Learning for Visual Synthesis and Transformation

Xinyuan Chen

Faculty of Engineering and Information Technology

University of Technology Sydney

A thesis submitted for the degree of

Doctor of Philosophy

April 2020

I would like to dedicate this thesis to my loving parents
Lianfu Chen and *Meiyun Mao*

Certificate of Original Authorship

I certify that the work in this thesis has not previously been submitted for a degree nor has it been submitted as part of requirements for the collaborative degree and/or fully acknowledged within the text.

I also certify that the thesis has been written by me. Any help that I have received in my research work and the preparation of the thesis itself has been acknowledged. In addition, I certify that all information sources and literature used are indicated in the thesis.

This research is supported by the Australian Government Research Training Program.

This thesis is the result of a Collaborative Doctoral Research Degreee program with Shanghai Jiao Tong University.

signature: Production Note: Signature removed prior to publication.

Acknowledgements

I would like to take this good opportunity to appreciate my advisors, several professors, my colleagues, my friends and my family for their significant help during my doctoral study in Shanghai Jiao Tong University.

My deepest gratitude goes first and foremost to my supervisor Prof. Dacheng Tao for his continuous support, unlimited patience and supportive guidance. He was always ready to help, and was very willing to teach everything he knows to me. His incredible enthusiasm and high demand motivate me to set up high academic research standards and submit papers to the leading journals or conferences in my research field.

I would also like to thank my supervisor Prof. Xiaokang Yang from Shanghai Jiao Tong University. He has given me not only plenty of freedom to explore in my research field, but also numerous constructive suggestions to help me out of various difficulties in study and life. I am particularly impressed by his encouraging attitude and expert knowledge for my research. Without his high academic standard, patient guidance, and unreserved support, I can never imagine I could have finished this thesis.

Besides my principal supervisors, I would like to give special thanks to my advisor: Dr. Chang Xu. I am grateful for his great patience in answering my simple and nearly endless questions from research motivation to implementation details, as well as improving my writing word-for-word. He taught me how to find interesting ideas, how to develop solid algorithms, and how to write technical papers all from scratch. I also thank Dr. Youming Qiao, Dr. Guoqiang Zhang

in UTS, Dr. Tongliang Liu, Dr. Shaoli Huang in USYD for their generous help during my study in Sydney.

I would also like to thank Prof. Li Song who led me into the field of computer vision. During my early days of academic research in Shanghai Jiao Tong University, he gave me valuable instructions and generous support. I also would like to thank Dr. Chao Ma. As a senior, he set up a great role for me. As an advisor, he provided me a lot of suggestions and help. Moreover, I would like to thank Prof. Bingbing Ni, Yi Xu and Guangtao Zhai for their advices and help.

The most enjoyable thing during my Ph.D. study is the opportunity to meet my dear colleagues and friends. I would like to thank Yilin Dong, Mengyue Shi, Yihang Huang, Han Zhang, Shaowei Xie, Zhe Ren, Genning Zhang for being close friends and offering me timely companionship; thank Muming Zhao, Dr. Jiangchao Yang, and Guo Lu for generous help and company in both Shanghai and Sydney; and thank Dr. Yichao Yan, Minsi Wang, Jingwei Xu, Wenbo Bao, Yunqian Wen, Jun Ling, Han Xue and many others.

Then comes my dear colleagues and friends in my doctoral study in Sydney: Dr. Erkun Yang, Dr. Chaoyue Wang, Yuxuan Du, Yali Du, Dr. Shan You, Dr. Jianfeng Dong, Yuangang Pan, Dr. Baosheng Yu, Dr. Liu Liu, Dr. Xiyu Yu, Dr. Huan Fu, Shanshan Zhao, Lianbo Zhang, Dalu Guo, Zeyu Feng, Zhe Chen, Rui Geng, Gengxing Wang and Xiaofei Liu. To my friends, thanks for all your support and company during my joyful and stressful time, and I cherish our friendship. To my colleague, it is fantastic to have the opportunity to work with you.

Finally, I want to dedicate this thesis to my parents who gave me unreserved love and support. Every Friday night, you always wait for my phone to hear my experience of study and life, share my joys and sorrows. Because of you, I never feel alone and frustrated through my difficult times. Thanks for always being there for me.

Abstract

Visual synthesis is one of the most fundamental problems in computer vision and artificial intelligence. Visual synthesis aims to create pixel-level data (e.g., images and videos) based on descriptions such as texts, noise, semantic annotations and images. Recently, deep generative learning has greatly promoted the development of visual synthesis. However, the existing generative methods still suffer from several issues, including model interpretation, controllability, stability, efficiency and performance. In this thesis, several generative models are proposed to address these challenges. This thesis makes the following contributions:

First, this thesis introduces an attention generative model for local image synthesis, so as to improve the controllability and interpretation of the generative model. How to precisely locate the foreground region in the image and generate the target object to the specified region is the key problem in the local image synthesis task. The object transfiguration task is an application of the local image synthesis, which aims to transform the object of images to another object. Existing generative methods often fail to decompose the foreground and background. In this thesis, the attention mechanism is incorporated into generative models, so as to transform the object of our interests without altering the background. The model is built by decomposing the generative network into two separate networks, each of which is dedicated to one sub-task: to detect the region of interests and to generate the object from one object to another. The attention network predicts spatial attention maps of images, and the transformation network focuses on translating objects. The attention network produces attention maps which are encouraged to be sparse so that the model

can only pay attention to the objects of interest. Also, a novel perceptual loss is introduced to improve the quality of transformed images in the high-level feature space. Experimental results demonstrate the necessity of investigating attention in image-to-image transformation, and that the improvement of the quality of generated images.

Second, this thesis proposes a multi-domain generative model that multiple styles of images can be generated in a single network. The major challenge is how to efficiently generate multiple styles in a single network. Our model is capable to extract the content and style feature of images, and apply multiple style features to the content image.This thesis proposes a gated generative model that consists of three modules: an encoder, a gated transformer, and a decoder. Different styles can be achieved through different branches of gated transformers while the encoder and decoder are used for capturing structure information sharing weights for all styles. A discriminative network is used to distinguish whether the input image is a stylized or genuine image. An auxiliary classifier is used to recognize the style categories of transferred images, thereby helping to generate images in multiple styles. In addition, to stabilize the adversarial training process, an auto-encoder reconstruction loss is introduced by combining the encoder and decoder module. Extensive experiments demonstrate the stability and effectiveness of the proposed model for multi-domain image synthesis.

Third, this thesis investigates the video synthesis problem on the long-term horizon. A temporal generative model is proposed for long-term video frame prediction. The existing generative model for video prediction usually cannot output high-quality predictions for a long-time horizon. The reason is that those methods recursively output subsequent frames by taking the newly generated frames as observations, consequently the prediction error accumulates dramatically. The introduced retrospection process is designed to look back on what has been learned from the past and rectify the prediction deficiencies.

To this end, a retrospection network is built to reconstruct the past frames given the currently predicted frames. On the other hand, an auxiliary route is built by reversing the flow of time and executing a similar retrospection. These two routes interact with each other to boost the performance of retrospection network and enhance the understanding of dynamics across frames, especially for the long-term horizon.

Overall, this thesis investigates the deep generative model and solves several practical issues for visual synthesis and transformation. For local image synthesis, we propose an attention generative model. We also propose a gated generative model for generating multi-domain of images in a single generative network. For video synthesis, a temporal generative model is proposed to output long-term video frames by incorporating the prediction and retrospection process in the model. Extensive experimental results on large-scale benchmark datasets demonstrate that the proposed methods in this thesis perform favorably against previous visual synthesis algorithms in terms of efficiency, controllability, and robustness.

Contents

List of Figures

Chapter 1

Introduction

1.1 Background

Vision is the way for humans to describe and understand the outside world. With the rapid development of information systems and online sharing media, visual content is intensively consumed in daily life. In computer vision, it is an important problem to automatically deal with this visual information as intelligently as human beings. So far, the most striking successes in computer vision involve visual understanding methods, which usually map high-dimensional pixel-level inputs to high-level concepts and semantic representations, such as image classification [51, 122, 131], image segmentation [20, 47], object detecting [116, 154], etc.

On the other hand, an advanced intelligent agent is also expected to have the ability of creativity, such as creating paintings, designing clothing, etc. Visual synthesis is another side of visual intelligence in computer vision, that produces pixel-level visual data from high-level concepts. As shown in Figure 1.1, visual synthesis operates in the opposite direction of visual understanding. In recent decades, visual synthesis has been an active research area in computer vision, such as image generation [16, 44, 64], texts to image generation [147, 161], image-to-image translation [59, 151, 183], and video synthesis [37, 148, 159], etc. In other words, visual synthesis is the process of creating pixel-level data (e.g., images, and videos) from some forms of descriptions, such as random noise, texts, and images.

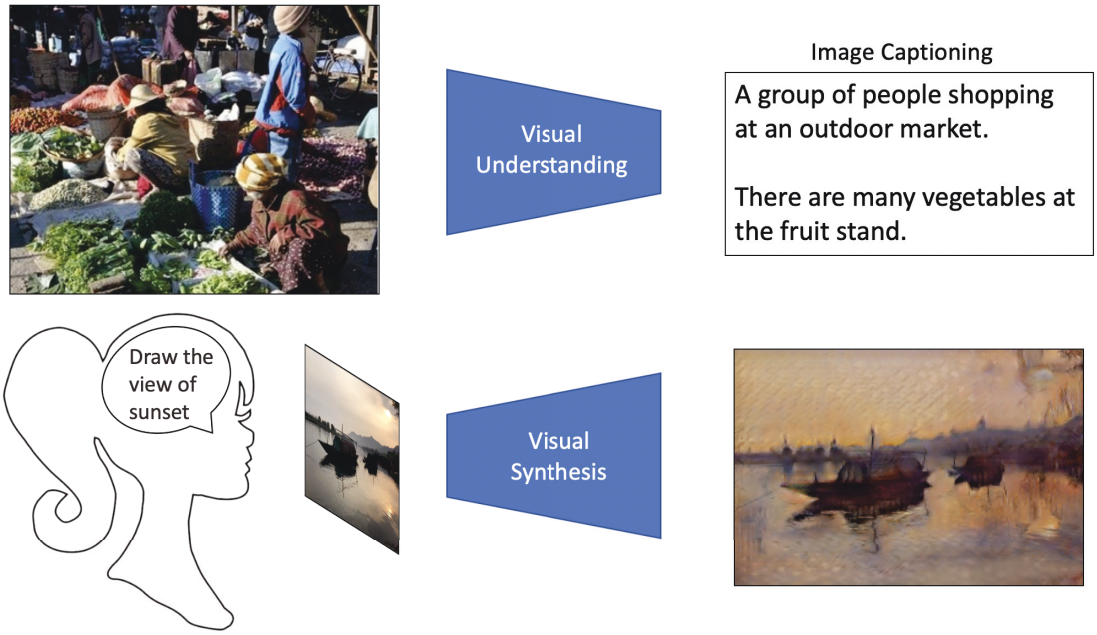

Figure 1.1: Computer vision and deep learning algorithms have shown great progress in visual understanding, e.g., image caption [147]. Our work focuses on an opposite way, visual synthesis with the goal of generating pixel-level visual data from abstraction concept.

The generative adversarial networks (GANs), which have recently attracted attention and had promising results, are used to generate data from noise. Based on the conventional GANs model, the conditional GAN has been proposed to apply on the image-to-image and video-to-video transformation. Image-to-image transformation, also called image-to-image translation, is used to map images from one domain to another and has been widely used such as style transfer, object transfiguration, video frame prediction, etc.

Realistic visual synthesis is in high demand in both industry and personal life. In industry, visual synthesis can be applied to produce realistic imagery and visual effects (VFX) to save a lot of money and time for film-makers. Also, it helps cartoon producers automatically transform a picture of the landscape or city photograph to cartoon style paintings. In personal life, visual synthesis enables us to express compact concepts in the visual form (through photographs, paintings, sculptures, videos, etc.). For example, when users take photos of beautiful landscapes, the visual synthesis method helps to re-render the landscapes on canvas

as painted by artists as shown in Figure 1.1. Visual synthesis is also an important technique for image and video coding, whose decompressed process intrinsically requires the generation of good samples, such as image super-resolution, video frame rate up-conversion, etc. From the aspect of the theory of the generative model, visual synthesis can be used to test the models' ability to represent and manipulate high-dimensional data (e.g., natural images, videos), which are important in a wide variety of applied math and engineering domains. Moreover, visual synthesis can provide predictions on inputs with missing paired data and assist deep learning algorithms to be trained in the unsupervised or semi-supervised settings. Modern deep learning algorithms typically require extremely many labeled examples so that they can generalize well. Such unsupervised learning is one strategy for reducing the number of labels. The learning algorithm can improve its generalization by studying a large number of unlabeled examples, which are usually easier to obtain.

This thesis deals with the problem of generating realistic visual data, including images and videos. We explore more flexible, and controllable generative models, as well as investigate the efficient and stable generative model. Also, we study the video synthesis algorithm to generate high-quality frames in the long-term horizon.

Below we first review previous works on visual data synthesis and manipulation (Section 1.2). We then discuss the current challenges of visual synthesis in Section 1.3. In Section 1.4, we summarize the contribution of our works. Lastly, we give an overview of this thesis in Section 1.5.

1.2 Overview

Visual data synthesis aims to generate pixel-level data (e.g., images and video-frames), whose methods are usually based on generative models. In this section, we first introduce the related generative models. Then we briefly review the techniques and applications of visual data synthesis and transformation. Finally, we conclude the challenges in the visual synthesis problem.

1.2.1 Deep Generative Models

One main goal of statistics and machine learning is to represent and manipulate high-dimensional probability distributions of real-world data, such as natural images. Generative models are devised to represent the estimated probability distributions after observing the training samples. The unobserved data thus can be generated by sampled from the estimated probability distributions.

To estimate the probability distributions of the real-world data, most generative models are based on the principle of maximum likelihood. They define an explicit density function $p_{model}(x|\theta)$ for each training example and directly calculate the maximum likelihood. However, it is difficult to maintain computational tractability, especially when dealing with high-dimensional complex data distributions [43]. In contrast, implicit generative models are devised to avoid explicit defining a density function. Some of these implicit models based on drawing samples from a Markov chain transition operator that must be run several times to obtain a sample from the model. Markov chains easily fail to sample from high-dimensional spaces and usually increase the computational cost. Generative adversarial networks (GANs) are one of the approaches for implicit generative models. GANs are designed as a minimax game with two players, a 'generator' and a 'discriminator'. The generator intends to create samples that come from the same distribution as the training data while the discriminator examines samples whether they are real or fake. The generator ends up creating realistic samples that are indistinguishable from the training data.

Deep learning is well known for its powerful capacity of computation and representation. With the rapid development of the graphics processing units (GPUs), deep learning algorithms can construct very deep neural networks to fit large scale datasets. GANs are one of the deep generative models that can use deep networks as their generator and discriminator. Comparing to traditional generative models, GANs with deep architectures have better performance in generalization and expression. GANs are good at learning high-dimensional data distributions. Also, GANs can generate several samples in parallel at a time by going through the pre-trained generative network. However, GANs still face some challenges. GANs are notoriously difficult to train and often suffer from mode

collapse issues. Appropriate network architectures and training parameters (e.g., batch size, learning rate, and updating steps), are critical as unsuitable settings will significantly reduce the training stability and generative performance [68,78]. On the other hand, since neural networks are often used as black boxes, the interpretation and controllability for generative networks remain problems.

1.2.2 Visual Synthesis and Transformation

In my thesis, I focus on synthesizing images and videos in the pixel-level. Image synthesis is a fundamental problem for visual synthesis. Image synthesis is the process of creating new images from some forms of image descriptions. The forms of image descriptions include test patterns, noise from specific distribution, semantic annotations, and images. Among them, synthesizing images from input images is called image-to-image transformation or image-to-image translation. Many applications in image processing and computer vision can be viewed as image-to-image transformation, such as rendering grey images into RGB images, super-resolution, translating semantic labels into a street scene. Figure 1.2 shows the applications of image synthesis given input of image. The application of image synthesis can be classified as local image synthesis and global image synthesis. Local image synthesis refers to generate or manipulate certain parts of images while keeping the remaining region consistent. The applications for local image synthesis including virtual try-on [48], object transfiguration [24, 183]. Global image synthesis refers to generate images as a whole, such as style transfer [61], aerial photos into maps [183], and so on. Traditional image synthesis methods usually rely on low-level principles (e.g., the similarity of color, gradients or patches) and do not capture higher-level information about natural images. Examples of basic editing and synthesis include changing the color properties of an image either globally [119] or locally [75], or synthesizing texture regions [34, 35] and even larger coherent image content [49]. More advanced editing methods were proposed such as content-aware image resizing [6], image warping [55], or structured image editing [11] that intelligently reshuffle the pixels in an image following a user's edits. While achieving impressive results in the hands of an expert, when these types of methods fail, they usually produce unnatural and unrealistic results. The reason is that they neither have the concept of the editing object, nor

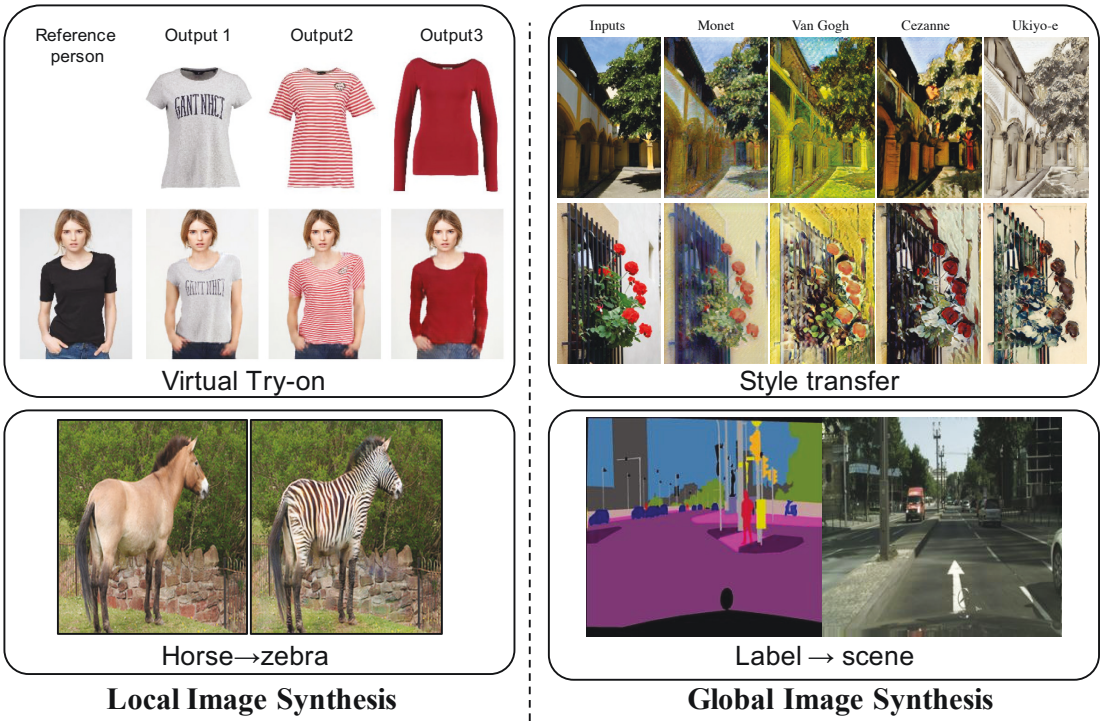

Figure 1.2: The applications of image synthesis given input of image. Local image synthesis refers to generate or manipulate certain parts of images while keeping the remaining region consistent. Global image synthesis refers to generate images as a whole.

the clue about what makes their output look like real natural images.

The deep generative learning frame enables the generator to learn high-level information from the training sets. With the invention of the GANs, a lot of neural networks have been trained to output high-resolution and high fidelity images [16, 64]. Meanwhile, some works exploit the condition GANs to generate images from semantic descriptions, such as translating semantic labels into the street scene, object edges into pictures, aerial photos into maps and so on [59, 67, 169, 183]. Others integrated GANs to improve image quality in the classic image processing tasks, e.g., super-resolution [72], image de-raining [175], and image in-painting [110] etc.

However, simply applying generative networks to image synthesis can not perfectly solve all the practical problems of visual synthesis. Most of the existing methods use the neural network as a black box, which simply trains the model to

output the image as a whole. For local image synthesis problems, they can hardly achieve satisfying performance as they cannot control the generating process. Also, it is inefficient to deal with the multi-domain image synthesis problem. In multi-domain image synthesis tasks, models are required to generate different styles or attributes of images. Existing approaches have limited scalability and robustness in handling more than two domains, since different models should be built independently for every image domain.

Video synthesis refers to generating a sequence of consecutive video frames, which is a widely explored problem. Accurately generating frames is an important technique in video coding, video completion, robotics, autonomous driving and intelligent agents that interact with their environment. The specific tasks of video generation differ in the type of the provided signal [26]. One side of the task deals with unconditional video synthesis where the task is to generate any videos so long as they follow the training distribution. Another side of the problem is occupied by strongly-conditioned models, including generation conditioned on another video for object transfer [10], per-frame segmentation masks [152], or pose information [164, 165]. In the middle ground there are tasks that are more structured than unconditional generation problems, and yet are more challenging from a modeling perspective than the strongly conditional generation (which gets a lot of information about the generated video through its input). Future video prediction refers to the generation of subsequent video given initial frames. These problems differ in several aspects, but share a common requirement of realistic temporal dynamics.

In video generating problems, temporal consistency and dynamics are important factors that should be considered. Generating each frame independently cannot guarantee temporal consistency and would lead to video flickering. To generate consecutive video-frames, existing methods often execute in a recursive manner by taking the generated frames as observations to generate subsequent frames. However, they usually only produce high-quality frames for the first few steps. The generating frames would then dramatically degrade.

1.2.3 Challenges

The challenges of visual synthesis mainly are summarized as the following five categories:

Training Instability for generative adversarial networks. Though GAN-based models have achieved promising results, they are often difficult to train. If the data and the model distribution do not substantially overlap, the generator will encounter gradient vanishing. On the other hand, GAN models often suffer from mode collapse issues, in which a trained model assigns all its probability to a small region in the target space [125, 134] and often choose to generate from very few modes.

Interpretation and controllability for local image synthesis. Existing methods use a neural network as the generator and simply generate the output as a black-box process. Since they treat the generator as a black box, it is hard for users to control the image synthesis process and manipulate the images as their desire, such as the specific position, semantic object, and context. For instance, in the image-to-image transformation problem, users might hope to manipulate certain objects in the reference images.

Inefficiency for multi-domain image synthesis. Existing approaches are designed for generating a specific domain or category output [59, 183]. They are inefficient and ineffective in handling more than two domains since multiple models should be built independently for different image domains. It is time-consumed to train multiple models for multi-domain synthesis and costs lots of storage space for saving models. Meanwhile, they are ineffective even though there exist global features that can be learned from images of all domains, separately training generators cannot fully utilize the entire training data.

Quality for long-term video frame synthesis. Video-frames synthesis is another problem of visual synthesis, which refers to generating a sequence of consecutive video frames. How to extend the image synthesis to video remains a challenging problem. In the video generating problem, temporal consistency is an important factor that should be considered. Generating each video-frame independently would lead to flickering. To generate temporal consistent video-frames, existing methods often execute in a recursive manner by taking the gen-

erated frames as observations. However, they usually only produce high-quality predictions for the first few steps. The generated frames would then dramatically degrade, and could even lead to totally missing the video context.

1.3 Contributions

This thesis explicitly addresses the challenges of visual data synthesis and transformation. Advanced image synthesis methods are devised to improve image quality, controllability and efficiency. Also, the long-term video synthesis is investigated by considering temporal consistency via a bi-direction generative model. The detailed contributions contained in the thesis are summarized as follows:

- An attention generative model is present for local image synthesis, which is able to only manipulate and generate object on the certain region of images. The model is built by decomposing the generative network into two separate networks, each of which is only dedicated to one particular subtask: to detect the objects of interests and to convert the object from one object to another. The attention network predicts spatial attention maps of images, and the transformation network focuses on translating objects. The attention map produced by the attention network is encouraged to be sparse so that major attention can be paid to objects of interest. No matter before or after transformation, attention maps should remain constant. In addition, a novel perceptual loss is introduced to improve the quality of transformed images in the high-level feature space.

- A novel gated generative model is proposed for multi-domain image synthesis. With the proposed model, multiple styles of images can be generated in a single model. The generative network is decomposed into three modules: an encoder, a gated transformer, and a decoder. Different styles can be achieved by passing input images through different branches of the gated transformers. We introduce an auto-encoder reconstruction loss for our generative model that is capable of improving training stability. The auto-encoder reconstruction loss is achieved by combining the encoder and

decoder module. An auxiliary classifier is used to recognize the style categories of transformed images, thereby helping the generative network generates images in multiple styles.

- A temporal generative model is proposed via criticization and retrospection for long-term video synthesis. The introduced retrospection process is designed to look back on what has been learned from the past and rectify the prediction deficiencies. To this end, a retrospection network is built to reconstruct the past frames given the currently predicted frames. On the other hand, an auxiliary route is built by reversing the flow of time and executing a similar retrospection. These two routes interact with each other to boost the performance of retrospection network and enhance the understanding of dynamics across frames, especially for the long-term horizon. Extensive experiments on natural video datasets demonstrate the advantage of introducing retrospection processes in long-term video prediction.

1.4 Organization

This thesis consists of six chapters as shown in Figure 1.3. We briefly summarize the main topics in each chapter as follows:

Chapter 1 introduces the background of visual synthesis and transformation. In the following of an overview of the main technique, applications and challenges of visual synthesis are presented. We summarize the main contributions of this thesis.

Chapter 2 reviews state-of-the-art visual synthesis algorithms and applications. We first introduce and formalize the related deep generative models. We then review the image synthesis and image-to-image transformation applications. Lastly, we introduce video synthesis methods and applications.

Chapter 3 demonstrates a novel attention-aware generative model to improve the quality of output image. We present our motivation to decompose the generation network as an attention network and a transformation network. We show our observations regarding the limitations of existing image transformation methods using an end-to-end generative network. We then describe the background of the attention model as well as generative networks incorporating the attention

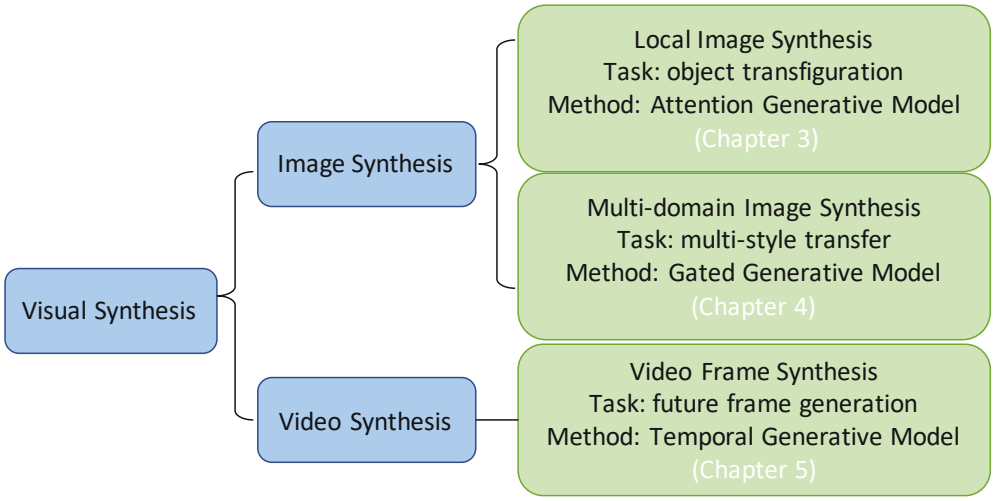

Figure 1.3: The organization overview of this thesis.

mechanism. In the following, we introduce the formula of the proposed method as well as the implementation detail. In the end, extensive experimental results demonstrate the necessity of investigating attention in image transformation, and that the proposed algorithm can learn accurate attention prediction and improve the quality of generated images.

Chapter 4 presents an effective and efficient generative model for multi-domain image synthesis. The proposed gated generative network which consists of three modules: encoder, gated-transformer, and decoder. The gate controls which transformer is connected to the encoder and decoder, so that users can switch gate to choose between different style. In the following, we introduce a novel auto-encoder reconstruction loss which is able to increase training stability and avoid mode collapse. Then we introduce our implementation details including training strategy and network configurations. Specially, we present the visualization of learned features in the gated transformer. The visualization reveals that each transformed learns style representations e.g., color, stroke, etc. In the end, we show extensive experimental results to demonstrate the stability and effectiveness of the proposed model for multi-style transfer.

Chapter 5 presents a robust long-term video synthesis model for video frame prediction. We show our observation that the limitation of existing methods usually produces high-quality predictions for the first few steps and the quality

of prediction would decrease dramatically. We present our motivation that a qualified generator would also be able to predict videos in a backward manner. We then introduce our novel long-term video synthesis which consists of a prediction process and retrospection process. After predicting in a few time steps, we pause the prediction process and look back to rectify the prediction deficiencies in a backward manner. Then we discuss the detailed implementations of updating the predict network and retrospection network as well as the network architecture. Extensive experiments on natural video datasets demonstrate the advantage of the introduced retrospection process in long-term video prediction.

Chapter 6 concludes the thesis and sheds the light on future research work in visual data synthesis and transformation.

Chapter 2

Literature Review

2.1 Generative Models

Generative models are one of the fundamental branches in statistics and machine learning, which are widely used for computer vision and visual synthesis tasks. In statistics, the goal of generative models is to represent the estimated probability distributions of real-world data, such as natural images. Generative models can randomly generate unobservable values based on a joint probability distribution. For machine learning researchers, generative models are generally used to generate values of any variable in the model [71]. Generative models have been in the forefront of deep unsupervised learning for the last decade, as they offer an efficient way to analyze and understand unlabeled data.

Generative models can be divided into two categories: explicit generative models and implicit generative models. In explicit generative models, the data distributions are directly defined by an explicit density function $p_{model}(x|\theta)$ for each example of training so that the maximum likelihood can be straight calculated. Most explicit generative models are based on the principle of maximum likelihood. The maximum likelihood estimation is a method of estimating the parameters of a statistical model given observations, by finding the parameter values that maximize the likelihood of making the observations given the parameters. While the model parameters are optimized by maximizing the sum of the likelihood of all training data, these generative models can gradually understand

and represent data distributions. However, it is difficult to maintain computational tractability, especially when dealing with high-dimensional complex data distributions. Implicit generative models are devised to estimate the probability distributions without explicitly defining the density functions. Some of these implicit generative models based on drawing samples from p_{model} define a Markov chain transition operator that must be run several times to obtain a sample from the model. Markov chains easily fail to sample from high-dimensional spaces and usually increase the computational cost of using the generative model.

Over the past decade, there has been a substantial growth of deep generative models [71]. Since deep belief networks (DBNs) [54] were proposed in 2006, a significant increase of study for deep generative models has emerged. Deep generative models build generative models with the flexibility and scalable learning of deep neural networks. Comparing with the shallow architectures, models with deep architectures have better performance in generalization and expression. Nowadays, deep generative models have been applied to a series of machine learning applications and received remarkable effects, such as object detection [42], debarring [30], and image generation [33].

Generative adversarial networks (GANs) are one of the approaches of implicit generative models that leverage deep neural networks. GANs are designed to avoid problems of traditional generative models as well as take advantage of deep architectures. GAN-based methods have been widely applied to image generation [23, 39, 105, 173], image editing [59, 149, 150, 183], video prediction [90, 148], and many other tasks [92, 143, 178].

2.2 Generative Adversarial Networks

Generative adversarial networks (GANs) [44] are one of the branches of deep generative models. GANs consist of a generator and a discriminator, and perform as a min-max game. The illustration for GANs is shown in Figure 2.1. While using the generator to synthesize semantic meaning data from standard signal distributions, the discriminator is trained to distinguish *real* samples from *fake* samples synthesized by the generator. As an adversary, the generator aims to deceive the discriminator by producing ever more realistic samples. The training

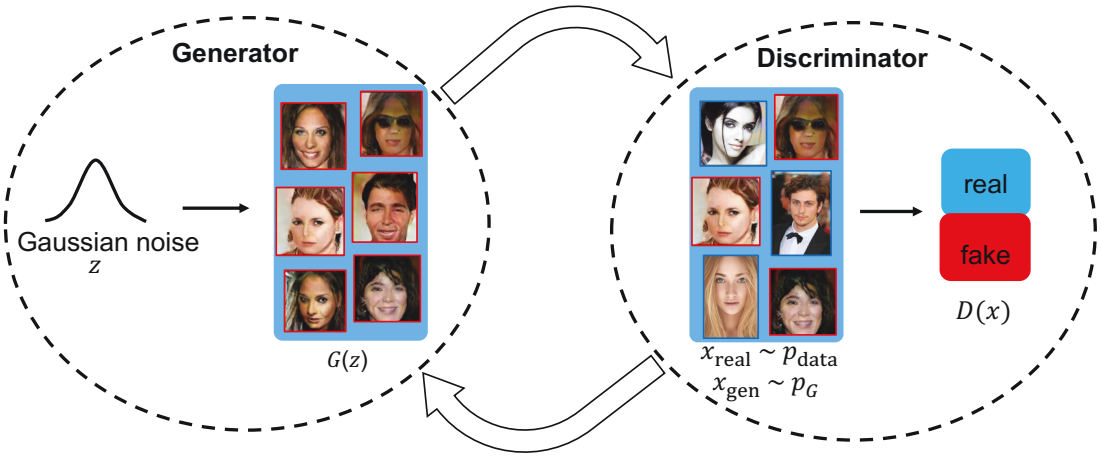

Figure 2.1: The illustration of the adversarial training procedure.

procedure continues until the generator wins the adversarial game; that is, the discriminator cannot make a better decision than randomly guessing whether a particular sample is fake or real. In the implementation, the generator and discriminator can be constructed by deep neural networks.

To learn the generator's distribution p_g over data x, GANs define a prior on input noise variables $p_z(z)$, then represent a mapping to data space as $G(z;\theta_g)$, where G is a differentiable function represented by a neural network with parameters θ_g. Another neural network $D(x;\theta_d)$ is defined to output the probability that the input sample x comes from the real data p_{data} rather than p_g. The discriminator D is trained to maximize the probability of assigning the correct label to both training examples and samples from G. Simultaneously, the generator G is trained to minimize $\log(1 - D(G(z)))$. D and G play a two-player min-max game with value function $V(G, D)$ as follow:

$$\min_G \max_D V(D,G) = \mathbb{E}_{x\sim p_{data}(x)}\left[\log D(x)\right] + \mathbb{E}_{z\sim p_z(z)}\left[\log\left(1 - D(G(z))\right)\right]. \quad (2.1)$$

Generative adversarial nets are trained by simultaneously updating the generator $G(z)$ and the discriminator $D(\cdot)$. If G and D have enough capacity, they will reach the optimum, more specifically a Nash equilibrium [115]. In that point, both cannot improve as the generator is able to output realistic data because $p_g = p_{data}$. The discriminator is unable to differentiate between the two distribution

as $D(x) = \frac{1}{2}$.

Though GANs can avoid many of the inherent difficulties of previous generative models, they still face some challenges. First, GAN models are notoriously difficult to train. Since training the generator is equivalent to minimizing the Jensen-Shannon divergence between the data distribution and the generated distribution, which easily causes the vanishing gradient when the two distributions have no substantially overlap. Second, GAN models often suffer from mode collapse issues, in which the trained model assigns all its probability to a small region in the target space [125, 134] and often produce limited varieties of samples. Third, the selections of network architectures and training parameters are sensitive for training stability, and unsuitable settings (e.g., batch size, learning rate, and updating steps) will significantly reduce the training stability and generative performance [68, 78].

A lot of works have investigated to improve the training stability and image quality of GANs. To solve the gradient vanishing problem in original GANs, a non-saturating heuristic objective (i.e., '$-\log D$ trick') [123] replaced the minimax objective function in Equation 2.1. DCGAN was devised by [113] and proposed several heuristic tricks (e.g., feature matching, one-side label smoothing, virtual batch normalization) to improve training stability. The classical GANs actually aims to minimize Jensen-Shannon distance between real data distribution and model distribution. The classical GANs easily cause gradient vanishing as JS distance does not provide gradient when the two distributions do not substantially overlap. Recently, Wasserstein GAN (WGAN) [4] was proposed to minimize Wasserstein distance between real data distribution and model distribution. WGAN reduces training instability since Wasserstein distance is continuous everywhere and differentiable almost everywhere under only minimal assumptions.

2.3 Image Synthesis

Image synthesis is the process of creating new images from some forms of image description. The kinds of images description include: test patterns, noise from specific distribution, semantic annotations and images. The classical GANs [44] were devised to generate data from noise. GANs were also applied in condi-

tional settings, e.g., discrete labels, text, and images. The simplest conditional GANs [102] generate MNIST digits conditioned on class labels. [118] proposed the Generative Adversarial What-Where Network (GAWWN), which synthesizes images given instructions describing what content to draw in which location. [94] proposed a pose guided person generation network that allows synthesizing person images in arbitrary poses, based on an image of that person and a pose. [124] proposed an architecture that is conditioned on sketched boundaries and sparse color strokes to generate realistic cars, bedrooms, or faces.

Image-to-image transformation is one type of image synthesis tasks, which aims to translate images from a source domain to another target domain, e.g., greyscale to color images, and image to semantic label. Previous works of the image-conditional models have tackled inpainting [110], image super-resolution [31], image manipulation guided by user constraints [182], video frame prediction [100], future state prediction [179], low-dynamic-range (LDR) image to a high-dynamic-range (HDR) image transfer [106] and style transfer [77]. Each of these methods was tailored for a specific application.

