Group 5

Cartoon to human recovery with Cycle-GAN

Li Yu, Hong harry.cs09@nycu.edu.tw
Szu-Wei, Chen szuwei15.cs09g@nctu.edu.tw
Yu-Lin, Kuo samlong6108.cs09@nycu.edu.tw

Introduction

Recently, Voila AI Artist has been an overnight sensation and goes viral worldwide. It's a new application that people can convert any picture into a realistic Disney-like cartoon has a hit on social media platforms including Facebook, Instagram and Twitter. This new viral app has been developed by **Wemagine.ai** and is available for animated 2D and 3D conversion. Our goal is to reconvert the cartoon image back to the original person image with the cycle gan model, meanwhile combining the blending techniques. In this work, we have collected the dataset on our own and implemented the Cycle Gan model training to obtain the final result.

Related work

In this work, we have thought about the Original GAN[3] model and Cycle GAN[4] model to train and modify from these two bases. The paper [5] provides a better framework for cycle GAN by using a weighting on the loss function which sets the lower weighting for cycle loss in earlier training and higher weighting for cycle loss in later training . In addition, this paper also revised the cycle loss with weighting combining the pixel wise loss and the feature wise (with CNN) loss which in of feature wise loss should start low because discriminator features are not good at beginning, and gradually linearly increase to a high value close but not equal to 1 because some fraction of pixel level consistency is needed to prevent excessive hallucination on background and unrelated objects in the images. The Conference NIPS 2016 [6] has shown some useful tips for improving GAN model training by soft labeling. When training the discriminator, we can soft and noisy labels by using a random number between 0 and 0.1 to represent labels 0 (real images) and a random number between 0.9 and 1.0 to represent labels 1(generated images). The paper [7] analyzes several of its characteristic artifacts, and proposes changes in both model architecture and training methods to address them. They also redesign the

instance normalization layer , analyze alternative network architectures and the resolution usage issue .

Design and Implementation

- ### Dataset

  To accomplish this work, we need to collect the dataset on our own since there are no open dataset related to it. So we have used the instagram crawler InstaLooter [2] to crawl 20000 photos with hashtag voila. Furthermore, we have to make sure the photos are in the cartoon style we wanted. Therefore, we have filtered the dataset to form with four segments like Img.1. Next, we preprocessed our photos to the 2 images we want which are the Real and Cartoon images. Finally, we have gathered a dataset with Real it's Cartoon picture with size 10509 pairs of images.

Img.1

- ### Model Architecture

  Downsample and Upsample block:
  As shown in figure1 , downsample block let the width and height shrink twice but the channel doubled . And the upsample block lets width and height doubled but the channel shrinks twice .

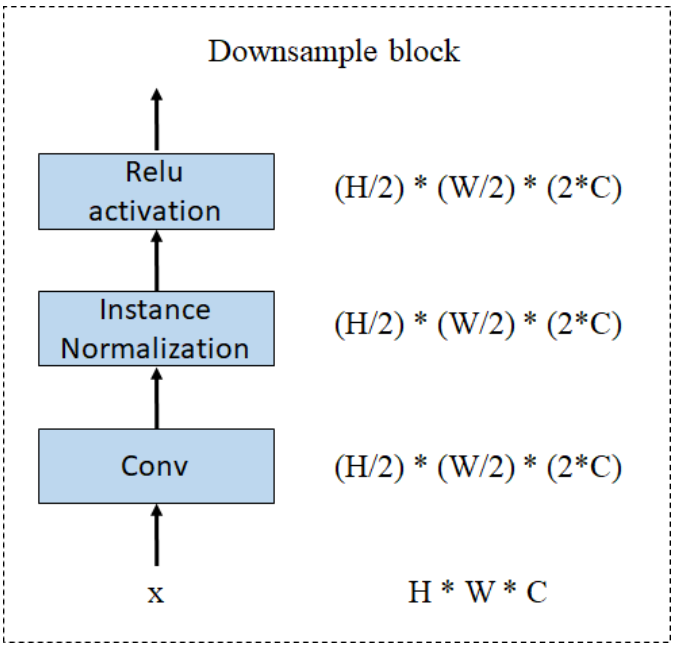


Figure1. The structure of Downsample block

Residual block :

As shown in figure 2 , the residual block has the skip connection to avoid the gradient vanishing and two convolution layers .

