



Article

Cyclic Consistent Image Style Transformation: From Model to System

Jun Peng ^{1,†} , Kaiyi Chen ^{2,†}, Yuqing Gong ³, Tianxiang Zhang ³ and Baohua Su ^{2,*} ¹ School of Education, City University of Macau, Macao 999078, China; junpeng@cityu.edu.mo² College of Chinese Language and Culture, Jinan University, Guangzhou 510610, China³ School of Computer Science, Zhuhai College of Science and Technology, Zhuhai 519041, China; gongyuqing1975@zcst.edu.cn

* Correspondence: subaohua@hwy.jnu.edu.cn

† These authors contributed equally to this work.

Abstract: Generative Adversarial Networks (GANs) have achieved remarkable success in various tasks, including image generation, editing, and reconstruction, as well as in unsupervised and representation learning. Despite their impressive capabilities, GANs are often plagued by challenges such as unstable training dynamics and limitations in generating complex patterns. To address these challenges, we propose a novel image style transfer method, named C3GAN, which leverages CycleGAN architecture to achieve consistent and stable transformation of image style. In this context, “image style” refers to the distinct visual characteristics or artistic elements, such as the color schemes, textures, and brushstrokes that define the overall appearance of an image. Our method incorporates cyclic consistency, ensuring that the style transformation remains coherent and visually appealing, thus enhancing the training stability and overcoming the generative limitations of traditional GAN models. Additionally, we have developed a robust and efficient image style transfer system by integrating Flask for web development and MySQL for database management. Our system demonstrates superior performance in transferring complex styles compared to existing model-based approaches. This paper presents the development of a comprehensive image style transfer system based on our advanced C3GAN model, effectively addressing the challenges of GANs and expanding application potential in domains such as artistic creation and cinematic special effects.

Keywords: CycleGAN; image style; deep learning; unsupervised learning



Citation: Peng, J.; Chen, K.; Gong, Y.; Zhang, T.; Su, B. Cyclic Consistent Image Style Transformation: From Model to System. *Appl. Sci.* **2024**, *14*, 7637. <https://doi.org/10.3390/app14177637>

Academic Editor: Byung-Gyu Kim

Received: 29 July 2024

Revised: 26 August 2024

Accepted: 27 August 2024

Published: 29 August 2024



Copyright: © 2024 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/>).

1. Introduction

Image style transfer is a powerful technique in the field of computer vision that involves modifying the visual style of an image while preserving its original content. Image style refers to the distinctive visual characteristics or artistic elements of an image, such as color schemes, textures, and brushstroke patterns, which collectively define its overall appearance.

This research focuses on image style transfer specifically in domains such as artistic creation [1] and cinematic special effects [2], where the ability to creatively manipulate visual styles is essential for innovation and expressive power. In artistic creation, style transfer can be used to apply the aesthetic of famous artworks to new images, enhancing creativity and providing novel visual experiences. In cinematic special effects, it enables the transformation of visual elements to achieve desired artistic effects or mood, thereby enhancing the visual storytelling of films and media.

Generative Adversarial Networks (GANs) have become a pivotal class of deep learning models, particularly noted for their prowess in unsupervised learning tasks. A typical GAN architecture comprises two adversarial components: a generator and a discriminator [3]. The generator is tasked with creating data samples that mimic real-world data, while the discriminator’s role is to distinguish between genuine samples and those produced by the

generator. This adversarial interplay forms the core of GAN training, where both models are iteratively optimized by learning from each other's outputs. Through this process, GANs have demonstrated remarkable ability in generating high-quality and diverse data samples across various domains, including images, videos, audio, and text [4].

Despite their promising potential, GANs are often beset by challenges such as unstable training dynamics [5,6] and limited diversity in the generated outputs [7]. The instability typically stems from the delicate balance required between the generator and discriminator during training. Should one model significantly outperform the other, the entire training process may destabilize, potentially leading to suboptimal models or even complete training failure. Additionally, GANs frequently struggle with producing a wide variety of samples, a limitation often manifested as mode collapse [8,9], where the generator repeatedly generates a narrow range of similar outputs, thus reducing the diversity of the generated data.

To address these challenges, we propose a novel GAN model, referred to as C3GAN. The C3GAN model introduces cyclic consistency as a mechanism to stabilize the training process and enhance the diversity of generated samples. Cyclic consistency ensures that the transformation between the input and output of the generator remains coherent, facilitating more stable learning and significantly reducing the risk of mode collapse.

In addition to the novel model, we have developed an image style transfer system based on C3GAN. Our system is implemented using the Python programming language, the Flask web development framework, and MySQL database technology.

Generally, our contribution includes the following:

1. **Introduction of C3GAN:** We propose the C3GAN model, which incorporates cyclic consistency to address the challenges of unstable training dynamics and limited diversity in generated samples. This novel approach ensures more reliable learning and reduces the occurrence of mode collapse.
2. **Development of an Image Style Transfer System:** We have implemented an efficient and robust image style transfer system based on the C3GAN model. This system is designed to perform superior style transfer tasks, enhancing both the stability and diversity of the results.
3. **Technological Integration:** Our system leverages modern web development (Flask) and database technologies (MySQL) to create a comprehensive and scalable image style transfer platform, broadening its applicability in various creative domains.
4. **Application in Creative Industries:** By utilizing the C3GAN-based system, we expand the potential applications of GANs in fields such as artistic creation and special effects in film and television, offering a new tool for creative professionals.

2. Related Work

2.1. Background of GAN Research

Generative Adversarial Networks (GANs) are a type of deep learning model based on game theory, where data generation is achieved through the adversarial interplay between a generator G and a discriminator D . The generator G aims to create realistic data samples to deceive the discriminator D , while the discriminator D tries to distinguish between generated samples and real ones.

The training process involves adversarial optimization, where both models continuously adjust their strategies to optimize their respective objective functions. The generator G attempts to maximize the probability of the discriminator D making a mistake, while the discriminator D tries to minimize this probability. The objective functions for these models are defined as follows:

For the discriminator D :

$$\mathcal{L}_D = -\mathbb{E}_{x \sim p_{data}(x)}[\log D(x)] - \mathbb{E}_{z \sim p_z(z)}[\log(1 - D(G(z)))]$$

For the generator G :

$$\mathcal{L}_G = -\mathbb{E}_{z \sim p_z(z)}[\log(D(G(z)))]$$

In these equations, $p_{data}(x)$ represents the distribution of real data, $p_z(z)$ is the distribution of the latent variables, $D(x)$ is the discriminator's probability that x is real, and $G(z)$ is the generator's output when given a latent vector z . The generator G aims to minimize the loss function \mathcal{L}_G by producing data that are increasingly similar to real data, while the discriminator D aims to maximize \mathcal{L}_D by accurately distinguishing between real and generated data.

This adversarial training process continues until Nash Equilibrium is reached. At this equilibrium, the generator produces samples that are nearly indistinguishable from real data, indicating that the generator and discriminator are in balance. GANs have been widely applied in various domains, including image generation [10–12], video generation [13,14], and natural language processing [15,16]. For instance, Conditional GANs (cGANs) [17] have been used for posture-guided character image generation and facial expression animation [18]. GANs also demonstrate exceptional capabilities in image editing and reconstruction tasks, such as super-resolution [19,20]. However, GANs often encounter issues such as mode collapse and training instability, which limit their application in more complex tasks. These challenges have prompted researchers to explore various optimization techniques and improvements to enhance GAN performance and applicability [21,22]. The development and evolution route of GANs is shown in Figure 1.

