

Article

Deep Fashion Designer: Generative Adversarial Networks for Fashion Item Generation Based on Many-to-One Image Translation

Jaewon Jung ¹, Hyeji Kim ² and Jongyoul Park ^{3,*}¹ Huraypositive Corp, Seoul 06628, Republic of Korea; woodcook486@huray.net² AI Convergence and Open Sharing System, Seoul National University of Science and Technology, Seoul 01811, Republic of Korea; kimhyejee923@seoultech.ac.kr³ Department of Applied Artificial Intelligence, Seoul National University of Science and Technology, Seoul 01811, Republic of Korea

* Correspondence: jongyoul@seoultech.ac.kr

Abstract: Generative adversarial networks (GANs) have demonstrated remarkable performance in various fashion-related applications, including virtual try-ons, compatible clothing recommendations, fashion editing, and the generation of fashion items. Despite this progress, limited research has addressed the specific challenge of generating a compatible fashion item with an ensemble consisting of distinct categories, such as tops, bottoms, and shoes. In response to this gap, we propose a novel GANs framework, termed Deep Fashion Designer Generative Adversarial Networks (DFDGAN), designed to address this challenge. Our model accepts a series of source images representing different fashion categories as inputs and generates a compatible fashion item, potentially from a different category. The architecture of our model comprises several key components: an encoder, a mapping network, a generator, and a discriminator. Through rigorous experimentation, we benchmark our model against existing baselines, validating the effectiveness of each architectural choice. Furthermore, qualitative results indicate that our framework successfully generates fashion items compatible with the input items, thereby advancing the field of fashion item generation.



Academic Editors: Sinan Chen and Jialong Li

Received: 6 November 2024

Revised: 29 December 2024

Accepted: 30 December 2024

Published: 7 January 2025

Citation: Jung, J.; Kim, H.; Park, J. Deep Fashion Designer: Generative Adversarial Networks for Fashion Item Generation Based on Many-to-One Image Translation. *Electronics* **2025**, *14*, 220. <https://doi.org/10.3390/electronics14020220>

Copyright: © 2025 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

Keywords: generative adversarial networks; fashion image synthesis; fashion compatibility

1. Introduction

Developing a new fashion item without any initial guidance presents a significant challenge for fashion experts, including stylists and designers. For a new design to be marketable, it must complement existing fashion products. However, even for experienced professionals, this task is challenging and requires a significant amount of time and effort.

To tackle this challenge, we introduce a novel framework called DFDGAN as illustrated in Figure 1. This framework synthesizes a new, compatible fashion item based on a series of fashion items from different categories that constitute an outfit. Consequently, the provided items are presumed to be compatible with one another.

Our framework holds potential for integration into intelligent recommendation systems within fashion e-commerce, assisting customers in finding garments that complete a stylish outfit. Moreover, as the framework generates novel items rather than merely retrieving existing products, it inspires fashion designers to create innovative designs.

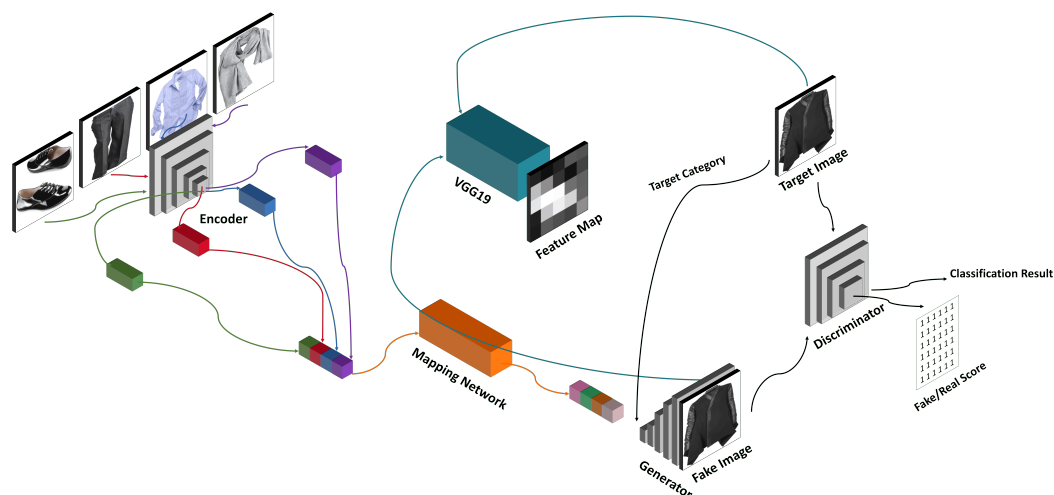


Figure 1. The overview of our framework. Four fashion images (shoes, suit pants, suit top, and a muffer) are input into the encoder, which generates four corresponding latent vectors. These latent vectors are then transformed into a single combined latent vector by the mapping network. The generator uses this combined latent vector along with the category of a fashion item to produce a synthetic image. Feature maps are extracted based on the synthetic and target images from VGG19 [1]. Finally, the discriminator evaluates the generated image and its associated category.

GANs have garnered considerable attention in machine learning research and have been widely adopted [2]. Due to the success of GANs-based studies in image generation and translation, the fashion industry has increasingly utilized GANs to address challenging problems. Notable applications include virtual try-on networks [3–5], fashion image editing [6–8], clothing out [9–11], fashion image translation [12,13], and fashion image generation [14–18].

We propose a fashion image generation framework that addresses a unique challenge: given a set of compatible fashion items across multiple categories, what is the most compatible fashion item to complement them?

Previous studies on fashion image generation primarily focus on generating a fashion item from a single garment with text descriptions through compatibility learning [14–17]; however, a study by Moosaei et al. aligns with our approach, using multiple categories of source items to synthesize a fashion item [18].

OutfitGAN [18] comprises a pre-trained fashion compatibility network, which is essential for training the image generation components, along with multiple generators and discriminators. The image generation framework is trained using a multi-stage learning approach, and it produces images with a resolution of 128×128 pixels.

In contrast, our framework employs an end-to-end design, eliminating the need for auxiliary networks to model fashion compatibility and the complexity of multi-stage training. Additionally, it produces high-resolution images at 256×256 pixels. The proposed fashion compatibility batch algorithm facilitates the generation of innumerable unsuitable outfits, enabling them to learn fashion compatibility.

Our framework consists of four key components: an encoder, a mapping network, a generator, and a discriminator, as presented in Figure 1. The encoder processes multiple images to create a feature representation, which is then transformed by the mapping network into a combined feature space. The generator uses this transformed feature with the category of a fashion item to generate a synthetic image, while the discriminator compares the synthetic image to the ground truth.

The contributions are recapped as follows:

- **End-to-end manner:** Our framework is trained through a fully end-to-end process, eliminating the need for additional networks. This design simplifies the training process and makes it less challenging compared to OutfitGAN, as it avoids the complexity of multi-stage training.
- **Fashion compatibility batch algorithm:** The proposed algorithm enables the framework to effectively learn fashion compatibility by leveraging numerous unsuitable outfits, facilitating the generation of plausible fashion images even on unseen data.
- **A large image resolution:** The resolution of the generated images is significantly improved, quadrupling that of OutfitGAN [18] from 128×128 pixels to 256×256 pixels.

The organization of the remaining paper is as follows: Related work, including studies on GANs variants and fashion image generation, is reviewed in Section 2. The proposed method, including problem definition, framework, and training algorithm, is detailed in Section 3. Section 4 introduces the dataset, pre-processing, and implementation details. Model evaluation metrics and experimental results are described in Section 5, followed by an analysis of the trained model in Section 6. Section 7 discusses the limitations of existing baselines, and, finally, Section 8 concludes with a summary of contributions.