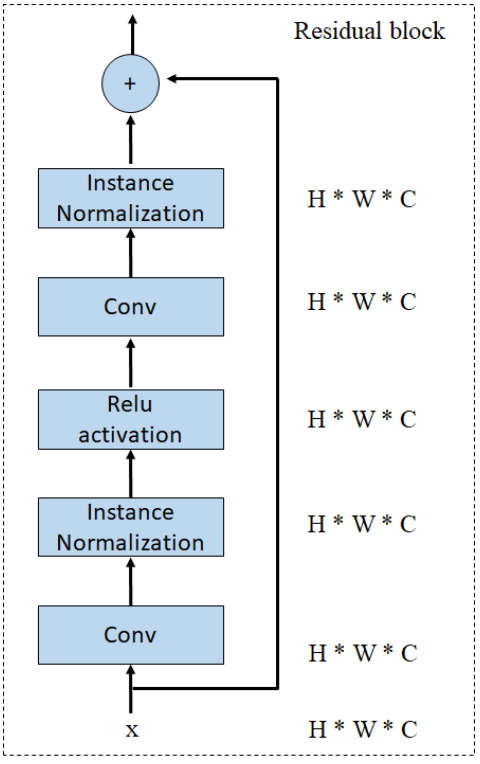


Figure 2 . The structure of the residual block .

Generator :

The structure of generator has shown in Figure 3

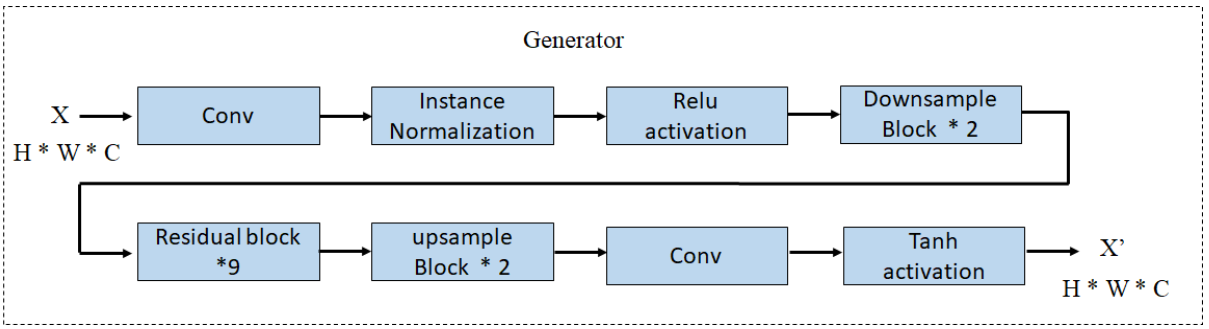


Figure 3. The structure of the Generator .

Discriminator:

The structure of the discriminator is shown in Figure 4 . And followed by [7] reveals that using skip-connections or residual blocks in the Network could improve the performance , so we decided to design another discriminator , shown in figure 5 .

