A DEEP LEARNING APPROACH FOR FASHION IMAGE PROCESSING WITH CONTROLLABLE SYNTHESIS AND FLEXIBLE EDITING

ZHENGWENTAI SUN
MPhil

The Hong Kong Polytechnic University
2024

The Hong Kong Polytechnic University
School of Fashion and Textiles

A Deep Learning Approach for Fashion Image Processing with Controllable Synthesis and Flexible Editing

Zhengwentai SUN

A thesis submitted in partial fulfillment of the requirements for the degree of Master of Philosophy

December 2023

CERTIFICATE OF ORIGINALITY

I hereby declare that this thesis is my own work and that, to the best of my knowledge and belief, it reproduces no material previously published or written, nor material that has been accepted for the award of any other degree or diploma, except where due acknowledgement has been made in the text.

(Signed)

_____

Zhengwentai SUN (Name of student)

_____

ABSTRACT

Fashion design typically involves composing elements and concepts, where designers select and harmonize colors, patterns, prints, and consider functional attributes like collar types, sleeve length, and overall fit. This process, reflecting the designer's creativity and market preferences, usually requires iterative modifications and can be time-consuming even for experts. Although recent advances in generative models offer efficient and effective way of processing of fashion images, applying these models in design remains challenging. The generative models primarily map random noise into an image, and the process is arbitrary and uncontrollable that requires multiple attempts to achieve a satisfactory image, meeting certain specific requirements.

A primary solution in enhancing the experience of generating desired garment images could involve detailed supervisory information. For instance, by collecting a fashion garment dataset with detailed annotations of each design element, the generative models could learn a conditional mapping from specific elements to the desired garment image. However, an obvious drawback of such a solution is the requirement of tedious annotation, which could be time-consuming and expensive. Moreover, those labels usually consider a discrete attribute where each element will be assigned to a category. When using such a model to consider the design process, its flexibility is limited as there are multiple design elements that are hard to categorize, e.g., colors and/or

textures.

To address the above-mentioned challenges in controllability and flexibility, this study develops generative models involving a decoupling method in the data collection and training. The overall motivation is to decouple a garment image into different modalities of data, each representing different design elements. For instance, the HED model is utilized to extract sketches that represent spatial level attributes like collars, lengths, and overall shapes. At the texture level, the cropped image patches are employed. These decoupled data, derived partially from the original garment images, are used to train generative models with the capable of reconstructing the original images. The trained model enables control over the synthesized garment image by selecting specific design elements during the inference stage.

Building on this capability, this thesis introduces an image processing system that involves two models: a controllable generation model and a flexible editing model, each targeting different fashion image processing tasks. The first model, called SGDiffs, focuses on the control over texture, the generation model leverages randomly cropped texture patches and text prompts to reconstruct garments. Once trained, it uses texture patches as decoupled style condition to control the synthesized garment images. Subsequently, an editing model, called CoDE-GAN, is introduced to modify the shape of fashion images. It learns the editing function by reconstructing masked images using

sketch maps. The two models can work independently or integratively as one system, enabling effective and flexible control in the generation and editing of fashion images. Both models have been comprehensively evaluated to demonstrate their specific advantages in comparison of other state-of-the-art models.

Keywords: Fashion image generation; image editing; generative adversarial network; diffusion model; decoupled conditions

PUBLICATIONS

1. Sun, Z., Zhou, Y., & Mok, P. Y. (2024). CoDE-GAN: Content Decoupled and Enhanced GAN for Sketch-guided Flexible Fashion Editing. Submitted to *ACM Transactions on Multimedia Computing, Communications and Applications*.
2. He, H., Zhou, Y., Sun, Z., Fan, J., & Mok, P. Y. (2024). Human Skeleton-aware Try-on via Fashion Landmarks and Garment Deformation. In *International Textile and Apparel Association Annual Conference Proceedings* (Vol. 80, No. 1). Iowa State University Digital Press.
3. Sun, Z., Zhou, Y., He, H., & Mok, P. Y. (2023, October). SGDiff: A Style Guided Diffusion Model for Fashion Synthesis. In *Proceedings of the 31st ACM International Conference on Multimedia* (pp. 8433-8442).
4. He, H., Sun, Z., Fan, J., & Mok, P. Y. (2023). SP3F-GAN: Generation Seamless Texture Maps for Fashion. *IADIS International Journal on Computer Science and Information Systems*, 18(2).
5. He, H., Sun, Z., Fan, J., & Mok, P. Y. (2023). SP3F-GAN: Seamless Texture Maps Generation by Adversarial Extension for Fashion. In *17th International Conference on Computer Graphics, Visualization, Computer Vision and Image Processing* (pp. 161-167).

ACKNOWLEDGEMENTS

I would like to express my deep gratitude to my supervisor, Dr. P. Y. Mok, for recognizing my potential and offering me an MPhil position at the Hong Kong Polytechnic University. Her support over the past two years, allowing me to explore various fields and guiding me through academic training, has been invaluable. Any progress I make in my career will undoubtedly be attributed to her significant influence.

Special thanks go to Dr. Zhou and Ms. Zhou for their enriching academic and technical discussions, and for sharing their valuable experiences of living in Hong Kong. Their kindness greatly eased my transition into a city with a culture and customs different from my own. I am also grateful to my colleagues in rooms ST604 and ST605. Our time spent working and exploring various scenes and restaurants in Hong Kong has been both enjoyable and a perfect balance between work and life.

My deepest appreciation extends to my parents for their unwavering support and respect for my decisions. Without their sacrifices, I could not pursue my passions. I also want to thank my friends for their encouragement, especially Mr. Cai, with whom I have traveled extensively, and Mr. Wan and Mr. Chen for their patience and support during challenging times.

Lastly, I wish to acknowledge the invaluable role of the university and the city of Hong Kong in broadening my understanding of the world's diversity and possibilities.

TABLES OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Chapter 1. INTRODUCTION

1.1 Research Background

Artificial intelligence (AI) has become a competitive necessity after decades of scientific fantasy, according to a report by the Deloitte Institute for Artificial Intelligence (Davenport, 2018). This advancement has significantly enhanced various aspects of the fashion design process. Furthermore, AI's role in simplifying and enhancing the design process is particularly evident in the domain of fashion. In this study, we refer to computer vision-based fashion technology as intelligent fashion. This is largely due to the visual nature of fashion, which has attracted many computer vision researchers to realize the immense potential of AI technology in this filed. The growing interest in intelligent fashion extends across the domain of computer vision and multimedia, as evidenced by the numerous applications of machine learning and neural networks with a fashion focus.

The advancements in computer vision, especially in the areas such as deep learning, have led to significant breakthroughs (Cheng et al., 2021). Figure 1-1 shows a few research applications of intelligent fashion. For instance, fashion clothing classification (Zhang et al., 2020) recognizes product attributes from fashion images, which benefits the analyses of fashion trend. For another example, fashion landmark localization (Qian et al., 2021) detects the key points of clothing which contributes more accurate

extraction and recognition of fashion attributes. Moreover, fashion parsing

Figure 1-1 *Several Research Work of Intelligent Fashion: (a) Fashion Clothing Classification and Attribute Recognition (Zhang et al., 2020), (b) Fashion Landmark Localization (Qian et al., 2021), (c) Semantic Segmentation of Fashion Images (Gong et al., 2018), and (d) Image-based virtual try-on (Neuberger et al., 2020).*

(Gong et al., 2018) achieves a pixel-level classification of fashion item and body parts

on fashion images, assisting a higher level of image understanding. Taking advantage of these research work, virtual try-on (Neuberger et al., 2020) is a downstream task that allows people to virtually try-on new clothing from the internet that shows great potentials on commercial usage. In addition to virtual try-on, the research work of intelligent fashion can also provide auxiliary information, benefiting other downstream applications like fashion recommendation (Dubey, 2021; Hou et al., 2019; Yang et al., 2021; Zhan et al., 2021) or fashion recognition (P. Li et al., 2019; Su et al., 2020; Zhang et al., 2019).

Other than the above-mentioned applications, another key area where AI proves invaluable is in generating and editing design images. By automating repetitive tasks, AI not only reduces costs but also accelerates the creation of new designs, a process that traditionally takes designers extensive time and effort to accomplish. This rapid generation of diverse design drawings by AI, which would be impossible for human designers in comparable time, is particularly crucial in meeting the dynamic demands of the fashion industry.

The diverse generative capability of AI effectively enhances the design process, especially in the creation and updating of prototype images. Traditionally, designing and creating these prototypes has been a complex, expensive, and labor-intensive task, primarily due to the time-consuming process of transforming initial drafts into detailed

design drawings. Fashion designers have historically relied heavily on the expertise to bring ideas to life, with close attention to materials, colors, silhouettes, patterns, and construction techniques. With the availability and accessibility of ample digital resources of fashion images online, e.g. via e-commerce platforms and trend research repository, the process of design has undergone a significant change. Designers nowadays can conduct extensive design research more efficiently through these online resources to come up with new design ideas.

Given the challenges in efficiently creating design prototypes and the need for collecting extensive image data in intelligent fashion applications, there emerges a demand for an advanced system capable of generating and editing high-quality fashion images. Therefore, this study develops an intelligent fashion image processing system that could efficiently generate and edit fashion images, thereby addressing the needs of both the design and intelligent fashion domains.

To develop such a system, visual generative models like generative adversarial networks (GANs) and diffusion models are adapted. GANs train a generator model to convert random noise into a real image, with a discriminator model learning a distance metric for distributions (Goodfellow et al., 2014). Instead, diffusion models diffuse an image to Gaussian noise and then learn to reverse this process to generate an image (Ho et al., 2020). However, both GANs and diffusion models primarily map noise to images

unconditionally, limiting their controllability over the synthesized results.

To tackle this challenge in controllability, several researchers proposed incorporating more informative conditions into the synthesis process. For instance, Chen et al. (2016) introduced Info-GAN, which uses category information to control the process. Isola et al. (2017) developed a method for image translation, treating the synthesis process as a translation from an existing image. Nichol et al. (2022) proposed GLIDE, a UNet-like structure for posterior probability estimation in the denoising process, to incorporate text conditions so as to control synthesis directions. Furthermore, Rombach et al. (2022) investigated LDM model synthesizing high-resolution images with reasonable semantics using a Variational Autoencoder (VAE) to compress images into latent space and applying diffusion models to learn denoising in the latent space.

Nevertheless, the above-mentioned works rely on labeled datasets for controllable generation, collecting category information, semantic segmentation maps, and text-image pairs, thus limiting their application to manually labeled datasets. Moreover, these methods typically use a single data modality, such as texts or images, to control the generation. Considering the variations in generating high-quality images that capture the essence of the desired design elements, their approaches are inflexible.

Therefore, to overcome the above-discussed challenges of controllability and flexibility,

this study proposes a two-stage framework that utilizes decoupled conditions for generating or editing fashion images without intensive manual labeling.

1.2 Statements of the Problem

There are three main challenges in designing and developing an effective generation and editing system for fashion images:

1. The existing state of the art generation method primarily achieve high-fidelity results through text input. However, in the fashion domain, many design elements cannot be adequately described by natural language. The challenge lies in enabling the existing methods to incorporate style conditions as input while maintaining their original generation capabilities.
2. For flexibly image shape editing, this thesis plan to use sketch map as a modification reference. A key challenge in fashion editing is managing significant changes over a large area. As the editing area increases, how could the model synthesis an image that reflects the shape of the sketch map while generating a texture consistent with the original image.
3. The training of current gradient-based model algorithms typically requires supervisory information. Collecting data labels, such as pairs of style conditions and corresponding images or the pairs of sketch maps, input images, and edited images, can be both time-consuming and expensive. Developing a training scheme that could effectively utilize the existing datasets presents a beneficial yet

challenging task.

1.3 Research Aim and Objectives

This study aims to develop a system that can controllably generate fashion images and flexibly edit their shapes. In the generation stage, the user could employ textual description to control the design elements such as the cloth category and detailed attribute, and utilize texture image as style conditions to simultaneously control the texture of the results. In the editing stage, users could modify either a previously generated image or a real image. By providing a rough mask map to determine the editing region and a sketch map as condition for the target clothing shape, the proposed system can effectively generate the edited results. The specific objectives of this study are as follows:

I. To comprehensively review the techniques for generating and editing images using generative adversarial networks and denoising diffusion probabilistic models.

II. To fine-tune the existing text-to-image diffusion model in a parameter-efficient manner, allowing it to accept style conditions for controlling the synthesized cloth textures.

III. To design a sketch-guided large-region editing pipeline to improve the performance of editing fashion images.

IV. To discuss a unified model that integrates the aforementioned design concepts

into a complete system to achieve robust performance across multiple datasets.

1.4 Methodology Overview

The key motivation behind this work is to view the generation and editing process as an image reconstruction process. To effectively utilize the existing datasets, this study has formulated a reconstruction strategy based on the decoupled conditions, which are obtained through several automatic processes, avoiding the need for manual labeling. A two-stage system is proposed in the current study, involving a generation model and an editing model.

In the generation model, a foreground segmentation network (Qin et al., 2019) is used to determine the foreground region, from which it then randomly crop an image patch to obtain style conditions c_{style}. The textual description c_{text}, can be synthesized by BLIP model (Li et al., 2022), an image captioning tool. During the training phase, the generation model $\mathbb{G}_\theta$ is designed to reconstruct the original image I_g using conditions c_{style} and c_{text} simultaneously. Through this reconstruction scheme, the $\mathbb{G}_\theta$ learns a decoupled representation for c_{style} and c_{text}. This process is described as below:

$$I_g = \mathbb{G}_\theta(c_{text}, c_{style})\,. \tag{1-1}$$

Since the generation requires the generated samples to be diverse, this paper adopts a diffusion model-based structure to implement $\mathbb{G}_\theta$. The diffusion models are superior in synthesizing data with high diversity that reflects the nature of a distribution. This character is illustrated in Section 2.3.3.

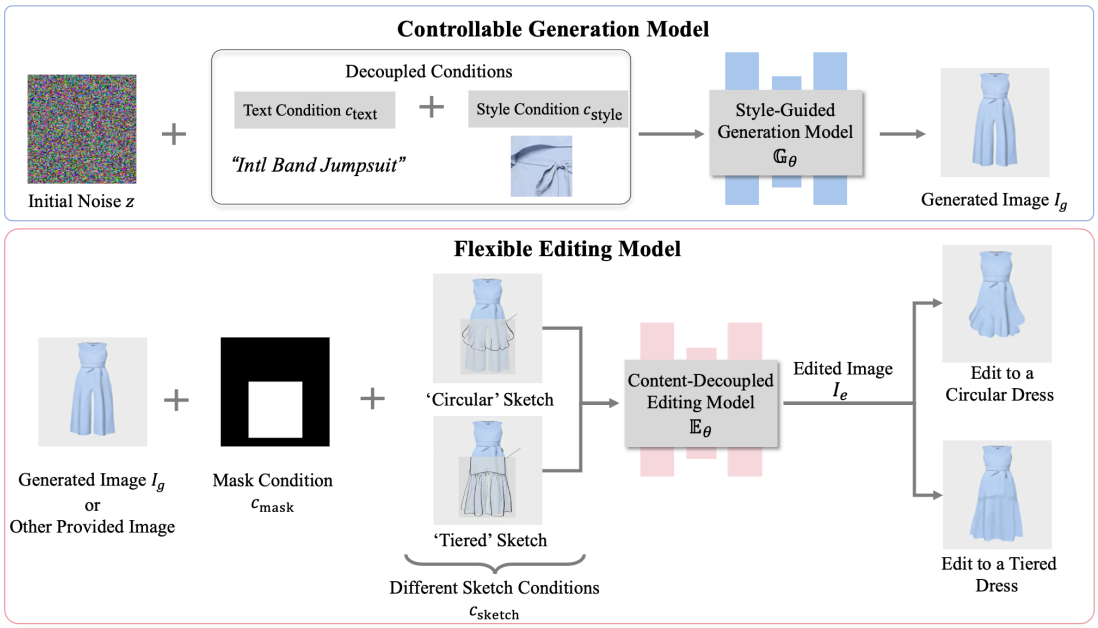


Figure 1-2 The Overview of the Proposed Fashion Image Processing System in Generation and Editing.

In the editing model, denoted as $\mathbb{E}_\theta$, a network is trained in a reconstruction manner as well. Given an image I_e, its sketch map c_{sketch} can be obtained through edge detection (Xie & Tu, 2015). The image I_e is randomly masked with a given c_{mask}, and $\mathbb{E}_\theta$ is used to reconstruct $\hat{I}_e$ from the valid parts, using c_{mask}, and c_{sketch} as conditions. This process is illustrated as:

$$\hat{I}_e = \mathbb{E}_\theta(I_e, c_{sketch}, c_{mask}) . \tag{1-2}$$

When editing a fashion garment, it may involve iterative modification until the result satisfies the user. Therefore, this editing model $\mathbb{E}_\theta$ is achieved by a generative adversarial network instead of a diffusion model. This is because the diffusion model generates an image through a progressive generation pipeline that reverses a random noise to real data. It takes a lot of time to synthesize an image, which is not suitable for this flexible editing task.

After completing the reconstruction training as described in Equation (1-1) and (1-2), the models $\mathbb{G}_\theta$ and $\mathbb{E}_\theta$ are capable of independently conducting generation and editing tasks, respectively. Figure 1-2 illustrates the overall pipeline of the proposed method in the inference stage. The overall system has learned decoupled representations for c_{text}, c_{style}, c_{mask}, and c_{sketch}, allowing users to replace these conditions with their own to represent different design elements. As a result, the proposed system can flexibly generate or edit an image. The detailed model design and its effectiveness examination will be discussed in Chapter 3 and Chapter 4, respectively.

1.5 Organization of the Thesis

This thesis is organized as follows. Chapter 2 reviews the fundamental development of deep learning methods. Section 2.1 provides the essential background on neural networks and their key mechanisms. Section 2.2 provides a basic definition of reinforcement learning and explains its fundamental working prototype. Section 2.3 illustrates the scenario of generative models. Section 2.3.2 covers the development of generative adversarial networks, focusing on the improvements in stabilizing its training process and image quality. Section 2.3.3 reviews the recent advances in diffusion models capable of synthesizing high-fidelity images from textual descriptions.

Chapter 3 introduces the generation model, called SGDiffs, for generating cloth images

that reflect user-provided text and style conditions. Section 3.1 introduces the model overall, while Section 3.2 reviews related works that in simultaneously utilizing text and image as inputs. The proposed methods are presented in Section 3.3. Section 3.4 details the experimental setup in the generation model and discusses both qualitative and quantitative results.

Chapter 4 describes the editing model, CoDE-GAN. Section 4.1 proposes an overview of this framework. Section 4.2 reviews the related works in sketch-controlled editing methods. Section 4.3 explains the detailed architecture of the model. Section 4.4 discusses the data collection and experimental results.

Finally, Chapter 5 concludes the current research findings and suggests directions for future work.

Chapter 2. LITERATURE REVIEW

This study considers the general image editing task as a data-driven image generative task. The data-driven refers that the whole pipeline requires to learn from existing data. Since this research domain mainly adopts deep learning methods, section 2.1 illustrates how the deep learning methods acquire intelligence from the data. For the generative task, section 2.3 illustrates how to project a sample from source distribution to target distribution. Section 2.3.2 introduces the generative adversarial networks that utilize deep learning blocks to generate new data. Section 2.3.3 reviews the denoising diffusion probabilistic models and how it achieves controllable generation.

2.1 Deep Learning Method

2.1.1 An Overview Development of Deep Learning

Deep learning methods originated in 1943 as the neural network model (Fitch, 1944), which was known as multi-perceptron at that time. However, in 1969, Minsky and Papert (2017) proved that neural networks could not handle XOR problems. While also limited by the computer processor's performance at that time, the development of neural networks stagnated for a considerable period. It was not until Rumelhart et al. (1986) proposed backpropagation optimization algorithm, which allowed neural networks to solve the XOR problems by stacking fully connected layers and nonlinear activation functions. From then on, neural networks could be called *deep learning* as well. Deep learning methods really came into the limelight from the ImageNet

challenge, where Krizhevsky et al. (2012) improved the neural networks with convolutional layers. Their proposed AlexNet outperformed than any other machine learning algorithms on the image classification task. Thus, deep learning methods began to be widely used in various AI tasks. This led to extensive research on deep learning for various artificial intelligence tasks.

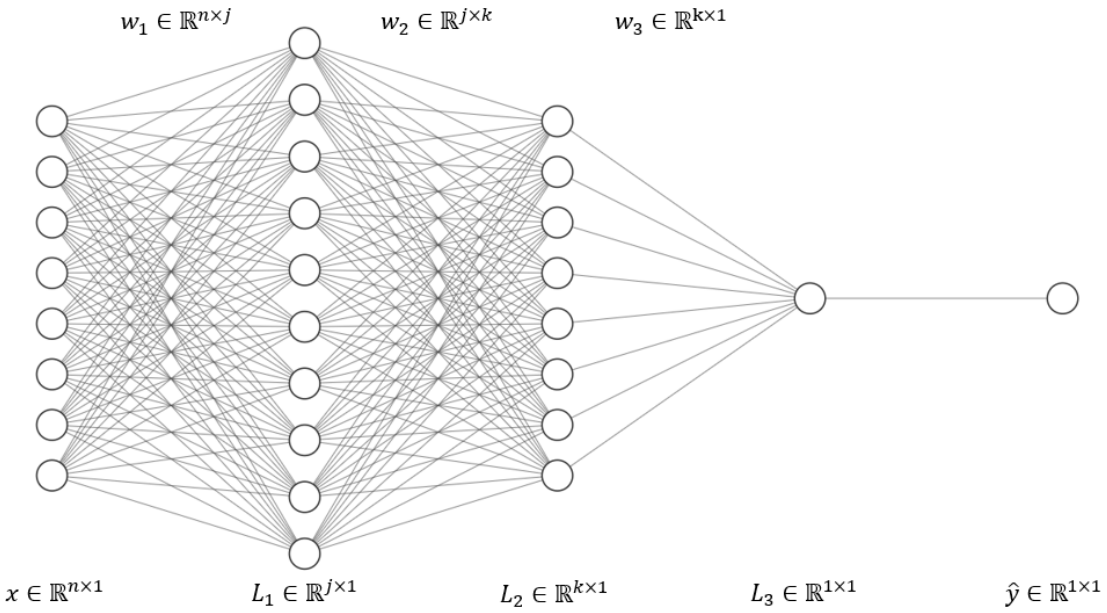


Figure 2-1 A Multi-Layer Perceptron.

2.1.2 Fully Connected Neural Networks

A fully connected neural network can also be called a multilayer perceptron. Figure 2-1 shows a multilayer perceptron structure with single input and output layers and three hidden layers. This thesis denotes input $X \in \mathbb{R}^{n\times 1}$ which is a vector, output $\hat{y}$ which is a scalar. $L_i \in \mathbb{R}^{m\times 1}$ denote the ith layer with m nodes. For any given two layers $L_{i-1} \in \mathbb{R}^{m\times 1}$ and $L_i \in \mathbb{R}^{n\times 1}$, the nodes between the two layers are connected two by two. Therefore, there are $m \times n$ edges with weights to represent their connections. The process of connecting nodes between two layers through weighted edges can be seen

as a weighted summation.