2. Related Work

2.1. Generative Adversarial Networks

There are four primary types of GANs: noise to image, one-to-one image translation, reference-based image translation, and many-to-one image translation. It is important to emphasize that these methods are not mutually exclusive and may overlap in their applications and characteristics.

Noise to image: The works cited in [2,19–22] fall under the category of noise-to-image GANs. In this approach, images are generated from noise, specifically from a latent vector sampled from a uniform or Gaussian distribution.

For instance, in [2,19], digits, bedrooms, and human faces are generated from noise. In contrast, the StyleGAN, StyleGAN2, StyleGAN3 [20–22] extend this approach by learning styles from noise using a constant tensor, enabling the generation of high-resolution human faces.

One-to-one image translation: In this category, both the input and output are images, with attributes in the output image being modified from those in the input image.

For example, pix2pix [23] performs image-to-image translation by converting gray-scale images into colorful ones and vice versa. CycleGAN [24] addresses the limitation of pix2pix, which requires paired ground-truth and labeled images, by introducing cycle consistency to enable unpaired image translation. DiscoGAN [25] expands on this concept by translating images between two distinct domains, such as from bags to chairs. StarGAN [26] further advances the field by enabling the simultaneous translation of multiple facial attributes, including hairstyle, gender, and age, within a single person's image.

Reference-based image translation: In this approach, a reference image is utilized to extract specific attributes, which are then injected into the input image.

For example, StarGAN2 [27] extracts the hair attributes from the reference image and incorporates them into the input image using a single generator. Similarly, the StyleGAN series, StyleGAN, StyleGAN2, and StyleGAN3 [20–22] also belong to this category, where a reference image is used to extract style features that are subsequently applied to the input image.

Many-to-one image translation: CharacterGAN [28], DCTON [29], and the work by Gafni et al. [30] are examples of many-to-one image translation. These studies utilize multiple input images to generate a single output image. The key distinction in these approaches lies in the fact that the input images serve as components that directly contribute to the composition of the generated image.

Our framework is inspired by the type of “noise to image” GANs’ approach. This type of GANs maps a latent vector from a distribution with data. When the multiple images are transformed together into a latent vector, training a model that learns the mapping between a certain image and given several images is available. We applied this framework to generate a suitable fashion image based on multiple categories of fashion images.

2.2. Fashion Item Generation

The studies discussed below propose methods for generating fashion items. These methods focus on generating a compatible fashion item based on a given fashion item, either with or without accompanying text descriptions, while considering overall compatibility.

MrCGAN Shih et al. [14] proposed a method involving projected compatibility distance (PCD) and metric-regularized conditional generative adversarial networks (MrCGAN). The PCD function is employed to train multiple models, enabling the learning of various embedding spaces through these multiple models.

FARM Lin et al. [15] proposed the Fashion Recommendation Machine (FRAM), a neural co-supervision learning framework, which jointly trains both the generator and the recommender. Their approach utilizes a variational transformer along with visual and layer-to-layer matching, as well as description matching within the network, to generate a suitable fashion item based on text descriptions.

MGCM Liu et al. [16] proposed a multi-modal generative compatibility modeling (MGCM) framework that generates lower garments based on given upper garments and accompanying text descriptions. Their approach incorporates a loss function designed to ensure compatibility between items (item-to-item compatibility) as well as between items and text (item-to-text compatibility), utilizing a text-based convolutional neural network (CNN).

CMRGAN Liu et al. [17] presented a multi-modal model that integrates visual and text descriptions using a U-Net architecture [31]. The model learns a compatibility space based on a triplet structure, where the discriminator evaluates the compatibility between a pair of ground-truth images and a pair of generated images.

OutfitGAN Moosaei et al. [18] introduced a novel framework that utilizes multiple input images to generate a compatible output image. This framework comprises a multi-stage generative adversarial network architecture. Training the framework requires a pre-trained fashion compatibility module, which is derived from the dataset provided by [32]. By conditioning on specific categories, the framework generates images that are well suited to the given input images.

In summary, MrCGAN, FARM, MGCM, and CMRGAN [14–17] align with our research in their focus on fashion image generation with compatibility considerations. However, these methods operate within the scope of a one-to-one image translation framework, where the goal is to generate a fashion image based on a single-input item, such as tops paired with shoes or bottoms paired with hats. In contrast, both our framework and OutfitGAN [18] adopt a many-to-one image translation framework, where the inputs consist of multiple categories of fashion items, and the output is a cohesive and compatible fashion image.

Fashion compatibility: Fashion compatibility refers to the harmonious combination of two or more fashion items within an outfit. Han et al. [32] explored this concept by employing a Bi-LSTM model [33] and introduced the task of compatibility prediction to

evaluate the coherence of an outfit. Vasileva et al. [34] extended this approach by utilizing conditional similarity networks (CSNs) [35] to achieve type-aware fashion compatibility. Moosaei et al. [18] further advanced the field by obtaining fashion compatibility through relation networks [36] using the Polyvore Dataset [32].

3. Methodology

3.1. Problem Definition

Let $x_{\text{generated}}$ represent a generated image, G is a generator, and f be an encoder and mapping network. The latent vectors are represented by z_i for $i = 1, 2, \dots, N$, and cat denotes the category of the fashion item.

Equation (1) represents the fashion image generation process, where $x_{\text{generated}}$ is the output image produced by the generator G . The generator takes as input a category of fashion item, and a combined latent vector, $f([z_1, z_2, \dots, z_N])$, which is formed by the latent vectors z_i (for $i = 1, 2, \dots, N$) through an encoder and the mapping network f .

$$x_{\text{generated}} = G(f([z_1, z_2, \dots, z_N]), \text{cat}) \quad (1)$$

To synthesize a suitable fashion item corresponding to a set of multiple categories, such as bottoms, shoes, eyeglasses, and earrings, collectively termed source images, the framework maps these items to the target category, for instance, tops. The proposed framework comprises an encoder, a mapping network, a generator, and a discriminator.

The encoder performs a vital role in the generation process by producing latent vectors that capture various aspects of fashion items, with the distribution of these latent vectors denoted as $p(z_i)$. These latent vectors are concatenated and fed into the mapping network, which generates a combined latent vector z_{combined} .

This combined latent vector is then input to the generator, G , which synthesizes a suitable fashion image $x_{\text{generated}}$, reflective of the learned fashion compatibility distribution. Concurrently, the discriminator assesses the fashion compatibility and authenticity of the generated images by comparing them to both real images and the source images from the dataset, thereby introducing adversarial learning dynamics.

The process of constructing the input for Equation (1) and its training methodology is detailed in Section 3.5.

3.2. Model Architecture

Encoder The encoder utilizes six residual blocks with instance normalization and Leaky-ReLU activation with a scale of 0.01 integrated with a convolutional block attention module (CBAM) [37], a global average pooling, and a convolutional layer.

Initially, the size of the image tensor is increased to $128 \times 128 \times 64$ at the first layer. As the tensor passes through each subsequent layer, its depth is doubled while its spatial dimensions are halved. In the final layer, the depth of the tensor is reduced to 128. The encoder generates four latent vectors, each with dimensions of $1 \times 1 \times 128$, corresponding to four different categories of fashion items. These four latent vectors are then concatenated into a single vector with dimensions of $1 \times 1 \times 512$.

