XIAMEN UNIVERSITY MALAYSIA

| | | |
|---|---|---|
| Course Code | : | AIT301 |
| Course Name | : | Advanced Machine Learning |
| Lecturer | : | Zhang Yingqian |
| Academic Session | : | 2022/09 |
| Assessment Title | : | Project(Final Assessment) |
| Submission Due Date | : | 2022/12/15 |

Prepared by :

| Student ID | Student Name |
|---|---|
| AIT2009357 | Bi Xiaoyang |
| AIT2009362 | He Enhao |
| AIT2009376 | Wang Qipeng |
| | |
| | |
| | |
| | |
| | |
| | |
| | |

Date Received :

Feedback from Lecturer:

Marking Scheme for report

| **Items** | **Total** | **Marks** |
|---|---|---|
| Language writing | 4 | |

| | | |
|---|---|---|
| Technique content with method | 4 | |
| Format | 2 | |
| | | |
| | | |
| | | |

Mark:

## Own Work Declaration

I/We hereby understand my/our work would be checked for plagiarism or other misconduct, and the softcopy would be saved for future comparison(s).

I/We hereby confirm that all the references or sources of citations have been correctly listed or presented and I/we clearly understand the serious consequence caused by any intentional or unintentional misconduct.

This work is not made on any work of other students (past or present), and it has not been submitted to any other courses or institutions before.

Signature: [illegible] [illegible] Wang Qipeng

Date: 2022/12/21

# An Attempt On Rebuilding And Improving the Original CycleGAN

***Prepared by***

Bi Xiaoyang

He Enhao

Wang Qipeng

# Content

## Abstract

Style transfer is a heated computer vision topic that has been broadly mentioned in both academic research and practical applications, for example, the style transfer technique is behind the various filters in daily used image processing tools. However, it had been a major drawback in this field that its training set required strictly paired images, until the introduction of CycleGAN, a combination of style transfer technique and Generating Adversarial Networks (GAN), which has also been one of the most popular technologies in machine learning since its advent. CycleGAN is an image style transfer technique proposed by Jun-Yan Zhu et al. at the University of California, Berkeley. The paper proposed a way that can be performed without the strict pairs of the source image and style image so that a wider range of dataset options can be incorporated into this field. The generator in the CycleGAN network consists of an encoder, converter, and decoder, which can play the role of preserving the original image features and converting images. In our research, we found that there is still certain scope for improvement in the original model, so we conducted a further study on the basis of rebuilding the original CycleGAN. We studied the CycleGAN framework, which is an unsupervised learning style transfer model based on generative adversarial networks, and proposed an improved nerual network model by modifying the adversarial loss, and compared it with the original CycleGAN network. The results showed that the modified CycleGAN can obtain more realistic images with better visual effects than original CycleGAN after training in fewer rounds.

**Keywords: CycleGAN, style transfer, loss function, deep learning**

## 1. Introduction

If Vincent Van Gogh, the Dutch painter who lived in the 19th century, had traveled through time and space to modern times, what would the beautiful campus architecture of Xiamen University look like in his Expressionist style of painting? If Qi Baishi, the master of Chinese painting, had been able to visit Xiamen University Malaysia, how would he have used his brush to paint the grass and trees on our campus?

With the development of artificial intelligence technology, deep learning has been applied more and more widely, among which the combination of computer graphics and deep learning has produced many excellent algorithms and has been extensively used in many fields such as image style transfer [1] and image segmentation [2] subsequently. Among them, GAN (Generative Adversarial Network) has brought a lot of surprises to image translation tasks and has received a lot of attention from researchers. For example, the "edges2cats" project, in which the edges of an object are outlined and the neural network automatically completes the image information to generate the corresponding image, has been applied to this project, and four types of GANs, including DCGAN and WGAN, have been used to generate cat images with good results.

Researches on GAN networks are in a spurt, and the following figure shows the number of papers named GAN from 2014 to 2018.

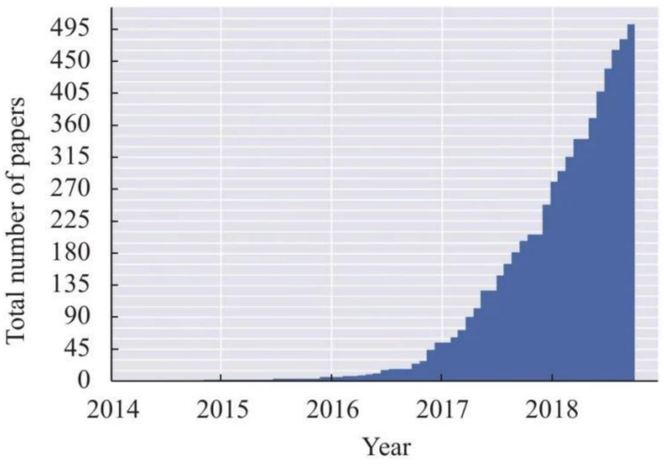


***Fig 1. Number of papers that includes GAN from 2014 to 2018[30]***

CycleGANs are special variants of traditional GANs. It can also create new data samples, but this process is achieved by transforming the input samples instead of generating them starting from completely random noise. In other words, they learn to transform data from two data sources. These data can be selected by the scientist or developer who provided the dataset for this algorithm. CycleGAN works impressively well, allowing different painterly styles of painting to be reduced to photographs, turning summer into winter, horses into zebras, and oranges into apples. This is the main reason why we have chosen cycleGAN as our baseline. But although cycleGAN has achieved significant success in many of these tasks, even with a 100% success rate for specific targets, we believe it still falls short.

We focused on discovering the reasons behind such excellent results of cycleGAN and the possible improvement, and we found the following problems with cycleGAN.

a. Failure occurs when **geometric transformations** are involved, such as dog to cat or vice versa.

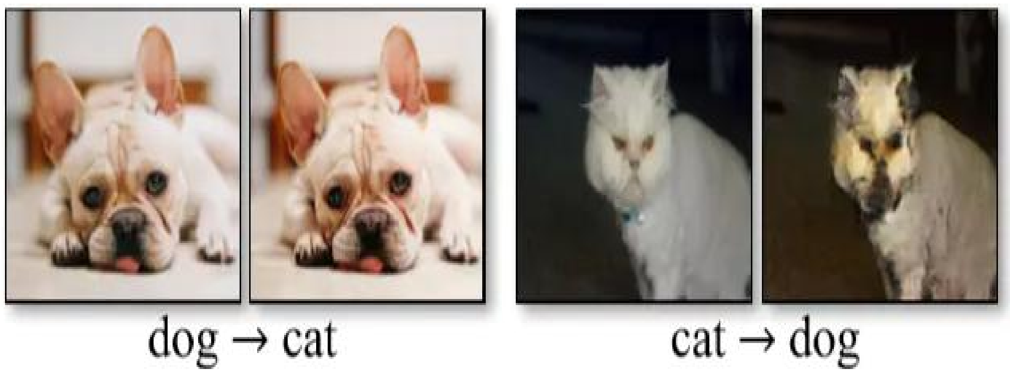


***Fig 2. Failure in transfer between dog and cat[28]***

b. Failure caused by **training and distribution features** may occur. For example, when training the wild horse and zebra conversion, the network does not take into account a circumstance that a person is riding on the horse. We supposed it is probably because the network only extracts the contour part.

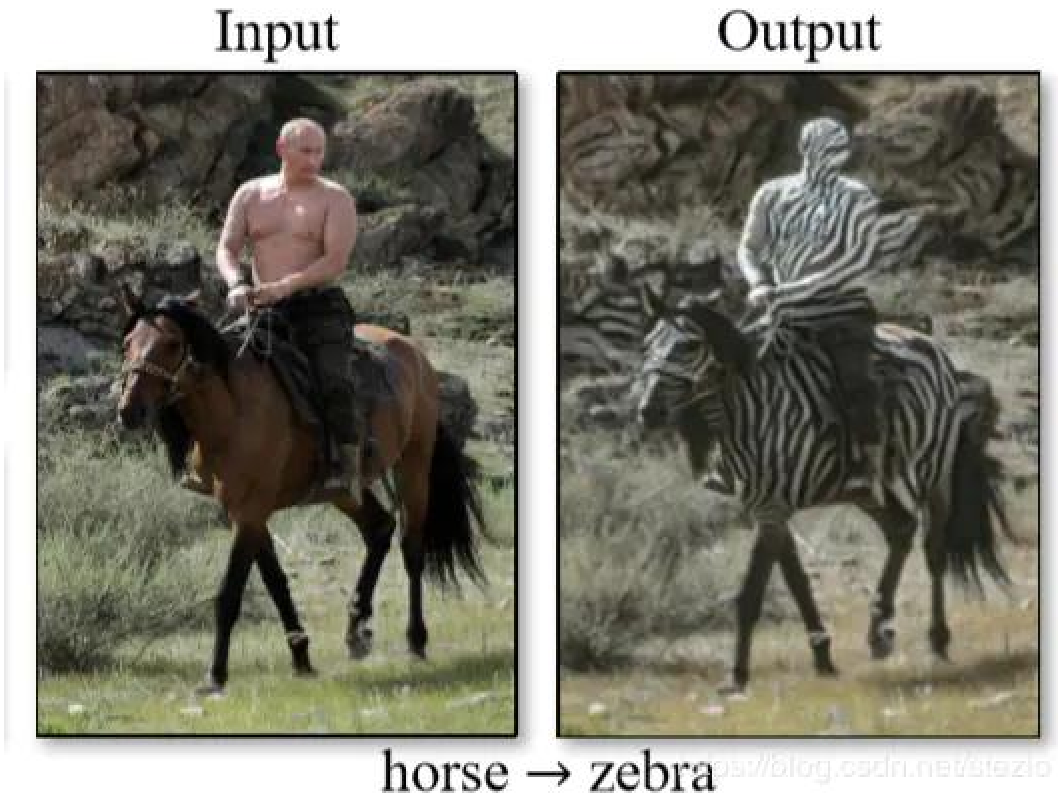


***Fig 3. Failure in transfer between horse and zebra[29]***

c. There is still a **performance gap between** results gained from the datasets with **paired style** and source image groups and those with **unpaired datasets**, although the authors proposed that some weak semantic supervision needs to be incorporated to solve this problem.