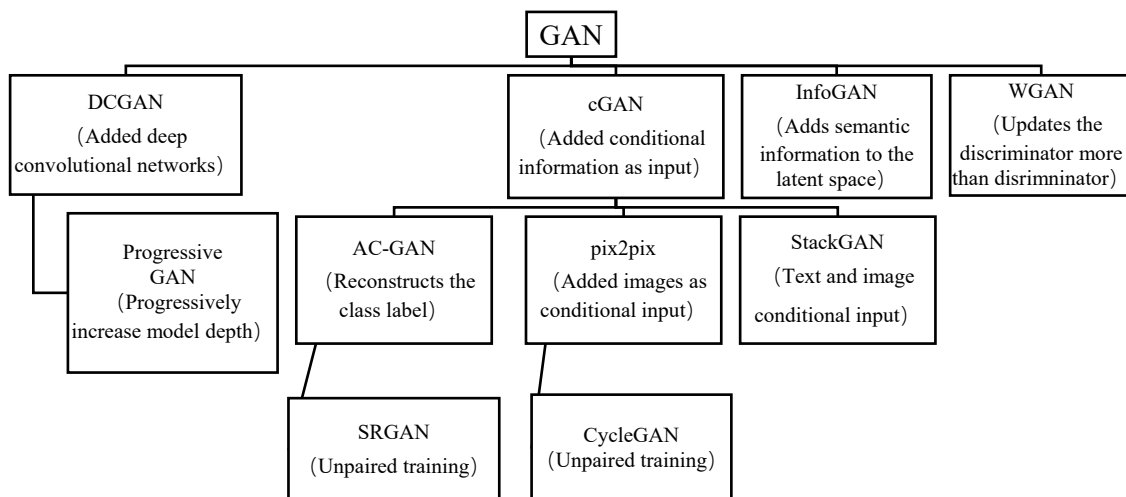


Figure 1. Evolution of GAN technology.

2.2. Unstable Training Dynamics

The instability in GAN training primarily arises from the adversarial game between the generator and the discriminator. During training, both models continuously adjust their strategies, making it difficult to maintain a stable training process. Specifically, issues such as gradient vanishing or exploding occur due to the adversarial relationship between the generator and discriminator. Gradient vanishing happens when the discriminator is too powerful, causing the generator's gradient signals to become very weak and impeding its learning. Conversely, if the generator is too powerful, it can lead to gradient explosion in the discriminator, resulting in training instability [5,6]. Mode collapse is another common issue, where the generator tends to produce a limited number of sample variations, lacking diversity [7–9]. To address these problems, several improvements have been proposed. Wasserstein GAN (WGAN) [23,24] replaces the traditional JS divergence with Wasserstein distance, improving training stability. Additionally, spectral normalization [25] normalizes the weights of the discriminator to control the network's Lipschitz constant, thus enhancing training stability. Gradient penalty, introduced in WGAN-GP [26], helps avoid gradient explosion and vanishing issues. Moreover, techniques such as progressive training and dynamic learning rate adjustments have been employed to improve training stability.

2.3. Generation of Sample Diversity

The issue of insufficient sample diversity is another significant challenge faced by GANs. Mode collapse results in the generator producing samples with limited variability, lacking sufficient diversity [7]. This problem arises primarily because traditional GAN objective functions focus too much on generating realistic samples without encouraging diversity [8,9]. Additionally, the adversarial nature of the game between the generator and discriminator can lead to mode collapse, particularly when the discriminator is excessively powerful. To improve sample diversity, researchers have proposed several effective strategies. One approach is to introduce diversity rewards, which maximize the mutual information between samples to encourage the generator to produce more diverse outputs. Conditional GANs [17] guide the generator by incorporating conditional information during training, resulting in samples with higher diversity and relevance. Models combining Variational Autoencoders (VAEs) with GANs (e.g., VAE-GANs) [27] utilize the latent space modeling capabilities of VAEs along with the generative capabilities of GANs to enhance both sample diversity and quality. Furthermore, gradient penalty and other regularization techniques are used during training to constrain the generator's outputs, reducing mode collapse [28,29]. These improvements offer effective solutions for enhancing GAN performance in generating diverse samples.

3. Image Style Transfer Technology Based on GAN

Image style transfer is an important application of GANs in image processing. Image style migration can apply the style of one image to another image to generate a new image, such that the new image retains the content characteristics of the original image while having the style characteristics of the other image [30].

3.1. Image Style Transfer Based on CycleGAN

Despite their effectiveness, GAN-based image style transfer techniques have several limitations, including the need for extensive data, significant computational resources, and lengthy training times. During training, the generator and discriminator can become unstable, complicating convergence to an optimal solution. Additionally, the generator may overfit to fine details and noise in the target style image, resulting in generated images that lack clarity and accuracy.

Traditional image-to-image translation methods often require separate models for different tasks. For example, converting a black-and-white photo to color demands one model, while transforming a horse image into a zebra requires another. The CycleGAN architecture addresses these issues by introducing a cyclic consistency loss function, which enables image translation between domains without paired training data [31]. This innovation significantly reduces the cost and time required for training. Further advancements have enhanced the CycleGAN model by improving the cyclic consistency loss function to learn many-to-many mappings without paired data, thereby boosting GAN performance in unsupervised settings [32]. Moreover, some studies have refined CycleGAN's image translation capabilities in complex scenes by incorporating a patch-based discriminator [33].

Other research has explored the integration of network architectures like U-Net and ResNet into GANs to achieve more effective style transfer. For instance, DU-GAN [34] incorporates a U-Net-based discriminator within the GAN framework. This U-Net-based discriminator offers dual benefits: it provides pixel-level feedback to the denoising network through its output while also considering global structure at the semantic level via its intermediate layers. In another study [35], a signal-to-image conversion method was developed to transform time-domain fault signals into RGB image format, which serves as the input datatype for ResNet-50. This led to the proposal of a new TCNN (ResNet-50) structure, which has been tested on multiple datasets, including the bearing damage dataset from KAT datacenter, the motor bearing dataset from Case Western Reserve University (CWRU), and a self-priming centrifugal pump dataset.

Ma et al. [36] introduced an iterative architecture for Restorable Arbitrary Style Transfer (RAST) to address content leak issues in arbitrary style transfer. RAST achieves the transmission of both content and style information through multi-restorations, controlling the content–style balance in stylized images through the precision of image restoration. To ensure the effectiveness of this architecture, two novel loss functions—multi-restoration loss and style difference loss—were designed. Additionally, a new quantitative evaluation method was proposed to assess content preservation and style embedding performance.

3.2. Comparison and Improvement of an Image Style Transfer System Based on CycleGAN

CycleGAN represents an advancement over traditional GAN architectures, incorporating key improvements such as cyclic consistency, the ability to operate with non-paired data, multi-modal output, and the integration of identity loss. Compared to other GAN variants like CoGAN [37], BiGAN/ALI [38], and SimGAN [39], CycleGAN is particularly well-suited for image style transfer across different domains. The following outlines the differences and improvements of CycleGAN over these other models:

Firstly, compared to CoGAN, CycleGAN leverages cyclic consistency loss, which ensures that an image, once transformed to another domain and then back, retains its original content. CoGAN, lacking this loss function, may suffer from content loss or distortion. Moreover, CycleGAN employs identity loss, allowing the generator to output the original image without transformation when the input is already in the target domain, thus preserving the image’s style. CoGAN does not utilize identity loss, leading to potential over-transformation or blending of styles. Additionally, CycleGAN supports multi-modal output, enabling the generation of images in various styles based on the input’s characteristics and random noise. In contrast, CoGAN only allows for single-modal output, producing images in just one style.