Mapping Network: The mapping network comprises six fully connected layers, each incorporating Leaky-ReLU activation with a negative slope of 0.01. The concatenated latent vector is input into this network, which processes it to generate an integrated latent vector of the same dimensions as the input vector.

Generator: The generator employs transposed convolution blocks with instance normalization and Leaky-ReLU activation, using a negative slope of 0.01. Initially, the size of the latent vector is increased to $4 \times 4 \times 2048$ in the first layer. As the vector passes through each subsequent layer, its depth is halved while its spatial dimensions are doubled. Ultimately, the generator produces fashion item images with dimensions of $256 \times 256 \times 3$.

Discriminator: The discriminator employs two parallel streams, each composed of five residual blocks with instance normalization and Leaky-ReLU activation, utilizing a negative slope of 0.01. Both the generated (fake) images and the ground-truth images, along with the given set of compatible fashion items, are input into the discriminator. The output consists of compatibility features and classification logits. During training, the network learns to understand the compatibility between different fashion items and their respective categories.

3.3. Fashion Compatibility Batch Algorithm

The Polyvore Dataset for Fill In the Blank task [32] comprises only positive examples, meaning that all outfits in the dataset are matched sets of fashion items. To enable the model to learn from mismatched sets of fashion items, randomly generated outfits are created on the fly and fed to the model during training. For these randomly generated outfits, black images are used as the ground truth.

As demonstrated in Algorithm 1, fashion images featuring mismatched outfits are sampled across all categories when the sample probability falls below the mismatch probability threshold; however, in [18], exclusively the mismatched outfits from the Polyvore Dataset for Fashion Compatibility Classification task [32] are utilized to train the scoring network in understanding fashion compatibility.

Algorithm 1 Pseudocode for fashion compatibility batch algorithm.

```

1: Input:  $H, W \leftarrow$  height and width of an image
2: Input:  $mismatch\_probability \leftarrow$  float from 0 to 1, hyper-parameter
3: Input:  $fashion\_image\_list \leftarrow$  list of fashion items
4: Input:  $outfit\_data\_list \leftarrow$  list of suitable outfit list
5:  $prob \leftarrow \text{random.uniform}(0.0, 1.0)$ 
6: if  $prob < mismatch\_probability$  then
7:    $target\_image \leftarrow \text{torch.zeros}(3, H, W)$ 
8:    $source\_image\_list \leftarrow \text{random.sample}(fashion\_image\_list, 4)$ 
9: else
10:   $idx \leftarrow$  randomly sampled index from  $outfit\_data\_list$ 
11:   $target\_image, source\_image\_list \leftarrow outfit\_data\_list[idx]$ 
12: end if
13: return  $target\_image, source\_image\_list$ 

```

3.4. Objectives

The λ_{g_adv} , λ_{gt} , and $\lambda_{perceptual}$ are coefficients of loss functions, respectively. $\mathcal{L}_{GT}$ and $\mathcal{L}_{perceptual}$ are l1 loss function between target and fake images, and the perceptual loss function mentioned in [38].

The λ_{d_adv} , λ_{cls} , and λ_{gp} correspond to the adversarial, classification, and gradient penalty [39] coefficients of loss functions, respectively. The term CE is the cross-entropy loss function. $D_{cls}(x)$ and y_{true} represent the logit for classification and the ground-truth category, respectively.

G and D denote the generator and discriminator, respectively. z_{combined} is the latent vector obtained from the mapping network, while cat denotes the category of the fashion item, as shown in Equation (2).

The variables x and x_{source} denote the target and source images, respectively. The term $\hat{x}$ represents a sample taken from between the target image x and the generated image $G(z_{\text{combined}}, \text{cat})$, as shown in Equation (1). $\nabla_{\hat{x}} D(\hat{x}, x_{\text{source}})$ is a gradient of given input $\hat{x}$ and x_{source} . Gradient penalty [39] enforces Lipschitz continuity by penalizing gradient norm deviations, stabilizing training and ensuring reliable convergence.

Section 3.5 provides a detailed explanation of the construction of Equations (2) and (3) in the DFDGAN training algorithm, as well as the training processes of the generator and discriminator. The model utilizes two primary loss functions, Equations (2) and (3), for the generator and discriminator.

$$\begin{aligned} \mathcal{L}_G = \lambda_{\text{g_adv}} \cdot \frac{1}{2} \mathbb{E}_{z \sim p(z)} \left[(D(G(z_{\text{combined}}, \text{cat}), x_{\text{source}}) - a)^2 \right] \\ + \lambda_{\text{gt}} \cdot \mathcal{L}_{\text{GT}} + \lambda_{\text{perceptual}} \cdot \mathcal{L}_{\text{perceptual}} \\ + \lambda_{\text{cls}} \cdot \mathbb{E}_{z \sim p(z)} \text{CE}(D_{\text{cls}}(G(z_{\text{combined}}, \text{cat})), y_{\text{true}}) \end{aligned} \quad (2)$$

$$\begin{aligned} \mathcal{L}_D = \lambda_{\text{d_adv}} \cdot \mathbb{E}_{x \sim p_{\text{data}}(x)} \left[(D(x, x_{\text{source}}) - b)^2 \right] + \lambda_{\text{cls}} \cdot \mathbb{E}_{x \sim p_{\text{data}}(x)} \text{CE}(D_{\text{cls}}(x), y_{\text{true}}) \\ + \lambda_{\text{d_adv}} \cdot \mathbb{E}_{z \sim p(z)} \left[(D(G(z_{\text{combined}}, \text{cat}), x_{\text{source}}) - c)^2 \right] \\ + \lambda_{\text{gp}} \cdot \mathbb{E}_{\hat{x} \sim p_{\text{interp}}} \left[(\|\nabla_{\hat{x}} D(\hat{x}, x_{\text{source}})\|_2 - 1)^2 \right] \end{aligned} \quad (3)$$

The generator loss function, $\mathcal{L}_G$, incorporates adversarial loss, ground-truth (GT) loss, classification loss, and perceptual loss. The adversarial loss, based on LSGAN-GP [40], guarantees that the generated images are indistinguishable from real ones. The GT loss measures the L1 distance between the generated and target images. The classification function ensures that the image generated is the same as the input category vector, while the perceptual loss, as applied in style transfer [38], ensures high-level feature similarity. The encoder and mapping network are integrated with the generator to produce a combined latent vector for image generation.

The discriminator loss function, $\mathcal{L}_D$, comprises adversarial loss terms for distinguishing between real and generated images, a classification loss using cross-entropy for categorizing fashion items, and a gradient penalty term to enforce the Lipschitz constraint. These components collectively improve the model's capacity to generate realistic and contextually accurate fashion images.

3.5. Training Algorithm

Algorithm 2 provides a detailed explanation of the training procedure for DFDGAN, aligning with the process visually represented in Figure 1. Equation (1) is a generation process of a synthesis image of given multiple categories of fashion images. Equations (2) and (3) are loss functions for a generator and discriminator in our framework.