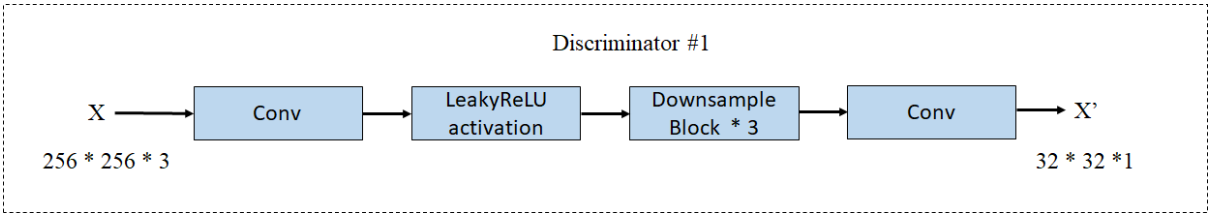


Figure 4. The structure of Discriminator #1

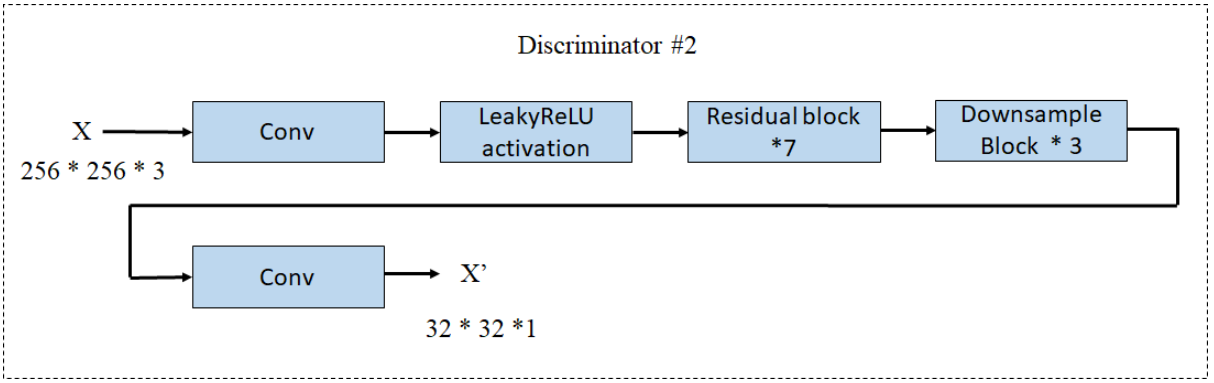
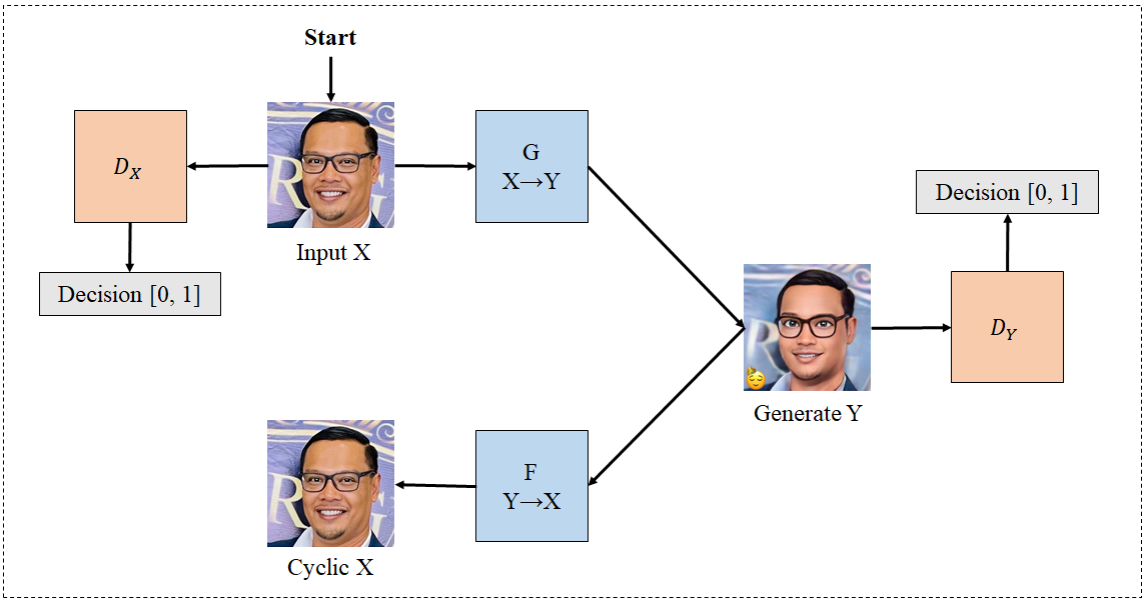