Secondly, in comparison to BiGAN/ALI, CycleGAN avoids using an encoder to map input images to a latent space, instead directly transforming the input image to the target domain using a generator. This approach reduces model complexity and computation while avoiding errors and information loss introduced by the encoder. Unlike BiGAN/ALI, which relies on supervised data pairs for training, CycleGAN can utilize unsupervised datasets, allowing for broader application and more diverse data sources, while also mitigating issues related to incomplete or inaccurate data. Furthermore, CycleGAN supports multi-domain transformations, enabling the generator to process multiple source and target domains simultaneously, whereas BiGAN/ALI is limited to two-domain conversions.

Lastly, when compared to SimGAN, CycleGAN employs adversarial loss and a discriminator to train the generator, improving the quality and realism of the generated images by ensuring they can fool the discriminator. SimGAN, on the other hand, uses perceptual loss and local perceptual discriminators to train the generator, which can result in images that are similar to those in the real image domain but may be lacking in quality and authenticity. CycleGAN’s use of cyclic consistency loss and identity loss helps maintain both the content and style of the original image. In contrast, SimGAN relies on self-supervised learning, which, while gradually enhancing the synthetic image’s defects, might alter the original content and style. Moreover, CycleGAN allows for conversion between any two different image domains without requiring paired data, whereas SimGAN is restricted to converting synthetic images to real images, necessitating synthetic data as the input.

Building on these principles, this study proposes an image style transfer system based on CycleGAN, which not only achieves unique artistic effects by applying the style of one image to another but also enhances adaptability across various datasets and broadens the system’s application scenarios.

4. Method

4.1. Overview

The proposed C3GAN model is built upon the foundational principles of Generative Adversarial Networks (GANs). In a traditional GAN, two networks, the generator G and

the discriminator D compete in a zero-sum game. The generator G aims to generate data samples that are indistinguishable from real data, while the discriminator D strives to distinguish between real and generated data. The overall objective function for a basic GAN can be expressed as follows:

$$\min_G \max_D \mathbb{E}_{x \sim p_{data}(x)} [\log D(x)] + \mathbb{E}_{z \sim p_z(z)} [\log(1 - D(G(z)))]$$

However, one limitation of traditional GANs is their difficulty in maintaining training stability and generating diverse outputs. To address these challenges, we propose the C3GAN model, which introduces cyclic consistency to enhance both stability and diversity.

The schematic diagram of the C3GAN model is shown in Figure 2. The model comprises four neural networks: two generator networks and two discriminator networks. Specifically, one generator network is responsible for transforming images from domain A to domain B, while the other performs the inverse transformation. Each discriminator is trained to distinguish between real images and those generated by the corresponding generator.

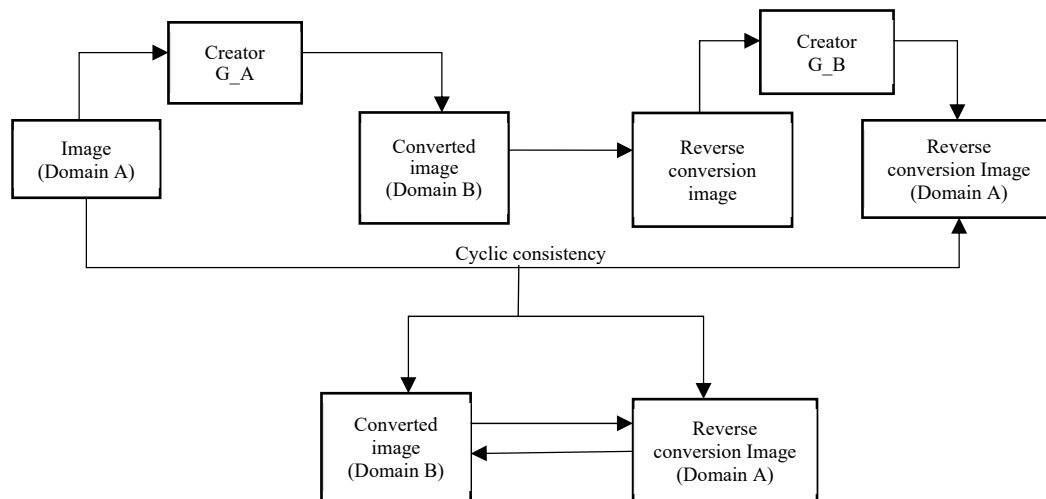


Figure 2. The schematic diagram of the C3GAN model.

Generator Networks: The C3GAN model employs two generator networks, denoted as G_A and G_B , for bidirectional transformation between domains A and B . Both generators utilize a convolutional neural network (CNN) architecture. For an input image x_A from domain A , G_A generates a corresponding image $x_B = G_A(x_A)$ in domain B . Conversely, G_B maps an image x_B from domain B back to domain A , producing $x'_A = G_B(x_B)$. The structure of these networks includes Batch Normalization to stabilize training and ReLU activation functions for non-linearity.

Discriminator Networks: The model also incorporates two discriminator networks, denoted as D_A and D_B , which evaluate the realism of images in domains A and B , respectively. D_A is trained to distinguish between real images x_A and generated images $G_B(x_B)$, while D_B discriminates between x_B and $G_A(x_A)$. The discriminator networks are also implemented using CNN architectures, with LeakyReLU activations to handle negative inputs more effectively.

Cyclic Consistency Loss: A critical feature of C3GAN is the introduction of cyclic consistency loss, which ensures that the transformations are reversible and consistent. This loss is defined as the L_1 distance between the original images and the images reconstructed after a complete cycle of transformations.

The training process of the C3GAN model, illustrated in Figure 3, begins with two inputs: image sets from two distinct domains that are labeled as domain A and domain B . These image sets are composed of 256×256 pixel RGB images, ensuring that both domains share the same number of channels and resolution. The model outputs two sets of images—one set

transforming images from domain A to domain B , and the other performing the reverse—with the outputs retaining the original resolution and channel dimensions.

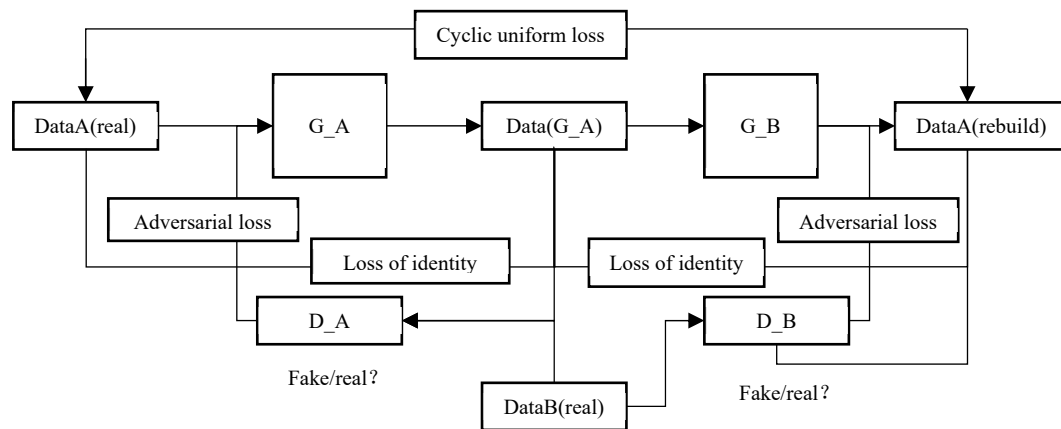


Figure 3. Training of C3GAN Model.