Image-to-image tranformation can be solved in a general framework by adopting the conditional generative model. Recently, [59] solved image conditional generation problems in a general framework by treating them as an image-to-image translation problem. [59] used conditional GANs to learn a mapping from input to output images, with the supervision of the ground truths in the target domain. In the absence of paired examples settings, a series of unsupervised image-to-image translation works have emerged by combining classical adversarial training with carefully designed constraints, e.g., circularity constraint [66, 169, 183], f-consistency constraint [140], and distance constraints [12]. Although there is no paired data, these constraints establish the connections between two domains so that meaningful analogs are obtained. Circularity constraint [66, 169, 183] requires a sample from one domain to the other that can be mapped back to produce the original sample. f-consistency requires both input and output in each domain should be consistent with each other in intermediate space of a neural network. [12] learned the image translation mapping in a one-sided unsupervised way by enforcing a high cross-domain correlation between matching pairwise distances computed in source and target domains.

Existing methods used a neural network as a generator and simply generated the output as an end-to-end process. Since they treated the generator as a black-box, it was hard for users to control the image synthesis process and manipulate the images as their desire, such as specific the position, semantic object, and context. For instance, in the image-to-image transformation problem, users might hope to manipulate certain objects in the reference images.

2.4 Video Synthesis

Video synthesis refers to generating a sequence of consecutive video frames, which is a widely explored problem. Accurately generating frames is an important technique in video coding, video completion, robotics, autonomous driving and intelligent agents that interact with their environment. Similar to image synthesis, the specific tasks of video generation differ in the type of conditioning signal provided [26]. One side of the task refers to unconditional video synthesis where the task is to generate any video so long as following the training distribution. Another side of problem is occupied by strongly-conditioned models, including generation conditioned on another video for object transfer [10], per-frame segmentation masks [152], or pose information [164,165]. In the middle ground there are tasks that are more structured than an unconditional generation, and yet are more challenging from a modeling perspective than a strongly conditional generation (which gets a lot of information about the generated video through its input).

Future video prediction is concerned with generating subsequent video given initial frames. The problems differ in several aspects, but share a common requirement of needing to generate realistic temporal dynamics. Different from image generation, in video generating problem, temporal consistency and dynamics are important factors that should be considered. Generating each frame independently cannot guarantee temporal consistency and would lead to video flickering. To generate consecutive video-frames, existing methods often execute in a recursive manner by taking the generated frames as observations to generate subsequent frames. One popular hypothesis is that a video sequence could be decomposed as content and motion. By independently modeling the motion

and content, MCNET [144] predicted the next frame by combining the predicted motion feature and the extracted content feature. DRNET [29] designed an adversarial training strategy to disentangle the motion and content representations. Another assumption is that the outcome of an event is stochastic as a consequence of the latent events, so different possible future for each sample of its latent variables can be predicted [7]. [159] incorporated a two steam generation architecture to deal with high frequency and low-frequency video content separately so as to output structured prediction. [155] produced a mask that outlines the predicted foreground object to achieve a better quality of prediction. However, they usually only produce high quality frames for the first few steps. The generating frames would then dramatically degrade. To generate long-term video prediction is still an challenging problem.

Chapter 3

Attention Generative Model for Local Image Synthesis

In this chapter, we propose an attention generative model for object transfiguration. Object transfiguration is a typical problem for local image synthesis. The generative network in classical GANs for object transfiguration often undertakes a dual responsibility: to detect the objects of interests and to convert the object from source domain to another domain. A simple neural network is hard to control the region of the transformed part and often leads to low generation quality. In our method, we decompose the generative network into two separate networks, each of which is only dedicated to one particular sub-task. The attention network predicts spatial attention maps of images, and the transformation network focuses on translating objects. Attention maps produced by the attention network are encouraged to be sparse so that major attention can be paid on objects of interest. No matter before or after object transfiguration, attention maps should remain constant. In addition, the attention network can receive more instructions, given the available segmentation annotation s of images. On the other hand, previous works only transfer the low-level information by adopting PatchGAN whose discriminator classifies the patches of images. To improve the quality of transformed images in the high-level feature space, we introduce a perceptual loss by minimizing the feature distance between the translations and overall samples of the target domain. Experimental results demonstrate the necessity of investigating

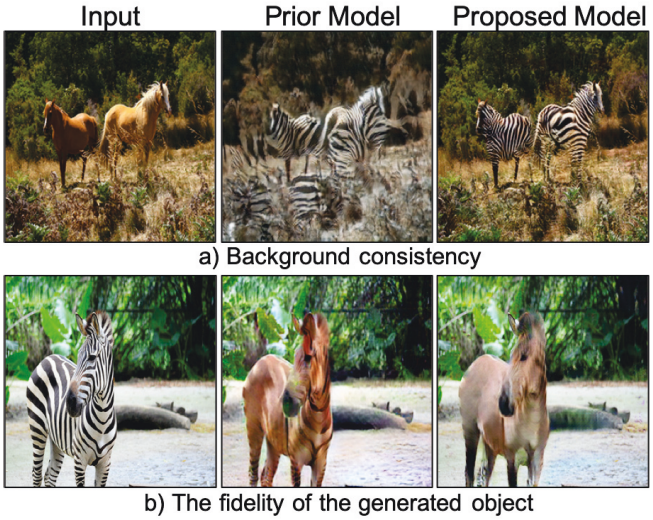

Figure 3.1: Comparisons of object transfiguration examples. From left to right: the input images, the transformed results of the prior model, and the transformed results of our proposed model. a) An example of horse → zebra; b) An example of zebra → horse.

attention in object transfiguration, and that the proposed algorithm can learn accurate attention and improve the quality of generated images.

3.1 Introduction

The task of image-to-image translation aims to translate images from a source domain to another target domain. Applications include greyscale to color and image to semantic labels. Many approaches on image-to-image translation have been produced in the supervised setting, where ground truths in the target domain should be available. [89] learned a parametric translation function using CNNs by minimizing the discrepancy between generated images and the corresponding target images. [59] used conditional GANs to learn a mapping from input to output images. Similar ideas have been applied to various tasks such as generating photographs from sketches or semantic layout [63, 124], and image super-resolution [31]. However, it requires data from each domain to be paired or under alignment, which restricts applications and may not even be possible for some domains. To achieve image-to-image translation in the absence of

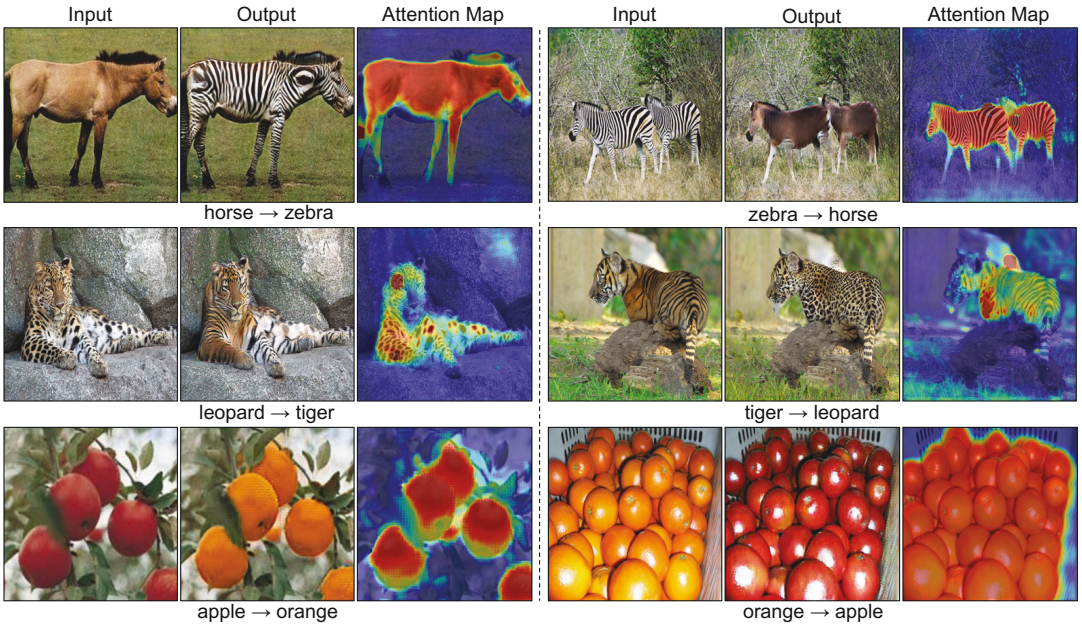

Figure 3.2: Results of object transfiguration on different tasks: horse ↔ zebra, leopard ↔ tiger and apple ↔ orange. In each case, the first image is the original image, the second image is the synthesized image, and the third image is the predicted attention map. Our proposed model only manipulates the attention parts of the image and preserves the background consistency.

paired examples, a series of works has emerged by combining classical adversarial training [44] with different carefully designed constraints, e.g., circularity constraint [66, 169, 183], f-consistency constraint [140], and distance constraints [12]. Although there is no paired data, these constraints can establish the connections between two domains so that meaningful analogs are obtained. Circularity constraint [66, 169, 183] requires a sample from one domain to the other that can be mapped back to produce the original sample. f-consistency requires both input and output in each domain should be consistent with each other in intermediate space of a neural network. [12] learned the image translation mapping in a one-sided unsupervised way by enforcing a high cross-domain correlation between matching pairwise distances computed in source and target domains.

Object transfiguration is a special task in the image-to-image translation problem. Instead of taking the image as a whole to accomplish the transformation, object transfiguration aims to transform a particular type of object in an image to

another type of object without influencing the background regions. For example, in the top line of Figure 3.1, horses in the image are transformed into zebras, and zebras are transformed into horses, but the grassland and the trees are expected to be constant. Existing methods [12, 183] used to tackle object transfiguration as a general image-to-image task, without investigating unique insights into the problem. In such a one-shot generation, a generative network actually takes two distinct roles: detecting the region of interests and converting the object from the source domain to the target domain. However, incorporating these two functionalities in a single network would confuse the aims of the generative network. In iterations, it could be unclear whether the generative network should improve its detection of the objects of interests or boost its transfiguration of the objects. The quality of generated images is often seriously influenced as a result, e.g. some background regions might be taken into transformation by mistake.

On the other hand, to generate realistic output, the existing works adopt a Markovian discriminator in the adversarial training (termed as PatchGAN). PatchGAN classifies whether the image patches are real or fake. Such a discriminator models the image as a Markov random field, assuming independence between pixels separated by more than a patch diameter, which can be viewed as a texture or style loss. As a result, they only transfer low-level information which leads to low quality of the transformed images in the object transfiguration problem. For example, in Figure 3.1 (b) for zebra $\rightarrow$ horse, we can still find zebra stripes in the transformed horse in the prior model while the brown color is rendered on the horse body. Recently, in image-to-image transformation, many approaches have incorporated high-level information for suppressing artifacts [175] and improving the perceptual quality of the output images [72, 177]. However, the aforementioned techniques are used by optimizing the discrepancy between the high-level features of the output and the ground-truth images. In the absence of the paired images, the existing works cannot be applied in the unsupervised image-to-image transformation.

To disentangle the image background from object regions, we propose an attention-GAN algorithm for the object transfiguration problem. The generative network has been factorized as two separated networks: an attention network to predict where the attention should be paid, and a transformation network that

actually carries out the transformation of objects. A sparse constraint is applied over the attention map so that limited attention energy can be focused on regions of priority rather than spread on the whole image at random. A layered operation is adopted to finalize the generated images by combining the transformed objects and the original background regions with the help of the learned sparse attention mask. A discriminative network is employed to distinguish real images from these synthesized images, while attention network and transformation network cooperate to generate synthesized images that can fool the discriminative network. Cycle-consistent loss [66, 169, 183] is adopted to handle unpaired data. Figure 3.2 shows some examples of results by our model. It can be observed that our model succeeds in predicting the attention map and only transforms the foreground while preserving the background consistency. Moreover, if segmentation results of images are available, the attention network can be learned in a supervised manner and the performance of the proposed algorithm can be improved accordingly.

To improve the fidelity of output images in high-level features, we introduce the perceptual loss by considering the transformed images and the overall samples in the target domain. With the idea that the transformed images in the target domain should have a similar feature of images in the target domain. The perceptual loss aims to minimize the expectation of distance between the transformed image and the real samples of the target domains. Experimental results on three object transfiguration tasks, i.e. horse $\leftrightarrow$ zebra, tiger $\leftrightarrow$ leopard, and apple $\leftrightarrow$ orange, suggest the advantages of investigating attention in object transfiguration, and the quantitative and the qualitative performance improvement of the proposed algorithm over state-of-the-art methods.

The rest of this chapter is organized as follows. In Section 3.2, we summarize related works. We briefly review the prior algorithm that achieves unpaired image-to-image translation in Section 3.3. We then introduce the proposed method in Section 3.4 and provide implementation details in Section 3.5. Experimental results of the proposed method and the comparison with existing methods are illustrated in Section 3.6. Section 3.7 concludes our work of this chapter.

3.2 Related Work

In this section, we first review the generative adversarial networks (GANs) and its applications, and then we introduce the representative methods on image-to-image translation. Last, we briefly summarize the related works of the attention mechanism in the deep networks.

3.2.1 Generative Adversarial Networks

Generative adversarial networks (GANs) [44] is an implicit generative model, implemented by a two-player game: a generator and a discriminator. The generator aims to generate realistic images to fool the discriminator, while the discriminator aims to classify whether the images are real or fake. Although GANs typically produce promising results, they are often very hard to train and output low-resolution results. To make them stable to train, [113] proposes a set of constraints on the architectural topology of convolutional GANs, e.g. batch normalization, fractional-strided convolutions, etc. To overcome the well-known mode collapse and gradient vanishing issue of GAN, a lot of training objectives have been developed, such as WGAN [4], feature matching loss [123], least squares loss in LSGAN [98]. [14] derived a way of controlling the trade-off between image diversity and visual quality. In this work, followed [183], we adopt the LSGAN [98] which performs stably during training and generates high-quality results.

On the other hand, a series of multi-stage generative models have been proposed to generate more realistic images [56, 70, 166]. [70] proposes a composite generative adversarial network (CGAN), that disentangles complicated factors of images with multiple generators in which each generator generates some part of the image. The layered recursive GANs [166] learned to generate image background and foregrounds separately and recursively, and stitch the foregrounds on the background in a contextually relevant manner to produce a complete natural image. In contrast, in our work, we explicitly decompose the generative network into two separated networks, an attention network, and a transformation network. The introduced attention network is able to disentangle the foreground and background of images so that the transformation network could concentrate on improving the transformation quality.

3.2.2 Image-to-Image Transformation

GANs have shown great success on a variety of conditional models, e.g., image generation [174], image-to-image translation, text-to-image generation. Different from the original GANs, which generate images from the latent space variable, conditional GANs synthesize images based on the input information. The simplest conditional GANs [102] generated MNIST digits conditioned on class labels. [118] proposed the Generative Adversarial What-Where Network (GAWWN), which synthesizes images given instructions describing what content to draw in which location. [94] proposed a pose guided person generation network that allows synthesizing person images in arbitrary poses, based on an image of that person and a pose. [124] proposed an architecture that is conditioned on sketched boundaries and sparse color strokes to generate realistic cars, bedrooms, or faces.

In image-to-image translation problem, [59] investigated conditional adversarial networks as a general-purpose solution, such as sketch to photo, map to aerial photo, day to night, etc. After this, CycleGAN [183], DiscoGAN [66], and DualGAN [169] introduced cycle-consist loss to achieve unpaired image to image translation problem and also conducted experiments on semantic translation (e.g. horse to zebra and apple to orange). Some unsupervised translation approaches assume the existence of a shared latent space between the source and target domains. Coupled GAN (CoGAN) [87] learned an estimate of the joint data-generating distribution using samples from the marginals, by enforcing source and target discriminators and generators to share parameters in low-level layers. After that, [86] built an unsupervised image-to-image translation network (UNIT) based upon Coupled GAN by assuming the existence of a shared low-dimensional latent space between the source and target domains. Once the image is mapped to its latent representation, then a generator decodes it into its target domain version. [83] proposed a mask-conditional contrast-GAN architecture that realized the attentive semantic manipulation by conditioning on masks of object instances. However, contrast-GAN required segmentation mask annotations both in the training phase and test phase, which is hard to obtain in practice. In our work, we take a further step towards manipulating an object by sepa-

rating the background and foreground in an unsupervised way. [101]'s unsupervised attention-guided image-to-image translation (UAIT) was a contemporaneous work that shares our goal of learning an attention map for image translation. UAIT [101] trained the model by two-step. They first train the overall network in the first 30 epochs, and then stop the training of the attention networks. In contrast, we proposed the attention sparse loss to force the attention only focus on the foreground, so that the overall model could be trained simultaneously.

3.2.3 Attention Model in Networks

Motivated by human attention mechanism theories [120], attention mechanism has been successfully introduced in computer vision and natural language processing tasks, e.g. image classification [104, 157, 179], image generation [156], image captioning [160], and visual question answering [158]. Rather than compressing an entire image or sequence into a static representation, attention allows the model to focus on the most relevant part of images or features as needed. [104] learned an attention model that adaptively selects image regions for processing. [8] proposed an attention model that softly weights the importance of input words in a source sentence when predicting a target word for machine translation. Following this, [160] and [167] used attention models for image captioning and video captioning respectively. The model automatically learns to fix its gaze on salient objects while generates the corresponding words in the output sequence. In visual question answering, [158] used the question to choose relevant regions of the images for computing the answer.

The approach to obtaining the attention maps in deep convolutional neural networks is an open problem. The gradient-based way defined attention as gradient w.r.t. input [130, 133], which can be viewed as an input sensitivity map, i.e., attention at an input spatial location encodes how sensitive the output prediction is. Another approach obtained the attention map by making use of class activation maps [130]. They removed the top average-pooling layer and converted the linear classification layer into a convolutional layer, producing attention maps per each class. In visual question answering, the attention mechanism [158] in the network selected certain parts of the neuron activations, which stored information from different spatial regions of the image.

In image generation, [46] proposed a recurrent model to generate images with the attention mechanism that allows the generator to focus on smaller regions of an input image and generate an image a few pixels at a time. [46] also modified their approach to generate two digits per image, and observed that their attention mechanism ensured that the model focused on generating one number at a time. Followed by this, [65] introduced adversarial training in the recursive attention generative network, and generates more realistic images on the MNIST dataset. In our work, rather than sequential attention generation, the attention networks learn the location of the object in one step.

3.3 Preliminaries

In the task of unsupervised image-to-image translation, we have two domains X and Y with training samples $\{x_i\}_i^{N_X} \in X$ and $\{y_j\}_j^{N_Y} \in Y$. The goal is to learn mapping from one domain to the other $\mathcal{G} : X \to Y$, (e.g. horse→zebra). The discriminator D_Y aims to distinguish real image y from translated images $\mathcal{G}(x)$. On the contrary, the mapping function $\mathcal{G}$ tries to generate images $\mathcal{G}(x)$ that looks similar to images in Y domain to fool the discriminator. The objective of *adversarial loss* in LSGAN [98] is expressed as:

$$\mathcal{L}_{GAN}(\mathcal{G}, D_Y, X, Y) = \mathbb{E}_{y \in Y}\left[D_Y^2(y)\right] + \mathbb{E}_{x \in X}\left[(D_Y(\mathcal{G}(x)) - 1)^2\right], \tag{3.1}$$

The mapping function $\mathcal{F} : Y \to X$, in the same way, tries to fool the discriminator D_X:

$$\mathcal{L}_{GAN}(\mathcal{F}, D_X, X, Y) = \mathbb{E}_{x \in X}\left[D_X^2(x)\right] + \mathbb{E}_{y \in Y}\left[(D_X(\mathcal{F}(y)) - 1)^2\right]. \tag{3.2}$$

The discriminators D_X and D_Y try to maximize the loss while mapping functions $\mathcal{G}$ and $\mathcal{F}$ try to minimize the loss. However, a network of sufficient capacity can map the set of input images to any random permutation of images in the target domain. To guarantee that the learned function maps an individual input x to a desired output y, the *cycle consistency loss* is proposed to measure the discrepancy occurred when the translated image is brought back to the original

image space:

$$\mathcal{L}_{cyc}(\mathcal{G},\mathcal{F}) = \mathbb{E}_{x\in X}\left[\|\mathcal{F}(\mathcal{G}(x)) - x\|_1\right] + \mathbb{E}_{y\in Y}\left[\|\mathcal{G}(\mathcal{F}(y)) - y\|_1\right]. \tag{3.3}$$

Taking advantages of adversarial loss and cycle consistency loss, the model achieves a one-to-one correspondence mapping, and discovers the cross-domain relation [66]. The full objective is:

$$\mathcal{L}(\mathcal{G},\mathcal{F},D_X,D_Y) = \mathcal{L}_{GAN}(\mathcal{G},D_Y,X,Y)+\mathcal{L}_{GAN}(\mathcal{F},D_X,Y,X)+\lambda\mathcal{L}_{cyc}(\mathcal{G},\mathcal{F}), \tag{3.4}$$

where λ controls the relative importance of the two objectives. However, the generative mapping functions $\mathcal{G}$ and $\mathcal{F}$ actually takes a dual responsibility for object transfiguration: to detect the objects of interest and to transfigure the object, which confuses the aims of the generative network.

On the other hand, we notice that the model can be viewed as two "autoencoders": $\mathcal{F}\circ\mathcal{G} : X \to X$ and $\mathcal{G}\circ\mathcal{F} : Y \to Y$, where the translated image $\mathcal{G}(x)$ and $\mathcal{F}(y)$ can be viewed as intermediate representations trained by adversarial loss. In object transfiguration task, the generative mappings $\mathcal{G}$ and $\mathcal{F}$ are trained to generate objects to fool the discriminator. Therefore, the image background can be coded as any representation so long as it can be decoded back to the original, which does not guarantee background consistency before and after transformation. As a result, the proposed Attention-GAN that decomposes the generative network into two separated network: an attention network to predict the object of interests and a transformation network focuses on transforming object.

3.4 Attention Generative Model

In this section, we introduce the proposed attention generative model for object transfiguration. The overview of the proposed model is illustrated in Figure 3.3. The grey dotted frame represents the generator of our model. The generator model consists of two players: an attention network, a transformation network. The attention network predicts the region of interest from the original image. The transformation network focuses on transforming the object from one domain

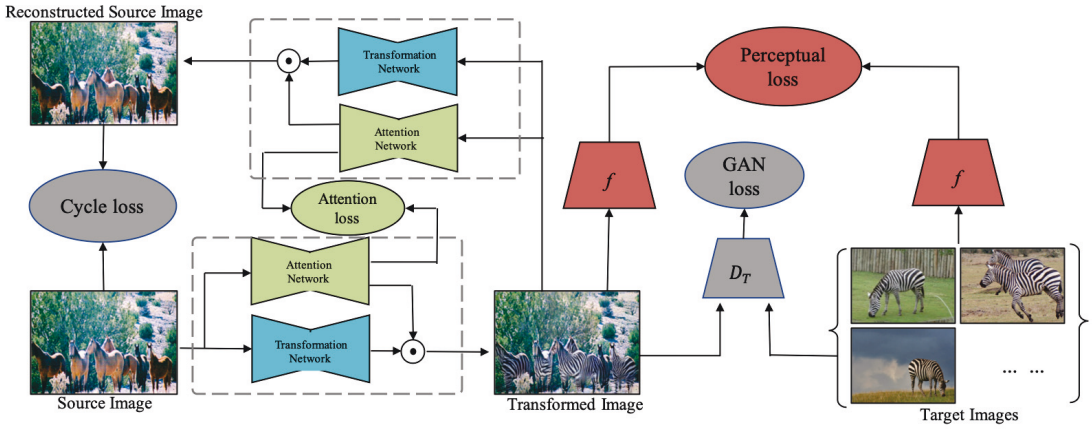

Figure 3.3: The architecture of the proposed method. For clarity the target cycle is omitted. The grey dotted frame represents the generator of our model, which consists of a transformation network and an attention network. Detailed illustration of the generator is shown in Figure 3.4. The perceptual loss minimizes the distance in feature space between the transformed image and the overall images of the target domain.

to the other. The resulting image is, therefore, a combination of the transformed object and the background of the original image with a layered operator. A detailed formulation of the generative model is described in Section 3.4.1. The Attention loss is introduced in order to learn an accuracy attention map of the object, which is discussed in Section 3.4.2. To improve the quality of the transformed object, we proposed the perceptual loss (red) by optimizing the distance between the transformed image and overall datasets of target images in feature space (see details in Section 3.4.3). The cycle loss and the GAN loss (grey) are adopted as described in Section 3.3. For notation simplicity, we only show the forward process that transforms images from domain X to domain Y, and the backward process from domain Y back to the domain X can be easily obtained in the similar approach.

3.4.1 Generative Model

The detailed generator architecture is shown in Figure 3.4. Given an input image x in the domain X, the attention network A_X outputs a spatial score map $A_X(x)$, whose size is the same as the original image x. The element value of the score map

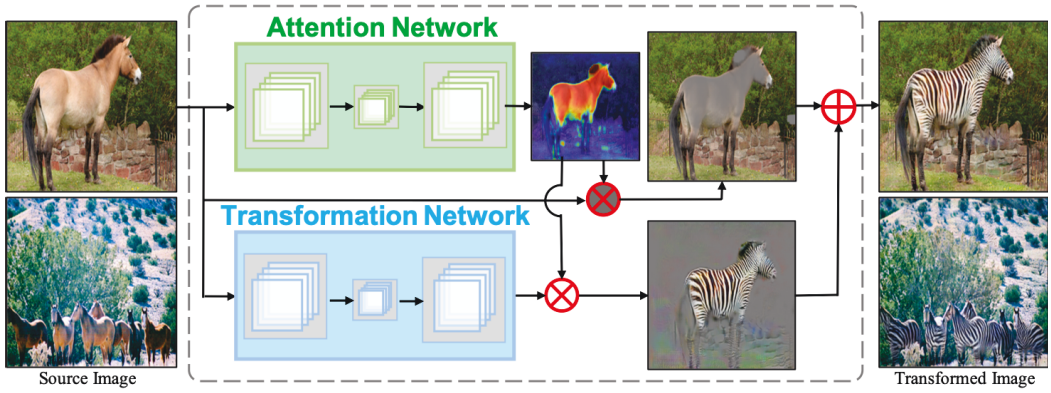

Figure 3.4: The generator of Attention-GAN transforms the source image from one class to another. The attention network predicts the attention maps. The transformation network synthesizes the target object. A layered operation is applied to the background and transformed images to output the resulting image.

is from 0 to 1. The attention network assigns higher scores of visual attention to the region of interest while suppressing background. In another branch, the transformation network T outputs the transformed image $T(x)$ that looks similar to those in the target domain Y. Then we adopt a layered operation to construct the final image. Given transformed region $A_X(x)$, a transformed image $T_X(x)$ and image background from original image x are combined as:

$$\mathcal{G}(x) \equiv A_X(x) \odot T_X(x) + (1 - A_X(x)) \odot x, \tag{3.5}$$

where $\odot$ denotes the element-wise multiplication operator. Another mapping function $\mathcal{F}$ is introduced to bring transformed images $\mathcal{G}(x)$ back to the original space $\mathcal{F}(\mathcal{G}(x)) \approx x$. The mapping from an image y in target domain Y to the source domain follows:

$$\mathcal{F}(y) \equiv A_Y(y) \odot T_Y(y) + (1 - A_Y(y)) \odot y. \tag{3.6}$$

Followed by Section 3.3, the *adversarial loss* (Equations (3.1) and (3.2)) and the *cycle consistency loss* (Equation (3.3)) are introduced to learn the overall mappings $\mathcal{G}$ and $\mathcal{F}$. In classical GANs [66, 169, 183], the generative mapping $\mathcal{G}$ transforms the whole image to target domain and then the generative mapping $\mathcal{F}$

is required to bring the transformed image back to original image $\mathcal{F}(\mathcal{G}(x)) \approx x$. However, in practice, the background of the generated image appears to be unreal and significantly different from the original image background, so that the cycle consistency loss can hardly reach 0. In our method, the attention network outputs a mask that separates the image into the region of interest and background. The background part will not be transformed so that the cycle consistency loss in the background reaches 0.

3.4.2 Attention Loss

Similar to cycle consistency, the attention map of the object x in the domain X predicted by attention network A_X should be consistent with the attention map of the transformed object by attention network A_Y. For example, if a horse is transformed into a zebra, the region of the zebra should be brought back to the horse as a cycle. That is to say, the regions of interest in the original image and the transformed image should be the same: $A_X(x) \approx A_Y(\mathcal{G}(x))$. Similarly, for each image y from domain Y, attention network A_Y and A_X should satisfy consistency: $A_Y(y) \approx A_X(\mathcal{F}(y))$. To that end, we propose an attention cycle-consistent loss:

$$\mathcal{L}_{A_{cyc}}(A_X, A_Y) = \mathbb{E}_{x \in X}\left[\|A_X(x) - A_Y(\mathcal{G}(x))\|_1\right] + \mathbb{E}_{y \in Y}\left[\|A_Y(y) - A_X(\mathcal{F}(y))\|_1\right] \tag{3.7}$$

In addition, we introduce a sparse loss to encourage the attention network to pay attention to a small region related to the object instead of the whole image:

$$\mathcal{L}_{sparse}(A_X, A_Y) = \mathbb{E}_{x \in X}\left[\|A_X(x)\|_1\right] + \mathbb{E}_{y \in Y}\left[||A_Y(y)||_1\right]. \tag{3.8}$$

Considering Equation (3.7), the attention maps of $A_X(\mathcal{F}(y))$ and $A_Y(\mathcal{G}(x))$ should be consistent to $A_Y(y)$ and $A_X(x)$, so they do not include additional sparse loss on $A_X(\mathcal{F}(y))$ and $A_Y(\mathcal{G}(x))$.

3.4.3 Perceptual loss

Previous models (e.g., CycleGAN [173] and Attention-GAN [24]) adopted Markovian discriminator (PatchGAN) which classifies whether the image patches instead of the entire images are real or fake. Patch-GAN models the image as a Markov random field which assumes independence between pixels separated by more than a patch diameter, which can be viewed as a form of texture/style loss. Comparing to the traditional GAN (ImageGAN) that penalizes the entire images, such a patch-level discriminator architecture has fewer parameters than a full-image discriminator and can achieve better performance [59]. However, in the object transfiguration problem, objects in a category also have their high-level features, while Patch-GAN with patch-level adversarial loss can only transform low-level information. As a result, we propose a perceptual loss to improve the transformed images in high-level feature space.

The proposed perceptual loss has been used on the image generation problems [72, 177]. Compared to previous works, the differences of our perceptual loss can be summarized as follow: first, previous works [72] used the perceptual loss by optimizing the perceptual similarly distance between the generated images and the ground truth. In our work, as we lack paired images, we design the loss by optimizing the distance between the output and the overall samples of the target domain. On the other hand, some previous works used in the low-level and middle-level features (e.g., features in the first four layers of VGG19 [61]) to construct texture or style features to instruct the image generation process. In contrast, we use high-level perceptual similarity (e.g., features in the last layers of VGG19) with the idea that samples in the same domain should have similar high-level features.

We propose the perceptual loss to force the high-level feature representation of the transformed images to be close to those of samples in the target domain. Let $f(\cdot)$ represent a feature transformation function, and the x denote a sample from domain X. Our aim is to minimize the expectation distance between the transformed image $\mathcal{G}(x)$ and the real samples $\{y_j\}_j^{N_Y}$ of the target domain Y in the feature space. Similarly, for each image y, the transformed image $\mathcal{F}(y)$ should be close to the domain X in their feature space. To this end, the perceptual loss

is given by:

$$\mathcal{L}_{percep}(\mathcal{G},\mathcal{F}) = \mathbb{E}_{x\in X}\mathbb{E}_{y\in Y}\|f(y)-f(\mathcal{G}(x))\|_1+\mathbb{E}_{y\in Y}\mathbb{E}_{x\in X}\|f(x)-f(\mathcal{F}(y))\|_1. \quad (3.9)$$

In practice, we use the pretrained VGG19 [131] as the feature mapping. The features of images are extracted from the fully connected layer before the last softmax layer. To reduce the computational complexity, we compute the expectation over samples of each domain that belongs to the same minibatch. Since we use the L1-norm as the perceptual distance metric, the incremental computational cost is limited.

To further reduce the space of possible mapping functions by the conditional generator, we also use the cycle-consistency loss in Equation (3.3). The adversarial losses in Equations (3.1), (3.2) are also used to encouraged to output realistic translations. Overall, our full objective is:

$$\begin{aligned}
&\mathcal{L}(T_X, T_Y, D_X, D_Y, A_X, A_Y) \\
&\quad = \mathcal{L}_{GAN}(\mathcal{G}, D_Y, X, Y) + \mathcal{L}_{GAN}(\mathcal{F}, D_X, X, Y) + \lambda_{percep}\mathcal{L}_{percep}(\mathcal{G},\mathcal{F}) \quad (3.10)\\
&\quad + \lambda_{cyc}\mathcal{L}_{cyc}(\mathcal{G},\mathcal{F}) + \lambda_{A_{cyc}}\mathcal{L}_{A_{cyc}}(A_X, A_Y) + \lambda_{sparse}\mathcal{L}_{sparse}(A_X, A_Y),
\end{aligned}$$

where λ_{sparse}, λ_{cyc} and λ_{percep} balance the relative importance of different terms. The attention networks, transformation networks and discriminative networks in both X domain and Y domain can be solved in the following min-max game:

$$\arg \min_{T_X,T_Y,A_X,A_Y} \max_{D_X,D_Y} \mathcal{L}(T_X, T_Y, D_X, D_Y, A_X, A_Y). \quad (3.11)$$

The optimization algorithm is described in the section 3.5.

3.4.4 Extra Supervision

In some cases, segmentation annotations can be collected and used as attention maps. For example, our horse $\rightarrow$ zebra image segmentation of horse is exactly the region of interest. We, therefore, supervise the training of the attention network by segmentation labels. Given a training set $\{(x_1, m_1), \cdots, (x_N, m_N)\}$ of N examples, m_i indicates the binary dense labels for each pixel. To learn the

attention maps for both X domain and Y domain, we minimize the discrepancy between predicted attention maps $A(x_i)$ and segmentation label m_i. The total attention loss can be written as:

$$\mathcal{L}_{A_{sup}}(A_X, A_Y) = \sum_{i=1}^{N_X} \|m_i - A_X(x_i)\|_1 + \sum_{j=1}^{N_Y} \|m_j - A_Y(y_j)\|_1. \tag{3.12}$$

The full objective thus becomes:

$$\begin{aligned} &\mathcal{L}(T_X, T_Y, D_X, D_Y, A_X, A_Y) \\ &\quad = \mathcal{L}_{GAN}(\mathcal{G}, D_Y, X, Y) + \mathcal{L}_{GAN}(\mathcal{F}, D_X, X, Y) \\ &\quad + \lambda_{cyc}\mathcal{L}_{cyc}(\mathcal{G}, \mathcal{F}) + \lambda_{A_{sup}}\mathcal{L}_{A_{sup}}(A_X, A_Y), \end{aligned} \tag{3.13}$$

where λ_{cyc} and $\lambda_{A_{sup}}$ control the relative importance of the objectives. As the attention maps are supervised by semantic annotations, we do not incorporate the constraints of Equations (3.7) and (3.8).

3.5 Implementation

For all experiments, the networks were trained with an initial learning rate of 0.0002 for the first 100 epochs and a linearly decaying rate that goes to zero over the next 100 epochs. We used the Adam solver [67] with batch size of 8. We updated the discriminative networks using a randomly selected sample from a buffer of previously generated images followed by [129]. The training process is shown in Table 3.1 The architectures of transformation networks and attention networks are based on Johnson *et al.* [61]. The discriminators are adapted from the Markovian Patch-GAN [58, 77, 169, 183]. The architecture of the transformation network and attention network is based on Johnson *et al.* [61], which contains two stride-2 convolutions, several residual blocks and two fractionally-convolutions with $\frac{1}{2}$ stride. Instance Normalization [123] is used after convolutional layers. For discriminative network, we adapt the Markovian Patch-GAN architecture [58, 77, 169, 183]. Instead of distinguishing full-images, the discriminative network distinguishes overlapping patches sampling from real and generated images. By doing this, the discriminative network has fewer parameters

Table 3.1: Architecture of the attention network, the transformation network and the discriminative network.

	Operation	Kernel size	Stride	Padding	Feature maps	Normalization	Nonlinearity
Attention network	Convolution	7	1	3	32	Instance Normalization	ReLU
	Convolution	3	2	1	64	Instance Normalization	ReLU
	Convolution	3	2	1	128	Instance Normalization	ReLU
	Residual block × 6	3	1	1	128	Instance Normalization	ReLU
	Fractional-convolution	3	1/2	1	64	Instance Normalization	ReLU
	Fractional-convolution	3	1/2	1	32	Instance Normalization	ReLU
	Convolution	7	1	0	1	-	tanh
Transformation network	Convolution	7	1	3	32	Instance Normalization	ReLU
	Convolution	3	2	1	64	Instance Normalization	ReLU
	Convolution	3	2	1	128	Instance Normalization	ReLU
	Residual block × 9	3	1	1	128	Instance Normalization	ReLU
	Fractional-convolution	3	1/2	1	64	Instance Normalization	ReLU
	Fractional-convolution	3	1/2	1	32	Instance Normalization	ReLU
	Convolution	7	1	0	3	-	tanh
Discriminative network	Convolution	4	2	1	64	-	LeakyReLU
	Convolution	4	2	1	128	Batch Normalization	LeakyReLU
	Convolution	4	2	1	256	Batch Normalization	LeakyReLU
	Convolution	4	2	1	512	Batch Normalization	LeakyReLU
	Convolution	4	1	1	512	Batch Normalization	LeakyReLU
	Convolution	4	1	1	1	-	Linear
	Resnet Block	Residual block that contains 3×3 convolutional layers					
	LeakyReLU	Slope 0.2					

and can be applied to any size of the input. Patch size is set to 70×70. Details are listed in Table 3.1. The optimization algorithm of Equation (9) is shown in Algorithm 1. In practice, K_t, K_d and K_a are set to 1.