2.1.2.1 The Feedforward Process

This *fully-connection* process is regarded as *feedforward* process. Denote $w_i \in R^{m\times n}$ as the weight matrix of the edges between layer $L_{i-1} \in \mathbb{R}^{m\times 1}$ and $L_i \in \mathbb{R}^{n\times 1}$, the feedforward process is represented as equations:

$$L_i = w_i^T L_{i-1}. \tag{2-1}$$

Hornik (1991)'s work reveals a universal approximation theorem for neural networks that a neural network with more than one hidden layer, if coupled with a nonlinear activation function, can fit any function with arbitrary accuracy through a finite number of nodes. One of the most used non-linear activation functions is sigmoid which is defined as follows:

$$\sigma(x) = \frac{1}{1+e^{-x}}. \tag{2-2}$$

The Sigmoid function maps the input x to a number with a value range between $[0, 1]$. This property allows the output of the network to be used as a probability. At the same time, Sigmoid possesses a good derivative property that facilitates the subsequent optimal solution of:

$$\sigma(x)' = \sigma(x)\big(1-\sigma(x)\big). \tag{2-3}$$

Therefore, the Equation (2-1) could be re-formulated as:

$$\begin{matrix} L_i = \sigma(w_i^T L_{i-1}) \\ y_i = \sigma(L_i) \end{matrix}. \tag{2-4}$$

In Figure 2-1, there are three hidden layers $L_1 \in \mathbb{R}^{j\times 1}$, $L_2 \in \mathbb{R}^{k\times 1}$, and $L_3 \in \mathbb{R}^{l\times 1}$, with corresponding weights matrix: $w_1 \in \mathbb{R}^{n\times j}$, $w_2 \in \mathbb{R}^{j\times k}$, and $w_3 \in \mathbb{R}^{k\times l}$. The feed-forward process of this neural network can be defined as:

$$\begin{cases} L_1 = w_1^T X \\ y_1 = \sigma(L_1) \\ L_2 = w_2^T y_1 \\ y_2 = \sigma(L_2) \\ L_3 = w_3^T y_2 \\ \hat{y} = y_3 = \sigma(L_3) \end{cases} \tag{2-5}$$

2.1.2.2 Commonly Used Loss Functions

After defining the model, certain criteria are needed to measure the goodness of the model in order to have an optimization direction. The criteria also called as loss function that usually is a distance measurement. It measures the difference between the model output and the true label. And according to the regression and classification problems, there are usually two types of loss functions as follows:

1) L_1 or L_2 Distance:

   The definition of L_1 or L_2 distance comes from L_p Norm. When the norm number p is taken as 1 or 2, the L_1 or L_2 is the p norm of vector x. The vector x is usually obtained by making a difference between the output $\hat{y}$ predicted by the model and the true value y. For the L_1 loss function, the extreme value of the optimization can be reached only when each component of x is close to 0. Therefore, the L_1 loss function is chosen as the optimization objective to obtain a sparser solution. And L_2 loss function, as the most used loss function, can measure the Euclidean distance between two vectors, because it does the square operation for each component of x, which is more sensitive to outliers.

$$L_p = ||x||_p = \sqrt[p]{\sum_{i=1}^{n} |x_i|^p} \tag{2-6}$$

2) Cross Entropy:

The definition of the cross-entropy loss function comes from the KL Divergence. In machine learning, practitioners often need to measure the difference between two distributions $p(x)$ and $q(x)$ for the same random variable. $p(x)$ represents the true distribution of the sample and $q(x)$ represents the predicted distribution of the sample. This is when the KL divergence (also known as relative entropy) is measured, and the smaller the value of KL divergence, the closer the two distributions are:

$$\begin{aligned} D_{KL}(p \parallel q) &= \sum_{i=1}^{n} p(x_i)\log\left(\frac{p(x_i)}{q(x_i)}\right) \\ &= \sum_{i} p(x_i)\log\big(p(x_i)\big) - \sum_{i} p(x_i)\log\big(q(x_i)\big) \end{aligned} \tag{2-7}$$

The $p(x)$ involved in the first term, which is the true value, is used as a constant when the network is trained. While the second term contains the prediction of the network, which is the cross-entropy:

$$\text{CE } = y = -\sum_{i} p(x_i)\log\big(q(x_i)\big) \tag{2-8}$$

2.1.2.3 Error Backpropagation Algorithm

From the feedforward process of Equation (2-5), the nature of the neural network is a composite function with parameters $w_1,\ w_2,\ w_3$ to be optimized. This parameterized function can be optimized by the error backward propagation algorithm. For input X, the predicted output $\hat{y}$ can be obtained by forward propagation of Equation (2-5). If the true output corresponding to input X is defined as y, and there is a difference J between $\hat{y}$ and y. The error back propagation is the process of propagating the output error

through the nodes of the network to the input nodes by layers. The errors will be distributed to different edges according to their gradients. By applying the error signals to correct the weights, the w is iteratively updated in one round of learning species until the error is reduced to an acceptable level or the number of iterations reaches an upper limit. For simplicity, this paper illustrates the backpropagation process by taking L_2 loss function as an example. There are:

$$J = \frac{1}{2}||\hat{y} - y||_2^2 \tag{2-9}$$

The optimization objectives are reducing the difference between the predicted output of the neural network and the true output as small as possible. So, the process of optimizing the network parameters $w_1,\ w_2,\ w_3$ is essentially solving an optimization problem:

$$\min_w J = \frac{1}{2}||\hat{y} - y||_2^2 \tag{2-10}$$

The essence of the error backpropagation algorithm is solving this optimization problem by gradient descent. Given a constant η as the learning rate, there is an update formula for the neural network parameters as:

$$\begin{aligned} \Delta w_i &= -\eta \frac{\partial J}{\partial w_i} \\ w_i &= w_i + \Delta w_i \end{aligned} \tag{2-11}$$

In the other words, the parameters of the neural network can be updated iteratively by the chaining law to find the partial derivatives for each w_i in 2-11. Here it firstly calculates the partial derivative for each layer of L_i:

$$\begin{aligned} \frac{\partial J}{\partial L_3} &= \frac{\partial J}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial L_3} \\ &= (\hat{y} - y) \cdot \hat{y} \cdot (1 - \hat{y}) \end{aligned} \tag{2-12}$$

$$\frac{\partial J}{\partial L_2} = \frac{\partial J}{\partial L_3} \cdot \frac{\partial L_3}{\partial L_2} = \frac{\partial J}{\partial L_3} \cdot w_3 \cdot y_2 \cdot (1 - y_2) \quad (2\text{-}13)$$

$$\frac{\partial J}{\partial L_1} = \frac{\partial J}{\partial L_2} \cdot \frac{\partial L_2}{\partial L_1} = w_2 \frac{\partial J}{\partial L_2} \cdot y_1 \cdot (1 - y_1) \quad (2\text{-}14)$$

The corresponding partial derivatives of each w_i is:

$$\frac{\partial J}{\partial w_3} = \frac{\partial J}{\partial L_3} \cdot \frac{\partial L_3}{\partial w_3} = y_2 \cdot \frac{\partial J}{\partial L_3} \quad (2\text{-}15)$$

$$\frac{\partial J}{\partial w_2} = \frac{\partial J}{\partial L_2} \cdot \frac{\partial L_2}{\partial w_2} = y_1 \frac{\partial L_2}{\partial w_2} \quad (2\text{-}16)$$

$$\frac{\partial J}{\partial w_1} = \frac{\partial J}{\partial L_1} \cdot \frac{\partial L_1}{\partial w_1} = X \frac{\partial L_1}{\partial w_1} \quad (2\text{-}17)$$

2.1.2.4 Optimization Methods

Section 2.1.2.3 discussed the basic optimization method, which is the error backpropagation, for solving the neural networks. This section will illustrate more detailed improvements on the optimization.

1) Stochastic Gradient Descent (Bottou, 2012)

The core idea of stochastic gradient descent (SGD) is that for each sample X, the feedforward process computes predicted output $\hat{y}$ once. By applying the loss function for getting the error J, it is possible to calculate the partial derivatives $\frac{\partial J}{\partial w_i}$ for updating w_i. For each sample, the parameters will be updated once. Since the presenting of

samples is stochastic, the updating of gradients is stochastic as well. For a dataset D with M samples, the one round completion of training is the traverses of the entire dataset. The parameters w_i are updated a total of $M \times N$ times. The advantages of this method are that the samples randomly input into the network will carry some noise, which can avoid the overfitting phenomenon to some extent. The network can easily converge to the global optimal point with proper learning rate although the update of the network weights is not stable enough.

2) Batch Gradient Descent (Bottou, 2012)

Performing the SGD way of optimizing w_i will update the parameters $M \times N$ times. The batch gradient descent (BGD), however, goes to an extreme in the opposite direction of SGD that it updates the parameters only once for each epoch. The BGD will accumulate the gradients of all samples and calculate its mean value. For every epoch, the w_i will be updated by the mean gradients once. When the network undergoes a complete BGD training with M epochs, the parameters w_i are updated a total of M times. The number of updates is independent of the size of the dataset and only related to the number of training epochs M.

3) Mini-Batch Gradient Descent

Mini-Batch is the current main method for training networks for deep learning, which combines SGD and BGD in a compromise. A Mini-Batch takes B samples,

which is equivalent to dividing a sample set of number N into $\frac{N}{B}$. When the mini-batch optimization takes total sample amount M of the dataset as batch size, it degrades to batch gradient descent. When the batch size takes 1, it becomes the stochastic gradient descent.

4) Momentum Updating (Fan et al., 2016)

As shown in Figure 2-2, stochastic gradient descent updating for optimizing the networks is not smooth enough. This is because the random input samples carry a certain amount of noise, which can be considered as causing some bias to the training of the network. However, it could be mitigated to some extent by adding a momentum term to the formula for the parameter update.

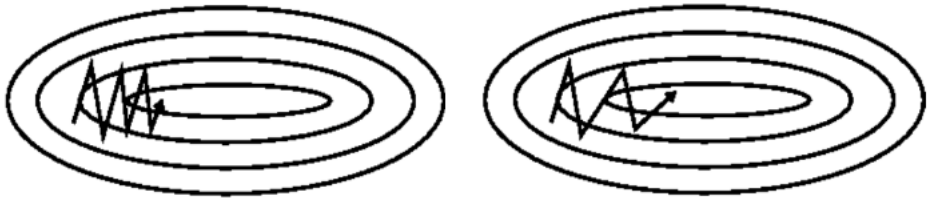


Figure 2-2 Effectiveness of Applying the Momentum Term (Ruder, 2016).

The process of updating parameters by SGD can be oscillating during optimization, which can slow down the learning process. Adding the momentum term effectively relief the oscillation that allows to a more stable descent toward the optimization goal. This is because the momentum method simulates the second-order gradient in an inexpensive way. When the optimization falls into a saddle point, it is still more likely to leave the flat because of momentum. The update equation with the momentum term is shown below:

$$v_t = \gamma v_{t-1} + \eta \frac{\partial J}{\partial w} \tag{2-18}$$

$$w = w - v_t \,, \tag{2-19}$$

where v represents the momentum term, the subscript t indicates the number of training rounds, and γ is a constant that is the coefficient of the momentum term. The range of γ usually is taken as a number less than 1. Therefore, the weight of the previous momentum term can be decayed by iteration. When t takes 0, v_0 is initialized to 0. The momentum term approach allows the network to have some *memory* when the parameters are updated. So that the parameters are not updated in the current batch in a direction completely different from the previous updates. Thus, globally, the direction of the parameter updates can be more homogeneous, allowing the loss value to decrease faster towards the optimal point.

2.1.3 Convolutional Based Neural Networks

Convolutional based neural networks (also known as CNN) have become the main network architecture among the deep learning methods. Different from the fully connected networks, CNN adopts convolutional kernel to capture features from input data. This section will introduce the mechanism of convolution and the main convolutional based neural networks.

2.1.3.1 Convolutional Kernel

The fully connected neural network described in the previous section has a major drawback in that the dense connection between nodes requires a large number of

parameters. Although it has been shown to have the ability to *fit arbitrary functions*, this requires that the network be wide enough and deep enough. A wide and deep network can lead to a significant increase in the number of parameters, making the optimization of the network difficult. At the same time, due to the lack development of CPU's computing power, it has led to the fact that neural networks have not been able to perform as well as they should.

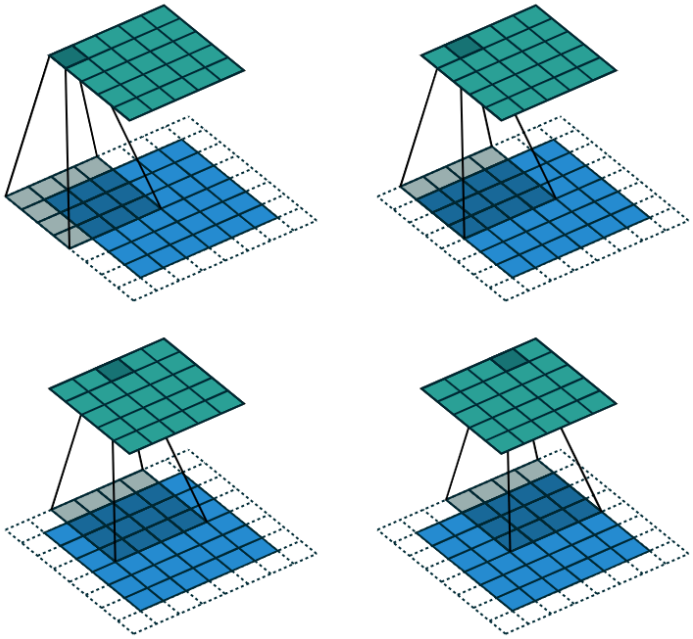


Figure 2-3 Schematic Diagram of a 2D Convolution of a Single Channel (Dumoulin & Visin, 2016).

kernels. The light gray 4 × 4 region represents the size of the convolution kernel, which slides over the blue 6 × 6 region. The convolutional kernel multiplies and sums the elements at the corresponding positions with the grey region and obtains an element in the top green region. If the blue region is taken as the image of the input network, then the green region is the feature map obtained after the input image has been convolved

and computed. There is a dashed transparent box around the blue area, indicating 0 padding, which can be used to obtain different sizes of output feature maps with different sizes of convolution kernels and different sizes of sliding window steps.

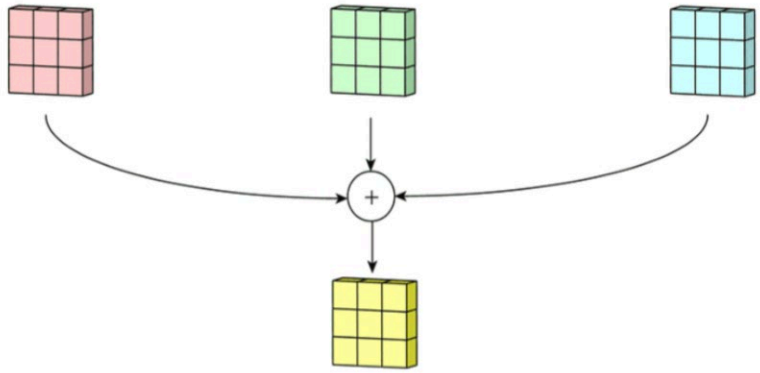


Figure 2-4 Schematic for Multi-channel convolution (Dumoulin & Visin, 2016).

As shown in Figure 2-4, when the input image is multi-channel, the convolution is slightly different from that of single channel. For a single-channel convolution kernel, its size can be $3 \times 3 \times 1$ that each dimension indicates the length, width, and channel of this convolution kernel. For a three-channel input image (e.g., an image in RGB format), the size of the convolution kernel with the same length and width should be $3 \times 3 \times 3$, indicating that there are three channels. These three channels are first filtered by sliding windows on the channels on the corresponding input images separately, and finally superimposed together in the form of summation. No matter the input data is three-channel or single-channel, the output is one channel after the operation of one convolution kernel.

Convolution is computed as a local fully connected neural network, where they share

weights over some regions. The convolution kernel performs a sliding window on the feature map, weighting the pixels in the corresponding region on the feature map with the convolution kernel to sum. The number of sliding windows determines the size of the output feature map. Defining p as the number of turns of the input image for 0-paddings, k as the size of the convolution kernel of $k \times k$, s as the step size of the convolution kernel when it performs a sliding window, and the size of the input image of $w_{in} \times w_{in}$, then there is the size of the output image $w_{out} \times w_{out}$:

$$w_{out} = \frac{w_{in} - k + 2 * p}{s} + 1. \tag{2-20}$$

When $2 * p + s - k = 0$ is satisfied, there will be $w_{out} = s * w_{in}$. So that the input and output dimensions are the same for network design, which usually makes $s = 1$, when $k = 3$ and $p = 1$. The set of parameters is the structure of a common set of convolutional layers since it brings simplicity for organizing the size of feature maps.

2.1.3.2 Classical Architectures for CNN

Since the AlexNet model proved the superior performance of deep convolutional neural networks in dealing with pattern recognition problems, various improvements to AlexNet have emerged over time. In this section, this thesis gives a brief description of the improved methods of VggNet, GoogleNet, ResNet, and MobileNet in chronological order of development.

1) VggNet

The VggNet (Visual Geometry Group) was proposed by Simonyan and Zisserman (2014). Section 2.1.3.1 illustrates that the convolutional kernel could be considered as

a feature extractor. Following the process where data passes through the convolution layer, resulting in a feature map, it is observed that since convolution is locally connected, each pixel in the feature map contains local information of the image. To enhance this aspect, the field then introduces the concept of Receptive Field, aimed at increasing the amount of extracted local information from the feature map. Receptive field is shown as the grey region in the Figure 2-3. Receptive field refers to each pixel in the feature map that output by the network after mapping back to the original image. Since convolution is a locally connected operation, each pixel on the feature map corresponds to a part of the input image. The size of the region can bring a large impact on the network's ability to extract features. It is generally believed that when the perceptual field is large enough, the pixel points on the feature map possess relatively more local information. If the size of the receptive field is the same as the size of the input image, it can even be considered that the pixels at that point incorporate the global information of the input image.

2.2 Reinforcement Learning

2.2.1 Sequential Decision-Making Tasks

Section 2.1 introduced the basics of neural networks that are mainly used to solve recognition problems. The recognition problem mainly classify or detection of certain information. This task only generates a signal for the input data and expects it to be consistent with the observable signal in the future without changing the future situation.

However, in the field of machine learning, there is an important type of task similar to the human decision-making process, that is, sequential decision-making tasks. Different from the recognition tasks, decision tasks usually bring *consequences*. Therefore, the decision-maker needs to be responsible for the future and make further decisions at future time (Sutton & Barto, 2018).

To address this task, reinforcement learning is introduced as a computational method for a machine to achieve goals through interaction with the environment. One round of interaction between the machine and the environment includes: the machine makes an action decision in a state of the environment, applies this action to the environment, and the environment changes accordingly and feeds back the reward and the next state back to the machine. This process is iterative, and the goal of the machine is to maximize the expected cumulative reward during multiple rounds of interaction. The above-mentioned process is implemented by an *agent* (Mnih et al., 2015). The agent is quite different from the so-called *model* in supervised learning. The agent not only perceives environmental information but also directly changes the environment through decision-making, rather than just giving a prediction signal.

2.2.2 Agents Formulation and Objectives

The agent of reinforcement learning completes sequential decision-making through interaction with the dynamic environment. The dynamic means that the environment will continuously evolve as certain factors change, which is usually described by a stochastic process in mathematics and physics. If the agent's actions are added as an

external disturbance factor in this stochastic process, the probability distribution of the next state of the environment will be jointly determined by the current state and the agent's actions. This process is expressed as:

$$s_{t+1} = f(s_t, a_t), \tag{2-21}$$

where s_t represents the current state, a_t represents the action taken by the agent in the current state, and s_{t+1} represents the next state, and f represents the state transfer function (Kober et al., 2013).

From Equation 2-21, the actions of the agent act on the environment will cause the state of the environment to change. And then the agent continues to make decisions in the new state. In the above dynamic environment, every time the agent interacts with the environment, the environment will generate a reward signal, which is usually represented by a real scalar. This reward signal is similar to the score in a game, indicating the goodness or badness of the current state or action. The reward signal of each round of interaction is accumulated to form the overall return of the agent, which is similar to the final score of a game. Due to the dynamics of the environment, even if the initial state and strategy remain unchanged, the interaction result may be different, and the return will also be different. Therefore, reinforcement learning focuses on the expectation of return and defines it as value, which is the optimization goal of the agent's learning process (Schulman et al., 2017).

This agent could be obtained in a supervised manner that the goal is to find an optimal model to minimize a given loss function on the training data set. Under the

independent and identically distributed assumption, this goal represents minimizing the generalization error of the model over the entire data distribution. Briefly expressed by the formula as:

$$\theta^* = \arg \min_{\theta} \mathbb{E}_{(x,y)\sim\mathcal{D}}[\ell(f_\theta(x), y)], \tag{2-22}$$

where θ represents the model parameters, (x, y) represents the input and corresponding label, $\mathcal{D}$ represents the data distribution, and ℓ represents the loss function (Sutton & Barto, 2018).

In contrast, the ultimate optimization goal of the reinforcement learning task is to maximize the value of the agent's strategy during the interaction with the dynamic environment. The value of the strategy can be equivalently transformed into the expectation of the reward function on the measure of the strategy occupancy, that is:

$$\pi^* = \arg \max_{\pi} \mathbb{E}_{s,a\sim\pi}[R(s, a)], \tag{2-23}$$

where π represents the agent's strategy, s and a respectively represent the state and action, and R represents the reward function (Mnih et al., 2015).

Therefore, compared with general supervised learning models, reinforcement learning focuses on finding an agent strategy to generate the optimal data distribution during the interaction with the dynamic environment, thereby maximizing the expectation of the reward function.

2.3 Generative Models

2.3.1 Generative Tasks

Most of the deep learning tasks could be demonstrated as the searching of a projection that projects input probabilistic distribution to another on. Most of the deep learning tasks could be demonstrated as the searching of a projection that projects input probabilistic distribution to another distribution (Goodfellow et al., 2014). For classification tasks, the model samples a picture or a length of text from real data distribution and output with a probabilistic distribution of class. The probabilistic distribution of class usually represented by a one-hot coded vector. In addition, tasks like segmentation could be considered as a pixel-wise classification task. For regression tasks, the model outputs a continuous distribution.

In the early stage, generative tasks take flatten noise vector as input which differs from the above-mentioned types of input (Creswell et al., 2018). The noise vector is commonly sampled from a simple distribution such as a normal distribution. And it will output a distribution of images or other types of data. The model that projects the input distribution to a complex distribution is called generator.

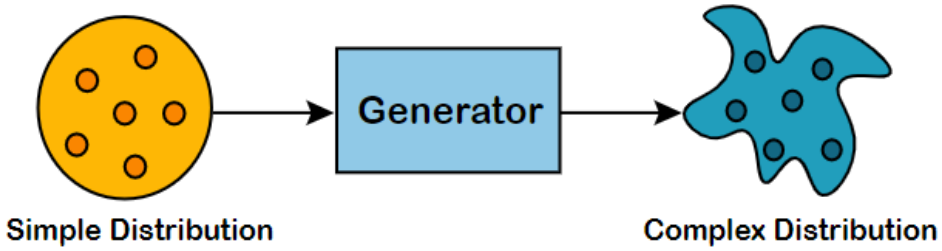


Figure 2-5 Projection from Simple to Complex Distribution.

However, the noises as input are out of control which limits its potential availability. In the later study, researchers suggested conditional generative tasks (Radford et al., 2015) which have variable types of input. Moreover, some researchers modeled the segmentation tasks as generative task. Since the segmentation results could be treated as a special data distribution (Isola et al., 2017). One fact should be pointed is that generative tasks mostly require semi- or unsupervised learning. Since it could be expensive for reaching data pairs like style transferring, generative tasks generally lack enough paired data for optimization. Therefore, a semi- or unsupervised learning methods benefit the generative task well.

2.3.2 Generative Adversarial Networks

2.3.2.1 Overview of GAN

Generative Adversarial Networks (known as GAN) was firstly introduced by Goodfellow et al. (2014). GAN was designed to perform generative tasks through a deep learning-based method that optimizing through gradients descent. Unlike the classic end-to-end deep learning networks, there are two networks. The Generator network produces images and Discriminator determines whether the image comes from the generator or real data distribution. Therefore, the training of this model is adversarial. Shown as Figure 2-5, the primer GAN takes random noises as input.

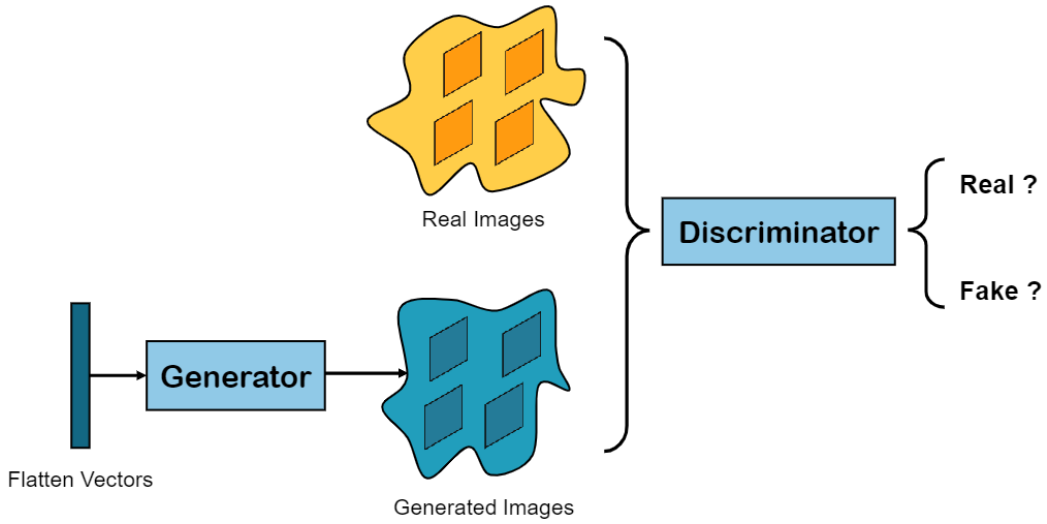


Figure 2-6 The Prototype of GAN.

