Exploring images with deep learning for classification, retrieval and synthesis

Liu, Y.

Chapter 8

Applications of Image Synthesis

After classification and retrieval, in this chapter we turn to address the third research theme: *synthesis*. In particular, we focus on two practical applications: image-to-image translation and fashion style transfer.

Image-to-image translation between different domains aim to arbitrarily manipulate the source image content given a target one. For RQ7, we need to study what factors influence the performance of cycle-consistent generative networks (CycleGAN), which have become a fundamental approach for general-purpose image-to-image translation, while few work investigate the important factors within it. To this end, we present an extensive and empirical study on cycle-consistent generative networks. We exploit two extended models which can promote the generation quality. Then, we conduct comprehensive experiments to evaluate these models for several translation tasks.

As for fashion style transfer, we aim towards developing a novel approach to perform the problem of person-to-person clothing swapping (RQ8). It is challenging due to varying pose deformations between different person images. We address this challenge by proposing a novel multi-stage generative network (SwapGAN) that integrates three generators based on different synthesis conditions. The SwapGAN model is end-to-end trainable with adversarial loss and mask-consistency loss. We demonstrate the effectiveness of our approach through both quantitative and qualitative evaluations on the DeepFashion dataset. This work can serve as a benchmark for future research on this task.

Keywords

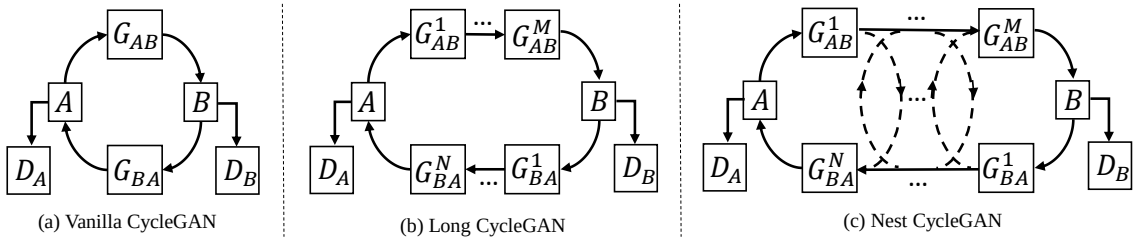
Image synthesis, Image-to-image translation, Fashion style transfer, Generative adversarial networks

8.1 Image-to-Image Translation

Image-to-image translation has achieved increasing attention in recent research. This task learns to synthesize the translated image in the target domain, given one image in the source domain. With the emergence of generative adversarial networks (GANs) [79] in recent years, some efforts have been made to employ unpaired image samples to model mapping functions between two different domains [89, 90, 91]. The translation task therefore becomes an unsupervised problem as the corresponding ground-truth images in the target domain are unknown. In addition, these approaches make use of the adversarial mechanism involved in GANs, to make the generated images undistinguished from real ones in the target domain. One challenging problem is that the domain mappings in these unsupervised approaches are under-constrained due to lack of ground-truth labels. To tackle the challenge, CycleGAN [94] introduces a cycle-consistency loss by reconstructing the generated image back to the source domain. In conjunction with the original adversarial loss, the cycle-consistency loss is beneficial to aid the unsupervised domain mappings. Moreover, this additional loss can help the model in avoiding mode collapse, from which the original GANs often suffer. Figure 8.1(a) illustrates the conceptual architecture of CycleGAN. Due to its high effectiveness and generalization ability, CycleGAN has been a fundamental model to address the task of unsupervised image translation, while few works have examined what factors may influence its performance. This fact motivates our research question **RQ 7: What factors will affect the performance of generative models on the translation tasks?**

Driven by this, in this work we extend the vanilla CycleGAN with new improvements, which can present more insights into what factors promote its performance on unsupervised image-to-image translation. Specifically, our improved models focus on studying the effects of two key factors in CycleGAN: one is the number of generators and another is the number of cycles. For the first factor, we build an extended model called **Long CycleGAN**, which can cascade more generators to perform the translation within a long cycle. For example, in Figure 8.1(b), we can incorporate M and N of different generators for A-to-B and B-to-A translation, respectively. Advantageously, the long cycle can leverage more generators to further increase the generation abilities of the model and improve the quality of the synthesized images. In terms of the second factor, another extended model with additional nested cycles is developed, namely **Nest CycleGAN**. As illustrated in Figure 8.1(c), this model attempts to exploit many inner cycles nested within the outer cycle. In this way, the inner cycles are able to directly connect the intermediate generators and provide more cycle-consistency losses to guide the domain mappings. Nest CycleGAN is used to demonstrate the benefit of adding more cycles among generators.

The contributions of this work are as follows:



(a) Vanilla CycleGAN (b) Long CycleGAN (c) Nest CycleGAN

Figure 8.1: Illustration of three cycle-consistent generative adversarial networks. Based on the (a) Vanilla CycleGAN [94], we build two extended models: (b) Long CycleGAN and (c) Nest CycleGAN. Long CycleGAN can promote the generative abilities by cascading more generators, and Nest CycleGAN is able to add extra inner cycles to enhance the mapping constraints.

- We propose two extended models to explore the important factors in CycleGAN. In addition, we present the initialization networks for the extended models. We conduct qualitative and quantitative evaluation to assess these models, for translation tasks including photo↔label and photo↔sketch.
- Our results witness the superiority of the extended models over the vanilla one. The results can act as an indication that CycleGAN equipped with more generators and cycles would achieve better generation quality.

The rest is structured as follows. Section 8.1.1 describes the vanilla CycleGAN and two extended models. The initialization networks are introduced in Section 8.1.2. The experiments are shown from Section 8.1.3 to Section 8.1.5.

8.1.1 Methodology

Problem Formulation

Assume that there are two unpaired image sets: $\{a_i\}_{i=1}^N$ in domain A and $\{b_j\}_{j=1}^M$ in domain B. The task aims to learn bi-directional mapping functions to map any $a_i \in A$ to $b_j \in B$, and vice versa. We omit the subscript i and j for notational simplicity. Notably, the images in the two sets are unaligned with each other, and the input images lack of ground-truth images to provide correct correspondences.

To tackle this problem, GANs [79] are used to generate realistic-looking target samples by incorporating a generator G and a discriminator D. Taking the A-to-B mapping for example, G_{AB} learns to simulate real images in domain B given the images in domain A. Then D_B need to distinguish real images b from synthetic images $G_{AB}(a)$. The original GANs compute the adversarial loss based on the negative log likelihood. Instead, we employ the least square loss designed in LSGAN [233], due to its proper stability of training and quality of generated images. The adversarial

loss for translating a to b is expressed with

$$\mathcal{L}_{GAN}(G_{AB}, D_B) = \mathbb{E}_{b\sim p_{data}(b)}[(D_B(b)-1)^2] + \mathbb{E}_{a\sim p_{data}(a)}[D_B(G_{AB}(a))^2]. \quad (8.1)$$

Here, p_{data} is the empirical distribution of training images. The generator and discriminator are trained for a minimax objective: $\min_{G_{AB}} \max_{D_B} \mathcal{L}(G_{AB}, D_B)$. Similarly, we can employ another generator and discriminator for the B-to-A mapping, and compute its corresponding adversarial loss: $\mathcal{L}_{GAN}(G_{BA}, D_A)$.

Vanilla Cycle-consistent GAN

Unsupervised image translation relies on adversarial loss to ensure the synthesized images in accordance with the target domain. However, it is important to add extra losses to enhance the constraints of unsupervised mapping functions. CycleGAN [94] develops a cycle-consistent loss by coupling two generators G_{AB} and G_{BA} in a reconstruction-based cycle. To be specific, the generated image $G_{AB}(a)$ is further fed into G_{BA} to obtain the reconstructed image $\hat{a} = G_{BA}(G_{AB}(a))$. Similarly, we can have $\hat{b} = G_{AB}(G_{BA}(b))$. Then, the difference between the input images and their reconstructed ones is computed with the L1 norm :

$$\begin{aligned}\mathcal{L}_{Rec}(G_{AB}, G_{BA}) =& \mathbb{E}_{a\sim p_{data}(a)}[||G_{BA}(G_{AB}(a)) - a||_1] \\ &+ \mathbb{E}_{b\sim p_{data}(b)}[||G_{AB}(G_{BA}(b)) - b||_1].\end{aligned} \quad (8.2)$$

Finally, the full objective in CycleGAN considers minimizing both the adversarial loss and the cycle-consistent loss:

$$\begin{aligned}\mathcal{L}_{Cycle}(G_{AB}, G_{BA}, D_A, D_B) =& \mathcal{L}_{GAN}(G_{AB}, D_B) + \mathcal{L}_{GAN}(G_{BA}, D_A) \\ &+ \lambda \mathcal{L}_{Rec}(G_{AB}, G_{BA}),\end{aligned} \quad (8.3)$$

where λ adjusts the weight of the reconstruction loss. As suggested in CycleGAN [94], the cycle-consistent constraint can help avoid the mode collapse problem, that is, the generated samples may only come from several modes of the real data distribution, but discard many other modes.