3.6 Experiments

In this section, we first compare the proposed method against state-of-the-art works. Then, we study the function of the *attention sparse loss* and compare our method against some variants. Next, we demonstrate empirical results in a supervised manner. Last, we discuss the effect of the proposed method for global image-to-image transform, e.g., summer $\rightarrow$ winter. For all experiments, we denote our model with full objective function as "Attention-GAN+" and represent our model without perceptual loss λ_{percep} as "Attention-GAN".

We evaluate our method on three tasks: horse $\leftrightarrow$ zebra, tiger $\leftrightarrow$ leopard and apple $\leftrightarrow$ orange. The images for horse, zebra, apple, and orange are provided by CycleGAN [183]. The images for tiger and leopards are from ImageNet [27], which consists of 1,444 images for tiger, 1,396 for leopard. We randomly select 60 images for testing and the rest for the training set. In the supervised experiment, we perform the horse $\leftrightarrow$ zebra task where images and annotations are from the MSCOCO dataset [84]. For each object category, images in the MSCOCO training set are used for training and those in the MSCOCO validation set are for testing. For all experiments, the training samples are first scaled as 286×286, and then randomly flipped and cropped as 256×256. In the test phase, we scale input images to the size of 256×256.

We use the recently proposed Kernel Inception Distance (KID) [15] to quantitatively evaluate our image translation framework. KID computes the squared maximum mean discrepancy (MMD) between feature representations of real and generated images. Such feature representations are extracted from the Inception network architecture [139]. KID has an unbiased estimator, which makes it more reliable, especially when there are fewer test images than the dimensionality of the inception features. Following UAIT [101], the mean KID values reported are averaged over 10 different splits, each of which has 50 images randomly sampled from each domain.

Algorithm 1 Algorithm for training the attention generative network.

Require: The set of X domain sample $\{x_i\}_{i=1}^{N_X} \in X$, the set of Y domain sample $\{y_i\}_{j=1}^{N_Y} \in Y$, the number of discriminative network updates per step K_d, the number of transformation network updates per step K_t, the number of attention network updates per step K_a.

for number of training iterations **do**
  for K_d steps **do**
    Sample minibatch of n examples x_i from domain X and n examples y_i from domain Y.
    Generate images $\hat{y}_i = \mathcal{G}(x_i)$ in Equation (5) and $\hat{x}_i = \mathcal{F}(y_j)$ in Equation (6).
    Update discriminator D_X and D_Y by ascending along its stochastic gradient:

$$\nabla_{\theta_{D_X}} \frac{1}{n} \sum_{i=1}^{n} [(D_X(x_i) - 1)^2] + [D_X(\hat{x}_i)^2].$$
$$\nabla_{\theta_{D_Y}} \frac{1}{n} \sum_{i=1}^{n} [(D_Y(y_i) - 1)^2] + [D_Y(\hat{y}_i)^2],$$

  end for
  for K_t steps **do**
    Sample minibatch of n examples x_i from domain X and m examples y_j from domain Y.
    Generated images $\hat{y}_i = A_X(x_i) \odot T_X(x_i) + (1 - A_X(x_i)) \odot x_i$, and $\hat{x}_j = A_Y(y_j) \odot T_Y(y_j) + (1 - A_Y(y_j)) \odot y_j$.
    Update transformation network T_X and T_Y by descending along their stochastic gradient:

$$\nabla_{\theta_{T_X}} \frac{1}{n} \sum_{i=1}^{n} \Big(D_Y [(\hat{y}_i)^2] + \frac{1}{m} \sum_{j=1}^{m} \|f(y_j) - f(\hat{y}_i)\|_1 \Big),$$
$$\nabla_{\theta_{T_Y}} \frac{1}{m} \sum_{j=1}^{m} \big(D_X [(\hat{x}_j)^2] + \frac{1}{n} \sum_{i=1}^{n} \|f(x_i) - f(\hat{x}_j)\|_1 \big).$$

  end for
  for K_a steps **do**
    Sample minibatch of n examples x_i from domain X and n examples y_i from domain Y.
    Generate images $\hat{y}_i = A_X(x_i) \odot T_X(x_i) + (1 - A_X(x_i)) \odot x_i$, and $\hat{x}_j = A_Y(y_j) \odot T_Y(y_j) + (1 - A_Y(y_j)) \odot y_j$.
    Update attention network A_X and A_Y by descending along their stochastic gradient:

$$\nabla_{\theta_{A_X}} \frac{1}{n} \sum_{i=1}^{n} (D_Y [(\hat{y}_i)^2] + \|A_X(x_i)\|_1 + \|A_X(x_i) - A_Y(\hat{y}_i)\|_1),$$
$$\nabla_{\theta_{A_Y}} \frac{1}{n} \sum_{i=1}^{n} (D_X [(\hat{x}_i)^2] + \|A_Y(y_i)\|_1 + \|A_Y(y_i) - A_X(\hat{x}_i)\|_1).$$

  end for
end for

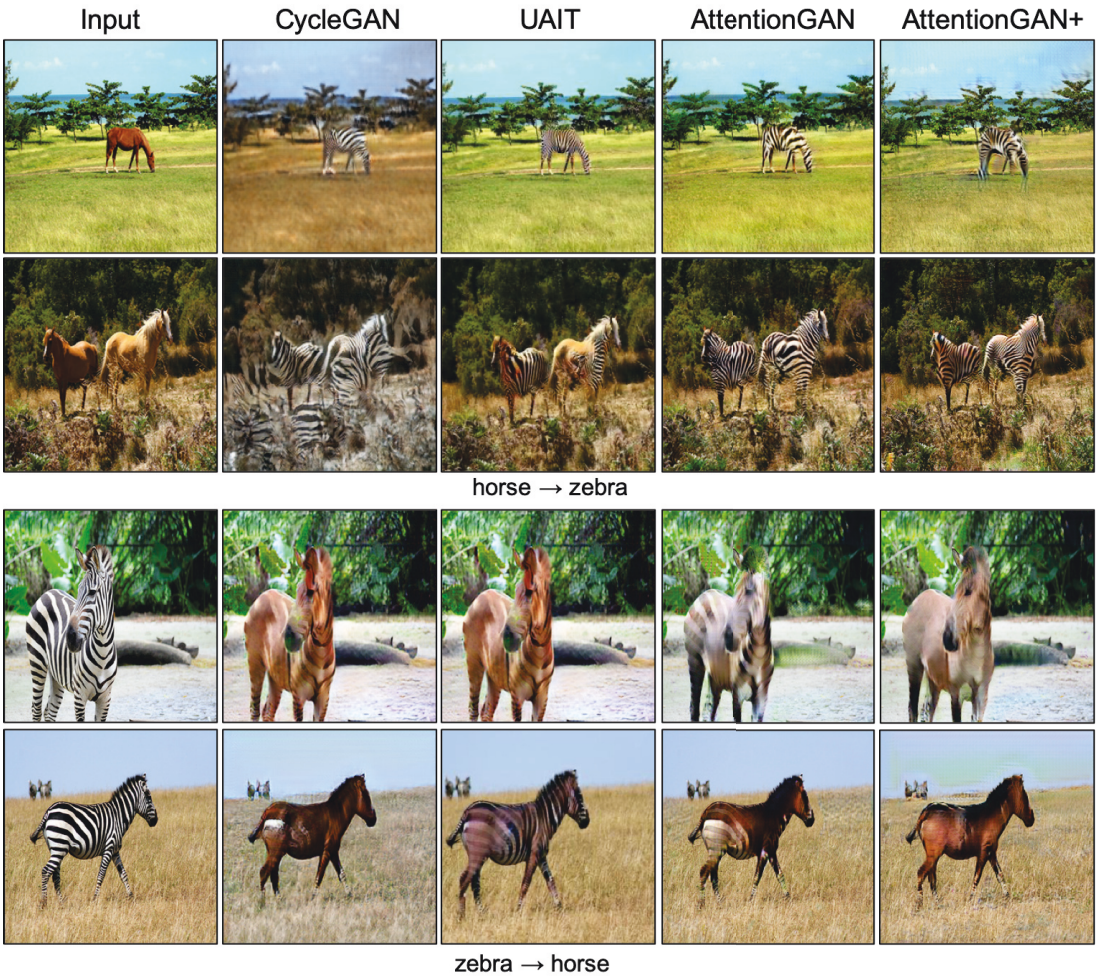

Figure 3.5: Comparison with CycleGAN [183], UAIT [101] and Attention-GAN [24] on horse ↔ zebra. In each case, the first image is the input image, the second is the result of CycleGAN, the third is the results of UAIT, the forth is the result of Attention-GAN and the last is the result of our Attention-GAN+.

We compare the proposed method to DiscoGAN [66], CycleGAN [183] and DualGAN [169], which were similar but use different adversarial losses. DiscoGAN used a standard GAN loss [44], CycleGAN uses a least-squared GAN loss [98], and DualGAN used a Wasserstein GAN loss [4]. We also compare with [86]'s UNIT algorithm, which leveraged the latent space assumption between each pair of source/target images. Additionally, we compare with [101]'s unsupervised attention-guided image-to-image translation (referred to "UAIT"), which is a

contemporaneous work of our attention-GAN. UAIT shared our goal of introducing an attention network to disentangle the image background with the object, but we adopt a different method to obtain an accurate attention map. UAIT stops the training of the attention networks after 30 epochs to prevent it from focusing on the background as well. In contrast, we learn the attention map through a cyclic loss and a sparse loss.

The comparison is shown in Table 3.2. Our approach with perceptual loss achieves the lowest KID score in all tasks, with Attention-GAN and UAIT [101] as the next best performing approaches. UAIT [101] is a contemporaneous work with Attention-GAN [24] sharing the goal of learning an attention map for image translation. Both of two methods achieve closed performance, while Attention-GAN slightly outperformed UAIT. After that, UNIT [86] achieves the fourth-lowest KID score. Finally, DualGAN [169] and DiscoGAN [66] achieve the worst performance in terms of mean KID values. In all, the results verify the superiority of our method by incorporating the attention mechanism and perceptual loss.

3.6.1 Qualitative Comparisons

In this section, we compare our Attention-GAN, Attention-GAN+ with CycleGAN and UAIT in terms of visual quality. Results of horse $\leftrightarrow$ zebra are shown in Figure 3.5. We observed that our method provides translation results of higher visual quality in both the consistency of background and the fidelity of the transformed object. In terms of the background consistency, we observe that all the methods except CycleGAN could remain the background in the transformation. UAIT, however, cannot precisely detect the region of the object in some complex background. For example, in the first row of Figure 3.5, CycleGAN transforms the green grass into brown when the horse is transformed into the zebra. In contrast, UAIT, Attention-GAN, and Attention-GAN+ maintain the grass color as the original ones. In the second row of Figure 3.5, UAIT only transformed part of horses into the zebra texture. We consider the underlying reason is that UAIT only optimizes the attention network in the first 30 epochs, to prevent it from focusing on the background, but it is hard to tell if the attention network has reached the optimization point. In contrast, our Attention-GAN and

Table 3.2: Kernel Inception Distance $\times 100 \pm$ std. $\times 100$ for different image translation algorithms. Lower is better.

Task	apple $\rightarrow$ orange	orange $\rightarrow$ apple	zebra $\rightarrow$ horse	horse $\rightarrow$ zebra
DiscoGAN [66]	18.34 ± 0.75	21.56 ± 0.80	16.60 ± 0.50	13.68 ± 0.28
DualGAN [169]	13.04 ± 0.72	12.42 ± 0.88	12.86 ± 0.50	10.38 ± 0.31
UNIT [86]	11.68 ± 0.43	11.76 ± 0.51	13.63 ± 0.34	11.22 ± 0.24
CycleGAN [183]	8.48 ± 0.53	9.82 ± 0.51	11.44 ± 0.38	10.25 ± 0.25
UAIT [101]	6.44 ± 0.69	5.32 ± 0.48	8.87 ± 0.26	6.93 ± 0.27
Attention-GAN	6.34 ± 0.69	4.51 ± 0.54	8.37 ± 0.32	5.82 ± 0.29
Attention-GAN+ (Ours)	3.90 ± 0.51	3.84 ± 0.52	4.12 ± 0.22	4.32 ± 0.21

Attention-GAN+ with sparse loss and attention cycle loss optimize all the players of the model in the entire training procedure, which leads to a better attention prediction.

Comparison results on tiger $\leftrightarrow$ leopard and apple $\leftrightarrow$ orange are shown in Figure 3.6. The results of Attention-GAN are more visually pleasing than those of CycleGAN. In most cases, CycleGAN can not preserve background consistency, e.g., the blue jeans in the first image are transformed into yellow, the blue water in the third image is transformed into yellow and the yellow weeds in the last image are transformed into green color. One possible reason is that our Attention-GAN disentangles the background and object of interests by the attention network and only transforms the object, while the compared method only uses one generative network that manipulates the whole image.

On the other hand, we observe that our method with perceptual loss (Attention-GAN+) achieves the best fidelity of the transformed object. For instance, in the zebra $\rightarrow$ horse task, we can still find zebra stripes in the transformed horse of other methods while the brown color is rendered on the horse body. In contrast, the transformed horses of our method are more realistic, which is consistent with the quantitative results in Table 3.2.

3.6.2 Background Consistency Comparison

Since object transfiguration is required to predict the region of interest and transform the object while preserving the background, we introduce metrics to assess the background consistency of the translations. We compute PSNR and SSIM be-

Table 3.3: Background consistency performance of different tasks for background PSNR and SSIM.

	Task	CycleGAN	UAIT	Attention-GAN (Unsupervised)	Attention-GAN (Supervised)	Attention-GAN+ (Unsupervised)	Attention-GAN+ (Supervised)
PSNR	horse → zebra	18.19	21.63	22.26	24.59	22.23	23.40
	zebra → horse	18.10	21.36	21.54	23.93	21.92	23.40
SSIM	horse → zebra	0.6725	0.8585	0.9003	0.9482	0.8809	0.9354
	zebra → horse	0.7155	0.8603	0.8988	0.9534	0.9027	0.9432

tween generated image background and original image background. PSNR is an approximation to the human perception of reconstruction quality, which is defined via mean squared error (MSE). Given testing samples $\{(x_1, m_1), \cdots, (x_N, m_N)\}$, we use pixels-wise multiplication $\odot$ by the segmentation mask to compute image background PSNR:

$$\frac{1}{N}\sum_{i=1}^{N} PSNR\left(x_i \odot (1 - m_i), \mathcal{G}(x_i) \odot (1 - m_i)\right), \tag{3.14}$$

where x_i indicates original image, $\mathcal{G}(x_i)$ indicates the resulting image, $(1 - m_i)$ indicates the image background, the pixels-wise multiplication $x_i \odot (1 - m_i)$ indicates the background of original image, and $\mathcal{G}(x_i) \odot (1 - m_i)$ indicates the background of generated image. Similarly, we use SSIM to assess the structural similarity between the background of the original image and composited output by using pixels-wise multiplication:

$$\frac{1}{N}\sum_{i=1}^{N} SSIM\left(x_i \odot (1 - m_i), y_i \odot (1 - m_i)\right). \tag{3.15}$$

In the experiment, we use MSCOCO [84] dataset's test images and segmentation mask to evaluate background quality of the generated image. We compare our method with CycleGAN [183], UAIT [101] and our Attention-GAN in terms of the image background PSNR and SSIM (Equations (3.14) and (3.15)). The test dataset is from the MSCOCO dataset [84]. As the MSCOCO dataset does not have the classes of tiger or leopard, and apples and oranges in images are too small, we only compare the results of horse $\leftrightarrow$ zebra. Results are shown in Table 3.3. As can be seen, for both PSNR and SSIM, our method in unsupervised fashion outperforms CycleGAN by a large margin, which indicates that the proposed model predicts an accurate attention map and achieves a better performance of transformation quality. We also notice our method slightly outperforms UAIT,

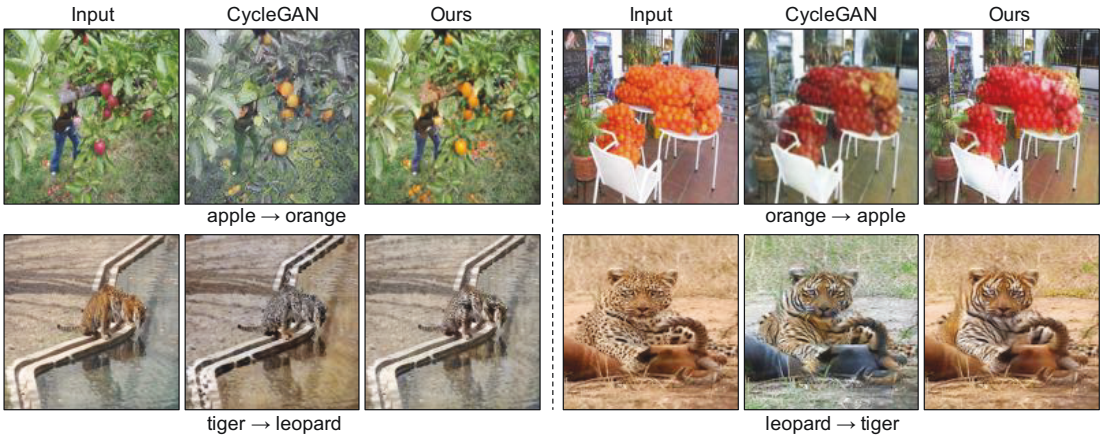

Figure 3.6: Comparison with CycleGAN on apple ↔ orange and tiger ↔ leopard. In each case: input image (left), result of CycleGAN [183] (middle), and result of our Attention-GAN (right).

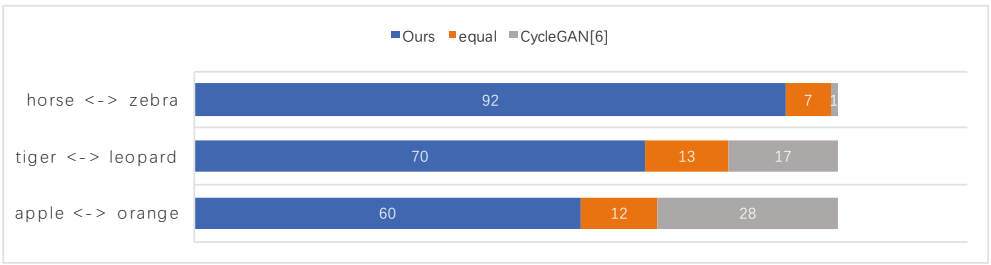

Figure 3.7: The stacked bar chart of participants' preferences for our methods compared to CycleGAN [173]. The blue bar indicates the number of images that more participants prefer our results. The gray bar indicates the number of images that more participants prefer CycleGAN's results. The orange bar indicates the number of images where two methods get an equal number of votes from 10 participants.

which is consistent with the qualitative result in Figure 3.5. As our method has introduced attention sparse loss and attention cycle-consistent loss, the attention network could be optimized in the overall training procedure, which helps the attention network fall in the optimization point.

Table 3.4: Background consistency performance of different λ_{sparse} in terms of background PSNR and SSIM.

	Task	$\lambda_{sparse} = 5$	$\lambda_{sparse} = 1$	λ_{sparse}=0.8	λ_{sparse}=0.5	λ_{sparse}=0.3	λ_{sparse}=0.1	$\lambda_{sparse} = 0$
PSNR	horse → zebra	24.2173	22.2629	21.4270	21.2627	21.2097	21.6522	19.8521
	zebra → horse	24.5339	22.0778	22.5059	22.2908	22.3544	21.2098	20.3272
SSIM	horse → zebra	0.9367	0.8988	0.8844	0.8744	0.8691	0.8685	0.8291
	zebra → horse	0.9542	0.9003	0.9262	0.9182	0.9160	0.8994	0.8903

3.6.3 Human Perceptual Study

We further evaluate our algorithm via a human study. We perform pairwise A/B tests deployed on the Amazon Mechanical Turk platform. We follow the same experiment procedure in [21, 153]. The participants are asked to select a more realistic image from each pair. Each pair contains two images translated from the same source image by two approaches. We test the tasks of horse ↔ zebra, tiger ↔ leopard and apple ↔ orange. In each task, we randomly select 100 images from the test set. Each image is compared by 10 participants. Figure 3.7 shows the participants' preference among 100 examples. We observe that 92 results of our methods outperform results of CycleGAN in horse ↔ zebra task. In tiger ↔ leopard, still only 17% results of the compared method beat ours. It indicates that our attention-guided model achieves better qualitative performance than the existing methods that transform objects in a generative network. We also notice that in apple ↔ orange task, only 60 results of our Attention-GAN outperform the compared method. We consider the reason is that a large portion of images in the apple and orange dataset are close-up images whose background is simple so that CycleGAN could reach a competitive result.

3.6.4 Model Analysis

3.6.4.1 Analysis of Intermediate Model Results

We perform model analysis by visualizing the intermediate results of our model in Figure 3.8. In the second column, the attention results are visualized by heat-map. As can be seen, while being completely unsupervised, the attention network of the model successfully disentangles the objects of our interests and the background from the input image. The third column is the output of the

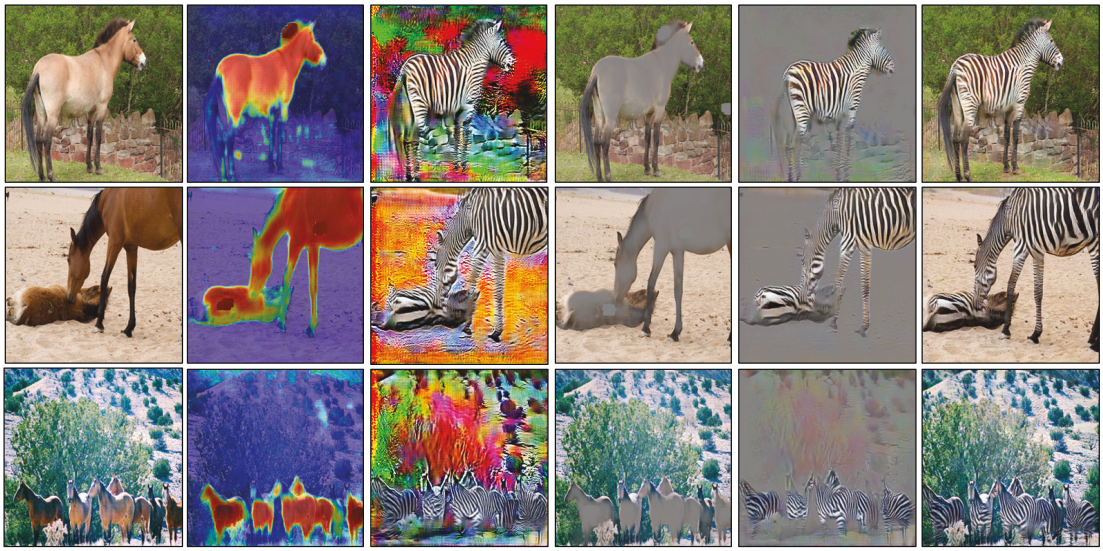

Figure 3.8: Generation results of our model on horse → zebra. From left to right: Inputs, attention maps, outputs of transformation network, background images factorized by attention maps, object of images factorized by attention maps, final composite images.

transformation network, where the transformed zebra is visually pleasing while the background parts of images are meaningless. It demonstrates that the transformation network only focuses on transforming the object of interests. The final output images in the last column are combined with the background parts in the fourth column and the objects of interests in the fifth column.

3.6.4.2 Analysis of Parameter

We further explore the effect and sensitivity of the attention sparse loss. Figure 4.14 shows the qualitative results of variants of our model on horse → zebra. We observe that without the sparse loss ($\lambda_{sparse} = 0$), the attention network would predict some parts of the image background as regions of interest. When λ_{sparse} was set to 5, the attention mask shrinks too much to cover the whole object of interests. It is because if we emphasize too much on the relative importance of the sparse loss, the attention network can not comprehensively predict the object location. This indicates that the λ_{sparse} can be viewed as a parameter that balance the performance of background consistency and transformation quality.

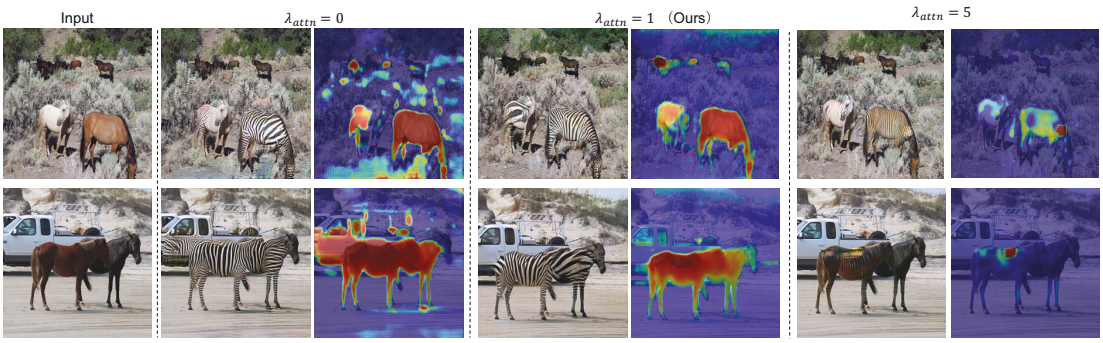

Figure 3.9: The effect of sparse loss with different parameters λ_{sparse} for mapping horse $\rightarrow$ zebra. From left to right: input, output and attention map without sparse loss, input and attention map when $\lambda_{sparse} = 1$, input and attention map when λ_{sparse}=5.

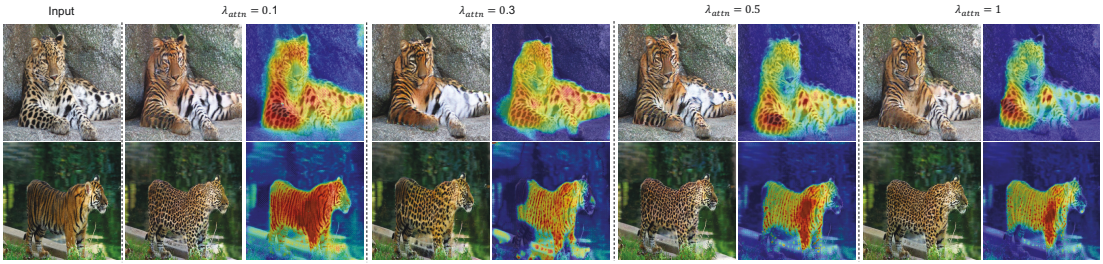

Figure 3.10: The results of tiger$\leftrightarrow$leopard with different values on $\lambda_{sparse} = \{0.1, 0.3, 0.5, 1\}$.

We find that $\lambda_{sparse} = 1$ is an appropriate choice, which makes a good balance to pay enough attention to the objects of interests. Table 3.4 shows the background consistent results on horse $\leftrightarrow$ zebra. We observe that the performance is close with λ_{sparse} between 0.1 to 1. The tiny fluctuations are mainly caused by the initialization of the network. We also explore the effect of λ_{sparse} on tiger $\leftrightarrow$ leopard task. As shown in Figure 3.10, all the results of attention maps focus on the objects of interests. That is to say, λ_{sparse} in the range $[0.1, 1]$ does not drastically affect the results and is suitable for different object transfiguration task.

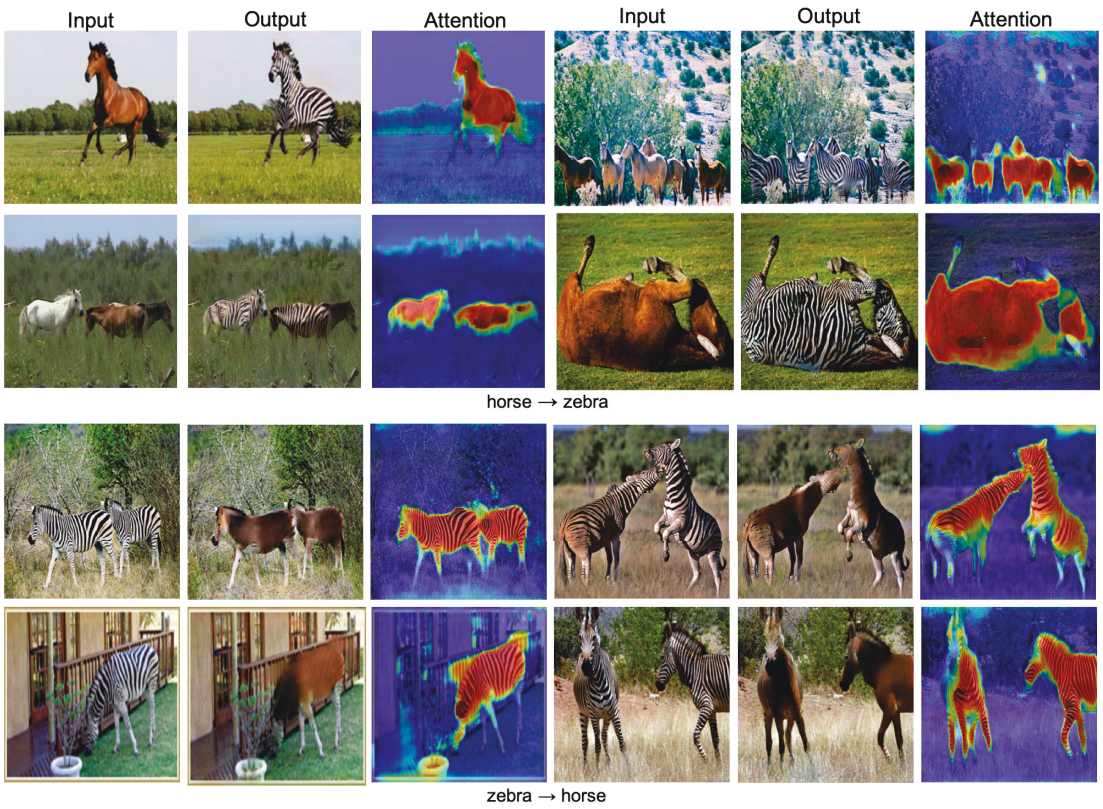

Figure 3.11: Results of horse ↔ zebra by unsupervised Attention-GANs. From left to right: input images, outputs of the proposed Attention-GAN, the predicted attention maps of Attention-GAN.

3.6.4.3 Analysis of Attention Prediction

We evaluate the foreground mask of horse in terms of UoI and mAPr@0.5. The unsupervised Attention-GAN got 28.1% of UoI and 20.3% of mAPr@0.5. On the other hand, the supervised Attention-GAN got 37.8% of UoI score and 30.5% of mAPr@0.5. Although our algorithm is not particularly designed for semantic segmentation, the proposed attention network is able to learn the object of interests in an unsupervised way and achieve a reasonable performance. More transformed results and attention prediction are shown in Figure 3.11, 3.12 and 3.13, which achieve satisfied performance.

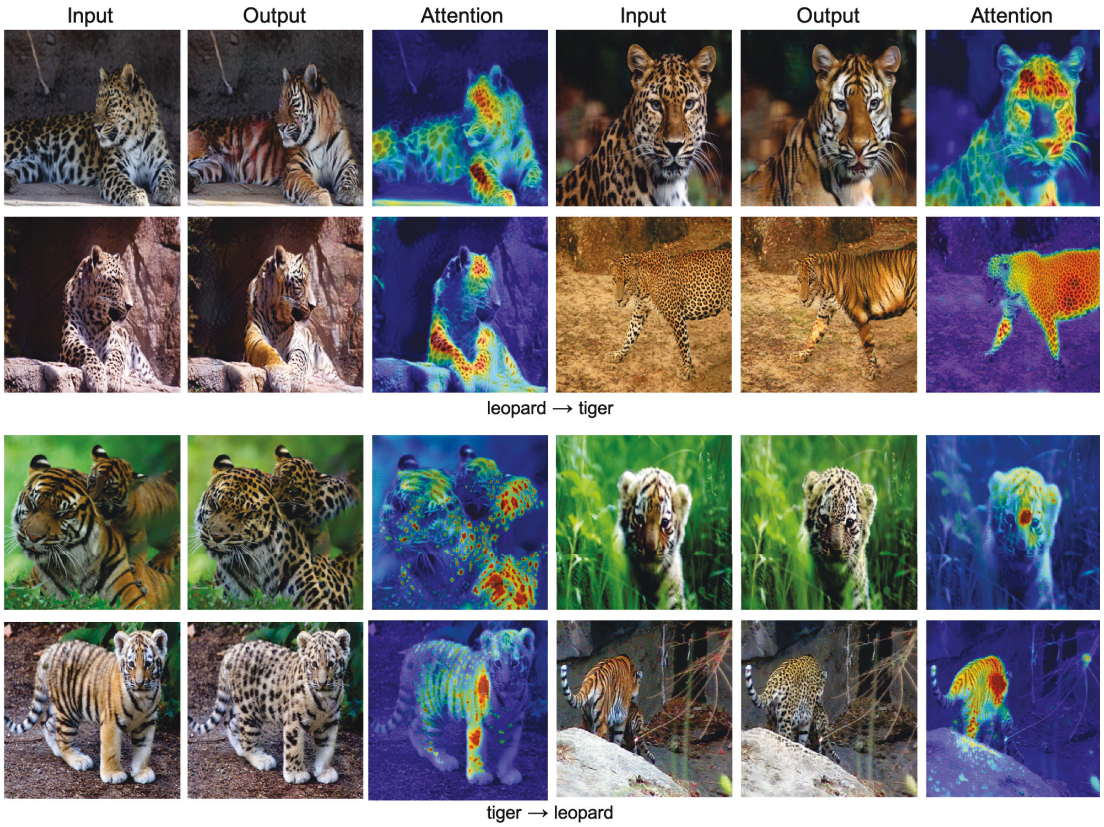

Figure 3.12: Results of tiger ↔ leopard by unsupervised Attention-GANs. From left to right: input images, outputs of the proposed Attention-GAN, the predicted attention maps of Attention-GAN.

3.6.5 Comparison of Supervised Results

In this subsection, we compare the performance of CycleGAN, Attention-GAN, and Attention-GAN+ in the unsupervised and supervised way. We compare quantitative results by computing PSRN, SSIM of background region between generated and original images in horse ↔ zebra task. In Table 3.3, the Attention-GAN and Attention-GAN+ with supervision outperform unsupervised version and CycleGAN from the perspective of background consistency. This demonstrates that the attention network predicts the object of interests more accurately with the segmentation mask. Also, we find that both the Attention-GAN and Attention-GAN+ achieves satisfying results in terms of background consistency. Figure 3.14 shows the qualitative results as well as the predicted attention maps. Results

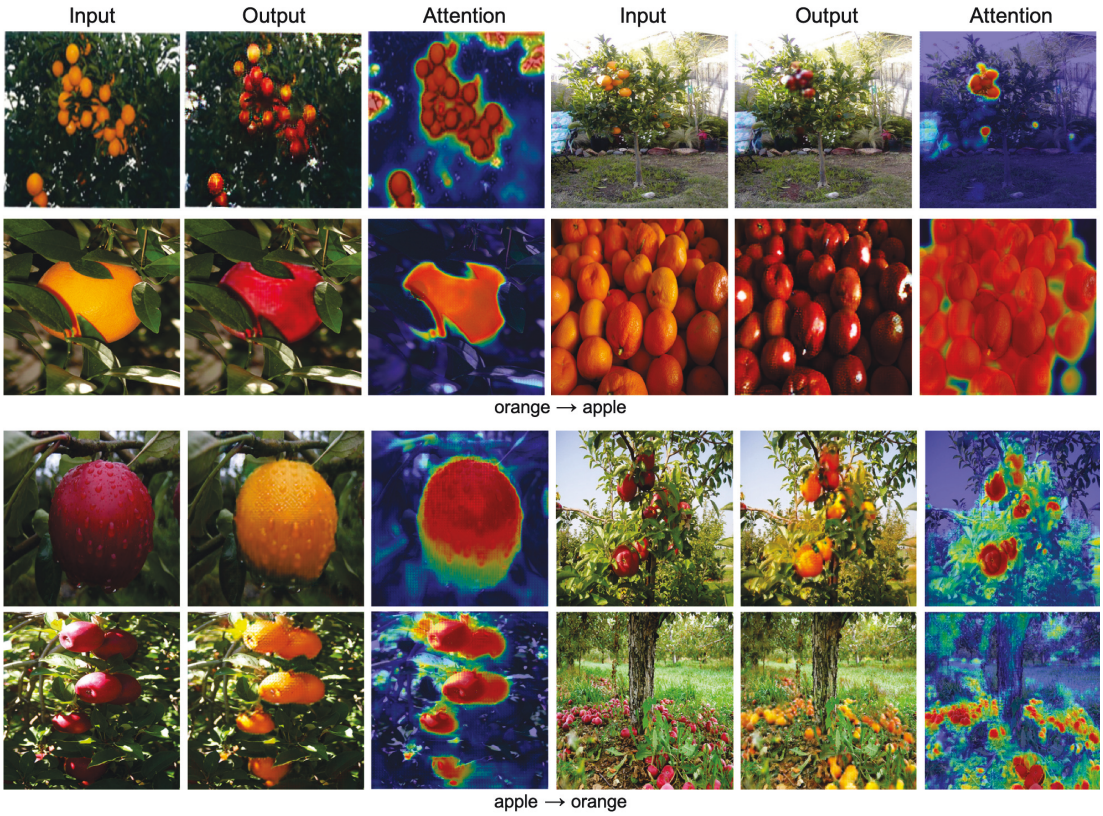

Figure 3.13: Results of apple ↔ orange by unsupervised Attention-GANs. From left to right: input images, outputs of the proposed Attention-GAN, the predicted attention maps of Attention-GAN.

demonstrate that the attention maps with supervision predict more accurately than the unsupervised version. Also, the attention maps tend to be dark red or dark blue, which indicates the supervised attention network predicts with higher confidence and disentangles the background and object of interests more clearly. In some cases, CycleGAN and Attention-GAN in an unsupervised way would mistake to predict some parts of the background as targets. For example, CycleGAN transforms the person (the first case in Figure 3.14) and the carriage (the second case in Figure 3.14) into the texture of zebra. The unsupervised Attention-GAN fails to predict the region of the horse in the first case and mistake to predict the carriage as a horse in the second case. Comparing to the Attention-GAN, results show that Attention-GAN+ with the proposed perceptual loss outputs more realistic transformed results. For instance, in the first row and the second

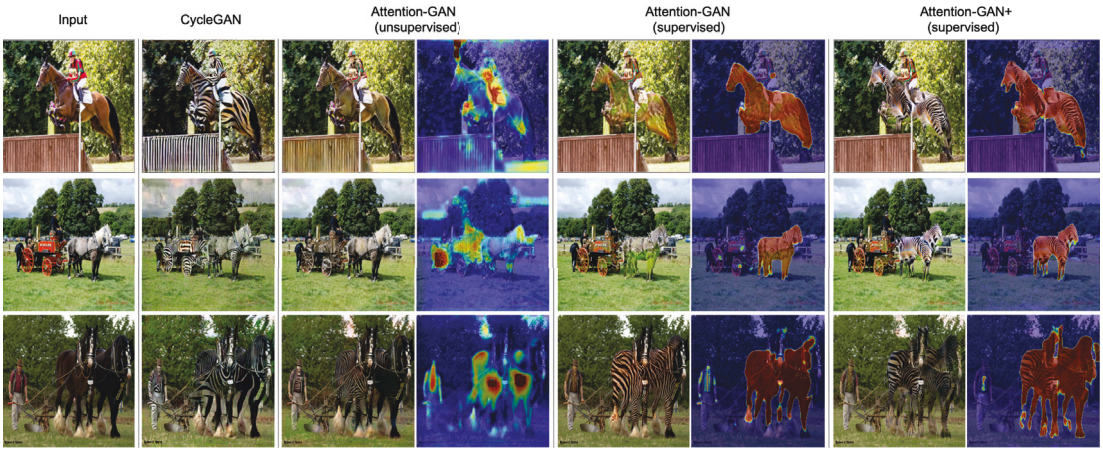

Figure 3.14: Comparison of horse → zebra between CycleGAN [183], unsupervised Attention-GAN, supervised Attention-GAN, and supervised Attention-GAN+.

row, the generated zebra of the Attention-GAN would be green which is the color of the background, while the Attention-GAN+ generated zebra of high fidelity.

