國立陽明交通大學

多媒體工程研究所

碩士論文

Institute of Multimedia Engineering

National Yang Ming Chiao Tung University

Master Thesis

基於深度學習之特定畫家風格轉換模擬研究

The Synthesis of Specific Painter's Style Via Deep Learning

研究生：王俊皓 (Wang , Chun-Hao)

指導教授：施仁忠 (Shih, Zen-Chung)

中華民國一一○年十月

October 2021

基於深度學習之特定畫家風格轉換模擬研究

The Synthesis of Specific Painter's Style Via Deep Learning

研 究 生：王俊皓　　Student：Chun-Hao Wang

指導教授：施仁忠　　Advisor：Zen-Chung Shih

國立陽明交通大學

多媒體與工程研究所

碩士論文

A Thesis
Submitted to Institute of Multimedia Engineering
College of Computer Science

National Yang Ming Chiao Tung University

in partial Fulfillment of the Requirements

for the Degree of

Master of Science

October 2021
Taiwan, Republic of China

中華民國 一一〇年十月

國 立 陽 明 交 通 大 學

博碩士論文紙本暨電子檔著作權授權書

(提供授權人裝訂於紙本論文書名頁之次頁用)

本授權書所授權之學位論文，為本人於國立陽明交通大學多媒體工程研究所 _ _ _ _ _ _組，110 學年度第 一 學期取得碩士學位之論文。

論文題目：基於深度學習之特定畫家風格轉換模擬研究
指導教授：施仁忠

一、紙本論文授權

紙本論文依著作權法第15條第2項第3款之規定辦理，「依學位授予法撰寫之碩士、博士論文，著作人已取得學位者...推定著作人同意公開發表其著作」。

二、論文電子檔授權

本人授權將本著作以非專屬、無償授權國立陽明交通大學、台灣聯合大學系統圖書館及國家圖書館。

論文全文上載網路公開之範圍及時間：	
中英文摘要	■ 立即公開
本校及台灣聯合大學系統區域網路	■ 立即公開
校外網際網路及國家圖書館	■ 立即公開

說明: 基於推動「資源共享、互惠合作」之理念與回饋社會及學術研究之目的，得不限地域、時間與次數，以紙本、光碟或數位化等各種方式收錄、重製與利用；於著作權法合理使用範圍內，讀者得進行線上檢索、閱覽、下載或列印。

授權人：王俊皓 （親筆簽名）

中華民國 110 年 10 月 15 日

國立陽明交通大學碩士學位論文審定同意書

多媒體工程研究所　王俊皓 君

所提之論文

題目：基於深度學習之特定畫家風格轉換模擬研究

The synthesis of specific painter’ s style via deep learning

經學位考試委員會審查通過，特此證明。

學位考試委員會（簽名）

口試委員：

魏德樂（召集人）　蔡倩莊

論文已完成修改

指導教授 _____（簽名）

所　　長 _____（簽名）

中華民國 110 年 10 月 04 日

基於深度學習之特定畫家風格轉換模擬研究

研 究 生：王俊皓　　指導教授：施仁忠

國立陽明交通大學多媒體與工程研究所

摘　要

近年來，風格轉換用深度學習的方法常使用風格圖片當作輸入，這些方法會造成輸出結果過度仰賴風格圖片的既有形式。我們為了解決這問題，提出了一個基於深度學習的特定畫家風格轉換方法。首先我們設計一個有兩個串流的生成器(generator)，一個用來訓練生成具有風格的圖片，一個用來輔助使生成的圖片保有輸入圖片的線條和構造。此外，我們用預訓練的技術來讓整個網路架構更穩定產生想要的結果。我們的方法可以成功地模擬出三個特定畫家的風格。

關鍵字:深度學習、風格轉換

The Synthesis of Specific Painter's Style Via Deep learning

Student：Chun-Hao Wang Advisor：Prof. Zen-Chung Shih

Institute of Multimedia Engineering

National Yang Ming Chiao Tung University

ABSTRACT

Recently, current style transfer methods usually rely on the reference image as input. However, these methods will cause overdependence on the style of reference image. To solve this problem, we proposed a method based on deep learning to simulate the specific painter's style. We design a two-stream generator which would maintain the structure of input image and handle the style of painters. The main stream is to generate stylish image and compute adversarial loss. The sub-stream is to maintain the structure of content image and compute auto-encoder reconstruction loss. To stabilize the training, we also use pre-training in our algorithm. Our method successfully simulates three styles of impressionist painters.

Keywords: Deep Learning, Style Transfer,

Contents

List of Figures

List of Tables

Chapter 1

Introduction

For thousands of years, people have been attracted by the art of painting with many appealing masterpieces. Timeless in their beauty, these paintings have transcended time and artistic concepts to create history, e.g., Monet's "Impressive Sunrise". In the past, redrawing an image in a specific painter's style needs a well-trained artist and huge amount of time. At end of the 20th century, not only the artists but many computer science researchers [4][6][12][13] have attracted by these masterpieces and redrawing issues. They use computer techniques exploring how to turn image into synthetic artistic painting rapidly and effectively.

Recently, along with the development of convolution neural networks (CNNs), Gatys et al. [3] first used a CNN to produce famous painting styles on photorealistic images. Benefiting from convolution neural networks' ability on feature extraction of images, Gatys et al. generated very impressive visual results at the time. This seminal work of Gatys et al. has attracted attention from academia researchers. Inspired by Gatys et al. , many researches have been achieved with learning-based stylization. Although these learning-based stylization methods can preserve content well and match the overall style of the reference style images, they will also destroy the local texture that the famous artistic painting has, resulting unpleasing artistic stylizations. To address this issue, we propose a GAN-based model, which focuses on specific painter's style. Besides, our method will generate images to match the style of scenes that the

famous painter did not draw. We will discuss our method in more detail in Chapter 3 and show our results in Chapter 4.

In summary, our main contributions are as follows:

- We propose a learning-based method that effectively learns styles from famous artist's paintings and preserves texture details and colors.
- We introduced an auto-encoder reconstruction loss to maintain the structure of input image, such as boundaries of objects and texture of region. We also propose the novel layers to handle the style for each painter.
- We provide qualitative comparisons with previous style transfer methods. We also provide analysis between our results and famous paintings on artistic stylization.

Chapter 2

Related Work

In this chapter, we briefly review related researches of image style transfer. This topic has been studied for more than two decades. First, we introduce some traditional style transfer methods and its limitations. Since our framework is based on the neural network, we first discuss the convolutional neural network, which used for feature extraction of images. Then, we will talk about the generative adversarial network that our architecture is based on it.

2.1 Traditional Style Transfer Approaches

Before the increasing popularity of learning-based stylization, image style transfer has expanded into computer graphics under the label of non-photorealistic rendering (NPR). There are many non-photorealistic rendering methods proposed to mimic specific artistic styles [13]. Such as region-based rendering [12][6], they identify segments of content images in the image plane and filling them by using algorithms to render in artistic styles. However, the problem of region-based rendering is that only one algorithm is not able to simulate an arbitrary style.

Thus, Hertzman et al. [9] proposed a new framework, called “image analogies”. This framework has two stages, as shown in Figure 2.1. In the first stage, a pair of images aims to learn a mapping filter from this two example training pairs. Then, in the second stage, the mapping filter is applied to new target image to produce the result. This method can create result

effectively and be used for arbitrary styles. However, pairs of mapping images will not always exist in practice.

Based on the above observations, these traditional style transfer methods have some limitations: the styles variety and constraints of input data. Therefore, we need algorithms to overcome these problems. This makes the field of Neural Style Transfer appear [2][4][12].