Long Cycle-consistent GAN

A key purpose of generative models is improving the quality of synthesized image samples. One favorable solution is introducing more generators to promote the generative abilities of the whole model. Driven by this, we extend CycleGAN by stacking a few generators, and investigate its effects on the generation quality. In Figure 8.1(b), we illustrate the first extended model called Long CycleGAN. Assume that there are M generators translating image samples from domain A to B, and at the same time N generators to map image samples from B to A. The whole mapping procedure can be performed in a chained fashion: the output of the current generator

is taken as input of the next generator. Formally, we can compute the output of each generator with

$$G_{AB}^{m}(a) = F(G_{AB}^{m-1}(a), W_{AB}^{m}), m = 1, \ldots, M, \tag{8.4}$$

$$G_{BA}^{n}(b) = H(G_{BA}^{n-1}(b), W_{BA}^{n}), n = 1, \ldots, N. \tag{8.5}$$

We define F and H as the mapping functions for A-to-B and B-to-A. W_{AB}^{m} and W_{BA}^{n} correspond to their mapping weights. Finally, we can rewrite the full objective for Long CycleGAN

$$\begin{aligned}\mathcal{L}_{Long}(\sum_{m=1}^{M} G_{AB}^{m}, \sum_{n=1}^{N} G_{BA}^{n}, D_A, D_B) =& \mathcal{L}_{GAN}(G_{AB}^{M}, D_B) + \mathcal{L}_{GAN}(G_{BA}^{N}, D_A) \\ &+ \lambda \mathcal{L}_{Rec}(G_{AB}^{M}, G_{BA}^{N}).\end{aligned} \tag{8.6}$$

We note that, when $M = N = 1$, $G_{AB}^{0}(a) = a$ and $G_{BA}^{0}(b) = b$. In this case, Long CycleGAN is the same as the vanilla one and therefore can be viewed as a generalized model.

Nest Cycle-consistent GAN

Furthermore, we present another extended model by nesting more inner cycles apart from a single outer cycle used in Long CycleGAN. T additional cycle-consistency losses based on new inner cycles can help constrain the mapping space between two domains. The extended model named by Nest CycleGAN is shown in Figure 8.1(c). On the one hand, the outer cycle in Nest CycleGAN (in solid line) performs the complete mappings between two domains by using all generators. On the other hand, the inner cycles (in dashed line) aim to build additional connections to bridge intermediate generators in the two chains. Notably, each inner cycle can be viewed as an auto-encoder model that can reconstruct the input image based on latent representations learned from intermediate generators. For instance, the m-th inner cycle for domain A is associated with two sets of generators, *i.e.* $\{G_{AB}^{1}, \ldots, G_{AB}^{m}\}$ and $\{G_{BA}^{N-m+1}, \ldots, G_{BA}^{N}\}$. In addition, we task the output of G_{AB}^{m} as input of G_{BA}^{N-m+1}, which can be denoted as

$$G_{BA}^{N-m+1}(G_{AB}^{m}(a)) = H(G_{AB}^{m}(a), W_{BA}^{N-m+1}). \tag{8.7}$$

After that, the image sample further passes from G_{BA}^{N-m+1} to G_{BA}^{N}, and the reconstructed image based on the m inner cycle can be formulated as

$$\hat{a}_m = G_{BA}^{N}(G_{AB}^{m}(a)). \tag{8.8}$$

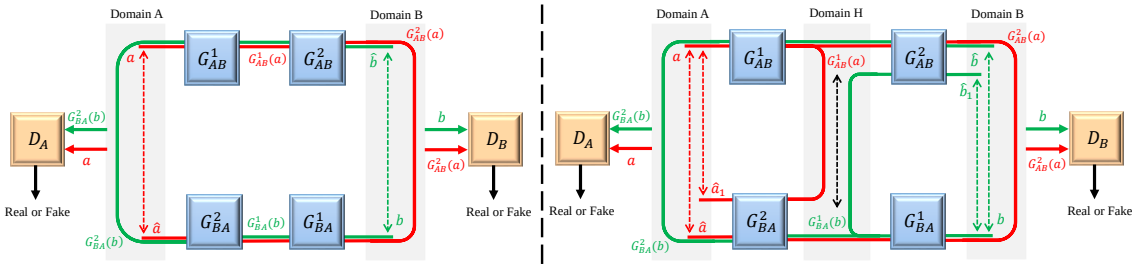


Figure 8.2: Instantiation Networks. Left: Long CycleGAN; Right: Nest CycleGAN. Details can be seen in Section 8.1.2.

Similarly, we can obtain $\hat{b}_n = G^M_{AB}(G^n_{BA}(b))$ for the n-th inner cycle with respect to domain B. Finally, the reconstruction loss with additional inner cycles is

$$\begin{aligned}\mathcal{L}_{Inner}(\sum_{m=1}^{M} G^m_{AB}, \sum_{n=1}^{N} G^n_{BA}) = &\sum_{m=1}^{M} \mathbb{E}_{a\sim p_{data}(a)}[||\hat{a}_m - a||_1] \\ &+ \sum_{n=1}^{N} \mathbb{E}_{b\sim p_{data}(b)}[||\hat{b}_n - b||_1].\end{aligned} \tag{8.9}$$

Particularly, when $m = M$ and $n = N$, the inner cycle turns to be the outer cycle, which can be included in the formulation. The objective of Nest CycleGAN is

$$\begin{aligned}\mathcal{L}_{Nest}(G^M_{AB}, G^N_{BA}, D_A, D_B) =&\mathcal{L}_{GAN}(G^M_{AB}, D_B) + \mathcal{L}_{GAN}(G^N_{BA}, D_A) \\ &+\lambda\mathcal{L}_{Inner}(\sum_{m=1}^{M} G^m_{AB}, \sum_{n=1}^{N} G^n_{BA}).\end{aligned} \tag{8.10}$$

8.1.2 Instantiation network

To assess the effectiveness of the three CycleGAN variants, we build their instantiation networks as follows.

Vanilla CycleGAN. We reproduce the standard CycleGAN with the generator and discriminator in [94]. (1) *Generator*: it consists of an encoder, several residual blocks and a decoder. The encoder module contains three convolutional layers; each residual block adds a skip connection on two 3×3 convolutional layers; the decoder module has two deconvolutional layers using stride-$\frac{1}{2}$ convolutions to upsample, and one stride-1 convolutional layer to output the synthesized image. The convolutional layers are followed by instance normalization [83] and ReLU [4]. (2) *Discriminator*: it is based on the Markovian network from PatchGANs [82, 88], which can run convolutationally across an image to classify if overlapping patches are real or fake. It contains four convolutional layers and the last layer produces a 1-dimensional feature map as the predicted output.

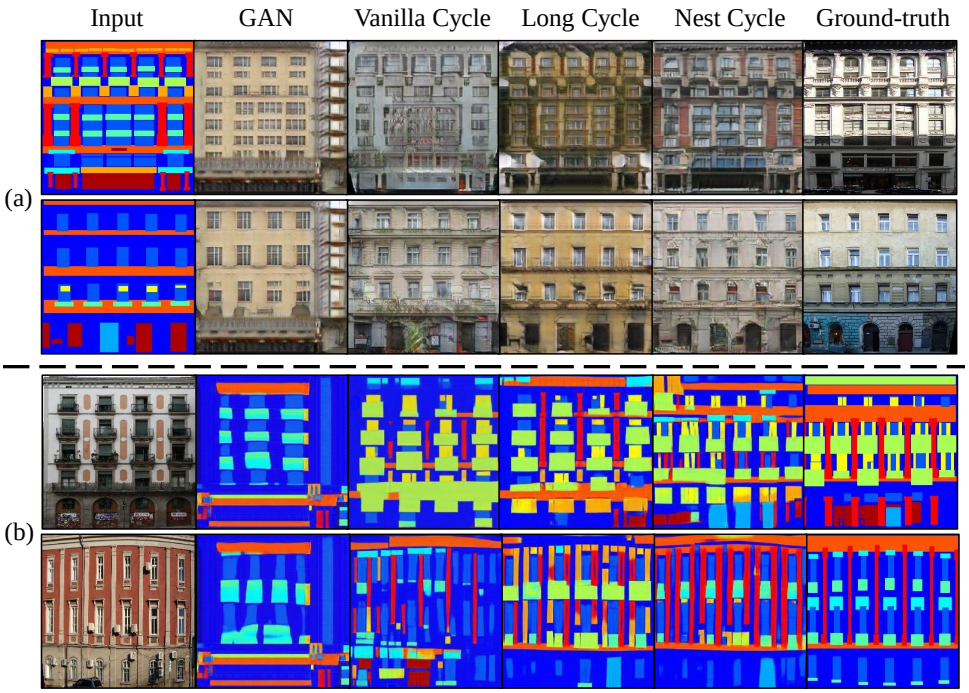


Figure 8.3: Generated samples of (a) the label→photo translation and (b) the photo→label translation evaluated on the CMP-Facade dataset.