3.6.6 Global Image Transformation

We discuss the feasibility of our model in global image transformation. In the global image transformation task, there is no obvious object need to be transformed in the images, e.g., summer ↔ winter. Both local and global image transformation is important. The proposed attention-GAN is effective to identify important regions in image-to-image transformation, and it can also lead to some interesting observations in global image transformation. In summer ↔ winter, there is no explicit object of interest, but the algorithm does recognize some regions with more attention, e.g., grass and trees in Figure 3.15, which are usually green in summer and brown in winter. Meanwhile, regions without distinctive characteristics, e.g., the blue sky would not be attended.

3.7 Summary

This chapter introduces the attention mechanism into the generative adversarial nets considering image context and structure information on object transfigura-

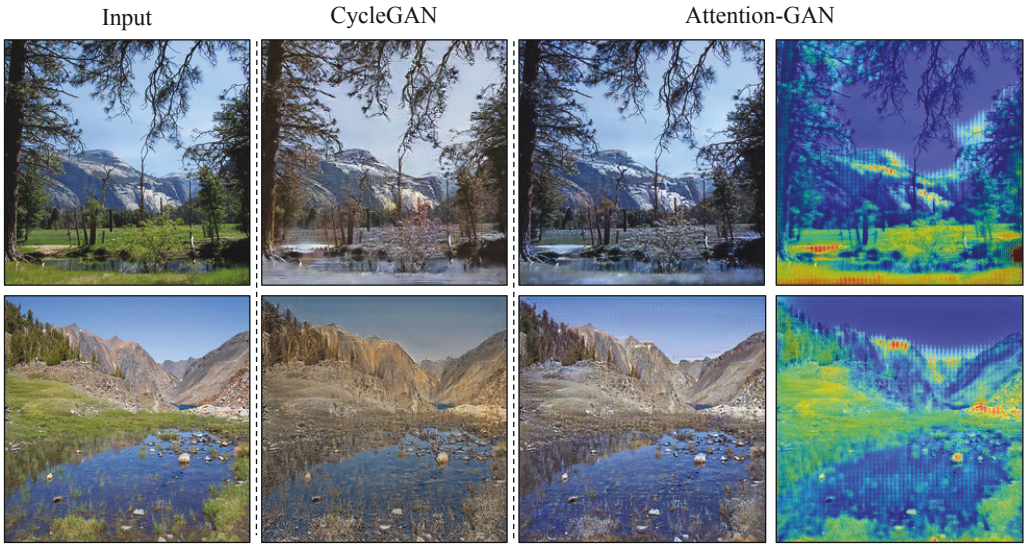

Figure 3.15: Results of Summer → Winter comparing with CycleGAN. From left to right: input images, results of CycleGAN, final outputs of the proposed Attention-GAN, the predicted attention maps of Attention-GAN.

tion problem. To improve the fidelity of the outputs in the high-level features, we introduce a perceptual loss by considering the similarity of the transformed images and the overall target images in the high-level feature space. In our model, we develop a three-player model that consists of an attention network, a transformation network, and a discriminative network. The attention network predicts the regions of interest whilst the transformation network transforms the object from one class to another. We show that our model has advantages over the previous generation methods in preserving background consistency and transformation quality.

Chapter 4

Gated Generaitive Model for Global Image Transformation

In this chapter, we investigate multi-domain image synthesis for style transfer. Style transfer describes the rendering of an image's semantic content as different artistic styles. Recently, generative adversarial networks (GANs) have emerged as an effective approach in style transfer by adversarially training the generator to synthesize convincing counterfeits. However, traditional GAN suffers from the mode collapse issue, resulting in unstable training and making style transfer quality difficult to guarantee. Also, the GAN generator is only compatible with one style, so a series of GANs must be trained to provide users with choices to transfer more than one kind of style. In this chapter, we focus on tackling these challenges and limitations to improve style transfer. We propose adversarial gated networks (Gated-GAN) to transfer multiple styles in a single model. The generative networks have three modules: an encoder, a gated transformer, and a decoder. Different styles can be achieved by passing input images through different branches of the gated transformer. To stabilize training, the encoder and decoder are combined as an auto-encoder to reconstruct the input images. The discriminative networks are used to distinguish whether the input image is a stylized or genuine image. An auxiliary classifier is used to recognize the style categories of transferred images, thereby helping the generative networks generate images in multiple styles. In addition, Gated-GAN makes it possible to explore

a new style by investigating styles learned from artists or genres. Our extensive experiments demonstrate the stability and effectiveness of the proposed model for multi-style transfer.

4.1 Introduction

Style transfer refers to redrawing an image by imitating another artistic style. Specifically, given a reference style, one can make the input image look like it has been redrawn with a different stroke, perceptual representation, color scheme, or that it has been retouched using a different artistic interpretation. Manually transferring the image style by a professional artist usually takes considerable time. However, style transfer is a valuable technique with many practical applications, for example quickly creating cartoon scenes from landscapes or city photographs and providing amateur artists with guidelines for painting. Therefore, optimizing style transfer is a valuable pursuit.

Style transfer, as an extension of texture transfer, has a rich history. Texture transfer aims to render an object with the texture extracted from a different object [5, 35, 52, 74]. In the early days, texture transfer used low-level visual features of target images, while the latest style transfer approaches are based on semantic features derived from pre-trained convolutional neural networks (CNNs). Gatys *et al.* [40] introduced the neural style transfer algorithm to separate natural image content and style to produce new images by combining the content of an arbitrary photograph with the styles of numerous well-known works of art. A number of variants emerged to improve the speed, flexibility, and quality of style transfer. Johnson *et al.* [61] and Ulyanov *et al.* [141] accelerated style transfer by using feedforward networks, while Chen *et al.* [19], Li *et al.* [80] and Odena *et al.* [108] achieved multi-style transfer by extracting each style from a single image. Ulyanov *et al.* [142] and Luan *et al.* [93] enhanced the quality of style transfer by investigating instance normalization in feedforward networks [141].

CNN-based style transfer methods can now produce high-quality imitative images. However, these methods focus on transferring the original image to the style provided by another style image (typically a painting). In contrast, collection style transfer aims to stylize a photograph by mimicking an artist's or genre's

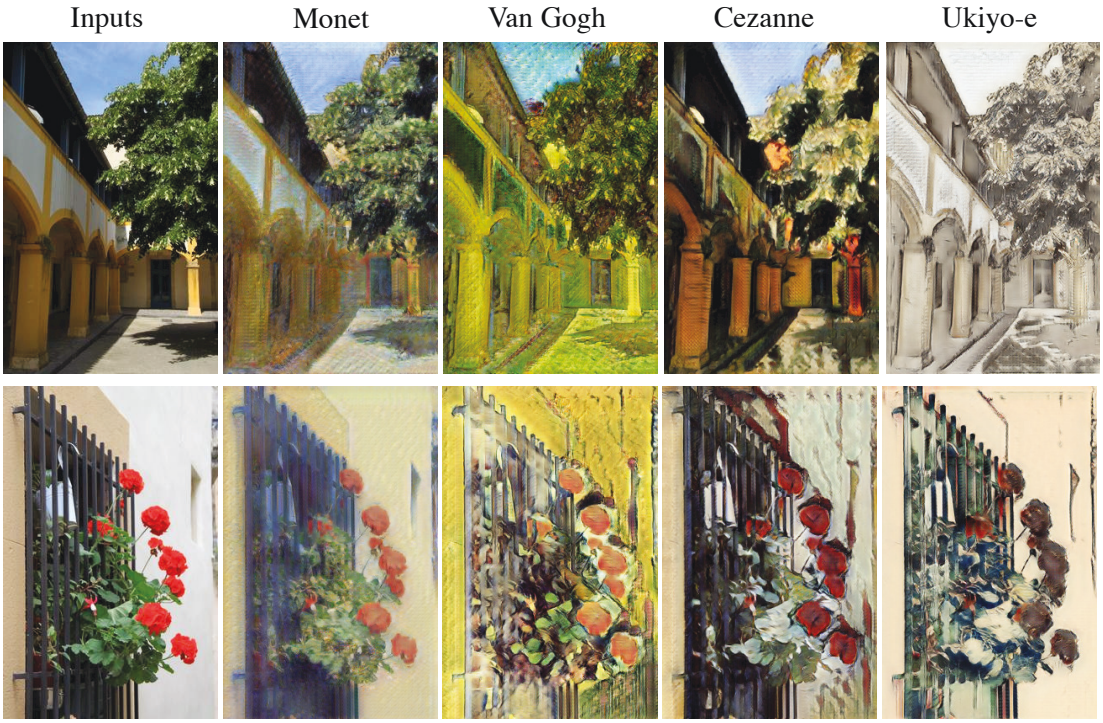

Figure 4.1: Gated-GAN for multi-collection style transfer. The images are produced from a single model with a shared encoder and decoder are shared. Styles are controlled by switching different gated-transformer module. From left to right: original images, transferred images in Monet style, transferred images in Van Gogh's style, transferred images in Cezanne's style, transferred images in Ukiyoe-e's style.

style. In practice, when a user takes a picture of a beautiful landscape, he might hope to re-render it on canvas such that it appears to have been painted by an artist, e.g., *Monet*, or in the style of a famous animation, e.g., *Your Name*. Given an in-depth understanding of an artist's collection of paintings, it is possible to imagine how the artist might render the scene. With this in mind, generative adversarial networks (GANs) [44] can be applied to learn the distribution of an artist's paintings. GANs are a framework in which two neural networks compete with each other: a generative network and a discriminative network. The generative and discriminative networks are simultaneously optimized in a two-player game, where the discriminative networks aim to determine whether or not the input is painted by the artist, while the generative network learns to generate im-

ages to fool the discriminative networks. However, the GAN training procedure is unstable. In particular, without paired training samples, the original GANs cannot guarantee that the output imitations contain the same semantic information as that of the input images. CycleGAN [183], DiscoGAN [66], DualGAN [169] proposed cycle-consistent adversarial networks to address the unpaired image-to-image translation problem. They simultaneously trained two pairs of generative networks and discriminative networks, one to produce imitative paintings and the other to transform the imitation back to the original photograph and pursue cycle consistency.

Considering the wide application of style transfer on mobile devices, space-saving is an important algorithm design consideration. Methods of CycleGAN [183], DiscoGAN [66], DualGAN [169] could only transfer one style per network. In this work, we propose a gated transformer module to achieve multi-collection style transfer in a single network. Moreover, previous methods adopted cycle-consistent loss requires an additional network that converts the stylized image into the original one. With the increase in the number of transferred styles, the training algorithm will become complicated if we adopt cycle-consistent loss. Also, style transfer is actually a one-sided translation problem, which does not expect style images to be transformed into content images. In our method, we adopt encoder-decoder subnetwork and an auto-encoder reconstruction loss to guarantee that the outputs have consistent semantic information with the content images. With auto-encoder reconstruction loss, our algorithm achieves one-sided mapping, which needs fewer parameters and can be easily generalized for multiple styles.

The proposed adversarial gated networks (Gated-GAN) realize the transfer of multiple artists or genre styles in a single network (see Figure 4.1). Different to the conventional encoder-decoder architectures in [61, 113, 183], we additionally consider a gated-transformer network between the encoder and decoder consisting of multiple gates, each corresponding to one style. The gate controls which transformer is connected to the model so that users can switch gate to choose between different styles. If the gated transformer is skipped, the encoder and decoder are trained as an auto-encoder to preserve semantic consistency between input images and their reconstructions. At the same time, the mode collapse issue

is avoided and the training procedure is stabilized. The gated transformer also facilitates generating new styles through weighted connections between the transformer branches. Our discriminative network architecture has two components: the first to distinguish synthesized images from genuine images, and the other to identify the specific styles of these images. Experiments demonstrate that our adversarial gated networks successfully achieve multi-collection style transfer with a quality that is better or at least comparable to existing methods.

The remainder of this chapter is organized as follows. In Section 2, we summarize related work. The proposed method is detailed in Section 3. The results of experiments using the proposed method and comparisons with existing methods are reported in Section 4. We conclude in Section 5.

4.2 Related Work

In this section, we introduce related style transfer works. We classify style transfer methods into four categories: texture synthesis-based methods, optimization-based methods, feedforward network-based methods, and adversarial network-based methods.

4.2.1 Traditional Texture Transfer Method

Style transfer is an extension of texture transfer, the goal of the latter being to render an object with a texture taken from a different object [5, 35, 52, 74]. Most previous texture transfer algorithms relied on texture synthesis methods and low-level image features to preserve the target image structure. Texture synthesis is the process of algorithmically constructing an unlimited number of images from a texture sample. The generated images are perceived by humans to be of the same texture but not exactly like the original images. A large range of powerful parametric and non-parametric algorithms exist to synthesize photo-realistic natural texture [1, 34, 41]. Based on texture synthesis, [36] and [38] used segmentation and patch matching to preserve information content. However, the texture transfer methods use only low-level target image features to inform texture transfer and take a long time to migrate a style from one image to another.

4.2.2 Optimization-based Methods

The success of deep CNNs for image classification [69, 138] prompted many scientists and engineers to visualize features from a CNN [96]. DeepDream [138] was initially invented to help visualize what a deep neural network sees when given an image. Later, the algorithm became a technique to generate artworks in new psychedelic and abstract forms. Based on image representations derived from pre-trained CNNs, Gatys *et al.* [40] introduced a neural style transfer algorithm to separate and recombine image content and style. This approach has since been improved in various follow-up papers. Li *et al.* [76] studied patch-based style transfer by combining generative Markov random field (MRF) models and the pre-trained CNNs. Selim *et al.* [127] extended this idea to head portrait painting transfer by imposing novel spatial constraints to avoid facial deformations. Luan *et al.* [93] studied photorealistic style transfer by assuming the input to output transformation was locally affine in color space. Optimization-based methods can produce high quality results but they are computationally expensive, since each optimization step requires a forward and backward pass through the pre-trained network.

4.2.3 Feedforward Networks-based Methods

Feedforward network-based methods accelerate the optimization procedure, which first iteratively optimizes a generative model and produces the styled image through a single forward pass. Johnson *et al.* [61] and Ulyanov *et al.* [141] trained a feedforward network to quickly produce similar outputs. Based on [141], Ulyanov *et al.* [142] then proposed to maximize quality and diversity by replacing the batch normalization module with instance normalization. After that, several works explored multi-style transfer in a single network. Dumoulin *et al.* [146] proposed conditional instance normalization, which specialized scaling and shifting parameters after normalization to each specific texture and allowed the style transfer network to learn multiple styles. Huang *et al.* [163] introduced an adaptive instance normalization (AdaIN) layer that adjusted the mean and variance of the content input to match those of the style input. Since the StructAE [112] is able to map input data into nonlinear latent spaces while preserving the local

and global subspace structure, [19] introduced StyleBank, which was composed of multiple convolutional filter banks integrated in an auto-encoder, with each filter bank an explicit representation for style transfer. [80] took a noise vector and a selection unit as input to generate diverse image styles. Although adopting different methods to achieve multi-style transfer, they all explicitly extracted style presentations from style images based on the Gram matrix [40]. Gram matrix based methods could do collection style transfer if they use several images as style. Though those methods are designed to transfer the style of a single image, they could also transfer the style of several images by averaging their Gram matrix statistics of pretrained deep features. On the other hand, our methods learn to output samples in the distribution of the style of a collection. [81] achieved universal-style transfer, by applying the style characteristics from a style image to content images in a style-agnostic manner. By whitening and coloring transformation, the feature covariance of content images could exhibit the same style of statistical characteristics as the style images. In contrast, we are interested in the multi-collection style transfer problem. In contrast, we are interested in the multi-collection style transfer problem. A single image is difficult to comprehensively represent the style of an artist, and thus we study multi-collection style transfer to abstract the style of an artist from a collection of images.

4.2.4 Adversarial Network-based Methods

GANs [44] represented a generative method using two networks, one as a discriminator and the other as a generator, to iteratively improve the model by a minimax game. Chuan *et al.* [77] proposed Markovian GANs for texture synthesis and style transfer, addressing the efficiency issue inherent in MRF-CNN-based style transfer [76]. Spatial GAN (SGAN) [60] successfully achieved data-driven texture synthesis based on GANs. PSGAN [13] improved Spatial GAN to learn periodical textures by extending the structure of the input noise distribution. [136] proposed a max-min method to transform features into a low-dimensional subspace.

By adopting adversarial loss, many works have generated realistic images for conditional image generation, e.g., frame prediction [90], image super-resolution [72], image dehazing [181] and image-to-image translation [59]. However, these

approaches often required paired images as input, which are expensive and hard to obtain in practice. Several studies have been conducted investigating domain transfer in the absence of paired images. [66, 168, 183] independently reported a similar idea of cycle-consistent loss to transform the image from the source domain to the target domain and then back to the original image. Taigman *et al.* [140] proposed Domain Transfer Network, which employed a compound loss function, including an adversarial loss and constancy loss, to transfer a sample in one domain to an analog sample in another domain.

In contrast, some works have generated different image types from noise in a single generative network. One strategy was to supply both the generator and discriminator with class labels to produce class-conditional samples [102]. Another strategy was to modify the discriminator to contain an auxiliary decoder network to output the class label for the training data [107, 123] or a subset of the latent variables from which the samples were generated [23]. AC-GAN [108] added auxiliary multi-class category loss to supervise the discriminator, which was used to generate multiple object types. Our work is different in that it focuses on exploring migrating different styles to content images.

4.3 Gated Generative Model

In this section, we introduce the proposed gated generative model for multi-style transfer. We first consider the style transfer for one domain. We have two sets of unpaired training samples: one set of input images $\{x_i\}_{i=1}^{N} \in X$ and the target set of collections for artist or genre $\{y_i\}_{i=1}^{M} \in Y$. We aim to train a generative network that generates images $G(x)$ in the style of a target artist or genre, and simultaneously we train a discriminative network D to distinguish the transferred images $G(x)$ from the real style image y. The generative network implicitly learns the target style from *adversarial loss*, aiming to fool the discriminator. The whole framework has three modules: an encoder, a gated-transformer, and a decoder. The encoder consists of a series of convolutional layers that transform input image into feature space $Enc(x)$. After the encoder, a series of residual networks [51] become the transformer: $T(\cdot)$. The input of the residual layer in gated function T is the feature maps from the last layer of encoder module $Enc(x)$. The output

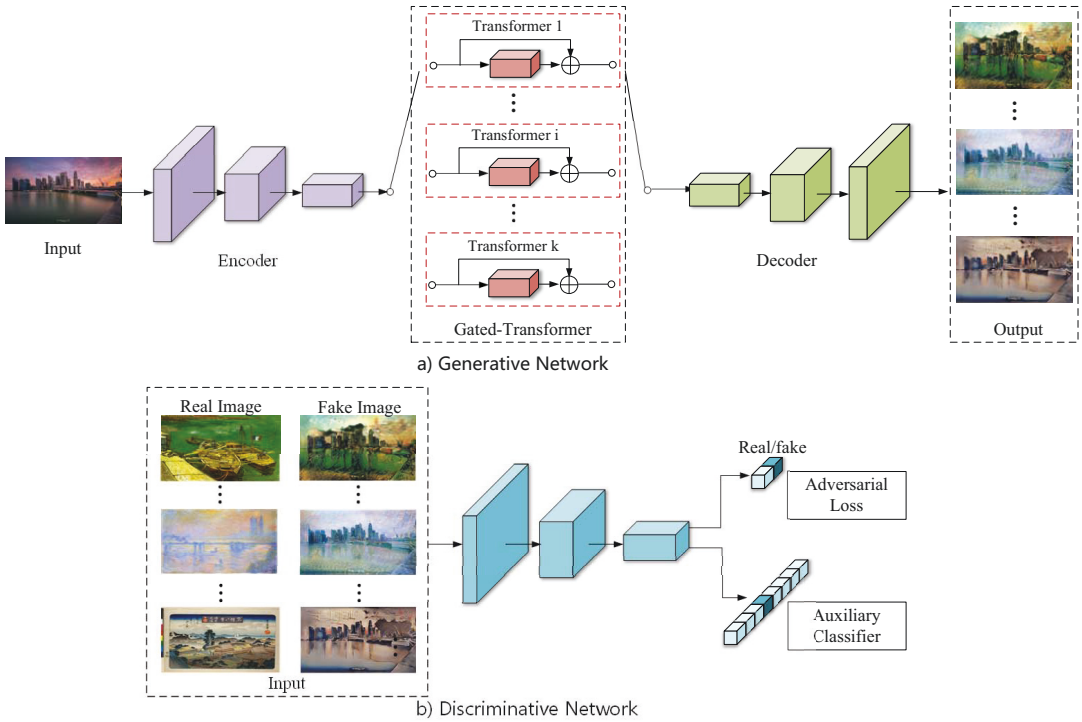

Figure 4.2: The architecture of the proposed adversarial gated networks: a generative network and a discriminative network. The generative network consists of three modules: an encoder, a gated transformer, and a decoder. Images are generated to different styles through branches in the gated transformer module. The discriminative network uses the adversarial loss to distinguish between stylized and real images. An auxiliary classifier supervises the discriminative network to classify the style categories.

of the gated function is the activations $T(Enc(x))$. Then, a series of fractionally-strided convolutional networks decode the transformed feature into output images $G(x) = Dec(T(Enc(x)))$. To stabilize training, we introduce the *auto-encoder reconstruction loss*. We introduce the gated transformer module to integrate multiple styles within a single generated network. The network architecture is shown in Figure 4.2, and the overall architecture is called the adversarial gated network (Gated-GAN).

4.3.1 Adversarial Network for Style Transfer

To learn a style from the target domain Y, we apply adversarial loss [44], which simultaneously trains G and D as the two-player minimax game with loss function $L(G, D)$. The generator G tries to generate an image $G(x)$ that looks similar in style to target domain Y, while the discriminator D aims to distinguish between them. Specifically, we train D to maximize the probability of assigning the correct label to target image y and transferred image $G(x)$, meanwhile training G to minimize the probability of the discriminator assigning the correct label to transferred image $G(x)$. The original generative adversarial value function is expressed as follows:

$$\min_G \max_D V(G, D) = \mathbb{E}_{y \in Y} \left[\log D(y)\right] + \mathbb{E}_{x \in X} \left[\log(1 - D(G(x)))\right]. \tag{4.1}$$

We employ the least squares loss (LSGAN) as explored in [98], which provides a smooth and non-saturating gradient in the discriminator D. The adversarial loss $\mathcal{L}_{GAN}(G, D)$ becomes:

$$\mathcal{L}_{GAN}(G, D) = \mathbb{E}_{y \in Y} \left[(D(y) - 1)^2\right] + \mathbb{E}_{x \in X} \left[D(G(x))^2\right]. \tag{4.2}$$

where the discriminator uses the 0-1 binary coding that 0 and 1 are the labels for fake data and real data.

4.3.2 Auto-encoder Reconstruction Loss for Training Stabilization

The original GAN framework is known to be unstable, as it must train two neural networks with competing goals. [183] pointed out that one reason for instability is that there exist non-unique solutions when the generator learns the mapping function. Due to unpaired training samples, the same set of input images can be mapped to any random permutation of images in the target domain. To reduce the space of possible mapping functions, we introduce the *auto-encoder reconstruction loss*. In our model, the auto-encoder is obtained by directly connecting the encoder and decoder modules. That is, the network is encouraged to produce

output $Dec(Enc(x))$ identical to input image x after learning the representation (encoding: $Enc(x)$) for the input data. We define the L1 loss between the reconstructed output and input as the *auto-encoder reconstruction loss*:

$$\mathcal{L}_R = \mathbb{E}_{x \in X} \left[||Dec(Enc(x)) - x||_1 \right]. \tag{4.3}$$

Mode collapse is a common problem in vanilla GAN [3], where all input images might be mapped to the same output image, and the optimization fails to make progress. In collection style transfer, if the networks trained with adversarial loss alone have sufficient capacity, content images would be mapped to an arbitrary output as long as it matches the target style. The proposed encoder-decoder subnetwork aims to reconstruct input images so that structures of the output are expected to be consistent with the input image, which guarantees the diversity of the output along with different inputs.

4.3.3 Adversarial Gated Network for Multi-Style Transfer

4.3.3.1 Gated Generated Network

In multi-style transfer, we have a set of input images $\{x_i\}_{i=1}^{N} \in X$ and collections of paintings $Y = \{Y_1, Y_2, ..., Y_K\}$, where K denotes number of collections. In each collection, we have M_c numbers of images $\{y_i\}_{i=1}^{M_c} \in Y_c$, where c indicates the index of collection. The proposed gated generative network aims to output images $G(x, c)$ by assigning specific style c. Specifically, the gated-transformer (red blocks in Figure 4.2) transforms the input from encoded space into different styles by switching trigger to different branches:

$$G\left(x, c\right) = Dec\left(T\left(Enc(x), c\right)\right). \tag{4.4}$$

In each branch, we employ the residual network as the transfer module. The encoder and decoder are shared by different styles, so the network only has to save the extra transformer module parameters for each style.

4.3.3.2 Auxiliary Classifier for Multiple Styles

If we only use the adversarial loss, the model tends to confuse and mix multiples styles together. Therefore, we need a supervision to separate categories of styles. One solution is to adopt LabelGAN [123]. [123] generalized binary discriminator to multi-class case with its associated class label $c \in \{1, \cdots, K\}$, and the $(K+1)$-th label corresponds to the generated samples. The objective functions are defined as:

$$\mathcal{L}_G^{lab} = \mathbb{E}_{x \in X} \left[H \left([1, 0], [D_r \left(G(x) \right), D_{K+1} \left(G(x) \right)] \right) \right], \tag{4.5}$$

$$\mathcal{L}_D^{lab} = \mathbb{E}_{(y,c) \in Y} \left[H(v(c), D(y)] + \mathbb{E}_{x \in X} \left[H(v(K+1), D(G(x)) \right] \right. \tag{4.6}$$

where denotes the probability of the sample x to have the i-th style. $D(x) = [D_1(x), D_2(x), \cdots, D_{K+1}(x)]$ and $v(c) = [v_1(c), \ldots, v_{K+1}(c)]$ with $v_i(c) = 0$ if $i \neq c$ and $v_i(c) = 1$ if $i = c$. H is the cross-entropy, defined as $H(p, q) = -\sum_i p_i \log q_i$. In LabelGAN, the generator gets its gradients from the K specific real class logits in the discriminator and tends to refine each sample towards being one of the classes. However, LabelGAN actually suffers from the overlaid-gradient problem [180]: all real class logits are encouraged at the same time. Though it tends to make each sample be one of these classes during the training, the gradient of each sample is a weighted averaging over multiple label predictors.

In our method, an auxiliary classifier (denoted as C) is added in the consideration of leveraging the side information directly:

$$\mathcal{L}_G^{Gated} = \lambda_{CLS} \mathbb{E}_{x \in X} \left[H(u(c), C(G(x, c)) \right] + \mathcal{L}_{GAN} \tag{4.7}$$

where $u(\cdot)$ is the vectorizing operator that is similar to $v(\cdot)$ but defined with K classes, and $C(G(x, c))$ is the probability distribution over K real classes given by the auxiliary classifier. L_{GAN} indicates the adversarial loss (in Equation 4.2) that encourages to generate realistic images. In the first term of Equation 4.7, we optimize entropy to make each sample have a high confidence of being one of the classes, so that the overlaid-gradient problem can be overcome. The loss can

be written in the form of log-likelihood:

$$\min_{C} \mathcal{L}_{CLS}(C) = -\mathbb{E}_{(y,c)\in Y} \left\{ \log C(Style = c|y) \right\}. \tag{4.8}$$

The classifier C is encouraged to correctly predict the log-likelihood of the correct class given real images. Meanwhile, the generator aims to generate images that can be correctly recognized by classifier:

$$\min_{G} \mathcal{L}_{CLS}(G) = -\mathbb{E}_{x\in X} \left\{ \log C(Style = c|G(x,c)) \right\}. \tag{4.9}$$

In practice, the classifier shares low-level convolutional layers with the discriminator, but they have exclusive fully connected layers to output the conditional distribution.

4.4 Implementation

4.4.1 Network Configuration

Our generative network architecture contains two stride-2 convolutions (encoder), one gated residual blocks (gated-transfer), five residual blocks, and two fractionally-convolutions with $\frac{1}{2}$ stride (decoder). Instance normalization [123] is used after the convolutional layers. Details are provided in Table 4.1.

For the discriminators and classifiers, we adapt the Markovian Patch-GAN architecture [58, 77, 168, 183]. Instead of operating over the full images, the discriminators and classifiers distinguish overlapping patches, sampling from the real and generated images. By doing so, the discriminators and classifiers focus on local high-frequency features like texture and style and ignore the global image structure. The patch size is set to 70×70. Also, PatchGAN has fewer parameters and can be applied to any size of inputs.

Table 4.1: Generative Network of Gated-GAN

	Operation	Kernel size	Stride	Feature maps
Encoder	Convolution	7	1	32
	Convolution	3	2	64
	Convolution	3	2	128
Gated-transformer	Residual block			128
Decoder	Residual block			128
	Residual block			128
	Residual block			128
	Residual block			128
	Residual block			128
	Frac-convolution	3	1/2	64
	Frac-convolution	3	1/2	32
	Convolution	7	1	3

4.4.2 Training Strategy

To smooth the generated image $G(x, c)$, we make use of the *total variation loss* [2, 61, 121], denoted by $\mathcal{L}_{TV}$:

$$\mathcal{L}_{TV} = \sum_{i,j} \left[(G(x)_{i,j+1} - G(x)_{i,j})^2 + (G(x)_{i+1,j} - G(x)_{i,j})^2 \right]^{\frac{1}{2}} \tag{4.10}$$

where $i \in (0, \cdots, H-1)$ and $j \in (0, \cdots, W-1)$ and $G(x)$ is the generated image whose dimension is $H \times W$. The full objective of the generator is minimizing the loss function:

$$\mathcal{L}(G) = \mathcal{L}_{GAN} + \lambda_{CLS}\mathcal{L}_{CLS} + \lambda_{TV}\mathcal{L}_{TV} \tag{4.11}$$

where λ_{CLS} and λ_{TV} are parameters that control relative importance of their corresponding loss functions. Alternatively, we train an auto-encoder by minimizing the weighted reconstruction loss in Equation 4.3: $\lambda_R\mathcal{L}_R$. The discriminator maximizes the prediction of real images and generated images $\mathcal{L}(D) = \mathcal{L}_{GAN}$, while the classifier in Equation 4.8 maximizes the prediction of collections from different artists or genres. For all experiments, we set $\lambda_{CLS} = 1$, $\lambda_R = 10$, and $\lambda_{TV} = 10^{-6}$. The networks are trained with a learning rate of 0.0002, using the Adam solver [67] with the batch size of 1.

The input image is 128×128. The training samples are first scaled to 143×143, and then randomly flipped and cropped to 128×128. We train our model with an input size of 128×128 for two reasons. First, randomly cropping raw input could augment the number of the training set. Secondly, relatively smaller size of the image decreases the computational cost, so that speeds up training procedure. In the test phase, We test images with their original resolution to receive a clearer exhibition.

To stabilize training, we update the discriminative networks using a history of transferred images rather than the ones produced by the latest generative network [129]. Specifically, we maintain an image buffer that stores 50 previously generated images. At each iteration of discriminator training, we compute the discriminator loss function by sampling images from the buffer. The training process is shown in Algorithm 3. θ_{Enc} denotes the parameter of encoder module and θ_{Dec} denotes the parameters of decoder module. In practice, K_g and K_d are set to 1.

Algorithm 2 Algorithm for training the gated generative network.

Require: The set of training sample $\{x_i\}_{i=1}^{N} \in X$, The set of style images with category $\{y_i, c_i\} \in Y$, number of discriminator network updates per step K_d, number of generative network updates per step K_g.

Ensure: Gated generative newtworks:

$$G = Dec(T(Enc(\cdot), \cdot)).$$

for number of training iterations **do**
  for K_d steps **do**
    Sample minibatch of style images (y_i, c_i) and training images x_i.
    Generate stylized image $G(x_i, c_i)$ in Equation 4.4.
    Update discriminator D and classifier C

$$\Delta_{\theta_D} \leftarrow \nabla_{\theta_D} \mathcal{L}_{GAN},\ \Delta_{\theta_C} \leftarrow \theta_D \nabla_{\theta_C} \mathcal{L}_{CLS}.$$

  end for
  for K_g steps **do**
    Sample training images x_i
    Update generator G :

$$\Delta_{\theta_G} \leftarrow \nabla_{\theta_G}(\mathcal{L}_{GAN} + \lambda_{CLS}\mathcal{L}_{CLS} + \lambda_C \mathcal{L}_C + \lambda_{TV}\mathcal{L}_{TV}).$$

  end for
  Update encoder and decoder module $\theta_{Enc}, \theta_{Dec}$:

$$\Delta_{\theta_{Enc}, \theta_{Dec}} \leftarrow \nabla_{\theta_{Enc}, \theta_{Dec}}(\lambda_R \mathcal{L}_R).$$

end for

4.5 Experiments

In this section, we evaluate the effectiveness, stability, and functionality of the proposed model. We first introduce a quantitative assessment of image quality. Then, we set up a texture synthesis experiment and visualize the filters in the gated transformer branches. Lastly, we train the model for multiple style transfer and compare results with state-of-the-art algorithms.

4.5.1 Assessment of Image Quality

We use FID score [53] to quantitatively evaluate the quality of results. FID score measures the distance between the generated distribution and the real distribution. To this end, the generated samples are first embedded into a feature space given by (a specific layer) of Inception Net. Then, taking the embedding layer as a continuous multi-variate Gaussian, the mean and covariance are estimated for both the generated data and the real data. The Fréchet distance between these two Gaussians is then used to quantify the quality of the samples, i.e.,

$$FID(x, g) = \|\mu_x - \mu_g\|_2^2 + Tr(\Sigma_x + \Sigma_g - 2(\Sigma_x\Sigma_g)^{\frac{1}{2}}) \tag{4.12}$$

where (μ_x, Σ_x) and (μ_g, Σ_g) are the mean and covariance of sample embeddings from the real data distribution and generative model distribution, respectively. In our experiment, we use paintings of artists as samples of real distribution and stylized images as samples of generated distribution. That is to say, we compute the FID between generated images and authentic work of painting.

4.5.2 Texture synthesis

To explicitly understand the gated-transformer module in the proposed Gated-GAN, we design an experiment to explore what the gated-transformer learns. We use our Gated-GAN to achieve synthesize texture and visualize the gated-transformer filters. For each style, the training set is a textured image. The training samples are first scaled to 143×143, and then randomly flipped and cropped to 128×128. The generative network input is Gaussian noise. After

Figure 4.3: Four cases of texture synthesis using Gated-GAN. For each case, the first column shows examples of texture, and the other three are synthesized results given different samples of Gaussian noise as inputs.