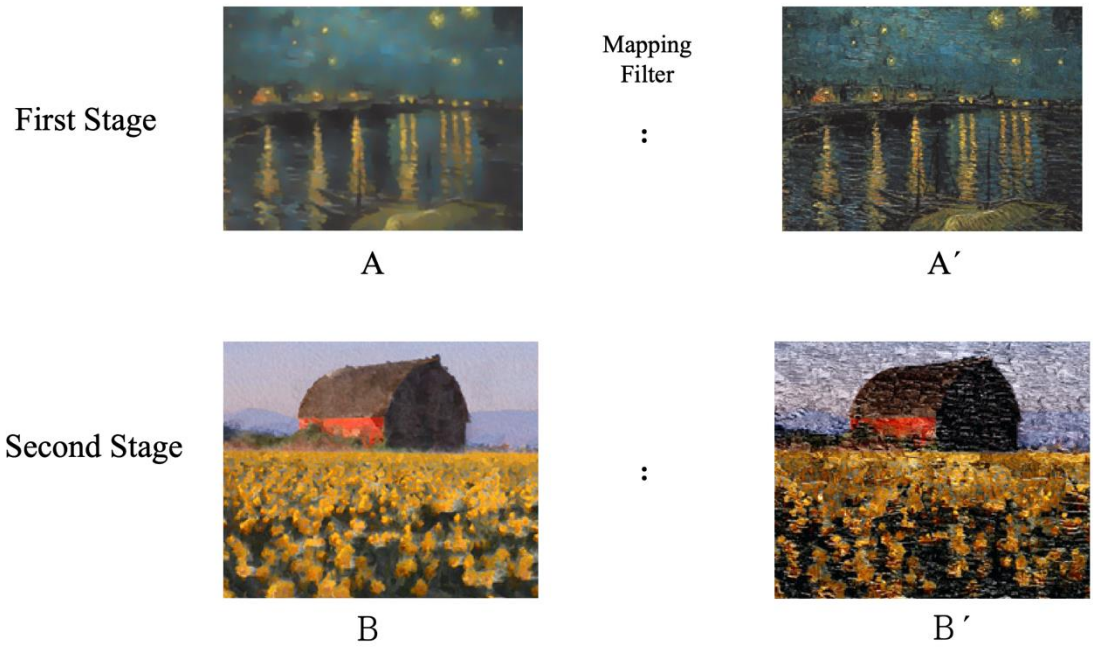

Figure 2.1 The goal of image analogies is to compute a new image B′ that relates to B. Besides A′ is also related to A. In this algorithm, A, B, A′ are inputs and B′ is the output. From Hertzman et al. [9].

2.2 Neural Style Transfer Methods

Traditional style transfer approach uses low-level image features which just finds edges or lines in an image and not capture image structures well, such as making boundaries of objects clear. Since Gatys et al. [4] first proposed CNN feature of deep networks which can represent image styles using the Gram matrix [3], it uses high-level algorithms concerning the interpretation or classification of the whole scene. However, the Gram-based style representation is a method to capture the global information and discard spatial correlation, producing unwanted artifacts for modelling regular textures.

Due to the limitations of Gram-based method, Liu et al. [14] proposed a method which focused on salient regions with the region loss in style transfer. This region loss which includes the cam loss and category loss is calculated from a localization network. By adding this region loss, Liu et al. provides more attractive visual effects since the silent regions are preserved. Cheng et t al. [2] used two subnetworks to get the depth and edge information to reconstruct the structure of content images. As a result, this network will solve the issue that some style textures tend to distribute over the stylized outputs and destroy the structure of content images. Liu et al. [12] generates a new artwork of image with saliency map which fuses the real-world image and artistic image. This output not only contain real-world background regions but also highlight the stylized salient regions.

In the above of methods, this structure-preserving style transfer methods demonstrate impressive results. However, these methods just choose a style image to refer. The outputs will be affected by the style image. This will have a very large impact on the result of different scenes that original painters will draw practically.

2.3 Generative Adversarial Network

Goodfellow et al [7] proposed a novel framework called Generative Adversarial Network (GAN), as shown in Figure 2.2. This framework simultaneously trains two models: the generator captures data distribution and produce data to deceive discriminator, and the discriminator estimates the probability of real images rather than generator. By training the discriminator and the generator iteratively, the generator can generate images to be more realistic.

To achieve arbitrary style transfer, there are many GAN-based methods holistically adjusted the features of content image to match the features of style image. Inspired by Generative Adversarial Network, Isola et al. [10] investigated conditional-GAN [5] as an effective solution to image-to-image translate problem. This method can achieve the transformation between two domains. Furthermore, Zhu et al. [14] proposed a CycleGAN to learn the mapping between two unpaired images. However, these methods can not handle well when using only one single model. Some structure regions with artistic details have artifacts and some smooth regions have unneglectable continuity.

Chen et al [1] proposed Gate-GAN to transfer different artistic styles in only one single model. To integrate multiple styles into a single network, they propose a gated transformer to distinguish specific artworks and their styles. They also use an auto-encoder reconstruction loss to stable GAN training. As shown in Figure 2.3, the encoder and the decoder are shared by every style. The gated transformer saves every style in each branch.

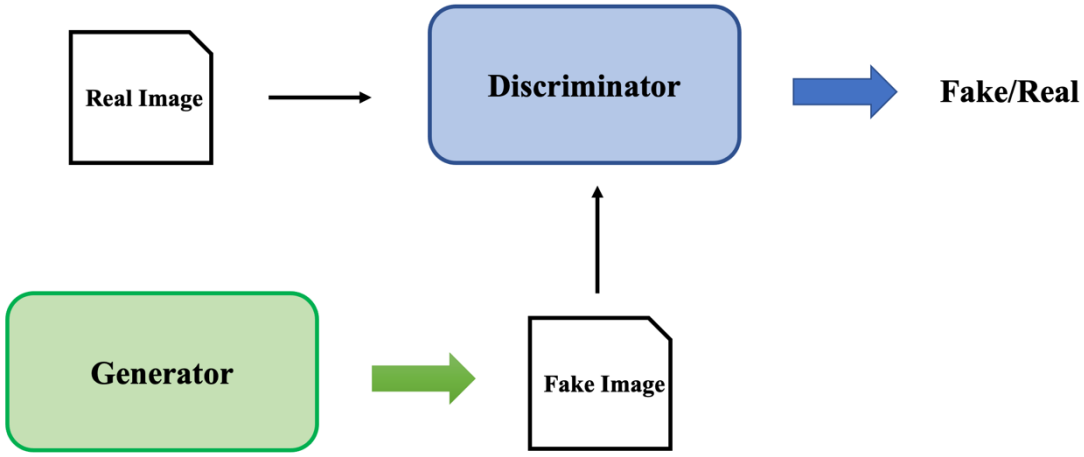

Figure 2.2 Generative Adversarial Network consists of two models (Discriminator and Generator) to compete with each other.

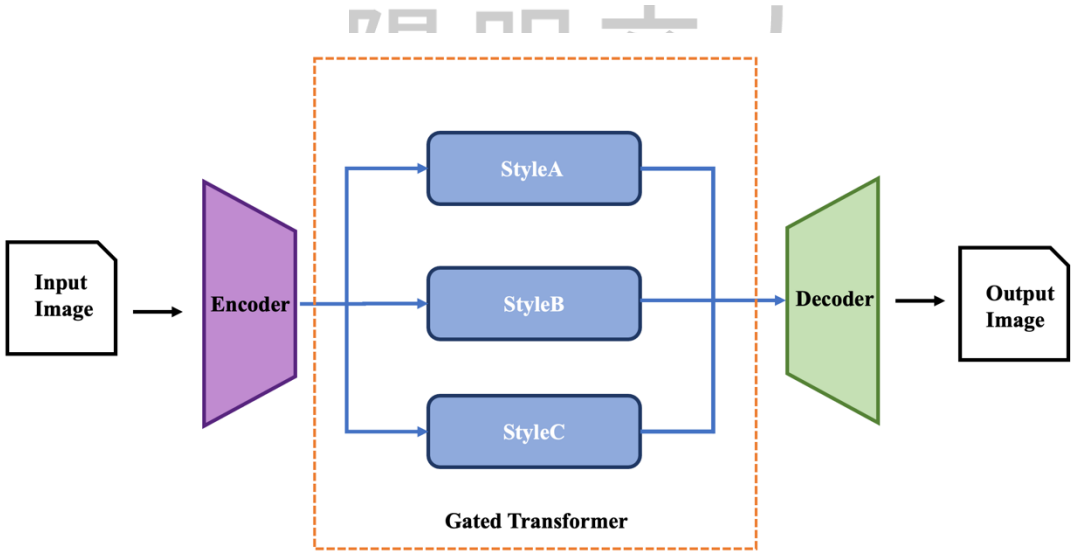

Figure 2.3 The architecture of the Gate-GAN: an encoder, a gate transformer, and a decoder. In each branch, images generate their own style through the gate transformer.