Considering $G(\cdot, \theta_g)$ as generator and $D(\cdot, \theta_d)$ as discriminator, θ_g and θ_d is their parameters correspondingly. The optimization goal could be described as follows:

$$\min_G \max_D V(G,D) = \mathbb{E}_{x\sim\mathbb{P}_d(x)}\left[log\left(D(x)\right)\right] + \mathbb{E}_{z\sim\mathbb{P}_z(z)}\left[log\left(1 - D\left(G(z)\right)\right)\right], \quad (2\text{-}24)$$

where x sampled from the real images and z sampled from a random noise distribution.

The intuitive understanding of Equation (2-21) is that the discriminator should be considered as a parameterized loss function for the generator. When the parameters of discriminator are fixed, it is more like a binary classifier that classify whether the image is real or fake. Therefore, the optimization goal of discriminator is to maximize the term $log(D(x))$ and $log(1 - D(G(z)))$ firstly. For the generator, it is expected that it can produce images that are as realistic as possible to fool the discriminator. In Algorithm 1, the second term $log(1 - D(G(z)))$ is thereafter optimized for the generator, as the first term becomes a constant when the discriminator is fixed.

2.3.2.2 Optimality of GAN

Goodfellow et al. (2014) proved the optimality of discriminator by expanding the expectation form of Equation (2-21).

$$\begin{aligned} V(G,D) &= \int_x p_d(x)\log\left(D(x)\right)dx + \int_z p_z(z)\log\left(1-D(G(z))\right)dz \\ &= \int_x p_d(x)\log\left(D(x)\right) + p_g(x)\log\left(1-D(x)\right)dx \end{aligned} . \quad (2\text{-}25)$$

Let $\frac{\partial V}{\partial D} = 0$, the optimal discriminator D is:

$$D_G^*(x) = \frac{p_d(x)}{p_d(x)+p_g(x)}. \quad (2\text{-}26)$$

Replace discriminator term D in Equation (2-21) by Equation (2-23), the optimizing of generator could be:

$$\begin{aligned} C(G) &= \max_D V(G,D) \\ &= \mathbb{E}_{x\sim p_d}\left[\log\frac{p_d(x)}{p_d(x)+p_g(x)}\right] + \mathbb{E}_{x\sim p_g}\left[\log\left(1-\frac{p_d(x)}{p_d(x)+p_g(x)}\right)\right]. \\ &= \mathbb{E}_{x\sim p_d}\left[\log\frac{p_d(x)}{p_d(x)+p_g(x)}\right] + \mathbb{E}_{x\sim p_g}\left[\log\frac{p_g(x)}{p_d(x)+p_g(x)}\right] \end{aligned} \quad (2\text{-}27)$$

Here introduces Kullback-Leibler Divergence (KLD) which is widely used in optimizing classification tasks:

$$D_{KL}(P||Q) = \mathbb{E}_{x\sim p_X} log\frac{P(x)}{Q(x)}. \quad (2\text{-}28)$$

Therefore, replace the term in Equation (2-25) by KLD, the min-max game of equation is actually the optimization of Jensen-Shannon Divergence (JSD):

$$
\begin{aligned}
C(G) &= D_{KL}\left(p_d \parallel p_d + p_g\right) + D_{KL}\left(p_g \parallel p_d + p_g\right) \\
&= -\log(4) + D_{KL}\left(p_d \parallel \frac{p_d + p_g}{2}\right) + D_{KL}\left(p_g \parallel \frac{p_d + p_g}{2}\right). \\
&= -\log(4) + 2 \cdot D_{JS}\left(p_d \parallel p_g\right)
\end{aligned}
\tag{2-29}
$$

The range of JSD is 0 to 1. When the two probability distributions are totally similar, it reaches its minimum as 0. So, the minimal loss value for the generator is $-\log(4)$. If the discriminator and generator achieved Nash Equilibrium, their loss value will converge to oscillate around $-\log(4)$.

2.3.2.3 Improvement on Training Stability

The early exploration doubted the training stability of GAN since researchers found that it unstable and hard to optimize GAN. One of the possible reasons is that GAN is an unsupervised method. When there is a need to generate something, this indicates that the output lacks sufficient and direct target samples for learning. Therefore, GAN provides an adversarial way of trying to use less paired input-output data. From a supervised point of view, discriminator is more like a loss function for generator to learn.

The optimization of generator amounts to the optimization of the second term in Equation (2-21). The most informative gradients come from the sample that confused the discriminator. However, it required the discriminator to have the ability to classify samples in a very strict extend. The discriminator can neither be too strong nor weak. A powerful discriminator leads zero gradient to generator. A weak discriminator leads

to feeble performance of the generator.

When the networks convergent, it reaches the Nash Equilibrium. But this did not indicate that the GAN has enough ability to generate very realistic images. Since the generator can easily find a short way to pass the discriminator. For instance, the generator might just have memory of training data distribution. And there is no guarantee for the discriminator to measure the diversity of the generator.

Table 2-1 Several Distance Metrics of Probability Distribution

Kullback-Leibler	$D_{KL}(P_d \| \| P_g) = \sum_{x \sim \chi} log \frac{P_d(x)}{P_g(x)} P_d(x)$	(2-30)
Jensen-Shannon	$D_{JS}(P_d, P_g) = D_{KL}(P_d \| \| P_g) + D_{KL}(P_g \| \| P_d)$	(2-31)
Wasserstein	$D_W(P_d, P_g) = \inf_{\gamma \in \prod(P_d, P_g)} E_{(x,y) \sim \gamma} \|\|x - y\|\|$	(2-32)

Analyzing from the perspective of optimization, the loss function of Equation (2-21) is actually to minimize the Jensen-Shannon Divergence (JSD). Table 2-1 shows some of the distance metrics for measuring probability distribution. JSD is an improved form for Kullback-Leibler Divergence (KLD) for considering the symmetric of distance. However, when there is no overlap between two distributions, JSD saturates in its maximum one. It lacks a soft way for un-overlap situations.

2.3.2.4 Early-Stage Tricks on Stabilization

As aforementioned, it is hard to optimize GAN because of the intractable estimation the distance between probability distributions. Along with this motivation, here are some of the possible directions for improvements that trying to make the overlap of the distribution.

1) Input Noise

During the early stage of training, the generator probably lacks the ability to output a distribution which overlap with the real data. And JSD is not capable to measure the exact distance when there is no overlap. One possible solution that led to overlap is that if there are enough randomly samples be feed forwarded into the generator, there might be overlap which enhances the training. This could be one explanation of why the vanilla GAN takes noises as input.

2) Soft Output

Soft output aims at leaving more margin on the output of discriminator. The nature of discriminator is a binary classifier. It outputs 1 or 0 for classifying whether the input image is real of fake. But it is too confident that ignore the possible overlapped part of distribution since the generator may have captured partial realistic regions of images.

a) Label Smoothing:

Label smoothing was proposed by Szegedy et al. (2016) in deep learning domain. Instead of forcing the classifier to fit an absolute label of 1, it encourages the classifier to have more margin about its confidence of the prediction. Moreover, this also

indicates that the label for the input images might be unreliable or inaccurate. This indication fits the GAN's situation that an image be classified as fake by discriminator might have partial realistic region. Warde-Farley and Goodfellow (2016) showed that label smoothing may reduce the vulnerability of GAN. In the practices described by Salimans et al. (2016), replacing the positive and negative samples with constants α and β results in the optimal discriminator being as shown below:

$$D_G^*(x) = \frac{\alpha P_d(x) + \beta P_g(x)}{P_d(x) + P_g(x)}. \tag{2-33}$$

They found that if the $P_d(x)$ is close to zero and $P_g(x)$ is much greater, this may lead to numerically unstable. So, it is more effective to set a soft α and keep the negative samples zero.

b) Relativistic GAN:

Another proposal of soft output is to compare a pair of real and fake samples. Considering the layers before the output sigmoid layer as C, the discriminator $D(x) = sigmoid(C(x)$. Jolicoeur-Martineau (2019) proposed a form of loss function:

$$L_D = -\mathbb{E}_{(x_d,x_g)\sim(P_d,P_g)}\left[log\left(sigmoid\left(C(x_d) - C(x_g)\right)\right)\right] \tag{2-34}$$

$$L_G = -\mathbb{E}_{(x_d,x_g)\sim(P_d,P_g)}\left[log\left(sigmoid\left(C(x_g) - C(x_d)\right)\right)\right]. \tag{2-35}$$

The objective is similar to metric learning but at an image level. It forces the discriminator to predict the extent that the real image is more realistic than the generated image. Therefore, it has some capability to capture the distributions distance.

3) Training Strategies of Discriminator

The training of GAN requires the discriminator to convergent to a certain level that

can classify part of the real or fake sample pairs. Only can the fake sample that fool the convergent discriminator provides informative gradients for optimizing the generator. Therefore, fine-tune the training of discriminator benefits the generator.

a) Historical Averaging:

From this motivation, Salimans et al. (2016) proposed an updating method that takes parameters in time-series into consideration. Taking $\theta[i]$ is the parameters both of generator and discriminator in i th time step, there will be a penalty term in the loss function:

$$V(G,D)^* = V(G,D) + \lambda\left|\theta - \frac{1}{t}\sum_{i=1}^{t}\theta[i]\right|^2. \quad (2\text{-}36)$$

Historical averaging actually applied constraints on the space of parameters which in somehow meets the Lipschitz Constraints. There will be more discussion about it in Section 2.3.2.5. But historical averaging requires to keep t times parameters which increases the consumption of GPU memory.

b) Two Timescale Update Rule:

Heusel et al. (2017) addressed two timescale methods for achieving Nash Equilibrium in the min-max game. The discriminator uses a greater learning rate than the generator. Heusul suggested that the learning rate of discriminator is four times greater of generator. Therefore, the discriminator could convergent quicker than the generator. Since the generator only can learn from the discriminator, it enables these two networks to accelerate training process.

4) Feature Match

Salimans et al. (2016) argued the reliability of discriminator. Instead of learning from the discriminator, they expected the generator to learn statics features of real data. Considering fi is the intermediate layer of a discriminator D. The objective of feature match is:

$$L(G) = min\left|\left|\mathbb{E}_{x\sim P_r} f(x) - \mathbb{E}_{x\sim P_z} f\big(G(x)\big)\right|\right|_2^2 . \quad (2\text{-}37)$$

The original type of feature match required the discriminator to be trained with its original objective that classify real or fake images. The later studies on deep fakes (Korshunova et al., 2017) replace the binary classifier discriminator with other network pre-trained by related tasks, e.g.: face recognition. This development showed that if there are enough pre-trained models which related to generated domain, feature match benefits the generator the most.

2.3.2.5 Wasserstein GAN

Although there are numerous improvements on the training tricks, they failed to face the fetal issue of GAN that its optimization goal of JSD lacks capability of measuring two totally separate distributions. Arjovsky et al. (2017) introduced this disadvantage and proposed a novel loss function Wasserstein Distance:

$$W\big(P_d, P_g\big) = \inf_{\gamma\in\prod(P_d,P_g)} \mathbb{E}_{(x,y)\sim\gamma}\big[\left|\left|x-y\right|\right|\big]. \quad (2\text{-}38)$$

$\prod(P_d, P_g)$ is the set of all joint distributions $\gamma(x, y)$ whose marginals are respectively P_d and P_g . However, it is intractable to calculate the infimum in Equation (2-35). According to Kantorovich-Rubinstein duality, the Equation (2-35) could be transferred as:

$$W(P_d, P_g) = \sup_{||f||_L \leq 1} \mathbb{E}_{x \sim P_d}[f(x)] - \mathbb{E}_{x \sim P_g}[f(x)]. \quad (2\text{-}39)$$

This requires the function *f* to be 1-Lipschitz function. In the other words, it requires the discriminator to be Lipschitz Continuous. Hence, the optimization goal for W-GAN is:

$$\min_G \max_D V(G, D) = \mathbb{E}_{x \sim P_d(x)} D(x) - \mathbb{E}_{z \sim P_z(z)} D\big(G(z)\big). \quad (2\text{-}40)$$

If the function *f* is Lipschitz Continuous, there should be a constant L which meets:

$$\big||f(x) - f(y)|\big| \leq L\big||x - y|\big| \Rightarrow \frac{\big||f(x) - f(y)|\big|}{\big||x - y|\big|} \leq L. \quad (2\text{-}41)$$

Intuitively, the Lipschitz Constant *L* constrains the slope of *f*. As the discriminator is bounded, Equation (2-38) is optimizable. From this motivation, researchers applied different measures for meeting the Lipschitz Continuous.

1) Weight Clipping

The first proposal in Arjovsky et al. (2017) is weight clipping. Their primary idea is if the parameters of model is bounded, then the output is bounded. After the weight is updated, this proposal clips the updated weight to [−0.01, 0.01]. However, they also found that momentum-based optimizer like Adam failed. Because the clipping will force the weights' distribution concentrated on ±0.01. Even though weight clipping is computational effective, it is unstable to train.

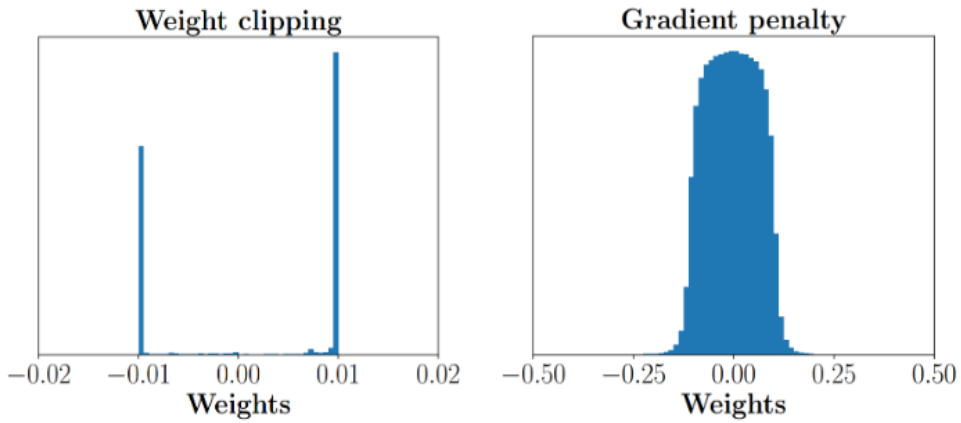


Figure 2-7 Weights Distribution of Weight Clipping and Gradient Penalty (Gulrajani et al., 2017).

2) Gradient Penalty

Gulrajani et al. (2017) thereafter applied constraints on the gradients to meet the Lipschitz Continuity. By introducing gradient penalty term in Equation (2-39):

$$\min_G \max_D V(G,D) = \mathbb{E}_x[D(x)] - \mathbb{E}_z[D(G(z))] + \lambda \mathbb{E}_{\hat{x}}[\| \nabla D(\hat{x}) \|. \quad (2\text{-}42)$$

However, it is difficult to calculate all samples from x. They simply introduced an interpolation method for simulating sample $\hat{x}$:

$$\hat{x} = \alpha x_d + (1-\alpha) x_g. \quad (2\text{-}43)$$

By randomly sample x_d from P_d and x_g from P_g, interpolate these two samples by $\alpha \sim U(0, 1)$. And then constraints the gradients of discriminator by mean squared error to one.

This method effectively improved the training stability of GAN. And it is possible to optimize it with Adam. Figure 2-7 showed the distributions of the weight in different penalty methods. Nevertheless, when the input is complex such as conditional GAN, it is hard to interpolate samples from both P_d and P_g.

3) Spectral Normalization

Faragallah et al. (2020) introduced spectral normalization for GAN. Along with the core motivation of Wasserstein GAN, they applied Lipschitz Constraints with spectral normalization which no requirements for the input interpolation. Considering function *f* is a network which applies a linear transformation with weight W to input hidden layer *h*. They pointed that the Lipschitz constant of function f equals to its spectral norm:

$$||f||_{Lip} = \sup_{h} \sigma(\nabla f(h)) = \sup_{h} \sigma(W) = \sigma(W). \tag{2-44}$$

For a given weight matrix W that transform $h_{in} \rightarrow h_{out}$, $\sigma(W)$ is its spectral norm:

$$\sigma(W) \coloneqq \max_{h:h\neq 0} \frac{||Wh||_2}{||h||_2} = \max_{||h||_2 \leq 1} ||Wh||_2. \tag{2-45}$$

This indicates that the spectral norm number of W equals to its largest singular number. By applying spectral normalization to each layer in the discriminator:

$$W_{SN}(W) = \frac{W}{\sigma(W)}. \tag{2-46}$$

Therefore, the Lipschitz Constant will be constrained to one.

2.3.2.6 Improvement on Image Quality

Even though GAN has shown its great potential in generating images, it has been argued about its generated artifact and blurry region. It is still easy for human to distinguish whether the image is real or fake.

On the other hand, vanilla GAN only takes flatten random noise vector as input which is nonsense to the target image domain. Some of the training failed to capture the target

real image distribution but simply remember these training images. In such a way, the generator can pass the discriminator but has no capability to generate various images.

2.3.2.7 High Resolution Image Generation

Research works before the year of 2017 found it is hard to optimize a generator of deep layers. The Wasserstein GAN proposed in 2017 alleviated this issue. However, it is still difficult to generate image from flatten noise vector. In 2018, Progressive GAN applied a progressive method that cascaded generators and discriminators in different resolution (Karras et al., 2018). The training algorithm will train GAN in lower resolution until its convergent. And it will progressively combine higher resolution modules with the lower modules. But this way of training will increase the training time since the generators with lower resolution may be trained several times.

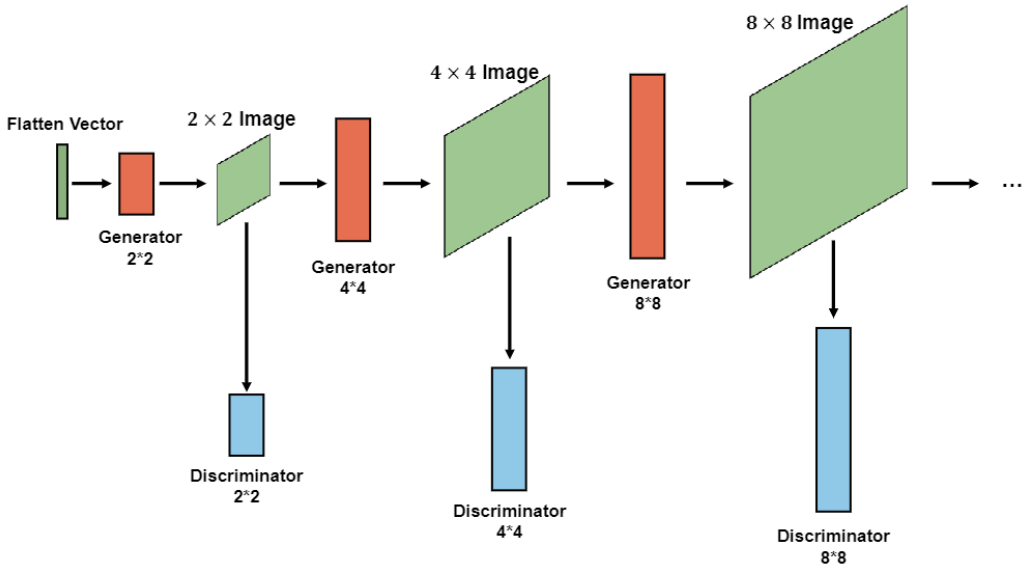


Figure 2-8 Progressive GAN.

Followed with progressively growing, StyleGAN suggested a way of decoding the input flatten noise vector into explainable style code that brings more details on

generated human face (Karras et al., 2019). They applied cascaded fully connected layer for decoding the noises into style code space w. They additionally introduced Noises B in different scales which 7 brought stochastic changes in different aspects. Their work showed huge potential of neural network that represents features in a high and abstract dimension. More papers thereafter explored the GAN inversion for getting controllable feature representations followed with StyleGAN. MSG-GAN is the state-of-the-art GAN architecture for generating human faces with high resolution (Karnewar & Wang, 2020). It takes the advantages of ProGAN and StyleGAN by introducing discrimination in different resolution.

In addition of the improvement of generator, the adjustment of discriminator benefits the image quality as well. PatchGAN was proposed in image-to-image translation tasks (Isola et al., 2017). The vanilla discriminator only predicts scalar number for determining whether the image is real. By predicting the real-fake game in image patches, the discriminator is able to focus more local information which enhances the details. When patch size was set to be 1, it degraded to vanilla GAN. When patch size equals to pixel's number, it will lose some global information. In the work of Isola et al. (2017), they set patch size to be $70 * 70$.

2.3.2.8 Improve the Mode Diversity

The objective of vanilla GAN optimizes JSD which has no constraints on the diversity of generated images. Since the generator learns from the discriminator, the way of

sampling is one of the reasons that lead to mode collapse. There are researchers explored different methods for having the discriminator to classify the diversity of images.

1) Sampling on Discriminator:

Since the informative gradients information were provided by discriminator, it is intuitively that allowing the discriminator to sample more images in advance. Salimans et al. (2016) believe that a greater number of batch size benefits GAN's diversity a lot. In Metz et al. (2017)'s work, they mentioned that the update of generator should take the after k times update of discriminator into consideration.

2) Classification Head for Discriminator:

Additional classification head in discriminator is another pipeline. This method often applied when involved in conditional GAN. Around 2016, there are multiple of methods have been proposed. Semi-Supervised GAN acclaimed that the utilizing of class information is a kind of semi-supervise learning (Odena, 2016). The discriminator would output not only the real or fake game, but also the class vector. The InfoGAN (Chen et al., 2016) and Auxiliary Classifier GAN (Odena et al., 2017) is very similar to Semi-Supervised GAN. The major difference is that these two GANs have the additional class information as input in generator. For the discriminator, the only difference is whether choose another branch to output the class vector. But these two

branches would share same bottom weights on discriminator. Figure 2-9 showed the general structure of classification head in discriminator.

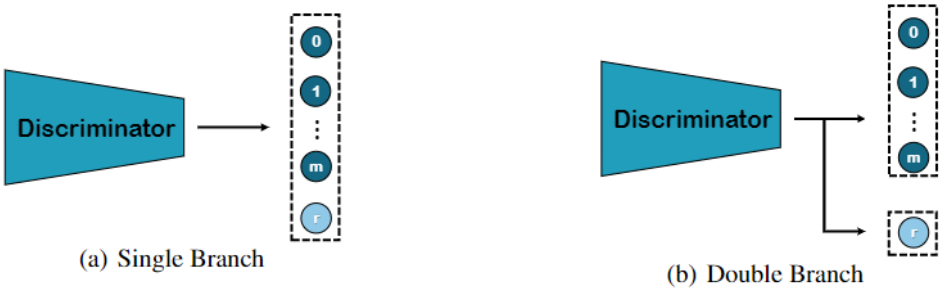


Figure 2-9 Classification Heads in Discriminators.

2.3.3 Denoising Diffusion Probabilistic Models

2.3.3.1 Fundamentals of Diffusion Models

Divergent from the previously discussed GAN, the denoising diffusion probabilistic model (DDPM) proposed by Ho et al. (2020) simulates a Markov chain. Although both methods are trying to learn a distribution of target data, GAN achieves this by implicitly learning a discriminator to measure the distribution distance while DDPM learns denoising probability to map Gaussian distribution to target data distribution.