We suggested the following possibilities based on the literature review and group discussions.

**a. The effect of cycle consistency loss is too strong.**

We can imagine that if the input image and the output image are exactly the same, the cycle consistency loss will be minimal. Nevertheless, since we want to translate a dog into a cat, it is not valid to make them exactly the same. That is to say, part of the information of the input image has to be sacrificed, which requires the cycle consistency loss to increase. Therefore, too much effect of cycle consistency loss will guide the network to produce the same output as the input image.

**b. Local characteristics of dogs and cats are very similar.**

The discriminators ($D_X$, $D_Y$) of cycleGAN are trained by the mechanism of patchGAN[3]. In discriminating whether the input image is a generator-generated image or a source image, instead of the whole image, the local part of the image (patch) is used for discrimination. Thus, due to the many similarities between the local features of cats and dogs, the model will make many mistakes in this case.

For the two potential causes above that may cause unsuccessful style transfer of cycleGAN, we suggested the following idea as a possible solution to these problems: Substituting a more powerful adversarial loss function for the original one in cycleGAN.

Our **contribution** is summarized as follows.

**[1]** We first compared the quality of generated images using different objective functions in WGAN-GP, WGAN, LSGAN and the original GAN. It was found that WGAN-GP can stabilize the training process and generate more authentic images.

**[2]** We chose WGAN-GP's objective function as the replacement for the adversarial loss function of cycleGAN. The results showed that the modified cycleGAN yields more authentic images with better visual effects.

The rest of the report will be organized as follows.

Section **2** provides a comprehensive review of the outstanding past work in the area involved in our project, and the advanced methods we have applied.

Section **3** illustrates the research methodology and proposes problems and our solutions to them. In this section, we also introduce the approaches of GAN and style transfer respectively.

Section **4** describes in detail the derivation and proof in mathematical terms of GAN and CycleGAN, with an overall description of our code and the algorithms applied.

Section **5** presents the experimental results and related images and data.

Section **6** concludes the whole project.

## 2. Literature review

In this section we surveyed the literature on GAN and style transfer, which allowed us to learn about the forefront research in these fields and to apply them in our subsequent project.

### 2.1. GAN

GAN (Generative Adversarial Networks) was proposed by Ian J. Goodfellow et al.[] in their paper "Generative Adversarial Nets" presented at the NIPS conference in October 2014. Since then, due to the high feasibility and excellent performance of this framework, GAN-related researches have grown rapidly and more and more kinds of GAN networks are utilized in various applications. However, with the application of GAN networks, a series of problems such as difficult convergence, unstable training, and uncontrollable models have emerged, prompting a large number of scholars to research on such problems. For example, **DCGAN** (Deep Convolutional GAN) proposed by Radford et al.[4] combines deep convolutional neural networks with GAN, which has been proven to be applicable in multiple novel tasks. ARTIN et al. [5] proposed **WGAN** (Wasserstein GAN) for the problem of unstable training of GAN networks, which uses Earth-Mover distance instead of JS scatter as the objective function of discriminator D, effectively improving the training stability. Gulrajani et al. [6] came up with **WGAN-GP**, which introduced gradient penalty, to address the problem that WGAN sometimes fails to converge to the optimum. Mao et al. [7] used the least squares loss function(**LSGAN**) instead of the loss function of the original GAN in order to alleviate the problems of vanishing gradients and unstable training processes in GANs. **BEGAN** (Boundary Equilibrium GAN), proposed by Berthelot et al.[8], adopted an equilibrium concept to balance between the generator and the discriminator. The authors used a novel way to formulate the loss function as well. Moreover, Zhu et al [9] proposed a cycle-consistent generative adversarial network (**CycleGAN**), which can perform image-to-image translation based on unpaired datasets. Our project is based on the existing research above.

### 2.2. Style Transfer

The main objective of style transfer is separating the style and content of one image (style image) and then transfer the style towards another image (content image), while preserving as much of the semantic content of the other image as possible. The main challenges in style migration are:

***(1) how to define and separate style and content and***

***(2) how to change the style of an image whilst also preserving its content.***

Before the development of the style migration method based on neural networks, many image processing methods were commonly used for style migration, including Stroke-based rendering (SBR)[11], Image analogy[12], Image filtering[27], Texture synthesis[13] etc. Stroke-based rendering is a method that renders a picture with a specific style by adding virtual strokes to a digital canvas. Most application scenarios are limited to oil painting, watercolour, sketching and so on, which are not flexible enough. Image analogy aims to learn a mapping between a pair of source and target images in order to obtain stylized images in a supervised learning manner. The image analogy training set consists of pairs of uncorrected source images and corresponding styled programmed images. The analogical method is moderately effective, but the difficulty is that paired training data is hard to acquire in practice. As to the image filtering method, it uses some combination of image filters (such as bilateral and Gaussian filters, etc.) to render a given image considering that image style migration is actually a process of image simplification and abstraction. Texture synthesis is the process of adding similar textures to the source texture image. The "texture" here refers to the visual patterns that are repeatedly present in images. The texture synthesis based algorithms utilize only low-level image features, which limits their performance.

However, the field advanced significantly in 2015 with the emergence of "Neural Style Transfer", a method for image style transfer based on deep learning models. This can be seen as a new beginning of neural networks in the field of style migration. Gatys et al. first proposed this approach in two papers. The first paper proposed a method for modelling textures using deep learning, and an important assumption for texture generation mentioned above is that textures can be described by local statistical models. The manual modelling approach is too complex, whereas the deep learning model can better automate the feature extraction and texture modelling steps. The second paper proposes using VGG-19[14] (a convolutional neural network [15]) to extract features from images and then using the Gram matrix to compute correlations between the features of content and style images and to model the textures. This method can effectively transfer texture from a style image to a content image. In practice, nonetheless, the network must be trained for each migration, which consumes too much time and prohibits real-time migration. Additionally, style migration on photos is prone to distortion. After that, with various GAN models (e.g., Ming-Yu Liu et al.'s Coupled Generative Adversarial Networks [16] Bingzhe Wu et al.'s Perceptual Generative Adversarial Network [26]), techniques of image-to-image style migration has been greatly improved. Unlike the previous methods that only learn the style of one image, GAN methods can learn the style in a group of images extensively and better capture the similar features. Because of the extremely rapid development of GAN in the field of computer graphics, the methods of image style migration have reached maturity, and the DualGAN[17], CycleGAN[18], Pix2Pix[19] and SSIM-GAN[21] algorithms are widely used.

Before CycleGAN, there were already some style transfer models which all require the training data to be paired. In real life, however, it is quite difficult to find images that appear in pairs in two domains. That’s why CycleGAN was invented, which only requires data from two domains without one-to-one correspondence, which makes CycleGAN more widely used. It uses a convolutional deep learning model to extract features from images, and uses adversarial loss to train generators and discriminators with cycle consistency loss to prevent contradiction between them. The main difference between this method and common neural network approaches is that we achieve overall style conversion between two image sets rather than two individual images.

## 3. Research Methodology

### 3.1. GAN

GAN consists of two competing neural networks: one is the Generator (G), which generates sample images, and the other is the Discriminator (D), which distinguishes real samples from generated samples. The network consists of a generator G and a discriminator D, where the G network transforms a noise in the form of a vector into a sample that is highly similar to the real data, and the D network determines whether the input data comes from the real sample or from the fake data generated by the G network. There is a dynamic process between them. The optimization problem of GAN network is actually a minimal-maximization problem, i.e., first fix the generator G and optimize the discriminator D to maximize the discriminative accuracy of D; then fix the discriminator D and optimize the generator G to minimize the discriminative accuracy. When the two are equal, the global optimum is reached. The entire network is shown below:

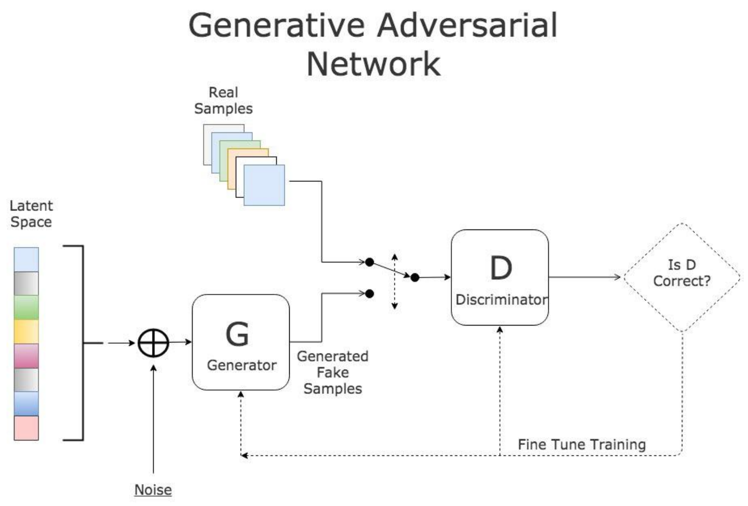


*Fig 4. Structure of Generative Adversarial Network(GAN)[25]*

## 3.2. Style Transfer and CycleGAN

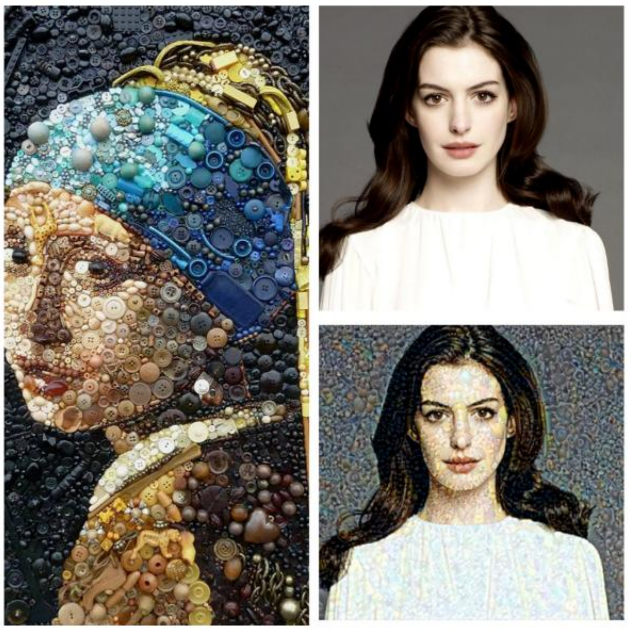


***Fig 5. Style migration of a single image using an ordinary neural network, comparing the content image and the result image after style migration (left is the style image, top right is the content image, bottom right is the result image)***