Algorithm 2 DFDGAN training algorithm.

```

1: Input:  $E$ : Encoder,  $M$ : Mapping Network,  $G$ : Generator,  $D$ : Discriminator
2: Input:  $adv\_loss$ : Adversarial loss function,  $cls\_loss$ : Cross-entropy loss function
3: Input:  $gradient\_penalty$ : Gradient penalty function from WGAN-GP
4: Input:  $\lambda_{g\_adv}$ : Weight for adversarial loss for Generator
5: Input:  $\lambda_{cls}$ : Weight for classification loss
6: Input:  $\lambda_{perceptual}$ : Weight for perceptual loss
7: Input:  $\lambda_{gt}$ : Weight for ground truth loss
8: Input:  $\lambda_{d\_adv}$ : Weight for adversarial loss for Discriminator
9: Input:  $\lambda_{gp}$ : Weight for gradient penalty
10:  $target\_cat$ : category of target fashion image
11: for  $e$  in epochs do
12:   for  $target\_image, target\_cat, source\_image\_list$  in  $data\_loader$  do
13:      $latent\_vector\_list \leftarrow []$ 
14:     for  $source\_image$  in  $source\_image\_list$  do
15:        $latent\_vector\_z \leftarrow E(source\_image)$ 
16:        $latent\_vector\_list.append(latent\_vector\_z)$ 
17:     end for
18:
19:      $latent\_vector \leftarrow \text{concat}(latent\_vector\_list)$ 
20:      $combined\_latent\_vector \leftarrow M(latent\_vector)$ 
21:      $fake\_image \leftarrow G(combined\_latent\_vector, target\_cat)$  ▷ Equation (1)
22:
23:      $target\_feature, fake\_feature \leftarrow VGG(target\_image), VGG(fake\_image)$ 
24:      $fake\_score, cls\_logit \leftarrow D(fake\_image, source\_image\_list)$ 
25:      $g\_loss\_adv \leftarrow \lambda_{g\_adv} \cdot adv\_loss(fake\_score, \text{torch.ones\_like}(fake\_score))$ 
26:      $g\_loss\_cls \leftarrow \lambda_{cls} \cdot cls\_loss(cls\_logit, target\_cat)$ 
27:      $g\_loss\_perceptual \leftarrow \lambda_{perceptual} \cdot l1(fake\_feature, target\_feature)$ 
28:      $g\_loss\_gt \leftarrow \lambda_{gt} \cdot l1(fake\_image, target\_image)$ 
29:      $g\_loss \leftarrow g\_loss\_adv + g\_loss\_cls + g\_loss\_perceptual + g\_loss\_gt$  ▷
30:     Equation (2)
31:     Perform G backpropagation
32:
33:      $target\_score, cls\_logit \leftarrow D(target\_image, source\_image\_list)$ 
34:      $fake\_score, \_ \leftarrow D(fake\_image.detach(), source\_image\_list)$ 
35:      $d\_loss\_real \leftarrow \lambda_{d\_adv} \cdot adv\_loss(target\_score, \text{torch.ones\_like}(real\_score))$ 
36:      $d\_loss\_fake \leftarrow \lambda_{d\_adv} \cdot adv\_loss(fake\_score, \text{torch.zeros\_like}(fake\_score))$ 
37:      $d\_loss\_adv \leftarrow d\_loss\_real + d\_loss\_fake$ 
38:
39:      $d\_loss\_cls \leftarrow \lambda_{cls} \cdot cls\_loss(cls\_logit, target\_cat)$ 
40:      $\alpha \leftarrow \text{random.uniform}(0.0, 1.0)$ 
41:      $x\_hat \leftarrow \alpha \cdot target\_image + (1 - \alpha) \cdot fake\_image$ 
42:      $gp\_score, \_ \leftarrow D(x\_hat, source\_image\_list)$ 
43:      $d\_loss\_gp \leftarrow \lambda_{gp} \cdot gradient\_penalty(gp\_score, x\_hat)$ 
44:      $d\_loss \leftarrow d\_loss\_adv + d\_loss\_cls + d\_loss\_gp$  ▷ Equation (3)
45:     Perform D backpropagation
46:   end for
47: end for

```

4. Experimental Evaluation

4.1. Dataset and Pre-Processing

The Polyvore Dataset, as described in [32] and the Apache-2.0 License, contains a collection of fully assembled and well-matched fashion outfits. Each outfit includes upper and lower garments, shoes, bags, accessories, and other fashion items.

In the Polyvore Dataset, the outfit data often include images of non-fashion items or individuals wearing fashion items. To refine the dataset, images featuring people are filtered out using YOLO v3 [41], ensuring that only fashion items are displayed. Additionally, outfits containing non-fashion items or images that are not wearable by a person are removed, resulting in a dataset that clearly focuses on individual fashion items.

In the outfit data, the first five fashion items are treated as a complete outfit. For each outfit, one fashion item is designated as the target item, while the remaining four are considered source items. Consequently, each outfit is viewed as consisting of five distinct fashion items. In each row, the first image on the left is the target image, and the four images to the right are the source images, as illustrated in Figure 2.

The dataset comprises 3141 outfits, which are allocated into training, validation, and test sets in an 8:1:1 ratio, resulting in 12,560, 1570, and 1575 outfit data in the training, validation, and test sets, respectively.

Table 1 presents the statistics of the dataset. The tops category includes both outer and inner garments, while the “bottoms” category comprises various types of skirts and pants. The majority of the dataset consists of women’s outfits, primarily sourced from users in Western cultures.



Figure 2. Polyvore Dataset example. Each row of five images represents a single batch. The leftmost image depicts the ground truth, called a target, while the subsequent four images serve as input to DFDGAN and are referred to as sources 1 to 4.

Table 1. Dataset statistics.

	Outfits	Tops	Bottoms	Dresses	Shoes	Bags	Eyeglasses	Earrings
train	2512	4146	2214	346	2515	2349	396	594
val	314	515	281	37	315	285	53	37
test	315	519	286	38	317	294	47	74

4.2. Implementation Details

In this study, a batch size of 16 is used, with the learning rates for the generator, mapping network, and encoder set at 3×10^{-5} , while the learning rate for the discriminator is set at 1×10^{-6} . The Adam optimizer [42] is employed during training with β_1 : 0.5, β_2 : 0.999. No data augmentation is applied to the images. The fifth convolutional layer of the VGG19 network [1] is utilized for the perceptual loss function. The generator and discriminator are adversarially trained for 1000 epochs.

Training and evaluation were conducted on an NVIDIA(R), Santa Clara, CA, USA A100-PCIE-80 GB GPU and an Intel(R), Santa Clara, CA, USA Xeon(TM) Processor (Cascade-lake), using PyTorch v2.1.0, CUDA v11.8, CuDNN v8.7, and Python 3.9. The coefficients λ_{g_adv} , λ_{gt} , $\lambda_{perceptual}$, λ_{d_adv} , λ_{cls} , and λ_{gp} are set to 1, 1, 50, 1, 1, and 10, respectively. The parameters a, b, and c are 1, 1 and 0 in Equations (2) and (3) and Algorithm 2. The mismatch_probability is 0.35 in Algorithm 1.

5. Evaluation

5.1. Baseline and Metrics

As far as we are aware, OutfitGAN could serve as a baseline for this research. However, the code, detailed configuration, and dataset are not publicly available, making a fair comparison impossible. Therefore, a fair comparison cannot be conducted. To show the achievement of this study, pix2pix [23] and CycleGAN [24] are selected as the baselines as conducted in [13] in this study.

For the pix2pix [23] and CycleGAN [24], the input images are combined along the channel dimension. The identity loss function is removed in CycleGAN because the target and source images do not share identical shapes. Additionally, the two generators in CycleGAN have been modified to accommodate the specific requirements of this task.

To evaluate this method, Inception Score (IS), Fréchet Inception Distance (FID), and Learned Perceptual Image Patch Similarity (LPIPS) [43–45] are utilized. For calculating IS and FID, an Inception-v3 [46] is trained to classify tops, bottoms, dresses, earrings, bags, shoes, and eyeglasses using the original Polyvore Dataset [32].

