and loss are same with the implementation in AnimeGANv2 [Chen 2020].

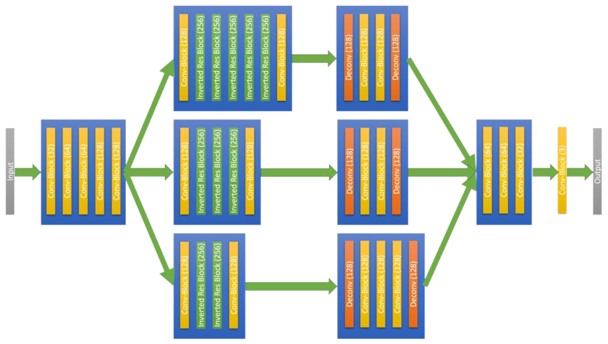


Figure 5: Modified Generator Structure.

The Deconv block used in the generator is proposed by Odena et al. in [Odena et al. 2016]. The optimizer used in the modified model is also changed from Adam to RMSProp since the authors of LSGAN's suggestion in [Mao et al. 2017]. The experiment result of the modified is shown in Figure 6.

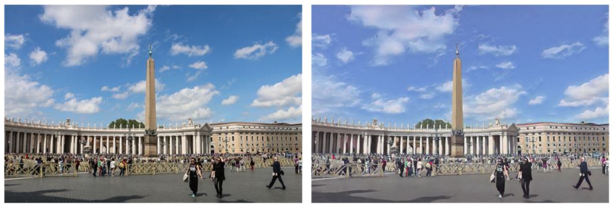


Figure 6a: Experiment Result of Modified Model.

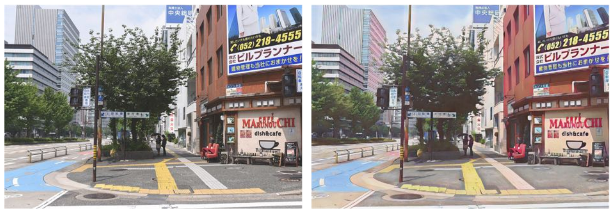


Figure 6b: Experiment Result of Modified Model.

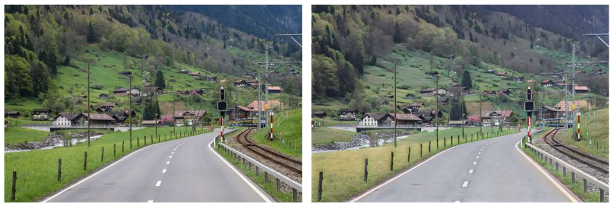


Figure 6c: Experiment Result of Modified Model.

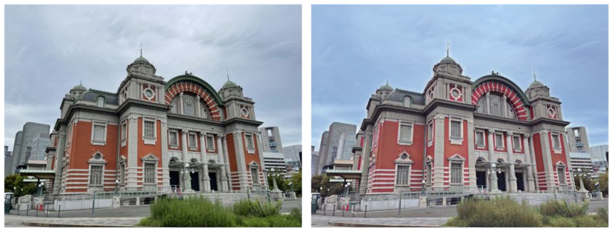


Figure 6d: Experiment Result of Modified Model.

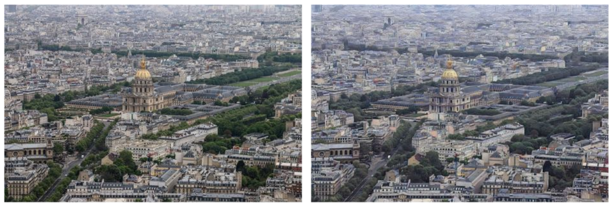


Figure 6e: Experiment Result of Modified Model.

The modified model performs better than the original AnimeGANv2 with the Yurucamp dataset in most all cases. Compared with CartoonGAN, it prevents the over-processing phenomenon in most cases, while CartoonGAN can have better performance in some scenes, such as Figure 3/6a. However, it should be noted that the modified produces strange black and white stripe in the case of complex patterns, such as the stone on the riverbank in Figure 6c and the leaves in Figure 7, which does not appear in CartoonGAN and is negligible in AnimeGANv2.