Chapter 3

Method

In this chapter, we introduce the main idea of our method in Section 3.1. Then, we illustrate the detail of our model architectures in Section 3.2. Finally, we present the loss functions which we used in Section 3.3.

3.1 Overview

The goal of our approach is to transfer the image of a real photo to a painting in the style of a famous painter. Unlike Gatys et al. [4], our method does not need to rely on the reference image in the input stage. Referring to a reference image in the input stage will cause overdependence on the style of scene that reference image has. Also, the similarities between content image and reference image will affect the quality of output. To synthesize the style of painters without limited to reference image, we only input a real photo to achieve this goal, as shown in Figure 3.2. The framework aims to generate artistic images without reference image while still maintaining the artistic style. We further build a sub-stream to obtain auto encoder loss which reconstructs the content structures [2]. Besides, we use reference image to compute adversarial loss in the discriminative network instead. We also use total variation loss to smooth the generated image. The architectures of our proposed method are shown in Figure 3.1. The specific method is detailed in Section 3.2. By comparing with previous methods in Section 4,

our framework will synthesize more realistic results and match the scene that original painters may draw practically.

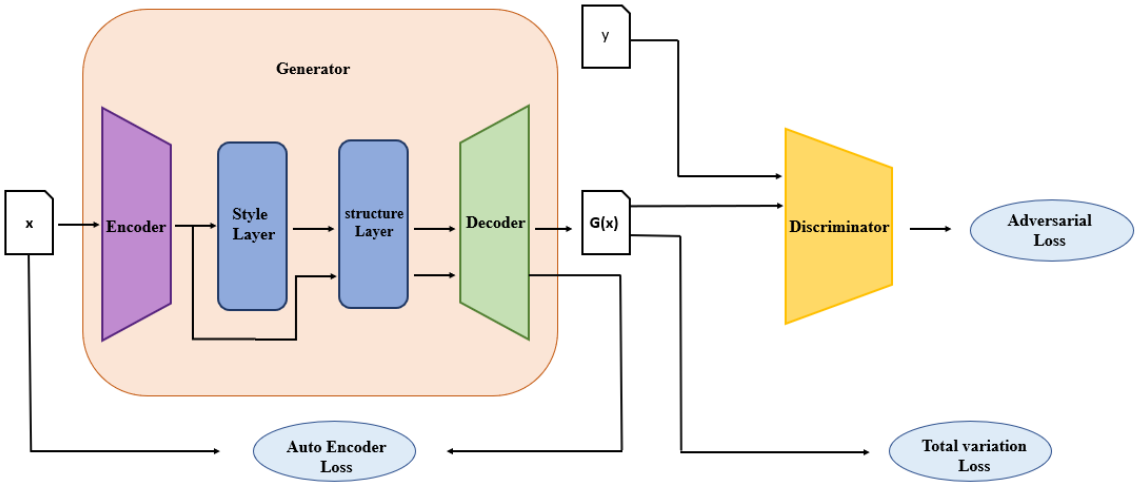

Figure 3.1 Architectures of our method.

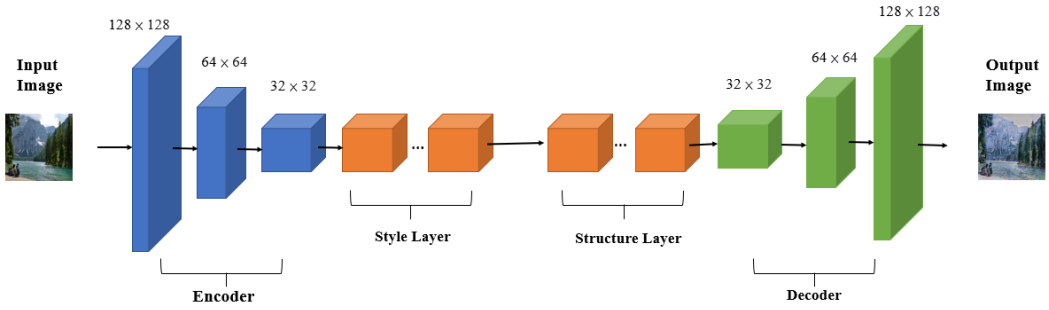

Figure 3.2 The process of our approach only needs one input.

3.2 Network Architecture

We propose a GAN-based method to transfer input domain X to style domain Y. We collect a series of real photos $\{x_i\}_{i=1}^{N} \in X$ and artistic images $\{y_i\}_{i=1}^{M} \in Y$. Therefore, we train a generator G to generate image G(x) which is in painter's style, and we train a discriminator D to distinguish the generated images G(x) from the artistic image y simultaneously.

3.2.1 The Generator Network

Our task is to learn the styles in the style domain Y, we apply an architecture of generator network to generate images which look like artistic images of specific styles. In our proposed generator network, as shown in Figure 3.3, it consists of a style layer and a structure layer in the middle of encoder-decoder structure. The original encoder-decoder structure is unstable [1], as the GAN framework need to train two competition neural networks. One of the reasons is that it has more than one solution when generator learns the mapping function. To reduce the solutions of mapping function, we add style layer and structure layer to control diversity of images in target domain. The pipeline of generator consists of two streams. The top stream processes the input image x through encoder, style layer, structure layer and decoder in order to generate stylish results G(x). The bottom stream processes the input image X through encoder, structure layer, decoder would successfully reconstruct the input image as well as constrain the structure [2] of image R(x).At the same time, G(x) and R(x) are optimized during each training iteration. G(x) transfers the input image to an artistic image. R(x) generates a reconstructed image for using in the loss function.

Our detailed generator network has four parts: an encoder, a style layer, a structure layer, a decoder. The encoder consists of a 7×7 convolutional block and two 3×3 down-convolution blocks with stride 2 [2] which transform input images into feature space *Enc*(x). The style layer

S(.) includes three residual blocks [8], followed by encoder. The structure layer *T*(.) includes five residual blocks, followed by style layer. The decoder is constructed with two 3×3 transposed convolutional blocks with stride 1/2 and a 7×7 convolutional block [2] which decode the feature map from output of the structure layer *Dec*(.), as shown in Table 3.1.

In style layer, we found three residual blocks provide more accurate results and increase the training efficiency. We will present our experiment in section 4. To stable the training, we also use pretraining to make the parameters have proper initial weights. Totally, the output images of generator network can be represented as:

$$G(x) = Dec(T(S(Enc(x)))) \tag{3.1}$$

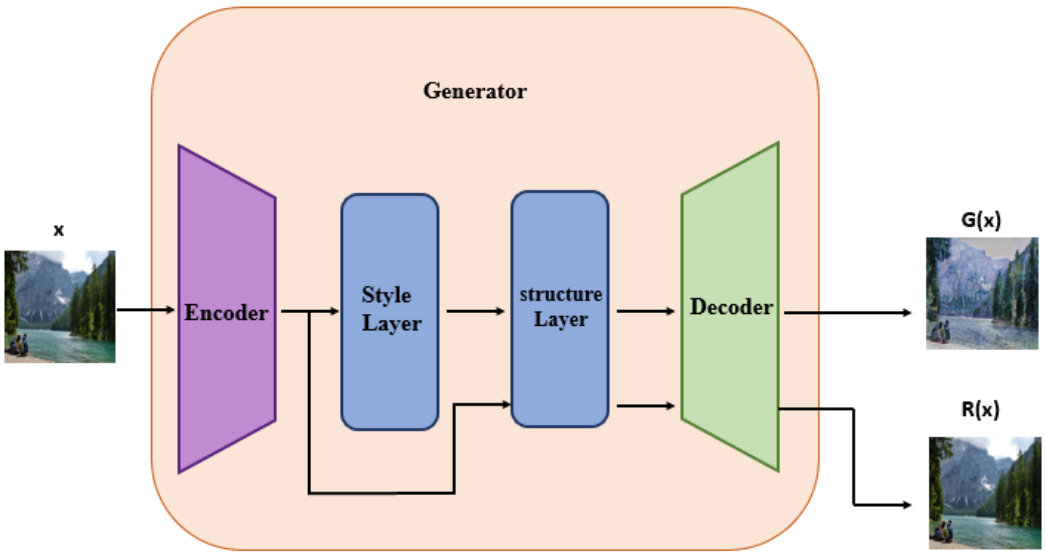

Figure 3.3 The pipeline of our generator. x is input image, G(x) is generated stylish result and R(x) is a reconstructed image to compute auto-encoder reconstruction loss.