Style transfer is a task in the field of image-to-image translation. The inputs are two images as shown in Figure 1, the left one is called "style" and the top right one is called "content". What we want to do is generating a new image (bottom right) that has both the style of the left image and the content of the top right one.

CycleGAN was invented to realize style transfer for a series of photos. Its purpose is to learn the mapping between two image sets, domain X and domain Y. CycleGAN is made up of two mappings (generators) and two discriminators, namely mapping G: X->Y, mapping F: Y->X, discriminator $D_x$, and discriminator $D_Y$. G's objective is to ensure that mapped X (G(X)) match the distribution of Y ($G(X) \sim p_{data}(Y)$). $D_Y$'s purpose is to distinguish between y and G(x). F and $D_X$ work in the same way, but in the opposite direction. CycleGAN's objective function has two parts: (1) adversarial loss for updating generators and discriminators, and (2) cycle consistency loss for preventing conflicts between the mappings G and F. The left half of figure 5 shows that the input image x is sent to the generator G to produce a target domain image $G(x)$. And then the generated $G(x)$ is sent to the generator F to produce $F(G(x))$. Likewise, $G(F(y))$ is generated by two generators from y. Then we calculate cycle consistency loss by equation (2), which aims at preventing contradiction of generator

G and F. Consequently, our objective is to min-max the total loss function (equation (3)) that combines adversarial loss and cycle consistency loss together in order to obtain the optimal G and F.

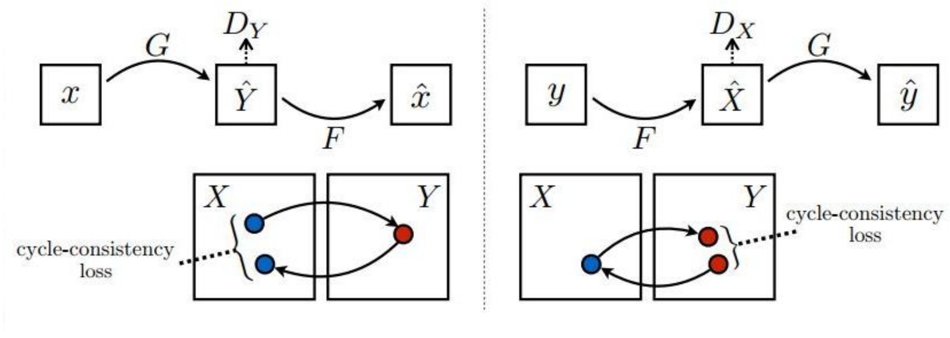


***Fig 6: The principle of how CycleGAN work[24]***

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

$$L_{\text{cyc}}(G, D_Y, X, Y) = E_{x\sim p_{data}(x)}[\| \mathrm{F}(G(x)) - x \|]_1 + E_{y\sim p_{data}(y)}[\| \mathrm{G}(F(y)) - y \|]_1 \ldots\ldots\ldots\ \ldots..(2)$$

$$L_{\text{total}}(G, \mathrm{F}, D_x, D_y) = L_{\text{GAN}}(G, D_Y, X, Y) + L_{GAN}(F, D_x, Y, X) + \lambda L_{cyc}(G, F) \ldots\ldots\ldots\ldots\ldots\ldots\ldots(3)$$

## 3.3. Datasets

For GAN, we us e d the anime face dataset from Professor Hung-yi Lee's 2022 machine learning tutorial with 71,314 images from the Crypko website.

For CycleGAN, we use the Monet2Photo. The Monet2Photo dataset is made up of 1193 Monet paintings and 7038 natural photos, which are divided into train and test groups. This dataset was collected from the official directory of CycleGAN Datasets at UC Berkeley.

## 3.4. Code and Algorithm Implementation

For GAN’s implementation in our code, we can roughly divide its training into three steps.

**Step 1**: Initialize the parameters of the generator and discriminator.

**Step 2**: Train the discriminator first: fix the generator to update the discriminator, and the discriminator gives a low score to the generated image of the generator and a high score to the real labeled image. The training set is labeled with true label (1) and the fake image generated by the generator is labeled with false label (0) and sent to the discriminator together to train the discriminator. The loss is calculated so that the discriminator gets close to true (1) for the real data (Training Set) input and close to false (0) for the fake image generated by the Generator. **In this process, only the parameters of Discriminator are updated, and the parameters of Generator are not updated.**

**Step 3**: Then the generator is trained: the discriminator is fixed and the generator's parameters are updated, and the generator tries to 'fool' the discriminator by generating an image from the input randomly sampled vector so that it can be scored high. The Gaussian distributed noise z (Random noise) is fed to the Generator, and then the Fake image generated by the Generator is fed to the Discriminator with a true label (1). The loss is calculated so that the discriminator discriminates the fake image generated by the Generator to be close to true (1). **In this process, only the parameters of the Generator are updated, not the parameters of the Discriminator.**

As for the implementation of rebuilding CycleGAN, similar to other deep neural networks, CycleGAN has been trained using Stochastic gradient descent (SGD) and Back Propagation (BP) algorithms. Due to the limitation of GPU resources, during the training process, the dataset is divided into batches of 4 images in size, and the average of the gradient of the loss function of all samples in a small batch is calculated each time, and then the loss function is changed in the direction of negative gradient, and the change is propagated forward by the back propagation algorithm to propagate the error (gradient). And, as a GAN, the structure and implementation of the generator and discriminator determine the performance of the network.The generator of CycleGAN is divided into three parts: "downsampling block", "residual block", "upsampling". The "downsampling block" has 3 "convolutional blocks", each of which contains a 2D convolutional layer, an Instance Normalization layer and a ReLU. The first Instance Normalization layer is followed by the ReLU activation function, and these are connected using the residuals. Then there are 3 "upsampling blocks", each containing a 2D transposed convolutional layer, an Instance Normalization and a ReLU activation function. The last layer is a 2D convolutional layer, using tanh as the activation function, which generates an image with the shape (256, 256, 3). And, its input and output sizes are identical, both are (256, 256, 3). The Discriminator is a network with 16x16 image chunks as input, which is responsible for discriminating whether the input 16x16 image region is a real sample or not. In this network, there are four layers, each consisting of a 4x4 convolution kernel with a step size of 2. The activation function is a Leaky ReLU with a slope of 0.2. After the last layer, a convolution is used to output a 1-dimensional output to characterize the probability that the input image belongs to the real sample. the number of filters in the four convolutional layers is 64, 128, 256 and 512, respectively.

## 4. Experiment

### 4.1. Experimental Design

#### 4.1.1. Experimental environment and setup

We used an Intel Core i7 octa-core CPU, NVIDIA 3060 GPU, 64GB RAM in our experiments. The code runs on Windows, CUDA 11.3 development environment, pyTorch deep learning framework based on python programming language. In our experiments, we applied the Monet-style CycleGAN art style transfer and trained three other style migration models using three datasets we made (Qi Baishi flower and bird painting set, Picasso surrealist painting set, and Picasso portrait painting set). For the consideration of concise typography, we only show some of the results in this section, more results can be found in our appendix or in the jupyter notebook file.

#### 4.1.2. Assumption

As CycleGAN was first proposed in 2017, more advanced GAN models have emerged since then and shown their outstanding performance.

A reasonable hypothesis is thus proposed:

***Taking the improvements of these state-of-the-art models, in our context, the loss function applied to CycleGAN can improve performance.***

### 4.1.3. Controlled Trial

Our aim in setting up the control experiment is to explore to get the best loss function to replace the CycleGAN original one with it. Therefore, our idea is to first train the four models WGAN, WGAN_GP, LSGAN and DCGAN with the same architecture but different loss functions with the same dataset and parameters, and select the best performing one by comparing IS and FID.

Next, we will replace the original one of CycleGAN with the optimal loss function we obtained and train the new generated model and the unmodified CycleGAN with the same dataset under the same conditions. After full training, we put the generated results to the judgment of people both with and without art history backgrounds, by comparing their chances of being fooled (the higher the better) we draw the conclusion of which model performs better in image transfer.