To achieve such a process, DDPM progressively adds small amount of Gaussian noise ϵ to an image x_0 by t steps and then reverse this process by predicting the reversed distribution:

$$q(\mathbf{x}_{1:\mathcal{T}} \mid \mathbf{x}_0) = \prod_{t=1}^{\mathcal{T}} q(\mathbf{x}_t \mid \mathbf{x}_{t-1}). \tag{2-47}$$

$$q(\mathbf{x}_t \mid \mathbf{x}_{t-1}) = \mathcal{N}\left(\mathbf{x}_t; \sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t \mathbf{I}\right), \tag{2-48}$$

where $\mathcal{N}(\cdot)$ denotes a Gaussian distribution, t denotes time step, $\mathcal{T}$ denotes total time

step, and $\{\beta_t \in (0,1)\}_{t=1}^{\mathcal{T}}$ denotes a series of scalars to weight the strength of Gaussian noise ϵ. By applying the above equations, x_0 can be progressively transformed into $x_{\mathcal{T}}$, which is considered as a standard Gaussian noise. Later, the model is required to reverse this diffusion process by estimating the probability:

$$p_\theta(\mathbf{x}_{0:\mathcal{T}}) = p(\mathbf{x}_{\mathcal{T}}) \prod_{t=1}^{\mathcal{T}} p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t), \tag{2-49}$$

where p_θ is the reverse posterior probability. Since the forward diffusion process models each time step as a Gaussian, the $p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t)$ could be considered as a Gaussian as well:

$$p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \boldsymbol{\mu}_\theta(\mathbf{x}_t, t), \Sigma_\theta(\mathbf{x}_t, t)). \tag{2-50}$$

By simplifying Σ_θ as constant β_t, Ho et al. (2020) proposes that $\boldsymbol{\mu}_\theta$ is tractable as:

$$\boldsymbol{\mu}_\theta(\mathbf{x}_t, t) = \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_t\right). \tag{2-51}$$

During the training process, $\mathbf{x}_t$ is known and can be obtained by Equation (2-45). Therefore, the posterior can be obtained by a simple loss function:

$$\mathcal{L}_t^{\text{simple}} = \mathbb{E}_{t\sim[1,\mathcal{T}],\mathbf{x}_0,\epsilon_t}\left[\left\|\epsilon_t - \epsilon_\theta\left(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\epsilon_t, t\right)\right\|^2\right], \tag{2-52}$$

where $\alpha_t = 1 - \beta_t$ and $\bar{\alpha}_t = \prod_{i=1}^{t} \alpha_i$ for the simplicity.

2.3.3.2 Allowing Conditional Generation in Diffusion Models

Although several researchers have reported advancements achieved by diffusion models (Ho et al., 2020; Nichol & Dhariwal, 2021; J. Song et al., 2021), allowing control signals to guide synthesis direction remains a challenge. Unlike the previously

discussed GANs that can model a conditional probability $p_\theta(y|c)$, diffusion models synthesis an image from a latent variable z, which is dimensionally aligned with the target image. Therefore, the diffusion models not ideally suited for explicitly incorporating a condition to model ϵ_θ.

To tackle the issue of controllability, Dhariwal and Nichol (2021) propose using classifier guidance to introduce conditions in controlling the synthesis direction. Their work is inspired by GANs, where the discriminator can receive one-hot conditions, allowing the generator to synthesis an image reflecting the condition. During the synthesis process in diffusion models, the gradients from an explicit classifier can convey the conditional information to the estimated x_{t-1}:

$$x_{t-1} = \mathcal{N}\left(\mu + s\Sigma\nabla_{x_t}\log\, p_\phi(y \mid x_t), \Sigma\right), \tag{2-53}$$

where μ and Σ are mean and standard deviation estimated from the model ϵ_θ, p_ϕ is the explicit classifier, y is the desired condition, ∇_{x_t} are the gradients from the cross-entropy loss of y and x_t, and s is a scalar controlling the strength of the conditional gradients. By applying Equation (2-50), the final synthesized image x_0 can reflect the condition y.

However, a significant drawback of the explicit classifier guidance strategy is that the classifier can only take the intermediate noised image x_t as input. Existing classification models were trained on denoised images x_0. The distribution gap between

x_0 and x_t hinders the classifier's effectiveness. Additionally, this process is inefficient as it requires gradients at each denoising step.

To overcome this drawback, Ho and Salimans (2021) proposed classifier-free guidance (CFG). Their motivation involves transforming the explicit classifier into an implicit one. Primarily, they input the condition c into the model as $\epsilon_\theta(x_t, t, c)$, embedding the condition with the same dimension as t. Later, they consider the estimation of $\hat{\epsilon}$ as scores (Y. Song et al., 2021). The conditional generation can then be presented as:

$$\hat{\boldsymbol{\epsilon}}_t = (1 + w)\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, \mathbf{c}) - w\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, \emptyset), \tag{2-54}$$

where w is the CFG scalar controlling the importance of the given condition c, and $\emptyset$ denotes an empty condition, presented as a zero vector. By applying the classifier-free guidance, the training process allows the model to accept condition c with a dropout probability (e.g., 0.2). During the inference stage, the model $\boldsymbol{\epsilon}_\theta$ will execute twice, once with the condition and once without, treating the latter as empty. The CFG approach in controlling the synthesis direction has become the most accepted method, as it does not require training a noise-aware explicit classifier.

2.3.3.3 Towards Text-to-Image Synthesis

Diffusion models have recently emerged as a powerful branch of generative models, demonstrating their superior capabilities of handling image, text, audio as well as other modalities of data (Leng et al., 2022; Meng et al., 2022; Nichol et al., 2022; Rombach et al., 2022; Su et al., 2023). These models aim to learn the data distribution by

performing a Markov chain, simulating the data generation process in reverse (Dhariwal & Nichol, 2021; Ho et al., 2020; Nichol & Dhariwal, 2021; J. Song et al., 2021; Y. Song et al., 2021).

Despite the many research studies are focusing on synthesizing high-resolution images using diffusion models, there is a growing body of research that is interested in more controlled synthesis. Hertz et al. (2023) investigated a Prompt-to-Prompt mechanism of text-to-image generation, where text features activate feature maps through cross-modal attention. InstructPix2Pix (Brooks et al., 2023) combines the large pretrained language model GPT3 (Brown et al., 2020) and the state-of-the-art text-to-image LDM (Rombach et al., 2022) model to synthesize a dataset for text-driven image editing. Although these methods can synthesize images with corresponding semantics, they are trained on large open-domain datasets and have difficulty in capturing terms specific to the fashion domain. Recently, Textual Inversion (Gal et al., 2023) and DreamBooth (Ruiz et al., 2023) can adapt pre-trained diffusion models with new styles. Model retraining is, however, needed for every new style.

2.4 Chapter Summary

This chapter first reviews the fundamental architectures and optimization algorithms of deep learning technology in Section 2.1, highlighting that deep learning methods often adopts a supervisory approach and requiring annotations to train a model.

Subsequently, Section 2.2 illustrates the fundamentals of reinforcement learning and explains its modeling on sequential decision-making. Although reinforcement learning is powerful for enabling human-like intelligence, it focuses on *making decision* instead of graphically synthesis or editing an image. This thesis only provides a basic background of reinforcement learning.

Section 2.3 introduces the scenario of generative tasks that learn to map random noise to an image. Unlike commonly deep learning methods that function as discriminative models, the generative tasks require the model to learn a distribution. Later, Sections 2.3.2 and 2.3.3 describe more details of the GAN and diffusion models, respectively, the two most common and powerful types of generative model. In summary, GANs are lightweight and quick models that can respond to user input immediately. In contrast, diffusion models are powerful generative models that rely on an iterative generation process and require significant computational resources. Therefore, this thesis proposes using diffusion models to generate fashion images and employing GANs for image editing.

Chapter 3. CONTROLLABLE GENERATION MODEL

3.1 Introduction

Figure 3-1 A Visualization Demonstrating the Capability of the Proposed Model to Simultaneously Control Clothing Texture and Attributes.

The controllable generation model aims to achieve detailed control over synthesized fashion images in terms of both correct garment attributes and garment textures (styles). Figure 3-1 illustrates a scenario in which the style of Vincent van Gogh's 'Starry Night' is transferred to garments with various attributes.

Controlling detailed garment textures using natural language is challenging, therefore, the proposed model, named as SGDiff and illustrated in Figure 3-2, takes two inputs: a text condition (c_T) describing the garment attributes and a style condition (c_S) guiding

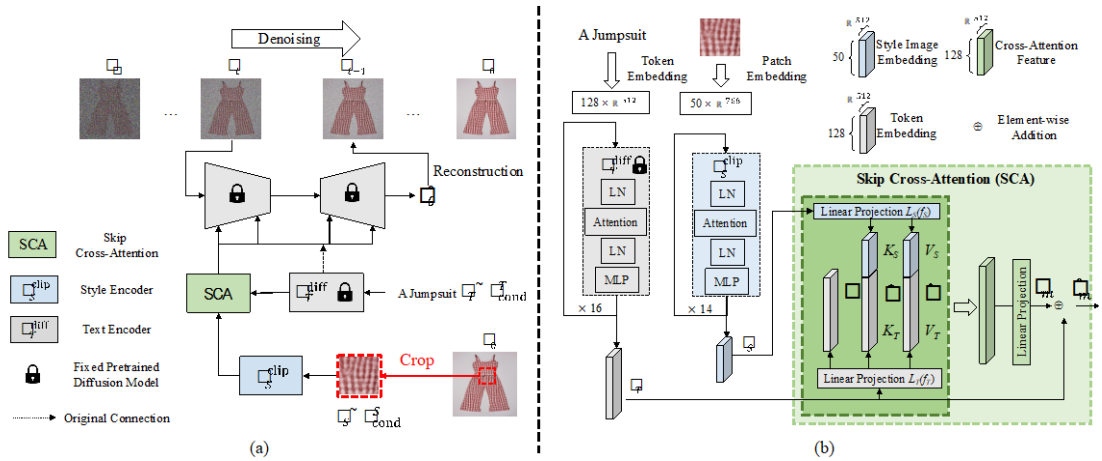


Figure 3-2 Overview of the Controllable Generation Model, namely Style-Guided Diffusion Model (SGDiff).

the synthesized garment texture. The text encoder E_T^{diff} of the diffusion model encodes the semantic representation f_T, and the style encoder E_S^{clip} of a pretrained CLIP model encodes the style representation f_S. The diffusion network ϵ_θ estimates the noise $\hat{\epsilon}_t$ as follows:

$$\hat{\epsilon}_t = \epsilon_\theta\left(x_t, E_T^{\text{diff}}(c_T), E_S^{\text{clip}}(c_S)\right). \tag{3-1}$$

To avoid labor-intensive data annotation, the conditioned image synthesis is formulated as an image reconstruction task, as shown in Figure 3-2 (a), in which a randomly image patch cropped from the garment image is taken as style condition c_S, the model is then trained to reconstruct garment according to the style guidance c_S. To achieve efficient training, SGDiff utilizes the pre-trained text-to-image diffusion model fine-tuned on a domain-specific dataset using text as input condition, according to a classifier-free guidance approach (Ho & Salimans, 2021). Next, by fixing the diffusion network parameters, the specially designed SCA module is optimized, and fine-tune a pretrained

image encoder E_S^{clip} with multiple conditions of text description and style guidance, which will be discussed in detail in Section 3.5.

3.2 Related Works

3.2.1 Fashion Synthesis

Fashion synthesis, an emerging research area within the broader field of computer vision and generative models, concentrates on generating and manipulating fashion-related images, such as clothing and accessories as well as fashion models. Virtual try-on (VTON) has generated considerable attention in some recent studies (Cui et al., 2021; Ge et al., 2021; Hu et al., 2022; Kim et al., 2020; Lewis et al., 2021; Xu et al., 2021), which typically employ human parsing maps and pose estimation techniques to transfer textures from a desired garment onto a target person. Although these VTON approaches successfully synthesize consistent clothing attributes, they primarily focus on human-centric scenarios.

Several recent studies have investigated garment-centric fashion synthesis, with the aim to generate novel and diverse clothing items. For example, Jiang et al. (2022) developed FashionG to transfer styles onto a garment without changing its original image content. Other researchers (Ding et al., 2023; C. Yu et al., 2019; D. Zhou et al., 2022) explored the synthesis of compatible fashion based on a given garment image as a query. These aforementioned studies are all using visual modality input as control for image

synthesis, their ability to control the detail attributes of the generated fashion is rather limited.

Text-to-image fashion synthesis remains relatively unexplored compared to other fashion synthesis approaches. Zhu et al. (2017) proposed a method that uses textual descriptions to edit images of garments worn by humans. X. Zhang et al. (2022) developed an ARMANI model for fashion synthesis based on multi-modal inputs including text descriptions and edge or regional detail in image modality. Although the above approaches successfully enable control over the synthesized garments, they generally fail to achieve detailed control of the synthesized textures or styles.

3.2.2 CLIP Model Guided Modality Fusion

The CLIP model, introduced by OpenAI (Radford et al., 2021), has revolutionized the field of computer vision by leveraging the power of large-scale transformers trained on both images and text. One of the main strengths of the CLIP model is its zero-shot learning capability, namely no learning is needed, which allows it to handle new tasks without requiring any task-specific fine-tuning. Its zero-shot capability has been exploited in various applications, such as image classification (Esmaeilpour et al., 2022; R. Zhang et al., 2022), object detection (Shi et al., 2022; Teng et al., 2021), and semantic segmentation (Liang et al., 2022; C. Zhou et al., 2022; Z. Zhou et al., 2022).

CLIP models have been integrated with generative models like GANs (X. Liu et al., 2021; Patashnik et al., 2021) and VQ-VAEs (Crowson et al., 2022) to produce impressive results in various tasks, from text-to-image synthesis to image editing. For example, StyleCLIP (Patashnik et al., 2021) utilizes a pretrained StyleGAN (Karras et al., 2020) and the CLIP model to align image and text features within the style space. VQGAN-CLIP (Crowson et al., 2022) uses CLIP as additional guidance to control the generation direction in pretrained generative model. FuseDream (X. Liu et al., 2021) is a training-free method integrating the latent generation space with CLIP embeddings. DALL·E (Ramesh et al., 2021) combines the CLIP model with a discrete VAE to generate high-quality images from textual descriptions. All these models adopt a training-free pipeline and treat the CLIP model as a gradient guidance to interpret the generation of latent space. Although these methods could integrate pretrained generation models with CLIP for text-to-image synthesis, they synthesize every image as a separate optimization process, which are computationally costly, and they fail to capture domain-specific text descriptions.

3.3 Method

3.3.1 Skip Cross-Attention Module

Figure 3-2 (b) illustrates the process of integrating two different modalities, namely text description of garment attributes c_T and image of style guidance c_S, in the proposed SGDiff model. The integration of the two input modalities is achieved through the

specially designed Skip Cross-Attention (SCA) module. Both encoders, E_T^{diff} and E_S^{clip}, employ transformer-based structures and the output features $f_T \in \mathbb{R}^{128\times512}$ and $f_S \in \mathbb{R}^{50\times512}$ represent two modalities of input. Such aligned features of f_T and f_S enable easy integration of the two representations by attention mechanism (Vaswani et al., 2017). To do so, the semantic representation f_T is linearly projected into query and key-value pairs:

$$Q, K_T, V_T = L_T(f_T), \tag{3-2}$$

where L_T represents linear projection, and query Q and key-value pairs K_T, V_T all have size $\mathbb{R}^{128\times512}$. The style representation f_S is projected into key-value pairs only:

$$K_S, V_S = L_S(f_S). \tag{3-3}$$

The style key-value pairs are concatenated with text key-value pairs:

$$\widehat{K} = K_S(+)K_T \text{ and } \widehat{V} = V_S(+)V_T, \tag{3-4}$$

where $(+)$ denotes length-wise concatenation.

Specifically, the semantic representation f_T is chosen as query Q because it provides key attribute information for garment synthesis. With f_T as query, style representation f_S is aligned with the garment attributes in order to improve the quality of the synthesized images. The cross-attention is implemented by integrating the key-value pairs from both modalities as follows:

$$f_m = \text{Attention}\,(Q, \widehat{K}, \widehat{V}) = \text{softmax}\left(\frac{Q\widehat{K}^T}{\sqrt{d_k}}\right)\widehat{V}. \tag{3-5}$$

Finally, the skip connection is applied, as shown in Figure 3-2:

$$\hat{f}_m = f_m + f_T. \tag{3-6}$$

The SCA module enables effective integration of text and image modalities, allowing the SGDiff model to control the synthesized texture without any reduction in semantic control.

3.3.2 Training Objectives

As discussed in Section 2.3.3.1, diffusion models implicitly learn to reconstruct an image from Gaussian noise. The network ϵ_θ estimates the noise in the current input noisy image $\mathbf{x}_t$. The training objective of DDPM (Equation (2-49)), however, does not address condition constraints explicitly. Therefore, SGDiff employs perceptual loss, in addition to Equation (2-49), to govern image synthesis. To this end, the reconstructed image $\hat{\mathrm{x}}_0$ is obtained at every time step t, according to the estimated noise $\hat{\epsilon}_t$ by Equation (3-1):

$$\hat{\mathrm{x}}_0 = \frac{1}{\sqrt{\bar{\alpha}_t}}\left(x_t - \sqrt{1-\bar{\alpha}_t}\hat{\epsilon}_t\right). \tag{3-7}$$

The Perceptual Loss (Johnson et al., 2016) is then calculated by:

$$\mathcal{L}_t^{\text{perc}} = \mathbb{E}_m \|\boldsymbol{\psi}_m(\hat{\mathbf{x}}_0) - \boldsymbol{\psi}_m(\mathbf{x}_0)\|_2, \tag{3-8}$$

where $\boldsymbol{\psi}_m$ denotes the m-th layer of VGG. Following Johnson et al. (2016), the layers of *relu1_2, relu2_2, relu3_2, relu4_2,* and *relu5_2* are used in Equation (3-8). The

overall training objective with Perceptual Loss, adapted from (Nichol & Dhariwal, 2021), is as follows:

$$\mathcal{L} = \lambda_s \mathcal{L}_t^{\text{simple}} + \mathcal{L}_t^{\text{vlb}} + \lambda_p \mathcal{L}_t^{\text{perc}}, \tag{3-9}$$

where λ_s and λ_p are balancing weights for the corresponding losses.

3.3.3 Multi-Modal Conditions

Classifier-free guidance (Ho & Salimans, 2021) has obvious advantages over classifier guidance (Dhariwal & Nichol, 2021) for conditioned generation with DDPMs. For more flexible control, the proposed SGDiff also adopts classifier-free guidance approach (Ho & Salimans, 2021), in which the model ϵ_θ is trained with conditional state c and unconditional state $\emptyset$ according to a certain probability $c \sim p_{\text{cond}}$:

$$\hat{\epsilon}_\theta(x_t, c) = \epsilon_\theta(x_t, \emptyset) + s[\epsilon_\theta(x_t, c) - \epsilon_\theta(x_t, \emptyset)], \tag{3-10}$$

Nevertheless, the above approach Equation (3-10) does not address more complex situation where conditions are multiple, happen in different combinations at varied probabilities. Until recently, InstrucPix2Pix (Brooks et al., 2023) suggested different weights for two conditions:

$$\begin{aligned} \widehat{\epsilon_\theta}(x_t, c_1, c_2) = \epsilon_\theta(x_t, \emptyset, \emptyset) \\ + s_1[\epsilon_\theta(x_t, c_1, \emptyset) - \epsilon_\theta(x_t, \emptyset, \emptyset)] \\ + s_2[\epsilon_\theta(x_t, c_1, c_2) - \epsilon_\theta(x_t, c_1, \emptyset)] \end{aligned}, \tag{3-11}$$

where s_1 and s_2 indicate the weight scale of condition $c_1 \sim p_{\text{cond}}^1$ and $c_2 \sim p_{\text{cond}}^2$, respectively. In Brooks et al. (2023)'s work, however, it was not discussed either the

order of c_1 and c_2 or the weight scales s_1 and s_2.

In the current task, Equation (3-11) is applied by setting the two conditions as c_T and c_S. The SGDiff is subjected to two conditions with independent conditional probability $p^S_{\text{cond}} = 0.8$ and $p^T_{\text{cond}} = 0.8$, which follows a typical classifier-free guidance training scheme (Ho & Salimans, 2021). In model training, like all text-to-image diffusion models, the unconditional state $\emptyset$ of textual condition c_T is set to padding token. The unconditional state $\emptyset$ of style guidance c_S is done by inputting a blank (background only) patch image.

Background masking: Apart from inputting a blank image patch as unconditional state, the background color in RGB space may also appear in the foreground. To avoid confusion, the background pixel values are masked to -255 to distinguish them from the normal RGB values. Such masking technique allows the model to focus more on the foreground texture. The effectiveness of such background masking setting will be evaluated in Section 3.4.

Condition order and weight scales: In order to explore the effect of the condition order, by setting $c_1 = c_S$ and $c_2 = c_T$, alternatively $c_1 = c_T$ and $c_2 = c_S$, in Equation (3-11), and $s_T = 1$, this will result in:

$$\hat{\epsilon}_\theta(x_t, c_S, c_T) = (s_S - 1)[\epsilon_\theta(x_t, c_S, \emptyset) - \epsilon_\theta(x_t, \emptyset, \emptyset)] + \epsilon_\theta(x_t, c_S, c_T). \quad (3\text{-}12)$$

$$\hat{\epsilon}_\theta(x_t, c_T, c_S) = (s_S - 1)[\epsilon_\theta(x_t, c_T, c_S) - \epsilon_\theta(x_t, c_T, \emptyset)] + \epsilon_\theta(x_t, c_T, c_S). \quad (3\text{-}13)$$

In the implementation, the model ϵ_θ takes c_S and c_T simultaneously, the two terms

$\epsilon_\theta(x_t, c_S, c_T)$ and $\epsilon_\theta(x_t, c_T, c_S)$ are therefore equivalent. Comparing Equation (3-12) wtih (3-13), thus $[\epsilon_\theta(x_t, c_S, \emptyset) - \epsilon_\theta(x_t, \emptyset, \emptyset)] = [\epsilon_\theta(x_t, c_T, c_S) - \epsilon_\theta(x_t, c_T, \emptyset)]$. It implies that if the style condition and text condition are independent, the condition order will not have a significant impact on the image generation. Moreover, the weight scale serves to adjust the influence of style guidance. When $s_S > s_T$ (i.e. $s_S > 1$ when $s_T = 1$), it introduces a positive conditioned direction to the denoising processing, emphasizing the influence of condition is guiding the synthesis. The multi-condition synthesis will be further evaluated in Section 4.4

3.4 Experiments

3.4.1 Datasets and Implementation Details

Figure 3-3 Overview of the Collected SG-Fashion Dataset.

In this study, a SG-Fashion dataset with 17,000 fashion product images was prepared, downloaded from e-commerce websites including ASOS, Uniqlo and H&M. A subset of 1,700 images was set aside as the test set. The dataset covers 72 product categories,

encompassing most types of garment items. Since SGDiff does not rely on textural descriptions, the original product titles were used as text descriptions. Apart from the SG-Fashion dataset, experiments were also conducted on the publicly available dataset of Polyvore (Han et al., 2017) using the same settings.

GLIDE (Nichol et al., 2022) was adopted as the backbone text-to-image diffusion model, which uses a low-resolution generation model for size 64×64 and a super-resolution model to upsample the generated low-resolution image to the size of 256×256. The generation model was fine-tuned and the super-resolution model was directly employed as the pretrained text-to-image model. For the pretrained CLIP image encoder, the vision transformer of ViT/32 was chosen. To speed up the synthesis process, DDIM (J. Song et al., 2021) scheduler with 100 sampling steps was adopted for all diffusion-based models.

The backbone model (GLIDE) was fine-tuned on the domain-specific dataset that the AdamW optimizer was used with a learning rate of $1e^{-4}$, and the model was optimized for 235,000 iterations. Due to GPU limitations, the batch size was set to 8, and the GLIDE was trained on a single RTX 3090 GPU. AdamW was also used, but with a learning rate of $1e^{-5}$ for training the SGDiff with 50,000 iterations for all experiments on a single RTX 3090 GPU. In terms of the SCA module, multi-head attention with 4 heads was adopted. In all experiments, $\lambda_s = 1$ and $\lambda_p = 0.001$ were set in Equation (3-

9). Since the training of SGDiff fixes the parameters of the pretrained backbone, a larger batch size of 16 could be used. For SGDiff training, a single texture patch was cropped from the foreground. To ensure this cropped patch provides sufficient style information, BASNet (Qin et al., 2019), a boundary-aware salient object segmentation method, was applied to obtain the foreground segmentation map.