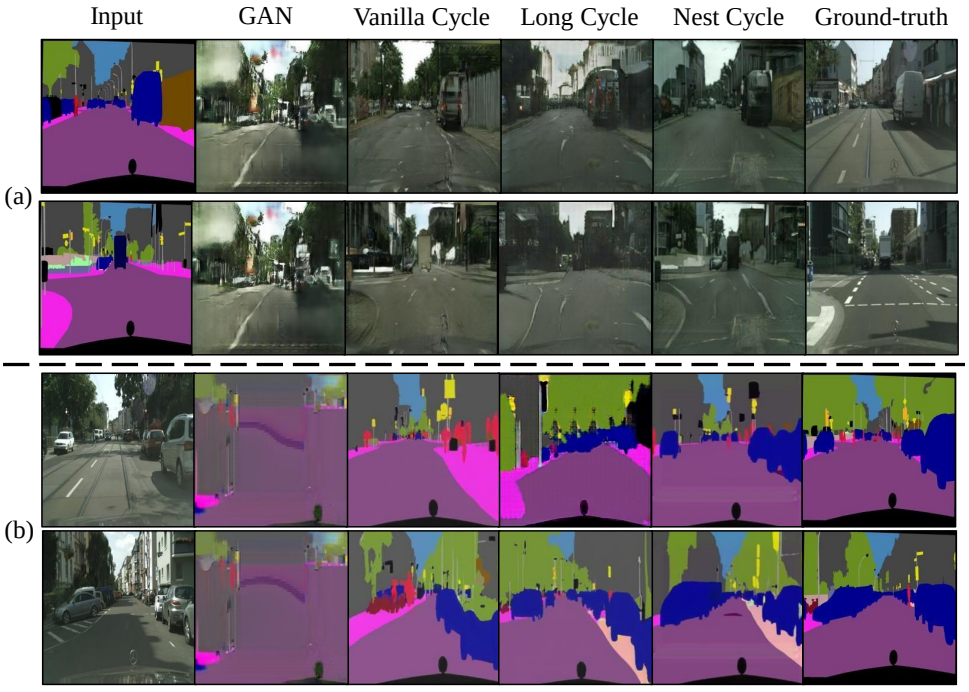


Figure 8.4: Qualitative results of (a) the label→photo translation and (b) the photo→label translation on the Cityscapes dataset.

Long CycleGAN. On top of the vanilla CycleGAN, we instantiate a Long CycleGAN by cascading two generators (*i.e.* $M = N = 2$), but the extension with more generators is straightforward. For fairness, all the generators and discriminators in

Long CycleGAN use the same networks with the vanilla CycleGAN. As illustrated in Figure 8.2 (Left), the model consists of two cycles which can be trained jointly. The red cycle starts with the input a in domain A and translates it to be $G^2_{AB}(a)$ in domain B. D_B learns to distinguish the fake image $G^2_{AB}(a)$ from the real image b. Then, $G^2_{AB}(a)$ is translated back to be the reconstructed image $\hat{a}$ in domain A. Likewise, the green cycle beginning from b performs an inverse translation.

Nest CycleGAN. Next, we build a Nest CycleGAN upon the above Long CycleGAN. In Figure 8.2 (Right), we exploit two additional inner cycles within the outer cycles. The inner cycles can also reconstruct the input images a and b, which are denoted by $\hat{a}_1$ and $\hat{b}_1$. We can see that, the first generated images, *i.e.* $G^1_{AB}(a)$ and $G^1_{BA}(b)$, act as intermediate states between A and B, then they should have implicit semantic similarities in some extend. Hence, we consider adding an extra loss to correlate them with

$$\mathcal{L}_{Sim}(G_{AB}, G_{BA}) = \mathbb{E}_{a\sim p_{data}(a)}[||G^1_{AB}(a) - G^1_{BA}(b)||_1]. \tag{8.11}$$

During training, $\mathcal{L}_{Sim}(G_{AB}, G_{BA})$ is added with $\mathcal{L}_{Nest}(G^M_{AB}, G^N_{BA}, D_A, D_B)$. Consequently, G^1_{AB} and G^1_{BA} can tend to gather in a common domain H between A and B, even though the inputs a and b are unpaired.

8.1.3 Experiment setup

To assess the three CycleGAN variants, we perform three image translation tasks, including photo↔label and photo↔sketch. The input and output images were scaled to 256×256. For fairness, some training parameters were consistent with CycleGAN [94], including mini-batch size of 1, learning rate of 0.0002, and weight decay of 0.0005. All the models were trained with 200 epoches and we fixed $\lambda = 10$ in the experiments, and optimized with the Adam optimizer [234]. Notice that, we randomly shuffled two domain-specific datasets to make sure they are totally unpaired. We implemented the models with TensorFlow [235] on a Titan X GPU card.

8.1.4 Results on photo↔label

For this translation task, we employed two semantic segmentation datasets: CMP-Facade [236] and Cityscapes [237]. CMP-Facade contains 606 images in total. We randomly select 400 images for training, and the remaining 206 images for testing. In Cityscapes, there are 2975 images for training and 500 images for testing. There are 12 and 19 semantic labels in CMP-Facade and Cityscapes, respectively.

Qualitative results. In Figure 8.3 and Figure 8.4, we compare the quality of generated images. For the label→photo task, three cycle-consistent GANs can synthesize more realistic images than the original GAN. It can be seen that, GAN suffers from

Table 8.1: Quantitative results of the label→photo translation evaluated on the CMP-Facade dataset. Higher numbers are better.

Method	CMP-Facade dataset			Cityscapes dataset		
	Per-pixel acc.	Per-class acc.	Class IOU	Per-pixel acc.	Per-class acc.	Class IOU
GAN	0.32	0.12	0.07	0.50	0.11	0.07
Vanilla CycleGAN	0.35	0.15	0.10	0.51	0.17	0.12
Long CycleGAN	0.43	0.19	0.13	0.54	0.18	0.13
Nest CycleGAN	0.49	0.22	0.15	0.57	0.20	0.14
Oracle	0.66	0.51	0.39	0.86	0.45	0.37

mode collapse, where the generated labels look almost identical for different input photos. However, the other three models can avoid this problem due to using cycle-consistency constraints. In addition, the two extended models can produce superior images over the vanilla one.

Quantitative results. In addition to the above qualitative evaluation, we further conduct quantitative experiments for this translation task. Considering the fact that the two datasets are not large scale, it is inappropriate to use the inception score (IS) to measure the generation quality. Instead, we used the FCN-score, *i.e.* a quantitative measurement as suggested in [88], to assess the label→photo task. First, a fully convolutional network (FCN) [26] for semantic segmentation was pre-trained using the real training photos and ground-truth labels. Then, each generated photo was fed into the FCN model to produce the predicted labels. The comparison with the ground labels can assess the generation photos. Commonly, FCN-score includes three standard metrics: per-pixel accuracy, per-class accuracy, and mean class intersection-over-union (IOU).

Table 8.1 reports quantitative results on CMP-Facade and Cityscapes. Comparably, all the three cycle-consistency models outperform the original GAN model. However, we can observe that the performance gap between Vanilla CycleGAN and GAN is not significant, while Long CycleGAN can improve the performance with more considerable gains. This demonstrates the benefit of employing more generators for raising the generative ability. Moreover, Nest CycleGAN can achieve better accuracy than Long CycleGAN due to adding new inner cycles. For a full comparison, we also provide the Oracle results by testing real photos, which can be seen as the upper-bound performance. Our results on CMP-Facade narrow the gap with Oracle.