5.2. Inception Score

IS is a popular metric to evaluate the quality and diversity of images generated by generative models, particularly GANs. It relies on a pre-trained Inception network to compute class probabilities for the generated images.

A higher IS represents that the generative model produces high-quality, diverse fashion images. It reflects outputs that are distinct, visually coherent, and confidently recognized by an Inception model. High IS also suggests that the model captures variations in style and design across categories, generating a wide range of fashion items rather than memorizing some fashion items.

Equation (4) describes the formula of IS. x is a generated image, p_g is the distribution of the generated images, $\mathbb{E}$ is the expectation operator averaging over the distribution of generated images, $\text{KL}(\cdot \parallel \cdot)$ is the Kullback–Leibler divergence, $p(y|x)$ is the class probability distribution for the generated image x , and $p(y)$ is the marginal distribution of the class probabilities over all generated images.

$$\text{IS} = \exp\left(\mathbb{E}_{x \sim p_g}[\text{KL}(p(y|x) \parallel p(y))]\right) \quad (4)$$

5.3. Fréchet Inception Distance

FID is a metric to evaluate the quality and realism of images generated by a model by comparing their statistical distribution to that of real images.

A lower FID indicates that the quality of the generated fashion images is closer to real data in terms of both appearance and distribution. It compares the mean and covariance of features from generated and real images, capturing realism and diversity. A low FID suggests realistic outputs with minimal artifacts, effectively representing the variability of input categories.

Equation (5) describes the formula of FID. μ_r and μ_g are the mean feature vectors of the real and generated image distributions, respectively, Σ_r and Σ_g are the covariance matrices of the real and generated image distributions, and $\|\cdot\|^2$ represents the squared Euclidean distance, while $\text{Tr}(\cdot)$ is the trace of the matrix.

$$\text{FID} = \|\mu_r - \mu_g\|^2 + \text{Tr}\left(\Sigma_r + \Sigma_g - 2(\Sigma_r \Sigma_g)^{\frac{1}{2}}\right) \quad (5)$$

5.4. Learned Perceptual Image Patch Similarity

LPIPS is a metric to assess the perceptual similarity between two images by comparing their feature representations.

The formula computes the squared Euclidean distance between feature representations extracted from a pre-trained network, with lower LPIPS values indicating greater perceptual similarity. LPIPS better aligns with human perception than pixel-based metrics, making it useful for evaluating image quality in generative models. A lower LPIPS for generated fashion items suggests a higher visual similarity to real fashion images, reflecting improved realism.

Equation (6) describes the formula of LPIPS. I and J are the two images being compared, $f_l(I)$ and $f_l(J)$ are the feature representations of these images at layer l of a pre-trained network, $\|\cdot\|_2^2$ is the squared Euclidean distance between the feature vectors, α_l is the learned weight for each layer l , and the sum over all layers l captures the total perceptual difference between the two images.

$$\text{LPIPS}(I, J) = \sum_l \alpha_l \|f_l(I) - f_l(J)\|_2^2 \quad (6)$$

5.5. Comparison with the Baselines

Our framework surpasses the baseline models across all metrics. As shown in Table 2, the architectures designed for one-to-one image translation are not applicable for this task, a point that will be further discussed in Section 6.1. The IS for the ground-truth image is 4.49 ± 0.20 .

Table 2. Method comparison with the baselines.

	IS	FID	LPIPS
DFDGAN	3.87 ± 0.18	80.9	0.642
pix2pix	1.61 ± 0.04	226.9	0.74
CycleGAN	1.56 ± 0.14	361.8	0.83

5.6. Architecture Configuration Study

As demonstrated in Table 3, adopting a single encoder outperforms the use of multiple encoders. The single encoder benefits from being trained on a larger set of images compared to each of the four individual encoders, resulting in more robust feature extraction. When four images are combined along the channel axis, the encoder tends to learn overlapping areas across multiple categories of fashion items during feature extraction. This overlap results in reduced performance, as demonstrated in Table 3. The perceptual loss leverages high-level features from a VGG19 network with batch normalization [1,47], trained on the Polyvore Dataset, to enhance image quality.

In Figure 3, the left and the right results show the latent vector visualization with and without the mapping network, respectively, using t-SNE [48]. In the visualization without the mapping network (right), data points for tops and shoes are dispersed across the embedding space, with significant overlap between categories. In contrast, the visualization with the mapping network (left) demonstrates a more disentangled and organized embedding space.

Unlike the approach in [20], the mapping network in this study prevents the generator from overfitting to specific categories within the concatenated latent vector. The mapping network contributes not only to constructing a disentangled embedding space but also improving performance, as shown in Table 3.

Table 3 demonstrates that a latent vector dimension of 512 is optimal for the model, balancing performance with resource usage, including training time and memory. Using the fifth convolutional layer of VGG19 with batch normalization, and weights pre-trained on the Polyvore Dataset [32], enhances image quality. However, employing more deeper layers results in a decline in performance.

Table 3. Architecture configuration study of DFDGAN models. DFDGAN is our final model. DFDGAN_{multi} uses four encoders instead of a single encoder. DFDGAN_{concat} uses a single encoder, but four images are concatenated along the channel axis. DFDGAN_{w/o_Mapping} does not use the mapping network. DFDGAN_{vanila_env} uses a vanilla encoder consisting only of 2D convolution blocks with the same normalization and non-linearity function. DFDGAN_{layer_number} indicates the VGG convolution layer number for perceptual loss. DFDGAN_{GT} does not use perceptual loss. DFDGAN_{128, 256, 1024, 2048} represents the dimension of the latent vector for the generator. DFDGAN_{wgan-gp} uses the WGAN-GP loss for training. DFDGAN_{multi_D} uses a multi-discriminator that uses only a single image with a classification layer.

	IS	FID	LPIPS		IS	FID	LPIPS
DFDGAN	3.87 ± 0.18	80.9	0.642	DFDGAN ₁₂₈	2.03 ± 0.07	189.8	0.71
DFDGAN _{multi}	3.57 ± 0.20	93.9	0.687	DFDGAN ₂₅₆	3.29 ± 0.09	116.3	0.700
DFDGAN _{concat}	3.55 ± 0.06	100.0	0.700	DFDGAN ₁₀₂₄	3.39 ± 0.07	98.3	0.689
DFDGAN _{GT}	3.37 ± 0.12	132.2	0.699	DFDGAN ₂₀₄₈	3.78 ± 0.11	82.8	0.676
DFDGAN _{w/o_Mapping}	3.38 ± 0.11	119.3	0.660	DFDGAN _{wgan-gp}	3.37 ± 0.15	96.8	0.682
DFDGAN _{vanila_env}	3.35 ± 0.12	107.6	0.680	DFDGAN _{multi_D}	3.57 ± 0.09	96.7	0.6557
DFDGAN _{layer_9}	3.20 ± 0.06	124.7	0.711	DFDGAN _{layer_13}	2.63 ± 0.11	154.4	0.770
DFDGAN _{layer_5,9}	3.41 ± 0.12	96.3	0.696	DFDGAN _{layer_5,9,13}	3.39 ± 0.08	96.5	0.698

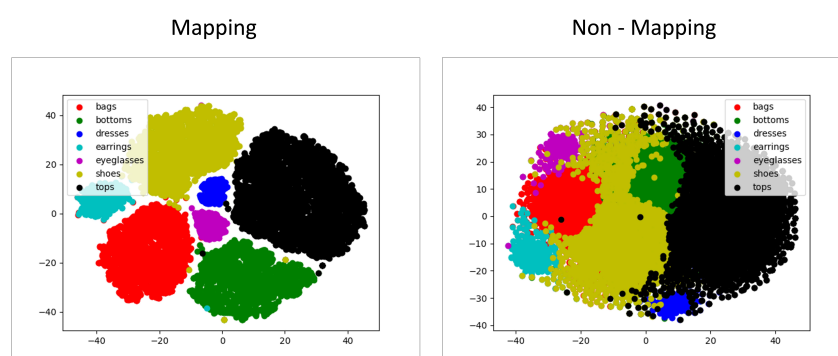


Figure 3. TSNE visualization; the left one is from the method with the mapping network and the right one is from the method without the mapping network.