TABLE 3.1

	Operation	Kernel size	Stride
Encoder	Convolution	7×7	1
	Convolution	3×3	2
	Convolution	3×3	2
Style layer	Residual block		
	Residual block		
	Residual block		
Structure layer	Residual block		
	Residual block		
	Residual block		
	Residual block		
	Residual block		
Decoder	Transposed Convolution	3×3	1/2
	Transposed Convolution	3×3	1/2
	convolution	7×7	1

3.2.2 The Discriminator Network

For the discriminator network, we use the Patch-GAN architecture [10]. Comparing to previous GAN discriminator, Patch-GAN captures style statistics at the scale of local image patches. It maps the image to N×N patch and judge the image is real or fake. Each point of N×N patch represents the region value of original image, as shown in Figure 3.4. Region value is the value of a patch. If region value is 1, it is real. Otherwise, it is fake. The advantage of this method is that we can concern more regions detail on each local patch. Besides the number of parameters are less than previous GAN which can make the training process faster. In our model, the discriminator network consists of five convolutional layers [10]. The patch size is equivalent to the receptive field [19]. Because the last layer of receptive field is 4, we choose 70×70 patches for our training. The function of receptive field can be formulated as follow:

$$\text{Receptive field} = (\text{output size} - 1) \times \text{stride} + \text{kernel size}. \quad (3.2)$$

In our proposed discriminator network, as shown in Figure 3.5, it consists of two inputs. The first input y is an artistic image which is randomly selected from collected dataset. The

second input G(x) is the output of generator network. Then, the discriminator network computes the adversarial loss of two inputs. The difference between Patch-GAN and previous GAN discriminator is that rather the previous GAN maps from an image to a single scalar output, which signifies real or fake. The Patch-GAN maps from an image to a N×N array output. We can trace back its receptive field to see which input pixels it is sensitive to. By calculating the adversarial loss of Patch-GAN, the generator network can handle the style of image more effectively.

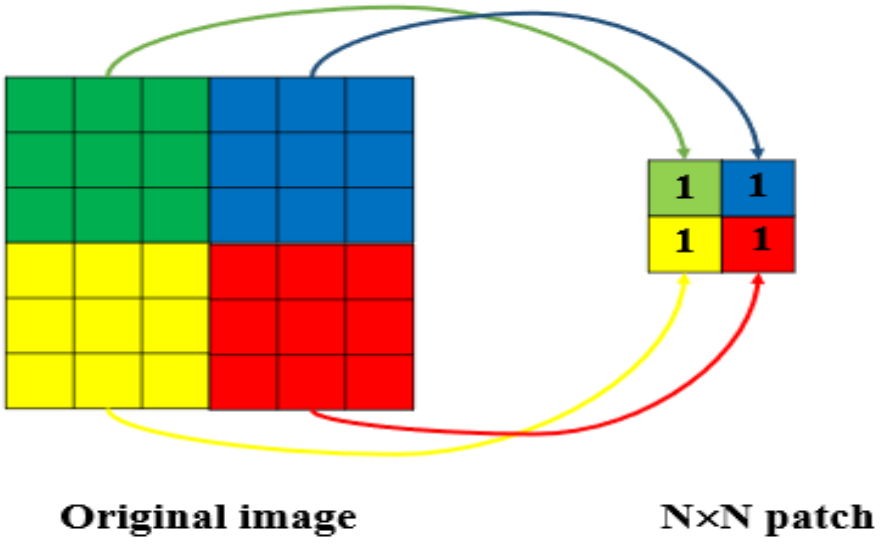

Figure 3.4 Patch-GAN map the image to N×N patch and judge it is real or fake. Region value is the value of a patch. If region value is 1, it is real. Otherwise, it is fake.

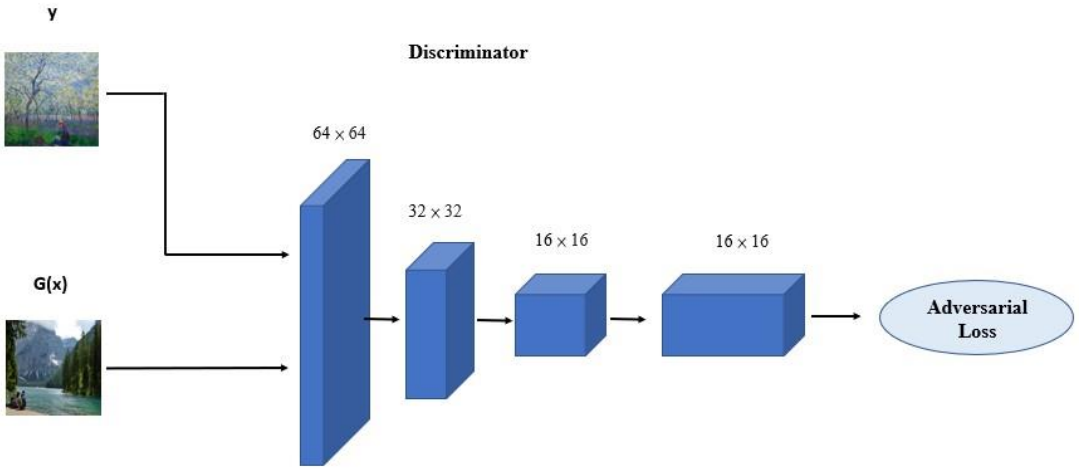

Figure 3.5 Our proposed discriminate network.

3.3 Loss Function

Based on the characteristics of style transfer, we introduce three losses for our objective function: (1) the adversarial loss L_{GAN}, which makes generator generate ideal images by competing in a two-player game. (2) the auto-encoder reconstruction loss L_{AE}, which minimizes the differences between the original input image and the generated reconstructed-image. (3) the total variation loss L_{TV}, which smooths away the noise in flat region and preserve the clear edges. The full objective function is shown as follow:

$$L = L_{GAN} + \lambda_{AE} L_{AE} + \lambda_{TV} L_{TV} \tag{3.3}$$

Where λ controls the relative proportion of these three losses. λ_{AE} maintains the balance of style transfer and content structure preserving. A larger λ_{AE} provides the output images with more content information, generating less stylish images. A smaller λ_{AE} learns more on the stylization, losing semantic information of content structure. λ_{TV} also have a lot of influence on content

structure preserving by smoothing the generated image. The proper λ_{TV} can preserve the clear edges and reduce noise. Empirically, we set $\lambda_{AE} = 10$ and $\lambda_{TV} = 10^{-6}$ in our objective function.

3.3.1 The Adversarial Loss

The generator G is to make the results look like the artistic images y, which is a collection of target domain images. Simultaneously, the discriminator D needs to distinguish a generated image between the synthesized images and the real artistic images. The generative adversarial function is shown as follow:

$$L_{GAN} = E_{y\sim Y}[log(D(y)] + E_{x\sim X}[log(1 - D(G(x))] \quad (3.4)$$

where y are artistic images from the distribution Y, x are real photos from the distribution X, $G(x)$ is generated images by generator G, $D(.)$ is discriminator.

3.3.2 The Auto-Encoder Reconstruction Loss

If the network architecture just trained with adversarial loss, an input image would map to multiple output images in the target domain. The auto-encoder network reduces data dimensions as well as reconstruct input images. First, the auto-encoder learns how to compress and encode data. Then, it learns how to reconstruct the data from the encoded data to an output image that is similar to the original input. We use auto-encoder loss to constrain output images in our sub-stream, so that the structure of the outputs would be similar to the input images. The auto-encoder is L1 loss which is defined as follow:

$$L_{AE} = E_{x\in X}[||Dec(Enc(x) - x)||_1] \quad (3.5)$$

where $Enc(.)$ is the encoder, $Dec(.)$ is the decoder.