The C3GAN architecture is built upon four key components: two generators (G_A and G_B) and two discriminators (D_A and D_B). The generators are designed to perform the image-to-image translation tasks, where G_A converts images from domain A to domain B , and G_B performs the inverse operation. The discriminators D_A and D_B evaluate the authenticity of images in their respective domains, determining whether the images are real or generated. Each of these networks is constructed from multiple convolutional layers, interspersed with normalization layers to stabilize training and activation layers to introduce non-linearity.

The training process of C3GAN involves optimizing three key loss functions:

Adversarial Loss: This loss ensures that the generators produce outputs that are indistinguishable from real images in the target domain. It drives the generators to create realistic and convincing images, fooling the discriminators into classifying them as real.

Cycle Consistency Loss: The cycle consistency loss maintains the integrity of the content information during the transformation. It guarantees that an image translated from domain A to domain B and then back to domain A closely resembles its original form, preserving essential features and structures.

Identity Loss: This loss encourages the generators to retain the style information of the input images when the output domain matches the input domain. For example, when G_A processes an image from domain B , the output should be identical to the input image, ensuring that the generator does not unnecessarily alter the image's style when no domain translation is required.

4.2. Generator

The architecture of the generator network in the C3GAN model is illustrated in Figure 4. The generator begins with an initial convolutional layer (Conv2d) with a kernel size of 7×7 , a stride of 1, 3 input channels (corresponding to the RGB channels of the image), and 64 output channels. After passing through this layer, the output feature map size becomes $250 \times 250 \times 64$, calculated as $(256 - 7 + 1) \times (256 - 7 + 1) \times 64$.

Following this, the generator processes the feature map through a series of residual blocks, which maintain the spatial dimensions (W , H) while increasing the number of channels. The model typically uses 6 ResNet blocks, each doubling the number of channels. As a result, after passing through all residual blocks, the number of channels increases from 64 to 1024, calculated as 64×2^4 .

Subsequently, the generator employs upsampling layers (ConvTranspose2d) to enlarge the spatial dimensions of the feature map while reducing the number of channels by half.

With two default upsampling layers, the spatial size expands from 250 to 1000, while the number of channels decreases from 1024 to 512, and finally to 256.

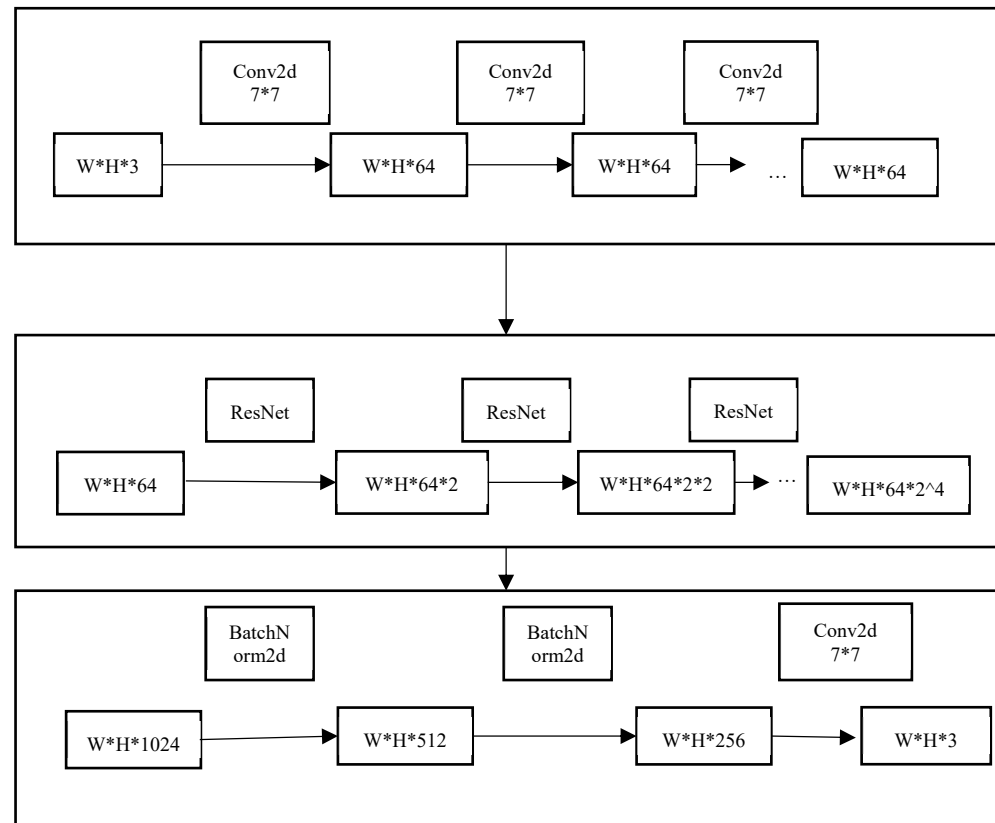


Figure 4. Generator of a C3GAN model.

The final Conv2d layer reduces the number of channels to match the output image's required channels, typically 3, corresponding to the RGB color channels of the output image.

4.3. Discriminator

The architecture of the discriminator network in the C3GAN model is depicted in Figure 5. The first convolutional layer (Conv2d) of the discriminator D has a kernel size of 4×4 and a stride of 2, with 3 input channels corresponding to the RGB channels of the image and 64 output channels. After passing through this layer, the resulting feature map has a size of $127 \times 127 \times 64$.

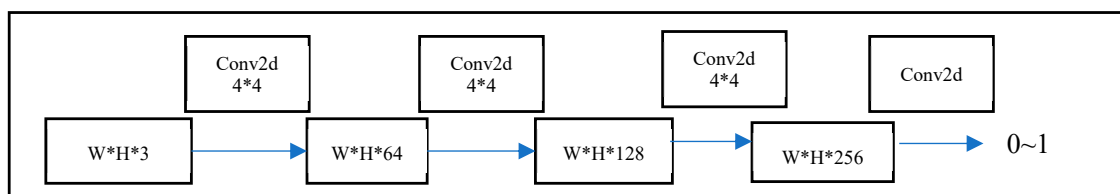


Figure 5. Discriminator of the C3GAN model.

As the feature map progresses through multiple convolutional layers, downsampling occurs, which gradually reduces the spatial dimensions while increasing the number of channels. The final convolutional layer increases the number of output channels to 256.

The last Conv2d layer reduces the number of channels to 1. This is followed by a sigmoid activation function, which outputs a value between 0 and 1, representing the probability that the input image is classified as a real image.

4.4. Design of an Image Style Transfer System Based on C3GAN

In common cases, software systems are required to be efficient and stable. We followed these principles when designing our image style transfer system based on our C3GAN model, such that our system can generate high-quality style transfer images under a short time limit. Additionally, we established an algorithm database for quick implementation of different algorithms. Meanwhile, our system also focused on security problems and user privacy. The architecture of our system is illustrated in Figure 6.