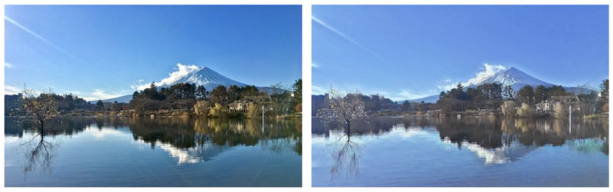


Figure 7a: Example of Strange Stripe.

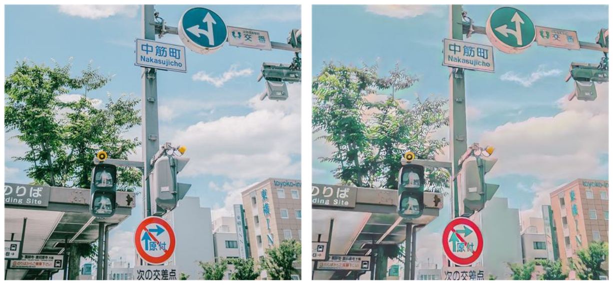


Figure 7b: Example of Strange Stripe.

4 CONCLUSIONS

In conclusion, the modified model has the best average performance with the Yurucamp dataset but results in a strange black and white stripe on some complex parts, such as trees and grass, of input photos. CartoonGAN has the best performance in some cases, but in lots of scenes, over-processing is a problem. Unfortunately, none of them processes trees satisfactorily. CartoonGAN tends to turn the tree into a blurred color block, while the results of the other two models are very similar to the inputs or include strange stripes.

For the animation-style image inpainting part, PI-REC inspired by EdgeConnect is an interesting and promising project, although it could only work well in some simple cases currently, such as animation characters, handbags and shoes.

ACKNOWLEDGEMENTS

Throughout the project research this semester, I have received a great deal of support and assistance. I want to acknowledge Professor Simo-Serra for his guidance, and I would also like to thank everyone in Simo-Serra Lab for their patient support.

REFERENCES

bryandlee. 2021. animegan2-pytorch. https://github.com/bryandlee/animegan2-pytorch.

Jie Chen, Gang Liu, and Xin Chen. 2019. AnimeGAN: A Novel Lightweight GAN for Photo Animation. Github: https://github.com/TachibanaYoshino/AnimeGAN, GitHub of v2: https://github.com/TachibanaYoshino/AnimeGANv2.

Xin Chen. 2020. AnimeGANv2. https://github.com/TachibanaYoshino/AnimeGANv2.

Yang Chen, Yu-Kun Lai, and Yong-Jin Liu. 2018. CartoonGAN: Generative Adversarial Networks for Photo Cartoonization. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 9465–9474. https://doi.org/10.1109/CVPR.2018.00986

Guilin Liu, Fitsum A. Reda, Kevin J. Shih, Ting-Chun Wang, Andrew Tao, and Bryan Catanzaro. 2018. Image Inpainting for Irregular Holes Using Partial Convolutions. In *Proceedings of the European Conference on Computer Vision (ECCV).*

Xudong Mao, Qing Li, Haoran Xie, Raymond Y.K. Lau, Zhen Wang, and Stephen Paul Smolley. 2017. Least Squares Generative Adversarial Networks. In *2017 IEEE International Conference on Computer Vision (ICCV)*. 2813–2821. https://doi.org/10.1109/ICCV.2017.304

mnicnc404. 2021. CartoonGan-tensorflow. https://github.com/mnicnc404/CartoonGan-tensorflow.