3.3.3 The Total Variation Loss

Inspired by Rudin et al. [18], we use total variation loss which encourages spatial smoothness in the generated images. As shown in Figure 3.6, the total variation loss removes unwanted details such as noise but preserving important edges. This consequence would also have a remarkable effectiveness for impressionist art. The total variation loss [16] is defined as follow:

$$L_{TV} = \sum_{i,j}[(G(x)_{i,j+1} - G(x)_{i,j})^2 + (G(x)_{i+1,j} - G(x)_{i,j})^2]^{1/2} \quad (3.6)$$

Where $G(x)$ is the generated image which is $H \times W$ dimension, $i \in (0,\ldots,H-1)$ and $j \in (0,\ldots,W-1)$.

Figure 3.6 Left: an original image. Right: an image with total variation loss. From Mahendran et al. [18].

Chapter 4

Implementation and Experimental Results

In this chapter, we show the implementation detail in Section 4.1. Then, we show the datasets which we collected for training and testing in Section 4.2. Finally, we present our results and compare our approach with previous approach in Section 4.3.

4.1 Implementation details

The hardware we used are Intel Core i7-8700CPU 3.20GHz and Nvidia GeForce GTX 1080 GPU. We use Pytorch and cuda 10.2 to implement our algorithm. Equation 3.3 is the total loss function we used to train the network. The default parameters of λ_{AE} and λ_{TV} in Equation 3.3 are 10 and 10^{-6}. We use Adam [11] for optimizing the learning rate. The learning rate is set to 0.0002. Due to lack of artistic image, we use pre-training in our algorithm. Pre-training means training a model to help it form parameters that can be used in other models. Using parameters that have been learned could help new models successfully have a good start from old experience instead of scratch. In the pre-training stage, we train 10 epochs for 13 hours; however, in the training stage, we train 200 epochs. In the first 100 epochs, we maintain the same learning rate. In the last 100 epochs, we decay the learning rate linearly. To make training faster, we initially scale the image to 143×143. Then, we randomly crop and flip the image to 128×128 which is used for data augmentation. The batchsize is set to 16. To train a painter's style, it takes about 23 hours. After training, the model takes about 0.37 seconds to generate a

256×256 image and 1.14 seconds to generate a 512×512 image. Time cost is shown in Table 4.1.

TABLE 4.1: Time cost at each stage

	Pre-training	Training	Generating a 256× 256 image	Generating a 512× 512 image
Times	13 hours	23 hours	0.37sec	1.14sec

4.2 Datasets

We choose the COCO dataset [13] for pre-training. The COCO dataset is one of the most popular image datasets which is available for public. It is also a supplement to style transfer, which the data used for one model serves as a starting point for another. In pre-training stage, total usage of images is 82768, and then split them into 5173 batches randomly. In training stage, we use 6288 content images for training and 526 content images for testing. In terms of reference images, we generate images in style of Monet, Van Gogh, Cezanne, whose datasets are from [20]. The amounts of images for Monet, Van Gogh, Cezanne are 1073, 400 and 526, respectively.

4.3 Experimental Result

We first compare our method to GateGAN [1] which is also generating images in style of Monet, Van Gogh, Cezanne. Figure 4.1, 4.2 and 4.3 show the comparison between our result and GateGAN. The resolution of both methods are 512×512. As you can see, the top left image is input and the top right image is painter's painting. In order to show the difference clearly, we choose the scenes which is closed to original painter's painting. In Figure 4.1, we can generate the blue black shadow which Van Gogh liked to use. In Figure 4.2, GateGAN not only fails to

keep the shape of building but also has artifacts. By using the auto-encoder reconstruction loss and two-stream generator, the result of our building has the sense of stereoscopic. We also generate various green colors that Cezanne would use in painting. Monet's brush stroke is used to portray the streamlined stroke on the gentle landscape. In Figure 4.3, the strokes of our method are more streamlined than GateGAN.

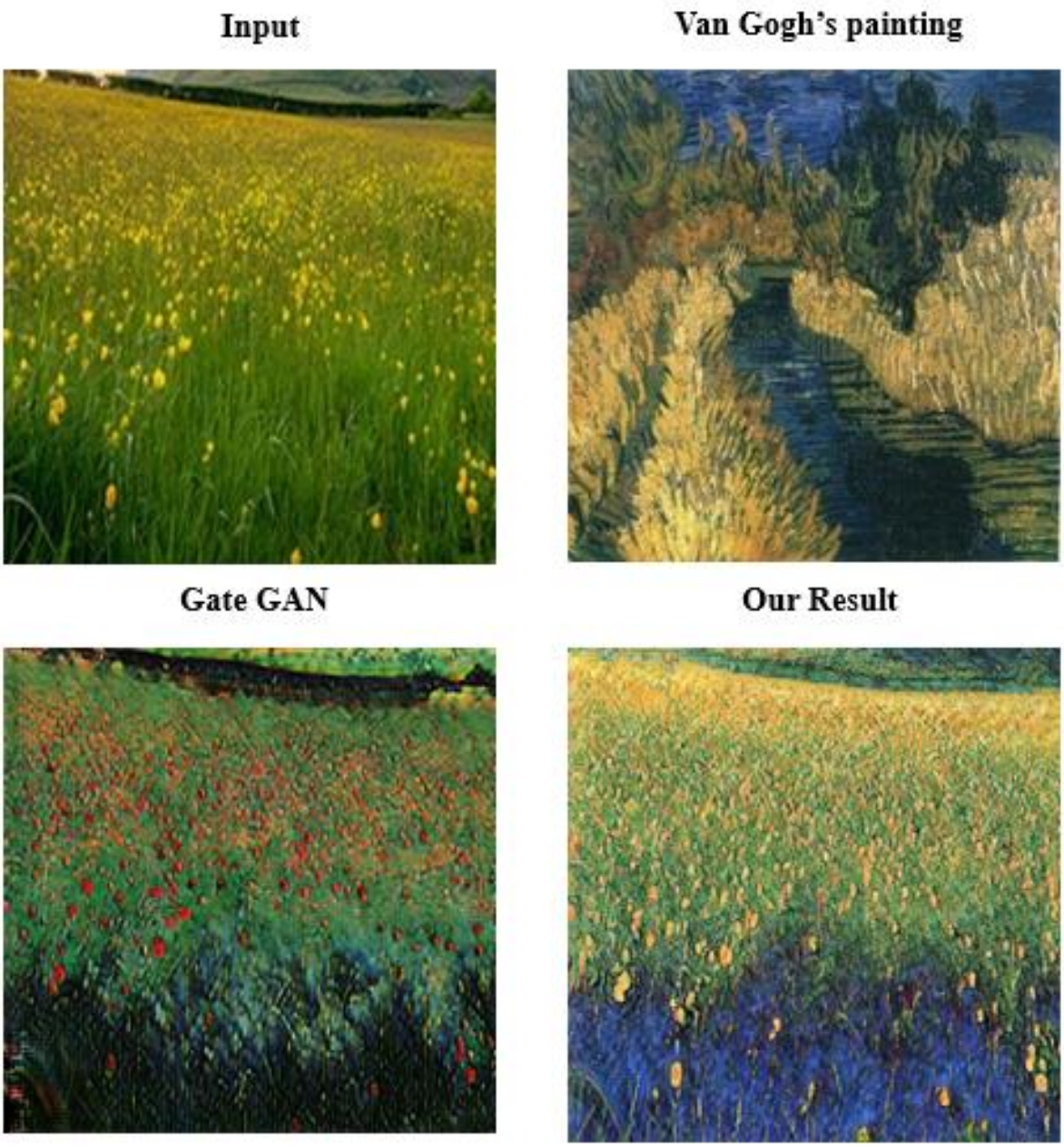

Figure 4.1 Comparison between Van Gogh's painting, GateGAN and our result.

Figure 4.2 Comparison between Cezanne's painting, GateGAN and our result.