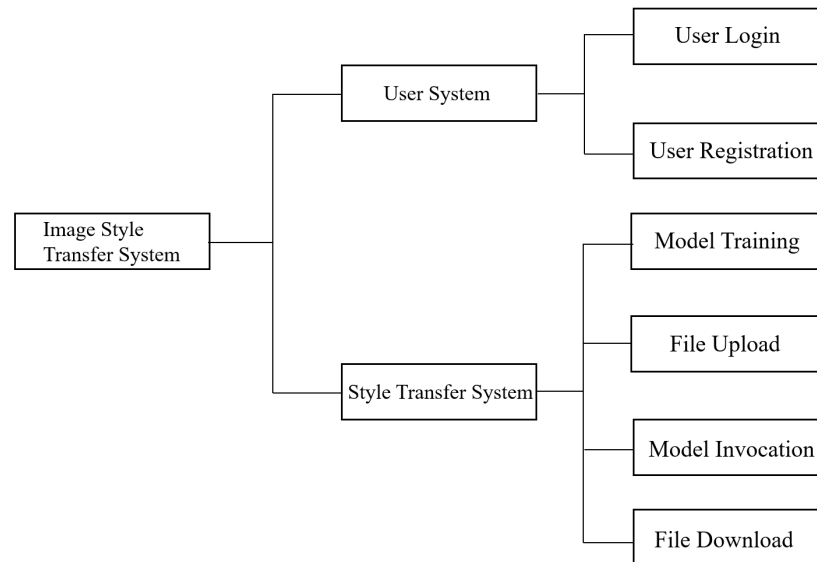


Figure 6. System architecture of our image style transfer system.

Our system is developed using Python and MySQL. Python serves as the core programming language, Flask is utilized as the web framework for frontend and backend interactions, and MySQL is employed as a lightweight database to store user and image information. Python is a high-level, easy-to-learn, open-source, and cross-platform language suitable for data science, artificial intelligence, web development, and other fields. Flask, a Python-based web framework, provides straightforward routing, template rendering, and request processing functionalities, making it ideal for small to medium-sized web application development. MySQL, a lightweight relational database management system that supports various platforms and is widely used in web application development. In this design, Python, Flask, and MySQL collaboratively build the development environment for the image style transfer system based on our C3GAN.

To be more specific, our system consists mainly of the User Login Module and the Main Interface Module.

User login module. The function of the user login module of this system is to verify the identity of different users and then display the main interface. The module supports two user types: ordinary users and user administrators. The module automatically jumps to different main interfaces according to the user type. Firstly, we created a web application based on Python Flask, then we created a Flask application named “app” and defined two routes, namely the root route “/”, “/login” and “/uploader”, and the sub-route “/admin”. Among them, the root route “/” corresponds to the login page; the route “/login” is used to receive the submission of the login form, verify the user’s identity, and jump to different main interfaces according to the user type; the routes “/uploader” and “/admin” correspond to the main interfaces of ordinary users and administrators. The user information is stored in the database, and the MySQL database is accessed by using the SQLAlchemy module of Python Flask. After reading the user information in MySQL, it is compared with the content filled in the form. If the username matches, the password is judged. If both match, the login is successful, otherwise the login fails.

Main interface module. The main interface of the system shows the effect of image style transfer based on the C3GAN model, as shown in Figure 7. The image upload interface has a button for uploading files and an area for displaying images, allowing users to upload their own images and perform style transfer, and then download the images. The website's graphical user interface is simple and beautiful, using the Bootstrap framework and providing important information and functions so that users can better use the system.

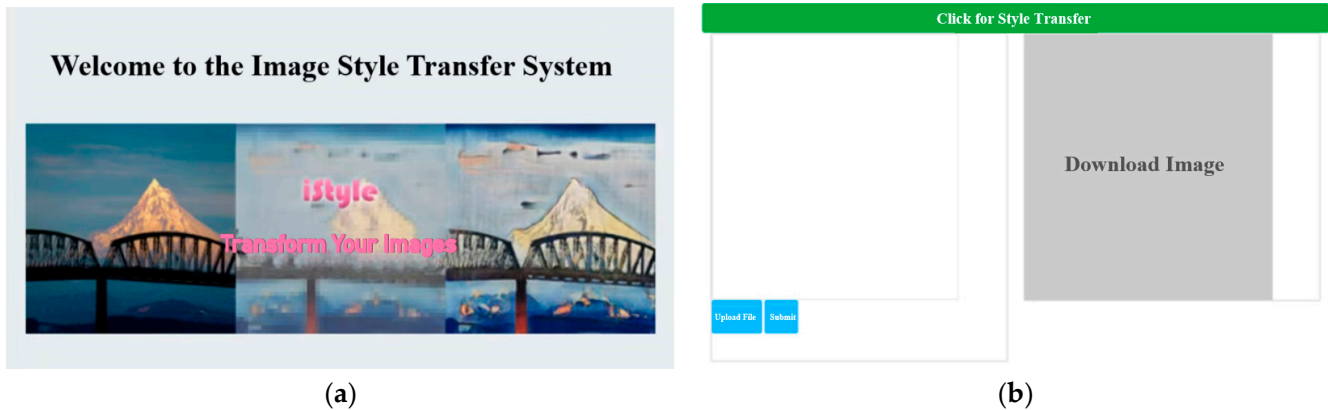


Figure 7. (a) Main user interface of our system and (b) specific operation interface.

The overall workflow of the system is illustrated in Figure 8. The frontend user interface provides a user-friendly page with options to upload files, allowing users to select and upload images for style conversion. The Flask backend processes user requests, invokes the C3GAN model for image style conversion, and returns the results to the front end. The pre-trained C3GAN model, loaded in the backend, performs the image style conversion, trained for tasks such as art style changes (e.g., Ukiyo-e) and seasonal changes.

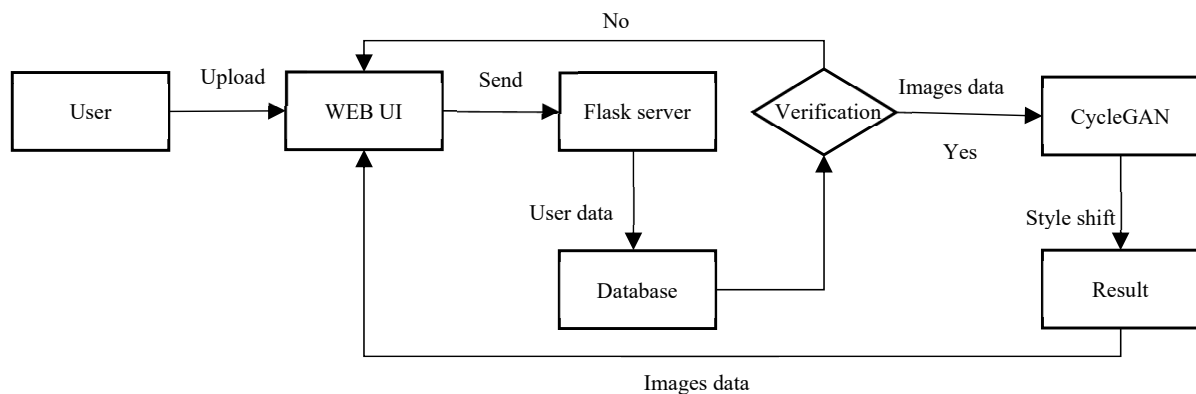


Figure 8. The overall architecture design of the system.