Figure 4.4: Visualization of learned features in the gated transformer of the generative networks. In each case, the left shows synthesized images and the right shows the corresponding features.

adversarial training, the generative network outputs realistic textured images (see Figure 4.3).

To explore style representations learned from the gated-transformer, we visualize the transformer filters in Figure 4.4. The features are decoded by $3 \times 3 \times 128$ tensors, where only one of the 128 channels is activated by Gaussian noise. They passed through different gated transformer filters but the same decoder. Since the output of the decoder contains three channels (RGB channels), we can observe the color of output decoded from the learned feature. This reveals that the transformer module learns style representations, e.g., color, stroke, etc. Another interpretation is that the transformer module learns the bases or elements of styles. Generated images can be viewed as linear combinations of these bases, with coefficients learned from the encoder module.

Figure 4.5: Collection style transfer on Photo → Monet. From left to right: input photos, Monet's paintings picked from a similar landscape theme, and our stylized images. The photo is transferred adaptively based on different themes.

4.5.3 Style Transfer

In this subsection, we present our results for generating multiple styles of artists or genres using a single network. Then, we compare our results with state-of-the-art image style transfer and collection style transfer algorithms. The model is trained to generate images in the style of Monet, Van Gogh, Cezanne, and Ukiyo-e, whose datasets are from [183]. Each contains 1073, 400, 526, and 563 paintings, respectively.

4.5.3.1 Multi-Collection Style Transfer

Collection style transfer mimics the style of artists or genres with respect to their features, e.g., stroke, impasto, perspective frame usage, etc. Figure 4.5 shows

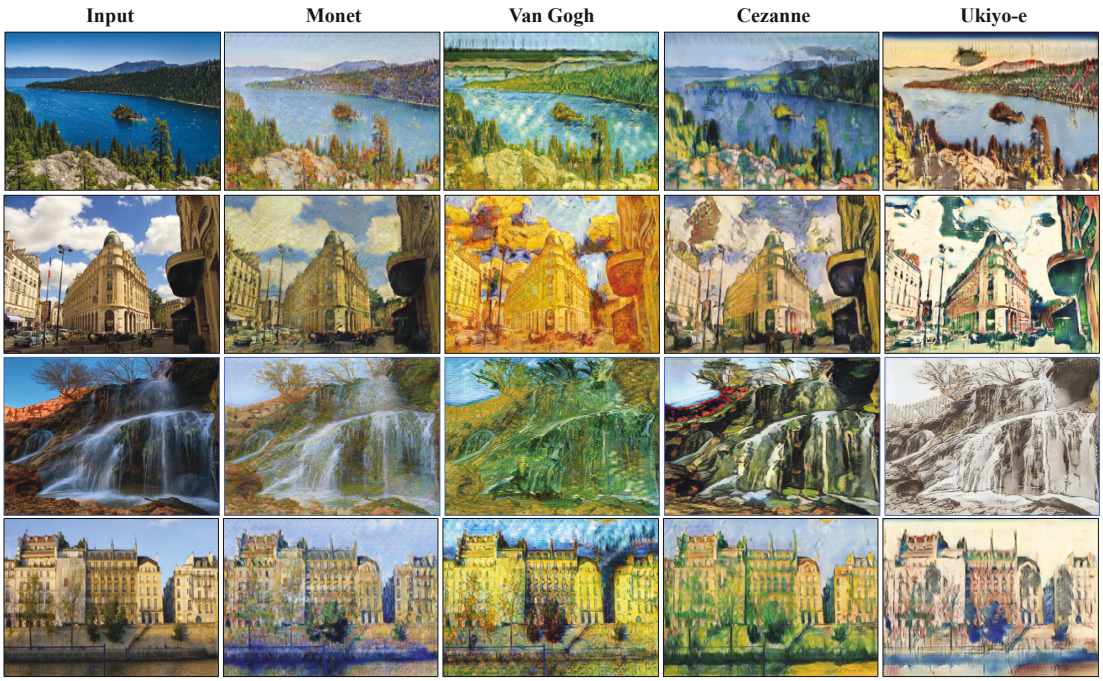

Figure 4.6: A four-style transfer network is trained to capture the styles of Monet, Van Gogh, Cezanne, and Ukiyo-e.

the results of the collection style transfer using our method. Original images are presented on the left, and the generated images are on the right. For comparison, Monet's paintings depicting similar scenes are shown in the middle. It can be seen that the styles of the generated images and their corresponding paintings are similar. Although the themes and colors of the two generated images are different, they still appear similar to Monet's authentic pieces. Our method can clearly mimic the style of the artist for different scenes. Figure 4.6 shows the results of applying the trained network on evaluation images for Monet's, Van Gogh's, Cezanne's, and Ukiyo-e's styles.

4.5.3.2 Comparison with Image Style Transfer

The image style transfer algorithm [40] focuses on producing images that combine the content of an arbitrary photograph and style of one or many well-known artworks. This is achieved by minimizing the mean-squared distance between the entries of the Gram matrix from the style image and the Gram matrix of the image

to be generated. We note some recent works on multi-style transfer [19, 80, 146], but these are all based on neural style transfer [40]. Thus, we compare our results with [40].

For each content image, we use two representative artworks as the reference style images. To generate images in the style of the entire collection, the target style representation is computed by the average Gram matrix of the target domain. To compare this with our method, we use the collections of the artist's artworks or a genre and compute the 'average style' as the target.

Figure 4.7 reports the difference of methods. We can see that Gatys et al. [40] requires manually picking target style images that closely match the desired output. If the entire collection is used as target images, the transferred style is the average style of the collections. In contrast, our algorithm outputs diverse and reasonable images, each of which can be viewed as a sample from the distribution of the artist's style.

4.5.3.3 Comparison with Universal Style Transfer

[81] aims to apply the style characteristics from a style image to content images in a style-agnostic manner. By whitening and coloring transformation, the feature covariance of content images could exhibit the same style of statistical characteristics as the style images without requiring any style-specific training.

We compare images generated from the proposed algorithm and those from [81]. The results are shown in Figure 4.8. Given a picture with bushes and flowers (see Figure 4.8 (a)), our method outputs what Monet might record this scenery (see Figure 4.8 (d)), in which the style of painting bushes and flowers is similar to Monet's painting of "Flowers at Vetheuil". What if the content image is a cityscape? Our method outputs images with foggy strokes (see Figure 4.8 (h)) since Monet produced a lot of cityscapes with fog in London (e.g. "Charing Cross Bridge"). On the other hand, [81] transfers images by following a particular style image. Taking "Flowers at Vetheuil" as the style image, Figure 4.8 (g) produced by [81] well inherits the style of Monet's "Flowers at Vetheuil" with green and red spots. However, Monet might not paint a cityscape with green and red spots as painting flowers.

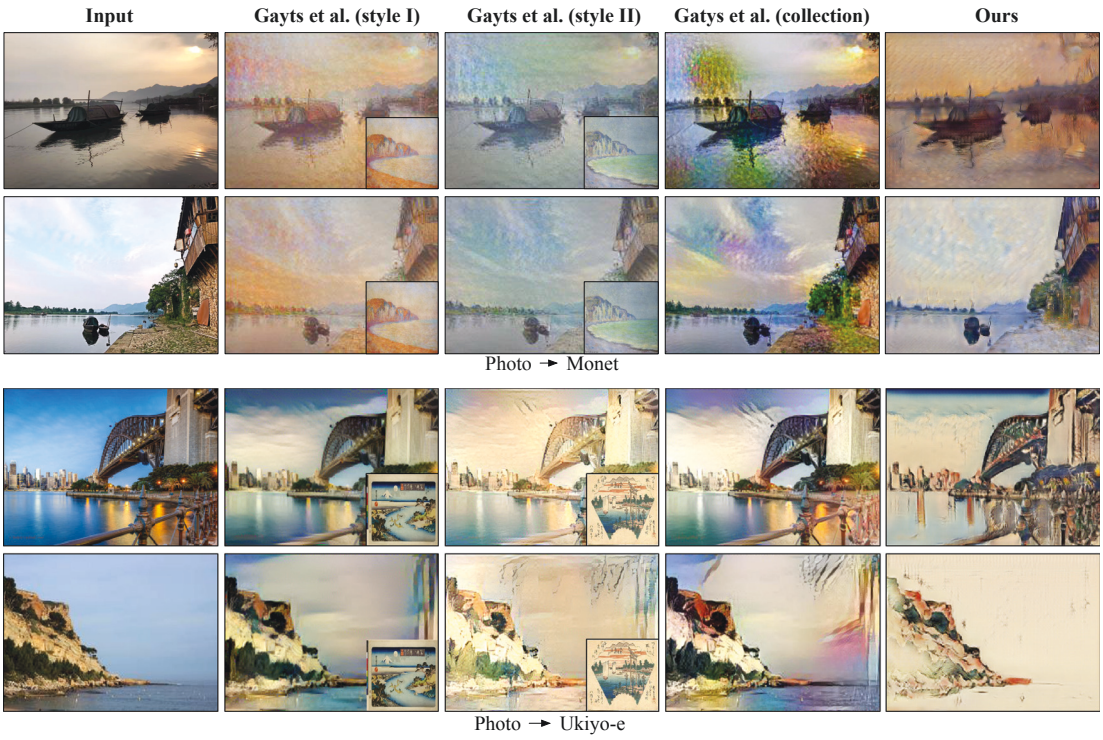

Figure 4.7: Comparison of our methods with image style transfer [40] on photo → Monet and photo → Ukiyo-e. From left to right: input photos, Gatys *et al.*'s results using different target style images, Gatys *et al.*'s results using the entire collection of artist and genre, our results for collection style transfer.

In summary, our task focuses on what the artists or genres might paint given content images, while the task of [81] is to apply style characteristics from a particular style image to any content images. Both [81] and our method output interesting results, and could be used in different scenarios.

4.5.3.4 Comparison with Collection Style Transfer

CycleGAN [183] previously showed impressive results on collection style transfer, so we compare our results with CycleGAN in this section. The generative network of baseline CycleGAN is composed of three stride-2 convolutional layers, 6 residual blocks, two fractional-convolutional layers and one last convolutional layer, which shares the same structure with our method in our experiment. Figure 4.9 demonstrates multi-collection style transfer by our method, which shows

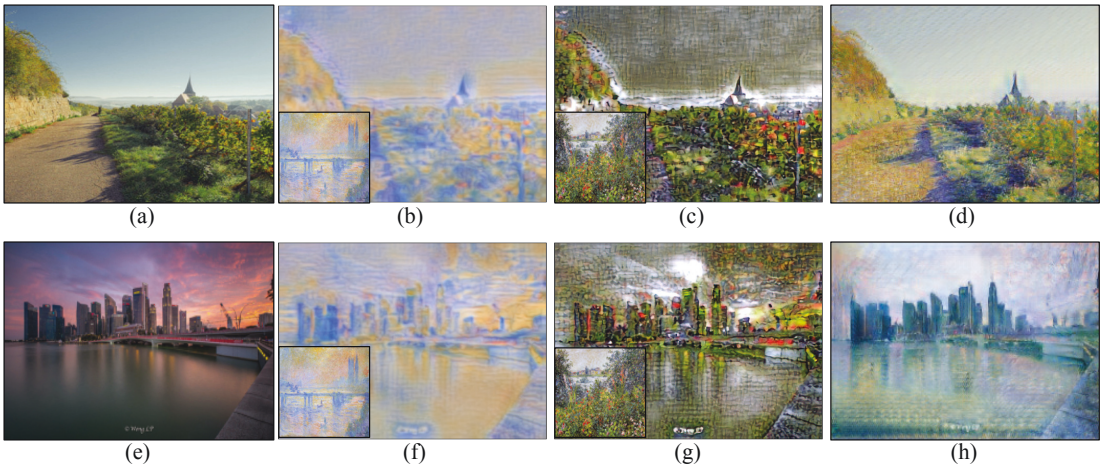

Figure 4.8: Comparison of our methods with universal style transfer [81] on photo → Monet. From left to right: input images, results of [81] with the style image: Monet *Charing Cross Bridge*, results of [81] with the style image: Monet *Flowers at Vetheuil*, and our results of Monet's collection style transfer.

Table 4.2: Quantitative evaluation on collection style transfer in terms of FID to measure the performance. Lower score indicates better quality.

Style	Content Images	CycleGAN [183]	Ours
Monet	86.50	64.14	55.13
Cezanne	186.73	106.96	107.27
Van Gogh	173.01	107.03	109.59
Ukiyoe	195.25	103.36	115.96
MEAN	160.37	95.37	96.99

that the proposed model produces comparable results to CycleGAN.

Quantitative results are shown in Table 4.2, though the quality of images generated from the proposed algorithm exhibits similar performance as those of CycleGAN, it is instructive to note that our four styles are produced from a single network. In the second column of Table 4.2, we compute the score of the corresponding content images of stylized images. We find that the stylized images achieve better performance than original content images. It demonstrates that the stylized images are more similar to the real authentic work of artists, which is consistent with our intuitive expectation.

Finally, we compare the model size with CycleGAN [183]. For fair comparison,

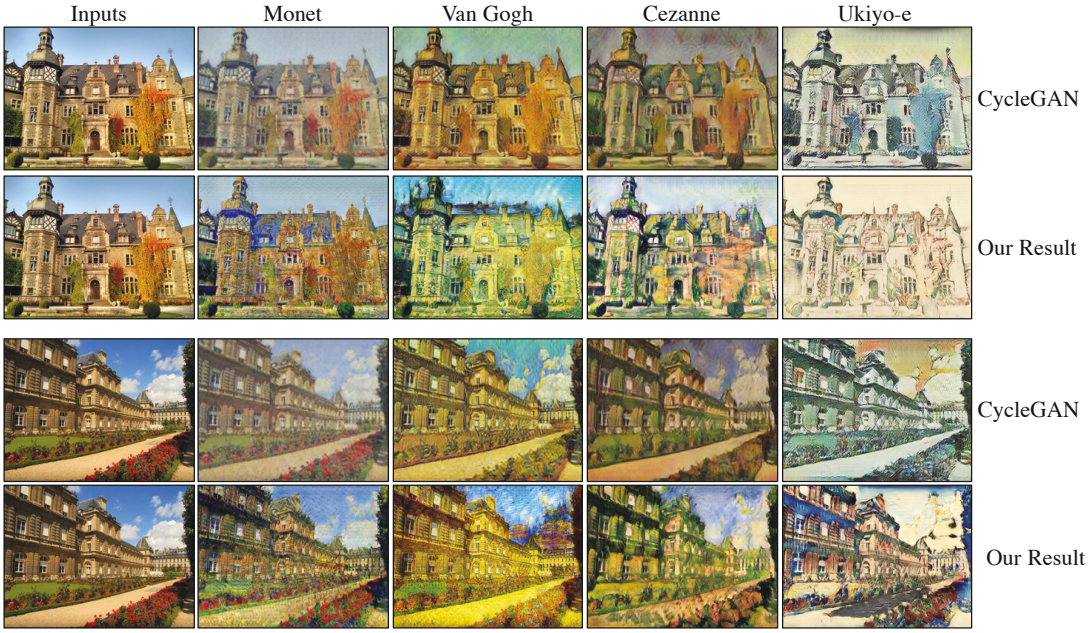

Figure 4.9: Comparison with CycleGAN [183]. From left to right: original images, stylized images in Monet's style, stylized images in Van Gogh's style, stylized images in Cezanne's style, stylized images in Ukiyo-e style. In each case, the first row shows the results produced by CycleGAN, and the second row shows our results.

the proposed Gated-GAN is composed of several convolutional layers and residual blocks with the same architecture as the generative network of CycleGAN. When the transformer module number is set to one, the parameters of the two models are the same. Given another N styles, CycleGAN must train N additional models. A whole generative network must be included for a new style. For Gated-GAN, the transformation operator is encoded in the gated transformer, which only has one residual block. A new style will thus only require a new transformer module (one red block in Figure 4.2 (a)) in the generative network. As a result, the proposed method saves storage space when the style number increases. In Figure 4.10, we compare the numbers of parameters with those of CycleGAN, whose The x-axis indicates style number and the y-axis indicates the model size. It demonstrates that with the increase of style number, only a small portion of parameters is needed in our model.

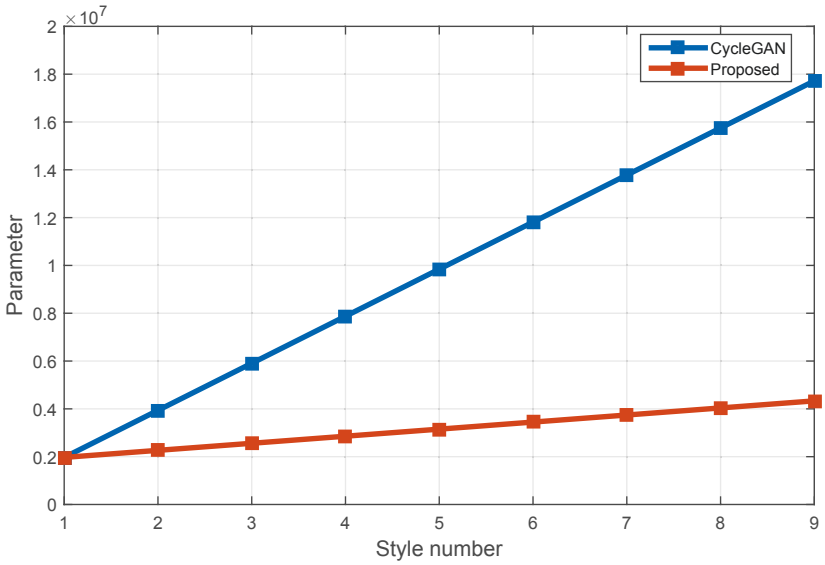

Figure 4.10: Model size. We compare the number of parameters between our model and CycleGAN [183]. The x-axis indicates style number and the y-axis indicates the model size.

4.5.3.5 Comparison with Conditional GAN

Conditional GAN [102, 108] model is a widely used method to generate classconditional image. When the conditional GAN is applied in multi style transfer, a stylized image $G(c, x)$ is generated from a content image x and a style class label c. We compare conditional GAN in experiments. The label of classes is represented by a one-hot vector with k bits where each represents a style type. k noise vectors of the same dimension as the content image are randomly sampled from a uniform distribution. The input of the generative network is obtained by concatenating the content image with the outer product of these noise vectors and the class label.

As we can see in Figure 4.11 (b), the conditional GAN fails to output meaningful results. This is because, in collection style transfer, conditional GAN lacks paired input-output examples. To stabilize the training of conditional GAN, we adopt cycle-consistent loss [183]. From the results of conditional GAN with cycle-consistent loss in Figure 4.11 (c), we can see that the results of different styles tend to be much similar, and only colors are changed at first sight. In contrast,

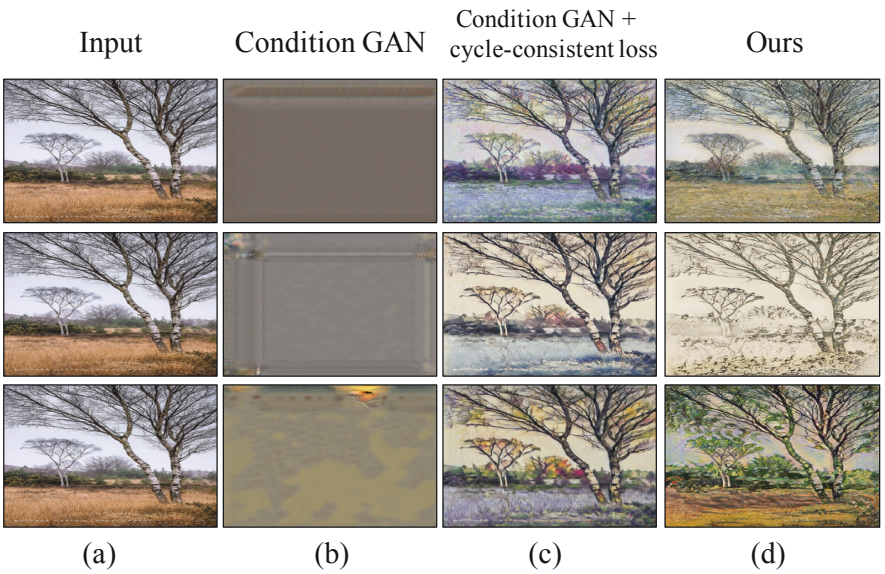

Figure 4.11: Comparison of our methods with Condition GAN and its variant. From left to right: input, condition GAN and condition GAN + cycle-consistent loss. Each row indicates different styles, from top to bottom: Monet, Ukiyo-e, Cezanne.

our results (see Figure 4.11 (d)) are more diverse in different styles in terms of strokes and textures.

4.5.4 Analysis of Loss Function

4.5.4.1 Influence of Parameters in Loss Function

In our model, we propose an auxiliary classier loss and an auto-encoder reconstruction loss, which are balanced by parameters λ_{CLS} and λ_R respectively. Now we analyze the influence of parameters. To explore the influence of parameters, we do experiments by considering $\lambda_{CLS} = \{0, 0.1, 1, 5, 10\}$ and $\lambda_R = \{1, 5, 10, 20\}$.

Figure 4.12 and Table 4.3 demonstrate the qualitative and quantitative comparisons of the influence of parameter $\lambda_{CLS} = \{0, 0.1, 1, 5, 10\}$. We can see that the classifier loss provides supervision of styles. Without classifier loss ($\lambda_{CLS} = 0$), our model will only transfer into one style, as shown in Figure 4.12 (b) that all the outputs through different gated-transformer are the same. We notice that if the weight of the classifier loss is set too large ($\lambda_{CLS} = 10$) in Figure 4.12 (e), the

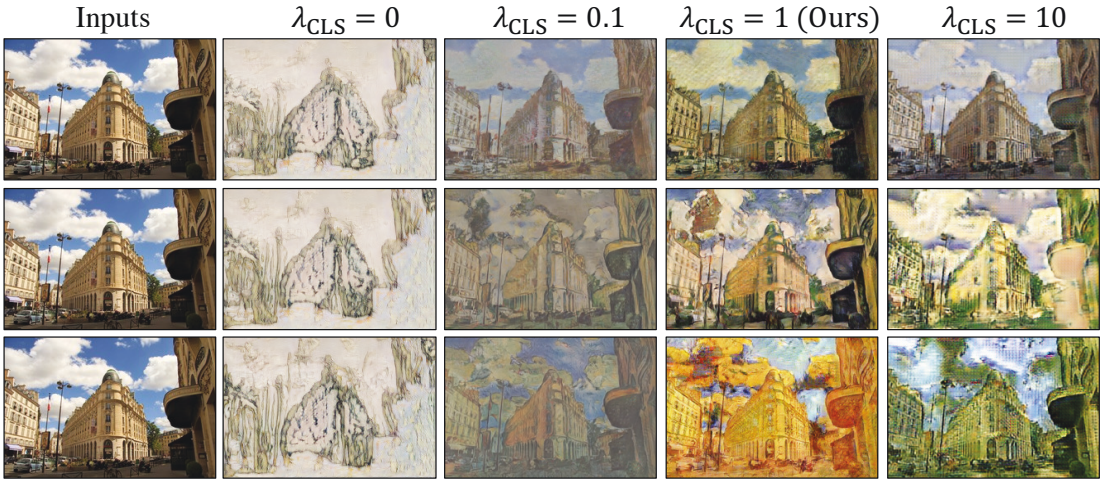

Figure 4.12: Qualitative comparison of the influence of parameter λ_{CLS}. The first column shows the input images. The rest columns demonstrate results with $\lambda_{CLS} = \{0, 0.1, 1, 10\}$. Each row demonstrates images transferred by different styles. From top to bottom: Monet, Cezanne, Van Gogh.

Table 4.3: Quantitative evaluation of parameter $\lambda_{CLS} = \{0, 0.1, 1, 5, 10\}$ in terms of FID score.

Style	$\lambda_{CLS} = 0$	$\lambda_{CLS} = 0.1$	$\lambda_{CLS} = 1$	$\lambda_{CLS} = 5$	$\lambda_{CLS} = 10$
Monet	204.82	63.35	55.13	62.66	61.48
Cezanne	234.02	136.35	107.27	127.77	143.39
Van Gogh	217.10	112.61	109.59	126.56	138.66
Ukiyoe	206.67	138.13	115.96	132.72	140.53
MEAN	215.65	112.61	96.99	112.42	121.02

model would produce images with some artifacts. The underlying reason is that larger classifier loss suppresses the function of the discriminative network so that the output becomes less realistic. We find that our model with $\lambda_{CLS} = 1$ outputs satisfying results. As a result, we set $\lambda_{CLS} = 1$ in our model.

Figure 4.13 and Table 4.4 reveal the qualitative and quantitative comparisons of the influence of parameter $\lambda_R = \{1, 5, 10, 20\}$. We can see that if we set λ_R too small ($\lambda_R = 1$), the outputs tend to be blurry and meaningless (see in Figure 4.13 (b)). It is because, without λ_R, the model cannot capture the structure and content information of the inputs. When $\lambda_R = \{5, 10, 20\}$ the visual qualities

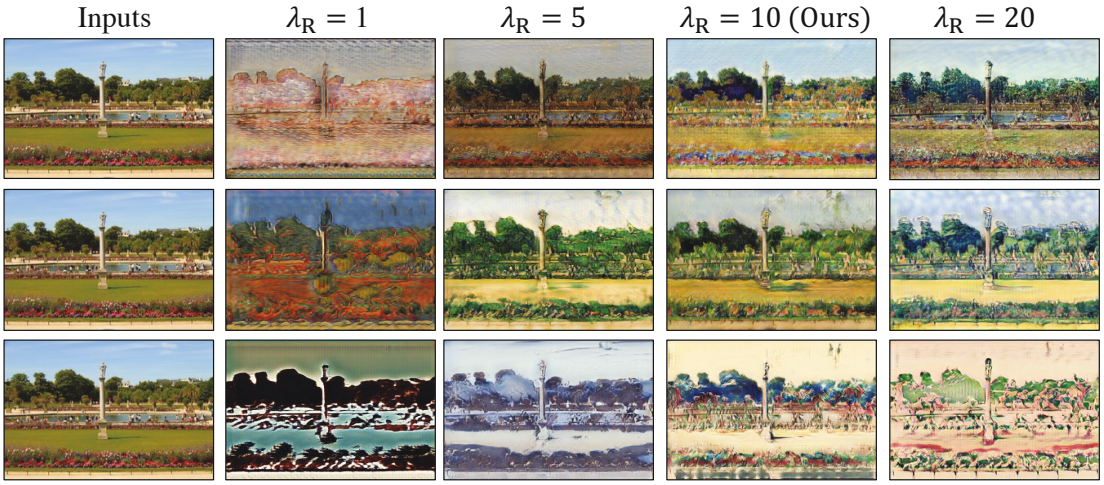

Figure 4.13: Qualitative comparison of the influence of parameter λ_R. The first column shows the input images. The rest columns demonstrate results with $\lambda_R = \{1, 5, 10, 20\}$. Each row demonstrates images transferred by different styles. From top to bottom: Monet, Cezanne, Ukiyo-e.

Table 4.4: Quantitative evaluation of parameter $\lambda_R = \{1, 5, 10, 20\}$ in terms of FID score.

Style	$\lambda_R = 1$	$\lambda_R = 5$	$\lambda_R = 10$	$\lambda_R = 20$
Monet	180.30	121.07	55.13	115.09
Cezanne	165.27	148.67	107.27	140.84
Van Gogh	148.43	139.87	109.59	134.13
Ukiyoe	166.69	134.26	115.96	138.54
MEAN	165.17	135.97	96.99	132.15

are similar while the FID score shows that $\lambda_R = 10$ achieves a slightly better quantitative performance. It demonstrates that our method is robust and easy to reproduce satisfying results. Since $\lambda_R = 10$ achieves the best quality, we set $\lambda_R = 10$ in our model.

4.5.4.2 Analysis of Auto-encoder Reconstruction Loss

We next justify our choice of L1-norm. Beyond L1-norm, L2-norm can also be used in Equation 4.3. In Table 4.5, we find that there is no significant difference between the results of L1 and L2 loss. In CycleGAN [14], L1-norm is used in

Table 4.5: Quantitative evaluation on L1-norm and L2 norm in terms of FID score.

Style	CycleGAN	Ours (L1-norm)	Ours (L2-norm)
Monet	64.14	55.13	56.09
Cezanne	106.96	107.27	101.54
Van Gogh	107.03	109.59	109.33
Ukiyoe	103.36	115.96	112.39
MEAN	95.37	96.99	94.84

Table 4.6: Experiment setup of network architecture analysis

	Expt1	Expt2	Expt3
Encoder	3 × Convolution	3 × Convolution	3 × Convolution
Gated-transformer	1 × Residual block	1 × Convolution	2 × Residual block
Decoder	5 × Residual block 2 ×Fractional-convolution 1 × Convolution	5 × Residual block 2 × Fractional-convolution 1 × Convolution	5 × Residual block 2 × Fractional-convolution 1 × Convolution

cycle-consistent reconstruction loss. As CycleGAN is an important comparison algorithm in our method, we adopt L1-norm in our auto-encoder reconstruction loss as well. Lastly, we analyze the influence of the *auto-encoder reconstruction loss* in stabilizing the adversarial training procedure. We train a comparative model by ignoring the *auto-encoder reconstruction loss* in Equation 4.3. In Figure 4.14, the model without Equation 4.3 generates images with random texture and tends to be less diverse after training for several iterations. In contrast, the full proposed model generates satisfying results. Without the *auto-encoder reconstruction loss*, the network only aims to generate images to fool the discriminative network, which often leads to the well-known problem of mode collapse [3]. Our encoder-decoder subnetwork is encouraged to reconstruct input images, and thus the semantic structure of the input is aligned with that of the output, as well as encourages diversity of outputs. In all, the full model outputs satisfying results.

4.5.5 Analysis of network architecture

We explore the influence of neural network structure. We setup variants of our model in Table 4.6. Variants of models have different configurations of gated-transformer module. The quantitative results in Table 4.7 reveal that the perfor-

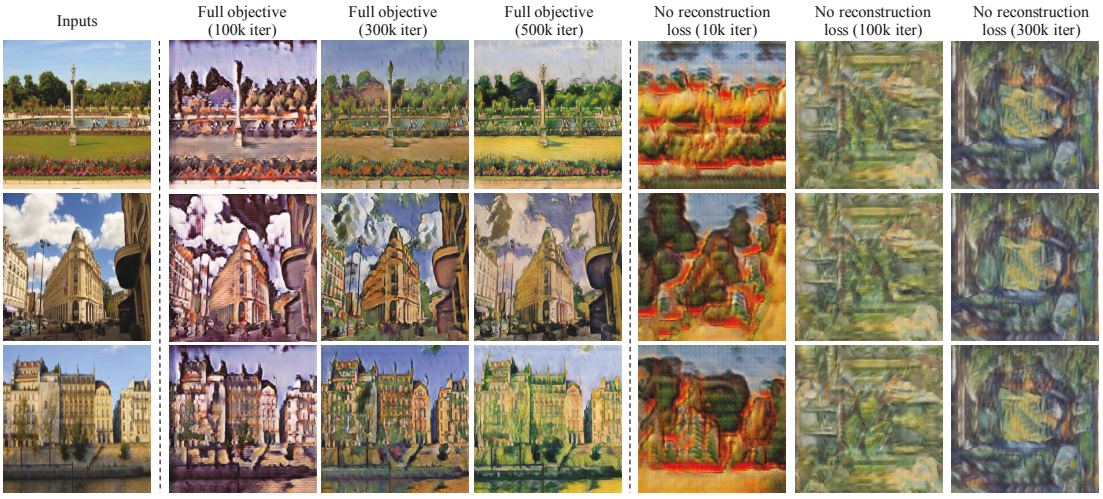

Figure 4.14: Comparison with a variant of our method across different training iterations for mapping images to Cezanne's style. From left to right: original images, results after training for 10k, 100k, and 300k iterations with and without auto-encoder reconstruction loss.

mance of variant 2 declines compared to that of the variant 1. From qualitative results in Figure 4.15, we observe that model of variant 2 cannot maintains content structure (see Figure 4.15 (c)). The underlying reason is that the residual block has a branch that skip the convolutional layer and directly connects between the encoder and decoder module. Since the encoder-decoder subnetwork learns the content information of input from reconstruction loss, residual blocks with skipping connection shuttle the encoded information to the decoder module, which helps our model to output results aligned with the structured of input images.

To analyze the influence of layer size of gated-transformer module, we set variant 3 whose gated-transformer consists of 2 residual blocks. In Table 4.7, we can see the model of variant 3 achieves a slightly better quantitative evaluation than variant 1. The reason is that with the number of residual blocks increasing, the expression capacity of network increases as well, which means the model could capture more details for each style. However, the performance rise of variant 3 is limited and the qualitative qualities are similar in Figure 4.15, which means one residual block of gated-transformer is sufficient in multi style transfer. As a

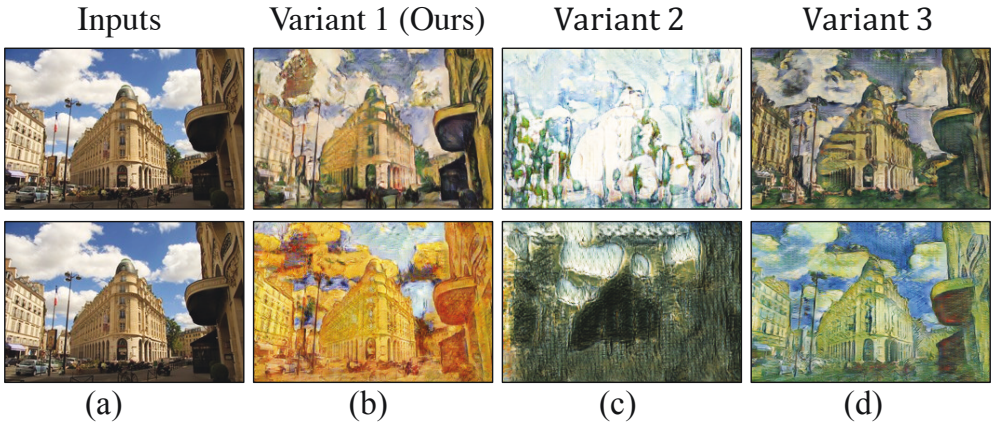

Figure 4.15: Qualitative comparison of the influence of different network structures. The first row is the results of photo → Cezanne, and the second row is the results of photo → Van Gogh.

Table 4.7: Quantitative evaluation on different network structure in terms of FID.

Style	Variants 1(Ours)	Variants 2	Variants 3
Monet	55.13	67.16	53.08
Cezanne	107.27	128.62	110.13
Van Gogh	109.59	199.71	109.07
Ukiyoe	115.96	195.87	100.32
MEAN	96.99	147.84	93.15

result, we adopt variant 1 of architecture as our method.

4.5.6 Incremental Training

By sharing the same encoding/decoding subnets, our model is compatible with adding new styles. For a new style, our model enables to add the style by learning a new branch in the gated-transformer while holding the encoding-decoding subnets fixed. In this subsection, we explore the ability and performance of our proposed model in terms of incremental training for the new style. In this experiment, we first jointly train the encoder-decoder subnetwork and gated-transformer (three collection styles: Cezanne, Ukiyo-e and Van Gogh) with the strategy described in Algorithm 3. After that, for the new style (Monet), we train a new branch of residual blocks in the gated-transformer. Figure 4.16 shows sev-

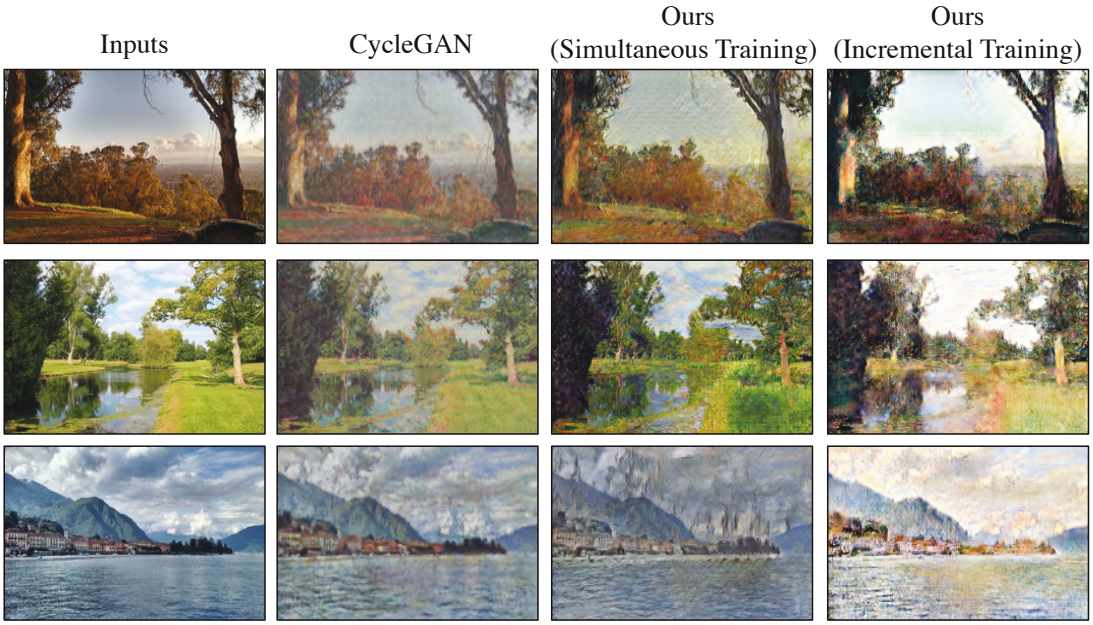

Figure 4.16: Comparison of incremental training. From left to right: original inputs, results of CycleGAN [14], results of our methods that all the styles are trained simultaneously, results of incremental training.

eral results of the newly introduced style by incremental training. It obtains very comparable stylized results to the CycleGAN, which trains the whole network with the style. We also evaluate the quantitative performance of the new style in terms of the FID score. The new style by incremental training gets a score of 57.27. Compared to 55.13 of our Gated-GAN and 64.14 of baseline CycleGAN, the incremental training achieves a competitive result.