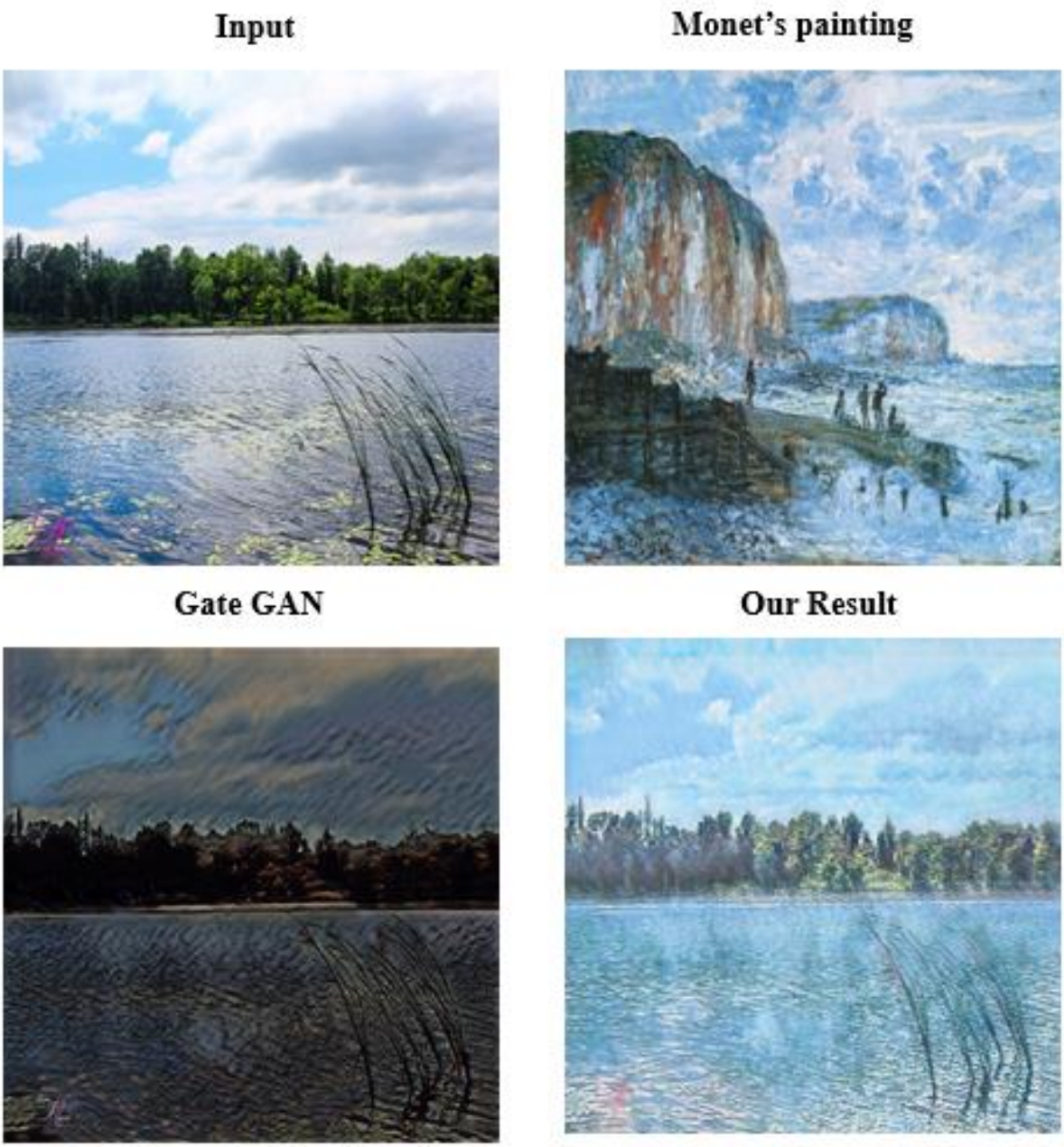

Figure 4.3 Comparison between Monet's painting, GateGAN and our result.

4.4 Ablation Study and Comparison

Figure 4.4 shows the difference between the one-residual style layer, two-residual style layer and three-residual style layer. The top left is input image. The top right is one-residual style layer image which not only fails to stylize on bridge region bot also produces artifacts on the sky. The bottom left is two-residual style layer image which also has some artifacts and a few ripples on the water region. The bottom right is three-residual style layer image which has better stylization result in the bridge region, sky region and water region. By this experiment, we use three residual blocks in our style layer.

Figure 4.5 shows the difference between our results and Van Gogh's painting. Van Gogh's paintings are special due to the fact that his depiction of figures, light, and landscape can be admired without the need for color. His bold palette is one of the most recognizable features of his painting. We choose the scene that is closed to Van Gogh's painting as an input. Van Gogh painted with dark and melancholy color that suited his mood in the painting. Our result has dark blue color in the shadow. Mountain and plants have melancholy green color.

Figure 4.6 shows the difference between our results and Monet's painting. Monet's brush stroke is a key feature of his paintings. His renowned usage of color is directly linked to his expression of light. Just like Monet's painting, the only tangible solidly painted form in our result is the man and everything else loses consistency depending on its distance from the ray of light. Besides, we also have widely ranged colors in our result.

Figure 4.7 shows the difference between our results and Cezanne's painting. Cezanne used heavy brush strokes and thickly layered paint onto canvas. We can see that the strokes of our result are also thick and heavy. We also have various green colors that Cezanne would use in painting

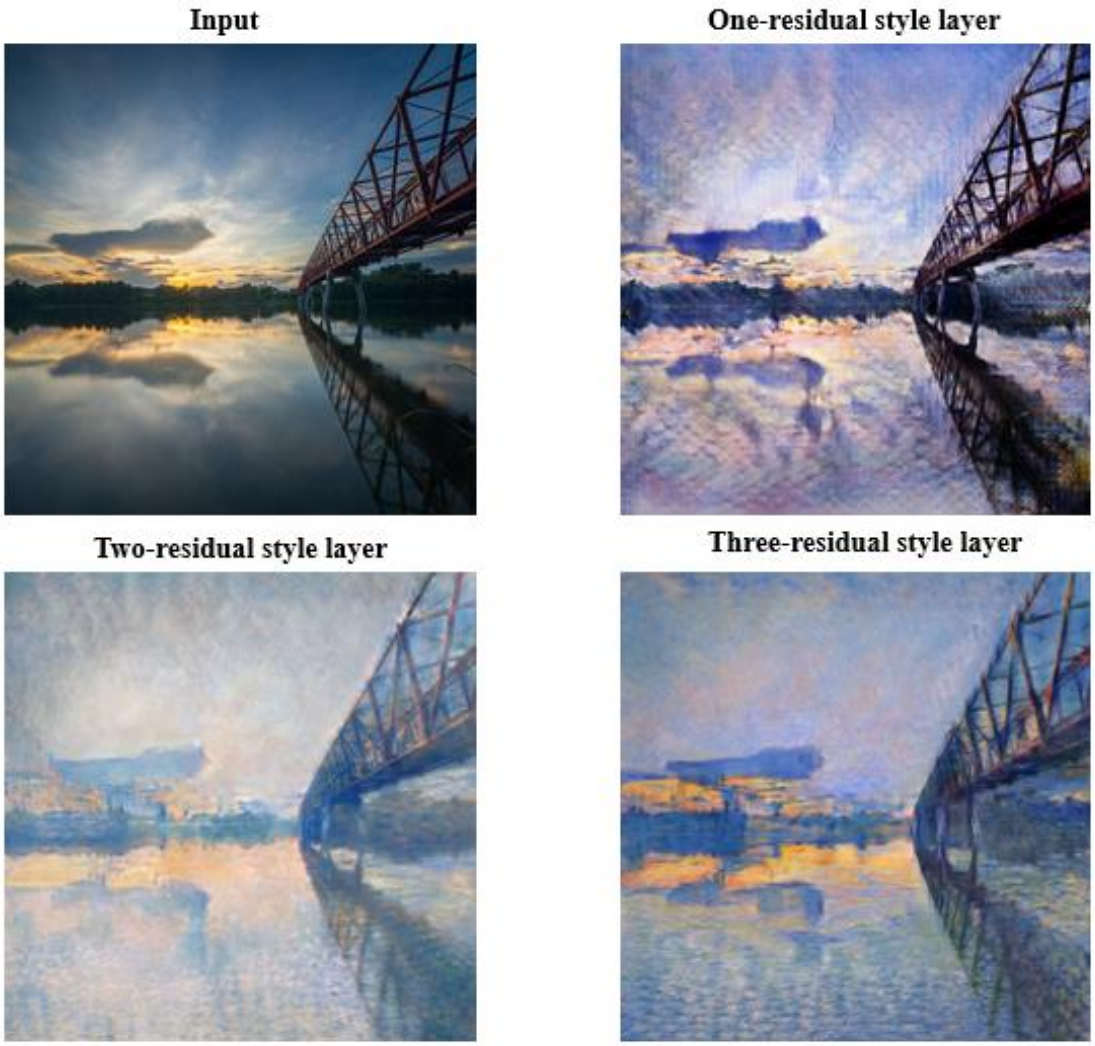

Figure 4.4 Comparison between the one-residual style layer, two-residual style layer and three-residual style layer.