The main steps are as follows:

1. Upload Data: The system receives the image file uploaded by the user.
2. Send Data: The image data are transmitted to the Flask backend via the frontend user interface.
3. Data Processing: The validated image data enter the C3GAN model.
4. Image Style Conversion: The backend calls the loaded C3GAN model, inputs the user's image into the model, and generates the converted image.
5. Temporary Storage and Transfer: The converted image is saved to a temporary folder, associating the file name with the user for later identification and storage.
6. Return Result to Front End: The backend sends the file path or name of the converted image back to the front end for display to the user.

7. **User Interface Update:** The front end updates the displayed image based on the file path or name returned by the backend, presenting the style conversion result and providing a download button.

5. Experiment

In this section, we utilized the Cityscapes dataset, a high-quality and widely used dataset in the field of computer vision, primarily for tasks such as semantic segmentation, object detection, and image synthesis. The dataset contains urban street scene images from 50 cities, with fine pixel-level semantic annotations, making it particularly suitable for visual tasks in urban environments. The dataset comprises a total of 5000 finely annotated images, which we split into a training set and a test set in an 8:2 ratio. Specifically, 4000 images were used for model training, while the remaining 1000 images were reserved for testing, ensuring the model's generalization ability across diverse scenes.

The experiments were conducted on a Windows 11 operating system, using Python 3.8 and PyTorch 1.9.0 as the deep learning framework. To accelerate computation, CUDA 11.1 was employed, and the training was performed on an NVIDIA RTX 3090 GPU, paired with an Intel i7 13700k CPU and 32 GB of 3200 MHz memory.

The models were trained with a batch size of 16, which provided a balance between memory usage and training stability. The Adam optimizer was utilized with default parameters ($\beta_1 = 0.9$ and $\beta_2 = 0.999$) and a learning rate of 0.0002, which is commonly used in GAN training for effective convergence. The training process spanned 100 epochs, allowing ample time for the models to learn from the high-resolution dataset. Additionally, the weights of the networks were initialized using the Xavier initialization method to ensure stable training. This comprehensive setup ensured the efficiency and stability of both model training and evaluation.

5.1. System Testing

The black box testing method was employed to validate the system's image style migration capabilities using a diverse range of test cases. The testing process began by selecting a landscape image with dimensions of 256×256 pixels (as shown in Figure 9). The initial steps involved verifying that both the operating environment and the development environment were functioning correctly. Following this, the web interface was tested, beginning with the login functionality. The test confirmed that users could successfully log in using registered accounts stored in the database, while unregistered accounts were appropriately denied access. The registration feature was also tested, ensuring that newly registered accounts were correctly added to the database and could be used for subsequent logins.



Figure 9. Landscape picture before style migration.

The core of the testing focused on the style migration functionality. After inputting the landscape image, the system automatically converted it into the Ukiyo-e style (as shown in Figure 10) and returned the stylized image to the web interface. The download function was also tested, confirming that users could successfully download the stylized images.



Figure 10. Landscape image after style transfer.

The scope of the testing was then broadened to include various categories of images such as animals, architecture, flowers, and portraits. The system consistently performed successful style conversions across all these categories. The original and converted images are displayed in Figures 9–18, demonstrating the effectiveness of the system across different types of input images.



Figure 11. Animal images before style migration.



Figure 12. Animal images after style migration.



Figure 13. Picture of a building before style migration.

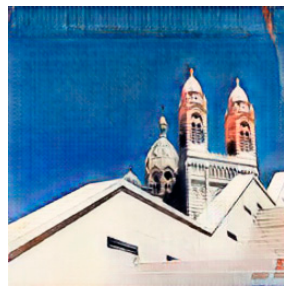


Figure 14. Picture of a building after style migration.

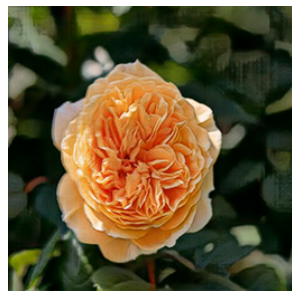


Figure 15. Flower picture before style migration.

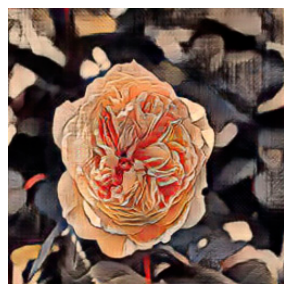


Figure 16. Flower picture after style migration.

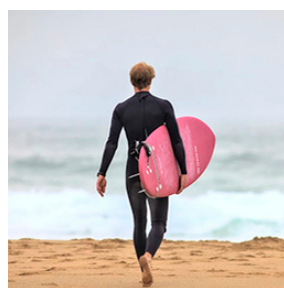


Figure 17. Character picture before style migration.

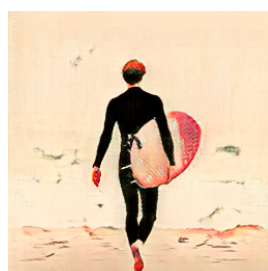


Figure 18. Character picture after style transfer.

5.2. Visualization Comparison against SOTA Methods

This experiment aimed to evaluate the performance of different image translation methods, particularly in converting aerial images into maps. To ensure the objectivity of the volunteers' evaluations, we designed a series of measures to minimize subjective bias.

First, the experiment selected multiple types of aerial images, converting them into map images and then converting the maps back into photos. All generated images were presented to the volunteers in random order to avoid any sequence effects that might influence the evaluation results. Additionally, each volunteer could only see one original image and its generated counterpart at a time, preventing them from making comparative judgments with other original or generated images. This design helps mitigate reference bias in their evaluations.

The experiment was conducted on the AMT crowdsourcing platform, recruiting a large number of volunteers. Each volunteer was asked to evaluate multiple pairs of images and identify the one they believed to be the most realistic. The volunteers' ratings were calculated using a weighted average approach, further reducing the impact of individual assessment biases. The results of these ratings are shown in Table 1.

Table 1. Summary of the scoring results of the same group of volunteers for the same picture.

	Map–Photo	Photo–Map
Method	Believed it to be the true	Believed it to be the true
CoGAN	16.65%	15.12%
SimGAN	16.68%	18.56%
BiGAN/ALI	18.28%	17.95%
L1+CNN	20.88%	20.24%
C3GAN	24.63%	22.90%

In the map-to-photo task, our C3GAN model was rated as producing the most realistic images, with 24.63% of volunteers believing the images were real. This was notably higher than other methods, such as CoGAN (16.65%) and L1+CNN (20.88%). Similarly, in the photo-to-map task, C3GAN again outperformed the others with a score of 22.90%, compared to SimGAN's 18.56% and L1+CNN's 20.24%. These results demonstrate C3GAN's superiority in generating realistic images, with a clear advantage of 4–8 percentage points over the other methods.

5.3. Metric Comparison against SOTA Methods

In this section, we delve into the performance of various image style transfer systems, focusing on key metrics such as Per-pixel Accuracy (PixelAcc), Per-class Accuracy (ClsAcc), and Mean Intersection over Union (mIoU). These metrics offer a comprehensive view of how well each model performs in translating image styles while preserving the semantic integrity of the content.