### 4.1.4. Parameters used in the experiment

*The same parameters in 4 Sets (WGAN, WGAN_GP, LSGAN, DCGAN):*

epochs=50, critic =5(G updates once critic updates 5 times, but in DCGAN critic = 1 because it can only be set as so), batch_size = 64, lr = 1e-4,

Input shape: (3, 100),

Output shape: (3, 3, 64, 64)

*Specific parameter only needed in WGAN:*

clip_value = 0.01,

opt_D = torch.optim.RMSprop(D.parameters(), lr=lr)，

opt_G = torch.optim.RMSprop(G.parameters(), lr=lr)

*Parameters used in other three(betas needed)：*

opt_D = torch.optim.Adam(D.parameters(), lr=lr, betas=(0.5, 0.999))，

opt_G = torch.optim.Adam(G.parameters(), lr=lr, betas=(0.5, 0.999))

## 4.2. Validation/Verification

In evaluating the quality of images generated by GAN, we have adopted three criteria: IS, standard deviation and FID. In the following we will focus on IS and FID because they are commonly applied to evaluate GAN.

### 4.2.1. Inception Score(IS)

The **Inception Score**[20][21] is arguably the most used metric in the literature. It uses an image classification model M and an Inception network (Szegedy et al., 2016) pre-trained on ImageNet (Deng et al., 2009), thus calculating.

The principle of IS evaluation for images generated by GAN is that, assuming x is the data feature and y the label, an image is fed into the neural network (Inception V3) and the output layer outputs the probability that it belongs to each category, i.e., the probability p(y|x) of determining which label is given the data feature, and the distribution of each label (category) is p(y).

$$\mathrm{IS(G)} = \exp(\mathrm{E}_{\mathrm{x \sim P_g}} D_{KL}(p(y \mid x) \,\|\, p(y)))$$

***Fig 7. Mathematical definition of IS[]***

$$\begin{aligned}
\mathrm{IS(G)} &= \exp(\mathrm{E}_{\mathrm{x \sim P_g}} D_{KL}(p(y \mid x) \,\|\, p(y))) \\
&= \sum_{\mathrm{x}} \mathrm{p(x)D_{KL}}(p(y \mid x) \,\|\, p(y)) \\
&= \sum_{\mathrm{x}} \mathrm{p(x)} \sum_{\mathrm{i}} p(y = i \mid x) \ln(\frac{p(y = i \mid x)}{p(y = i)}) \\
&= \sum_{\mathrm{x}} \sum_{\mathrm{i}} \mathrm{p(x, y = i)} \ln(\frac{p(x, y = i)}{p(x) p(y = i)}) \\
&= \mathrm{I(y;x)} \\
&= \mathrm{H(y) - H(y \mid x)}
\end{aligned}$$

***Fig 8. derive the meaning of the above equation[]***

And IS is judged by the following criteria.

**[1]** For a single generated image, the probability distribution of the Inception output should be as small as possible, the smaller it is the more likely the generated image belongs to a certain class and the higher the quality of the image.

**[2]** For a batch of images generated by a generator, the average probability distribution entropy value of Inception output should be as large as possible, which represents the diversity of the generator generation.

### 4.2.2. Fréchet Inception Distance (FID)

Fréchet Inception Distance (FID)[22], an improved version of IS, is a metric introduced by Heusel et al. (2017) and used to evaluate GANs. For an appropriate feature function φ (by default, the convolutional feature of the Inception network), FID models φ(P_r) and φ(P_g) as Gaussian random variables with sample means μ_r and μ_g and sample covariances C_r and C_g. The Fréchet distance (or equivalently, the Wasserstein-2 distance) of the two Gaussian distributions can be computed by the following equation:

$$\mathrm{FID}\,(\mathrm{P}_r, \mathrm{P}_g) = \| \mu_r - \mu_g \| + \mathrm{Tr}\,(C_r + \mathrm{C}_g - 2(C_r C_g)^{1/2})$$

The FID calculates the distance between two distributions, the smaller the distance means the generated distribution is closer to the real distribution, so the smaller the FID is, the better GAN performs.

Compared with IS, FID has better robustness to noise. Since FID only uses Inception V3 as a feature extractor and does not rely on it to determine the specific category of images, there is no need to worry about the difference between the training data of Inception V3 and the training data of the generative model. Also, since the distance between the distribution of the generated data and the real data is measured directly, there is no need to worry about mode collapse in the form of producing only identical images within each category.

***Limitations***

Although the method of FID is much improved compared to IS, the problem of overfitting on large-scale datasets like ImageNet is still not solved. Besides, FID is based on feature extraction, that is, it relies on the occurrence or non-occurrence of certain features, and thus cannot describe the spatial relationships of these features. For example, if a GAN is used to generate a human face, FID may also consider it a better generated result if the mouth grows above the eyes.

Overall, the same structure of Inception V3 is used to extract features, so FID is often used in GAN papers as a complement to IS, especially in diversity and mode collapse problems, FID has better evaluation performance, but also has the same drawbacks as IS, such as not suitable for use on datasets with large internal differences, unable to distinguish overfitting, etc.

## 4.3. Result Representation

### 4.3.1. Comparison of loss

| | **DCGAN** | **WGAN** | **WGAN-GP** | **LSGAN** |
|---|---|---|---|---|
| **IS** | 1.9692 | 2.1163 | 2.2166 | 1.9764 |
| **Standard Deviation** | 0.2913 | 0.3010 | 0.2132 | 0.3069 |
| **FID** | 217.4077(10th epoch) | 228.1258(43rd epoch) | 250.7911(49th epoch) | 259.9822(39th epoch) |