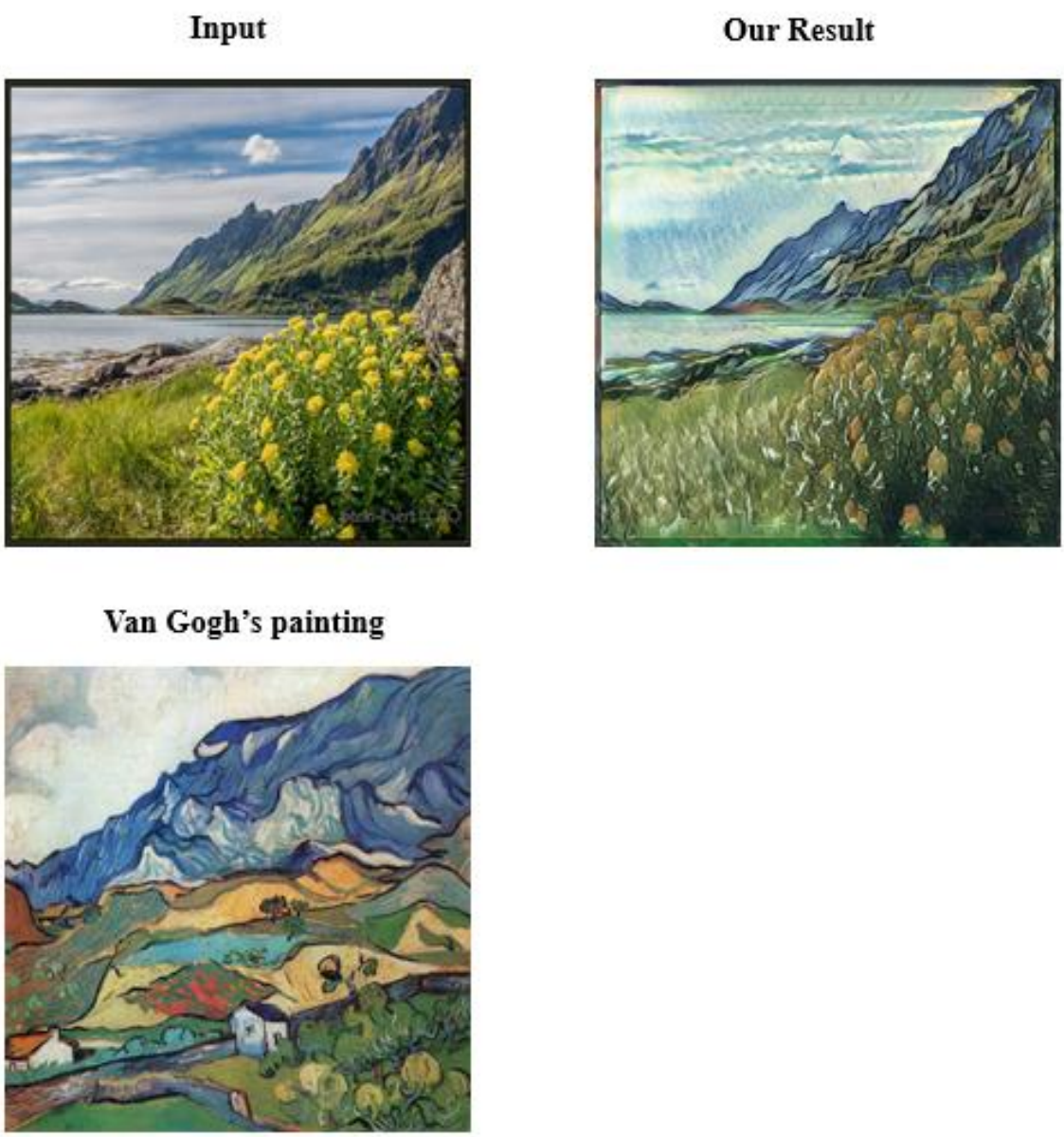

Figure 4.5 Comparison between our results and Van Gogh's painting.

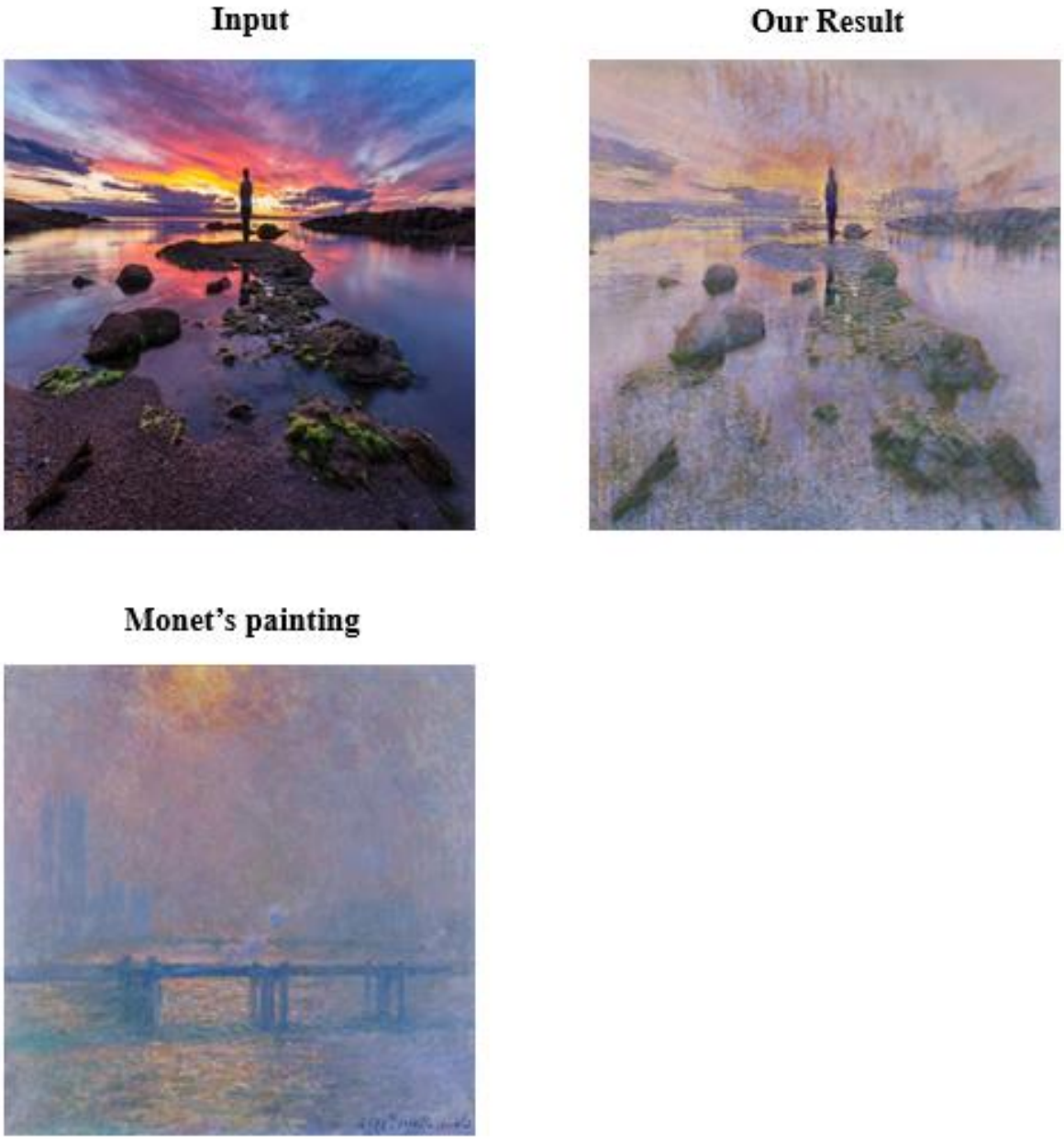

Figure 4.6 Comparison between our results and Monet's painting.