Pixel Accuracy (PixelAccuracy). Pixel Accuracy measures the proportion of correctly predicted pixels for each class over the total number of pixels. It is calculated using the following formula:

$$PixelAcc = \frac{\sum_i n_{ii}}{\sum_i t_i}$$

Class Accuracy (ClsAcc). Class Accuracy provides the average accuracy across all classes. It is calculated as follows:

$$ClsAcc = \frac{PixelAcc}{n_{cls}}$$

Mean Intersection over Union (mIoU). Mean IoU is a widely used metric in image segmentation that measures the overlap between the predicted segmentation and the ground truth. It is calculated as follows:

$$mIoU = \frac{1}{n_{cls}} \sum_i \frac{n_{ii}}{t_i + \sum_j (n_{ji} - n_{ii})}$$

where n_{ii} is the number of pixels predicted as class i that are actually class i , t_i is the total number of pixels in class i , n_{cls} is the number of classes, and n_{ji} is the number of pixels predicted as class i but are actually class j .

Our C3GAN model, enhanced with cyclic consistency loss, demonstrates clear superiority across all evaluated metrics, highlighting its effectiveness in producing high-quality image style transfers. As shown in Table 2, C3GAN achieved a remarkable PixelAcc of 0.76, outperforming all other methods, including the well-established Pix2Pix baseline, which scored 0.62. This substantial improvement in PixelAcc indicates that C3GAN excels in accurately predicting the pixel-level content of the transferred images.

Table 2. Comparison of various image style transfer systems.

Method	Per-Pixel ACC	Per-Class ACC	Class IOU
CoGAN	0.38	0.12	0.06
BiGAN/ALI	0.21	0.07	0.03
SimGAN	0.24	0.11	0.05
GAN	0.22	0.05	0.01
cGAN	0.57	0.22	0.16
L1+CNN	0.42	0.15	0.11
L1+GAN	0.58	0.19	0.14
L1+cGAN	0.60	0.22	0.16
Pix2Pix(baseline)	0.62	0.21	0.16
CycleGAN(ours)	0.76	0.28	0.19

When examining ClsAcc, which reflects the average accuracy across all classes, C3GAN again takes the lead with a score of 0.28. This is a notable enhancement over other methods like L1+cGAN and Pix2Pix, which scored 0.22 and 0.21, respectively. The improved ClsAcc underscores C3GAN's ability to maintain high accuracy across different classes within the images, suggesting that the model is particularly effective at preserving class-specific details during the style transfer process.

The mIoU metric, which measures the overlap between predicted segmentations and ground truth, further demonstrates C3GAN's strength. With an mIoU of 0.19, C3GAN surpasses other approaches such as cGAN and L1+cGAN, both of which scored 0.16. This improvement indicates that C3GAN not only preserves the overall structure of the images but also ensures a better match between the generated content and the actual semantic layout, leading to more coherent and contextually accurate style transfers.

The consistent performance across these metrics can be attributed to C3GAN's use of cyclic consistency, which effectively mitigates common issues in GAN-based models, such as mode collapse and training instability. By ensuring that the generated image can be accurately translated back to the original semantic tags, C3GAN achieves higher fidelity in the style transfer process.

5.4. Time-Efficiency Comparison against SOTA Methods

In evaluating the performance of different GAN-based models for image style transfer, inference time is a critical metric, especially for applications requiring real-time processing. Table 3 presents the inference times (in milliseconds) for various models at an input-output resolution of 512×512 .

Table 3. Time-efficiency comparison of various image style transfer methods.

Method	Time(ms)
CoGAN	8260
BiGAN/ALI	7032
SimGAN	7900
cGAN	6260
L1+CNN	6840
L1+cGAN	6480
C3GAN(ours)	6240

Among the models compared, our proposed C3GAN demonstrates the fastest inference time of 6240 ms, marginally outperforming other models, including cGAN and L1 + cGAN, which exhibit times of 6260 ms and 6480 ms, respectively. Traditional models like CoGAN and SimGAN show significantly longer inference times, at 8260 ms and 7900 ms, respectively, reflecting their higher computational demand.

6. Conclusions

This study introduces the C3GAN model, which was designed to address the issues of unstable training and limited pattern generation in traditional Generative Adversarial Networks (GANs). The core principle of C3GAN is the incorporation of cyclic consistency loss, which ensures the consistency of content during the image style transfer process while maintaining the style information of the input images. By leveraging cyclic consistency, C3GAN effectively stabilizes the training process and enhances the diversity and quality of the generated images.

Experimental results demonstrate that C3GAN significantly outperforms existing image style transfer models, such as CoGAN, BiGAN/ALI, and SimGAN, in terms of Per-pixel Accuracy (ACC), Per-class Accuracy (ACC), and Mean Intersection over Union (mIOU). These improvements highlight the effectiveness of cyclic consistency loss in enhancing the performance of GAN-based image style transfer systems.

Additionally, we designed a comprehensive image style transfer system that integrates the C3GAN model. This system, developed using Python, Flask 2.0.1, and MySQL 5.7, provides a user-friendly interface for uploading and processing images, leveraging the advanced capabilities of C3GAN to deliver superior style transfer results. This system not only demonstrates the practical applications of our model but also sets a new standard for future developments in the field of image style transfer.

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

Funding: This research was funded by the 2024 Macao Foundation Project (MF2342); the 2022 Research Topic of Online Open Course Guidance Committee of Undergraduate Universities in Guangdong Province (2022ZXKC041; 2022ZXKC561); and the “Four New” Experimental Teaching Curriculum Reform Project of Jinan University (SYJG202317).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data used to support the findings of this study can be obtained from the corresponding author upon request.

Conflicts of Interest: The authors declare that there are no conflicts of interest regarding the publication of this paper.