4.5.7 Linear Interpolation of Styles

Since our proposed model achieves multi-collection style transfer by switching gates c to different branches $T(Enc(x), c)$, we can blend multiple styles by adjusting the gate weights to create a new style or generate transitions between styles of different artists or genres:

$$\tilde{G}(x, c_1, c_2) = Dec(\alpha \cdot T(Enc(x), c_1) + (1 - \alpha) \cdot T(Enc(x), c_2)) \tag{4.13}$$

Figure 4.17: Style interpolation. The leftmost image is generated in Monet's style, and the rightmost image is generated in Van Gogh's style. Images in the middle are convex combinations of the two styles.

where c_1 and c_2 indicate the gates corresponding to different style branches and α indicates the weight for convex combination of styles. In Figure 4.17, we show an example of interpolation from Monet to Van Gogh with the trained model as we vary α from 0 to 1. The convex combination produces a smooth transition from one style to the other.

4.6 Conclusions

In this chapter, we study multi-collection style transfer in a single network using adversarial training. To integrate styles into a single network, we design a gated network that filters in different network branches with respect to different styles. To learn multiple styles simultaneously, a discriminator and an auxiliary classifier are proposed to distinguish authentic artworks and their styles. To stabilize GAN training, we introduce the auto-encoder reconstruction loss. Furthermore, the gated transformer module provides the opportunity to explore new styles by assigning different weights to the gates. Experiments demonstrate the stability, functionality, and effectiveness of our model and produce satisfactory results compared with a state-of-art algorithm, in which one network merely outputs images in one style.

Chapter 5

Temporal Generaive Model for Long-term Video Frame Synthesis

In this chapter, we investigate video synthesis for long-term video prediction. Video prediction refers to predicting and generating future video frames by observed a set of consecutive frames. Conventional video prediction methods continuously criticize the discrepancy between predicted and ground truth frames in a recursive generation process. As the prediction error accumulates recursively, these methods would easily become out of control and are often confined to the short-term horizon. In this paper, we introduce a retrospection process to rectify the prediction errors beyond criticizing the future prediction. The introduced retrospection process is designed to look back what have been learned from the past and rectify the prediction deficiencies. To this end, we build a retrospection network to reconstruct the past frames given the currently predicted frames. A retrospection loss is introduced to push the retrospection frames being consistent with the observed frames, so that the prediction error is alleviated. On the other hand, an auxiliary route is built by reversing the flow of time and executing a similar retrospection. These two routes interact with each other to boost the performance of retrospection network and enhance the understanding of dynamics across frames, especially for the long-term horizon. An adversarial loss is

employed to generate more realistic results in both prediction and retrospection process. Extensive experiments on the natural video dataset demonstrate the advantage of introducing retrospection process in long-term video prediction.

5.1 Introduction

Video prediction refers to predicting future video frames by observing a sequence of video frames. Accurately generating future frames is important in video coding, video completion, robotics, autonomous driving and intelligent agents that interact with their environment. For example, self-driving cars need to predict the passing vehicles. This prediction capability is also vital for autonomous systems in tasks of path planning and interaction with humans. In recent years, video prediction has attracted increasing attention from the computer vision community and improved the performance based on the deep neural networks and generative adversarial networks [82, 144, 145, 159].

To predict reasonable future frame in natural videos, different distance metrics have been introduced to measure the discrepancy between predicted frames and ground-truth frames, (e.g., Euclidean distance [25, 37, 109]) , and the discrepancy between the distribution of predicted frames and ground-truth frames (e.g., KL distance [144], and Wasserstein distance [82]). For instance, [25, 37, 109] trained predicted models by minimizing mean squared error (MSE) between the next ground-truth frame and the predicted next frame. Since there are many possible futures, the model with MSE tends to output an average over many possible images, and causes a blurry result. To achieve sharper and more realistic results, MCNET [144] combined adversarial loss [44] and MSE loss in the video prediction framework. DRNET [29] designed an adversarial training strategy to disentangle the motion and content representations, so that the video prediction is achieved by a combination of the extracted content features and the predicted motion features.

These existing methods often execute in a recursive manner by taking the newly generated frames as observations to generate subsequent frames. They usually produce high quality predictions for the first few steps. But the prediction would then dramatically degrade, and could even lead to totally missing the video

context or keeping a stationary frame. Looking at the future, these methods continuously criticize the discrepancy between predicted and ground truth frames in a recursive generation process without pause, and the prediction error would accumulate and gradually become difficult to control. In addition to criticizing the future prediction, we are better to retrospect what has been learned from the past and rectify any deficiencies. The proposed retrospection process is used to alleviate the prediction error accumulation between the predicted frames and the ground-truth frames. The introduced retrospection process provides an auxiliary loss to improve the quality of predicted frames. If the predictions are not accurate or in high quality, the retrospection network is hard to well reconstruct past frames. As a result, to minimize the difference between the retrospection frames and the original frames in the retrospection module, the predictive network is encouraged to produce high-quality predicted frames.

In this chapter, we propose to achieve long-term video predictions by criticizing the future and retrospecting the past. Instead of taking video prediction as a single feed-forward process, we suggest that a qualified generator would also be able to predict videos in a backward manner. A new retrospection network is developed to reconstruct the past frames given the currently predicted frames. During the prediction process, a prediction loss is employed to minimize the discrepancy between the predicted and the ground-truth frames. After predicting in a few time steps, we pause the prediction process and look back to rectify the prediction deficiencies in a backward manner. A retrospection loss is introduced to minimize the distance between the retrospected frames and the original frames. Also, we build an auxiliary route to train the retrospection network, where retrospection network first generates a few past frames in backward, then pauses to check if the generated frames could predict the future accurately by the prediction network. Standard feed-forward prediction networks can be flexibly integrated with the proposed video retrospection network to improve the training stage of the video generator. The retrospection operation will be dropped in the test, so that test efficiency can be preserved.

We conducted both qualitative and quantitative experiments on three natural video datasets, e.g., the KTH [126], Weizmann [45] and UCF-101 datasets [132]. Experiments demonstrate that the proposed algorithm can boost the visual qual-

ity of generated videos and lead to more precise results in long-term prediction, which significantly outperforms prior arts.

5.2 Related Work

Many deep learning techniques for video prediction have emerged recently. We review relevant studies related to video prediction using deep neural networks. Over the last few years, CNNs and recurrent neural networks (RNNs) have gained huge popularity and a number of studies [17,62,91,162] applied CNNs and RNNs to predict future frames from an image sequence. The video prediction problem was initially studied at the patch level containing synthetic motions [103, 135, 137]. [114] adopted a discrete vector quantization approach to performing patch-level video prediction. However, predicting patches encounters the well-known aperture problem that causes blockiness as prediction advances in time.

In pixel level, a lot of works have emerged on video prediction since convolutional LSTM (Conv-LSTM) [128] has successfully been applied in a large variety of computer vision research area [9, 111, 176]. Several more studies [17, 37, 91] adopted convolutional LSTM to take spatial and temporal contexts into account. [99] proposed a video generation framework which utilized the Conv-LSTM to encode short-term and long-term spatial-temporal context for semantic video generation using captions. [170] combined the ConvLSTM into a deep generative model which modeled the factorization of the joint likelihood of inputs in the form of video data. [17, 37, 91] adopted convolutional LSTM to take spatial and temporal contexts into account. [99] proposed a video generation framework which utilized the Conv-LSTM to encode short-term and long-term spatial-temporal context for semantic video generation using captions. [170] combined the ConvLSTM into a deep generative model which modeled the factorization of the joint likelihood of inputs in the form of video data.

On the other hand, some methods managed to ease the task by introducing various prior knowledge. One popular hypothesis is that a video sequence could be decomposed as content and motion. By independently modeling the motion and content, MCNET [144] predicted the next frame by combining the predicted motion feature and the extracted content feature. DRNET [29] designed an ad-

versarial training strategy to disentangle the motion and content representations. Another assumption is that the outcome of an event is stochastic as a consequence of the latent events, so different possible future for each sample of its latent variables can be predicted [7]. [159] incorporated a two steam generation architecture to deal with high frequency and low-frequency video content separately so as to output structured prediction. [155] produced a mask that outlines the predicted foreground object to achieve a better quality of prediction.

Other methods exploited external labeled notation to facilitate future prediction. For example, [109] developed an action-conditioned video prediction framework that utilized action prior knowledge as well as previous appearance information to predict futures in the game. [82] utilized labeled optical flow to simultaneously solve both video prediction and optical flow estimation. [97] took advantage of the scene geometry and use the predicted depth for generating the next frame prediction. [85] made use of geometry-aware deep network model by accessing camera intrinsic parameters to predict next frames of cityscapes. The mentioned methods used annotations to facilitate video prediction, such as action annotations, optical flow, and skeleton information. The annotated labels are required manually annotations. Some of the labels such as optical flow can be estimated by algorithms, but the estimation error would decrease the quality of the generated frame.

Traditional video prediction networks are trained by minimizing mean square error (MSE) between predicted and ground truth frames [25, 37, 109]. However, since there are many possible futures, the model with MSE tends to output an average over many possible images and causes a blurry result. Generative adversarial networks (GANs) [44] is an implicit generative model, implemented by a two-player game: a generator and a discriminator. The generator aims to generate realistic images to fool the discriminator, while the discriminator aims to classify whether the images are real or fake. The idea of an adversarial loss that forces the generated images to be, in principle, indistinguishable from real images. This is particularly powerful for image generation tasks, as this is exactly the objective that much of computer graphics aims to optimize. After the invention of adversarial training, many studies applied this scheme to generate images in the context of image generation [16], image-to-image translation [22,79,183], text-

Table 5.1: Notation Table

Notation	Description
$\mathbf{x}_t$	ground-truth frame at time t
$\hat{\mathbf{x}}_t^G$	frame predicted by network G given the true past frames
$\tilde{\mathbf{x}}_t^G$	frame predicted by network G given the generated past frames
$\hat{\mathbf{x}}_t^F$	frame generated by network F given the true past frames
$\tilde{\mathbf{x}}_t^F$	frame generated by network F given the predictions
$\mathbf{x}_{t_1:t_2}$	concatenation of frames $\mathbf{x}_t$ from t_1 to t_2
$[\mathbf{x}_{t_1}, \mathbf{x}_{t_2}]$	concatenation of frame $\mathbf{x}_{t_1}$ and frame $\mathbf{x}_{t_2}$
$[\mathbf{x}_{t_1:t_2}, \mathbf{x}_{t_3:t_4}]$	concatenation of frames from t_1 to t_2 and t_3 to t_4

to-image generation [117], etc. In video prediction area, MCNET [100] proposed adversarial training with multi-scale convolutional networks to generate sharper predictions. [88] exploited spatial and motion constraints in addition to intensity and gradient losses. They computed optical flow through FlowNet [32] and the flow information is used to predict temporally consistent frames. On the other hand, stochastic video prediction methods have emerged that output samples of possible future distribution. The stochastic video prediction methods are a promising solution to obtain realistic and sharp future frames, e.g., SAVP [73] and SVG [28].

In this work, we observe that existing video prediction methods often execute in a single recursive manner, and the prediction error would accumulate over time. Therefore, we introduce a retrospection process to look back what has been learned and rectify the prediction deficiencies. CycleGAN [183] brings the translated images back to the original image. While the high-level idea is similar, we explore the retrospection idea in video prediction, and a novel formulation has been developed.

5.3 Temporal Generative Model

Figure 5.1 illustrates our temporal generative model for long-term video prediction. Our model includes two direction mappings: prediction process and retrospection process. The prediction process aims to predict the subsequent frames while the retrospection process aims to reconstruct the observed frames

in a reverse chronological order. Let $\mathbf{x}_t \in \mathrm{R}^{w \times h \times c}$ denote the t-th frame in an input video, where w, h, and c denote the width, height, and number of channels, respectively. Assuming that we have T observed frames $\{\mathbf{x}_1, \cdots, \mathbf{x}_T\}$, the prediction frame at $(T+1)$ time index is produced by a video prediction model $\hat{\mathbf{x}}^G_{T+1} = G(\mathbf{x}_{1:T})$. Then the subsequent frames is generated by observing the predicted frames recursively (see blue lines). That is to say, at $(T+k)$ time index, the prediction $\hat{\mathbf{x}}_{T+k}$ is achieved by observing both the ground-truth frames $\{\mathbf{x}_1, \cdots, \mathbf{x}_T\}$ and predicted frames $\left\{\hat{\mathbf{x}}^G_{T+1}, \cdots, \hat{\mathbf{x}}^G_{T+k-1}\right\}$. We express the formula as:

$$\hat{\mathbf{x}}^G_{T+k} = \begin{cases} G(\mathbf{x}_{1:T}), & k = 1 \\ G([\mathbf{x}_{1:T}, \hat{\mathbf{x}}_{T+1:T+k-1}]), & 1 < k \leqslant K. \end{cases} \tag{5.1}$$

As a consequence, the quality of subsequent predictions relies on the accuracy of the previous predictions. In most of the time, the prediction deviation would accumulate and the quality of prediction frames would decrease dramatically. In our method, we introduce a retrospection process (e.g., green arrows in Figure 5.1) to further alleviate prediction deviation. In the retrospection process, the predicted frames are required to reconstruct the input frame in backward. The retrospection model F is constructed to estimate the reconstruction frames $\tilde{\mathbf{x}}^F_T = F(\hat{\mathbf{x}}^G_{T+K:T+1})$ given the observed predicted frames $\{\hat{\mathbf{x}}_{T+K}, \cdots, \hat{\mathbf{x}}_{T+1}\}$ in a reverse chronology order. The preceding retrospection frames are generated recursively:

$$\tilde{\mathbf{x}}^F_t = \begin{cases} F\left(\hat{\mathbf{x}}^G_{T+K:T+1}\right), & t = T \\ F\left(\left[\hat{\mathbf{x}}^G_{T+K:T+1}, \tilde{\mathbf{x}}^F_{T:t+1}\right]\right), & 1 \leqslant t < T, \end{cases} \tag{5.2}$$

where the $\left[\hat{\mathbf{x}}^G_{T+K:T+1}, \tilde{\mathbf{x}}^F_{T:t+1}\right]$ indicates the concatenation of the predicted frames $\hat{\mathbf{x}}^G_{T+K:T+1}$ and retrospections $\tilde{\mathbf{x}}^F_{T:t+1}$ in the channel of time.

The rest of this section organized as follow. We first briefly review the previous methods for predicting future frames. Then we introduce the proposed retrospection process in detail. Lastly, we summarize the full objective function of our model.

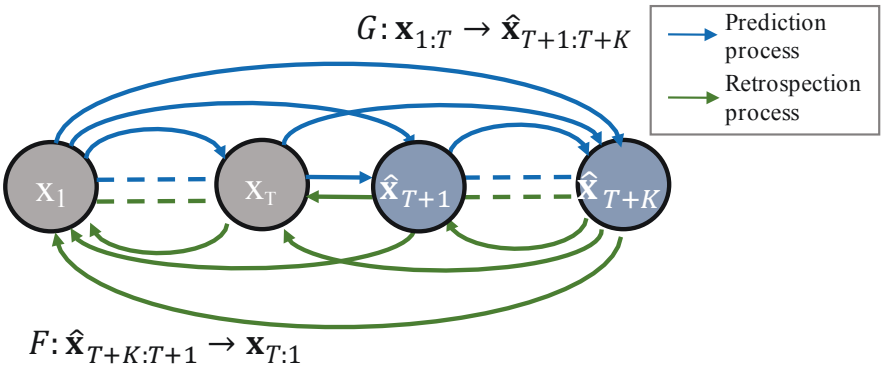

Figure 5.1: The illustration of our model. The generative model predicts the next frame conditioned on the previous frames (blue lines), while our model introduces a retrospection process to reconstruct the original input frames given predictions in a reverse chronological order (green lines).

5.3.1 Preliminaries: Prediction Process

We first briefly review the previous video prediction model which only contains the prediction process. Normally, the video prediction model G is trained to predict K subsequent frames given T observed frames as inputs. Let the training data set denote as $D = \{\mathbf{x}^{(i)}_{1,\cdots,T+K}\}_{i=1}^{N}$ where i indicates the index of videos. A prediction loss $\mathcal{L}_{img}$ is usually adopted to minimize the distance between predicted frames and ground-truth. The function can be expressed as follow:

$$\mathcal{L}_{img}(G) = \mathcal{L}_p(\mathbf{x}_{T+k}, \hat{\mathbf{x}}_{T+k}) + \mathcal{L}_{gdl}(\mathbf{x}_{T+k}, \hat{\mathbf{x}}_{T+k}), \tag{5.3}$$

where $\mathcal{L}_P$ guides the network to match the average pixel values, $\mathcal{L}_{gdl}$ guides our network to match the gradients of pixel values between ground-truth and predicted frames:

$$\mathcal{L}_p(\mathbf{y}, \mathbf{z}) = \sum_{k=1}^{K} ||\mathbf{y} - \mathbf{z}||_p^p, \tag{5.4}$$

$$\mathcal{L}_{gdl}(\mathbf{y}, \mathbf{z}) = \sum_{i,j}^{h,w} |\left(|\mathbf{y}_{i,j} - \mathbf{y}_{i-1,j}| - |\mathbf{z}_{i,j} - \mathbf{z}_{i-1,j}|\right)|^{\lambda} + |\left(|\mathbf{y}_{i,j-1} - \mathbf{y}_{i,j}| - |\mathbf{z}_{i,j-1} - \mathbf{z}_{i,j}|\right)|^{\lambda}.$$

Here $\mathbf{x}_{T+k}$ and $\hat{\mathbf{x}}_{T+k}$ are ground-truth and predicted frames at time $T+k$ respectively. Since training to generate average sequences would result in blurry generations [90], the additional adversarial loss $\mathcal{L}_{GAN}$ is used to produce realistic visual outputs [144]:

$$\mathcal{L}_{GAN}(G) = -\log D_g\left(\left[\mathbf{x}_{1:T}, \hat{\mathbf{x}}^G_{T+1:T+K}\right]\right), \tag{5.5}$$

where $\mathbf{x}_{1:T}$ is the concatenation of the input images, $\hat{\mathbf{x}}_{T+1:T+K^G}$ is the concatenation of all predicted images along the time dimension, and $D_g(\cdot)$ is the discriminator in adversarial training. The discriminative loss for discriminator D_g in adversarial training is defined by minimizing:

$$\mathcal{L}_{gan}(D_g) = -\log D_g([\mathbf{x}_{1:t}, \mathbf{x}_{t+1:t+k}]) - \log\left(1 - D_g\left(\left[\mathbf{x}_{1:t}, \hat{\mathbf{x}}^G_{T+1:T+K}\right]\right)\right), \tag{5.6}$$

where $\mathbf{x}_{T+1:T+K}$ is the concatenation of the ground-truth future images. $\mathcal{L}_{GAN}$ and $\mathcal{L}_{img}$ allow the model to criticize the quality of the prediction sequence, which focus on comparing the predicted frames with the ground-truth.

5.3.2 Retrospection Process

As shown in left-bottom of Fig 5.2, our model consists of two routes. One route $G \rightarrow F$ is to train prediction network, and the auxiliary retrospection network adopts to reconstruct the prediction to the past. The other route $F \rightarrow G$, on the other hand, is built to train retrospection network. In route $F \rightarrow G$, the retrospection network first generates a few past frames in backward, then pauses to check if the generated frames could be used to predict the future accurately by prediction network.

5.3.2.1 Route $G \rightarrow F$

In the route $G \rightarrow F$, our proposed model consists of two generative networks: a forward prediction network G and a retrospection network F. The forward network G is to perform predictions, which is similar to the previous work [144]; the retrospection network F is introduced to retrospect the past by observing current predictions in Equation 5.2. The retrospection loss $\mathcal{L}_{ret1}$ is introduced

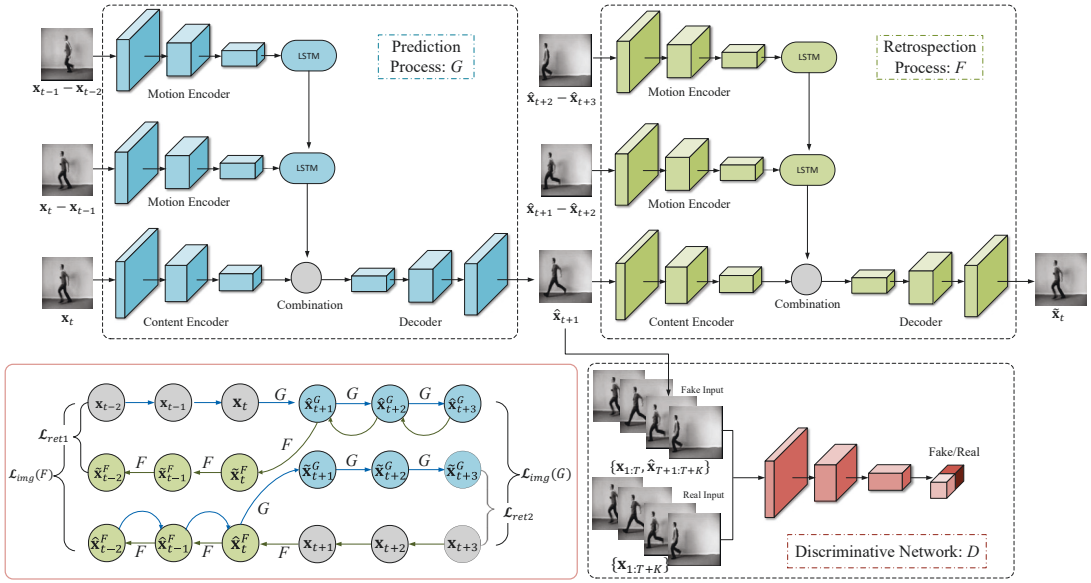

Figure 5.2: The overall architecture of the proposed network. Left bottom: our model contains two routes: one is $G \rightarrow F$, and the other is $F \rightarrow G$. Right top: illustration of the route $G \rightarrow F$ in detail. In each route, our model consists of two processes. The prediction process executes recursively by taking the observations to generate subsequent frames, while the retrospection process synthesizes frames by observing the predicted frames in a backward manner. The discriminative network uses an adversarial loss to distinguish between predicted and real frames.

to minimize the distance between the retrospections $\tilde{\mathbf{x}}^F_{1:T}$ and the original frames $\mathbf{x}_{1:T}$:

$$\mathcal{L}_{ret1}(F, G) = \mathcal{L}_p\left(\mathbf{x}_t, \tilde{\mathbf{x}}^F_t\right) + \mathcal{L}_{gdl}\left(\mathbf{x}_t, \tilde{\mathbf{x}}^F_t\right), t \in \{1:T\}. \tag{5.7}$$

Notice that only one route of $G \rightarrow F$ cannot guarantee to alleviate the deviation of predicted frames. Due to the capacity of a network, the retrospection network F might be trained to reconstruct the original frames from noisy frames. As a result, another route $F \rightarrow G$ is necessary to train the retrospection network by observing the ground-truth frames.

5.3.2.2 Route $F \rightarrow G$

In order to train the retrospection network F, the inverse route $F \rightarrow G$ is built. In route $F \rightarrow G$, we feed the inverse of frame sequence $\{\mathbf{x}_{T+K}, \cdots, \mathbf{x}_T\}$ into the

network, and encourage the network to generate preceding frames recursively:

$$\hat{\mathbf{x}}_t^F = \begin{cases} F\left(\mathbf{x}_{T+K:T+1}\right), & t = T \\ F\left(\left[\mathbf{x}_{T+K:T+1}, \hat{\mathbf{x}}_{T:t+1}^F\right]\right), & 1 \leqslant t < T, \end{cases} \tag{5.8}$$

where $\hat{\mathbf{x}}_{T:t+1}^F$ indicates the retrospection frames of network F. A reconstruction loss is defined similar to Equation 5.3 to minimize the distance between retrospections $\hat{\mathbf{x}}_t^F$ and ground-truth frames $\mathbf{x}_t$:

$$\mathcal{L}_{img}(F) = \mathcal{L}_p\left(\mathbf{x}_t, \hat{\mathbf{x}}_t^F\right) + \mathcal{L}_{gdl}\left(\mathbf{x}_t, \hat{\mathbf{x}}_t^F\right), \ t \in \{1:T\}. \tag{5.9}$$

Similar to route $G \rightarrow F$, the retropsections $\hat{\mathbf{x}}_{1:T}^F$ are required to capable of predicting the frames $\tilde{\mathbf{x}}_{T+1:T+K}^G$ by prediction network recursively:

$$\tilde{\mathbf{x}}_{T+k}^G = \begin{cases} G(\hat{\mathbf{x}}_{1:T}^F), & k = 1 \\ G(\left[\hat{\mathbf{x}}_{1:T}^F, \tilde{\mathbf{x}}_{T+1:T+k-1}^G\right]), & 1 < k \leqslant K. \end{cases} \tag{5.10}$$

The retrospection loss $\mathcal{L}_{ret2}$ is introduced to rectify the error between the predicted frames and original input frames:

$$\mathcal{L}_{ret2}\left(F, G\right) = \mathcal{L}_p\left(\mathbf{x}_{T+k}, \tilde{\mathbf{x}}_{T+k}^G\right) + \mathcal{L}_{gdl}\left(\mathbf{x}_{T+k}, \tilde{\mathbf{x}}_{T+k}^G\right), \quad k \in \{1:K\}, \tag{5.11}$$

where $\tilde{\mathbf{x}}_{T+k}^G$ indicates the reconstructed frames of network G.

In our model, the retrospection loss $\mathcal{L}_{ret1}$ in Equation 5.7 is used to alleviate the accumulation error produced from forward video prediction model G, meanwhile $\mathcal{L}_{ret2}$ in Equation 5.11 is used to alleviate the accumulation error produced from retrospection model F. We analyzed the function of the two-route model in Section 5.5.3. In order to train the retrospection network, we introduce the adversarial loss to cheat the discriminator D_f to classify the retrospection samples as real samples:

$$\mathcal{L}_{GAN}(F) = -logD_f(\left[\mathbf{x}_{T+K:T+1}, \hat{\mathbf{x}}_{T:1}^F\right]). \tag{5.12}$$

The discriminator aims to distinguish between the real frames and the retro-

spected frames in a reverse chronological order. The adversarial loss of discriminative network is defined by minimizing the function:

$$\mathcal{L}_{\text{GAN}}\left(D_f\right) = -\log D_f\left(\left[\mathbf{x}_{T+K:T+1}, \mathbf{x}_{T:1}\right]\right) - \log\left(1 - D_f\left(\left[\mathbf{x}_{T+K:T+1}, \hat{\mathbf{x}}^F_{T:1}\right]\right)\right), \tag{5.13}$$

where $\mathbf{x}_{T+K:T+1}$ indicates reversed ground-truth video sequence, while $\hat{\mathbf{x}}^F_{T+K:T+1}$ denotes reversed generated frames of retrospection network F. Here we adopt a conditional GAN [102] that conditioned on the reversed ground-truth frames $\mathbf{x}_{T+K:T+1}$.

5.3.3 Full Objective

Our full objective for generative models of two process can be summarized as:

$$\begin{aligned}\mathcal{L}(G, F) = &\alpha_1\mathcal{L}_{img}(G) + \alpha_2\mathcal{L}_{img}(F) \\ &+ \beta_1\mathcal{L}_{GAN}(G) + \beta_2\mathcal{L}_{GAN}(F) \\ &+ \gamma_1\mathcal{L}_{ret1}(F, G) + \gamma_2\mathcal{L}_{ret2}(F, G),\end{aligned} \tag{5.14}$$

where α_1, α_2, β_1, β_2, γ_1 and γ_2 control the relative importance. In our model, the predictions of network G are required not only to minimize the discrepancy with the ground-truth frames, but also to be able to retrospect to the original frames. As a result, the objective of the prediction network G is to solve:

$$arg\min_G \left[\mathcal{L}_{img}(G) + \beta_1\mathcal{L}_{GAN}(G) + \gamma_1\mathcal{L}_{ret1}(F, G)\right]. \tag{5.15}$$

The generative network with the image loss and adversarial loss would lead to the preliminary generation results. The introduced retrospection loss provides an auxiliary constraint to further enhance the quality of predicted frames. In addition, given a certain capacity of retrospective network, compared with a noisy input, high-quality predicted frames for the retrospection module tends to have a big chance to better reconstruct the past frames. That is to say, by minimizing the retrospection loss, the predicted frames would be further optimized to achieve higher quality. Meanwhile, the retrospection network is updated by observing the ground-truth frames in the route $F \rightarrow G$. The objective of retrospection network

F is to solve:

$$arg\min_{F}\left[\mathcal{L}_{img}(F)+\beta_2\mathcal{L}_{GAN}(F)+\gamma_2\mathcal{L}_{ret2}(F,G)\right]. \tag{5.16}$$

While the discriminative network is trained to minimize the loss function Equation 5.6 and Equation 5.13 in order to distinguish between the generated frames and real frames:

$$D_g^* = arg\min_{D_g}\mathcal{L}_{\text{GAN}}(D_g), \tag{5.17}$$

$$D_f^* = arg\min_{D_f}\mathcal{L}_{\text{GAN}}(D_f). \tag{5.18}$$

The detailed training procedure is shown in Algorithm 3.

5.4 Implementation

5.4.1 Network Configuration

We design the architecture for our prediction network G and retrospection networks F according to [144] which has shown impressive results for video prediction. The prediction network and retrospection network share the same architecture. While the prediction network aims to predict future frames, the retrospection network aims to generate fore-passed frames. As shown in Fig. 5.2, each network contains a motion encoder, a content encoder, an LSTM module, a combination layer and a decoder. The motion encoder and content encoder contain a series of convolutional layers and max pooling layers. The decoder consists of a series of deconvolution layers and unpooling layers. The unpooling layer [171] is an up-sampling operation which can be viewed as an inverse of max pooling. We adopt *tanh* function in the output layer of decoder, and the ReLU function in the rest of layers.

In addition, a discriminative network is built to facilitate more realistic outputs. The forward and backward discriminative network D_g and D_f are shared the same architecture, which consist of a series of convolutional layers. The Batch Normalization [57] and leaky-Relu [95] are used after convolutional layers. At the

Table 5.2: Architecture of the prediction and retrospection network

Module	Operation	Kernel size	Stride	Padding	Feature Maps
Motion encoder	Convolution	5	1	2	64
	MaxPooling	2	2	-	64
	Convolution	5	1	2	128
	MaxPooling	2	2	-	128
	Convolution	7	1	3	256
	MaxPooling	2	2	-	256
Content Encoder	Convolution	3	1	1	64
	Convolution	3	1	1	64
	MaxPooling	2	2	-	64
	Convolution	3	1	1	128
	Convolution	3	1	1	128
	MaxPooling	2	2	-	128
	Convolution	3	1	1	256
	Convolution	3	1	1	256
	Convolution	3	1	1	256
	MaxPooling	2	2	-	256
Decoder	Unpooling	2	2	-	256
	Deconvolution	3	1	1	256
	Deconvolution	3	1	1	256
	Deconvolution	3	1	1	128
	Unpooling	2	2	-	128
	Deconvolution	3	1	1	128
	Deconvolution	3	1	1	128
	Deconvolution	3	1	1	64
	Unpooling	2	2	-	64
	Deconvolution	3	1	1	64
	Deconvolution	3	1	1	1

last layer, the sigmoid function is used. Details are listed in Table 5.3.

Motion Encoder. The motion encoder of the prediction network captures the temporal dynamics of the scenes' components by recurrently observing the difference between frames $\mathbf{x}_t$ and $\mathbf{x}_{t-1}$. Correspondingly, in the retrospection network, the difference between the $\mathbf{x}_{t+1}$ and $\mathbf{x}_{t+2}$ is fed into its motion encoder. The outputs of motion feature are filtered by a series of convolutional layers. The architecture is similar to VGG16 [131] up to the third pooling layer, except that

Table 5.3: Algorithm for training the temporal generative network.

	Operation	Kernel size	Stride	Padding	Feature maps
Discriminative Network	Convolution	5	2	3	64
	Convolution	5	2	3	128
	Convolution	5	2	3	256
	Convolution	5	2	3	512
	Convolution	5	1	2	1
LeakyReLU: Slope 0.2					

the consecutive 3×3 convolutions are replaced by single 5×5, 5×5, and 7×7 convolutions in each layer.

Content Encoder. The content encoder extracts important spatial features from a single frame, such as the spatial layout of the scene and salient objects in a video. In our experiment, it takes the last observed frames $\mathbf{x}_t$ as an input in the prediction process, while it takes the first predicted frames $\hat{\mathbf{x}}_{t+1}$ as input in the retrospection process. The content encoder is also built with the same architecture as VGG16 [131] up to the third pooling layer.

Decoder Module. The decoder is used to generate a pixel-level prediction of the next frames in the prediction process or the past frames in retrospection process. To reduce the information loss caused by pooling at the encoding phase, a residual connection [155] is adopted by taking the computed features from both the motion and content encoders. We employ deconvolution network [172] for the decoder module. The architecture of the decoder is the mirror of the content encoder. The output layer is passed through a $\tanh(\cdot)$ activation function.

Discriminative Network. The discriminative network consists of 4 convolutional layers with kernel size 5×5 and stride 2, whose output channels are 64, 128, 256, 512 respectively. From the second layer, each layer is followed by a batch normalization layer [57]. After that, each layer is followed by a leaky ReLU layer [50]. The final layer is a fully connected layer with 1 hidden unit followed by a sigmoid function, aims to predict the possibility of input's label. Note that we concatenate the frames of both observation and prediction along the channel of time as the inputs.

The LSTM module predicts or retrospects the dynamics of frames in a chrono-

logical or reversed order recursively. We adopt a convolutional LSTM [128] as our LSTM module. The combination module is used to fuse the content representation and motion representation, which consists of 3 consecutive 3×3 convolutions (256, 128, and 256 channels in each layer).

5.4.2 Training Strategy

All networks are trained by the Adam optimization [67] for 100,000 iterations with the learning rate of 0.0001, the exponential decay rate for the first moment of 0.5 and the exponential decay rate for the second moment estimates of 0.999. Following MCNET [144], we set a margin to balance the adversarial training of generative network and discriminative model. To stabilize training, we set a margin to balance the adversarial training of generative network and discriminative model. Following MCNET [144], the loss margin m is set to be 0.3.

Algorithm 3 shows the detailed training procedure. In the training procedure, the generative network G and retrospective network F are trained alternatively.The route $G \rightarrow F$ is used to optimize the generative network G, and route $F \rightarrow G$ is used to optimize the retrospective network.r During training each network, we assume that the other network has been optimized. For instance, the retrospective network F is well optimized to generated realistic and sharp frames by the $\mathcal{L}_{GAN}(F)$. As a result, as long as the retrospection is pixel-wise consistent to the input frames, the error of predicted frames could be rectified. Similar idea has been applied in CycleGAN [183]. In CycleGAN [183], they trained the two generative networks by two directions, and in each direction a cycle-consistent loss is adopted.

5.5 Experiments

In this section, we present experiments using our model for long-term video prediction. We first evaluate our model on the KTH [135] and Weizmann datasets [45]. We then proceed to evaluate on a more challenging dataset, UCF-101 [132]. We compare our model with MCNET [144], which achieves state-of-the-art performance on the KTH [135], Weizmann [45] and UCF101 datasets [132]. For all

Algorithm 3 Adversarial training of our proposed model.