5.7. Quantitative Result

DFDGAN generates suitable fashion items, including upper and lower garments, shoes, bags, and accessories, based on given source items, effectively capturing the harmony between the generated image and the source items by considering the style and attributes of the source items, as illustrated in Figure 4.



Figure 4. DFDGAN image generation result. Sources 1 to 4 represent the order of fashion items of multiple categories that are inputs to the model. DFDGAN represents the images generated by our framework.

6. Method Analysis

6.1. Architectural Difference with the Baselines

As illustrated in Figure 5, pix2pix [23] fails to translate source items into a coherent target item. CycleGAN [24] is unsuccessful in generating fashion items across all cases. The source items are merely overlaid onto the generated image by CycleGAN, likely due to the requirement that the target item be mapped to the source items under cycle-consistency. As indicated in Table 2, the architectures of pix2pix and CycleGAN [23,24] are unsuitable for this particular task.



Figure 5. Image generation comparison. Sources 1 to 4 indicate the sequence of fashion items from multiple categories fed into the models. DFDGAN, pix2pix, and CycleGAN represent the images generated by each respective model.

6.2. Outfit Space Exploration

Latent space exploration, also known as latent space walking [19], is a technique used to assess the effectiveness of a GAN's training. In this study, the outfit space is explored using new source items obtained through an unsupervised clustering algorithm and deep features. To obtain new source items, all fashion items are clustered using the BiT and

FINCH algorithms [49,50]. For each dataset entry, one of the source items is replaced with a different item from the same category within the cluster.

If the training is successful, the generated image from the new source items will exhibit different styles and details while remaining as plausible as the image generated from the original source items. In the embedding space, the new data points should be situated near the original data points.

As shown in Figure 6, the eyeglasses and bags are generated from several source items obtained by above approach. For the left side of Figure 6, three shoes on source 3 are all different, and one top on source 1 is different. The generated eyeglasses have different colors and styles. For the right side of Figure 6, three bottom clothes on source 2 are all different. The generated bags have different colors and styles. The result in Figure 6 means that DFDGAN understands the harmony of multiple fashion items, recognizes each fashion item, and produces a plausible fashion item with various colors and shapes.



Figure 6. Outfit walking. Sources 1 to 4 represent the order of fashion items of multiple categories that are input into the model. DFDGAN represents the generated images.

7. Discussion

The failure of the architectures in pix2pix [23] and CycleGAN [24] to generate plausible fashion images can be attributed to the first transposed convolution layer, which halves depth while its spatial dimensions are doubled, and a residual path is identical to the input feature in the residual block.

The first transposed convolution of our generator increases the number of channels from 512 to 2048 and expands the spatial dimensions from 1×1 to 4×4 by a factor of four squared. This substantial increase in transformation capacity is intended to map the relationship between the target image and the latent vector.

The identical residual path is not utilized as it disturbs the transformation from a combined feature vector to a target image. Given that the features input to each convolution module of the generator are distinct from the images to be generated, it is essential that the visual features from previous layers do not persist through the convolutional modules. If the identical residual path is used, learning based solely on residuals would be insufficient to achieve meaningful visual transformations.

DFDGAN operates effectively only when using Instance Normalization [51]. Batch Normalization [47] does not contribute to successful network training due to the high variance observed along the batch axis. Each fashion item exhibits unique shapes, styles, colors, and sizes, making them visually dissimilar. As a result, when batch normalization was applied to our framework, it led to a significant increase in variance, which in turn caused learning instability and gradient divergence.

The advantage of our framework is that training difficulty is low compared to [18] because the framework is the end-to-end manner which has only a single generator and discriminator, and a prerequisite network is not required to obtain fashion compatibility.

The fashion compatibility batch algorithm in this study provides diverse mismatched outfits, but the fashion compatibility network in [18] is only trained with the Polyvore Dataset [32] with 3000 compatible outfits and 4000 incompatible outfits.

The limitations of our framework are that the generated images tend to be blurred, and four images are required to get a compatible fashion image. The findings from this study offer valuable insights into the structural design of many-to-one image generation models, with a particular focus on fashion compatibility and normalization.

8. Conclusions

This study introduces a many-to-one image-translation-based fashion item generation framework that utilizes complementary fashion item images from multiple categories, moving beyond previous one-to-one image translation approaches that rely on a single image from a single category, to better reflect real-world fashion scenarios.

Experimental results indicate that the novel architecture exceeds the baseline network architectures with respect to image quality and evaluation metrics. DFDGAN demonstrates superior performance compared to pix2pix [23] and CycleGAN [24] across all evaluation metrics. It achieves a significantly higher IS (3.87 ± 0.18 vs. 1.61 ± 0.04 and 1.56 ± 0.14), a markedly lower FID (80.9 vs. 226.9 and 361.8), and an improved LPIPS score (0.642 vs. 0.74 and 0.83), indicating enhanced image quality and perceptual similarity. These findings underscore the efficacy of DFDGAN's network architecture, specifically designed to excel in many-to-one image translation tasks.

DFDGAN produces images at a resolution of 256×256 pixels, which is four times larger in pixel count than the 128×128 images produced by OutfitGAN [18].

Training OutfitGAN requires a separate fashion compatibility-aware network, along with the simultaneous training of three generators and three discriminators. Alternatively, DFDGAN eliminates the need for an auxiliary network and simplifies the process by training a single generator and discriminator.

The fashion-compatibility-aware network in OutfitGAN is trained with a limited dataset of 7000 outfits, consisting of 4000 incompatible and 3000 compatible outfits. In contrast, DFDGAN leverages the fashion compatibility batch algorithm, which provides the model with an unlimited number of unsuitable outfits. This capability allows DFDGAN to achieve superior performance in learning and handling fashion compatibility compared to OutfitGAN.

Our future research will aim to enhance image quality further. The introduction of a novel residual convolutional block is expected to enhance sharpness, and texture fidelity, and reduce artifacts in fashion images. A refinement network will be incorporated to effectively eliminate noise, correct artifacts, and refine misgenerated parts of the image. To achieve a more disentangled outfit representation, a new feature concatenating approach will be applied to the mapping network, facilitating more accurate and meaningful feature space.

Author Contributions: Conceptualization, J.J.; methodology, J.J.; software, J.J.; validation, J.J.; formal analysis, J.J.; investigation, J.J.; resources, J.P.; data curation, J.J.; writing—original draft preparation, J.J.; writing—review and editing, J.J., H.K., and J.P.; visualization, J.J.; supervision, J.P.; project administration, H.K. and J.P.; funding acquisition, J.P. All authors have read and agreed to the published version of the manuscript.

Funding: This work was partially supported by the Korea government (MSIT) through Institute of Information & communications Technology Planning & evaluation (IITP) (No. RS-2022-00187238), by the Basic Research Laboratory through the National Research Foundation (NRF) (No. RS-2023-00221365) and by Culture, Sports and Tourism R&D Program through the Korea Creative Content Agency (KOCCA) (No. RS-2024-00340342, Contribution Rate: 50%).