References

- Chen, H.; Wang, Z.; Zhang, H.; Zuo, Z.; Li, A.; Xing, W.; Lu, D. Artistic style transfer with internal-external learning and contrastive learning. In Proceedings of the 35th Conference on Neural Information Processing Systems (NeurIPS 2021), Online, 6–14 December 2021; Volume 34, pp. 26561–26573.
- Savardi, M.; Kovács, A.B.; Signoroni, A.; Benini, S. CineScale: A dataset of cinematic shot scale in movies. *Data Brief* **2021**, *36*, 107002. [[CrossRef](#)] [[PubMed](#)]
- Goodfellow, I.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; Bengio, Y. Generative adversarial nets. In *Advances in Neural Information Processing Systems*; MIT Press: Cambridge, MA, USA, 2014; pp. 2672–2680.
- Goodfellow, I. NIPS 2016 tutorial: Generative adversarial networks. *arXiv* **2016**, arXiv:1701.00160.
- Fang, Z.; Shahbazi, M.; Probst, T.; Paudel, D.P.; Van Gool, L. Training dynamics aware neural network optimization with stabilization. In Proceedings of the Asian Conference on Computer Vision, Macau, China, 4–8 December 2022; pp. 4276–4292.
- Kunapinun, A.; Dailey, M.N.; Songsaeng, D.; Parnichkun, M.; Keatmanee, C.; Ekpanyapong, M. Improving GAN learning dynamics for thyroid nodule segmentation. *Ultrasound Med. Biol.* **2023**, *49*, 416–430. [[CrossRef](#)] [[PubMed](#)]
- Cheng, Y.C.; Lin, C.H.; Lee, H.Y.; Ren, J.; Tulyakov, S.; Yang, M.H. Inout: Diverse image outpainting via gan inversion. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, New Orleans, LA, USA, 18–24 June 2022; pp. 11431–11440.
- Kossale, Y.; Airaj, M.; Darouichi, A. Mode collapse in generative adversarial networks: An overview. In Proceedings of the 2022 8th International Conference on Optimization and Applications (ICOA), Sestri Levante, Italy, 6–7 October 2022; pp. 1–6.
- Ding, Z.; Jiang, S.; Zhao, J. Take a close look at mode collapse and vanishing gradient in GAN. In Proceedings of the 2022 IEEE 2nd International Conference on Electronic Technology, Communication and Information (ICETCI), Changchun, China, 27–29 May 2022; pp. 597–602.
- Radford, A.; Metz, L.; Chintala, S. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv* **2015**, arXiv:1511.06434.
- Reed, S.; Akata, Z.; Yan, X.; Logeswaran, L.; Schiele, B.; Lee, H. Generative adversarial text to image synthesis. In Proceedings of the International Conference on Machine Learning, New York, NY, USA, 20–22 June 2016; pp. 1060–1069.
- Gadelha, M.; Maji, S.; Wang, R. 3D shape induction from 2D Views of multiple objects. *arXiv* **2016**, arXiv:1612.05872.
- Mathieu, M.; Couprie, C.; Lecun, Y. Deep multi-scale video prediction beyond mean square error. *arXiv* **2015**, arXiv:1511.05440.
- Vondrick, C.; Pirsiaavash, H.; Torralba, A. Generating videos with scene dynamics. In Proceedings of the Conference on Neural Information Processing Systems 2016, Barcelona, Spain, 5–10 December 2016; pp. 613–621.
- Li, J.; Monroe, W.; Shi, T.; Ritter, A.; Jurafsky, D. Adversarial learning for neural dialogue generation. *arXiv* **2017**, arXiv:1701.06547.
- Yu, L.; Zhang, W.; Wang, J.; Yu, Y. SeqGAN: Sequence generative adversarial nets with policy gradient. *arXiv* **2016**, arXiv:1609.05473.
- Ma, L.; Jia, X.; Sun, Q.; Schiele, B.; Tuytelaars, T.; Van Gool, L. Pose Guided Person Image Generation. In Proceedings of the 31st Annual Conference on Neural Information Processing Systems (NIPS), Long Beach, CA, USA, 4–9 December 2017; pp. 406–416.
- Tang, H.; Xu, D.; Sebe, N.; Wang, Y. GANimation: Anatomically-aware Facial Animation from a Single Image. *IEEE Trans. Pattern Anal. Mach. Intell.* **2020**, *42*, 528–541.
- Qin, J.; Huang, Y.; Wen, W. Multi-scale feature fusion residual network for single image super-resolution. *Neurocomputing* **2020**, *379*, 334–342. [[CrossRef](#)]
- Zhang, Y.; Wang, Y.; Fritts, J.E.; Zhuang, H. Conditional Generative Adversarial Network for Single Image Super-Resolution. *IEEE Trans. Image Process.* **2021**, *30*, 4937–4949.
- Tschannen, M.; Djolonga, J.; Ritter, M.; Mahendran, A.; Houlsby, N.; Gelly, S. On Mutual Information Maximization for Representation Learning. In Proceedings of the 8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, 30 April 2020.
- Jeong, J.J.; Tariq, A.; Adejumo, T.; Trivedi, H.; Gichoya, J.W.; Banerjee, I. Systematic review of generative adversarial networks (GANs) for medical image classification and segmentation. *J. Digit. Imaging* **2022**, *35*, 137–152. [[CrossRef](#)] [[PubMed](#)]
- Adler, J.; Lunz, S. Banach wasserstein gan. In Proceedings of the 32nd Conference on Neural Information Processing Systems (NeurIPS 2018), Montreal, QC, Canada, 3–8 December 2018.
- Cao, J.; Mo, L.; Zhang, Y.; Jia, K.; Shen, C.; Tan, M. Multi-marginal wasserstein gan. In Proceedings of the 33rd Conference on Neural Information Processing Systems (NeurIPS 2019), Vancouver, BC, Canada, 8–14 December 2019.
- Miyato, T.; Kataoka, T.; Koyama, M.; Yoshida, Y. Spectral normalization for generative adversarial networks. *arXiv* **2018**, arXiv:1802.05957.
- 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 30, Long Beach, CA, USA, 4–9 December 2017.
- Gur, S.; Benaim, S.; Wolf, L. Hierarchical patch vae-gan: Generating diverse videos from a single sample. In Proceedings of the 34th Conference on Neural Information Processing Systems (NeurIPS 2020), Vancouver, BC, Canada, 6–12 December 2020; Volume 33; pp. 16761–16772.
- Pei, S.; Da Xu, R.Y.; Xiang, S.; Meng, G. Alleviating mode collapse in GAN via diversity penalty module. *arXiv* **2021**, arXiv:2108.02353.

29. Tran, N.T.; Bui, T.A.; Cheung, N.M. Improving GAN with neighbors embedding and gradient matching. In Proceedings of the AAAI Conference on Artificial Intelligence, Honolulu, HI, USA, 27 January–1 February 2019; Volume 33, pp. 5191–5198.
30. Alotaibi, A. Deep generative adversarial networks for image-to-image translation: A review. *Symmetry* **2020**, *12*, 1705. [[CrossRef](#)]
31. Zhu, J.Y.; Park, T.; Isola, P.; Efros, A.A. Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), Venice, Italy, 22–29 October 2017; pp. 2242–2251.
32. Almahairi, A.; Rajeswar, S.; Sordoni, A.; Bachman, P.; Courville, A. Augmented CycleGAN: Learning Many-to-Many Mappings from Unpaired Data. In Proceedings of the 35th International Conference on Machine Learning (ICML 2018), Stockholm, Sweden, 10–15 July 2018; pp. 300–309.
33. Habijan, M.; Gali, I. Generation of Artificial CT Images using Patch-based Conditional Generative Adversarial Networks. *arXiv* **2022**. [[CrossRef](#)]
34. Huang, Z.; Zhang, J.; Zhang, Y.; Shan, H. DU-GAN: Generative adversarial networks with dual-domain U-Net-based discriminators for low-dose CT denoising. *IEEE Trans. Instrum. Meas.* **2021**, *71*, 4500512. [[CrossRef](#)]
35. Wen, L.; Li, X.; Gao, L. A transfer convolutional neural network for fault diagnosis based on ResNet-50. *Neural Comput. Appl.* **2020**, *32*, 6111–6124. [[CrossRef](#)]
36. Ma, Y.; Zhao, C.; Li, X.; Basu, A. RAST: Restorable arbitrary style transfer via multi-restoration. In Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, Waikoloa, HI, USA, 2–7 January 2023; pp. 331–340.
37. Liu, M.Y.; Tuzel, O. Coupled generative adversarial networks. In Proceedings of the 30th Conference on Neural Information Processing Systems (NIPS 2016), Barcelona, Spain, 5–10 December 2016; Volume 29.
38. Ding, R.; Guo, G.; Yan, X.; Chen, B.; Liu, Z.; He, X. BiGAN: Collaborative filtering with bidirectional generative adversarial networks. In Proceedings of the 2020 SIAM International Conference on Data Mining, Cincinnati, OH, USA, 7–9 May 2020; pp. 82–90.
39. Shrivastava, A.; Pfister, T.; Tuzel, O.; Susskind, J.; Wang, W.; Webb, R. Learning from simulated and unsupervised images through adversarial training. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 2107–2116.

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.