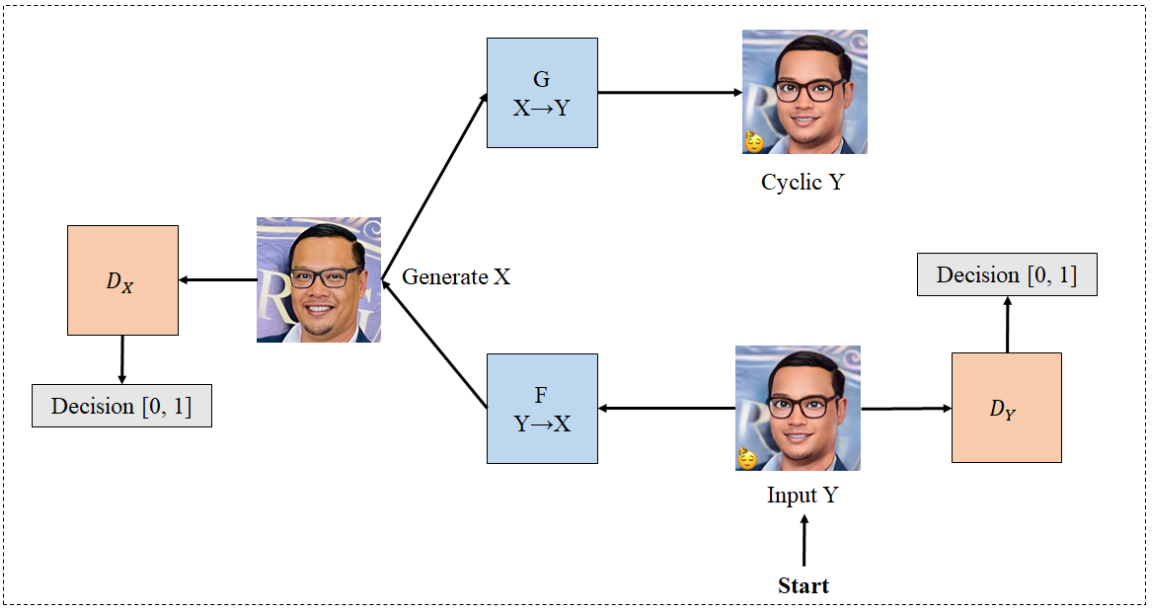
Figure 5. The structure of Discriminator #2

- **Procedure**

The two image domains are denoted as X and Y . Generate G takes domain X image to domain Y image . Generate F takes domain Y image to X image . Discriminator D_X and D_Y distinguish the authenticity of X domain and Y domain images respectively. First, we convert input X with Generator G into domain Y then using another Generator F to convert Y image back to domain X as Cyclic X. Furthermore, the original Input X

and the Generate Y will be put into Discriminators to tell whether the X is real or fake and Y is real or fake. In the opposite direction, the procedure will be the same as well to build a full cycle. Eventually, the model is formed by 2 discriminators and 2 generators.





- ## Loss function

In this part , we will analyze the loss function we used . To further improve the performance , we revised some details of the loss function .

Base :

Base GAN loss :

$$L_{GAN}(G,\ D_Y,\ X,\ Y)\ =\ E_{y_{\sim}\ p_{data}(y)}[log\ D_Y(y)]\ +E_{x_{\sim}\ p_{data}(x)}[log(1\ -D_Y(G(x))] \quad (1)$$

Base Cycle loss :

$$L_{cyc}(G,\ F,\ X)\ =\ E_{x_{\sim}\ p_{data}(x)}[||F(G(x))\ -\ x||_1] \quad (2)$$

Base identity loss :

$$L_{identity}(G,\ Y)\ =E_{y_{\sim}\ p_{data(Y)}}[G(y)\ -\ y]_2 \quad (3)$$

where generator G can transform style from X domain to Y domin . So identity loss could ensure the output has the same style as input when input is Y domain .

By integrating equation (1), equation (2) and equation (3) loss , the full objective for CycleGAN is

$$L(G,\ F,\ D_X,\ D_Y)\ =\ L_{GAN}(G,\ D_Y,\ X,\ Y)\ +\ L_{GAN}(F,\ D_X,\ Y,\ X)\ +\ \lambda L_{cyc}(G,\ F,\ X)$$
$$+\ \lambda L_{cyc}(F,\ G,\ Y)\ +\ L_{identity}(G,Y)\ +\ L_{identity}(F,\ X) \quad (4)$$

Generator loss :

We follow the simple idea from [8] . The equation (5) loss could allow the generated image to be close to the ground truth , and the equation (6) loss lets the generated image close to the value of ground truth image after passing discriminator , in other words , it can confuse Discriminator more . Due to the discriminator has poor ability in the earlier epoch , so purpose large weight coefficient λin the beginning , and linearly reduction when training.

$$L_{l_2}(G,\ x,\ x_g)\ =\ ||G(x)\ -\ x_g||_2 \quad (5)$$

$$L_{adv}(G,\ x,\ x_g)\ =\ max_D E_{x\in X}|D(x_g)\ -\ D(G(x))| \quad (6)$$

$$L_{generator}(G,\ x,\ x_g)\ =\ \lambda L_{l_2}(G,\ x,\ x_g)\ +\ (1\ -\ \lambda)L_{adv}(G,\ x,\ x_g) \quad (7)$$

where G denotes the Generator , x denotes input image, x_g denotes ground truth of $G(x)$, λdenotes weight coefficient .