8.1.5 Results on photo↔sketch

We conducted this task with the SBIR dataset [238] which includes two subsets: one for shoes and the other for chairs. In the shoe dataset, we used 304 samples for training and 115 ones for testing. The chair dataset consists of 200 training samples and 97 testing ones. Figure 8.5 and Figure 8.6 present the generated image samples on the two datasets. We can see that the two extended models are advantageous to the original GAN and Vanilla CycleGAN. It is worth noting that, the sketch→photo

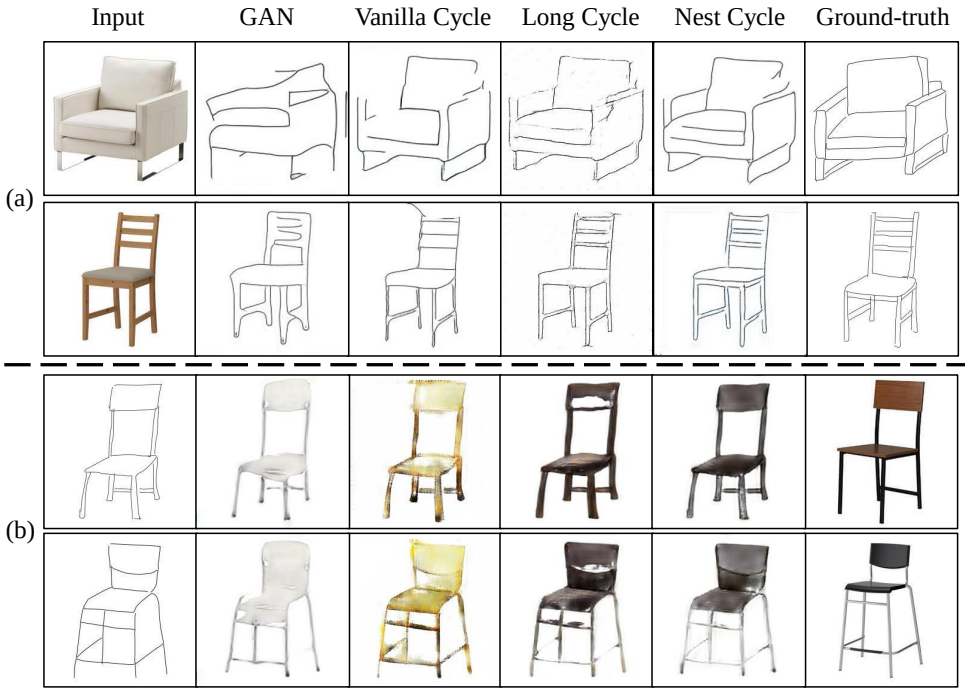


Figure 8.5: Qualitative results of (a) the photo→sketch translation and (b) the sketch→photo translation on the SBIR chairs dataset.

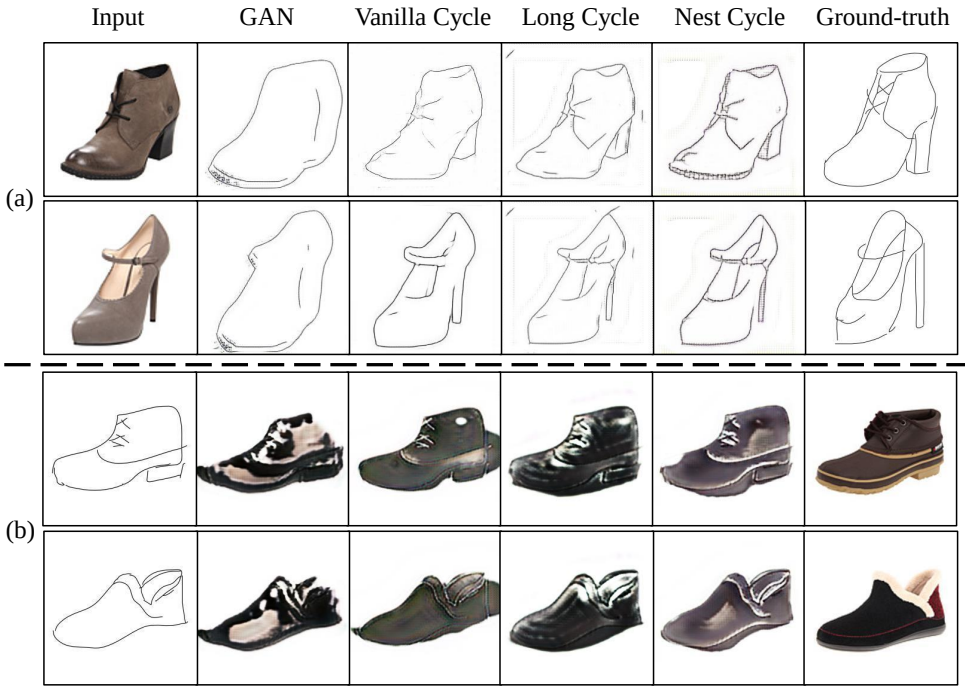


Figure 8.6: Qualitative results of (a) the photo→sketch translation and (b) the sketch→photo translation on the SBIR shoes dataset.

translation is more challenging than the photo→sketch translation. The main reason is that the sketch→photo mapping functions are more under-constrained, and one sketch image therefore may be synthesized with a variety of colors.

8.2 Fashion Style Transfer

Nowadays, online shopping has become an indispensable experience in our daily lives. Consequently, the huge market brought by fashion clothing shopping motivates an increasing variety of fashion relevant research, such as fashion clothing retrieval [95, 239], fashion recommendation [96, 240], fashion parsing [97, 241] and fashion aesthetics [242, 243]. In this work, we deal with the problem of fashion clothing swapping, which aims to visualize what the person would look like with the target clothes. From the practicality perspective, fashion clothing swapping is a useful experience for online consumers who need to virtually try on different clothes instead of wearing them physically. From the research perspective, fashion clothing swapping can be viewed as a specific task belonging to fashion style transfer. The challenge in this task is how to transform the target clothes fitting for the wearers while preserving their pose and body shape.

Traditionally, non-parametric methods [98, 101, 103, 244] are exploited to address this problem. They need to segment the target clothes from the condition image and then employ 2D image warping algorithms or 3D graphics methods to model the deformations between the clothes and the reference person body. However, these traditional methods rely on extra information (*e.g.* 3D measurements and geometric constraints) and complicated optimization algorithms (*e.g.* dynamic programming and dynamic time warping). In addition, non-parametric methods are not general, which means they need to estimate individual deformations for different image pairs. Also, it is non-tractable to match humans' key points due to non-rigid pose deformations.

In contrast to non-parametric methods, recent research [105, 106] turns to recast the clothing swapping as a 2D image synthesis problem. It is mainly driven by the rapid developments of deep generative networks, which have succeeded in many tasks involving synthesizing plausible images [84, 88, 245, 246]. Deep generative networks are able to synthesize the target images without requiring matching key points. Recently, FashionGAN [85] employs a textual description as condition to perform the clothing swapping (Figure8.7(a)). The methods in [105, 106] uses a stand-alone and flat clothing image to re-dress the reference person (Figure 8.7(b)). However, the target clothe is always worn on another person in practical scenarios, rather than is shown in a separate image. In this work, we aim to perform the person-to-person clothing swapping by transferring the clothes on the condition person images to the reference ones (Figure8.7(c)). It becomes more challenging due to the varying deformations among different human poses. Considering the challenge, we need to tackle the last research question **RQ 8: How can we exploit a generative model to directly transfer the fashion style between two person images?**

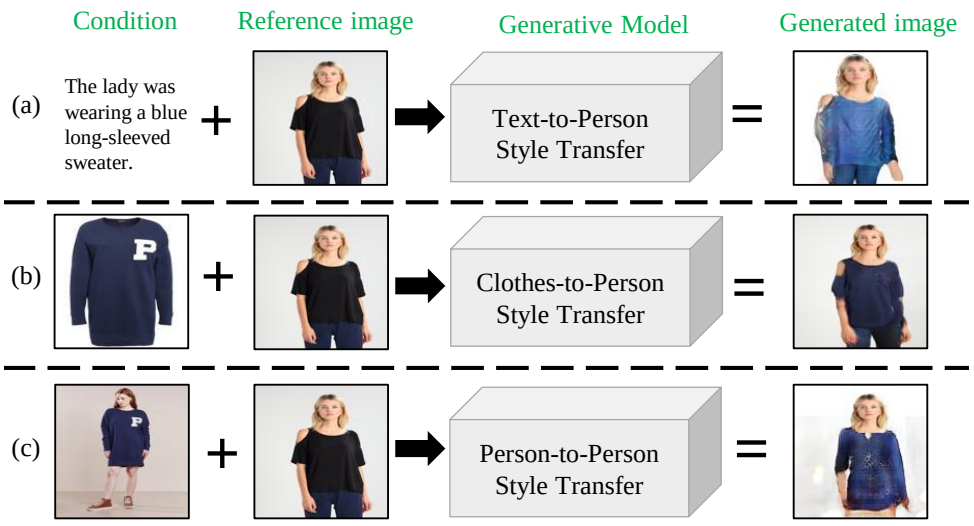


Figure 8.7: Three tasks of fashion clothing swapping conditioned on (a) textual description [85], (b) clothing image [106] and (c) person image, respectively. All the three cases aim to re-dress up the woman in the reference image with a long-sleeved sweather, while preserving her original pose and body shape. (c) shows the synthesized image based on our proposed SwapGAN.

To this end, we propose a multi-stage generative framework (SwapGAN), consisting of three generation stages conditioned on different priors. In the first stage, we interpret this problem as a pose-based person image synthesis process. We therefore exploit a pose-conditioned generative network (*i.e.* Generator I), which can manipulate the person in the condition image to have the same pose and body shape as the person in the reference image. Consequently, the new synthesized image can be viewed as the desired target image where the reference person wears the target clothes while preserving the original pose and body shape. Second, we further exploit a segmentation-conditioned generative network (*i.e.* Generator II) built on top of Generator I. The pose map in Generator I may mistake the clothing style (*e.g.* changing long sleeves to short sleeves), however, the segmentation in Generator II is used to retain the style due to its rich semantic information. To be specific, we take the segmentation map of the condition image into Generator II, to make sure that the synthesized image is consistent with the original condition image. Our hypothesis is that, *if a person image can be well transformed based on an arbitrary pose, then it should be feasible to reconstruct it based on its original segmentation map.* Moreover, we perform the third generation stage by using a mask generative network (*i.e.* Generator III). Generator III is used to explicitly constrain the body shape of the synthesized person images from both Generator I and Generator II. During the training procedure, we can train the entire SwapGAN end-to-end by integrating the adversarial loss from Generator I and Generator II and the mask-consistency loss from Generator III.