3.4.2 Qualitative Evaluation

The qualitative evaluation compares the SGDiff results with several SOTA text-to-image generation methods, including LDM (Rombach et al., 2022) and GLIDE (Nichol et al., 2022) for diffusion-based methods, and FuseDream (X. Liu et al., 2021) and VQGAN-CLIP (Crowson et al., 2022) for CLIP-guided GAN-based methods. All selected SOTA methods have zero-shot capability. Figure 3-4 presents a comprehensive qualitative comparison of these methods. The 2nd and 3rd rows illustrate the results of CLIP-based methods of VQGAN-CLIP (Crowson et al., 2022) and FuseDream (X. Liu et al., 2021), while the 4th and 5th rows illustrate the results of diffusion-based methods of LDM (Rombach et al., 2022) and GLIDE (Nichol et al., 2022). The 6th row illustrates SGDiff's ability to incorporate style images (the 7th row) into text conditions (the 1st row), successfully synthesizing garments with the desired textures. Generally speaking, FuseDream and LDM could synthesize garments in most cases, while VQGAN-CLIP and GLIDE could only synthesize fabrics. The proposed SGDiff could successfully implement the fashion synthesis with desired clothing category and style. Specifically,

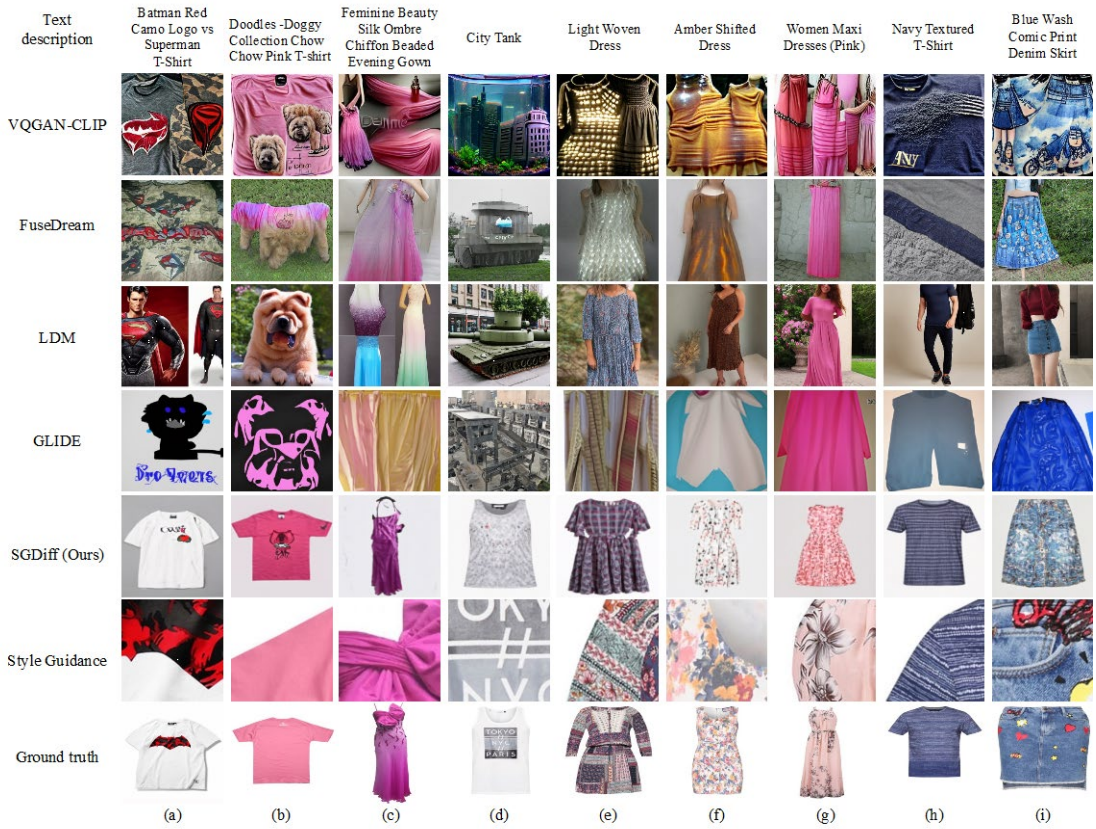


Figure 3-4 Qualitative comparison of SGDiff with state-of-the-art (SOTA) approaches.

when synthesizing a garment with complex text descriptions (see examples in columns (a), (b), and (c)), the other methods tend to ignore the key message but capture part of the semantics like Batman logo, pink doggy, or silk, while SGDiff tends to synthesize clothing and consider the style guidance to control the synthesized textures. Moreover, semantic confusion is one of main challenges in text-to-image synthesis. For instance, `*Tank*' refers to a specific type of upper clothing in the fashion domain. Column (d) of Figure 3-4 shows that both the diffusion-based and CLIP-based approaches have difficulty in capturing domain-specific semantics. Since their generation objective focuses on optimizing the CLIP-Score, the synthesis results may not always guarantee that the output is a piece of clothing. The other columns present cases when offering

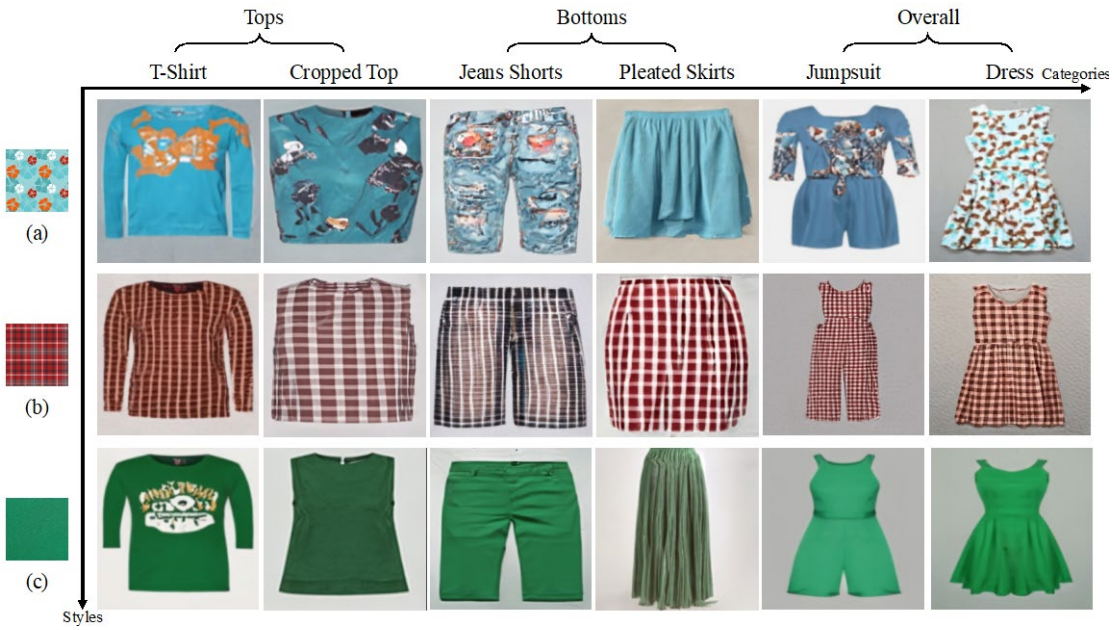


Figure 3-5 Illustration of SGDiff's capability to synthesize garments across various categories and styles, using style guidance of different colors.

textual descriptions like amber, light and pink, although the other SOTA methods could synthesize clothing with textures that are similar to the descriptions, they show greater differences to the ground truth images comparing to SGDiff. In conclusion, SGDfiff is suitable for fashion synthesis since it could capture the garment category and desired styles. Moreover, it performs consistently well across various clothing categories. In addition to the comparative analysis, Figure 3-5 illustrates the innovative capability of SGDiff in synthesizing garments across various categories and styles. With style guidance images under different color schemes, SGDiff effectively transfers styles from the guidance images to the synthesized garments, meeting the condition of garment attributes. Figure 3-5 shows a range of synthesized fashion under specific color scheme in each column, offering valuable inspiration for innovative fashion design. When conditioned generation are out of the training set, SGDiff can still exhibit a remarkable

generative capability by successfully blending different condition combinations, e.g., the jeans shorts with red check and green patterns showed in columns (b) and (c) are not existed in the training data. Moreover, the style guidance appears in interesting variations in the generated fashion. These results highlight the versatility and robustness of the SGDiff model in the realm of fashion design. Appendix A presents additional qualitative results of garments synthesized using SGDiff, as illustrated in Figure A-1 and Figure A-2.

3.4.3 Metrics and Quantitative Evaluation

Table 3-1 Quantitative evaluation and comparison of various SOTA methods.

Datasets	SG-Fashion			Polyvore		
Metrics	LPIPS ↓	FID ↓	CS ↑	LPIPS ↓	FID ↓	CS ↑
VQGAN-CLIP	0.7364	95.84	22.20	0.7122	68.01	**39.65**
FuseDream	0.7067	60.44	**38.03**	0.7032	**41.94**	38.53
LDM	0.7158	85.73	31.66	0.7214	59.79	31.89
GLIDE	0.6921	78.70	23.72	0.7164	63.85	23.28
Ground Truth	-	-	29.13	-	-	29.88
Baseline	0.5772	36.13	27.31	0.6637	43.50	26.24
SGDiff (Ours)	**0.4474**	**32.06**	27.53	**0.6369**	41.98	27.33

Table 3-1 shows the quantitative evaluation, in which three metrics, including FID (Heusel et al., 2017), LPIPS (Zhang et al., 2018) and CLIP-Score (CS) (Radford et al., 2021), are used to assess and compare the performance of SGDiff with other SOTA methods. FID and LPIPS measure the distance in feature space, with FID focusing on the overall distribution statistics of the generated/synthesized images and the ground truths, while LPIPS computes the distance between each pair of synthesized image and the corresponding ground truth, lower the FID and LPIPS values higher the image

quality. In contrast, the CLIP-score measures the semantic correspondence, namely the cosine similarity between synthesized images and their corresponding text descriptions, with higher scores indicating better alignment.

As shown in Table 3-1, SGDiff model performs the best in terms of LPIPS, comparing to other SOTA methods on both SG-Fashion and Polyvore datasets. SGDiff's FID value is also the lowest for SG-Fashion dataset and only slightly lower than FuseDream for Polyvore dataset by 0.04%. This demonstrates that the SGDiff model can generate better images fulfilling the conditions without sacrificing the image quality. The CS of the SGDiff is higher than GLIDE and the baseline (i.e. GLIDE being fine-tuned on the datasets), but lower than FuseDream and LDM, because FuseDream optimizes the BigGAN-256 (Brock et al., 2019) latent space using CLIP guidance and LDM leverages a vast text-to-image dataset consisting of billions of examples. Nevertheless, these methods did not consider the integration of the text feature and image feature for image generation, they indeed did not perform well in LPIPS and FID.

Table 3-2 Consumption of synthesizing an image with resolution of 256 × 256 on a RTX 3090 GPU.

	VQGAN-CLIP	FuseDream	LDM	GLIDE	Ours
Time	62 s	171 s	5.9 s	9 s	9.8 s
Memory	5686M	9296M	6570M	5550M	5986M

Table 3-2 compares the model memory and average time cost for synthesizing an image of size 256 × 256 on a RTX 3090 GPU. As shown, the running time of the SGDiff

model is much shorter than that of VQGAN-CLIP and FuseDream. Although the running time of the SGDiff model is slightly longer than LDM, the memory consumption is lower. Compared to the baseline, the increases in time and memory are relatively insignificant because only the image encoder and modality fusion module are fine-tuned. In summary, the SGDiff can be trained without much memory and can generate an image with good quality based on text and style conditions within 10 seconds on RTX 3090.

3.4.4 Ablation Study

Table 3-3 Ablation experiments on modality fusion methods and classifier-free approaches.

Classifier-free	Mask	Modality fusion	LPIPS ↓	FID ↓	CS ↑
Equation (3-10)		$\oplus$[1]	0.6833	42.63	25.63
Equation (3-10)		CA[2]	0.5650	38.88	25.39
Equation (3-10)		SCA	0.5607	39.21	25.98
Equation (3-10)	✓	SCA	0.5695	37.22	26.06
Equation (3-11)	✓	SCA	**0.4474**	**32.06**	**27.53**

[1] $\oplus$ refers to an element-wise addition operation, where the features f_T and f_S are projected onto the same dimension before operation;

[2] CA indicates SCA module without skip connection, w.r.t. Equation (3-5) without Equation (3-6).

Ablation study was conducted to evaluate the effect of each component of the proposed SGDiff on SG-Fashion dataset.

3.4.4.1 Effectiveness of the SCA:

As demonstrated in Table 3-3, the comparison between the element-wise addition of features and the cross-attention (CA) method shows that CA is significantly more effective in improving LPIPS and FID scores. However, it has the downside of causing a decline in semantic information, as CS decreases. To address this issue, the SCA module with skip connections was use. As shown in the third row of the table, SCA leads to improvements in both LPIPS and CS scores, demonstrating its ability to improve the similarity between synthesized images and ground truth images.

3.4.4.2 The effect of background masking:

As shown in Table 3-3, after applying background masking, the FID value decreases by 1.99 and the CS remains almost the same. This demonstrates that background masking is beneficial to improve image quality. The reason for slightly increased LPIPS is that LPIPS is sensitive to perceptual information, the lack of background may degrade LPIPS metric. However, the fashion synthesis task only focuses on the synthesized foreground, and the background could be easily removed by salient object segmentation model like BASNet (Qin et al., 2019).

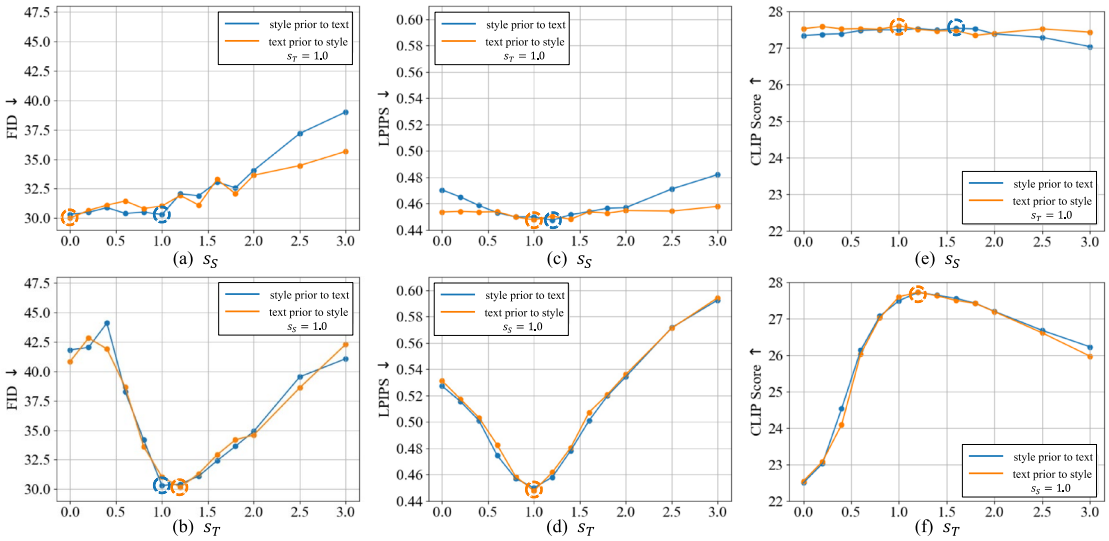


Figure 3-6 *Ablation study on the impact of style and text guidance on the performance of SGDiff in terms of (a) and (b) for FID, (c) and (d) for LPIPS and (e) and (f) for CLIP-score.*

3.4.4.3 The orders and weights for different conditions:

Figure 3-6 displays the relationship between FID, LPIPS and CS with different conditional weights and order settings. One conditional weight was set to vary in the range of [0, 0.2, 0.4, 0.6, 0.8, 1.0, 1.2, 1.4, 1.6, 1.8, 2.0, 2.5, 3.0], while the other conditional weight is fixed at 1.0. The trend of setting text prior to style is similar to setting style prior to text, indicating little impact on results with fixed $s_S = 1$ and varying s_T. In addition, it can be seen from Figure 3-6 that the optimal values (see the circled dots of Figure 3-6) of s_S and s_T are almost in the range of 1.0 to 1.6. More specifically, the setting of $s_S = 1.2$, $s_T = 1.0$, with *style prior to text*, was chosen as optimal. This setting achieves the best LPIPS which is important in controlling synthesized styles. The numerical results are shown in the last row of Table 3-3. Although the CLIP-Score is lower compared to other methods, the qualitative results

indicate that a higher LPIPS suggests better visual performance in this controllable generation task. Additionally, users can achieve a better CLIP-Score by increasing the text weight s_T.

3.5 Chapter Summary

This chapter reports on the implementation of the controllable generation model, referred to as the Style Guided Diffusion model (SGDiff), which forms a core component of the overall system. SGDiff represents a significant stride in the realm of image synthesis, specifically designed to address and overcome the limitations inherent in traditional diffusion models.

Central to SGDiff's innovation is the introduction of a style condition, which essentially acts as a decoupled condition within the model. This decoupling allows for a more controlled integration of style elements into the pretrained text-to-image diffusion frameworks. The effectiveness of SGDiff is highlighted by its ability to operate with a high degree of precision in texture synthesis, all while circumventing the need for extensive labelled datasets or computational resources.

Looking forward, SGDiff will be enhanced by refining the control over various texture attributes, including colour themes, patterns, and material qualities. This enhancement is anticipated to not only extend SGDiff's technical contributions but also to broaden its

applicability across diverse applications and fields of the controllable generation of synthesized images.

Chapter 4. FLEXIBLE EDITING MODEL

4.1 Introduction

(a) Edit to a Circular Skirt (b) Edit to a Tiered Skirt

Figure 4-1 Demonstration of Flexible Clothing Shape Editing in Application Usage.

This editing model is designed to flexibly modify arbitrary regions according to user-specified sketches. In the context of fashion editing, it enables the alteration of large, interesting areas of various shapes. For instance, as demonstrated in Figure 4-1, a designer might frequently modify their drafts, such as transforming a dress into different styles of skirts. Given the frequent need for modifications in drafts, this model employs a GAN-based architecture. It also marks a transition from a diffusion model to a GAN model. While diffusion models require several inference steps and can take up to 10 seconds to generate an image on a single RTX 3090 GPU, GANs can produce an image in approximately one second.

To implement this model, the general requirement for the input is an existing image I, a binary mask image M that denotes the editing area, and a user-provided sketch map S. Figure 4-2 illustrates this pipeline, referred to as CoDE-GAN.

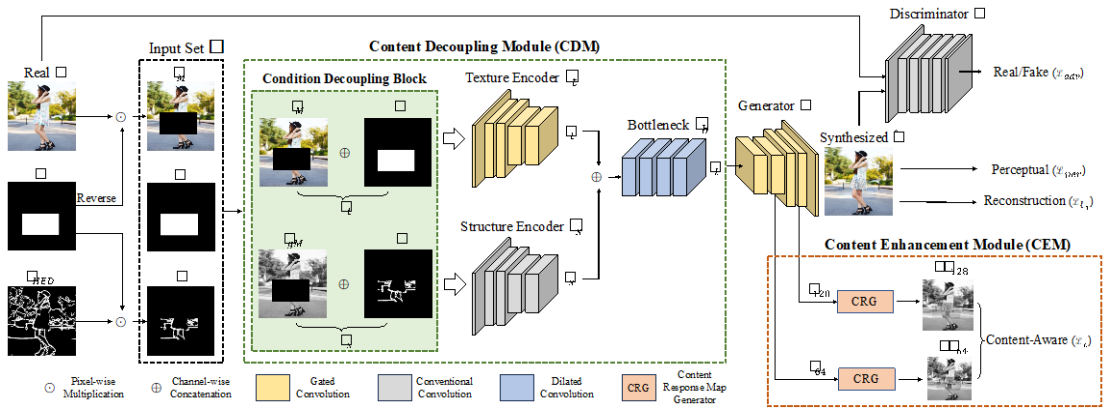


Figure 4-2 The Proposed CoDE-GAN Utilizing a Mask-Reconstruction Pipeline.

The proposed pipeline picks up the DeepFill V2 (J. Yu et al., 2019) as backbone. This backbone relies on the mask-reconstruction proxy task to relief the lack of paired data between the source image and the edited target image. The mask reconstruction could be considered as image inpainting but with the guidance of sketches. The DeepFill V2 is a benchmark algorithm in image inpainting task.

However, there is a gap between the image inpainting works and the proposed editing task. The inpainting approaches can only synthesis a region with no consideration on the user-provided conditions. To bridge this gap, this study elaborately designed the **C**ontent **D**ecoupled and **E**nhanced **GAN** (CoDE-GAN) using Content Decoupling Module and Content Enhancement Module to fit in the fashion editing task.

4.2 Related Works

4.2.1 Fashion Editing Tasks

In fashion editing tasks, researchers pay much attention to editing some attributes of a fashion image. In response to various kinds of input, the existing methods are capable to control the final editing results from different levels.

Motivated by the achievements of semantic synthesis (Park et al., 2019; Schnfeld et al., 2021) and human parsing (Ruan et al., 2019), it is possible to edit a fashion image by improving the shape of its parsing map. This editing could be considered as human synthesis as well. Frühstück et al. (2022) conditioned synthesis on a human parsing map. Dong et al. (2020) propose a two-stage fashion editing pipeline that generates an edited human parsing map firstly and synthesizes based on the parsing map. Their edited human image can respond to the conditioned sketch and color. In virtual try-on works (Cui et al., 2021; Neuberger et al., 2020; Wang et al., 2018; Yang et al., 2020), they are capable of editing the whole human image by offering human poses and specified fashion garments.

In addition to editing whole fashion human images, Dai et al. (2021) argue that it is important to edit design drafts. Their fashion editing workflow formulates the fashion editing task as a bidirectional image translation task. By translating an in-shop fashion garment to design drafts, it benefits the designer in making modifications. And then,

their pipeline is able to translate the edited drafts back into in-shop garments. TailorGAN (Chen et al., 2020) achieves fashion attribute editing by specifying a reference image. For addressing the lack of paired data between input garments and edited images, TailorGAN proposes a self-supervision training pipeline. By reconstructing a masked attribute region with the guidance of a reference image, TailorGAN has the ability to apply fashion editing tasks. Nevertheless, this method can only address collar and sleeves editing which leads to poor generalization to other attributes. Even though the existing works are capable of editing fashion garments to some extent. It is demanding to provide a user-friendly interaction in editing in-shop clothing.

4.2.2 Sketch-Guided Editing Tasks

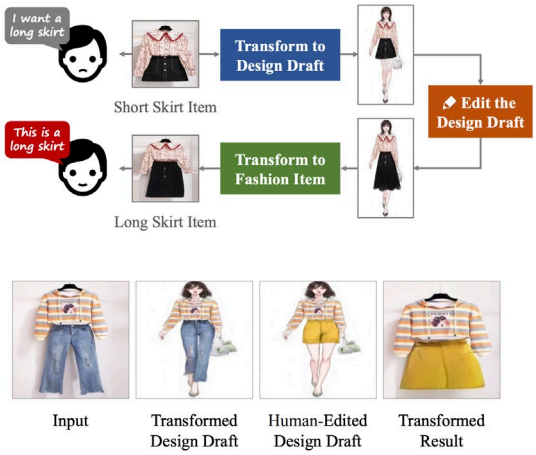
Editing tasks in fashion require location guidance. There are clear regions that user would like to edit. Hence, the wanted-editing region will be offered as input. By offering a mask map as wanted-editing region, editing tasks could be considered as image inpainting task. DeepFill V2 (J. Yu et al., 2019) offered a user-guided way of editing image. Besides the damaged image and reference target mask map, their network architecture takes user sketch as an additional input channel. Nazeri et al. (2019) proposed an edge connect way for reconstructing the sketch map in damaged region firstly. As prior information, the recovered edge map contributed to the completion task. Their edge connect pipeline enables user-guided editing as well. Jo and Park (2019)

introduced SC-FEGAN for addressing face editing tasks. By inputting additional user-guided color channel, SC-FEGAN is capable of editing face images with specific shape and color.

4.2.3 Image Translation

Recently, benefits from the succeeding of generative adversarial networks (GAN), it is possible to generate and edit the fashion image easier and faster. For instance, by reshaping the conditioned input parsing map, it is able to edit a whole human fashion image (Frühstück et al., 2022; Li et al., 2021). Cui et al. (2021) and Han et al. (2019) provide pose estimation to transfer different poses to a specified source human image. Chen et al. (2018), Y. Li et al. (2019), and Li et al. (2020) utilize text information to instruct attributes editing. Their works effectively consider the complex input condition to constrain the generation and achieve astonishing results. However, their input conditions are in-flexible to make modifications to the clothing. The parsing map could condition the shape of a fashion garment but failed to condition inner details. Pose estimation provides spatial prior information. It is effective in conditioning the viewpoint of an image but lacks the ability to edit the shape of a fashion garment. The text instructions semantically conditioned the editing but are hard to accurately control the length of sleeves or pants. For flexibly editing the image like a fashion designer, it is straightforward to provide in-complete sketches to edit a specific area.