Cycle consistency loss :

Followed by [8] . Because information is always lost when reconstruction , instead of reconstruction image by pixel-level , it should focus on

general structures . So [8] proposed feature extractor f_{D_X}, which outputs the middle layer of the discriminator , it can represent the feature of the image . Same concept as before , it has small weight coefficient γin the earlier training phase and linearly grow up when training .

$$L_{cyc}(G, F, D_X, X, \gamma) = E_{x \sim p_{data}(x)}[\gamma||f_{D_X}(F(G(x)) - f_{D_X}(x)||_1 + (1-\gamma)||F(G(x)) - x||_1] \quad (8)$$

Soft Label :

Label Smoothing, i.e. if you have two target labels: Real=1 and Fake=0, then for each incoming sample, if it is real, then replace the label with a random number between 0.9 and 1, and if it is a fake sample, replace it with 0.0 and 0.1 (for example).

- # Experiment settings

In order to train our models, we have used 3 servers to train on the docker setup environments. The servers are equipped with GTX 1080 Ti, RTX 3080 and GTX 1080 Ti respectively. We have finally set our training set to about 5000 pictures and testing set about 200 pictures. Eventually, we have trained about 200 epochs for each model. Each epoch takes around 20 - 30 minutes.

Model :

Model name	Modified
Gan v1	None
Gan v2	generator loss
Base cycle gan	None
Cycle gan v3	generator loss
Cycle gan v4	generator loss + Cycle consistency loss
Cycle gan v5	generator loss + Cycle consistency loss + Soft Label

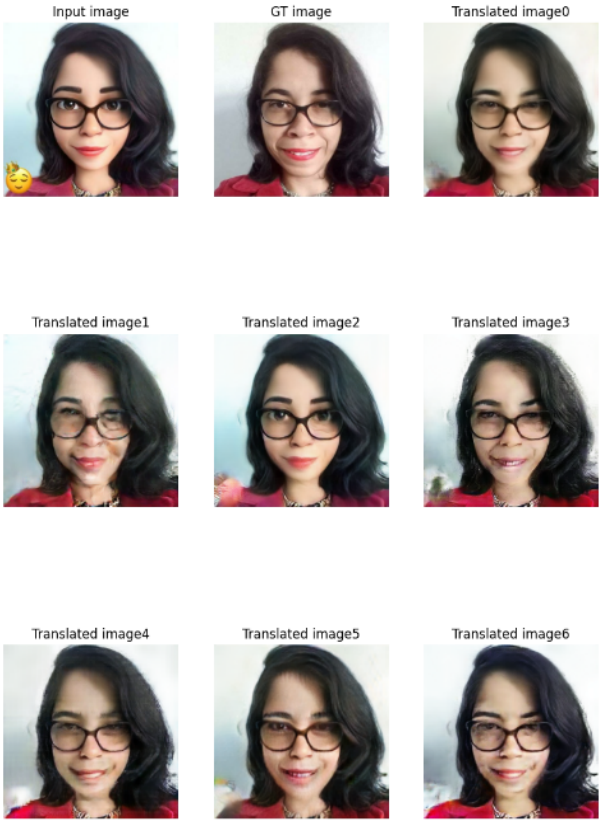
Cycle gan v6	generator loss + Cycle consistency loss + Soft Label + discriminator network

Evaluation and Results

We have referenced the way provided by the paper [8], which states that “In the convolutional space of a ResNet pretrained on ImageNet, both MMD and 1-NN accuracy appear to be good metrics in terms of discriminability, robustness and efficiency”. Therefore, we have decided to use these two approaches and the simplest manual evaluation to evaluate our result.

- **Manual Evaluation**

  Each time, a composite image x is shown to the subjects followed by three blended images produced by five different algorithms. The subjects are told to pick the most realistic image among these three blended images. Given the result from some models and shuffled randomly , the reviewer will vote for the best model fit on these images.