The contributions of this work are as follows:

- We prose a multi-stage generative framework for addressing a task of fashion style transfer, *i.e.* person-to-person clothing swapping. This is the first attempt to study it with a deep generative approach, to the best of our knowledge.
- In addition, our approach present the benefit of integrating multiple conditional GANs based on different priors. It can motivate tackling other research problems involved in deep generative networks.
- Furthermore, the experiments on the DeepFashion dataset verify the effectiveness of SwapGAN in terms of qualitative and quantitative evaluations. Our work can be a benchmark study to drive future research on this task. Also, it can enrich the application of deep generative approaches for solving practical problems.

The rest is structured as follows. Section 8.2.1 describes the proposed multi-stage generative model for person-to-person clothing swapping. The network architecture is detailed in Section 8.2.2. We report and discuss experimental results from Section 8.2.3 and Section 8.2.6.

8.2.1 Methodology

Problem Definition

We define the problem of person-to-person clothing swapping to be a conditional person image generation process. Its goal is to manipulate the person in the condition image to have the same pose and body shape as the person in the reference image. Additionally, we paste the head of the reference person onto the new synthesized image, in order to preserve the person identity. In this way, the reference person in the synthesized image can wear the target clothes in the condition image, while retaining the original pose and body shape.

Given a condition person image and a reference one, it may be infeasible to find the ground-truth target image in the dataset to supervise the synthesized image. Instead, we consider training the synthesis process using two images of the same person. To be specific, we have a training dataset of N image pairs, each of which is composed of two images of the same person with **the same clothes**, but with **different poses** (Figure 8.8). We randomly select one of the two images as a reference image, and the other one as a condition image. The reference and condition images are denoted with $X_r^{(i)}$ and $X_c^{(i)}$, $i = 1, \ldots, N$. Taking $X_c^{(i)}$ and the pose map of $X_r^{(i)}$ as input, our generator learns to create a fake $X_r^{(i)}$ during the training procedure. The discriminator needs to distinguish the fake $X_r^{(i)}$ from the real one. Ideally, when the discriminator cannot tell the differences between the real and fake images, the generators should be able to generate high-quality images.

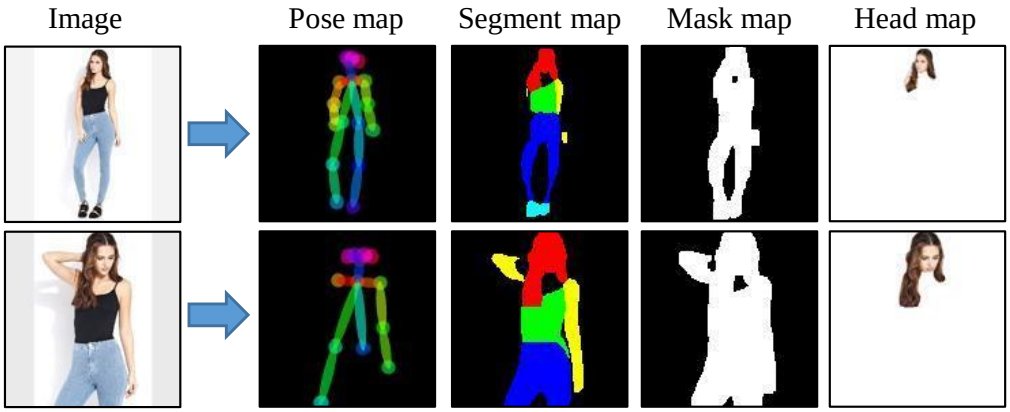


Figure 8.8: Representations for a pair of person images that have the same clothes but show different poses.

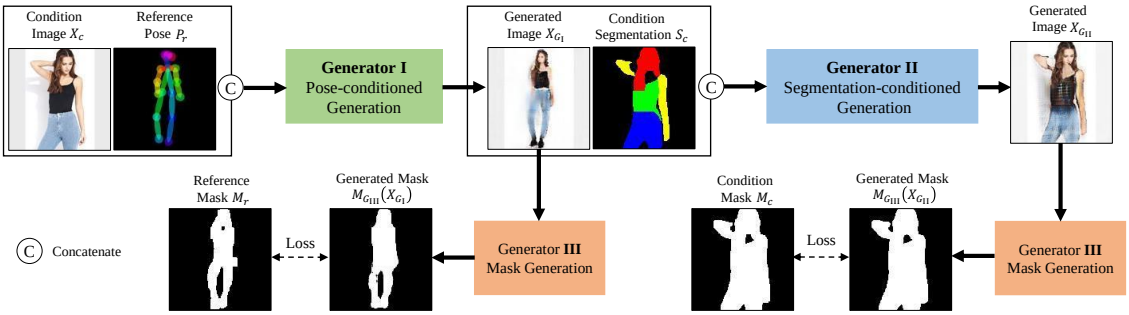


Figure 8.9: Overview architecture of the multi-stage generative framework in the proposed SwapGAN. **Generator** I can synthesize a new image X_{G_I} by manipulating the condition person image X_c based on the reference pose P_r. Then, **Generator** II takes as input X_{G_I} to produce a reconstructed X_c based on the segmentation map S_c. Moreover, **Generator** III is used to explicitly constrain the body shape during the synthesis process.

Person Representation

To specify the synthesis process, we need to extract a couple of person representations based on the person images. As shown in Figure 8.8, we utilize four feature maps described as follows:

1) *Pose map:* We employ one of the state-of-the-art pose estimators, OpenPose [247], to capture person pose information. For each person image, the pose estimator can localize 18 key-points in a pose map. In addition, the key-points are connected by color lines that can present the orientation of limbs. The pose map is used in Generator I.

2) *Segmentation map:* An off-the-shelf human semantic parser [248] is adopted to extract a person segmentation map. The original map can predict 20 fine classes for semantic segmentation. We further re-group the fine classes into five coarse classes, including head, arms, legs, upper-body clothes and lower-body clothes. We employ

this segmentation map in Generator II.

3) *Mask map:* Based on the above segmentation map, it is straightforward to obtain the binary mask of the person by merging all segmented regions. In contrast to the segmentation map, this mask map is used to retain the body shape without involving the semantic clues about the person. The mask maps of both the reference and condition person images are used for Generator III.

4) *Head map:* During the synthesis process, the details of the human face are hard to preserve due to its small size. However, it is needed to restore the identity of the reference person after swapping the clothes. To this end, we capture the head region (face and hair) based on the segmentation map, and paste it onto the new synthesized person image. This similar post-processing step is also used in FashionGAN [85].

For $X_r^{(i)}$ and $X_c^{(i)}$, we denote their four feature maps as $\{P_r^{(i)}, S_r^{(i)}, M_r^{(i)}, H_r^{(i)}\}$ and $\{P_c^{(i)}, S_c^{(i)}, M_c^{(i)}, H_c^{(i)}\}$, respectively. Subsequently, we will omit the superscript i for notational simplicity. We should mention that, these person representations are simple and efficient to extract without extra manual tuning. Note that, our representations are semantically richer than previous works [85, 105, 106].

Overview architecture

To render clothes from a person image on to another one, we propose an image synthesis framework (SwapGAN) based on conditional generative adversarial networks. Figure 8.9 illustrates the overview of SwapGAN, which has three different generators for pose-conditioned generation, segmentation-conditioned generation and mask generation, respectively.

Pose-conditioned generation

We begin to introduce the first generative stage conditioned on the pose map. As illustrated in Figure 8.9, we concatenate the condition image X_c and the reference pose map P_r together, and take them as input into the pose-based generative network, *i.e.* Generator I. We can express the synthesized image with

$$X_{G_\mathrm{I}} = G_\mathrm{I}(X_c, P_r). \tag{8.12}$$

We should mention that, the pose map can not only localize the human key-points, but also constrain the body shape of the synthesized person image to be the same as the reference person.