### 4.3.2. Image demonstration

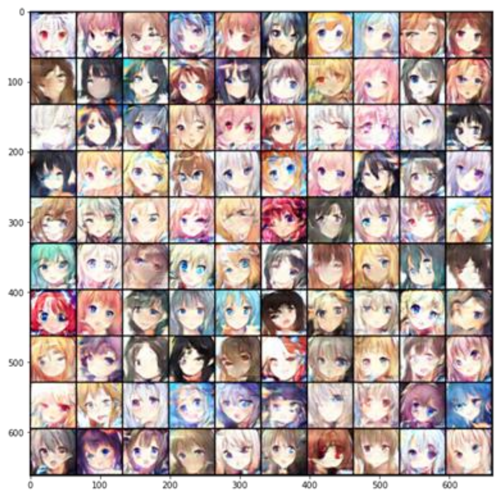


*Fig 9. Image DCGAN generated*

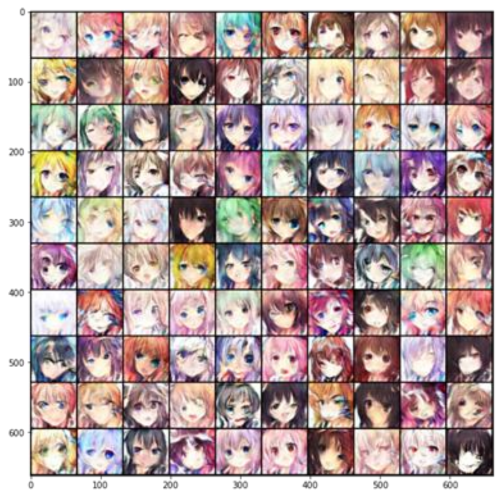


*Fig 10. Image LSGAN generated*

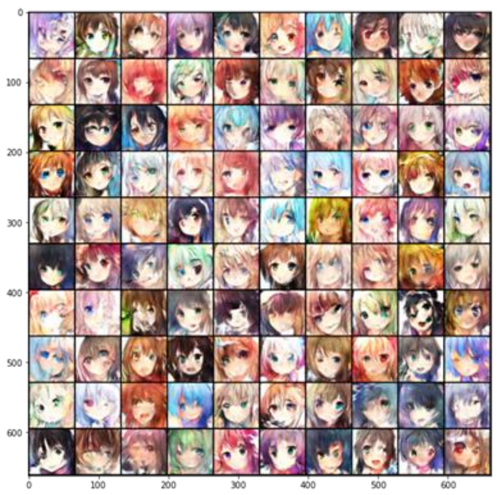


***Fig 11. Image WGAN  generated***

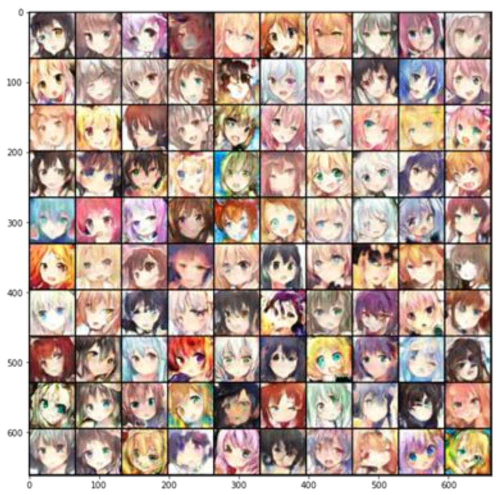


***Fig 12. Image WGAN-GP generated***

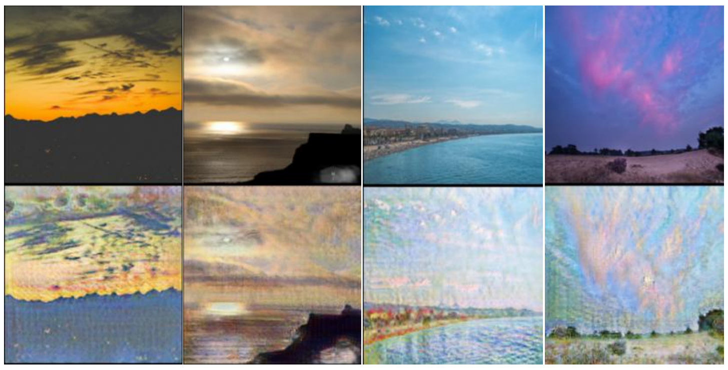


***Fig 13. Generated image of sky scene by CycleGAN(original loss function)***

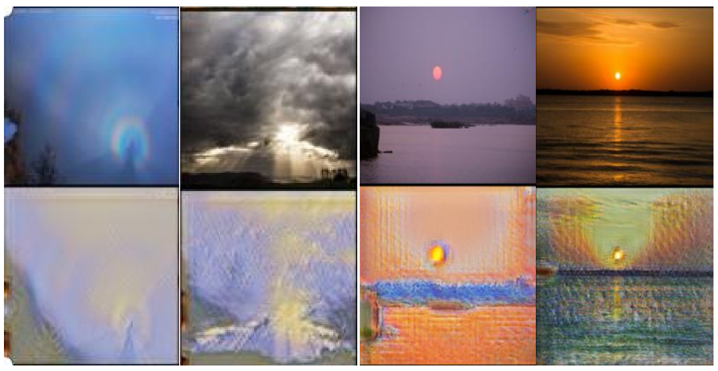


***Fig14. Generated image of sky scene by CycleGAN(WGAN loss)***

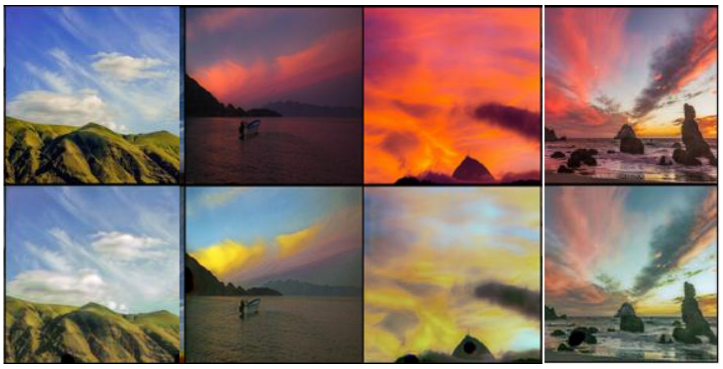


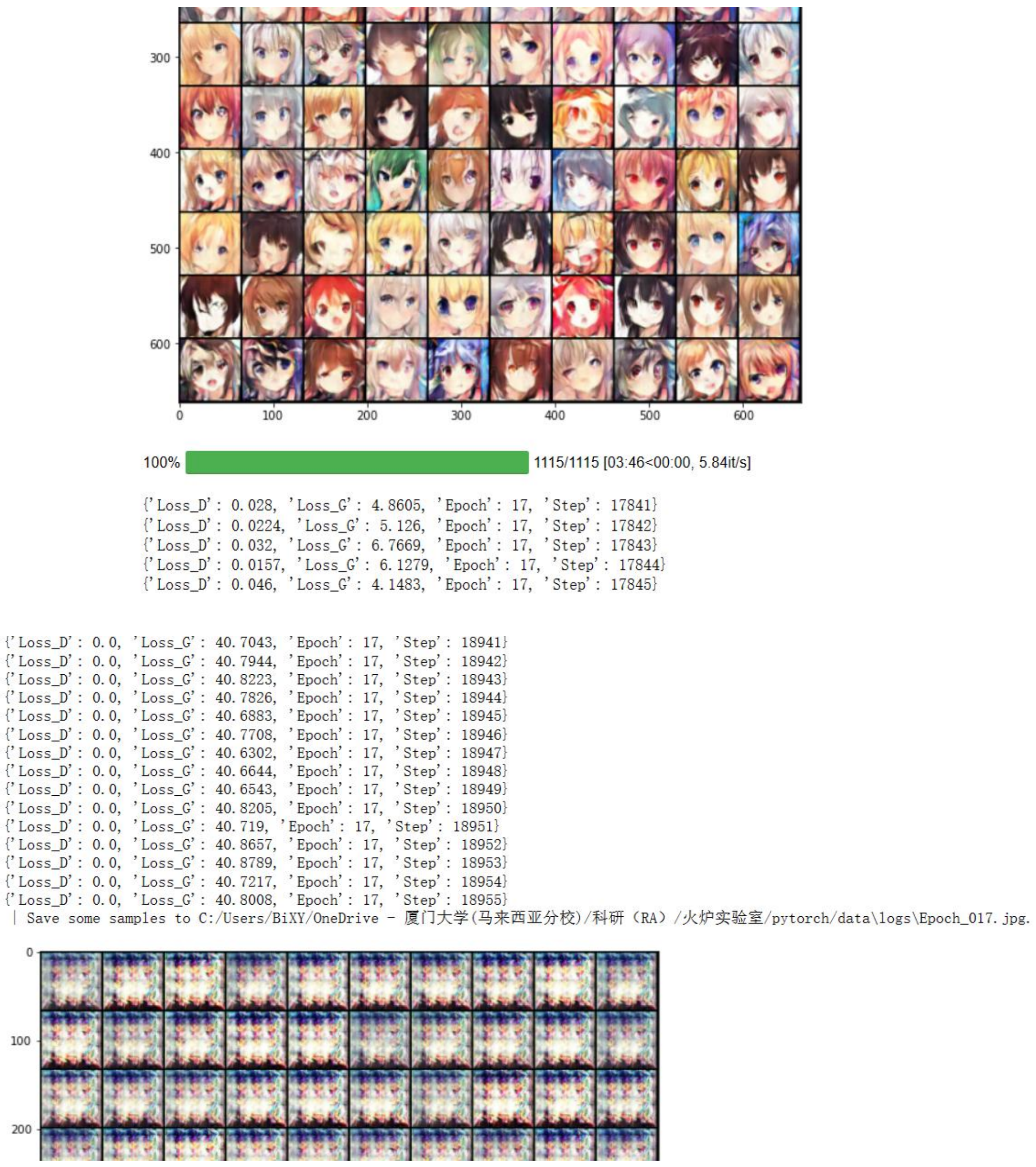
*Fig 15. Generated image of sky scene by CycleGAN(WGAN-GP loss)*

## 4.4. Analysis and Discussion

There is a crucial limitation in our project restricting us to furthur study. Since we are unable to afford high-performance GPU, we have to run all the programs on our own laptops, which is tremendously time-consuming. Therefore, due to the limited time for this project, we only ran 50 epochs for each GAN models to compare their results, which are relatively insufficient compared to the real research. Besides, in the training process of CycleGAN, we had merely run 35000-40000 steps to acquire the final results.

In spite of these objective limitations, we still gained some meaningful conclusions from the outputs. Comparing DCGAN and the other three GANs, we found that although DCGAN could achieve lower FID in the first 15 epochs than that in WGAN and WGAN-GP, it would encounter a severe problem of mode collapse after 15-20 epochs. We suppose that this sudden worsening is caused by the unstability of DCGAN, and thus it can’t be trained by too many epochs. By contrast, the FID of WGAN and WGAN-GP has a stable decreasing trend in 50 epochs, which satisfies our assumption well. Furthermore, when we compared WGAN, WGAN-GP and LSGAN, we found that WGAN-GP held the highest IS, which means the generated images of WGAN-GP have maximal clarity and diversity. Therefore, although WGAN-GP output slightly larger FID than WGAN, we still selected WGAN-GP as our improvement tool utilized on CycleGAN.

The images generated by our enhanced CycleGAN show slight improvement compared with original one in transfering the style on sky landscape image. Due to the lack of large video memory, we are not capacity to run enough steps to obtain excellent outputs. Therefore, we just compare the results under 35000 steps training. Although all of generated images can be easily distinguished from the real image sets, we can still conclude that our model generates better outputs. When we show generated images from different CycleGAN to our friends, the majority of them said images generated by our models are a little more realistic than those from original CycleGAN. Besides, we also tried the loss function in WGAN to replace the adversarial loss in CycleGAN and generated some outputs. The outputs are of poor quality suffering from the loss of color diversity in original images, which is in line with our expectation.



```
100%  1115/1115 [03:46<00:00, 5.84it/s]

{'Loss_D': 0.028, 'Loss_G': 4.8605, 'Epoch': 17, 'Step': 17841}
{'Loss_D': 0.0224, 'Loss_G': 5.126, 'Epoch': 17, 'Step': 17842}
{'Loss_D': 0.032, 'Loss_G': 6.7669, 'Epoch': 17, 'Step': 17843}
{'Loss_D': 0.0157, 'Loss_G': 6.1279, 'Epoch': 17, 'Step': 17844}
{'Loss_D': 0.046, 'Loss_G': 4.1483, 'Epoch': 17, 'Step': 17845}

{'Loss_D': 0.0, 'Loss_G': 40.7043, 'Epoch': 17, 'Step': 18941}
{'Loss_D': 0.0, 'Loss_G': 40.7944, 'Epoch': 17, 'Step': 18942}
{'Loss_D': 0.0, 'Loss_G': 40.8223, 'Epoch': 17, 'Step': 18943}
{'Loss_D': 0.0, 'Loss_G': 40.7826, 'Epoch': 17, 'Step': 18944}
{'Loss_D': 0.0, 'Loss_G': 40.6883, 'Epoch': 17, 'Step': 18945}
{'Loss_D': 0.0, 'Loss_G': 40.7708, 'Epoch': 17, 'Step': 18946}
{'Loss_D': 0.0, 'Loss_G': 40.6302, 'Epoch': 17, 'Step': 18947}
{'Loss_D': 0.0, 'Loss_G': 40.6644, 'Epoch': 17, 'Step': 18948}
{'Loss_D': 0.0, 'Loss_G': 40.6543, 'Epoch': 17, 'Step': 18949}
{'Loss_D': 0.0, 'Loss_G': 40.8205, 'Epoch': 17, 'Step': 18950}
{'Loss_D': 0.0, 'Loss_G': 40.719, 'Epoch': 17, 'Step': 18951}
{'Loss_D': 0.0, 'Loss_G': 40.8657, 'Epoch': 17, 'Step': 18952}
{'Loss_D': 0.0, 'Loss_G': 40.8789, 'Epoch': 17, 'Step': 18953}
{'Loss_D': 0.0, 'Loss_G': 40.7217, 'Epoch': 17, 'Step': 18954}
{'Loss_D': 0.0, 'Loss_G': 40.8008, 'Epoch': 17, 'Step': 18955}
 | Save some samples to C:/Users/BiXY/OneDrive - 厦门大学(马来西亚分校)/科研 (RA) /火炉实验室/pytorch/data\logs\Epoch_017.jpg.
```

*Fig 16. Mode collapse after 15-20 epochs of DCGAN training*

## 5. Conclusion

In this project, we mainly focused on how to enhance the original CycleGAN. The approach we proposed was replacing the adversarial loss function with a more powerful one. In the experiments, we compared the generated images from four different GAN models and concluded that the best one was WGAN-GP. Consequently, we substituted its loss function for that in CycleGAN to obtain the improved CycleGAN. The experimental results showed that our model indeed outperformed the original CycleGAN to small extent.