(a) Fashion Editing with Adversarial Parsing Learning (Dong et al., 2020)



(b) Fashion Draft Editing and Translation (Dai et al., 2021)

(c) Fashion Attribute Swapping (Chen et al., 2020)

(d) Fashion Attribute Shape Manipulation (Kwon et al., 2022)

Figure 4-3 Multiple Fashion Image Generation and Editing Tasks.

Although the existing image synthesis works (Dai et al., 2021) have achieved astonishing results, they mostly discuss unconditional generation that there are no constraints on the generation process. What's more, fashion domain often requires generating fashion images with specified types e.g., clothing texture, collar types, dress styles, etc. Achieving controllable fashion image generation is challenging. This controlled generation requires the model to synthesize images that accurately represent the desired design elements, such as color, pattern, and shape. These elements can be

specified in any manner, regardless of the number and types of elements involved. generation refers to generate images in respect to input conditions. The input conditions could be one-hot codes for denoting fashion attributes like colors or clothing types. As well, the conditions could be much more complex and abstract like pose estimation, texts, and so on. In general, the controlled generation translates the source image to the target image by applying the above-mentioned conditions to the source image. This process is regarded as image translation (Isola et al., 2017) or image editing (H. Liu et al., 2021).

Generally, the regular image editing works focus much on human face editing (Jo & Park, 2019; Korshunova et al., 2017; Portenier et al., 2018). Human face is the type of data that have been well-explored. On the one hand, face images could be easily collected from the Internet. On the other hand, there are plenty of research works about face detection (Yang et al., 2016), face recognition (Meng et al., 2021), and face deepfakes (Peng et al., 2021). Therefore, it's low-cost to collect aligned and cropped human face dataset for analyzing. In contrast to human face editing, there are fewer fashion image editing research works that modify and regenerate an actual fashion garment image with a high level of realism. Compared to face images, fashion image editing is more difficult due to the complexity of apparel attribute definition, which includes global attributes such as garment style, fabric color and texture. For apparel products, the design process is complex and expensive and labor-intensive, and the

most time-consuming part of the process is completing the design drawings, which are the transformation from a draft to a real image of the apparel. This is because the designer needs to imagine what colors and fabric materials will work with the design to show the style more perfectly.

4.3 Method

4.3.1 Problem Formulation

Let $I \in \mathbb{R}^{3\times w\times h}$ be the ground truth RGB image where w is the image width and h is the image height, $M \in \mathbb{R}^{1\times w\times h}$ be the binary mask where 1 indicates editing or masked area and 0 indicates the unmasked region, and S be the input sketch, the sketch-guided image editing model will generate a new image which is filled in the consistent texture in the masked region M and has the consistent sketch with S. During the training stage, sketch S is extracted by edge detection network HED (Xie & Tu, 2015) $H(\cdot)$ and multiplied with the mask M, which can be defined as:

$$x_t = I_M \oplus M, \tag{4-1}$$

where $\odot$ is the element-wise multiplication. Since HED can only output a greyscale sketch map, S is binarized by setting the threshold to 0.6 to simulate users' drawn sketches. During the inference stage, S is drawn by the users in the editing area. In general, the inputs of the sketch-guided image synthesis are the set $x = [I_M, M, S]$, where I_M is the masked RGB image obtained by:

$$I_M = I \odot (1 - M). \tag{4-2}$$

To make the model learn specific texture and structure representation for better image synthesis, the CDM is designed to learn the decoupled texture representation and structure representation and fuse them to obtain better latent representation for image generation. Let the latent representation be f_l, it can be represented by:

$$I_M = I \odot (1 - M). \tag{4-3}$$

The latent representation is then fed into a generator $\mathcal{G}$ to generate a synthesized image, which is defined by:

$$\hat{I} = \mathcal{G}(f_l). \tag{4-4}$$

Lastly, four loss functions are used to train the network to make the synthesized image $\hat{I}$ similar to the original image I as much as possible. The detail of the loss functions is illustrated in Section 4.3.5.

4.3.2 Content Decoupling Module

The content decoupling module consists of a Condition Decoupling Block (CDB), a structure encoder, a texture encoder, and a bottleneck.

a) **Condition Decoupling Block:**

This block decouples the input x into two types of conditions: the texture condition x_t and the structure condition x_s. Given the image I, the mask M and the sketch S, the x_t and x_s can be computed by:

$$x_t = I_M \oplus M \tag{4-5}$$

$$x_s = I_{gM} \oplus M \tag{4-6}$$

where $\oplus$ is channel-wise concatenation, and $I_{gM} \in \mathbb{R}^{1\times w\times h}$ is the grey image of I_M. It can be seen from the formula that the input $x_t \in \mathbb{R}^{4\times w\times h}$ aligns the setting of image inpainting and the input $x_s \in \mathbb{R}^{2\times w\times h}$ is conditioned to the sketch. Here, x_s incorporates sketch with the grey image instead of RGB image, because grey image is more effective to represent structural information than RGB image and reduces the representation space from $\mathbb{R}^3$ to $\mathbb{R}^1$. Moreover, traditional image processing algorithms, such as Canny edge detection, typically work with grey images to obtain edge details.

b) **Texture Encoder:**

The texture encoder ϵ_t feeds in the condition x_t and learn the texture representation by:

$$f_t = \epsilon_t(x_t), \tag{4-7}$$

where ϵ_t is the texture encoder. As the texture encoder mainly aims to reconstruct the texture of the masked region, which is the same as the image inpainting task, this model adopts the encoder structure of DeepFill V2 (J. Yu et al., 2019). DeepFill V2 designs a gated convolution that adapts a dynamic feature selection mechanism to make the convolution dependent on the soft mask that is automatically learned from data and improves the texture consistency and inpainting quality of the masked region. Specifically, for the input feature f_{in}, a gated convolution $Conv_g$ applies an additional convolution to obtain a soft weight map and then multiples it with a learned feature of f_{in}. It

is formulated as:

$$\mathrm{Conv}_g\ (f_{\text{in}}) = \mathrm{Conv}\ (f_{\text{in}}) \odot \sigma(\mathrm{Conv}_d(f_{\text{in}})), \tag{4-8}$$

where Conv is the conventional convolution, Conv_d is the convolution that outputs single-channel feature map, and σ is the sigmoid function that scales learned gating to range (0, 1).

c) **Structure Encoder:**

The structure encoder ϵ_s takes the input x_s and learns the structure representation f_s by:

$$f_s = \epsilon_s\ (x_s). \tag{4-9}$$

The structure of ϵ_s is same with ϵ_t, but the gated convolution is replaced with conventional convolution. There are two reasons for using conventional convolution here: 1) the intension is for the encoder to primarily focus on capturing the basic structure of the whole image, and thus the texture information learning is not that important and will be achieved by the texture encoder. 2) Gated convolution adapts an extra convolution to learn the soft weighting map, leading to an increase in computation cost.

d) **Bottleneck:**

Lastly, the texture representation and structure representation are fused by a bottleneck structure to reduce the representation space. The bottleneck structure consists of four dilated gated convolution blocks. First, f_t and f_s are concatenated, and then fed into a bottleneck ϵ_b to obtain the fused latent

representation f_l. It is formulated as:

$$f_l = \epsilon_b(f_t \oplus f_s). \tag{4-10}$$

4.3.3 Adversarial Generation

To allow the synthesized results more realistic and reasonable, the adversarial generation process is incorporated.

a) **Generator:**

Given the fused latent representation f_l, the generator $\mathcal{G}$ could synthesizes a fake image $\hat{I}$:

$$\hat{I} = \mathcal{G}(f_l) \odot M + I_M. \tag{4-11}$$

The $\mathcal{G}$ consists of five gated convolution blocks with twice upsampling which is symmetric to the structure of encoder ϵ_t.

b) **Discriminator:**

Following with Pix2Pix (Isola et al., 2017), a patch discriminator $\mathcal{D}$ was implemented, which output real/fake discrimination on image patches instead of the whole image. Its discrimination could focus on local details and enhance the fidelity of the generated image. The structure of $\mathcal{D}$ is like an encoder that only consists of six convolution blocks. Besides, to stabilize the adversarial training process, spectral normalization was adopted on the discriminator as well (Miyato et al., 2018).

4.3.4 Content Enhancement Module

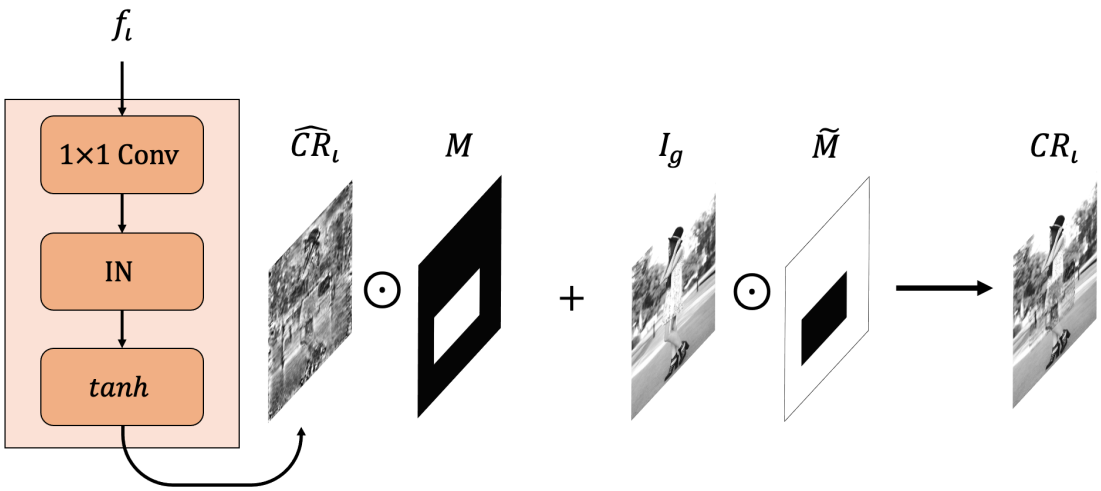


Figure 4-4 Content response map generator (CRG) transforms features into content response map. The response map is masked and fused with a grey image.

To further improve the consistency of synthesized content, a Content Enhancement Module (CEM) is applied to the generator $\mathcal{G}$. As shown in Figure 4-2, CEM extracts the features from the second and fourth blocks. The features have different resolutions and are denoted as f_{64} and f_{128} of which the subscript indicates the resolution of the feature map. Then, the two features are respectively fed into a **C**ontent **R**esponse Map **G**enerator (CRG) to generate the content response maps CR_{64} and CR_{128}. As Figure 4-4 illustrates, the content response map CR_i could be obtained by:

$$\begin{aligned} CR_i &= CEM(f_i) \\ &= \tanh[\mathrm{IN}(\mathrm{Conv_d}(f_i))] \odot M + I_g \odot (1 - M), \end{aligned} \tag{4-12}$$

where $i = 64$ or $i = 128$, $\mathrm{Conv_d}$ reduces the feature dimensionality of f_i to single, IN denotes an instance normalization layer, and tanh is a Tanh activation function. Then, the cosine similarity between CR_i and the grey image I_g is calculated and regarded as an objective function, which is computed by:

$$\mathcal{L}_c = \left(1 - \frac{CR_{64} \cdot I_g}{\|CR_{64}\|\|I_g\|}\right) + \left(1 - \frac{CR_{128} \cdot I_g}{\|CR_{128}\|\|I_g\|}\right). \quad (4\text{-}13)$$

The goal is to optimize the features of the generator through gradient backpropagation by minimizing the similarity distance between the CR_i and the grey image I_g.

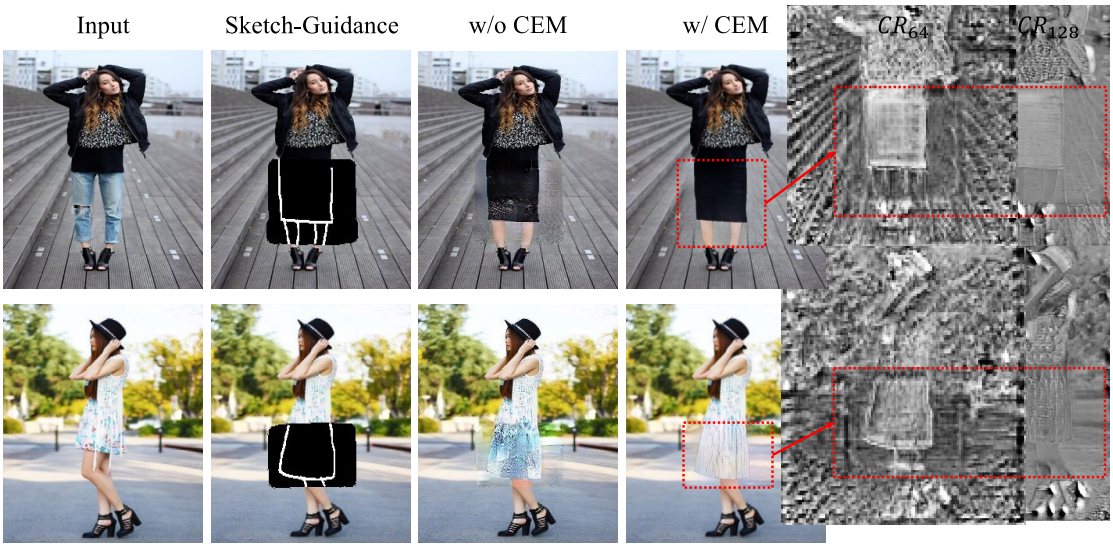


Figure 4-5 *Visualization of the synthesized content response map* CR *at resolution of* 64×64 *and* 128×128.

The content response maps at resolutions $64 \times 64(CR_{64})$ and $128 \times 128(CR_{128})$ were visualized in Figure 4-5. It could be observed that the CEM could learn the structure and texture of the image and the content response map with a higher resolution clearly exhibits more uniform content and sharper boundaries. Since the input sketch is sparse and gradually diminishes in the CNN feature space, it is important to inject the sparse sketch information in the CNN space, especially in the generator. In DeFlocNet (H. Liu et al., 2021), the control inputs are injected in all blocks of encoders and generators to preserve the guidance information. However, this method will add additional computation costs and cannot provide other content information except the

input controls, like the structure and texture information around the sketch. In this case, the features of the generator are optimized to resemble the original grey image, which contains rich structure and texture information. By doing so, the generator learns to recover the structure and texture of the masked region as shown Figure 4-5. Consequently, the proposed CEM is able to enhance and refine the content information, leading to more detailed and high-quality generation results.

4.3.5 Optimization Objectives

For training the CoDE-GAN, except for the above-mentioned content-aware loss, reconstruction loss, perceptual loss, and generative adversarial loss are used. In the following, these loss functions are introduced in the following:

a) **Reconstruction Loss:**

To ensure the generated image $\hat{I}$ is close to the RGB image I within the unmasked region, L1 loss is used between them on the unmasked region. It is defined by:

$$L_{\ell 1} = |I - \hat{I}|_1 \odot M. \tag{4-14}$$

b) **Perceptual Loss:**

Following style transfer, perceptual loss (Johnson et al., 2016) was introduced to keep the perceptual information as well. It is obtained by:

$$\mathcal{L}_{per} = \sum_i w_i \cdot L1\big(F_i(I) - F_i(\hat{I})\big), \tag{4-15}$$

where F_i stands for ith activation layer of VGG-19 network, and w_i is the corresponding weight. Specifically, the selected layers are *relu1_1, relu2_1,*

relu3_1, relu4_1 and *relu5_1*. In the experiments, all the corresponding weight w_i are set to 1.0.

c) **Generative Adversarial Loss:**

The synthesis process is conditioned to inputs $x = \{I_M, M, S\}$. To allow the discriminator $\mathcal{D}$ to consider the conditions, despite the real/fake image I and $\hat{I}$, $\mathcal{D}$ will take x as well. The hinge loss for optimizing spectral normalized discriminator $\mathcal{D}$ is adopted as:

$$\mathcal{L}_{adv}^{D} = \mathbb{E}_{I,x}[\min(0, -1 + \mathcal{D}(I, x))] + \mathbb{E}_{\hat{I}_x,x}[\min(0, -1 - \mathcal{D}(\hat{I}, x))], \quad (4\text{-}16)$$

And the adversarial loss for the total network CoDE-GAN:

$$\mathcal{L}_{adv}^{G} = -\mathbb{E}_{\hat{I},x}\big[\mathcal{D}\big(\hat{I}, x\big)\big]. \quad (4\text{-}17)$$

The overall objectives are:

$$\mathcal{L} = \lambda_{\text{per}}\mathcal{L}_{\text{per}} + \lambda_{\ell 1}L_{\ell 1} + \lambda_c\mathcal{L}_c + \mathcal{L}_{adv}^{G}, \quad (4\text{-}18)$$

where λ_{per}, $\lambda_{\ell 1}$, λ_c, λ_{adv} denotes the coefficients for perceptual loss, reconstruction, content-aware loss and adversarial loss respectively.

4.4 Experiment Verification and Results Discussions

4.4.1 Data Preparation

This section introduces the datasets collection for evaluating the proposed methods and the collection methods for the required pre-processed data.

4.4.1.1 Dataset Collection

Two fashion garment datasets for simulating a real fashion editing scenario were

selected: one fashion human dataset for testing the methods' robustness in a more complex situation, and one outdoor church dataset for determining its generalizability.

There are 9,636 upper garments in the Garment Dataset collected by Chen et al. (2020). The Garment Dataset mainly collected garments with different collar and sleeve types. The another dataset is Cafi-Garment Dataset collected by Zhou et al. (2019). There are 17,075 garments with 77 categories that include dresses, jeans, and T-shirts, etc.

For the fashion human dataset, the ATR dataset (Liang et al., 2015) was selected, comprising 7, 700 human images with different poses in the wild. The outdoor church dataset is a subset from LSUN dataset (Yu et al., 2015) that there are 126,227 images. These aforementioned datasets were split into train set and valid set with an 8: 2 ratio.

4.4.1.2 Sketch Generation

The sketch is the vital information that guide the model to synthesis user-controlled garments. Generally, the sketch should be drawn manually to reflect the users' intuitions. However, collecting sketch maps may be time-consuming and costly, which goes against the intended motivation. For reducing human workload and achieve a robust response to the sketch maps, the results of edge detection are used for simulating the manually drawn sketches. Figure 4-6 shows several level outputs of the detected edges by HED (Xie & Tu, 2015). HED is a benchmark work for edge detection and is capable

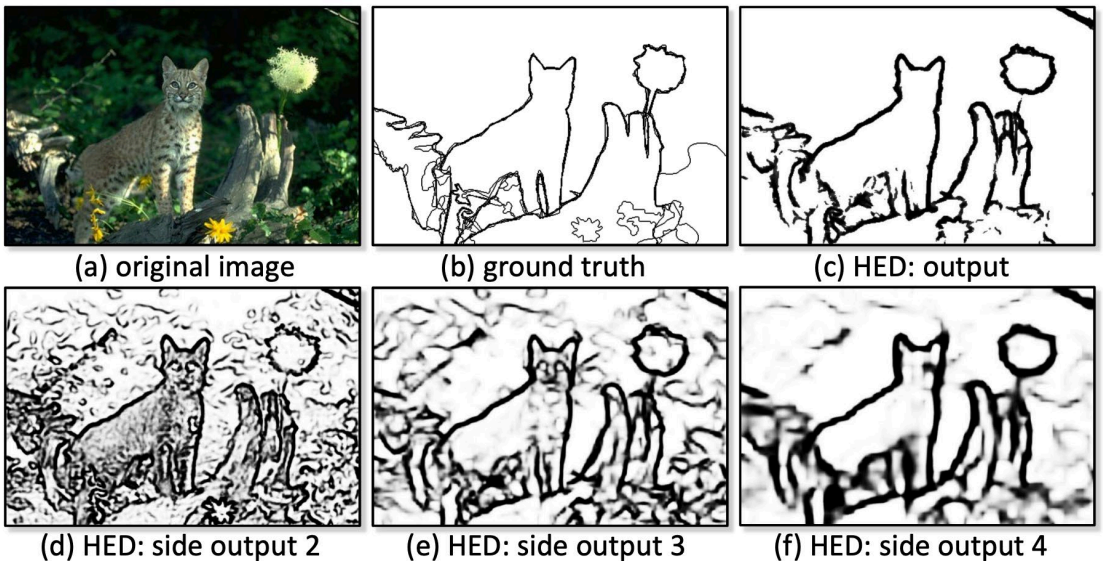
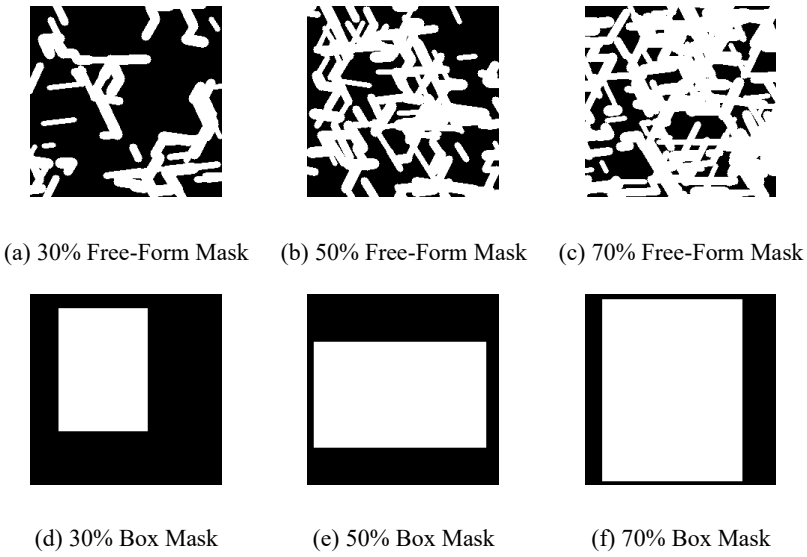


Figure 4-6 Edges Detected by HED (Xie & Tu, 2015).

of achieving promising edge maps. To ensure that the extracted edges faithfully simulate user hand-drawn sketches, these edges are binarized using a threshold of 127, which is the middle value of RGB pixels. Since the HED model mainly responds to the outline of an object, the editing model trained on HED primarily edits the shape of the cloth. The proposed model could be extended to edit other minor attributes like accessories (e.g., pockets, buttons) if the edge map is replaced by another edge extraction method such as Canny (Bao et al., 2005).

4.4.1.3 Mask Generation

Fashion editing often requires the transformation of a large continuous region. For simulating this characteristic, box or rectangular mask strategy are adopted. The mask ratio is set to 30%, 50%, and 70% with respect to the whole image area.



(a) 30% Free-Form Mask (b) 50% Free-Form Mask (c) 70% Free-Form Mask

(d) 30% Box Mask (e) 50% Box Mask (f) 70% Box Mask

Figure 4-7 Free-Form and Box Masks with Different Ratios.

However, most of the sketch-guided image editing work mainly discuss the free-form mask which is randomly generated strokes. For a fair and more general comparison, the experiments also trained the model with free-form masks followed with SC-FEGAN (Jo & Park, 2019).

4.4.2 Evaluation metrics

The general requirements of the generated images are realistic and various. However, the measurement of the generated images could be subjective in most cases. For instance, it's hard to quantify realistic. Therefore, researchers apply more implicit way to acquire metric scores.

4.4.2.1 Fréchet Inception Distance

Inception scores utilize InceptionNet V3 which pre-trained on ImageNet dataset

(Salimans et al., 2016). By classifying the generated images, the InceptionNet will output a classification probability distribution. If the image is fidelity enough, there will be a higher score on a certain class. If the images are various, there will be lower information entropy.

$$IS(G) = exp(\mathbb{E}_{x\sim p_g} D_{KL}(p(y|x)||p(y))) \quad (4\text{-}1)$$

However, if the ImageNet dataset did not include the class of generated images, it apparently lacks the ability of classifying. What's more, outputting a classification prediction with high confidence did not require the image realistic as human artifacts. These two-character harms the validness of inception score.

Fréchet inception distance (FID) proposed by Heusel et al. (2017) utilized the InceptionNet V3 as well. Unlike the inception scores, it only considered the features. Let m be mean, c be covariant, and tr be trace of matrix, subscript g and d denote feature comes from generator or real data respectively:

$$FID = ||m_g - m_d|| + tr(c_g + c_d - 2(c_g c_d)^{1/2}) \quad (4\text{-}2)$$

Since FID only calculate the statistic value of feature, it is more plausible on measuring GAN's capability. Heusel et al. (2017) also pointed that there is a much stronger relationship between FID and image quality.