Next, X_{G_I} and X_c are integrated together to fake the discriminator D. Compared with the real pair of X_r and X_c, G_I learns to produce more realistic-looking images

similar to X_r. Following the original GANs [79], we use the negative log likelihood to compute the adversarial loss *w.r.t.* G_{I}

$$\mathcal{L}_{G_{\mathrm{I}}} = \mathbb{E}_{X_c \sim p_{data}(X_c), P_r \sim p_{data}(P_r)}[\log(D(X_{G_{\mathrm{I}}}, X_c))], \tag{8.13}$$

where $p_{data}(\cdot)$ indicates the empirical distributions of training data. As suggested in LSGAN [233], the least square loss is efficient to improve both the stability of training and the quality of generated images. Driven by this, we turn to use the least-square adversarial loss to represent $\mathcal{L}_{G_{\mathrm{I}}}$:

$$\mathcal{L}_{G_{\mathrm{I}}} = \mathbb{E}_{X_c \sim p_{data}(X_c), P_r \sim p_{data}(P_r)}[(D(X_{G_{\mathrm{I}}}, X_c) - 1)^2], \tag{8.14}$$

The objective for Generator I is to minimize $\mathcal{L}_{G_{\mathrm{I}}}$.

Segmentation-conditioned generation

Given two arbitrary person images, Generator I can synthesize new images by exchanging the clothes and its results therefore can meet the goal of this task. However, the key-points in the pose map are mainly used to measure the localization information of body parts, but pay little attention to the style of the target clothes in the condition image. To address this limitation, we propose to leverage the person segmentation map, which can take into consideration semantic information about the clothes.

Empirically, if $X_{G_{\mathrm{I}}}$ has derived the target clothes from X_c, it should be possible to return the clothes back to the condition person again. In this way, the fashion style of the clothes can be reconstructed well during the synthesis process. This idea motivates the second generative stage that aims towards synthesizing another new image as similar as the condition image X_c. Specifically, we build a segmentation-based generative network (*i.e.* Generator II in Figure 8.9), on top of the output of Generator I. Generator II takes as input the concatenation of the synthesized image $X_{G_{\mathrm{I}}}$ and the condition segmentation map S_c. As a result, we can obtain a new synthesized image from the output of Generator II:

$$X_{G_{\mathrm{II}}} = G_{\mathrm{II}}(X_{G_{\mathrm{I}}}, S_c) = G_{\mathrm{II}}(G_{\mathrm{I}}(X_c, P_r), S_c). \tag{8.15}$$

Ideally, $X_{G_{\mathrm{II}}}$ should be as similar as the original input X_c. From X_c to $X_{G_{\mathrm{II}}}$, the integration of the first and second generative stages actually construct an auto-encoder paradigm. It can help improve the quality and semantics of the generated image $X_{G_{\mathrm{I}}}$. For instance, Generator I may mistake the fashion style by transferring long sleeves to be short sleeves. However, Generator II is capable of correcting the mistake, because the segmentation map includes the lost information about the long sleeves. Next, we incorporate X_r and $X_{G_{\mathrm{II}}}$ into the same discriminator D, and

compute the generative loss function of G_{II}

$$\mathcal{L}_{G_{\mathrm{II}}} = \mathbb{E}_{X_r \sim p_{data}(X_r), S_c \sim p_{data}(S_c)}[(D(X_r, X_{G_{\mathrm{II}}}) - 1)^2]. \tag{8.16}$$

Minimizing this loss can jointly optimize Generator II and Generator I.

Mask generation

Although the pose map and segmentation map have provided some information about the body shape, it is encouraged to learn another generative network to explicitly constrain the synthesized images. As shown in Figure 8.9, we employ a shared Generator III to perform the mask generation for both $X_{G_{\mathrm{I}}}$ and $X_{G_{\mathrm{II}}}$. Different from Generator I and Generator II, Generator III takes only one image as input without specifying other conditions. The two generated masks, denoted as $M_{G_{\mathrm{III}}(X_{G_{\mathrm{I}}})}$ and $M_{G_{\mathrm{III}}(X_{G_{\mathrm{II}}})}$, should consistently match the reference mask M_r and the condition mask M_c, respectively. We define their mask-consistency loss as follows:

$$\begin{aligned}\mathcal{L}_{G_{\mathrm{III}}} =& \mathbb{E}_{M_r \sim p_{data}(M_r)}[||M_{G_{\mathrm{III}}(X_{G_{\mathrm{I}}})} - M_r||_1] \\ &+ \mathbb{E}_{M_c \sim p_{data}(M_c)}[||M_{G_{\mathrm{III}}(X_{G_{\mathrm{II}}})} - M_c||_1].\end{aligned} \tag{8.17}$$

Both G_{I} and G_{II} can benefit from the loss $\mathcal{L}_{G_{\mathrm{III}}}$ to update the synthesis process. Note that, $\mathcal{L}_{G_{\mathrm{III}}}$ will not update the parameters of the discriminator D, because the generated masks are unnecessary to feed into the discriminator. In Figure 8.9, it can be seen that, after training, the generated masks end up similar to the reference and condition mask maps.

Full Objective

The SwapGAN model including three generators and one discriminator can be trained end-to-end. The total generation loss combines the adversarial loss (*i.e.* $\mathcal{L}_{G_{\mathrm{I}}}$ and $\mathcal{L}_{G_{\mathrm{II}}}$) and the mask-consistency loss (*i.e.* $\mathcal{L}_{G_{\mathrm{III}}}$)

$$\mathcal{L}_G = \mathcal{L}_{G_{\mathrm{I}}} + \mathcal{L}_{G_{\mathrm{II}}} + \lambda \mathcal{L}_{G_{\mathrm{III}}}, \tag{8.18}$$

where λ adjusts the weight of $\mathcal{L}_{G_{\mathrm{III}}}$, which we set to 5 in the experiments.

Figure 8.10 shows the structure of the discriminator D. Compared to prior work [246] comparing one real pair and one fake one, our discriminator is able to distinguish one real pair from two fake pairs. Formally, the discrimination loss in D can be

defined with

$$\begin{aligned}\mathcal{L}_D =& \mathbb{E}_{X_r \sim p_{data}(X_r), X_c \sim p_{data}(X_c)}[(D(X_r, X_c) - 1)^2] \\ &+ \mathbb{E}_{X_c \sim p_{data}(X_c), P_r \sim p_{data}(P_r)}[D(X_{G_{\mathrm{I}}}, X_c)^2] \\ &+ \mathbb{E}_{X_r \sim p_{data}(X_r), S_c \sim p_{data}(S_c)}[D(X_r, X_{G_{\mathrm{II}}})^2]. \end{aligned} \tag{8.19}$$

During the training procedure, it is a common practice to iteratively update the parameters of the generators and the discriminator. The full objective in the model is to minimize both $\mathcal{L}_G$ and $\mathcal{L}_D$. The generators attempt to generate more realistic-looking fake images to fool the discriminator. Once the discriminator cannot tell fake images from real ones, then the generators are supposed to properly accomplish the synthesis process. In the testing phase, taking a condition image and the pose map of a reference image as input, the synthesized image from Generator I, *i.e.* $X_{G_{\mathrm{I}}}$, can be used as the desired target image. Additionally, we need to paste the reference head map H_r onto $X_{G_{\mathrm{I}}}$ to make sure the person's identity is preserved.

8.2.2 Network architecture

This section introduces the details about the network architecture of the generators and the discriminator in the SwapGAN.

Generator I and II

By integrating several existing techniques, we design a new generative network for G_{I} and G_{II}. As shown in Figure 8.11, it consists of an encoder, several residual blocks and a decoder. (1) In the encoder, we use four consecutive convolutional layers to represent the input data. (2) There are totally six residual blocks, each of which has two 3×3 convolutional layers and a residual connection on them [10, 80]. (3) As for the decoder, we employ a nearest neighbor interpolation manner to upsample the feature maps, and then transfer the resized feature maps with a 1×1 convolutional layer. Compared with the deconvolution manner based on stride-$\frac{1}{2}$ convolutions, the interpolation manner is simple and efficient to alleviate the checkerboard artifacts, which often occur in generated images [249]. Figure 8.12 visibly compares the generated images by using the two upsampling manners.

In addition, we add skip connections to link the feature maps in the encoder and decoder. As suggested in U-Net [28], the skip connections allow to bridge the down-sampled feature maps directly with the up-sampled ones. They can help retain the spatial correspondences between the input pose/segmentation map and the synthesized image.

Generator III

Since the mask generation is less complicated than the pose-conditioned generation and the segmentation-condition generation, we can make use of a simple U-Net [28] to build G_{III}. Specifically, Generator III learns eight convolutional layers in the encoder and eight deconvolutional layers in the decoder. Similarly, the symmetric skip connections are added between the encoder and the decoder. The residual blocks are not used in G_{III}. Notably, G_{III} can be built as well with the same generative network as G_{I} and G_{II}, however, we find that it cannot bring further improvements for the generated masks.

Discriminator

We build the discriminator D based on the Markovian network from PatchGANs [88], which is encouraged to preserve local high-frequency features. As shown in Figure 8.10, D uses four consecutive layers to convolve the concatenated real or fake image pairs. Lastly, an additional convolutional layer can output a 1-dimensional feature map to classify the patches on the input images are real or fake.