## *Reference*

1. Gatys, L. A., Ecker, A. S., & Bethge, M. (2015). A neural algorithm of artistic style. *arXiv preprint arXiv:1508.06576*.

2. Cheng, H. D., Jiang, X. H., Sun, Y., & Wang, J. (2001). Color image segmentation: advances and prospects. *Pattern recognition*, *34*(12), 2259-2281.

3. Demir, U., & Unal, G. (2018). Patch-based image inpainting with generative adversarial networks. *arXiv preprint arXiv:1803.07422*.

4. Ian, G., Pouget-Abadie, J., Mirza, M., Xu, B., & Warde-Farley, D. (2014). Generative adversarial nets." In Advances in neural information processing systems.

5. Radford, A.; Metz, L.; Chintala, S. Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks. arXiv 2015, arXiv:1511.06434.

6. Arjovsky, M., Chintala, S., & Bottou, L. (2017, July). Wasserstein generative adversarial networks. In *International conference on machine learning* (pp. 214-223). PMLR.

7. Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., & Courville, A. C. (2017). Improved training of wasserstein gans. *Advances in neural information processing systems*, *30*.

8. Mao, X., Li, Q., Xie, H., Lau, R. Y., Wang, Z., & Paul Smolley, S. (2017). Least squares generative adversarial networks. In *Proceedings of the IEEE international conference on computer vision* (pp. 2794-2802).

9. Berthelot, D., Schumm, T., & Metz, L. (2017). Began: Boundary equilibrium generative adversarial networks. *arXiv preprint arXiv:1703.10717*.

10. Zhu, J. Y., Park, T., Isola, P., & Efros, A. A. (2017). Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision* (pp. 2223-2232).

11. Hertzmann, A. (2003, July). A survey of stroke-based rendering. Institute of Electrical and Electronics Engineers.

12. Hertzmann, A., Jacobs, C. E., Oliver, N., Curless, B., & Salesin, D. H. (2001, August). Image analogies. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques* (pp. 327-340).

13. Efros, A. A., & Leung, T. K. (1999, September). Texture synthesis by non-parametric sampling. In *Proceedings of the seventh IEEE international conference on computer vision* (Vol. 2, pp. 1033-1038). IEEE.

14. Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

15. Krizhevsky, A., Sutskever, I., Hinton, G. E., Pereira, F., Burges, C. J. C., Bottou, L., & Weinberger, K. Q. (2012). Advances in neural information processing systems.

16. Liu, M. Y., & Tuzel, O. (2016). Coupled generative adversarial networks. *Advances in neural information processing systems*, *29*.

17. Yi, Z., Zhang, H., Tan, P., & Gong, M. (2017). Dualgan: Unsupervised dual learning for image-to-image translation. In *Proceedings of the IEEE international conference on computer vision* (pp. 2849-2857).

18. Isola, P., Zhu, J. Y., Zhou, T., & Efros, A. A. (2017). Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 1125-1134).

19. Wang, Z., Bovik, A. C., Sheikh, H. R., & Simoncelli, E. P. (2004). Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, *13*(4), 600-612.

20. Barratt, S., & Sharma, R. (2018). A note on the inception score. *arXiv preprint arXiv:1801.01973*.

21. Xu, Q., Huang, G., Yuan, Y., Guo, C., Sun, Y., Wu, F., & Weinberger, K. (2018). An empirical study on evaluation metrics of generative adversarial networks. *arXiv preprint arXiv:1806.07755*.

22. Salimans, T., Goodfellow, I., Zaremba, W., Cheung, V., Radford, A., & Chen, X. (2016). Improved techniques for training gans. *Advances in neural information processing systems*, *29*.

23. ericxian1997/style-transfer-CycleGAN: Course project of Digital Image Processing. Image Style Transfer with CycleGAN. (github.com)

24. https://zhuanlan.zhihu.com/p/402819206

25. https://www.google.com/url?sa=i&url=https%3A%2F%2Fpaperswithcode.com%2Fmethod%2Fgan&psig=AOvVaw0rMsBgTKIcgPvnbBjSOzDB&ust=1671698241724000&source=images&cd=vfe&ved=0CBAQjRxqFwoTCIDXsLGnivwCFQAAAAAdAAAAABAZ

26. Li, J., Liang, X., Wei, Y., Xu, T., Feng, J., & Yan, S. (2017). Perceptual generative adversarial networks for small object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 1222-1230).

27. He, K., Sun, J., & Tang, X. (2012). Guided image filtering. *IEEE transactions on pattern analysis and machine intelligence*, *35*(6), 1397-1409.

28. http://static.extremevision.com.cn/donkey_4ed5e0c0-ec54-4614-907b-1d7e821dde78.jpg

29. https://arxiv.org/abs/1703.10593

30. https://mmbiz.qpic.cn/mmbiz_jpg/sKia1FKFiafgh1T3wDm40OaZVj3ohA9ibnbkmO8hibrSXbUD6WeyveibicI9r0cpZbhxQicgXa4u4BYibUwFZDBkPq7Miag/640?wx_fmt=jpeg&wxfrom=5&wx_lazy=1&wx_co=1

***Contribution of each team member***

In this section, we provide the work segregated among each team members.

| | | |
|---|---|---|
| **Bi Xiaoyang** | 1. Distributed tasks and design a schedule for other group members.<br>2. Responsible for experiment design and do coding work .<br>3. Developed a general framework of report. | 40% |
| **Wang Qipeng** | 1. Surveyed on previous related work and proposed a certain direction of improvement(but were dropped during further discussion)<br>2. Refined the framework of the report and responsible for writing it. | 30% |
| **He Enhao** | 1. Implemented visualization of FID and conducted part of experiments.<br>2. Wrote part of the report and did overall revision and language polishing. | 30% |

## Appendix

### 1. Source Code:

https://xmueducn-my.sharepoint.com/:f:/g/personal/ait2009357_xmu_edu_my/Eiueq2oSR3ZDl_yphGJ-XZoBzrE1sRjQq-137JTQyM5Ecg?e=cKBbUh