4.4.2.2 Structural Similarity

Structural Similarity (SSIM) (Wang et al., 2004) considers three aspects of image:

luminance, contrast, and structure. The subscript follows the definition of FID. And μ denotes mean value. σ denotes variance. c_1, c_2, c_3 are three different constant values.

• Luminance:

$$l(x_g, x_d) = \frac{2\mu_d\mu_g + c_1}{\mu_d^2 + \mu_g^2 + c_1} \tag{4-3}$$

• Contrast:

$$c(x_g, x_d) = \frac{2\sigma_d\sigma + c_2}{\sigma^2 + \sigma^2 + c_2} \tag{4-4}$$

• Structure:

$$s(x_g, x_d) = \frac{2\sigma_{dg} + c_3}{\sigma_d\sigma_g^2 + c_3} \tag{4-5}$$

And the total SSIM is:

$$SSIM(x_g, x_d) = l(x_g, x_d)c(x_g, x_d)s(x_g, x_d) \tag{4-6}$$

SSIM is score which compare two images. It usually is applied in image completion tasks.

4.4.2.3 Peak Signal-to-Noise Ratio

Similar to SSIM, Peak Signal-to-Noise Ratio (PSNR) (Faragallah et al., 2020) is the metric for measuring two images. Firstly, it calculates the mean square error of generated image and real image.

$$MSE(x_g, x_d) = \frac{1}{mn}\sum_{i=0}^{m-1}\sum_{j=0}^{n-1}[x_g(i,j) - x_d(i,j)]^2 \tag{4-7}$$

The PSNR as follows:

$$PSNR(x_g, x_d) = 10log_{10}\frac{MAX_g^2}{MSE(x_g, x_d)} \tag{4-8}$$

4.4.3 Results Discussions

The experiments were conducted on the data prepared in Section 4.4.1 with metrics mentioned in Section 4.4.2. It is cost to collect real edited data for evaluating the methods. Therefore, the mask-reconstruction proxy task was chosen to evaluate the quantitative metrics.

4.4.3.1 Quantitative Evaluation

A comparison was initially made between the proposed methods and Pix2Pix (Isola et al., 2017), and SC-FEGAN (Jo & Park, 2019). Table 4-1 and

Table *4-2* shows the overall comparisons under the training of box mask with 30% ratio.

Table 4-1 Comparisons in Garment-Based Dataset

Metrics	Garment Dataset			Cafi-Garment Dataset		
	Pix2Pix	SC-FEGAN	Ours	Pix2Pix	SC-FEGAN	Ours
FID ↓	9.0279	7.7746	**5.0172**	21.2413	22.0515	**13.6705**
SSIM ↑	0.8569	0.8618	**0.8882**	0.8968	0.906	**0.9162**
PSNR ↑	24.3152	24.1064	**26.1712**	28.4381	28.6801	**30.5279**

Table 4-2 Comparisons in Fashion Human and Outdoor Buildings Dataset

Metrics	ATR Dataset			LSUN Outdoor Church Dataset		
	Pix2Pix	SC-FEGAN	Ours	Pix2Pix	SC-FEGAN	Ours
FID ↓	73.0056	69.7000	**43.8257**	40.2195	39.3383	**30.6956**
SSIM ↑	0.7260	0.8200	**0.8596**	0.7025	0.7864	**0.8089**
PSNR ↑	20.2658	20.5500	24.7131	**20.0976**	18.9277	19.9848

The results show that the proposed methods outperformed not only in garment-based

dataset but also have the ability to generalize in more complex dataset. The methods employed achieved the best performance in FID and SSIM Metrics. There is only a slightly slack in PSNR when compared with Pix2Pix in LSUN Outdoor Church Dataset. PSNR evaluates peak signal noise rate with L_2 distance. Even though Pix2Pix is better in PSNR, it failed a lot in FID metrics which have a stronger relation with visual perceptual quality.

Additionally, significant results were achieved not only with the box mask but also with free-form mask with respect to larger mask ratio. This character was evaluated in Garment Dataset with two additional image inpainting methods Partial Convolution (Liu et al., 2018) and DeepFill V2 (J. Yu et al., 2019), which is designed to recover irregular free-form masked region. Table 4-5 present the overall comparisons on Garment Dataset with different training mask types and ratios. The proposed method is robustness in handling various masks. Especially, when the mask ratio increases to 70%, the model performs much better than the other network for implementing the reconstruction task.

Table 4-3 Evaluation on Garment Dataset with 30% Masked Region

Metrics	Free-Form Mask					Box Mask				
	Pix2Pix	Partial Conv	DeepFill V2	SC-FEGAN	Ours	Pix2Pix	Partial Conv	DeepFill V2	SC-FEGAN	Ours
FID↓	9.028	35.596	6.891	7.775	**5.017**	5.841	19.546	3.762	5.509	**2.789**
SSIM↑	0.857	0.750	0.873	0.862	**0.888**	0.901	0.854	0.931	0.918	**0.939**
PSNR↑	24.315	16.052	24.689	24.106	**26.171**	27.269	23.640	29.277	28.236	**30.617**

Table 4-4 Evaluation on Garment Dataset with 50% Masked Region

Metrics	Free-Form Mask					Box Mask				
	Pix2Pix	Partial Conv	DeepFill V2	SC-FEGAN	Ours	Pix2Pix	Partial Conv	DeepFill V2	SC-FEGAN	Ours
FID↓	7.581	62.314	14.339	9.484	**4.764**	18.451	139.845	12.092	14.138	**8.359**
SSIM↑	0.853	0.758	0.882	0.856	**0.894**	0.770	0.574	0.786	0.767	**0.816**
PSNR↑	25.561	20.661	26.693	25.391	**27.990**	21.320	13.032	21.666	21.126	**23.631**

Table 4-5 Evaluation on Garment Dataset with 70% Masked Region

Metrics	Free-Form Mask					Box Mask				
	Pix2Pix	Partial Conv	DeepFill V2	SC-FEGAN	Ours	Pix2Pix	Partial Conv	DeepFill V2	SC-FEGAN	Ours
FID↓	11.467	160.080	20.808	13.826	**7.102**	27.479	258.358	15.903	21.114	**11.547**
SSIM↑	0.796	0.597	0.824	0.787	**0.839**	0.708	0.417	0.718	0.686	**0.752**
PSNR↑	23.601	17.406	24.592	23.143	**25.727**	19.508	10.476	20.114	18.946	**21.906**

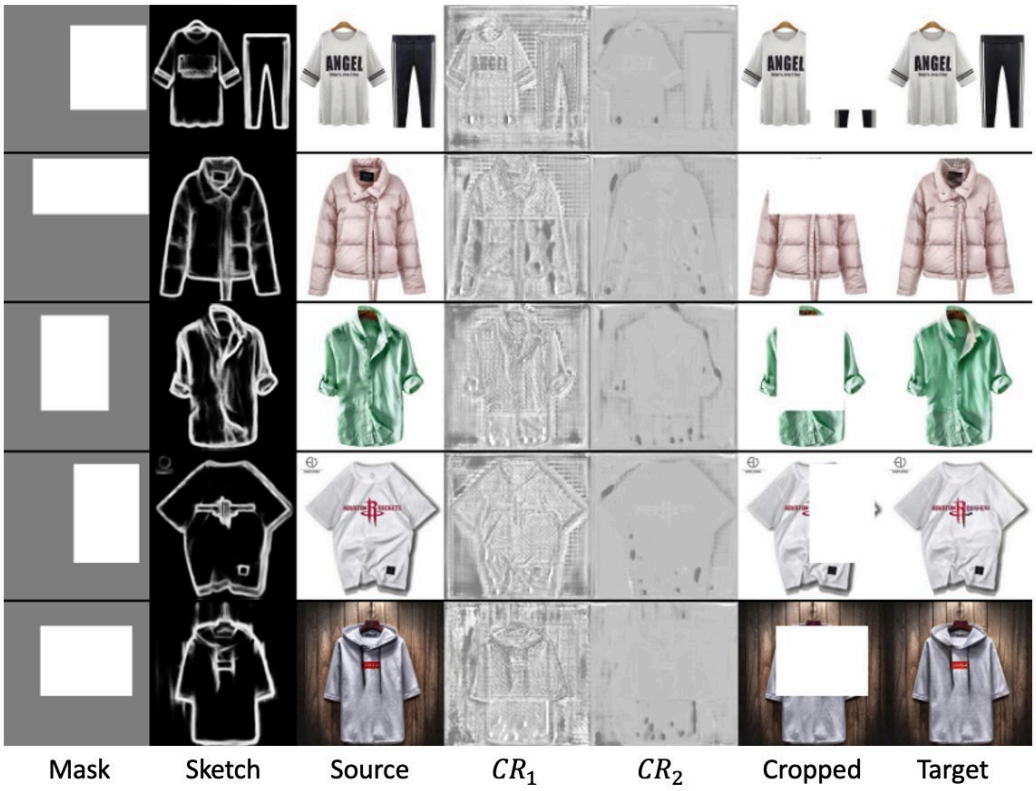


Figure 4-8 Qualitative Results on Reconstruction Task.

Figure 4-9 Qualitative Results on Sleeves & Collars Editing.

4.4.3.2 Qualitative Evaluation

Figure 4-8 shows the qualitative results while the model is training in recovering the masked region with the guidance of sketches. CR_1 and CR_2 are the content response map proposed in Section 4.3.4.

Figure 4-9 shows the qualitative results while editing a short sleeve to long sleeve. Even though the other method could synthesize the increased sleeves region and has clear boundary in respect to the sketch. They failed in filling texture in the content region. Appendix B exhibits more challenging edited results showcased in Figure B-1. Additionally, this study includes an interactive UI for editing fashion images. A video demonstration is accessible through the QR code provided in Figure B-2.

In conclusion, the proposed flexible editing model performs well in editing tops and bottoms, as these categories present relatively simple textures. However, when editing whole-body dresses with complex patterns, the model is less effective in reconstructing the texture.

4.4.3.3 Ablation Study

To demonstrate the effectiveness of the designed modules, an ablation study was conducted while training with free-form masks with 70% ratios. Table 4-6 and Table 4-7 showed the evaluation results in various mask types. The *wo grey* refers to remove the grey image in sketch encoding branch. The *single branch* refers to keep one gated convolution-based encoding branch for coding the sketch and source image. The *wo enhancement* refers to the removal of the specially designed content enhancement block. Each of these designed modules brings significant improvement on the qualitative metrics.

Table 4-6 Ablation Study on Free-Form Mask

Ablation	30%			50%			70%		
	FID↓	SSIM↑	PSNR↑	FID↓	SSIM↑	PSNR↑	FID↓	SSIM↑	PSNR↑
wo Grey	9.869	0.904	26.655	9.892	0.858	25.303	11.169	0.804	23.903
wo Segmentation	37.597	0.824	19.936	54.641	0.732	18.349	72.130	0.632	16.762
Single Branch	15.269	0.890	24.341	21.350	0.830	22.304	30.599	0.757	20.420
Whole Model	**6.269**	**0.921**	**28.189**	**6.148**	**0.885**	**27.136**	**7.102**	**0.839**	**25.727**

Table 4-7 Ablation Study on Box Mask

Ablation	30%			50%			70%		
	FID↓	SSIM↑	PSNR↑	FID↓	SSIM↑	PSNR↑	FID↓	SSIM↑	PSNR↑
wo Grey	19.799	0.835	20.127	26.958	0.731	18.072	29.716	0.642	16.524
wo Segmentation	32.603	0.777	15.548	57.907	0.632	13.528	92.449	0.488	12.225

Single Branch	21.627	0.817	19.260	41.615	0.695	16.696	79.753	0.580	14.174
Whole Model	**12.224**	**0.856**	**21.790**	**17.009**	**0.777**	**19.386**	**25.946**	**0.678**	**15.629**

The most notorious improvement is brought by the content enhancement block. It is the significant factor to improve the generalizability in various mask types.

4.5 Chapter Summary

This chapter introduces CoDE-GAN, a flexible editing model for fashion image content. By elaborately designing a reconstruction proxy task, CoDE-GAN first decouple the content of an image into structure and texture representations. Particularly, the structure representation, obtained through edge detection, enables an automatic pipeline for implementing this approach. By training to reconstruct an image through these decoupled conditions, sketch condition and texture condition, the model can effectively edit an image's content, even with out of distribution samples.

Furthermore, extensive experiments were conducted to validate the performance. The model was examined using the human ATR dataset and the garment-centric Garment and CafiGarment datasets, revealing that CoDE-GAN delivers superior performance in perceptual quality and editing flexibility when compared to existing state-of-the-art methods. This highlights its potential to significantly streamline image editing processes in the fashion industry. Beyond achieving the perceptual quality, CoDE-GAN also shows significant potential for adaptation in other applications, such as image inpainting or guided image reconstruction.

Nevertheless, there are limitations to further improving the proposed CoDE-GAN. This method mostly edits the shape of image content. Incorporating CoDE-GAN with style conditions presents a challenging aspect. Moreover, this system combines two distinct modules. It would be worthwhile to integrate CoDE-GAN into the previously discussed SGDiff to achieve both generation and editing in one unified model.

Chapter 5. CONCLUSIONS AND RECOMMENDATIONS FOR FUTURE WORK

5.1 Conclusions

This thesis presents a comprehensive exploration of fashion image generation and editing, introducing two distinct models: SGDiff and CoDE-GAN for controllable generation and flexible editing, respectively. SGDiff serves as a controllable generation model, enabling the creation of fashion images with a particular focus on texture control. CoDE-GAN, on the other hand, acts as a flexible editing model, efficiently modify cloth content in existing images. The two models are designed to work either independently or integratively as one single system.

A key innovation of this thesis is the use of decoupled conditions in both modules, significantly reducing the reliance on labeled training data for controllable image generation and editing. The significance of decoupled conditions extends to the broader field of image generative models. Utilizing a self-supervised reconstruction pipeline, the system effectively leverages various decoupled conditions, including sketches, text, and textures. This enables the system to mimic real user inputs and achieve high-fidelity image reconstructions.

SGDiff, as introduced in Chapter 3, presents advanced style transfer in the image

generation process. It introduces a novel approach by incorporating an image-based condition as style reference, leading to enhanced control over the synthesized textures. This model effectively utilizes the concept of decoupled conditions to reconstruct from randomly masked image patches, fine-tuning the pre-trained text-to-image diffusion model. The efficacy of SGDiff is underscored by its superior performance in LPIPS and FID scores, demonstrating its capability in synthesizing fashion images that closely reflect the given style reference. By extending the text-to-image diffusion model to include additional image-based inputs, SGDiff represents a significant step forward in image generation technology.

Chapter 4 introduces CoDE-GAN, the flexible editing model, which stands as an effective tool in fashion image editing. This model circumvents timing issues commonly associated with diffusion models by decoupling image content from texture and spatial representations through the decoupled sketch condition. This innovative approach effectively addresses challenges in content area construction, demonstrating superior performance in terms of perceptual quality and editing flexibility. The extensive experiments show its superior performance with other state-of-the-art method, showcasing its potential applications not only in the fashion industry but also in broader domain such as guided image reconstruction.

In summary, this thesis has not only contributed novel methodologies and tools in the

realm of digital fashion image generation and editing but also set a foundation for future research in this rapidly evolving field. The potential applications of SGDiff and CoDE-GAN extend far beyond their current scope, promising to revolutionize the way fashion imagery is approached and interacted in the digital age.

5.2 Recommendations for Future Work

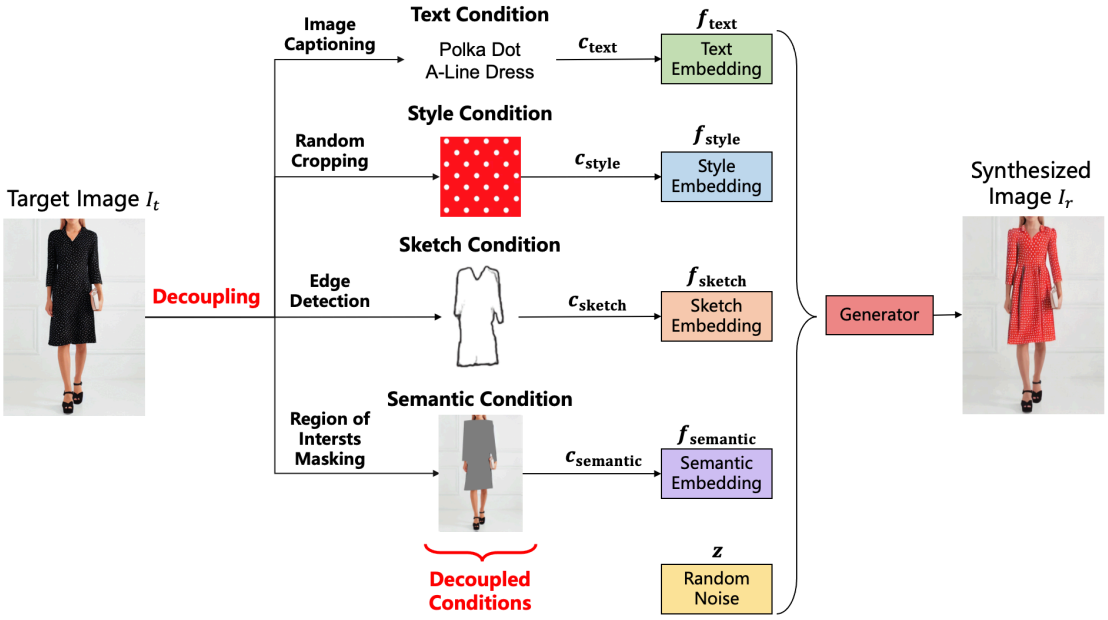


Figure 5-1 Overview of the Unified Generation and Editing System Using Decoupled Conditions.

Although Chapter 3 and Chapter 4 implemented two independent models for conducting fashion image generation and editing tasks respectively, merging these two distinct models into a unified one is both possible and beneficial. Figure 5-1 demonstrates this unified model. By utilizing decoupled conditions, the model can swap the original black polka dot with red ones, resulting in the generation of a red polka dot dress. This allows for editing the original image by trying on this newly generated red

dress.

To develop this unified system, it important to decouple multiple conditions effectively and simultaneously. These conditions represent partial information extracted from the target image with automatic preprocesses. For instance, a masked image could serve as semantic condition, as the model is required to edit the masked region to align semantically with the rest of image. In this method, partial information is carefully selected to mimic real user inputs. This approach enables the model to generate or edit images that accurately reflect these inputs, treating this information as decoupled conditions.

A key aspect of successfully decoupling an image into various conditions involves designing an appropriate reconstruction pipeline. This task, conducted in a self-supervised learning framework, hinges on the nature of the decoupled conditions. Several principles guide the selection of automatic preprocessing methods. Firstly, the preprocess must be fully automated, requiring no human intervention to avoid the need for labor-intensive and costly manual labeling. The information derived should encompass only a random selection of the target image's details. By reconstructing from this randomly selected information, the model can better adapt to actual usage scenarios and prevent overfitting. Additionally, that information should be compatible with real user inputs, such as sketch maps, text descriptions, texture maps, etc.

In detail, the conditions represented in Figure 5-1 can be achieved as follows. Given a target image, I_t, the decoupled conditions are achieved through various automatic methods. The sketch condition, c_{sketch}, is derived using an edge detection method (Xie & Tu, 2015), while the text condition, c_{text}, comes from image captioning work (Li et al., 2022) or is automatically sourced from the Internet. Style condition c_{style} is generated by randomly cropping a patch from the foreground image, obtainable through salient object segmentation work (Qin et al., 2019). The semantic condition, c_{semantic}, provides the image's content, sourced through random masking or human parsing techniques (Gong et al., 2018; Liang et al., 2015; Ruan et al., 2019), to identify areas of interest like clothing. As illustrated above, these decoupled conditions can be efficiently achieved using existing tools or algorithms, eliminating the need for extensive labeling.

Here are two major challenges for this unified model, which are further detailed in sub-Sections 5.2.1 and 5.2.2.

5.2.1 Multi-Modal Inputs and Representations

A unified generation and editing model may involve multi-modality inputs, as depicted in Figure 5-1. Designing or utilizing the pre-trained modality-specific encoder presents a challenge. This is because data from different modalities may contribute differently

to the overall generation or editing process. Aligning these distinct feature representations into a unified space is another challenge.

5.2.2 Visual Characteristics Preserve

Currently, the existing diffusion models utilize a noise estimation loss function for denoising images. This loss function supervises the overall image reconstruction. It lacks effective supervision on specific visual characteristics. Traditionally, GANs achieve this by explicitly applying a perceptual loss to attain better visual consistency. This approach is less effective in diffusion models, as it requires an extra step to convert a predicted latent code into a real image. This could cost much time and memory.

Appendix A. More Qualitative Results of SGDiff Generated Garments

Figure A-1 More Qualitative Results of SGDiff Generated Garments (1).

Figure A-2 More Qualitative Results of SGDiff Generated Garments (2).

Appendix B. More Qualitative Results of CoDE-GAN Edited Samples

Figure B-1 More Flexible Edited Results of CoDE-GAN.

Figure B-2 Interactive UI of CoDE-GAN.

REFERENCES

Arjovsky, M., Chintala, S., & Bottou, L. (2017). Wasserstein Generative Adversarial Networks. In *International Conference on Machine Learning* (pp. 214--223). PMLR.

Bao, P., Zhang, L., & Wu, X. (2005). Canny Edge Detection Enhancement by Scale Multiplication. *IEEE transactions on pattern analysis and machine intelligence*, *27*(9), 1485-1490.

Bottou, L. (2012). Stochastic Gradient Descent Tricks. In *Neural Networks: Tricks of the Trade* (pp. 421-436). Springer.

Brock, A., Donahue, J., & Simonyan, K. (2019). Large Scale Gan Training for High Fidelity Natural Image Synthesis. In *International Conference on Learning Representations*.

Brooks, T., Holynski, A., & Efros, A. A. (2023). Instructpix2pix: Learning to Follow Image Editing Instructions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 18392--18402).

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., & others. (2020). Language Models Are Few-Shot Learners. *Advances in Neural Information Processing Systems*, *33*, 1877-1901.

Chen, J., Shen, Y., Gao, J., Liu, J., & Liu, X. (2018). Language-Based Image Editing with Recurrent Attentive Models. In *Proceedings of the IEEE Conference on*

Computer Vision and Pattern Recognition (pp. 8721-8729).

Chen, L., Tian, J., Li, G., Wu, C.-H., King, E.-K., Chen, K.-T., Hsieh, S.-H., & Xu, C. (2020). Tailorgan: Making User-Defined Fashion Designs. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision* (pp. 3241-3250).

Chen, X., Duan, Y., Houthooft, R., Schulman, J., Sutskever, I., & Abbeel, P. (2016). Infogan: Interpretable Representation Learning by Information Maximizing Generative Adversarial Nets. *Advances in neural information processing systems*, *29*.

Cheng, W.-H., Song, S., Chen, C.-Y., Hidayati, S. C., & Liu, J. (2021). Fashion Meets Computer Vision: A Survey. *ACM Comput. Surv.*, *54*(4), Article 72. https://doi.org/10.1145/3447239

Creswell, A., White, T., Dumoulin, V., Arulkumaran, K., Sengupta, B., & Bharath, A. A. (2018). Generative Adversarial Networks: An Overview. *IEEE Signal Processing Magazine*, *35*(1), 53-65.

Crowson, K., Biderman, S., Kornis, D., Stander, D., Hallahan, E., Castricato, L., & Raff, E. (2022). Vqgan-Clip: Open Domain Image Generation And editing with Natural Language Guidance. In *European Conference on Computer Vision* (pp. 88-105). Cham: Springer Nature Switzerland.

Cui, A., McKee, D., & Lazebnik, S. (2021). Dressing in Order: Recurrent Person Image Generation for Pose Transfer, Virtual Try-on and Outfit Editing. In *Proceedings*

of the IEEE/CVF International Conference on Computer Vision (pp. 14638-14647).

Dai, Q., Yang, S., Wang, W., Xiang, W., & Liu, J. (2021). Edit Like a Designer: Modeling Design Workflows for Unaligned Fashion Editing. In *Proceedings of the 29th ACM International Conference on Multimedia* (pp. 3492-3500).

Davenport, T. H. (2018). From Analytics to Artificial Intelligence. *Journal of Business Analytics*, *1*(2), 73-80.