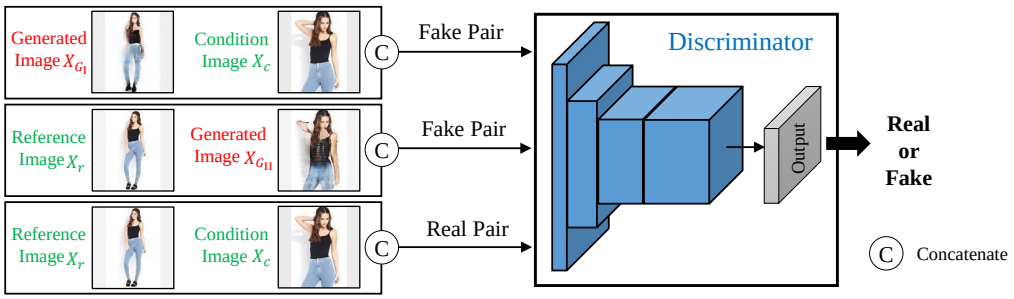


Figure 8.10: Overview of the discriminator D in SwapGAN. It aims to distinguish two fake image pairs from the real pair.

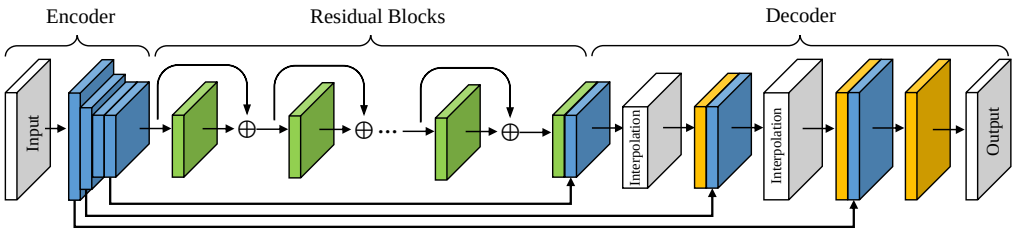
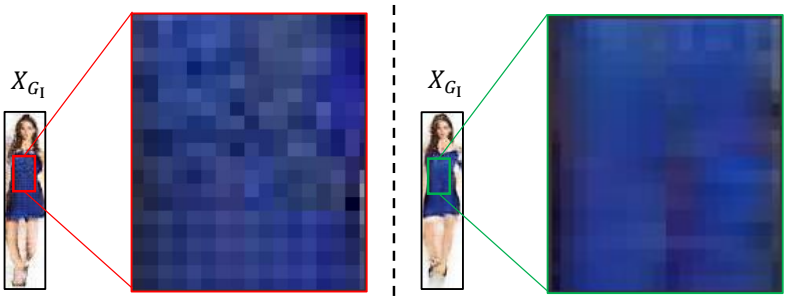


Figure 8.11: Network architecture of both Generator I and II. It is composed of three parts: encoder, residual blocks and decoder. We use additional skip connections to couple the feature maps in the encoder and decoder. In the decoder, we perform the upsampling with an interpolation manner instead of the traditional deconvolution manner.



(a) Deconvolution upsampling (b) Interpolation upsampling

Figure 8.12: Comparison of using two different upsampling manners in the generator. The deconvolution manner results in more checkerboard artifacts that will decrease the generation quality. To alleviate this issue, we use the interpolation manner to generate smooth images. See more details when zoomed-in.

(a) inappropriate person images

(b) appropriate person images

Figure 8.13: Examples of (a) inappropriate and (b) appropriate person images. Considering the goal of person-to-person clothing swapping, we collect the front-view images with both upper-body and lower-body clothes visible.

8.2.3 Experiment setup

Dataset protocol

Currently, DeepFashion [241] is one of the largest datasets for fashion oriented research. We used its In-shop Clothes Retrieval Benchmark, which has a number of in-shop person images with various poses and scales. However, many of the images are inappropriate to the clothing swapping task, due to some issues like missing human faces, back-view images and only upper-body clothes visible. To avoid these issues, we selected front-view person images where the clothing items are shown clearly. In Figure 8.13, we show some examples of inappropriate and appropriate person images. In the training set, we collected 6,000 person images corresponding to 3,000 image pairs, each of which has two images of the same person wearing the same clothes but showing different poses. The testing set contains 1,372 images.

Figure 8.14: Qualitative results of our SwapGAN on the test set. We show four reference images in the first row and four condition images in the first column. The reference person can wear the desired clothes in the condition image while preserving the original pose and body shape.

Implementation Details

We employed the Adam algorithm [234] to optimize the entire SwapGAN with $\beta_1 = 0.5$ and $\beta_2 = 0.999$. The initial learning rate for the generators and discriminator was 0.0002, and was linearly decayed after 50 epoches. The entire training procedure was terminated after 100 epochs. All the images were re-scaled to 128×128 pixels. We used a mini-batch size of 8. We implemented the method on the TensorFlow library [235] with a NVIDIA TITAN X GPU card.

Compared methods

We compare our SwapGAN with other three methods described as follows.

Poisson image blending [85]: it is the 2D non-parametric method that uses the Poisson image blending algorithm to apply the target clothes in the condition person image on the person of the reference image. This method is used as a baseline in FashionGAN [85].

TPS warping [106]: this is another non-parametric method. It first estimates a thin plate spline (TPS) transformation and then pastes the warped clothes on the reference image. This is a baseline method in VITON [106].

VITON [106]: in contrast to non-parametric methods, it proposes an encoder-decoder network to generate a new reference person image wearing the target clothes.

We note that, all the three compared methods require segmenting the target clothes from the condition person images. By this way, they can learn the transformations between two different images.

8.2.4 Results and discussion

First, we compare our SwapGAN with other compared methods in terms of both qualitative and quantitative evaluations. Then, we perform ablation study to provide deep insights into SwapGAN.

Qualitative evaluation

This experiment aims to qualitatively show the effectiveness of our method for person-to-person clothing swapping. Figure 8.14 shows our new synthesized images. As for each row, the clothes in the condition image are worn on different reference persons. Also, each column indicates that the same reference person is re-dressed with different clothes. It can be seen that all the reference persons can properly wear the target clothes in the condition images and retain their original poses and body shapes as well. Since we paste the reference head map to ensure the person's identity, some generated images therefore seems a little unnatural.

Next, we compare our results with those of the compared methods. In Figure 8.15, we present a reference image and three condition images. To assess the robustness for different pose deformations, the persons in the three condition images have small, moderate and large pose deformations, respectively, compared to the person in the reference image. From the results, we can see that the Poisson image blending method fails to perform this task. The similar observation is also presented in [85]. Instead of generating a new image, the TPS warping method learns to transform the target clothes and simply pastes it on the reference person. Although the color information can be well preserved in its results, we can notice obvious inconsistency between the warped target clothes and the body of the reference person. The results

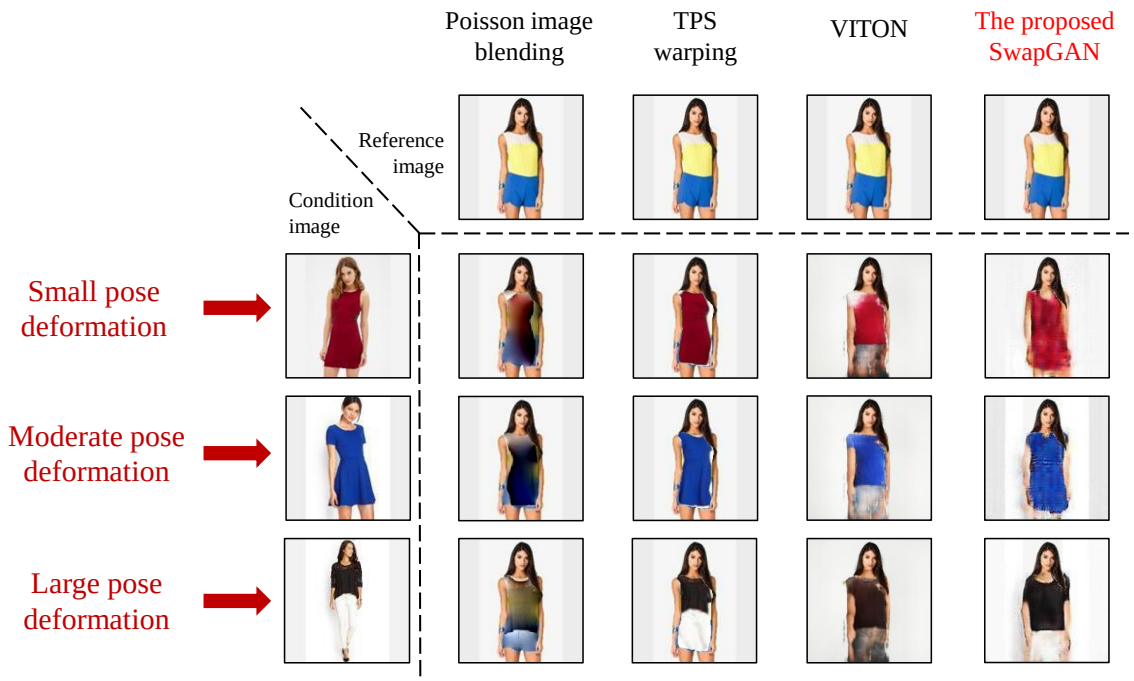


Figure 8.15: Qualitative comparison of different methods. When comparing with the person in the reference image, the persons in the three condition images have small, moderate and large pose deformations, respectively. Compared to other methods, our SwapGAN can visibly provide superior images. Our method is robust to different pose deformations, even the large case in the last row.

of VITON are not satisfactory, because their model is trained with simple stand-alone and flat clothes images, rather than various warped clothes on the condition persons. Compared to the above methods, SwapGAN can generate superior new images for all the condition images. In addition, our method is robust to different pose deformations, however, the three compared methods are weak in the robustness.