### 2. More Result Presentation:

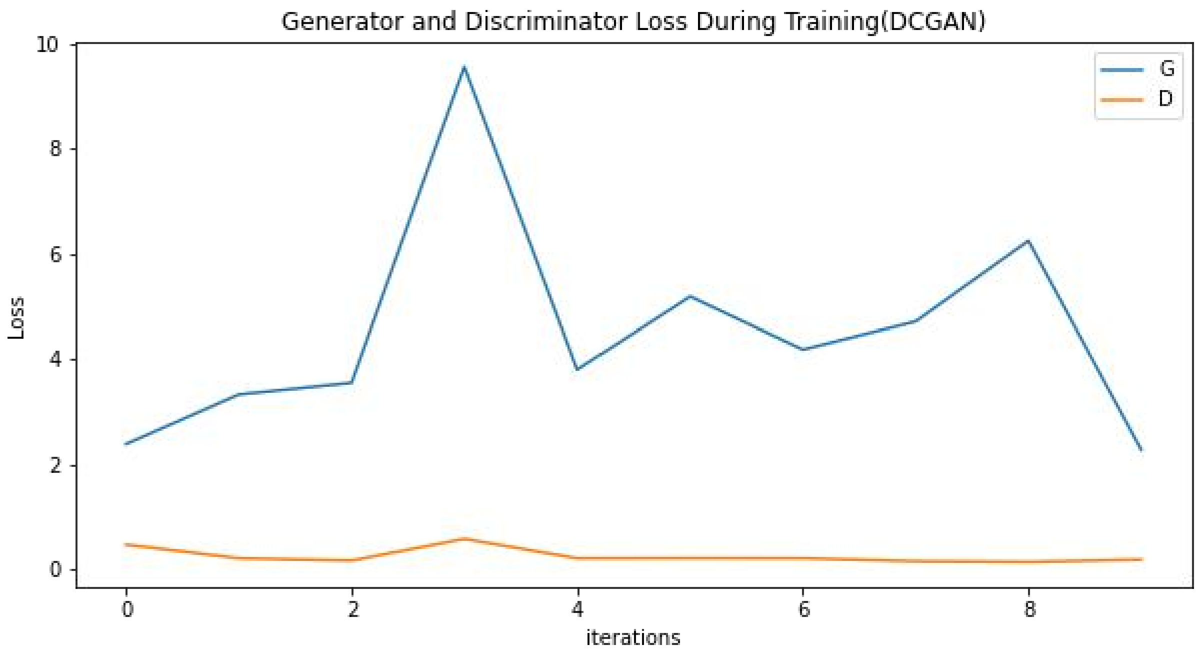


***Fig 17. change of  G's and D's loss of DCGAN***

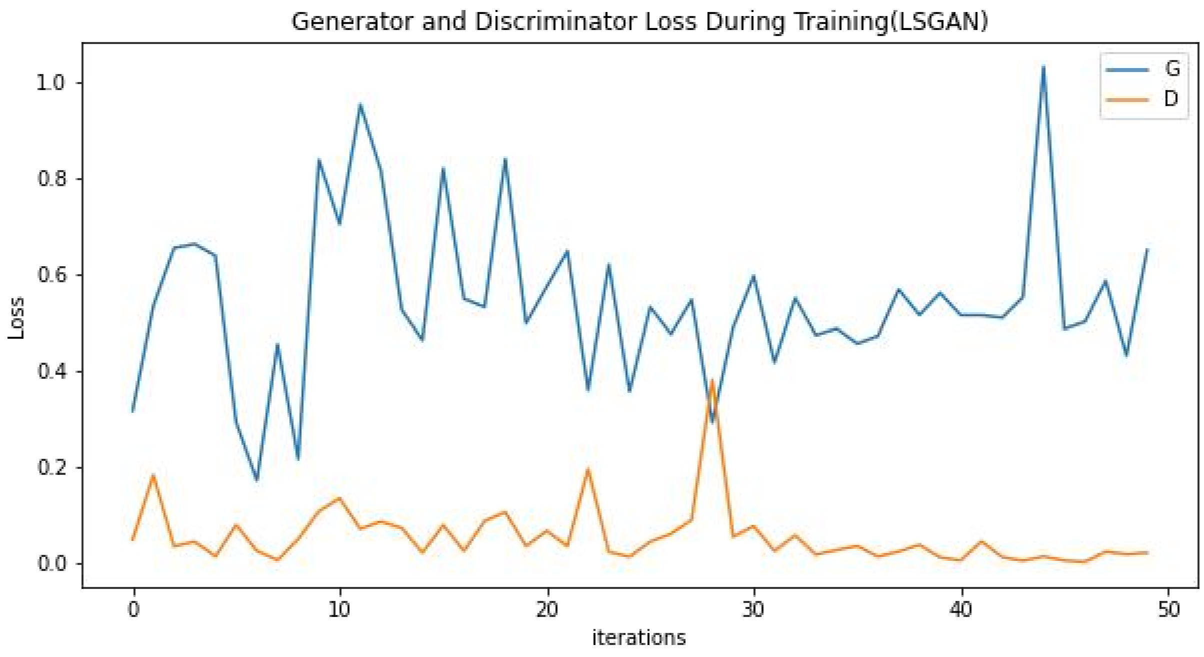


***Fig 18. change of  G's and D's loss of LSGAN***

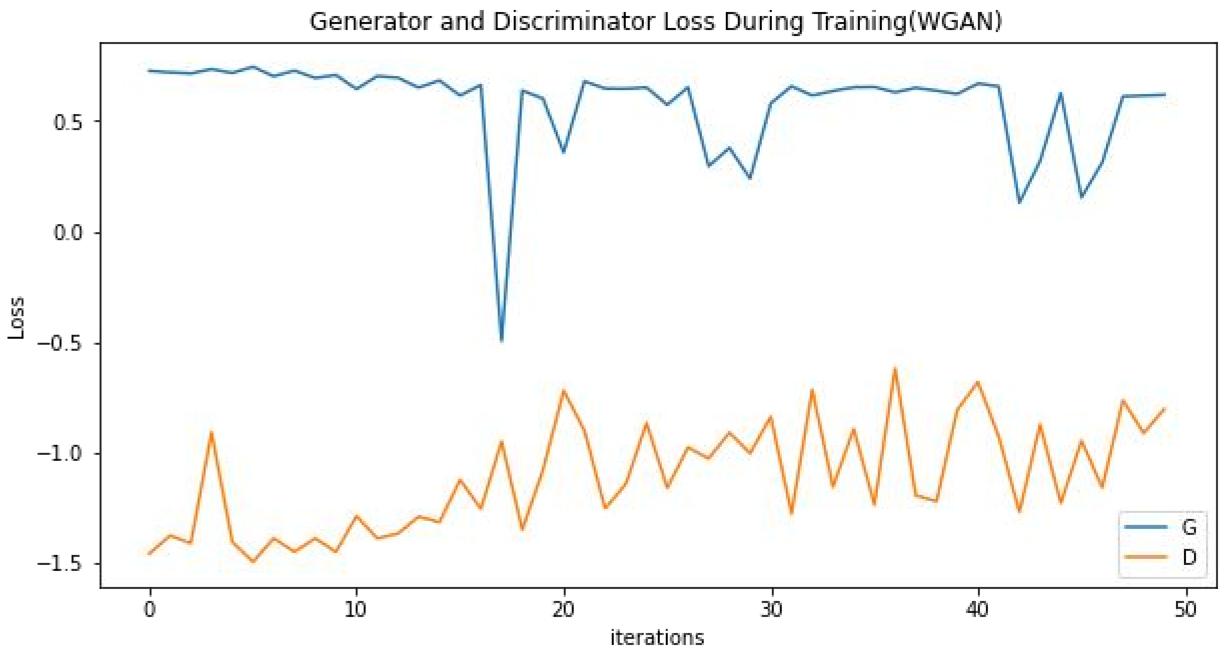


***Fig 19. change of G's and D's loss of WGAN***

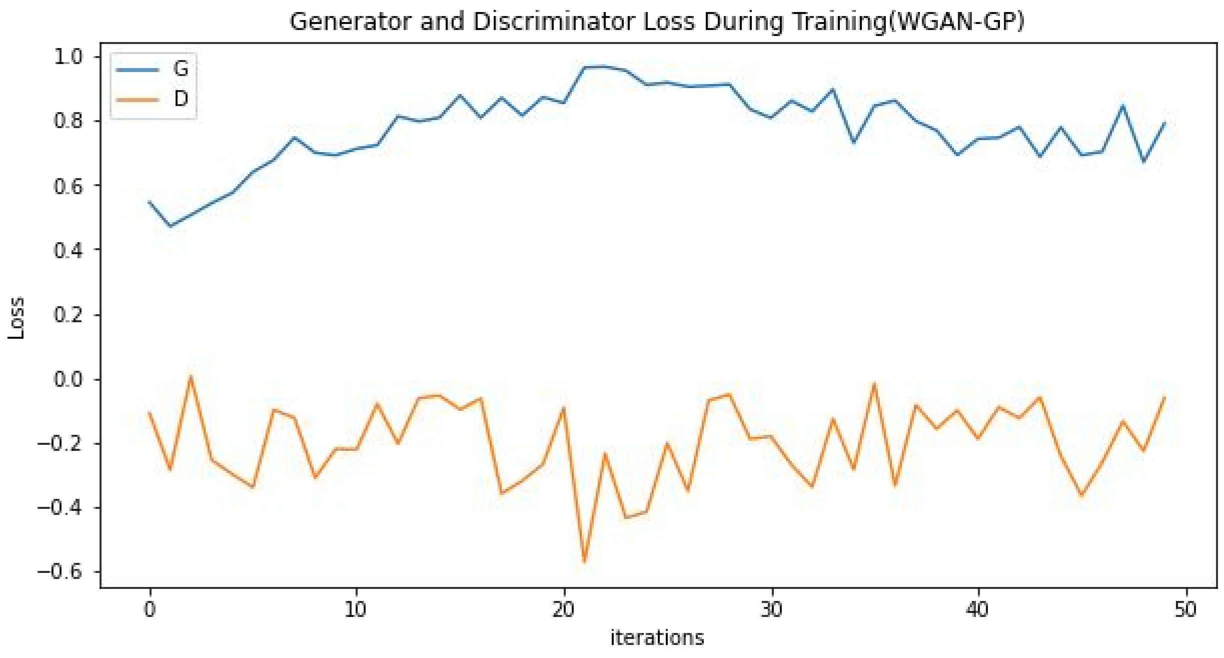


***Fig 20. change of G's and D's loss of WGAN-GP***

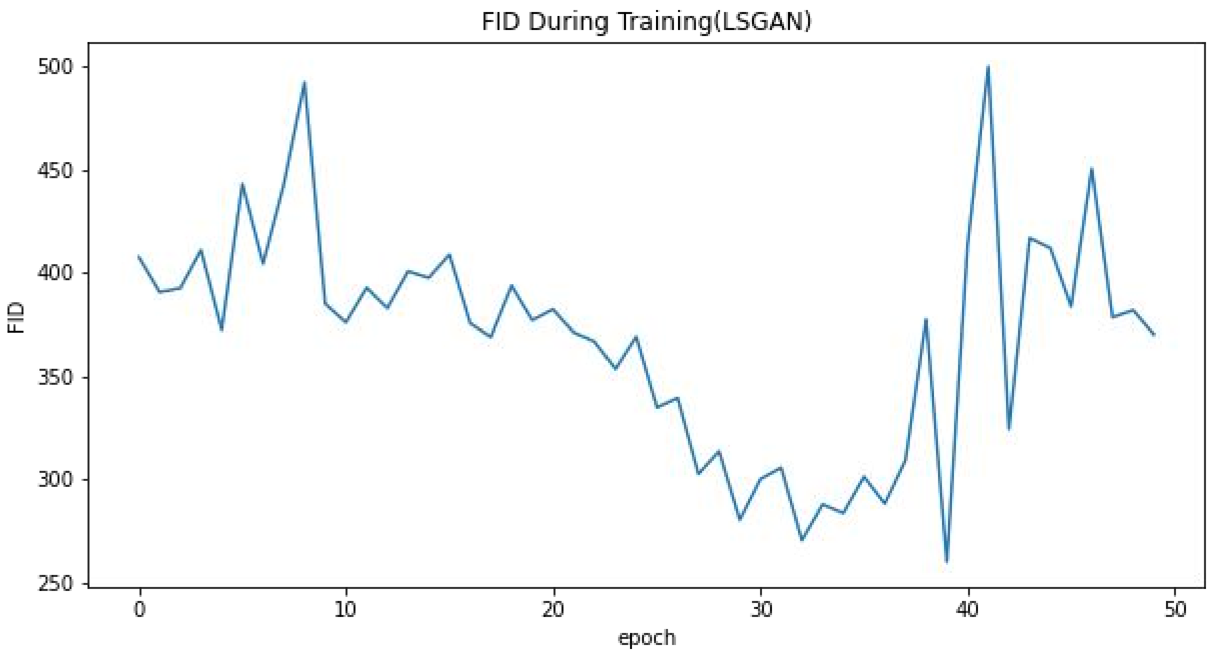


***Fig 21. change of  G's and D's FID of LSGAN***

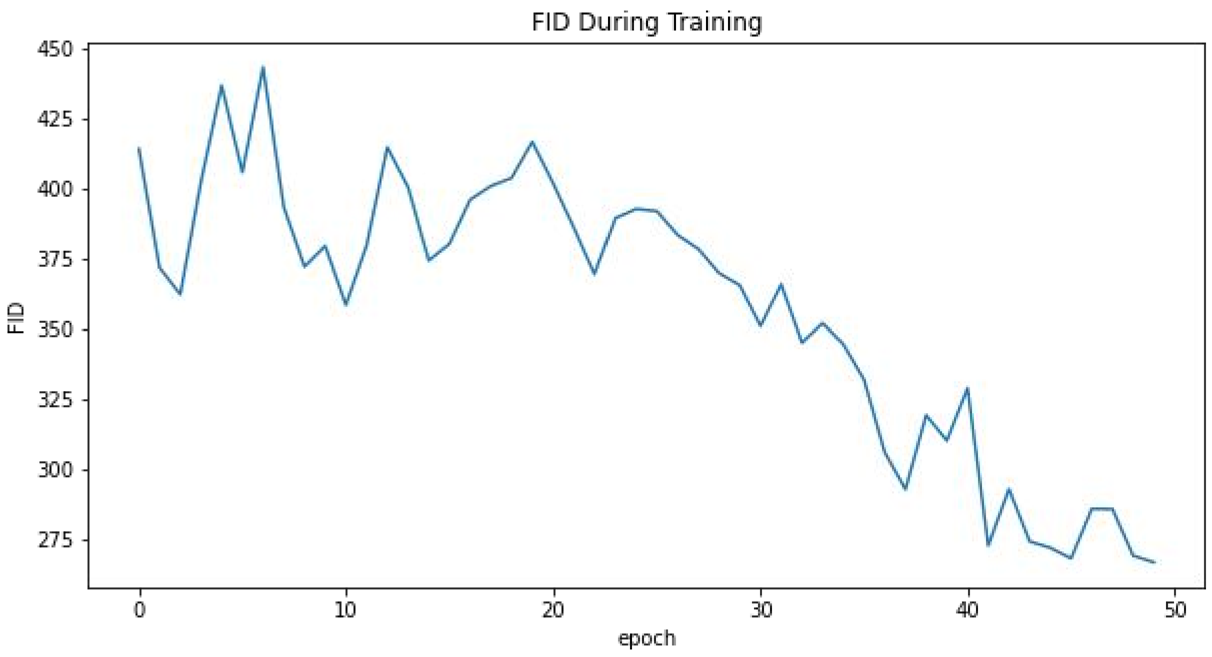


***Fig 22. change of  G's and D's FID of WGAN-GP***

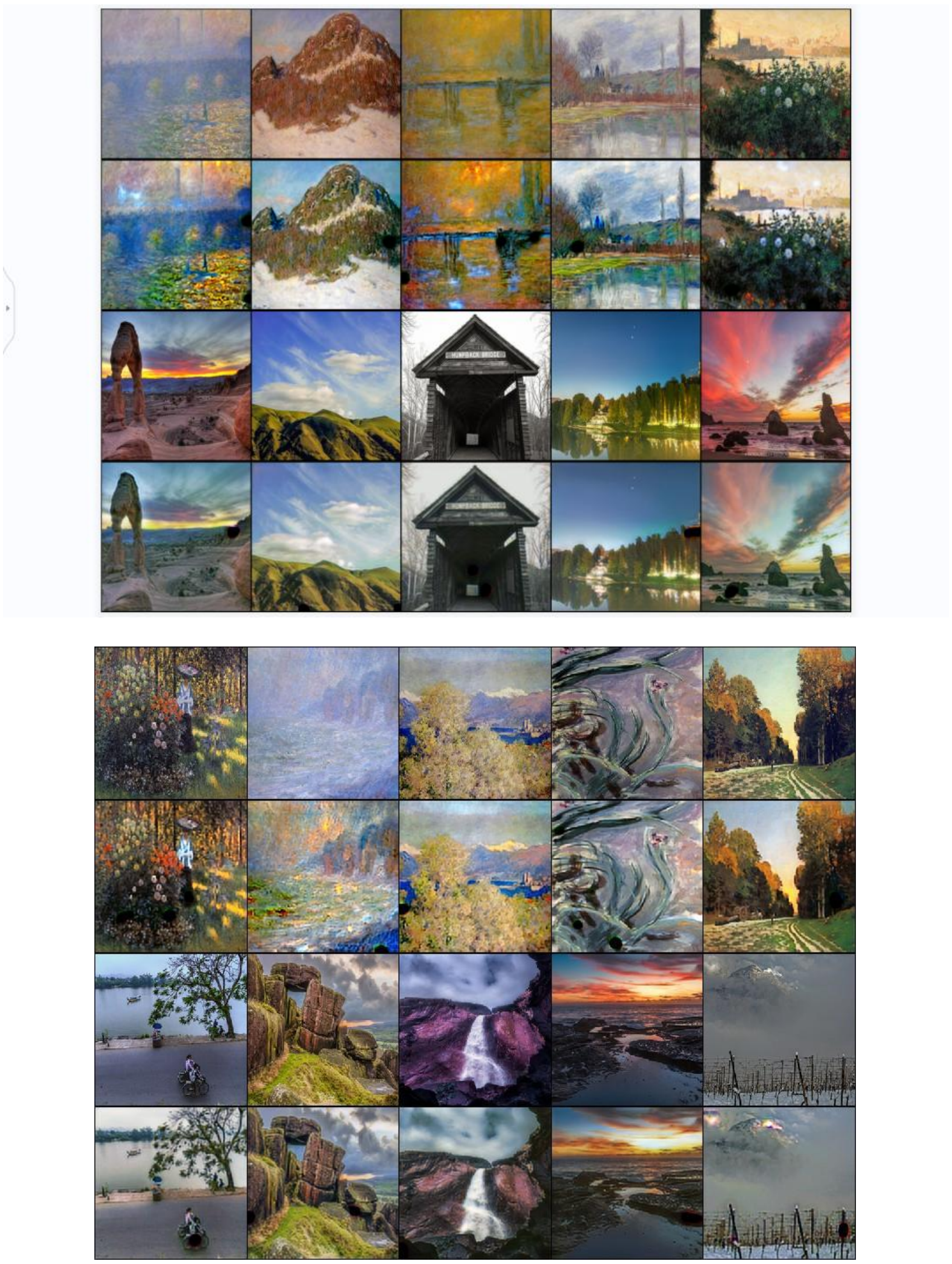


***Fig 23. The transfer results use our modified CycleGAN model after training nearly 35000-40000 epochs***

## MARKING RUBRICS

| Component Title | Final project Score and Descriptors | | | | | |
|---|---|---|---|---|---|---|
| **No.** | | **Poor** | **Average** | **Excellent** | **Total Marks** | **Marks** |
| Component 1: Project Development | | | | | | |
| | | **0 - 4** | **5 - 7** | **8 - 10** | **10** | |
| **1** | Code quality (CLO5) | Codes are poorly structured. Not fully functional. Not documented. | Codes are sufficiently documented and mostly functional. Satisfactorily structured | Codes are fully functional and well structured and documented | | |
| | | **0 - 4** | **5 - 7** | **8 - 10** | **10** | |
| **2** | Implementation of algorithm related to ML (CLO2) | Incomplete development of the method. | Complete development of method. | Completed and enhanced the developed methods | | |
| | | **0-2** | **3** | **4-5** | **5** | |