Data Availability Statement: The original dataset is available in [32] and can be accessed via <https://github.com/xthan/polyvore> (accessed on 12 October 2024) under the Apache-2.0 License. The modified dataset will be made publicly available through a GitHub repository.

Conflicts of Interest: Author Jaewon Jung was employed by the company Huraypositive Corp. The remaining authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

References

1. Simonyan, K.; Zisserman, A. Very Deep Convolutional Networks for Large-Scale Image Recognition. In Proceedings of the 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, 7–9 May 2015; Conference Track Proceedings; Bengio, Y., LeCun, Y., Eds.; Computational and Biological Learning Society: San Diego, CA, USA, 2015. Available online: <https://ora.ox.ac.uk/objects/uuid:60713f18-a6d1-4d97-8f45-b60ad8aebbce> (accessed on 12 October 2024).
2. Goodfellow, I.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; Bengio, Y. Generative Adversarial Nets. In *NeurIPS*; Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N.D., Weinberger, K.Q., Eds.; Curran Associates, Inc.: Red Hook, NY, USA, 2014; pp. 2672–2680.
3. Han, X.; Wu, Z.; Wu, Z.; Yu, R.; Davis, L.S. VITON: An Image-Based Virtual Try-on Network. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 7543–7552. [CrossRef]
4. Choi, S.; Park, S.; Lee, M.; Choo, J. VITON-HD: High-Resolution Virtual Try-On via Misalignment-Aware Normalization. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Virtual, 19–25 June 2021; pp. 14131–14140.
5. Neuberger, A.; Borenstein, E.; Hilleli, B.; Oks, E.; Alpert, S. Image Based Virtual Try-On Network From Unpaired Data. In Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Virtual, 14–19 June 2020; pp. 5183–5192. [CrossRef]
6. Dong, H.; Liang, X.; Zhang, Y.; Zhang, X.; Shen, X.; Xie, Z.; Wu, B.; Yin, J. Fashion Editing With Adversarial Parsing Learning. In Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Virtual, 14–19 June 2020; pp. 8117–8125. [CrossRef]
7. Hsiao, W.L.; Katsman, I.; Wu, C.Y.; Parikh, D.; Grauman, K. Fashion++: Minimal Edits for Outfit Improvement. In Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Republic of Korea, 27 October–2 November 2019; pp. 5046–5055. [CrossRef]
8. Han, X.; Wu, Z.; Huang, W.; Scott, M.; Davis, L. FiNet: Compatible and Diverse Fashion Image Inpainting. In Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Republic of Korea, 27 October–2 November 2019; pp. 4480–4490. [CrossRef]
9. Kwon, Y.R.; Kim, S.; Yoo, D.; Eui Yoon, S. Coarse-to-Fine Clothing Image Generation with Progressively Constructed Conditional GAN. In Proceedings of the VISIGRAPP, Prague, Czech Republic, 5–27 February 2019.
10. Zhang, H.; Sun, Y.; Liu, L.; Xu, X. CascadeGAN: A category-supervised cascading generative adversarial network for clothes translation from the human body to tiled images. *Neurocomputing* **2020**, *382*, 148–161. [CrossRef]
11. Zhang, H.; Sun, Y.; Liu, L.; Wang, X.; Li, L.; Liu, W. ClothingOut: A category-supervised GAN model for clothing segmentation and retrieval. *Neural Comput. Appl.* **2020**, *32*, 4519–4530. [CrossRef]
12. Jiang, S.; Fu, Y. Fashion Style Generator. In Proceedings of the IJCAI, Melbourne, Australia, 19–25 August 2017; pp. 3721–3727. [CrossRef]
13. Chen, L.; Tian, J.; Li, G.; Wu, C.H.; King, E.K.; Chen, K.T.; Hsieh, S.H.; Xu, C. TailorGAN: Making User-Defined Fashion Designs. In Proceedings of the 2020 IEEE Winter Conference on Applications of Computer Vision (WACV), Snowmass Village, CO, USA, 1–5 March 2020; pp. 3230–3239. [CrossRef]
14. Shih, Y.S.; Chang, K.Y.; Lin, H.T.; Sun, M. Compatibility family learning for item recommendation and generation. In Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence and Thirtieth Innovative Applications of Artificial Intelligence Conference and Eighth AAAI Symposium on Educational Advances in Artificial Intelligence, New Orleans, LA, USA, 2–7 February 2018.
15. Lin, Y.; Ren, P.; Chen, Z.; Ren, Z.; Ma, J.; de Rijke, M. Improving Outfit Recommendation with Co-supervision of Fashion Generation. In Proceedings of the The World Wide Web Conference, New York, NY, USA, 13–17 May 2019; WWW '19; pp. 1095–1105. [CrossRef]
16. Liu, J.; Song, X.; Chen, Z.; Ma, J. MGCM: Multi-modal generative compatibility modeling for clothing matching. *Neurocomputing* **2020**, *414*, 215–224. [CrossRef]