Require: the set of training data $\{\mathbf{x}^{(i)}_{1,\cdots,T+K}\}_{i=1}^{N}$, margin m.

for number of training iterations **do**
  Sample minibatch of sequence frames $\mathbf{x}_{1:T+K}$.
  Predict the subsequent frames $\hat{\mathbf{x}}_{T+1:T+K}$ by given the observed frames $\mathbf{x}_{1:T}$ in the prediction network $G : \mathbf{x}_{1:T} \rightarrow \hat{\mathbf{x}}^{G}_{T+1:T+K}$ in Equation 5.1.
  Retrospect the preceding frames $\mathbf{x}_{1:T}$ by given the predicted frames $\hat{\mathbf{x}}^{G}_{T+1:T+K}$ in the retrospection network $F : \hat{\mathbf{x}}^{G}_{T+1:T+K} \rightarrow \tilde{\mathbf{x}}^{F}_{1:T}$ in Equation 5.2.
  Generate the preceding frames $\tilde{\mathbf{x}}^{F}_{1:T}$ by given the ground-truth frames $\mathbf{x}_{T+K:T+1}$ in the retrospection network $F : \mathbf{x}_{T+K:T+1} \rightarrow \hat{\mathbf{x}}^{F}_{1:T}$ in Equation 5.8.
  Generate the subsequent frames $\tilde{\mathbf{x}}^{G}_{T+1:T+K}$ by given the generated frames $\hat{\mathbf{x}}^{F}_{1:T}$ in the network $G : \hat{\mathbf{x}}^{F}_{1:T} \rightarrow \tilde{\mathbf{x}}^{G}_{T+1:T+K}$ in Equation 5.10.
  if $\log(1 - D_g(\left[\mathbf{x}_{1:T}, \hat{\mathbf{x}}^{G}_{T+1:T+K}\right])) > m$ and $\log(D_g(\mathbf{x}_{1:T+K})) > m$ **then**:
    Update discriminator D_g using Equation 5.17:
$$\Delta_{\theta_{D_g}} \leftarrow \nabla_{\theta_{D_g}} \mathcal{L}_{\text{GAN}}(D_g).$$
  end if
  if $\log(1 - D_g(\left[\mathbf{x}_{1:T}, \hat{\mathbf{x}}^{G}_{T+1:T+K}\right])) < 1 - m$ and $\log(D_g(\mathbf{x}_{1:T+K})) < 1 - m$ **then**
    Update prediction network G using Equation 5.15:
$$\Delta_{\theta_G} \leftarrow \nabla_{\theta_G}(\alpha_1 \mathcal{L}_{img} + \beta_1 \mathcal{L}_{GAN} + \gamma_1 \mathcal{L}_{ret1}).$$
  end if
  if $\log(1 - D_f(\left[\mathbf{x}_{T+K:T+1}, \hat{\mathbf{x}}^{F}_{T:1},\right])) > m$ or $\log(D_f(\mathbf{x}_{T+K:1})) > m$ **then**
    Update discriminator D_f using Equation 5.18:
$$\Delta_{\theta_{D_f}} \leftarrow \nabla_{\theta_{D_f}} \mathcal{L}_{\text{GAN}}(D_f).$$
  end if
  if $\log(1 - D_f(\left[\mathbf{x}_{T+K:T+1}, \hat{\mathbf{x}}^{F}_{T:1}\right]))) < 1 - m$ or $\log(D_f(\mathbf{x}_{T+K:1})) < 1 - m$ **then**
    Update retrospection network F using Equation 5.16:
$$\Delta_{\theta_F} \leftarrow \nabla_{\theta_F}(\alpha_2 \mathcal{L}_{img} + \beta_2 \mathcal{L}_{GAN} + \gamma_2 \mathcal{L}_{ret2}).$$
  end if
end for

our experiments, we use $\lambda = 1$, and $p = 2$ in the loss function. We train our network by observing 10 frames and predicting 10 subsequent frames. For fairness, both the prediction process and retrospection process of our model adopted the same architecture with MCNET [144]. As the retrospection process would be discarded in the test phase, it costs the same test computation complexity. We demonstrate the quantitative comparison in terms of PSNR, SSIM. PSNR and SSIM are commonly used in previous works [144]. However, they are shallow functions, and not necessarily coincide with human perception. Recently, perceptual metrics are proposed by adopting the deep features in neural networks, such as LPIPS (Learned Perceptual Image Patch Similarity) [177] and FVD [18]. To partially mitigate the limitations of these metrics, we also evaluate on LPIPS, which have been shown to correspond better to human perceptual judgments. In addition, we used a person detection evaluation to test the quality of generation frames on person action dataset, with the idea that acceptable predictions should contain recognizable person. Meanwhile, we take two variants of our method for ablation study, details are presented in Section 5.5.3.

5.5.1 KTH and Weizmann Action Datasets

5.5.1.1 Experimental setting

The KTH human action dataset [135] contains 6 categories of periodic motions on a simple background: running, jogging, walking, boxing, hand-clapping and hand-waving. Following MCNET [144], we used person 1-16 for training and 17-25 for testing, and also resize frames to 128×128 pixels. During the evaluation, to demonstrate the effectiveness of our proposed long-term model, we predicted 100 subsequent frames given the previous 10 frames as input. We also selected the walking, running, one-hand waving, and two-hands waving sequences from the Weizmann action dataset [45] to verify the networks' generalization. Since most of the videos in the Weizmann dataset only contain around 70-80 frames, we test our model for the predictions of 70 time steps on the Weizmann dataset.

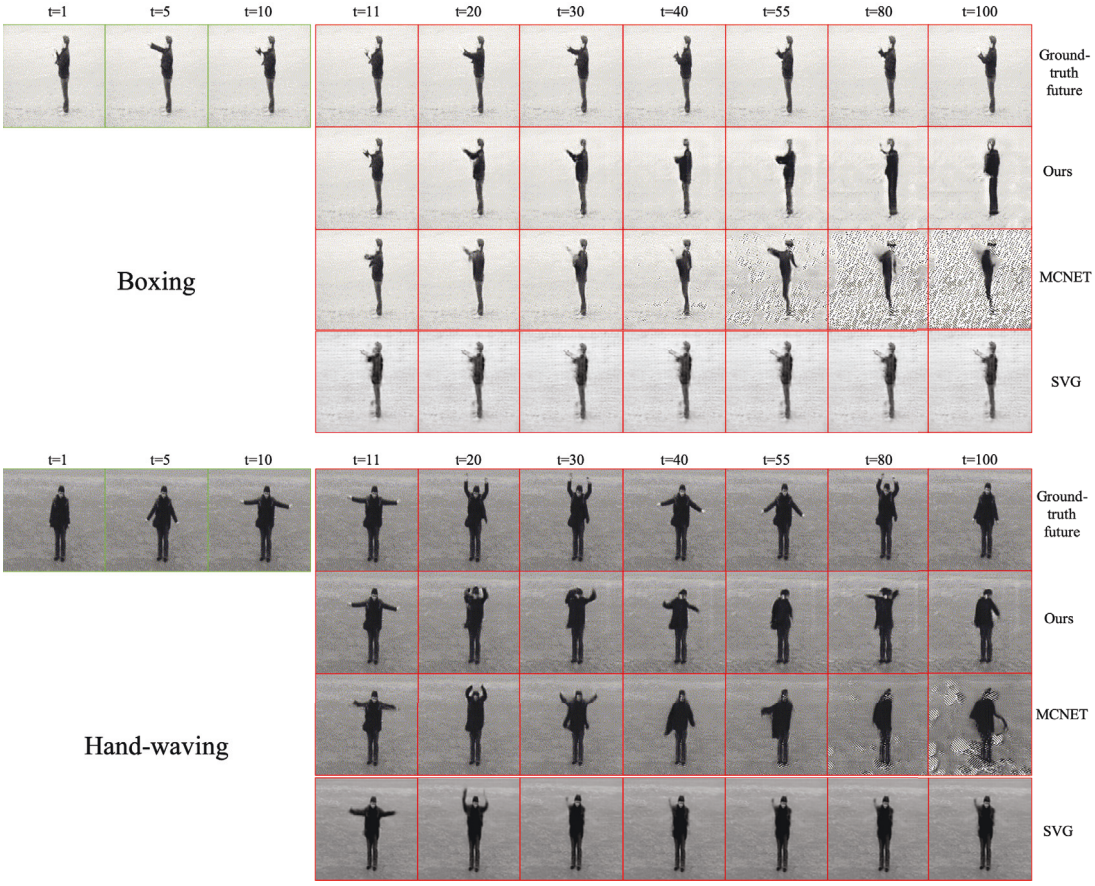

Figure 5.3: Qualitative comparison between our methods, MCNET and SVG on the KTH dataset. The top case corresponds to the action of boxing, and the lower case corresponds to the action of hand-waving.

5.5.1.2 Results Analysis

Figure 5.3 presents qualitative results of long-term prediction by our network and MCNET on the KTH dataset. As expected, prediction results by our full objective preserve a better quality with time passing. Our method could achieve more satisfying results at 100-time steps, while MCNET's results tend to collapse to meaningless noise (see the fourth column in each case at $t = 100$). In action of boxing, we notice that at $t = 40$, the output of MCNET appears some noisy pattern besides the person's feet. With the increase of time step, the region of noise becomes larger and eventually dominates the whole images. In testing of hand-waving action, we find that MCNET is able to achieve a competitive result

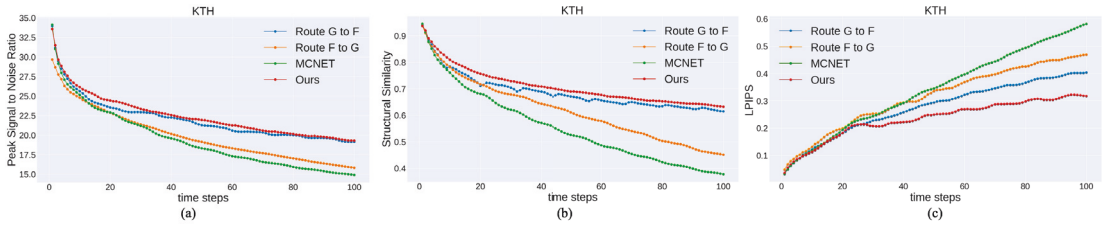

Figure 5.4: Quantitative comparisons between different variants of our method and MCNET baseline in terms of PSNR, SSIM and LPIPS on the KTH dataset. "Ours" denotes our method (MCNET+Retrospection) with full objective. "Route F to G" represents Route $F \rightarrow G$ alone (Equation 5.19). "Route G to F" indicates Route $G \rightarrow F$ alone (Equation 5.20). Given 10 input frames, the models predict 100 frames recursively. For PSNR and SSIM, higher is better. For LPIPS, lower is better.

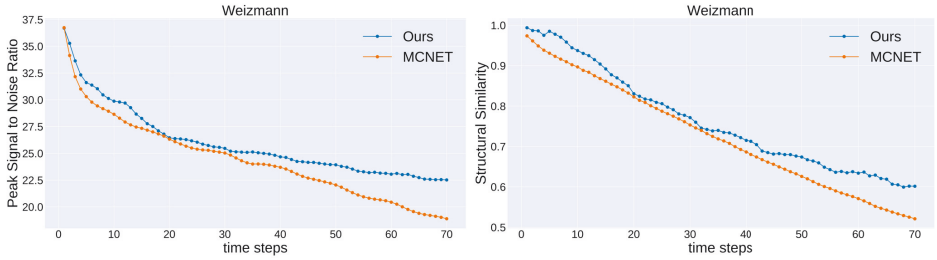

Figure 5.5: Quantitative comparisons between our method and MCNET in terms of PSNR, SSIM on the Weizmann dataset.

in the first several time steps. However, with the time passing, the prediction deviation is accumulated whose images appear noisier. Our method with retrospection loss, on the other hand, is able to output consistent high-quality results for a long time. The reason is that the proposed retrospection process is able to alleviate the prediction error accumulation.

Figure 5.4 summarizes the quantitative comparisons of our methods, MCNET and two variants of our model. In the KTH test set (Figure 5.4), our models and variants outperform the MCNET baseline in long-term video prediction. Although all four methods achieve comparable LPIPS scores for the first 20 future frames, with the increase of time step, the margin between our method (blue line) and MCNET (red line) becomes more significant. One reason for this result is that the baseline method only considers the prediction recursively in the forward time steps, which is easy to accumulate the prediction error and leads to dramatic

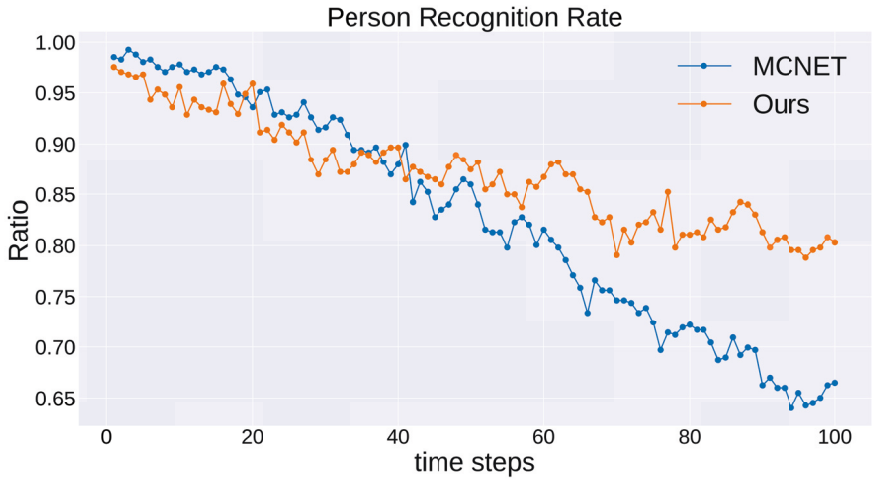

Figure 5.6: Comparison with MCNET [144] in terms of the recognition rate. The recognition rate of the person detector that a person is recognized in the predicted frame.

degradation of prediction performance. Also, we test on unseen data of Weizmann dataset [45] by the pretrained model. Performance shown in Figure 5.5 reveals that our method outperforms MCNET especially on the long-term horizon.

5.5.1.3 Person Detection Evaluation

we use a person detection evaluation to test the quality of generation frames on person action dataset, with the idea that acceptable predictions should contain recognizable person. Similar to [155], we use the pretrained Mask-RCNN[1] as the person detector, and calculated the percentage of frames that a recognizable person is in the image. We record the person recognition rate of Mask-RCNN in Figure 5.6. The proposed method stays relatively constant over prediction time-steps. For longer-term predictions, e.g., at $t = 100$, more than 80% of our generated frames could be recognized as a person, while the performance of MCNET drops to only 65% of predictions can be recognized. The evaluation shows that the proposed method is better than the baselines on the long-term horizon.

[1] `https://github.com/matterport/Mask_RCNN`

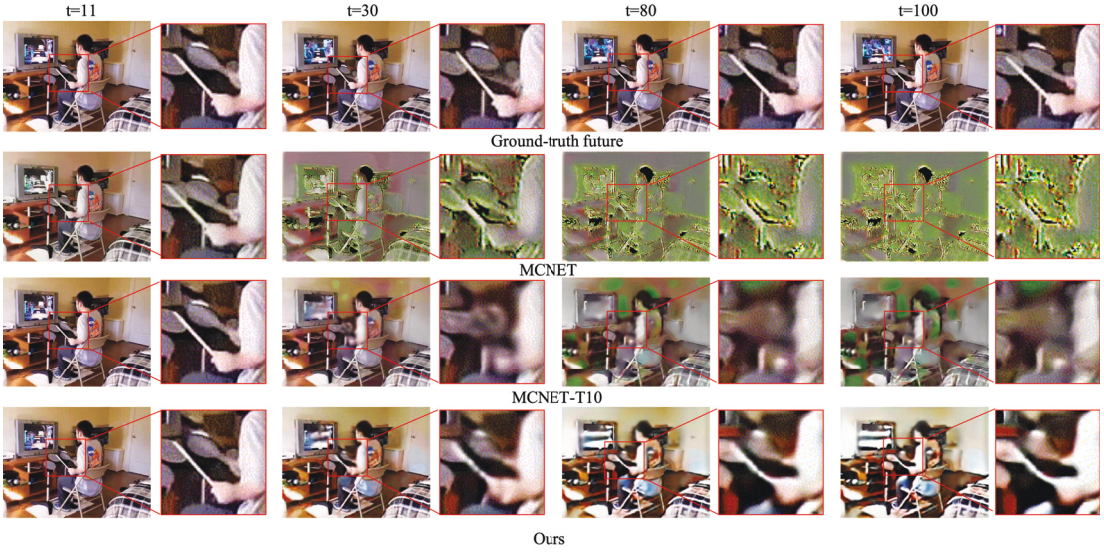

Figure 5.7: Quantitative comparison on the UCF-101 dataset. MCNET [144] trained to predict 10 frames is denoted as "MCNET-T10". The results are predicted by observing 10 previous frames. Our method less artifact and blur around the ambiguity region. The remarkable region is denoted in color and scaled.

5.5.2 UCF-101 Dataset

This section presents results on more challenging real-world videos dataset, the UCF-101 dataset [132]. Having collected from YouTube, the dataset contains 101 realistic human actions taken in a wild and exhibits various challenges, such as background clutter, occlusion, and complicated motion. The videos in 101 action categories are grouped into 25 groups, where each group consists of 4-7 videos of an action. We use group 1-7 for testing and group 8-25 for training. We employ the same network architecture as in the KTH dataset but resized frames to 240 $\times$ 320 pixels.

5.5.2.1 Results Analysis

Figure 5.8 show the quantitative comparisons between our network trained by full objective and MCNET. We test the official-released pre-trained model [1](denoted as "MCNET"), which is trained to predict one future frame by observing four

[1]`https://github.com/rubenvillegas/iclr2017mcnet`

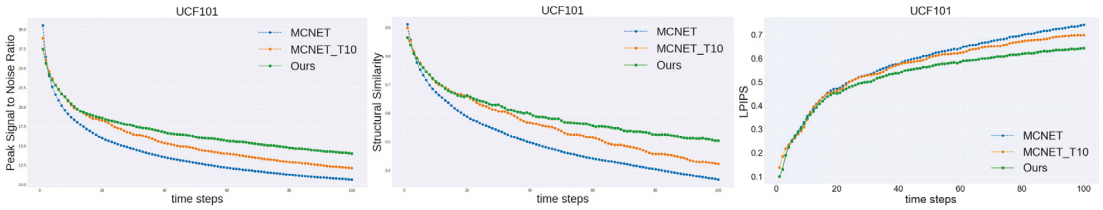

Figure 5.8: Quantitative comparisons between our model, MCNET [144] and MCNET trained by 10 input frame and 10 output frames (indicated by "MCNET-T10"). In the test phase, the models predict 100 frames recursively given 10 input frames.

frames. However, we find that only predicting one future frame is easily to miss the motion pattern of the video and leads to stationary output. As a result, the performance in terms of PSRN, SSIM and LPIPS drops with the increase of time (see blue line).

For fairness, we also compare with the MCNET [144] which is trained to predict 10 frames by given 10 input frames (denoted as "MCNET-T10" in orange line). We observe that our model still outperforms the compared method and the gap becomes more significant with the increase of predicted time. Figure 5.7 presents qualitative comparisons between frames generated by our method, MCNET and MCNET-10T. We observe that the results of MCNET is becoming blur while ours are still recognizable (see details in the zoomed region). In addition, the quality of MCNET's results falls dramatically with time passing, e.g., at $t = 100$ in the second row. Results of MCNET-T10 is better compared with MCNET, while we still could find some green noisy patches. Our method, on the other hand, could maintain structural outputs with reasonable dynamic.

5.5.2.2 Human Perceptual Study

We further evaluat our algorithm via a human study. We perform pairwise A/B tests deployed on a service similar to Mechanical Turk. We follow the experiment procedure in [145]. The participants are asked to select the more realistic video generated from our method and MCNET. Each pair contains two video predictions observed by the same input frames generated. We use the data sets of KTH, Weizmann and UCF101 [132]. In each data set, we collect comparisons of 100

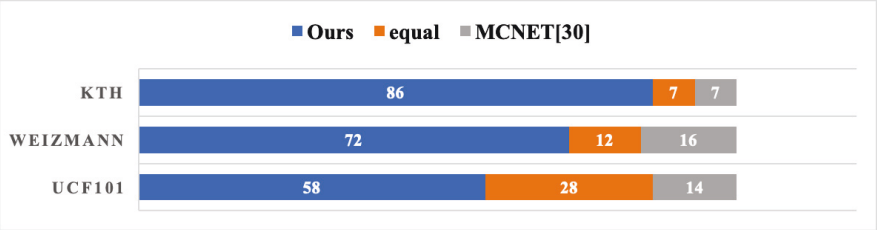

Figure 5.9: The stacked bar chart of participants preferences for our methods compared to MCNET [144]. The blue bar indicates the number of videos that more participants prefer our results. The gray bar indicates the number of videos that more participants prefer MCNET's results. The orange bar indicates the number of videos where two methods get a equal number of votes.

Table 5.4: Ablation study of the proposed model.

Model	PSNR	SSIM	LPIPS
MCNET (Only G)	19.42	0.5541	0.3410
Route $F \rightarrow G$	19.92	0.6186	0.3136
Route $G \rightarrow F$	22.05	0.6881	0.3749
Two routes (Ours)	**22.48**	**0.7092**	**0.2336**

generated videos, and presented each video to 10 human raters. The table in Figure 5.9 shows the video number of the most users preference. It demonstrates that most users prefer our results, which indicates that qualitative performance obtained by our proposed approaches are better than those obtained by existing methods.

5.5.3 Model Analysis

5.5.3.1 Ablation Study

We take two variants of our method for ablation study. The first variant is denoted as "Route $F \rightarrow G$". We first train a retrospection network and prediction network by solving:

$$\arg\min_{G,F} \left[\alpha_2 \mathcal{L}_{img}(F) + \beta_2 \mathcal{L}_{GAN}(F) + \gamma_2 \mathcal{L}_{ret2}(F, G)\right]. \tag{5.19}$$

The prediction is obtained by feeding the test frames to the trained prediction network.

The second variant is denoted as "route $G \rightarrow F$". We simultaneously train the prediction and retrospection network by removing the auxiliary route $F \rightarrow G$:

$$arg \min_{G,F} \left[\alpha_1 \mathcal{L}_{img}(G) + \beta_1 \mathcal{L}_{GAN}(G) + \gamma_1 \mathcal{L}_{ret1}(F, G)\right], \tag{5.20}$$

where α_1, β_2 and γ_1 are set to be the same as the full objective function. Table 5.4 presents results of ablation study on the KTH dataset. We observe that our model and its variants outperform the MCNET baseline which only has the prediction process. In "route $G \rightarrow F$", the generative network was optimized by the prediction loss $\mathcal{L}_{img}(G)$, pixel adversarial loss $\mathcal{L}_{GAN}(G)$, and rectified by the retrospection loss $\mathcal{L}_{ret1}(F, G)$. The retrospection loss enforces the predictions to reconstruct the original input. As a result, the performance is close to the two routes version. In our full model, an auxiliary route "route $F \rightarrow G$", is designed to reduce the accumulation error produced from F. In Figure 5.4, we observe that our method still outperforms the "route $G \rightarrow F$", which indicates a better retrospective network F is able to improve the accuracy of prediction. In "route $F \rightarrow G$", the generative network G only optimize the reconstruction loss: $\arg\min_G \mathcal{L}_{ret2}(F, G)$, whose aim is actually to generate the future frames $\left\{\tilde{x}^G_{T+1}, \cdots, \tilde{x}^G_{T+K}\right\}$ from the previous frames $\left\{\hat{x}^F_1, \cdots, \hat{x}^F_T\right\}$. That is to say, the generative network G in this route does not get the benefit from the retrospective process. As the results shown in Figure 5.4, the performance of "route $F \rightarrow G$" is close to the baseline (MCNET). Overall, the proposed training strategy of two routes achieves the best result.

5.5.3.2 Parameter Selection

We discuss our parameter selection strategy and analyze the function for the newly introduced parameters. In our model, Route $G \rightarrow F$ and Route $F \rightarrow G$ are symmetrical, so we set $\alpha_1 = \alpha_2$, $\beta_1 = \beta_2$, and $\gamma_1 = \gamma_2$. This selection strategy is similar to CycleGAN [183], which uses the same parameters in the same functionality loss (e.g., cycle-consistent loss) of the two translation directions. In our model, α_1 (α_2) and β_1 (β_2) balance the importance of image reconstruction

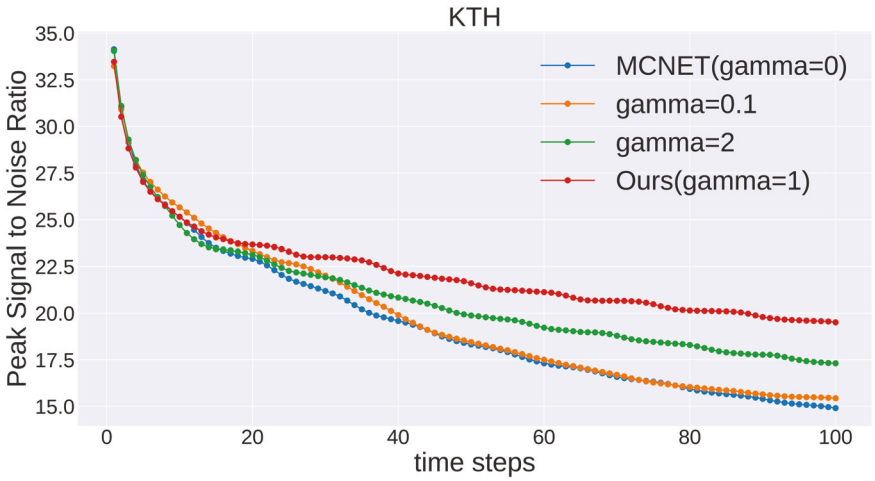

Figure 5.10: Quantitative comparison of the retrospection loss with different parameter γ.

loss and adversarial loss. We follow MCNET [144] to set these hyper-parameters as $\alpha_1 = \alpha_2 = 1$ in all experiments, $\beta_1 = \beta_2 = 0.02$ on KTH dataset, and $\beta_1 = \beta_2 = 0.001$ on UCF101 dataset. We analyze the selection the newly introduced parameter of retrospection loss γ_1 and γ_2. When $\gamma_1(\gamma_2) = 0$, our model are equivalent to MCNET. Figure 5.10 shows results of $\gamma_1(\gamma_2) = \{0.1, 1, 2\}$. We find $\gamma_1(\gamma_2) = 0.1$ is too small, and the performance drops to be similar to MCNET. $\gamma_1(\gamma_2) = 2$ is too large, which means the model emphasizes too much on the auxiliary network and ignore the importance for producing the accurate prediction. In all, $\gamma_1(\gamma_2) = 1$ achieves the best performance, so we set $\gamma_1(\gamma_2) = 1$ in our model.

5.5.3.3 Adversarial Loss

We explore the contribution of the adversarial loss in our model. We analyze the results of the function without the adversarial loss. The function without adversarial loss can be expressed as follow:

$$\mathcal{L}_{no\_adv}(G, F) = \alpha_1 \mathcal{L}_{img}(G) + \alpha_2 \mathcal{L}_{img}(F) + \gamma_1 \mathcal{L}_{ret1}(F, G) + \gamma_2 \mathcal{L}_{ret2}(F, G). \quad (5.21)$$

For fair comparison, we set $\alpha_1 = \alpha_2 = 1$ and $\gamma_1 = \gamma_2 = 1$, which are the same

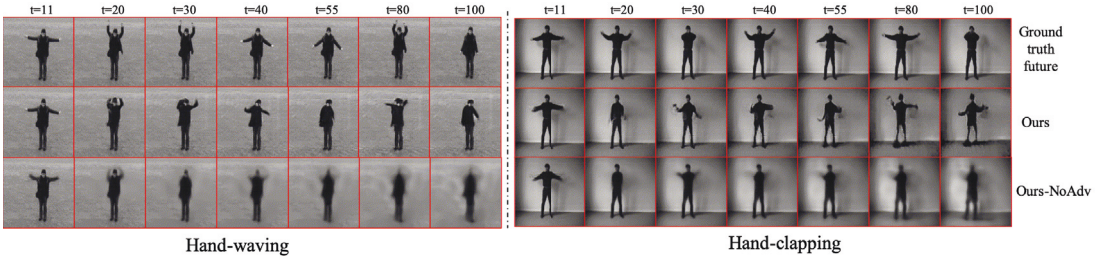

Figure 5.11: Qualitative analysis for the function of the adversarial loss. First Row: ground-truth frames; Second Row: our results with full objective; Third Row: results without adversarial loss in Equation 5.21.

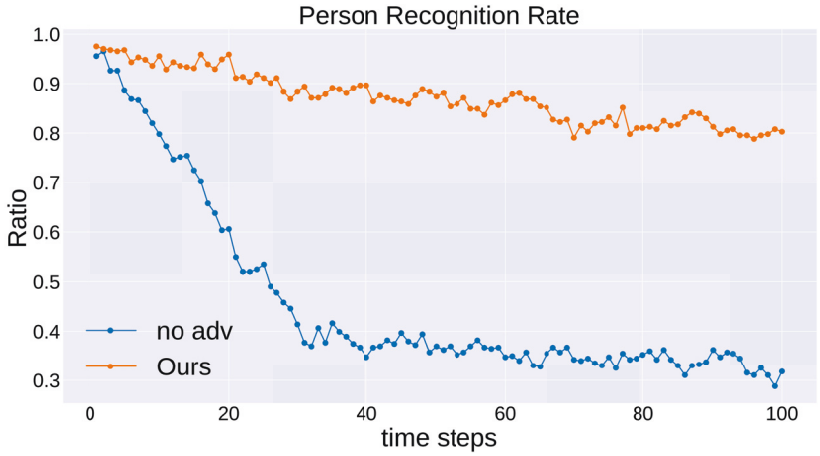

Figure 5.12: Ablation study for the function of the adversarial loss in terms of the recognition rate. The recognition rate of the person detector that a person is recognized in the predicted frame.

with our model. Results are demonstrated in Figure 5.3, which are denoted as "Ours-No Adv". We observe that the prediction frames are blurry and hard to be recognized. We also test the person recognition rate. As shown in Figure 5.12, the model without adversarial loss results in dramatically decrease in terms of person recognition rate. At t=[50,100], it turns out that only 34.5% generated frames could be recognized as person. In comparison, our method with adversarial losses produce 83.0% predictions with recognizable persons. The reason is that $\mathcal{L}_{img}$ guides the model to match the average pixel of all possible future frames. Training to generate average sequences, however, results in somewhat blurry generations. The adversarial loss allows our model to predict realistic looking frames.

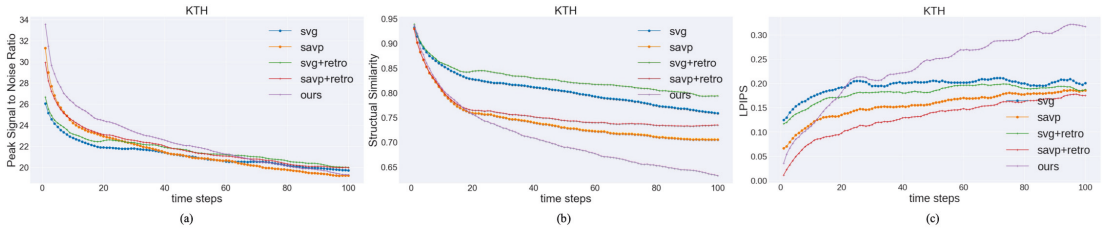

Figure 5.13: Quantitative comparisons between our method, SAVG, SVG w/o the retrospection process in terms of PSNR, SSIM and LPIPS.

5.6 Retrospection for Other Models

In our paper, we apply the retrospective training procedure to MCNET, which is a simple framework and outputs deterministic future frames. However, accurately predicting future for long-term is challenging, since the ambiguous nature of the problem would cause the model to average together possible futures into a simple, blurry prediction. Recently, works for stochastic video prediction methods have emerged that output a sample of possible future distribution. The stochastic video prediction methods are a promising solution to obtain realistic and sharp future frames, e.g., SAVP [73] and SVG [28]. In this section, we compare our model to SAVP and SVG, and extend those methods by incorporating the proposed retrospective training procedure.

For a fair comparison, we train the SVG, SAVP model by using the same data and image size to our model. Specifically, we use person 1-16 for training and 17-25 for testing, and the image size is of 128 $\times$ 128. The model is trained by observing 10 frames and predicting 10 subsequent frames. The code is used from authors' homepages[1] [2]. In the test phase, we predict 100 subsequent frames given the previous 10 frames as input. By incorporating the retrospection processing, we first train a retrospective generator F by feeding the true future frames and encourage the network F to generate preceding frames. Then we retrain the prediction network G by adding the retrospection loss, so that the predicted future frames can be enhanced for the reconstruction of past frames. Figure 5.3 demonstrates the qualitative comparison to our methods. Though SVG could

[1] `https://github.com/alexlee-gk/video_prediction`
[2] `https://github.com/edenton/svg`

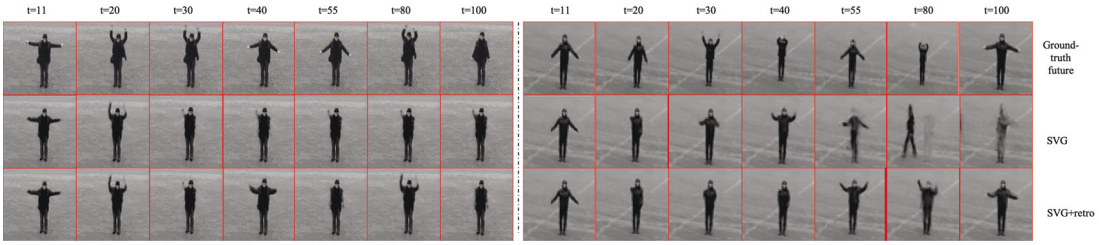

Figure 5.14: Qualitative comparison between SVG and "SVG+retro" on the action of hand-waving. "SVG+retro" incorporating the retrospection process.

output unblurred and realistic frames for a long-term horizon, we notice that after a few times step, the motion information in the SVG might be lost or incorrect. For instance, in the bottom row of Figure 5.3, the generated person keeps a fixed gesture after t=30. Figure 5.14 shows the comparisons between SVG and SVG by incorporating the retrospection process (denoted as "SVG+retro"). We find that with the retrospection process, the motion information can be preserved. For instance, in the right cases of Figure 5.14, the generated person of "SVG+retro" keeps hand-waving all the time. In constrast the person of SVG becomes the action of walking after t=80. The reason is that the introduced retrospection process tends to keep the action of prediction consistent with that in the past frames.

Results in Figure 5.13 shows the quantitative comparisons between our methods, SVG, and "SVG+retro". We observe that our method outperforms the compared method in terms of PSNR, while we get a lower score in terms of perceptual metrics, SSIM and LPIPS. Since our methods are merely based on the deterministic prediction, our results can get close to the ground truth future frames. However, compared stochastic-based methods achieve more realistic frames in terms of human perception. As shown in the plot of "SVG+retro" and "SAVP+retro", the performance has been improved, which demonstrates that our method can improve the stochastic based video prediction method as well.

5.7 Conclusion

We have proposed a long-term video prediction model via criticization and retrospection in the natural video. Our model consists of a prediction network and a

retrospection network. The prediction network is used to generate future frames given a set of consecutive frames. The retrospection network aims to look back to rectify the prediction deficiencies. In addition to considering the discrepancy between the predicted frames and ground truth frames, we feed the predicted frames to the retrospection network to minimize the discrepancy between the retrospective frames and the observed frames. To optimize the prediction network and the retrospection network, an auxiliary route is built by reversing the flow of time and executing a similar retrospection. Our method can alleviate the accumulation deviation during the recursive prediction process. Extensive experiments demonstrate the effectiveness of the proposed model on long-term video prediction.

Chapter 6

Conclusions

Visual synthesis is one of the fundamental problems in computer vision with the aspect of theory (e.g., generative model, unsupervised learning) and applications (e.g., art creation, super-resolution, video and image processing). In this thesis, we study the challenging problem of visual data synthesis. The proposed visual synthesis algorithms improve interpretation and controllability to generate object in our desired region, efficient and effective to achieve multi-domain synthesis in a single model, as well as achieve long-term video prediction. The summarization of the proposed methods will be given in this chapter and we will discuss some future directions on the algorithms and applications.

6.1 Summary of Conclusions

First, the attention-mechanism is incorporated into the generative adversarial model for image-to-image transformation. With the proposed method, attention-GAN is able to transform the object of the source image to the target class of object. We decompose the traditional generative network into two subnetworks: attention network and transformation network. Each of them is responsible for one subtask. The attention network is used to localize the region of the object and the transformation is used to generate the transformed object. In addition, to improve the quality of transformed images, a perceptual loss is introduced by minimizing the feature distance between the translations and overall samples

of the target domain. We show that our model has advantages over the one-shot generation method in preserving background consistency and transformation quality.

Then, an effective model is proposed for multi-domain image synthesis. An adversarial gated network (Gated-GAN) is designed to achieve multi-style image synthesis in a single network. The generative network of the proposed Gated-GAN consists of an encoder, a gated transformer, and a decoder. Different styles can be achieved by passing input images through different branches of the gated transformer. The discriminative networks are used to distinguish whether the input image is a stylized or genuine image. An auxiliary classifier is used to recognize the style categories of transferred images, thereby helping the generative networks generate images in multiple styles. An auto-encoder reconstruction loss is also introduced by combining the encoder and decoder directly. The proposed auto-encoder reconstruction loss is able to stabilize the training process.

Third, long-term video synthesis is investigated by considering temporal consistency. The thesis proposes a retrospection process to look back on what has been learned in the video generation process and rectify generation errors. An auxiliary route is built by reversing the flow of time, which assists the training of the retrospection network. These two routes interact with each other to boost the performance of retrospection network and enhance the understanding of dynamics across frames, especially for the long-term horizon. Extensive experiments on natural video datasets demonstrate the advantage of introducing the retrospection process in long-term video prediction.

6.2 Future Works

Although some useful methods have been present to improve the performance of visual data synthesis, there is still a way to go in this area. Future works will focus on synthesizing more realistic and challenging visual data (e.g., high-resolution (2K/4K) images, 3D point clouds data), as well as generating visual data conditioned on semantic and more complex information.

- **High-resolution data synthesis.** Our methods provide controllable, flex-

ible, and robust visual data synthesis algorithms, but they are limited in the resolution of up to 256×256. The main reason is that it is difficult to train GANs to produce high-resolution images, due to training instability and optimization issues. These challenges call for better network architectures, as well as more robust loss functions and stable training procedures. With the development of high-speed networks (e.g., 5G network) and high-resolution display devices (e.g., 4K screen), high resolution and fidelity visual data are in high demand. In the future, end-to-end training with more compact, simple, and memory-efficient architecture is a potential direction.

- **3D data synthesis.** This thesis focuses on synthesizing 2D visual data, such as images and video frames, but does not involve 3D data. The challenges lie in the inherent drawback of inefficient representation as the dimensionality per 3D shape sample can vary. With the development of AR/VR, self-driving cars, the 3D vision problem becomes more and more important since it provides much richer information than 2D. In the future, generating 3D visual data (e.g., RGB-D or 3D cloud point) is a promising and significant problem. How to efficiently fit and generate the 3D data presentation within conventional CNNs remains a problem.