Kamyar Nazeri, Eric Ng, Tony Joseph, Faisal Qureshi, and Mehran Ebrahimi. 2019. EdgeConnect: Structure Guided Image Inpainting using Edge Prediction. In *The IEEE International Conference on Computer Vision (ICCV) Workshops.*

Augustus Odena, Vincent Dumoulin, and Chris Olah. 2016. Deconvolution and Checkerboard Artifacts. *Distill* (2016). https://doi.org/10.23915/distill.00003

Maciej Pęśko, Adam Svystun, Paweł Andruszkiewicz, Przemysław Rokita, and Tomasz Trzciński. 2018. Comixify: Transform video into a comics. arXiv:1812.03473 [cs.CV]

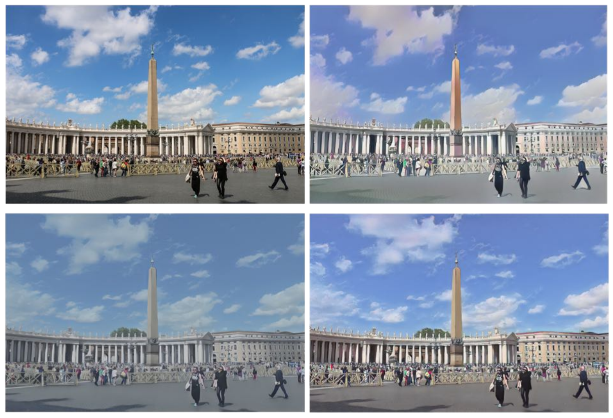
Sheng You. 2019. Anime-InPainting. https://github.com/youyuge34/Anime-InPainting.

Sheng You, Ning You, and Minxue Pan. 2019. PI-REC: Progressive Image Reconstruction Network With Edge and Color Domain. *arXiv preprint arXiv:1903.10146* (2019).

Bolei Zhou, Agata Lapedriza, Aditya Khosla, Aude Oliva, and Antonio Torralba. 2017. Places: A 10 million Image Database for Scene Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2017). http://places2.csail.mit.edu/

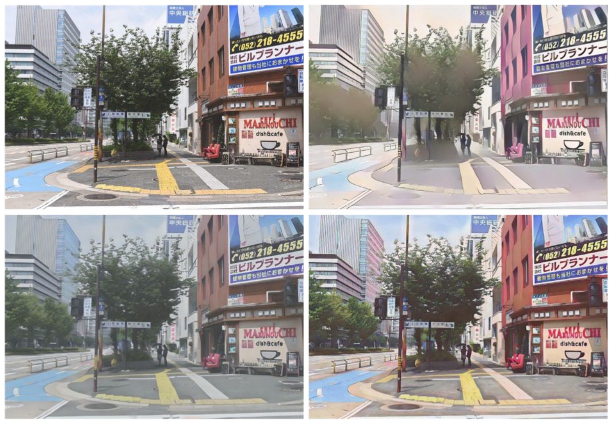
Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. 2017. Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks. In *Computer Vision (ICCV), 2017 IEEE International Conference on.*

A RESULTS COMPARISON



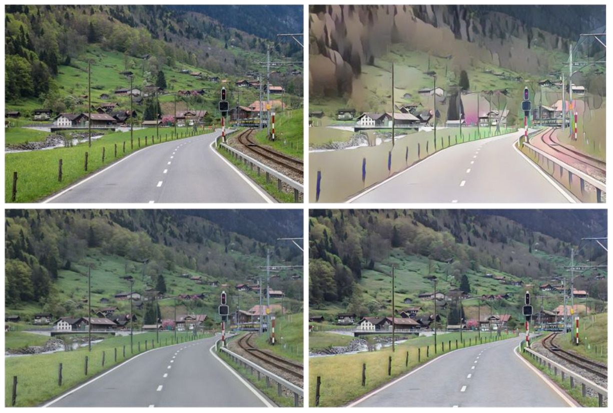
(a) Input (b) CartoonGAN
(c) AnimeGANv2 (d) Modified

Figure A.1: Results Comparison.