| **3** | Completeness and complexity of your project (CLO5) | The functionalities that are implemented are non-functional | All functionalities are implemented and functional | All the functionalities are fully implemented and functional Implemented extra functionalities. | | |
| | | | | **subtotal** | 25 | |
| Component 2: Project Report + Project Demonstration | | | | | | |
| | | **0 - 4** | **5 - 7** | **8 - 10** | **10** | |
| **1** | Project Report (CLO2) | Poor writing quality Poor or no formatting / presentation Lack of technical content | Satisfactory writing quality, grammar and flow. Substantial technical content with used method. | Good writing quality, grammar Well formatted and good presentation Demonstrate excellent technical content | | |
| | | **0 - 4** | **5 - 7** | **8 - 10** | **10** | |
| **2** | Presentation (CLO3) | Poor quality slides Poor time management | Satisfactory quality slides Speech that is | High quality slides Good time | | |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | Speech that is unclear | satisfactory and understandable | management Speech that is clear and impactful | | |
| | | **0-2** | **3** | **4-5** | **5** | |
| **3** | Demonstration (CLO3) | Poor demonstration that is unclear The developed application/ research is not functioning/implemented fully | Satisfactory demonstration The developed application/ research is partially functioning/ implemented fully | Good demonstration The developed application/ research is fully functioning and logical | | |
| | | | | **subtotal** | **25** | |
| | | | | **TOTAL** | **50** | |

Note to students: Please print out and attach this appendix together with the submission of coursework