There are 3 reviewers reviewing 50 pictures above and voting for the best model.

model	Reviewer1	Reviewer2	Reviewer3	Final votes
Gan v1	6	5	7	18
Gan v2	6	10	9	25
Base cycle gan	3	2	6	11
Cycle gan v3	17	14	10	**41**
Cycle gan v4	8	13	5	26
Cycle gan v5	3	2	1	6 worse!
Cycle gan v6	7	4	12	23

- **Kernel MMD**

$$MMD^2(P_r, P_g) = E_{x_r, x'_r \sim P_r, x_g, x'_g \sim P_g}[k(x_r, x'_r) - 2k(x_r, x_g) + k(x_g, x'_g)]$$

$$y = (x/(x_r, x'_r).mean * 2 * beta * beta)$$

A measure of dissimilarity between P_r and P_g for some fixed kernel function k. Given two sets of samples from P_r and P_g, the empirical MMD between the two distributions can be computed with finite sample approximation of the expectation. A lower MMD means that P_g is closer to P_r, which is a sign for better performance.

- **1NN**

1NN is used in two-sample tests to assess whether two distributions are identical . Given two sets of samples $S_r \sim P_r^n$ and $S_g \sim P_g^m$, with $|S_r| = |S_g|$, one can compute the leave-one-out accuracy of a 1-NN classifier trained on S_r and S_g with positive labels for S_r and negative labels for S_g. Classifier should yield a ~50% leave-one-out accuracy when $|S_r| = |S_g|$ is large, which means the 1NN classifier can't separate these two sets well as Figure 1. If Classifier yields a ~0% accuracy, it means the two sets of

images are close but in opposite labels as Figure 2. If Classifier yields a ~100% accuracy, it means GAN generates a completely different set of pictures and is able to separate 2 sets easily as Figure 3.

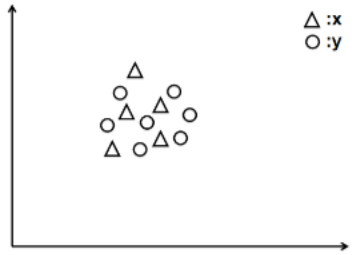


Figure 1

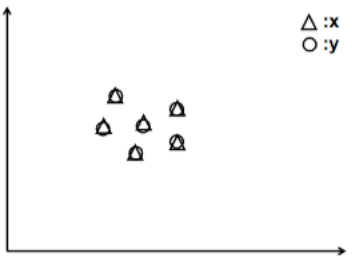


Figure 2

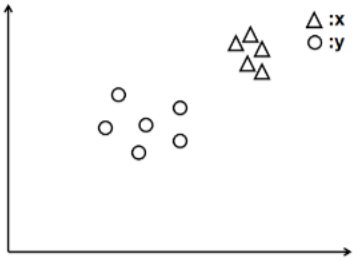


Figure 3

- **results of mmd and knn evaluation**

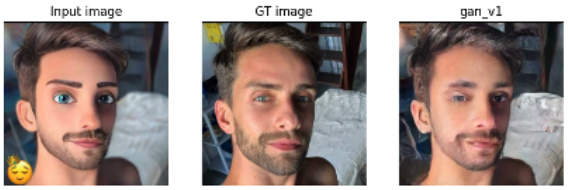
Model name	mmd ↓ (epoch)	1nn (epoch)
Gan v1	0.05003(161)	0.86329 (162)
Gan v2	0.05139 (134)	0.87341 (199)
Base cycle gan	0.05865 (124)	**0.83544** (126)
Cycle gan v3	**0.04959** (141)	0.87088 (149)
Cycle gan v4	0.05114 (159)	0.87848 (89)
Cycle gan v5	0.05170 (85)	0.88354 (102)
Cycle gan v6	0.05144 (54)	0.87594 (39)