- **Multiple sources information.** Visual synthesis is the process of creating pixel-level data (e.g., images, and videos) from some forms of descriptions. My thesis mainly solves the visual synthesis from images. In the future, I hope to explore more interesting and challenging visual data applications, such as texts-to-image, virtual clothing try-on, dance creation from music, etc.

References

[1] A. Akl, C. Yaacoub, M. Donias, J. P. D. Costa, and C. Germain, "Texture synthesis using the structure tensor," *IEEE Transactions on Image Processing*, vol. 24, no. 11, pp. 4082–4095, Nov 2015. 56

[2] H. A. Aly and E. Dubois, "Image up-sampling using total-variation regularization with a new observation model," *IEEE Transactions on Image Processing*, vol. 14, no. 10, pp. 1647–1659, 2005. 65

[3] M. Arjovsky and L. Bottou, "Towards principled methods for training generative adversarial networks," *arXiv preprint arXiv:1701.04862*, 2017. 62, 79

[4] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein generative adversarial networks," in *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, ser. Proceedings of Machine Learning Research, D. Precup and Y. W. Teh, Eds., vol. 70. PMLR, 2017, pp. 214–223. [Online]. Available: http://proceedings.mlr.press/v70/arjovsky17a.html 16, 25, 39

[5] N. Ashikhmin, "Fast texture transfer," *IEEE Computer Graphics and Applications*, vol. 23, no. 4, pp. 38–43, 2003. 53, 56

[6] S. Avidan and A. Shamir, "Seam carving for content-aware image resizing," *ACM Trans. Graph.*, vol. 26, no. 3, p. 10, 2007. [Online]. Available: https://doi.org/10.1145/1276377.1276390 5

[7] M. Babaeizadeh, C. Finn, D. Erhan, R. H. Campbell, and S. Levine, "Stochastic variational video prediction," *arXiv preprint arXiv:1710.11252*, 2017. 19, 88

[8] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," in *3th International Conference on Learning Representations (ICLR)*, April 2015. 27

[9] N. Ballas, L. Yao, C. Pal, and A. C. Courville, "Delving deeper into convolutional networks for learning video representations," *CoRR*, vol. abs/1511.06432, 2015. [Online]. Available: http://arxiv.org/abs/1511.06432 87

[10] A. Bansal, S. Ma, D. Ramanan, and Y. Sheikh, "Recycle-gan: Unsupervised video retargeting," in *The European Conference on Computer Vision (ECCV)*, September 2018. 7, 18

[11] C. Barnes, E. Shechtman, A. Finkelstein, and D. B. Goldman, "Patchmatch: a randomized correspondence algorithm for structural image editing," *ACM Trans. Graph.*, vol. 28, no. 3, p. 24, 2009. [Online]. Available: https://doi.org/10.1145/1531326.1531330 5

[12] S. Benaim and L. Wolf, "One-sided unsupervised domain mapping," in *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, 2017, pp. 752–762. 17, 22, 23

[13] U. Bergmann, N. Jetchev, and R. Vollgraf, "Learning texture manifolds with the periodic spatial gan," in *Thirty-fourth International Conference on Machine Learning (ICML)*, 2017. 58

[14] D. Berthelot, T. Schumm, and L. Metz, "Began: Boundary equilibrium generative adversarial networks," *arXiv preprint arXiv:1703.10717*, 2017. 25

[15] M. Bińkowski, D. J. Sutherland, M. Arbel, and A. Gretton, "Demystifying mmd gans," *arXiv preprint arXiv:1801.01401*, 2018. 37

[16] A. Brock, J. Donahue, and K. Simonyan, "Large scale GAN training for high fidelity natural image synthesis," in *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. [Online]. Available: https://openreview.net/forum?id=B1xsqj09Fm 1, 6, 88

[17] W. Byeon, Q. Wang, R. Kumar Srivastava, and P. Koumoutsakos, "Contextvp: Fully context-aware video prediction," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 753–769. 87

[18] C. Chan, S. Ginosar, T. Zhou, and A. A. Efros, "Everybody dance now," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 5933–5942. 101

[19] D. Chen, L. Yuan, J. Liao, N. Yu, and G. Hua, "Stylebank: An explicit representation for neural image style transfer," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017. 53, 58, 71

[20] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs," *IEEE transactions on pattern analysis and machine intelligence*, vol. 40, no. 4, pp. 834–848, 2017. 1

[21] Q. Chen and V. Koltun, "Photographic image synthesis with cascaded refinement networks," in *The IEEE International Conference on Computer Vision (ICCV)*, vol. 1, 2017. 44

[22] X. Chen, C. Xu, X. Yang, L. Song, and D. Tao, "Gated-gan: Adversarial gated networks for multi-collection style transfer," *IEEE Transactions on Image Processing*, vol. 28, no. 2, pp. 546–560, 2 2019. 88

[23] X. Chen, Y. Duan, R. Houthooft, J. Schulman, I. Sutskever, and P. Abbeel, "Infogan: Interpretable representation learning by information maximizing generative adversarial nets," in *Advances in Neural Information Processing Systems*, 2016, pp. 2172–2180. 14, 59

[24] X. Chen, C. Xu, X. Yang, and D. Tao, "Attention-gan for object transfiguration in wild images," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 164–180. xv, 5, 33, 39, 40

[25] S. Chiappa, S. Racanière, D. Wierstra, and S. Mohamed, "Recurrent environment simulators," *CoRR*, vol. abs/1704.02254, 2017. [Online]. Available: http://arxiv.org/abs/1704.02254 85, 88

[26] A. Clark, J. Donahue, and K. Simonyan, "Efficient video generation on complex datasets," *arXiv preprint arXiv:1907.06571*, 2019. 7, 18

[27] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on.* IEEE, 2009, pp. 248–255. 37

[28] E. Denton and R. Fergus, "Stochastic video generation with a learned prior," *arXiv preprint arXiv:1802.07687*, 2018. 89, 111

[29] E. L. Denton and V. Birodkar, "Unsupervised learning of disentangled representations from video," in *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, 2017, pp. 4417–4426. 19, 85, 87

[30] C. Dong, C. C. Loy, K. He, and X. Tang, "Learning a deep convolutional network for image super-resolution," in *European Conference on Computer Vision.* Springer, 2014, pp. 184–199. 14

[31] ——, "Image super-resolution using deep convolutional networks," *IEEE transactions on pattern analysis and machine intelligence*, vol. 38, no. 2, pp. 295–307, 2016. 17, 21

[32] A. Dosovitskiy, P. Fischer, E. Ilg, P. Hausser, C. Hazirbas, V. Golkov, P. Van Der Smagt, D. Cremers, and T. Brox, "Flownet: Learning optical flow with convolutional networks," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 2758–2766. 89

[33] A. Dosovitskiy, J. Tobias Springenberg, and T. Brox, "Learning to generate chairs with convolutional neural networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 1538–1546. 14

[34] A. A. Efros and W. T. Freeman, "Image quilting for texture synthesis and transfer," in *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH 2001, Los Angeles, California, USA, August 12-17, 2001*, L. Pocock, Ed. ACM, 2001, pp. 341–346. [Online]. Available: https://doi.org/10.1145/383259 5, 56

[35] A. A. Efros and T. K. Leung, "Texture synthesis by non-parametric sampling," in *Proceedings of the International Conference on Computer Vision, Kerkyra, Corfu, Greece, September 20-25, 1999*. IEEE Computer Society, 1999, pp. 1033–1038. [Online]. Available: http://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=6412 5, 53, 56

[36] M. Elad and P. Milanfar, "Style transfer via texture synthesis," *IEEE Transactions on Image Processing*, vol. 26, no. 5, pp. 2338–2351, 2017. 56

[37] C. Finn, I. Goodfellow, and S. Levine, "Unsupervised learning for physical interaction through video prediction," in *Advances in Neural Information Processing Systems (NIPS)*, 2016, pp. 64–72. 1, 85, 87, 88

[38] O. Frigo, N. Sabater, J. Delon, and P. Hellier, "Split and match: Example-based adaptive patch sampling for unsupervised style transfer," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 553–561. 56

[39] Z. Gan, L. Chen, W. Wang, Y. Pu, Y. Zhang, H. Liu, C. Li, and L. Carin, "Triangle generative adversarial networks," in *Advances in Neural Information Processing Systems (NIPS)*, 2017. 14

[40] L. A. Gatys, A. S. Ecker, and M. Bethge, "Image style transfer using convolutional neural networks," in *Proceedings of the IEEE Conference on Com-*

puter Vision and Pattern Recognition, 2016, pp. 2414–2423. xvii, 53, 57, 58, 70, 71, 72

[41] M. E. Gheche, J. F. Aujol, Y. Berthoumieu, and C. A. Deledalle, "Texture reconstruction guided by a high-resolution patch," *IEEE Transactions on Image Processing*, vol. 26, no. 2, pp. 549–560, Feb 2017. 56

[42] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 580–587. 14

[43] I. Goodfellow, "Nips 2016 tutorial: Generative adversarial networks," *arXiv preprint arXiv:1701.00160*, 2016. 4

[44] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in neural information processing systems*, 2014, pp. 2672–2680. 1, 14, 16, 22, 25, 39, 54, 58, 61, 85, 88

[45] L. Gorelick, M. Blank, E. Shechtman, M. Irani, and R. Basri, "Actions as space-time shapes," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 29, no. 12, pp. 2247–2253, 2007. [Online]. Available: https://doi.org/10.1109/TPAMI.2007.70711 86, 99, 101, 104

[46] K. Gregor, I. Danihelka, A. Graves, D. Rezende, and D. Wierstra, "Draw: A recurrent neural network for image generation," in *Proceedings of the 32nd International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, F. Bach and D. Blei, Eds., vol. 37. Lille, France: PMLR, 07–09 Jul 2015, pp. 1462–1471. [Online]. Available: http://proceedings.mlr.press/v37/gregor15.html 28

[47] R. A. Güler, N. Neverova, and I. Kokkinos, "Densepose: Dense human pose estimation in the wild," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 7297–7306. 1

[48] X. Han, Z. Wu, Z. Wu, R. Yu, and L. S. Davis, "Viton: An image-based virtual try-on network," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. 5

[49] J. Hays and A. A. Efros, "Scene completion using millions of photographs," *ACM Trans. Graph.*, vol. 26, no. 3, p. 4, 2007. 5

[50] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," in *2015 IEEE International Conference on Computer Vision, ICCV 2015, Santiago, Chile, December 7-13, 2015*, 2015, pp. 1026–1034. 98

[51] ——, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778. 1, 59

[52] A. Hertzmann, C. E. Jacobs, N. Oliver, B. Curless, and D. H. Salesin, "Image analogies," in *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*. ACM, 2001, pp. 327–340. 53, 56

[53] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, "Gans trained by a two time-scale update rule converge to a local nash equilibrium," in *Advances in Neural Information Processing Systems*, 2017, pp. 6629–6640. 67

[54] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural computation*, vol. 18, no. 7, pp. 1527–1554, 2006. 14

[55] T. Igarashi, T. Moscovich, and J. F. Hughes, "As-rigid-as-possible shape manipulation," *ACM Trans. Graph.*, vol. 24, no. 3, pp. 1134–1141, 2005. [Online]. Available: https://doi.org/10.1145/1073204.1073323 5

[56] D. J. Im, C. D. Kim, H. Jiang, and R. Memisevic, "Generating images with recurrent adversarial networks," *arXiv preprint arXiv:1602.05110*, 2016. 25

[57] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, 2015, pp. 448–456. 96, 98

[58] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," *arXiv preprint arXiv:1611.07004*, 2016. 35, 64

[59] ——, "Image-to-image translation with conditional adversarial networks," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017. 1, 6, 8, 14, 17, 21, 26, 33, 58

[60] N. Jetchev, U. Bergmann, and R. Vollgraf, "Texture synthesis with spatial generative adversarial networks," *arXiv preprint arXiv:1611.08207*, 2016. 58

[61] J. Johnson, A. Alahi, and L. Fei-Fei, "Perceptual losses for real-time style transfer and super-resolution," in *European Conference on Computer Vision*. Springer, 2016, pp. 694–711. 5, 33, 35, 53, 55, 57, 65

[62] N. Kalchbrenner, A. van den Oord, K. Simonyan, I. Danihelka, O. Vinyals, A. Graves, and K. Kavukcuoglu, "Video pixel networks," in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, 2017, pp. 1771–1779. 87

[63] L. Karacan, Z. Akata, A. Erdem, and E. Erdem, "Learning to generate images of outdoor scenes from attributes and semantic layouts," *arXiv preprint arXiv:1612.00215*, 2016. 21

[64] T. Karras, T. Aila, S. Laine, and J. Lehtinen, "Progressive growing of gans for improved quality, stability, and variation," in *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018. [Online]. Available: https://openreview.net/forum?id=Hk99zCeAb 1, 6

[65] Y. Kataoka, T. Matsubara, and K. Uehara, "Image generation using generative adversarial networks and attention mechanism," in *Computer and Information Science (ICIS), 2016 IEEE/ACIS 15th International Conference on*. IEEE, 2016, pp. 1–6. 28

[66] T. Kim, M. Cha, H. Kim, J. K. Lee, and J. Kim, "Learning to discover cross-domain relations with generative adversarial networks," in *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, 2017, pp. 1857–1865. 17, 22, 24, 26, 29, 31, 39, 40, 41, 55, 59

[67] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014. 6, 35, 65, 99

[68] N. Kodali, J. Abernethy, J. Hays, and Z. Kira, "On convergence and stability of gans," *arXiv preprint arXiv:1705.07215*, 2017. 5, 16

[69] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems 25*, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2012, pp. 1097–1105. 57

[70] H. Kwak and B.-T. Zhang, "Generating images part by part with composite generative adversarial networks," *arXiv preprint arXiv:1607.05387*, 2016. 25

[71] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015. 13, 14

[72] C. Ledig, L. Theis, F. Huszár, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang *et al.*, "Photo-realistic single image super-resolution using a generative adversarial network," *arXiv preprint arXiv:1609.04802*, 2016. 6, 23, 33, 58

[73] A. X. Lee, R. Zhang, F. Ebert, P. Abbeel, C. Finn, and S. Levine, "Stochastic adversarial video prediction," *arXiv preprint arXiv:1804.01523*, 2018. 89, 111

[74] H. Lee, S. Seo, S. Ryoo, and K. Yoon, "Directional texture transfer," in *Proceedings of the 8th International Symposium on Non-Photorealistic Animation and Rendering*. ACM, 2010, pp. 43–48. 53, 56

[75] Y. J. Lee, C. L. Zitnick, and M. F. Cohen, "Shadowdraw: real-time user guidance for freehand drawing," *ACM Trans. Graph.*, vol. 30, no. 4, p. 27, 2011. 5

[76] C. Li and M. Wand, "Combining markov random fields and convolutional neural networks for image synthesis," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 2479–2486. 57, 58

[77] ——, "Precomputed real-time texture synthesis with markovian generative adversarial networks," in *European Conference on Computer Vision*. Springer, 2016, pp. 702–716. 17, 35, 58, 64

[78] J. Li, A. Madry, J. Peebles, and L. Schmidt, "Towards understanding the dynamics of generative adversarial networks," *arXiv preprint arXiv:1706.09884*, 2017. 5, 16

[79] Y. Li, S. Tang, R. Zhang, Y. Zhang, J. Li, and S. Yan, "Asymmetric gan for unpaired image-to-image translation," *IEEE Transactions on Image Processing*, pp. 1–1, 2019. 88

[80] Y. Li, C. Fang, J. Yang, Z. Wang, X. Lu, and M.-H. Yang, "Diversified texture synthesis with feed-forward networks," *arXiv preprint arXiv:1703.01664*, 2017. 53, 58, 71

[81] ——, "Universal style transfer via feature transforms," in *Advances in Neural Information Processing Systems*, 2017, pp. 385–395. xvii, 58, 71, 72, 73

[82] X. Liang, L. Lee, W. Dai, and E. P. Xing, "Dual motion GAN for future-flow embedded video prediction," in *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, 2017, pp. 1762–1770. [Online]. Available: https://doi.org/10.1109/ICCV.2017.194 85, 88

[83] X. Liang, H. Zhang, L. Lin, and E. Xing, "Generative semantic manipulation with mask-contrasting gan," pp. 558–573, 2018. 26

[84] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *European conference on computer vision*. Springer, 2014, pp. 740–755. 37, 42

[85] M. Liu, X. He, and M. Salzmann, "Geometry-aware deep network for single-image novel view synthesis," *CoRR*, vol. abs/1804.06008, 2018. [Online]. Available: http://arxiv.org/abs/1804.06008 88

[86] M.-Y. Liu, T. Breuel, and J. Kautz, "Unsupervised image-to-image translation networks," in *Advances in Neural Information Processing Systems*, 2017, pp. 700–708. 26, 39, 40, 41

[87] M.-Y. Liu and O. Tuzel, "Coupled generative adversarial networks," in *Advances in neural information processing systems*, 2016, pp. 469–477. 26

[88] W. Liu, W. Luo, D. Lian, and S. Gao, "Future frame prediction for anomaly detection–a new baseline," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 6536–6545. 89

[89] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 3431–3440. 21

[90] W. Lotter, G. Kreiman, and D. D. Cox, "Unsupervised learning of visual structure using predictive generative networks," *CoRR*, vol. abs/1511.06380, 2015. [Online]. Available: http://arxiv.org/abs/1511.06380 14, 58, 92

[91] ——, "Deep predictive coding networks for video prediction and unsupervised learning," *CoRR*, vol. abs/1605.08104, 2016. 87

[92] J. Lu, A. Kannan, J. Yang, D. Parikh, and D. Batra, "Best of both worlds: Transferring knowledge from discriminative learning to a generative vi-

sual dialog model," in *Advances in Neural Information Processing Systems (NIPS)*, 2017. 14

[93] F. Luan, S. Paris, E. Shechtman, and K. Bala, "Deep photo style transfer," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017. 53, 57

[94] L. Ma, X. Jia, Q. Sun, B. Schiele, T. Tuytelaars, and L. Van Gool, "Pose guided person image generation," *arXiv preprint arXiv:1705.09368*, 2017. 17, 26

[95] A. L. Maas, A. Y. Hannun, and A. Y. Ng, "Rectifier nonlinearities improve neural network acoustic models," in *Proc. icml*, vol. 30, no. 1, 2013, p. 3. 96

[96] A. Mahendran and A. Vedaldi, "Understanding deep image representations by inverting them," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 5188–5196. 57

[97] R. Mahjourian, M. Wicke, and A. Angelova, "Geometry-based next frame prediction from monocular video," *CoRR*, vol. abs/1609.06377, 2016. [Online]. Available: http://arxiv.org/abs/1609.06377 88

[98] X. Mao, Q. Li, H. Xie, R. Y. Lau, Z. Wang, and S. Paul Smolley, "Least squares generative adversarial networks," in *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017. 25, 28, 39, 61

[99] T. Marwah, G. Mittal, and V. N. Balasubramanian, "Attentive semantic video generation using captions," in *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, 2017, pp. 1435–1443. 87

[100] M. Mathieu, C. Couprie, and Y. LeCun, "Deep multi-scale video prediction beyond mean square error," *CoRR*, vol. abs/1511.05440, 2015. [Online]. Available: http://arxiv.org/abs/1511.05440 17, 89

[101] Y. A. Mejjati, C. Richardt, J. Tompkin, D. Cosker, and K. I. Kim, "Unsupervised attention-guided image-to-image translation," in *Advances in Neural Information Processing Systems*, 2018, pp. 3693–3703. xv, 27, 37, 39, 40, 41, 42

[102] M. Mirza and S. Osindero, "Conditional generative adversarial nets," *arXiv preprint arXiv:1411.1784*, 2014. 17, 26, 59, 75, 95

[103] R. Mittelman, B. Kuipers, S. Savarese, and H. Lee, "Structured recurrent temporal restricted boltzmann machines," in *Proceedings of the 31th International Conference on Machine Learning, ICML 2014, Beijing, China, 21-26 June 2014*, 2014, pp. 1647–1655. 87

[104] V. Mnih, N. Heess, A. Graves *et al.*, "Recurrent models of visual attention," in *Advances in neural information processing systems*, 2014, pp. 2204–2212. 27

[105] A. Nguyen, J. Clune, Y. Bengio, A. Dosovitskiy, and J. Yosinski, "Plug & play generative networks: Conditional iterative generation of images in latent space," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 14

[106] S. Ning, H. Xu, L. Song, R. Xie, and W. Zhang, "Learning an inverse tone mapping network with a generative adversarial regularizer," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2018, Calgary, AB, Canada, April 15-20, 2018*. IEEE, 2018, pp. 1383–1387. [Online]. Available: https://doi.org/10.1109/ICASSP.2018.8462444 17

[107] A. Odena, "Semi-supervised learning with generative adversarial networks," *arXiv preprint arXiv:1606.01583*, 2016. 59

[108] A. Odena, C. Olah, and J. Shlens, "Conditional image synthesis with auxiliary classifier gans," in *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, 2017, pp. 2642–2651. 53, 59, 75

[109] J. Oh, X. Guo, H. Lee, R. L. Lewis, and S. Singh, "Action-conditional video prediction using deep networks in atari games," in *Advances in neural information processing systems*, 2015, pp. 2863–2871. 85, 88

[110] D. Pathak, P. Krähenbühl, J. Donahue, T. Darrell, and A. A. Efros, "Context encoders: Feature learning by inpainting," in *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*. IEEE Computer Society, 2016, pp. 2536–2544. [Online]. Available: https://doi.org/10.1109/CVPR.2016.278 6, 17

[111] V. Patraucean, A. Handa, and R. Cipolla, "Spatio-temporal video autoencoder with differentiable memory," *CoRR*, vol. abs/1511.06309, 2015. [Online]. Available: http://arxiv.org/abs/1511.06309 87

[112] X. Peng, J. Feng, S. Xiao, W. Yau, J. T. Zhou, and S. Yang, "Structured autoencoders for subspace clustering," *IEEE Transactions on Image Processing*, vol. 27, no. 10, pp. 5076–5086, Oct 2018. 57

[113] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," *arXiv preprint arXiv:1511.06434*, 2015. 16, 25, 55

[114] M. Ranzato, A. Szlam, J. Bruna, M. Mathieu, R. Collobert, and S. Chopra, "Video (language) modeling: a baseline for generative models of natural videos," *arXiv preprint arXiv:1412.6604*, 2014. 87

[115] L. J. Ratliff, S. A. Burden, and S. S. Sastry, "Characterization and computation of local nash equilibria in continuous games," in *2013 51st Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, Oct 2013, pp. 917–924. 15

[116] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," *arXiv*, 2018. 1

[117] S. Reed, Z. Akata, X. Yan, L. Logeswaran, B. Schiele, and H. Lee, "Generative adversarial text to image synthesis," in *Proceedings of The 33rd International Conference on Machine Learning*, vol. 3, 2016. 89

[118] S. E. Reed, Z. Akata, S. Mohan, S. Tenka, B. Schiele, and H. Lee, "Learning what and where to draw," in *Advances in Neural Information Processing Systems*, 2016, pp. 217–225. 17, 26

[119] E. Reinhard, M. Ashikhmin, B. Gooch, and P. Shirley, "Color transfer between images," *IEEE Computer Graphics and Applications*, vol. 21, no. 5, pp. 34–41, 2001. 5

[120] R. A. Rensink, "The dynamic representation of scenes," *Visual cognition*, vol. 7, no. 1-3, pp. 17–42, 2000. 27

[121] L. I. Rudin, S. Osher, and E. Fatemi, "Nonlinear total variation based noise removal algorithms," *Physica D: Nonlinear Phenomena*, vol. 60, no. 1-4, pp. 259–268, 1992. 65

[122] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "ImageNet Large Scale Visual Recognition Challenge," *International Journal of Computer Vision (IJCV)*, vol. 115, no. 3, pp. 211–252, 2015. 1

[123] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, "Improved techniques for training gans," in *Advances in Neural Information Processing Systems*, 2016, pp. 2234–2242. 16, 25, 35, 59, 63, 64

[124] P. Sangkloy, J. Lu, C. Fang, F. Yu, and J. Hays, "Scribbler: Controlling deep image synthesis with sketch and color," *arXiv preprint arXiv:1612.00835*, 2016. 17, 21, 26

[125] S. Santurkar, L. Schmidt, and A. Madry, "A classification-based perspective on gan distributions," 2018. 8, 16

[126] C. Schüldt, I. Laptev, and B. Caputo, "Recognizing human actions: A local SVM approach," in *17th International Conference on Pattern Recognition, ICPR 2004, Cambridge, UK, August 23-26, 2004.*, 2004, pp. 32–36. [Online]. Available: https://doi.org/10.1109/ICPR.2004.1334462 86

[127] A. Selim, M. Elgharib, and L. Doyle, "Painting style transfer for head portraits using convolutional neural networks," *ACM Transactions on Graphics (ToG)*, vol. 35, no. 4, p. 129, 2016. 57

[128] X. Shi, Z. Chen, H. Wang, D. Yeung, W. Wong, and W. Woo, "Convolutional LSTM network: A machine learning approach for precipitation nowcasting," in *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, 2015, pp. 802–810. 87, 99

[129] A. Shrivastava, T. Pfister, O. Tuzel, J. Susskind, W. Wang, and R. Webb, "Learning from simulated and unsupervised images through adversarial training," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017. 35, 66

[130] K. Simonyan, A. Vedaldi, and A. Zisserman, "Deep inside convolutional networks: Visualising image classification models and saliency maps," in *3th International Conference on Learning Representations Workshop (ICLR workshop)*, 2013. 27

[131] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *CoRR*, vol. abs/1409.1556, 2014. [Online]. Available: http://arxiv.org/abs/1409.1556 1, 34, 97, 98

[132] K. Soomro, A. R. Zamir, and M. Shah, "UCF101: A dataset of 101 human actions classes from videos in the wild," *CoRR*, vol. abs/1212.0402, 2012. [Online]. Available: http://arxiv.org/abs/1212.0402 86, 99, 105, 106

[133] J. T. Springenberg, A. Dosovitskiy, T. Brox, and M. Riedmiller, "Striving for simplicity: The all convolutional net," in *4th International Conference on Learning Representations Workshop (ICLR workshop)*, 2014. 27

[134] A. Srivastava, L. Valkov, C. Russell, M. U. Gutmann, and C. Sutton, "Veegan: Reducing mode collapse in gans us
ing implicit variational learning," in *Advances in Neural Information Processing Systems*, 2017, pp. 3308–3318. 8, 16

[135] N. Srivastava, E. Mansimov, and R. Salakhutdinov, "Unsupervised learning of video representations using lstms," in *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, 2015, pp. 843–852. 87, 99, 101

[136] B. Su, X. Ding, C. Liu, H. Wang, and Y. Wu, "Discriminative transformation for multi-dimensional temporal sequences," *IEEE Transactions on Image Processing*, vol. 26, no. 7, pp. 3579–3593, July 2017. 58

[137] I. Sutskever, G. E. Hinton, and G. W. Taylor, "The recurrent temporal restricted boltzmann machine," in *Advances in Neural Information Processing Systems 21, Proceedings of the Twenty-Second Annual Conference on Neural Information Processing Systems, Vancouver, British Columbia, Canada, December 8-11, 2008*, 2008, pp. 1601–1608. 87

[138] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 1–9. 57

[139] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 2818–2826. 37

[140] Y. Taigman, A. Polyak, and L. Wolf, "Unsupervised cross-domain image generation," April 2017. 17, 22, 59

[141] D. Ulyanov, V. Lebedev, A. Vedaldi, and V. S. Lempitsky, "Texture networks: Feed-forward synthesis of textures and stylized images." in *ICML*, 2016, pp. 1349–1357. 53, 57

[142] D. Ulyanov, A. Vedaldi, and V. Lempitsky, "Improved texture networks: Maximizing quality and diversity in feed-forward stylization and texture synthesis," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017. 53, 57

[143] A. van den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu, "Wavenet: A generative model for raw audio," in *9th ISCA Speech Synthesis Workshop*, pp. 125–125. 14

[144] R. Villegas, J. Yang, S. Hong, X. Lin, and H. Lee, "Decomposing motion and content for natural video sequence prediction," *CoRR*, vol. abs/1706.08033, 2017. [Online]. Available: http://arxiv.org/abs/1706.08033 xx, 19, 85, 87, 92, 96, 99, 101, 104, 105, 106, 107, 109

[145] R. Villegas, J. Yang, Y. Zou, S. Sohn, X. Lin, and H. Lee, "Learning to generate long-term future via hierarchical prediction," in *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, 2017, pp. 3560–3569. [Online]. Available: http://proceedings.mlr.press/v70/villegas17a.html 85, 106

[146] D. Vincent, S. Jonathon, and K. Manjunath, "A learned representation for artistic style," April 2017. 57, 71

[147] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan, "Show and tell: A neural image caption generator," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 3156–3164. xiv, 1, 2

[148] C. Vondrick, H. Pirsiavash, and A. Torralba, "Generating videos with scene dynamics," in *Advances In Neural Information Processing Systems (NIPS)*, 2016, pp. 613–621. 1, 14

[149] C. Vondrick and A. Torralba, "Generating the future with adversarial transformers," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 14

[150] C. Wang, C. Wang, C. Xu, and D. Tao, "Tag disentangled generative adversarial networks for object image re-rendering," in *Proceedings of the 26th International Joint Conference on Artificial Intelligence (IJCAI)*, 2017, pp. 2901–2907. 14

[151] C. Wang, C. Xu, C. Wang, and D. Tao, "Perceptual adversarial networks for image-to-image transformation," *IEEE Transactions on Image Processing*, vol. 27, no. 8, pp. 4066–4079, 2018. 1

[152] T.-C. Wang, M.-Y. Liu, J.-Y. Zhu, G. Liu, A. Tao, J. Kautz, and B. Catanzaro, "Video-to-video synthesis," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2018. 7, 18

[153] T.-C. Wang, M.-Y. Liu, J.-Y. Zhu, A. Tao, J. Kautz, and B. Catanzaro, "High-resolution image synthesis and semantic manipulation with conditional gans," *arXiv preprint arXiv:1711.11585*, 2017. 44

[154] X. Wang, A. Shrivastava, and A. Gupta, "A-fast-rcnn: Hard positive generation via adversary for object detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 2606–2615. 1

[155] N. Wichers, R. Villegas, D. Erhan, and H. Lee, "Hierarchical long-term video prediction without supervision," in *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, 2018, pp. 6033–6041. 19, 88, 98, 104

[156] S. Wu, M. Kan, S. Shan, and X. Chen, "Hierarchical attention for part-aware face detection," *International Journal of Computer Vision*, Mar 2019. [Online]. Available: https://doi.org/10.1007/s11263-019-01157-5 27

[157] T. Xiao, Y. Xu, K. Yang, J. Zhang, Y. Peng, and Z. Zhang, "The application of two-level attention models in deep convolutional neural network for fine-grained image classification," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 842–850. 27

[158] H. Xu and K. Saenko, "Ask, attend and answer: Exploring question-guided spatial attention for visual question answering," in *European Conference on Computer Vision*. Springer, 2016, pp. 451–466. 27

[159] J. Xu, B. Ni, Z. Li, S. Cheng, and X. Yang, "Structure preserving video prediction," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 1460–1469. 1, 19, 85, 88

[160] K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhudinov, R. Zemel, and Y. Bengio, "Show, attend and tell: Neural image caption generation with visual attention," in *International Conference on Machine Learning*, 2015, pp. 2048–2057. 27

[161] T. Xu, P. Zhang, Q. Huang, H. Zhang, Z. Gan, X. Huang, and X. He, "Attngan: Fine-grained text to image generation with attentional generative adversarial networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 1316–1324. 1

[162] T. Xue, J. Wu, K. Bouman, and B. Freeman, "Visual dynamics: Probabilistic future frame synthesis via cross convolutional networks," in *Advances in neural information processing systems*, 2016, pp. 91–99. 87

[163] H. Xun and B. Serge, "Arbitrary style transfer in real-time with adaptive instance normalization," April 2017. 57

[164] Y. Yan, B. Ni, W. Zhang, J. Xu, and X. Yang, "Structure-constrained motion sequence generation," *IEEE Transactions on Multimedia*, vol. 21, no. 7, pp. 1799–1812, July 2019. 7, 18

[165] Y. Yan, J. Xu, B. Ni, W. Zhang, and X. Yang, "Skeleton-aided articulated motion generation," in *Proceedings of the 25th ACM international conference on Multimedia*. ACM, 2017, pp. 199–207. 7, 18

[166] J. Yang, A. Kannan, D. Batra, and D. Parikh, "Lr-gan: Layered recursive generative adversarial networks for image generation," in *5th International Conference on Learning Representations (ICLR)*, 2017. 25

[167] L. Yao, A. Torabi, K. Cho, N. Ballas, C. Pal, H. Larochelle, and A. Courville, "Describing videos by exploiting temporal structure," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 4507–4515. 27

[168] Z. Yi, H. Zhang, P. T. Gong *et al.*, "Dualgan: Unsupervised dual learning for image-to-image translation," *arXiv preprint arXiv:1704.02510*, 2017. 59, 64

[169] Z. Yi, H. Zhang, P. Tan, and M. Gong, "Dualgan: Unsupervised dual learning for image-to-image translation," in *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017. 6, 17, 22, 24, 26, 31, 35, 39, 40, 41, 55

[170] J. S. Yoon, F. Rameau, J. Kim, S. Lee, S. Shin, and I. S. Kweon, "Pixel-level matching for video object segmentation using convolutional neural networks," in *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, 2017, pp. 2186–2195. 87

[171] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *European conference on computer vision*. Springer, 2014, pp. 818–833. 96

[172] M. D. Zeiler, G. W. Taylor, and R. Fergus, "Adaptive deconvolutional networks for mid and high level feature learning," in *IEEE International Conference on Computer Vision, ICCV 2011, Barcelona, Spain, November 6-13, 2011*, 2011, pp. 2018–2025. [Online]. Available: https://doi.org/10.1109/ICCV.2011.6126474 98

[173] H. Zhang, T. Xu, H. Li, S. Zhang, X. Wang, X. Huang, and D. N. Metaxas, "Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks," in *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017. xv, 14, 33, 43

[174] H. Zhang, B. S. Riggan, S. Hu, N. J. Short, and V. M. Patel, "Synthesis of high-quality visible faces from polarimetric thermal faces using generative

adversarial networks," *International Journal of Computer Vision*, Mar 2019. [Online]. Available: https://doi.org/10.1007/s11263-019-01175-3 26

[175] H. Zhang, V. Sindagi, and V. M. Patel, "Image de-raining using a conditional generative adversarial network," *CoRR*, vol. abs/1701.05957, 2017. [Online]. Available: http://arxiv.org/abs/1701.05957 6, 23

[176] K. Zhang, W. Luo, Y. Zhong, L. Ma, W. Liu, and H. Li, "Adversarial spatio-temporal learning for video deblurring," *IEEE Transactions on Image Processing*, vol. 28, no. 1, pp. 291–301, 1 2019. 87

[177] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang, "The unreasonable effectiveness of deep features as a perceptual metric," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 586–595. 23, 33, 101

[178] Y. Zhang, Z. Gan, and L. Carin, "Generating text via adversarial training," in *NIPS workshop on Adversarial Training*, 2016. 14

[179] Y. Zhou and T. L. Berg, "Learning temporal transformations from time-lapse videos," in *European conference on computer vision*. Springer, 2016, pp. 262–277. 17, 27

[180] Z. Zhou, H. Cai, S. Rong, Y. Song, K. Ren, W. Zhang, J. Wang, and Y. Yu, "Activation maximization generative adversarial nets," in *International Conference on Learning Representations*, 2018. 63

[181] H. Zhu, X. Peng, V. Chandrasekhar, L. Li, and J. Lim, "Dehazegan: When image dehazing meets differential programming," in *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden.*, 2018, pp. 1234–1240. [Online]. Available: https://doi.org/10.24963/ijcai.2018/172 58

[182] J.-Y. Zhu, P. Krähenbühl, E. Shechtman, and A. A. Efros, "Generative visual manipulation on the natural image manifold," in *European Conference on Computer Vision*. Springer, 2016, pp. 597–613. 17

[183] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017. xv, xvi, xviii, 1, 5, 6, 8, 14, 17, 22, 23, 24, 25, 26, 31, 35, 37, 39, 41, 42, 43, 50, 55, 59, 61, 64, 69, 72, 73, 74, 75, 88, 89, 99, 108

Publications

1. Xinyuan Chen, Chang Xu, Xiaokang Yang, Li Song, Dacheng Tao. Gated-GAN: Adversarial gated networks for multi-collection style transfer. *IEEE Transactions on Image Processing (TIP)*, 28(2), 546-560, 2019. (CCF A, IF: 5.071)

2. Xinyuan Chen, Chang Xu, Xiaokang Yang, Dacheng Tao. Attention-aware Object Transfiguration using Generative Adversarial Networks. *International Journal of Computer Vision (IJCV)*, 2019. (CCF A, IF: 11.541, under review)

3. Xinyuan Chen, Chang Xu, Xiaokang Yang, Dacheng Tao. Long-term Video Prediction via Criticization and Retrospection. *IEEE Transactions on Image Processing (TIP)*, 2019. (CCF A, under review)

4. Xinyuan Chen, Chang Xu, Xiaokang Yang, Dacheng Tao. Attention-GAN for object transfiguration in wild images. *Proceedings of the European Conference on Computer Vision (ECCV)*, 164-180, 2018. (CCF B)

5. Xinyuan Chen, Li Song, Xiaokang Yang. Deep RNNs for video denoising. *Applications of Digital Image Processing XXXIX. International Society for Optics and Photonics*, 9971: 99711T, 2016.