Dhariwal, P., & Nichol, A. Q. (2021). Diffusion Models Beat Gans on Image Synthesis. In *Advances in Neural Information Processing Systems* (pp. 8780--8794).

Ding, Y., Mok, P. Y., Ma, Y., & Bin, Y. (2023). Personalized Fashion Outfit Generation with User Coordination Preference Learning. *Information Processing \& Management*, *60*(5), 103434.

Dong, H., Liang, X., Zhang, Y., Zhang, X., Shen, X., Xie, Z., Wu, B., & Yin, J. (2020). Fashion Editing with Adversarial Parsing Learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 8120-8128).

Dubey, S. R. (2021). A Decade Survey of Content Based Image Retrieval Using Deep Learning. *IEEE Transactions on Circuits and Systems for Video Technology*, *32*(5), 2687-2704.

Dumoulin, V., & Visin, F. (2016). A Guide to Convolution Arithmetic for Deep Learning. *ArXiv e-prints*.

Esmaeilpour, S., Liu, B., Robertson, E., & Shu, L. (2022). Zero-Shot out-of-Distribution Detection Based on the Pre-Trained Model Clip. In *Proceedings of the AAAI conference on artificial intelligence* (pp. 6568-6576).

Fan, Q., Wu, W., & Zurada, J. M. (2016). Convergence of Batch Gradient Learning with Smoothing Regularization and Adaptive Momentum for Neural Networks. *SpringerPlus*, *5*(1), 1-17.

Faragallah, O. S., El-Hoseny, H., El-Shafai, W., Abd El-Rahman, W., El-Sayed, H. S., El-Rabaie, E.-S. M., Abd El-Samie, F. E., & Geweid, G. G. N. (2020). A Comprehensive Survey Analysis for Present Solutions of Medical Image Fusion and Future Directions. *IEEE Access*, *9*, 11358-11371.

Fitch, F. B. (1944). Warren S. Mcculloch and Walter Pitts. A Logical Calculus of the Ideas Immanent in Nervous Activity. Bulletin of Mathematical Biophysics, Vol. 5 (1943), Pp. 115--133. *The Journal of Symbolic Logic*, *9*(2), 49-50.

Frühstück, A., Singh, K. K., Shechtman, E., Mitra, N. J., Wonka, P., & Lu, J. (2022). Insetgan for Full-Body Image Generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 7723-7732).

Gal, R., Alaluf, Y., Atzmon, Y., Patashnik, O., Bermano, A. H., Chechik, G., & Cohen-or, D. (2023). An Image Is Worth One Word: Personalizing Text-to-Image Generation Using Textual Inversion. In *The Eleventh International Conference on Learning Representations*.

Ge, Y., Song, Y., Zhang, R., Ge, C., Liu, W., & Luo, P. (2021). Parser-Free Virtual Try-

on Via Distilling Appearance Flows. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 8485-8493).

Gong, K., Liang, X., Li, Y., Chen, Y., Yang, M., & Lin, L. (2018). Instance-Level Human Parsing Via Part Grouping Network. In *Proceedings of the European conference on computer vision (ECCV)* (pp. 770-785).

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., & Bengio, Y. (2014). Generative Adversarial Nets. *Advances in neural information processing systems*, *27*.

Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., & Courville, A. (2017). Improved Training of Wasserstein Gans. *arXiv preprint arXiv:1704.00028*.

Han, X., Hu, X., Huang, W., & Scott, M. R. (2019). Clothflow: A Flow-Based Model for Clothed Person Generation. In *Proceedings of the IEEE/CVF international conference on computer vision* (pp. 10471-10480).

Han, X., Wu, Z., Jiang, Y.-G., & Davis, L. S. (2017). Learning Fashion Compatibility with Bidirectional Lstms. In *Proceedings of the 25th ACM international conference on Multimedia* (pp. 1078-1086).

Hertz, A., Mokady, R., Tenenbaum, J., Aberman, K., Pritch, Y., & Cohen-or, D. (2023). Prompt-to-Prompt Image Editing with Cross-Attention Control. In *The Eleventh International Conference on Learning Representations*.

Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., & Hochreiter, S. (2017). Gans Trained by a Two Time-Scale Update Rule Converge to a Local Nash

Equilibrium. In *Proceedings of the 31st International Conference on Neural Information Processing Systems* (pp. 6629–6640). Red Hook, NY, USA: Curran Associates Inc.

Ho, J., Jain, A., & Abbeel, P. (2020). Denoising Diffusion Probabilistic Models. *Advances in Neural Information Processing Systems*, *33*, 6840-6851.

Ho, J., & Salimans, T. (2021). Classifier-Free Diffusion Guidance. In *NeurIPS 2021 Workshop on Deep Generative Models and Downstream Applications*.

Hornik, K. (1991). Approximation Capabilities of Multilayer Feedforward Networks. *Neural Networks*, *4*(2), 251-257.

Hou, M., Wu, L., Chen, E., Li, Z., Zheng, V. W., & Liu, Q. (2019). *Explainable Fashion Recommendation: A Semantic Attribute Region Guided Approach* Proceedings of the 28th International Joint Conference on Artificial Intelligence, Macao, China.

Hu, B., Liu, P., Zheng, Z., & Ren, M. (2022). Spg-Vton: Semantic Prediction Guidance for Multi-Pose Virtual Try-On. *IEEE Transactions on Multimedia*, *24*, 1233-1246.

Isola, P., Zhu, J.-Y., Zhou, T., & Efros, A. A. (2017). Image-to-Image Translation with Conditional Adversarial Networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 1125-1134).

Jiang, S., Li, J., & Fu, Y. (2022). Deep Learning for Fashion Style Generation. *IEEE Transactions on Neural Networks and Learning Systems*, *33*(9), 4538-4550.

Jo, Y., & Park, J. (2019). Sc-Fegan: Face Editing Generative Adversarial Network with User's Sketch and Color. In *Proceedings of the IEEE/CVF international conference on computer vision* (pp. 1745-1753).

Johnson, J., Alahi, A., & Fei-Fei, L. (2016). Perceptual Losses for Real-Time Style Transfer and Super-Resolution. In *European Conference on Computer Vision* (pp. 694–711). Amsterdam, The Netherlands: Springer.

Jolicoeur-Martineau, A. (2019). The Relativistic Discriminator: A Key Element Missing from Standard Gan. In *International Conference on Learning Representations*.

Karnewar, A., & Wang, O. (2020). Msg-Gan: Multi-Scale Gradients for Generative Adversarial Networks. In *Proceedings of the IEEE/CVF Conference on Computer Vsion and Pattern Recognition* (pp. 7799-7808).

Karras, T., Aila, T., Laine, S., & Lehtinen, J. (2018). Progressive Growing of Gans for Improved Quality, Stability, and Variation. In *International Conference on Learning Representations*.

Karras, T., Laine, S., & Aila, T. (2019). A Style-Based Generator Architecture for Generative Adversarial Networks. In *Proceedings of the IEEE/CVF Conference on Computer Vsion and Pattern Recognition* (pp. 4401-4410).

Karras, T., Laine, S., Aittala, M., Hellsten, J., Lehtinen, J., & Aila, T. (2020). Analyzing and Improving the Image Quality of Stylegan. In *Proceedings of the IEEE/CVF Conference on Computer Vsion and Pattern Recognition* (pp. 8110-8119).

Kim, B.-K., Kim, G., & Lee, S.-Y. (2020). Style-Controlled Synthesis of Clothing Segments for Fashion Image Manipulation. *IEEE Transactions on Multimedia*, *22*(2), 298-310.

Kober, J., Bagnell, J. A., & Peters, J. (2013). Reinforcement Learning in Robotics: A Survey. *The International Journal of Robotics Research*, *32*(11), 1238-1274.

Korshunova, I., Shi, W., Dambre, J., & Theis, L. (2017). Fast Face-Swap Using Convolutional Neural Networks. In *Proceedings of the IEEE international conference on computer vision* (pp. 3677-3685).

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet Classification with Deep Convolutional Neural Networks. *Advances in Neural Information Processing Systems*, *25*, 1097-1105.

Kwon, Y., Petrangeli, S., Kim, D., Wang, H., Swaminathan, V., & Fuchs, H. (2022). Tailor Me: An Editing Network for Fashion Attribute Shape Manipulation. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision* (pp. 3831-3840).

Leng, Y., Chen, Z., Guo, J., Liu, H., Chen, J., Tan, X., Mandic, D., He, L., Li, X., Qin, T., & others. (2022). Binauralgrad: A Two-Stage Conditional Diffusion Probabilistic Model for Binaural Audio Synthesis. *Advances in Neural Information Processing Systems*, *35*, 23689-23700.

Lewis, K. M., Varadharajan, S., & Kemelmacher-Shlizerman, I. (2021). Tryongan: Body-Aware Try-on Via Layered Interpolation. *ACM Transactions on Graphics*

(TOG), *40*(4), 1-10.

Li, B., Qi, X., Lukasiewicz, T., & Torr, P. H. (2020). Manigan: Text-Guided Image Manipulation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 7880-7889).

Li, J., Li, D., Xiong, C., & Hoi, S. (2022). Blip: Bootstrapping Language-Image Pre-Training for Unified Vision-Language Understanding and Generation. In *International Conference on Machine Learning* (pp. 12888-12900). PMLR.

Li, P., Li, Y., Jiang, X., & Zhen, X. (2019). Two-Stream Multi-Task Network for Fashion Recognition. In *2019 IEEE international conference on image processing (ICIP)* (pp. 3038-3042). IEEE.

Li, Y., Gan, Z., Shen, Y., Liu, J., Cheng, Y., Wu, Y., Carin, L., Carlson, D., & Gao, J. (2019). Storygan: A Sequential Conditional Gan for Story Visualization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 6329-6338).

Li, Y., Li, Y., Lu, J., Shechtman, E., Lee, Y. J., & Singh, K. K. (2021). Collaging Class-Specific Gans for Semantic Image Synthesis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision* (pp. 14418-14427).

Liang, F., Wu, B., Dai, X., Li, K., Zhao, Y., Zhang, H., Zhang, P., Vajda, P., & Marculescu, D. (2022). Open-Vocabulary Semantic Segmentation with Mask-Adapted Clip. *arXiv preprint arXiv:2210.04150*.

Liang, X., Liu, S., Shen, X., Yang, J., Liu, L., Dong, J., Lin, L., & Yan, S. (2015). Deep

Human Parsing with Active Template Regression. *IEEE transactions on pattern analysis and machine intelligence*, *37*(12), 2402-2414.

Liu, G., Reda, F. A., Shih, K. J., Wang, T.-C., Tao, A., & Catanzaro, B. (2018). Image Inpainting for Irregular Holes Using Partial Convolutions. In *Proceedings of the European Conference on Computer Vision* (pp. 85–100).

Liu, H., Wan, Z., Huang, W., Song, Y., Han, X., Liao, J., Jiang, B., & Liu, W. (2021). Deflocnet: Deep Image Editing Via Flexible Low-Level Controls. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 10765-10774).

Liu, X., Gong, C., Wu, L., Zhang, S., Su, H., & Liu, Q. (2021). Fusedream: Training-Free Text-to-Image Generation with Improved Clip+ Gan Space Optimization. *arXiv preprint arXiv:2112.01573*.

Meng, C., He, Y., Song, Y., Song, J., Wu, J., Zhu, J.-Y., & Ermon, S. (2022). Sdedit: Guided Image Synthesis and Editing with Stochastic Differential Equations. In *International Conference on Learning Representations*.

Meng, Q., Zhao, S., Huang, Z., & Zhou, F. (2021). Magface: A Universal Representation for Face Recognition and Quality Assessment. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 14225-14234).

Metz, L., Poole, B., Pfau, D., & Sohl-Dickstein, J. (2017). Unrolled Generative Adversarial Networks. In *5th International Conference on Learning*

Representations, {ICLR} 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings. OpenReview.net.

Minsky, M., & Papert, S. A. (2017). *Perceptrons: An Introduction to Computational Geometry*. MIT press.

Miyato, T., Kataoka, T., Koyama, M., & Yoshida, Y. (2018). Spectral Normalization for Generative Adversarial Networks. In *6th International Conference on Learning Representations, {ICLR} 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., & Ostrovski, G. (2015). Human-Level Control through Deep Reinforcement Learning. *nature*, *518*(7540), 529-533.

Nazeri, K., Ng, E., Joseph, T., Qureshi, F., & Ebrahimi, M. (2019, Oct). Edgeconnect: Structure Guided Image Inpainting Using Edge Prediction. In *The IEEE International Conference on Computer Vision (ICCV) Workshops*.

Neuberger, A., Borenstein, E., Hilleli, B., Oks, E., & Alpert, S. (2020). Image Based Virtual Try-on Network from Unpaired Data. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 5184-5193).

Nichol, A. Q., & Dhariwal, P. (2021). Improved Denoising Diffusion Probabilistic Models. In *International Conference on Machine Learning* (pp. 8162-8171).

Nichol, A. Q., Dhariwal, P., Ramesh, A., Shyam, P., Mishkin, P., Mcgrew, B., Sutskever,

I., & Chen, M. (2022). Glide: Towards Photorealistic Image Generation and Editing with Text-Guided Diffusion Models. In *International Conference on Machine Learning* (pp. 16784-16804).

Odena, A. (2016). Semi-Supervised Learning with Generative Adversarial Networks. *arXiv preprint arXiv:1606.01583*.

Odena, A., Olah, C., & Shlens, J. (2017). Conditional Image Synthesis with Auxiliary Classifier Gans. In *International conference on machine learning* (pp. 2642-2651).

Park, T., Liu, M.-Y., Wang, T.-C., & Zhu, J.-Y. (2019). Semantic Image Synthesis with Spatially-Adaptive Normalization. In *Proceedings of the IEEE/CVF Conference on Computer Vsion and Pattern Recognition* (pp. 2337-2346).

Patashnik, O., Wu, Z., Shechtman, E., Cohen-Or, D., & Lischinski, D. (2021). Styleclip: Text-Driven Manipulation of Stylegan Imagery. In *Proceedings of the IEEE/CVF International Conference on Computer Vision* (pp. 2085-2094).

Peng, B., Fan, H., Wang, W., Dong, J., & Lyu, S. (2021). A Unified Framework for High Fidelity Face Swap and Expression Reenactment. *IEEE Transactions on Circuits and Systems for Video Technology*, *32*(6), 3673-3684.

Portenier, T., Hu, Q., Szab, A., Bigdeli, S. A., Favaro, P., & Zwicker, M. (2018). Faceshop: Deep Sketch-Based Face Image Editing. *ACM Trans. Graph.*, *37*(4). https://doi.org/10.1145/3197517.3201393

Qian, S., Lian, D., Zhao, B., Liu, T., Zhu, B., Li, H., & Gao, S. (2021). Kgdet: Keypoint-

Guided Fashion Detection. In *Proceedings of the AAAI Conference on Artificial Intelligence* (pp. 2449-2457).

Qin, X., Zhang, Z., Huang, C., Gao, C., Dehghan, M., & Jagersand, M. (2019). Basnet: Boundary-Aware Salient Object Detection. In *Proceedings of the IEEE/CVF Conference on Computer Vsion and Pattern Recognition* (pp. 7479-7489).

Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., & others. (2021). Learning Transferable Visual Models from Natural Language Supervision. In *International Conference on Machine Learning* (pp. 8748-8763).

Radford, A., Metz, L., & Chintala, S. (2015). Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks. *arXiv preprint arXiv:1511.06434*.

Ramesh, A., Pavlov, M., Goh, G., Gray, S., Voss, C., Radford, A., Chen, M., & Sutskever, I. (2021). Zero-Shot Text-to-Image Generation. In *International Conference on Machine Learning* (pp. 8821-8831).

Rombach, R., Blattmann, A., Lorenz, D., Esser, P., & Ommer, B. (2022). High-Resolution Image Synthesis with Latent Diffusion Models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 10684-10695).

Ruan, T., Liu, T., Huang, Z., Wei, Y., Wei, S., & Zhao, Y. (2019). Devil in the Details: Towards Accurate Single and Multiple Human Parsing. In *Proceedings of the*

AAAI conference on artificial intelligence (pp. 4814-4821).

Ruder, S. (2016). An Overview of Gradient Descent Optimization Algorithms. *arXiv preprint arXiv:1609.04747*.

Ruiz, N., Li, Y., Jampani, V., Pritch, Y., Rubinstein, M., & Aberman, K. (2023). Dreambooth: Fine Tuning Text-to-Image Diffusion Models for Subject-Driven Generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 22500-22510).

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning Representations by Back-Propagating Errors. *nature*, *323*(6088), 533-536.

Salimans, T., Goodfellow, I., Zaremba, W., Cheung, V., Radford, A., & Chen, X. (2016). Improved Techniques for Training Gans. In *Proceedings of the 30th International Conference on Neural Information Processing Systems* (pp. 2234–2242). Red Hook, NY, USA: Curran Associates Inc.

Schnfeld, E., Sushko, V., Zhang, D., Gall, J., Schiele, B., & Khoreva, A. (2021). You Only Need Adversarial Supervision for Semantic Image Synthesis. In *International Conference on Learning Representations*.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., & Klimov, O. (2017). Proximal Policy Optimization Algorithms. *arXiv preprint arXiv:1707.06347*.

Shi, H., Hayat, M., Wu, Y., & Cai, J. (2022). Proposalclip: Unsupervised Open-Category Object Proposal Generation Via Exploiting Clip Cues. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp.

9611-9620).

Simonyan, K., & Zisserman, A. (2014). Very Deep Convolutional Networks for Large-Scale Image Recognition. *arXiv preprint arXiv:1409.1556*.

Song, J., Meng, C., & Ermon, S. (2021). Denoising Diffusion Implicit Models. In *International Conference on Learning Representations*.

Song, Y., Durkan, C., Murray, I., & Ermon, S. (2021). Maximum Likelihood Training of Score-Based Diffusion Models. *Advances in Neural Information Processing Systems*, *34*, 1415-1428.

Su, H., Wang, P., Liu, L., Li, H., Li, Z., & Zhang, Y. (2020). Where to Look and How to Describe: Fashion Image Retrieval with an Attentional Heterogeneous Bilinear Network. *IEEE Transactions on Circuits and Systems for Video Technology*, *31*(8), 3254-3265.

Su, X., Song, J., Meng, C., & Ermon, S. (2023). Dual Diffusion Implicit Bridges for Image-to-Image Translation. In *The Eleventh International Conference on Learning Representations*.

Sutton, R. S., & Barto, A. G. (2018). *Reinforcement Learning: An Introduction*. MIT press.

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., & Wojna, Z. (2016). Rethinking the Inception Architecture for Computer Vision. In *Proceedings of the IEEE/CVF Conference on Computer Vsion and Pattern Recognition* (pp. 2818-2826).

Teng, Z., Duan, Y., Liu, Y., Zhang, B., & Fan, J. (2021). Global to Local: Clip-Lstm-

Based Object Detection from Remote Sensing Images. *IEEE Transactions on Geoscience and Remote Sensing*, *60*, 1-13.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L. u., & Polosukhin, I. (2017). Attention Is All You Need. In *Advances in Neural Information Processing Systems*. Curran Associates, Inc.

Wang, B., Zheng, H., Liang, X., Chen, Y., Lin, L., & Yang, M. (2018). Toward Characteristic-Preserving Image-Based Virtual Try-on Network. In *Proceedings of the European Conference on Computer Vision* (pp. 589–604).

Wang, Z., Bovik, A. C., Sheikh, H. R., & Simoncelli, E. P. (2004). Image Quality Assessment: From Error Visibility to Structural Similarity. *IEEE transactions on image processing*, *13*(4), 600-612.

Warde-Farley, D., & Goodfellow, I. (2016). 11 Adversarial Perturbations of Deep Neural Networks. *Perturbations, Optimization, and Statistics*, *311*, 5.

Xie, S., & Tu, Z. (2015). Holistically-Nested Edge Detection. In *Proceedings of the IEEE international conference on computer vision* (pp. 1395-1403).

Xu, J., Pu, Y., Nie, R., Xu, D., Zhao, Z., & Qian, W. (2021). Virtual Try-on Network with Attribute Transformation and Local Rendering. *IEEE Transactions on Multimedia*, *23*, 2222-2234.

Yang, H., Zhang, R., Guo, X., Liu, W., Zuo, W., & Luo, P. (2020). Towards Photo-Realistic Virtual Try-on by Adaptively Generating-Preserving Image Content. In *Proceedings of the IEEE/CVF Conference on Computer Vsion and Pattern*

Recognition (pp. 7850-7859).

Yang, S., Luo, P., Loy, C.-C., & Tang, X. (2016). Wider Face: A Face Detection Benchmark. In *Proceedings of the IEEE/CVF Conference on Computer Vsion and Pattern Recognition* (pp. 5525-5533).

Yang, X., Song, X., Feng, F., Wen, H., Duan, L.-Y., & Nie, L. (2021). Attribute-Wise Explainable Fashion Compatibility Modeling. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, *17*(1), 1-21.

Yu, C., Hu, Y., Chen, Y., & Zeng, B. (2019). Personalized Fashion Design. In *Proceedings of the IEEE/CVF International Conference on Computer Vision* (pp. 9046-9055).

Yu, F., Zhang, Y., Song, S., Seff, A., & Xiao, J. (2015). Lsun: Construction of a Large-Scale Image Dataset Using Deep Learning with Humans in the Loop. *arXiv preprint arXiv:1506.03365*.

Yu, J., Lin, Z., Yang, J., Shen, X., Lu, X., & Huang, T. S. (2019). Free-Form Image Inpainting with Gated Convolution. In *Proceedings of the IEEE/CVF International Conference on Computer Vision* (pp. 4471-4480).

Zhan, H., Lin, J., Ak, K. E., Shi, B., Duan, L.-Y., & Kot, A. C. (2021). $ a^ 3$-Fkg: Attentive Attribute-Aware Fashion Knowledge Graph for Outfit Preference Prediction. *IEEE Transactions on Multimedia*, *24*, 819-831.

Zhang, R., Isola, P., Efros, A. A., Shechtman, E., & Wang, O. (2018, June). The

Unreasonable Effectiveness of Deep Features as a Perceptual Metric. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 586–595).

Zhang, R., Zhang, W., Fang, R., Gao, P., Li, K., Dai, J., Qiao, Y., & Li, H. (2022). Tip-Adapter: Training-Free Adaption of Clip for Few-Shot Classification. In *European Conference on Computer Vision* (pp. 493-510). Cham: Springer.

Zhang, S., Song, Z., Cao, X., Zhang, H., & Zhou, J. (2019). Task-Aware Attention Model for Clothing Attribute Prediction. *IEEE Transactions on Circuits and Systems for Video Technology*, *30*(4), 1051-1064.

Zhang, X., Sha, Y., Kampffmeyer, M. C., Xie, Z., Jie, Z., Huang, C., Peng, J., & Liang, X. (2022). Armani: Part-Level Garment-Text Alignment for Unified Cross-Modal Fashion Design. In *Proceedings of the 30th ACM International Conference on Multimedia* (pp. 4525–4535). New York, NY, USA: Association for Computing Machinery.

Zhang, Y., Zhang, P., Yuan, C., & Wang, Z. (2020). Texture and Shape Biased Two-Stream Networks for Clothing Classification and Attribute Recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 13538-13547).

Zhou, C., Loy, C. C., & Dai, B. (2022). Extract Free Dense Labels from Clip. In *European Conference on Computer Vision* (pp. 696–712). Springer.

Zhou, D., Zhang, H., Li, Q., Ma, J., & Xu, X. (2022). Coutfitgan: Learning to

Synthesize Compatible Outfits Supervised by Silhouette Masks and Fashion Styles. *IEEE Transactions on Multimedia*, 1-15.

Zhou, W., Mok, P., Zhou, Y., Zhou, Y., Shen, J., Qu, Q., & Chau, K. (2019). Fashion Recommendations through Cross-Media Information Retrieval. *Journal of Visual Communication and Image Representation*, *61*, 112-120.

Zhou, Z., Zhang, B., Lei, Y., Liu, L., & Liu, Y. (2022). Zegclip: Towards Adapting Clip for Zero-Shot Semantic Segmentation. *arXiv preprint arXiv:2212.03588*.

Zhu, S., Urtasun, R., Fidler, S., Lin, D., & Change Loy, C. (2017). Be Your Own Prada: Fashion Synthesis with Structural Coherence. In *Proceedings of the IEEE International Conference on Computer Vision* (pp. 1680-1688).