- **v3 model: different epoch**

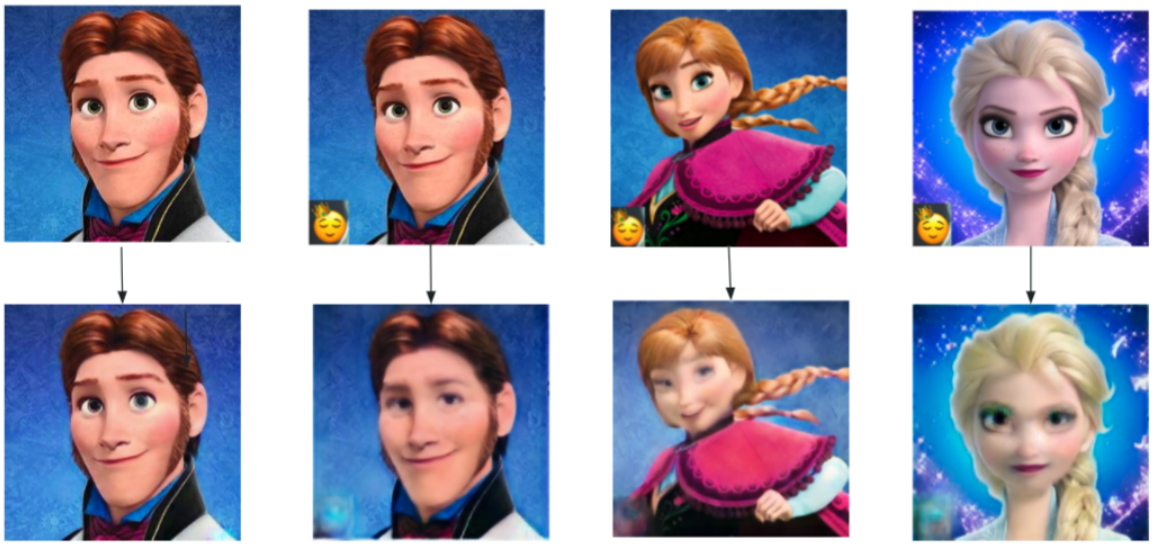
epoch	10	50	100	141	149	156
mmd	0.06171	0.07006	0.05726	**0.04959**	0.05548	0.05359
1NN	0.92405	0.94177	0.92405	0.92405	**0.87088**	0.92658

Group 5

- **result of each model**



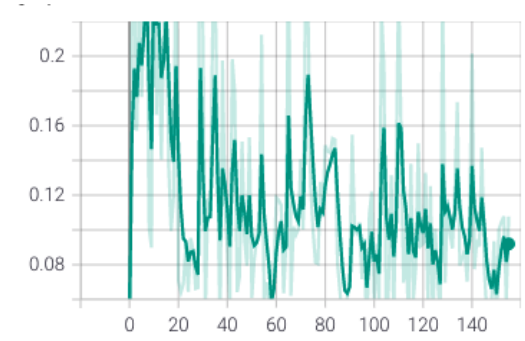
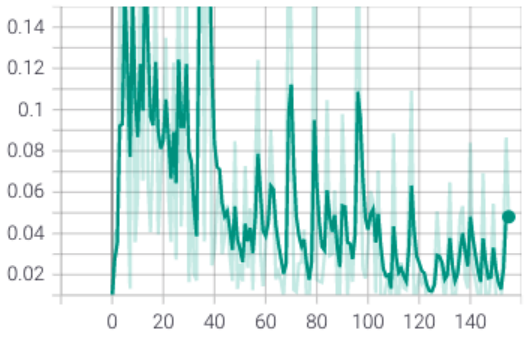
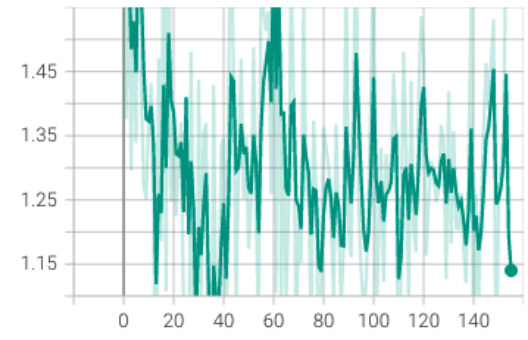
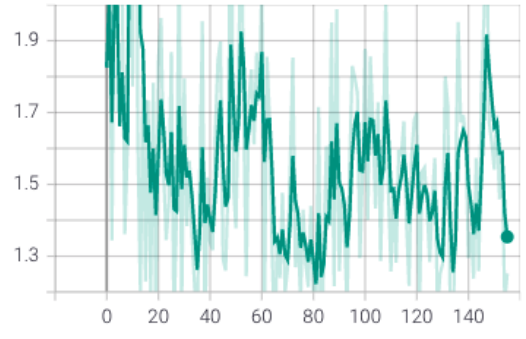
- **v3 model: images from disney cartoon**