Quantitative evaluation

In addition to qualitative results, we adopt a common quantitative metric, Inception Score (IS) [250], to assess the methods. IS is based on the Google's inception CNN model [11], which predicts a distribution $p(y|x)$, measuring the probability assigned to image x to belong to class y. Formally, the computation of IS is expressed by

$$IS = exp\left(\mathbb{E}_{x\sim p_g}\left[KL\left(p(y|x)\|p(y)\right)\right]\right), \tag{8.20}$$

where p_g indicates the distribution of a generative model. $KL(p(y|x)\|p(y))$ measures The Kullback-Leibler divergence [251] between $p(y|x)$ and $p(y)$:

$$KL\left(p(y|x)\|p(y)\right) = \sum_{k=1}^{K} p_k(y|x)\log\frac{p_k(y|x)}{p_k(y)}. \tag{8.21}$$

Table 8.2: Quantitative comparison of different approaches with inception scores (higher is better). Our SwapGAN can outperform the other three compared methods with considerable gains.

Method	Inception score
Poisson image blending	2.10 ± 0.14
TPS warping	2.45 ± 0.12
VITON	2.40 ± 0.05
SwapGAN	2.65 ± 0.09

For the 1,372 images in the test set, we iteratively make each image as the reference image, and then randomly select another 25 images to be its corresponding condition images. As a result, we can collect about 34,000 reference-condition pairs, each of which can produce an image to evaluate. Table 8.2 reports the inception scores towards the 34,000 images. Interestingly, the TPS warping method has a greater score than VITON, because it simply pastes the warped clothes on the reference image, which can help preserve the color information. However, it cannot generate a new image like VITON and SwapGAN. In [106], they also discuss the limitation of the TPS warping method. Overall, SwapGAN achieves a higher score than the other three methods.

8.2.5 Ablation study

We demonstrate ablation results about SwapGAN and analyze the effects of its generators on the performance. To be more specific, we implement two ablation models, which are variants of the full SwapGAN model. The first ablation model is named by Generator I&III, which excludes the segmentation-conditioned generation. The second one, called Generator I&II, keeps the first and second generations but excludes the mask generation. Figure 8.16 shows two generated image samples, from which we have the following observations:

(1) Effect of Generator II. As can be seen in the first row, Generator I&III mistakes the fashion style of the target clothes, because it changes the short sleeves in the condition image to be long sleeves in the new generated image. However, both the Generator I&II model and the full SwapGAN model can avoid this semantic inconsistency due to using the segmentation map in Generator II. It verifies the effectiveness of Generator II for maintaining the style.

(2) Effect of Generator III. Considering the generated images from the Generator I&II model, some parts of the human body are not preserved well, for example, the right arms. By running the mask generation, the full SwapGAN model can produce a more complete body shape similar with the reference image. This demonstrates the benefit of Generator III for our method.

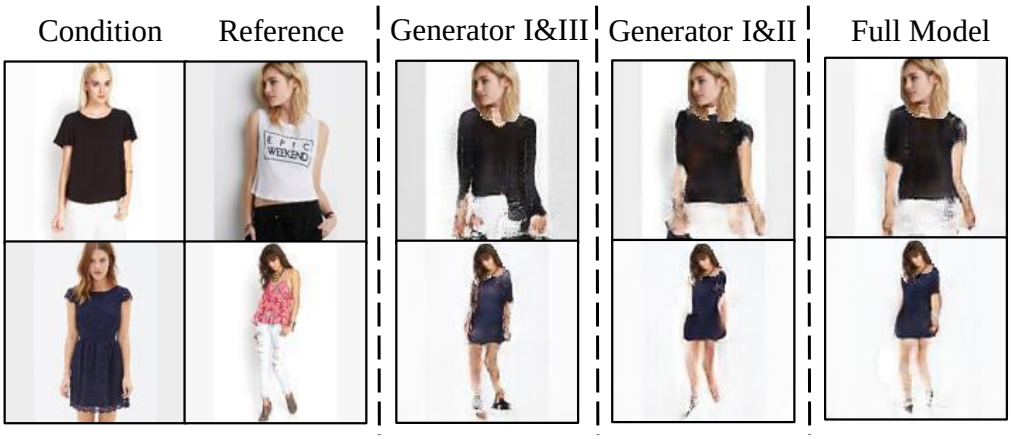


Figure 8.16: Ablation study on different variants of our method. Comparably, the full model can outperform the other two baseline models in terms of generation quality and semantics.

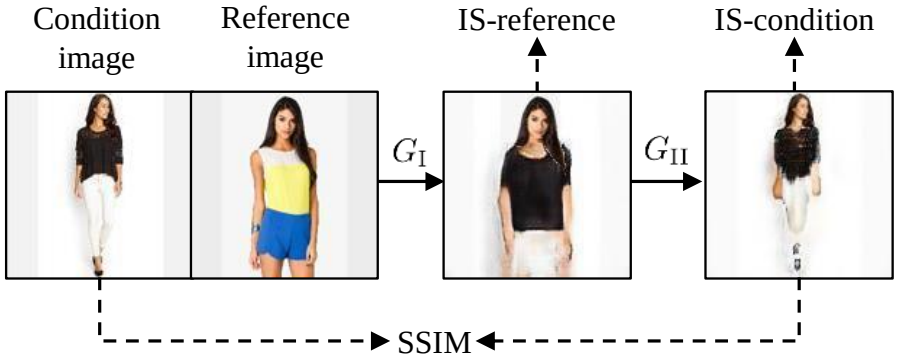


Figure 8.17: Pipeline of our testing procedure with computing two inception scores and a SSIM accuracy.

Figure 8.18: Failure cases of our method for synthesizing complicated color and texture on the clothes.

In terms of quantitative results, we exploit a new test procedure as shown in Figure 8.17. Since SwapGAN can synthesize two new images from G_{I} and G_{II}, we can compute their inception scores respectively, denoted as IS-reference and IS-condition. In addition, the synthesized image from G_{II} is a reconstructed image of the original condition image. Hence, we can adopt another quantitative metric, Structural Similarity (SSIM) [252], to measure the reconstructed similarity.

In Table 8.3, we compare the quantitative results between two ablation models and the full SwapGAN model. Notably, the Generator I&III model has no IS-

Table 8.3: Quantitative results of our different models.

Method	IS-reference	IS-condition	SSIM
Generator I&III	2.47 ± 0.11	–	–
Generator I&II	2.36 ± 0.14	2.66 ± 0.12	0.708
Full Model	2.65 ± 0.09	2.85 ± 0.12	0.717

condition and SSIM accuracy, because it excludes G_{II}. We can see that the full model consistently outperforms the other two ablation models by a considerable margin, in terms of both IS-reference and IS-condition metrics. Moreover, the full model achieves a higher SSIM accuracy than Generator I&II. These quantitative results are consistent with our observation achieved from the qualitative evaluation.

8.2.6 Limitations and discussion

Our method has achieved promising results in many cases, but still has some limitations. First, human faces become blurred in the synthesis process, because it is hard for the generator to restore the detailed face of the reference person. To alleviate this limitation, we employ a post-processing step by pasting the reference head map onto the synthesized image. Second, our method may fail to capture rich color and texture information of the clothes, for example, the failure cases in Figure 8.18. This problem is caused by the limited capability of the adversarial loss. One approach for solving it is to impose additional losses like the perception loss [80], but it will increase the memory cost and training time.

8.3 Chapter Conclusions

First, this work provided an extensive and empirical study on the cycle-consistent generative networks for unsupervised image translation. The comprehensive results demonstrated the effectiveness of our designed models. Besides, the insights observed in this work could help in designing other new cycle-consistency models. In the future, it is straightforward and promising to develop Long and Nest CycleGAN with more generators and cycles. Also, it is interesting to employ a weight-sharing mechanism to avoid increasing memory.

Second, we proposed a novel multi-stage generative adversarial framework to address the problem of person-to-person clothing swapping. Advantageously, it could render the clothing style and preserve the pose and body shape within a multi-stage model. In addition, our model was able to train end-to-end. Qualitative and quantitative results in the experiments demonstrated the effectiveness of our approach. In the future, we plan on developing our approach for images in the wild.