17. Liu, L.; Zhang, H.; Zhou, D. Clothing generation by multi-modal embedding: A compatibility matrix-regularized GAN model. *Image Vis. Comput.* **2021**, *107*, 104097. [[CrossRef](#)]
18. Moosaei, M.; Lin, Y.; Akhazhanov, A.; Chen, H.; Wang, F.; Yang, H. OutfitGAN: Learning Compatible Items for Generative Fashion Outfits. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops, New Orleans, LA, USA, 18–24 June 2022; pp. 2273–2277.
19. Radford, A.; Metz, L.; Chintala, S. Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks. In Proceedings of the ICLR, San Juan, Puerto Rico, 2–4 May 2016.
20. Karras, T.; Laine, S.; Aila, T. A Style-Based Generator Architecture for Generative Adversarial Networks. In Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 15–20 June 2019; pp. 4396–4405. [[CrossRef](#)]
21. Karras, T.; Laine, S.; Aittala, M.; Hellsten, J.; Lehtinen, J.; Aila, T. Analyzing and Improving the Image Quality of StyleGAN. In Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 13–19 June 2020; pp. 8107–8116. [[CrossRef](#)]
22. Karras, T.; Aittala, M.; Laine, S.; Härkönen, E.; Hellsten, J.; Lehtinen, J.; Aila, T. Alias-Free Generative Adversarial Networks. In Proceedings of the Advances in Neural Information Processing Systems, Virtual, 6–14 December 2021; Ranzato, M., Beygelzimer, A., Dauphin, Y., Liang, P., Vaughan, J.W., Eds.; Curran Associates, Inc.: Red Hook, NY, USA, 2021; Volume 34, pp. 852–863.
23. Isola, P.; Zhu, J.Y.; Zhou, T.; Efros, A.A. Image-to-Image Translation with Conditional Adversarial Networks. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 5967–5976. [[CrossRef](#)]
24. Zhu, J.Y.; Park, T.; Isola, P.; Efros, A.A. Unpaired Image-to-Image Translation Using Cycle-Consistent Adversarial Networks. In Proceedings of the 2017 IEEE International Conference on Computer Vision (ICCV), Venice, Italy, 22–27 October 2017; pp. 2242–2251. [[CrossRef](#)]
25. Kim, T.; Cha, M.; Kim, H.; Lee, J.K.; Kim, J. Learning to Discover Cross-domain Relations with Generative Adversarial Networks. In Proceedings of the 34th International Conference on Machine Learning, Sydney, Australia, 6–11 August 2017; pp. 1857–1865.
26. Choi, Y.; Choi, M.; Kim, M.; Ha, J.W.; Kim, S.; Choo, J. StarGAN: Unified Generative Adversarial Networks for Multi-domain Image-to-Image Translation. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 8789–8797. [[CrossRef](#)]
27. Choi, Y.; Uh, Y.; Yoo, J.; Ha, J.W. StarGAN v2: Diverse Image Synthesis for Multiple Domains. In Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 13–19 June 2020; pp. 8185–8194. [[CrossRef](#)]
28. Hinz, T.; Fisher, M.; Wang, O.; Shechtman, E.; Wermter, S. CharacterGAN: Few-Shot Keypoint Character Animation and Reposing. In Proceedings of the WACV, Waikoloa, HI, USA, 4–8 January 2022; pp. 1988–1997.
29. Ge, C.; Song, Y.; Ge, Y.; Yang, H.; Liu, W.; Luo, P. Disentangled Cycle Consistency for Highly-Realistic Virtual Try-On. In Proceedings of the CVPR, Virtual, 19–25 June 2021; pp. 16928–16937.
30. Gafni, O.; Ashual, O.; Wolf, L. Single-Shot Freestyle Dance Reenactment. In Proceedings of the CVPR, Virtual, 19–25 June 2021; pp. 882–891.
31. Ronneberger, O.; Fischer, P.; Brox, T. U-Net: Convolutional Networks for Biomedical Image Segmentation. In Proceedings of the MICCAI, Munich, Germany, 5–9 October 2015; Lecture Notes in Computer Science; Springer: Berlin/Heidelberg, Germany, 2015; Volume 9351, pp. 234–241.
32. Han, X.; Wu, Z.; Jiang, Y.G.; Davis, L.S. Learning Fashion Compatibility with Bidirectional LSTMs. In Proceedings of the 25th ACM International Conference on Multimedia, Mountain View, CA, USA, 23–27 October 2017; MM '17, pp. 1078–1086. [[CrossRef](#)]
33. Hochreiter, S.; Schmidhuber, J. Long Short-Term Memory. *Neural Comput.* **1997**, *9*, 1735–1780. [[CrossRef](#)]
34. Vasileva, M.I.; Plummer, B.A.; Dusad, K.; Rajpal, S.; Kumar, R.; Forsyth, D. Learning Type-Aware Embeddings for Fashion Compatibility. In Proceedings of the Computer Vision—ECCV 2018, Munich Germany, 8–14 September 2018; Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y., Eds.; Springer Science+Business Media: Cham, Switzerland, 2018; pp. 405–421.
35. Veit, A.; Belongie, S.; Karaletsos, T. Conditional Similarity Networks. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 1781–1789. [[CrossRef](#)]
36. Santoro, A.; Raposo, D.; Barrett, D.G.; Malinowski, M.; Pascanu, R.; Battaglia, P.; Lillicrap, T. A simple neural network module for relational reasoning. In Proceedings of the Advances in Neural Information Processing Systems, Long Beach, CA, USA, 4–9 December 2017; Guyon, I., Luxburg, U.V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., Garnett, R., Eds.; Curran Associates, Inc.: Red Hook, NY, USA, 2017; Volume 30.
37. Woo, S.; Park, J.; Lee, J.Y.; Kweon, I.S. CBAM: Convolutional Block Attention Module. In Proceedings of the Computer Vision—ECCV 2018: 15th European Conference, Munich, Germany, 8–14 September 2018; Proceedings, Part VII; Springer: Berlin/Heidelberg, Germany, 2018; pp. 3–19. [[CrossRef](#)]

38. Gatys, L.A.; Ecker, A.S.; Bethge, M. Image Style Transfer Using Convolutional Neural Networks. In *Proceedings of the IJCVPR*; IEEE Computer Society: Washington, DC, USA, 2016; pp. 2414–2423.
39. Gulrajani, I.; Ahmed, F.; Arjovsky, M.; Dumoulin, V.; Courville, A.C. Improved Training of Wasserstein GANs. In *Proceedings of the Advances in Neural Information Processing Systems*, Long Beach, CA, USA, 4–9 December 2017; Curran Associates, Inc.: Red Hook, NY, USA, 2017; Volume 30.
40. Mao, X.; Li, Q.; Xie, H.; Lau, R.Y.; Wang, Z.; Smolley, S.P. On the Effectiveness of Least Squares Generative Adversarial Networks. *TPAMI* **2019**, *41*, 2947–2960. [[CrossRef](#)] [[PubMed](#)]
41. Redmon, J.; Farhadi, A. YOLOv3: An Incremental Improvement. *arXiv* **2018**, arXiv:1804.02767.
42. Kingma, D.; Ba, J. Adam: A Method for Stochastic Optimization. In *Proceedings of the International Conference on Learning Representations (ICLR)*, San Diego, CA, USA, 7–9 May 2015.
43. Salimans, T.; Goodfellow, I.; Zaremba, W.; Cheung, V.; Radford, A.; Chen, X.; Chen, X. Improved Techniques for Training GANs. In *NeurIPS*; Lee, D.D., Sugiyama, M., Luxburg, U.V., Guyon, I., Garnett, R., Eds.; Curran Associates, Inc.: Red Hook, NY, USA, 2016; pp. 2234–2242.
44. Heusel, M.; Ramsauer, H.; Unterthiner, T.; Nessler, B.; Hochreiter, S. GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium. In *Proceedings of the Advances in Neural Information Processing Systems*, Long Beach, CA, USA, 4–9 December 2017; Guyon, I., Luxburg, U.V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., Garnett, R., Eds.; Curran Associates, Inc.: Red Hook, NY, USA, 2017; Volume 30.
45. Zhang, R.; Isola, P.; Efros, A.A.; Shechtman, E.; Wang, O. The Unreasonable Effectiveness of Deep Features as a Perceptual Metric. In *Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Salt Lake City, UT, USA, 18–23 June 2018; pp. 586–595. [[CrossRef](#)]
46. Szegedy, C.; Vanhoucke, V.; Ioffe, S.; Shlens, J.; Wojna, Z. Rethinking the Inception Architecture for Computer Vision. In *Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, NV, USA, 26 June–1 July 2016; pp. 2818–2826. [[CrossRef](#)]
47. Ioffe, S.; Szegedy, C. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. In *Proceedings of the 32nd International Conference on Machine Learning*, Lille France, 7–9 July 2015; Volume 37, pp. 448–456.
48. Van der Maaten, L.; Hinton, G. Visualizing Data using t-SNE. *J. Mach. Learn. Res.* **2008**, *9*, 2579–2605.
49. Kolesnikov, A.; Beyer, L.; Zhai, X.; Puigcerver, J.; Yung, J.; Gelly, S.; Houlsby, N. Big Transfer (BiT): General Visual Representation Learning. In *Proceedings of the Computer Vision—ECCV 2020: 16th European Conference*, Glasgow, UK, 23–28 August 2020; *Proceedings, Part V*; Springer: Berlin/Heidelberg, Germany, 2020; pp. 491–507. [[CrossRef](#)]
50. Saquib Sarfraz, V.S.M.; Stiefel, R. Efficient Parameter-free Clustering Using First Neighbor Relations. In *Proceedings of the CVPR*, Long Beach, CA, USA, 15–20 June 2019; pp. 8934–8943.
51. Ulyanov, D.; Vedaldi, A.; Lempitsky, V.S. Instance Normalization: The Missing Ingredient for Fast Stylization. *arXiv* **2016**, arXiv:1607.08022.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.