- **v3 model: Loss**

  x axis: epoch

  y axis: loss

Dx	Dy
	
G (X→ Y)	F(Y→X)
	

Discussion & Conclusion

- In this project, we use original gan model and cycle gan model with different settings and tricks to convert the cartoon style person to real style one. According to the table of results of MMD and 1NN evaluation, the MMD evaluation of the cyclegan v3 model is the best, while the performance of the cyclegan base model with 1NN evaluation outperforms others.
- We modify the loss function and architecture of the models which reference those related works to improve our model. Each higher version is what we expect to get better performance. However, we do not see a lot of difference from the results. There are more to be concerned about, if we want to do more research on this topic in the future. The reason why v5 and v6 do not get better results might be due to the fact that we don't have enough time to train more epochs.
- As what we mentioned above, if the 1NN evaluation is close to 50%, we consider the model performs well. However, the table of the 1NN evaluation shows that all results are close to 90% which is considered that the model is not good at generating the target images. We assume that this evaluation is not suitable in this case to compare the performances, and on the other hand, our models can not transfer the images really well, so the generated images and original real images are easy to seperate. From the table of the different epochs of v3 model, the 1NN results are always high and hardly become lower.
- We trust the result of MMD evaluation more since we have the ground truth of the input images and we expect the aligned pixel values are similar, of which MMD evaluation can show this property. Moreover, the MMD value becomes lower when we train the model longer, which can be confirmed by the table of the different epochs of v3 model.
- Following the previous point, we choose the models with the epoch which have the best MMD result to do the manual evaluation. The results are similar to what we get in the MMD result with slight differences due to the subjective judgement on manual evaluation.
- It is easy to see that the losses of the discriminators decrease continuously. Both losses of the discriminators and generators change a lot during training, and the generator seems hard to converge. We

understand that it is not easy to train the GAN model because the generator and discriminator are competing.

- According to the results above, the original GAN models are able to generate the target image as good as some cycle GAN models. We can consider different GAN architecture, such as Style GAN or some SOTA work, to get better performance in the future.
- We find some interesting points when implementing this project. First, we found that a man with a moustache can be converted well often while children and infants are usually converted awful. We suppose that a moustache is a good feature to focus when converting a person to real style. The face ratio of children is hard to recognize.
- We infer the model with some Disney image. At the beginning, we find that the image does not change at all. We figure out that since all the training images have an emoji on the left bottom, the model seems to learn this feature as a style. It considers if images without emoji are real style, which is not necessary to be converted due to the model design. After we paste an emoji on the image, the image is able to be transformed although the Disney images can not transform equally well as images from the original cartoon style.

Reference

[1] Voilà AI Artist: Create your cartoon avatar using viral app, Link
[2] InstaLooter, Link: https://github.com/althonos/InstaLooter
[3] Generative Adversarial Networks,author={Ian J. Goodfellow and Jean Pouget-Abadie and Mehdi Mirza and Bing Xu and David Warde-Farley and Sherjil Ozair and Aaron Courville and Yoshua Bengio}, year={2014}, eprint={1406.2661}, archivePrefix={arXiv},primaryClass={stat.ML}}
[4] Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks, author={Jun-Yan Zhu and Taesung Park and Phillip Isola and Alexei A. Efros}, year={2020}, eprint={1703.10593}, archivePrefix={arXiv}, primaryClass={cs.CV}}
[5]CycleGAN with Better Cycles
[6]Ian Goodfellow OpenAI, NIPS 2016 Tutorial: Generative Adversarial Networks
[7]Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. 2020b. Analyzing and Improving the Image Quality of StyleGAN. In CVPR.

[8] Huikai Wu, Shuai Zheng, Junge Zhang, Kaiqi Huang . 2013v3 . GP-GAN: Towards Realistic High-Resolution Image Blending
[9] Qiantong Xu and Gao Huang and Yang Yuan and Chuan Guo and Yu Sun and Felix Wu and Kilian Weinberger,An empirical study on evaluation metrics of generative adversarial networks,2018