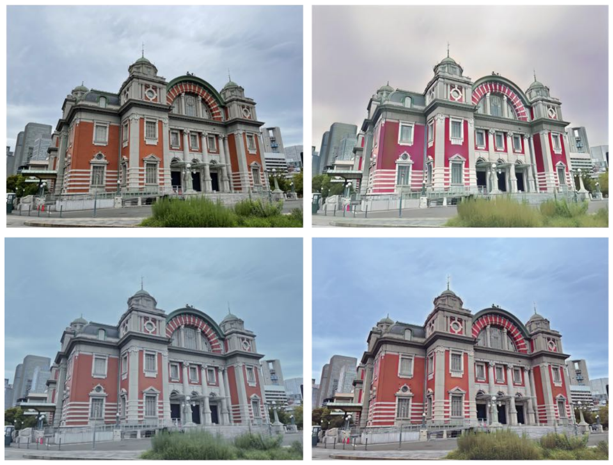
(a) Input (b) CartoonGAN
(c) AnimeGANv2 (d) Modified

Figure A.2: Results Comparison.



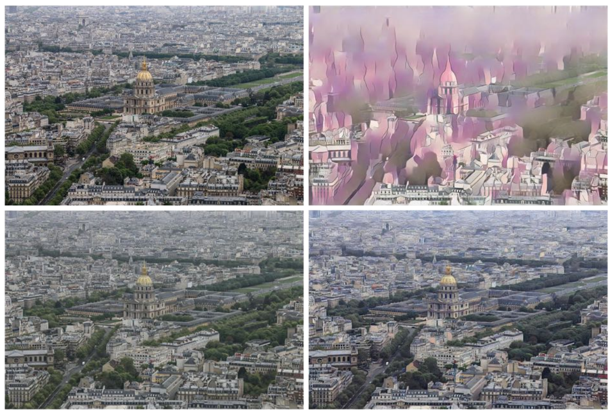
(a) Input (b) CartoonGAN
(c) AnimeGANv2 (d) Modified

Figure A.3: Results Comparison.



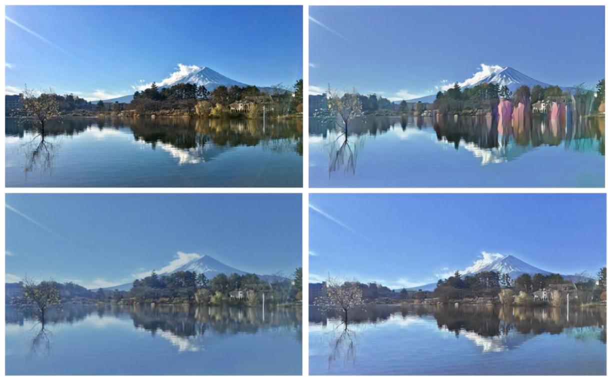
(a) Input (b) CartoonGAN
(c) AnimeGANv2 (d) Modified

Figure A.4: Results Comparison.



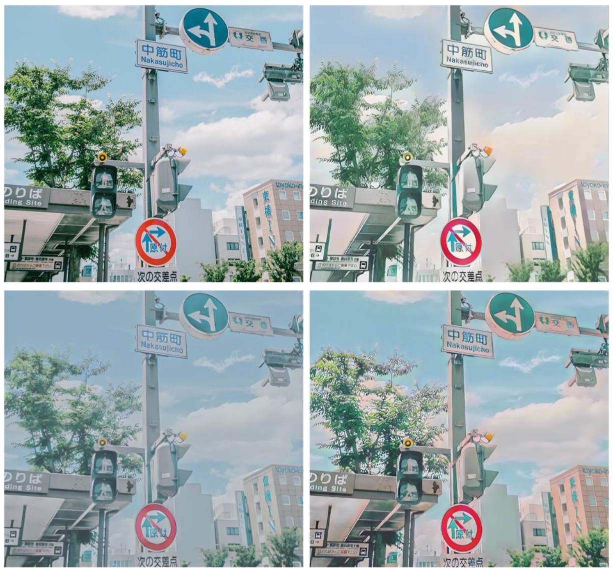
(a) Input (b) CartoonGAN
(c) AnimeGANv2 (d) Modified

Figure A.5: Results Comparison.



(a) Input (b) CartoonGAN
(c) AnimeGANv2 (d) Modified

Figure A.6: Results Comparison.



(a) Input (b) CartoonGAN
(c) AnimeGANv2 (d) Modified

Figure A.7: Results Comparison.