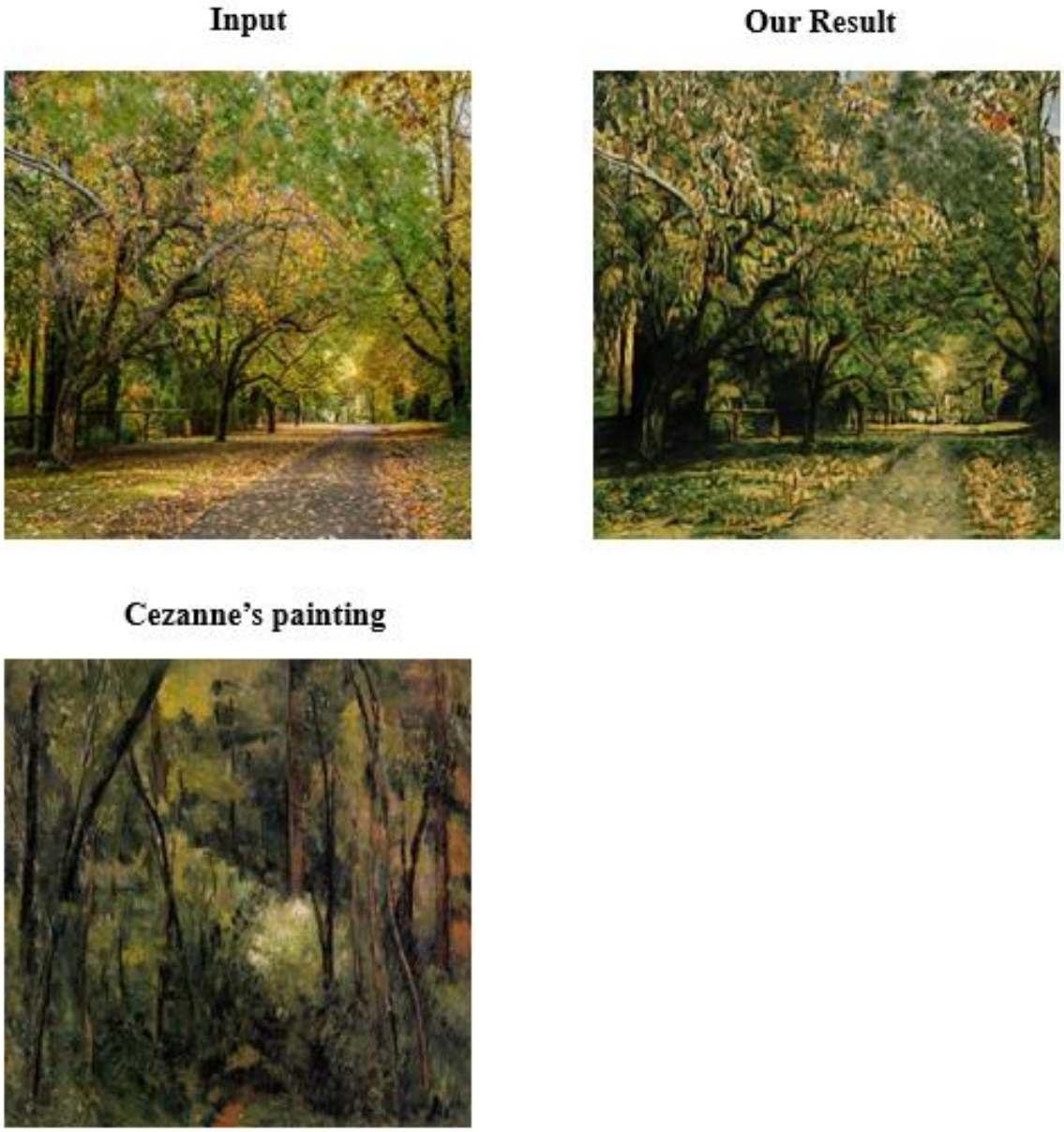

Figure 4.7 Comparison between our results and Cezanne's painting.

Chapter 5

Conclusions

In style transfer neural network, previous works [4, 14, 15, 21] rely on the reference image as input. However, referring to the reference image will cause overdependence on the style of reference image. Besides, the similarities between content image and reference image will affect the quality of output. Due to these disadvantages, we try to solve these problems and improve the quality.

In this thesis, we propose the synthesis of specific painter's style based on deep learning. The resolution of our results is 512×512, rather than 128×128. We proposed three main ideas in our algorithm. First, to reconstruct the content structure, we build a sub-stream to obtain auto-encoder reconstruction loss. Second, we use total variation loss to smooth the images and avoid unwanted artifact. Finally, we use pre-training to stabilize the GAN training. Experiments demonstrate our method can simulates the three styles of impressionist painters successfully.
In the future, we will apply our model to simulate other style of painting (e.g., Realism, Naturalism) and explore to generate difference style transfer results.

Reference

[1] Xinyuan Chen, Chang Xu, Xiaokang Yang, Li Song, and Dacheng Tao. Gated-gan: Adversarial gated networks for multi-collection style transfer. IEEE Transactions on Image Processing, 28(2):546–560, 2018.

[2] M.-M. Cheng, X.-C. Liu, J. Wang, S.-P. Lu, Y.-K. Lai, and P. L. Rosin, "Structure-preserving neural style transfer," *IEEE Trans. Image Process.*, vol. 29, pp. 909–920, 2020.

[3] L. Gatys, A. S. Ecker, and M. Bethge, "Texture synthesis using convolutional neural networks," in Proc. NIPS, 2015, pp. 262–27

[4] Leon A Gatys, Alexander S Ecker, and Matthias Bethge. Image style transfer using convolutional neural networks. In CVPR, 2016

[5] J. Gauthier. Conditional generative adversarial nets for convolutional face generation. Class Project for Stanford CS231N: Convolutional Neural Networks for Visual Recognition, Winter semester, 2014(5):2, 2014.

[6] B. Gooch, G. Coombe, and P. Shirley, "Artistic vision: painterly rendering using computer vision techniques," in *NPAR*, 2002.

[7] I. J. Goodfellow *et al.*, "Generative adversarial nets," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 2672–2680.

[8] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2016, pp. 770–778.

[9] A. Hertzmann, C. E. Jacobs, N. Oliver, B. Curless, and D. H. Salesin, “Image analogies,” in *Proceedings of the 28th annual confer- ence on Computer graphics and interactive techniques.* ACM, 2001.

[10] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, “Image-to-image translation with conditional adversarial networks,” in The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), July 2017.

[11] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” in *Proc. Int. Conf. Learn. Represent.,* Dec. 2014, pp. 1–15.

[12] A. Kolliopoulos, “Image segmentation for stylized non- photorealistic rendering and animation,” Ph.D. dissertation, Uni- versity of Toronto, 2005.

[13] T. Lin et al., “Microsoft COCO: Common objects in context,” in Proc. ECCV, 2014, pp. 740–755

[14] Y. Liu *et al.*, “Image neural style transfer with preserving the salient regions,” *IEEE Access*, vol. 7, pp. 40027–40037, 2019.

[15] X. Liu, Z. Liu, X. Zhou, and M. Chen, “Saliency-guided image style transfer,” in *Proc. IEEE Int. Conf. Multimedia Expo Workshops (ICMEW)*, Jul. 2019, pp. 66–71.

[16] A. Mahendran and A. Vedaldi. Understanding Deep Image Representations by Inverting Them. arXiv:1412.0035 [cs], Nov. 2014. arXiv: 1412.0035.

[17] P. Rosin and J. Collomosse, Image and Video-Based Artistic Stylisation, vol. 42. Springer, 2012

[18] L. Rudin, S. Osher, and E. Fatemi, “Nonlinear total variation based noise removal algorithms,” Phys. D, vol. 60, pp. 259–268, 1992.

[19] H. Sim, S. Ki, S. Y. Kim, J.-S. Choi, S. Kim, and M. Kim. High-resolution image dehazing with respect to training losses and receptive field sizes. In 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), 2018. 3, 6

[20] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, “Unpaired Image-to-Image translation using cycle-consistent adversarial networks,” in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV),* Oct. 2017, pp. 2223–2232.

[21] Yulun Zhang, Chen Fang, Yilin Wang, Zhaowen Wang, Zhe Lin, Yun Fu, and Jimei Yang. Multimodal style transfer via graph cuts. In ICCV, 2019.