applied sciences

MDPI

Article

Optimizing Generative Adversarial Network (GAN) Models for Non-Pneumatic Tire Design

Ju Yong Seong [1], Seung-min Ji [1], Dong-hyun Choi [2], Seungjae Lee [1,*] and Sungchul Lee [1]

[1] Division of Computer Science and Engineering, Sunmoon University, Asan 31460, Republic of Korea; humphery27@sunmoon.ac.kr (J.Y.S.); wltmdals0118@sunmoon.ac.kr (S.-m.J.); sungchul@sunmoon.ac.kr (S.L.)
[2] Department of Artificial Intelligence and Software Technology, Sunmoon University, Asan 31460, Republic of Korea; ehdgus275@sunmoon.ac.kr
* Correspondence: leeko@sunmoon.ac.kr

Abstract: Pneumatic tires are used in diverse industries. However, their design is difficult, as it relies on the knowledge of experienced designers. In this paper, we generate images of non-pneumatic tire designs with patterns based on shapes and lines for different generative adversarial network (GAN) models and test the performance of the models. Using OpenCV, 2000 training images were generated, corresponding to spoke, curve, triangle, and honeycomb non-pneumatic tires. The images created for training were used after removing highly similar images by applying mean squared error (MSE) and structural similarity index (SSIM). To identify the best model for generating patterns of regularly shaped non-pneumatic tires, GAN, deep convolutional generative adversarial network (DCGAN), StarGAN v2, StyleGAN v2-ADA, and ProjectedGAN were compared and analyzed. In the qualitative evaluation, the GAN, DCGAN, StarGAN v2, and StyleGAN v2-ADA models distorted the circle shape and did not maintain a consistent pattern, but ProjectedGAN retained consistency in the circle, and the pattern was less distorted than in the other GAN models. When evaluating quantitative metrics, ProjectedGAN performed the best among several techniques when the difference between the generated and actual image distributions was measured.

Keywords: ProjectedGAN; DCGAN; mean squared error; structural similarity index; StyleGAN; StarGAN

check for updates

Citation: Seong, J.Y.; Ji, S.-m.; Choi, D.-h.; Lee, S.; Lee, S. Optimizing Generative Adversarial Network (GAN) Models for Non-Pneumatic Tire Design. *Appl. Sci.* **2023**, *13*, 10664. https://doi.org/10.3390/app131910664

Academic Editors: Božidar Šarler, Haiping Liu, Jian Zhang and Jiaqi Li

Received: 21 August 2023
Revised: 18 September 2023
Accepted: 20 September 2023
Published: 25 September 2023

Copyright: © 2023 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

Pneumatic tires were invented in England in the late 1800s [1]. However, due to the limitations and problems of pneumatic tires, punctures can occur while driving and maintaining the proper amount of air pressure is essential and requires regular inspection and maintenance [2]. To solve these problems, non-pneumatic tires were introduced by creating polygonal elastomer spokes that replace and fill the air between the wheel and the tread [3].

Non-pneumatic tires are extremely durable because they cannot be punctured by air pressure. They are expected to be used in a variety of transportation vehicles, including military vehicles, forklifts, and lunar rovers, that travel over rough terrain [4]. However, non-pneumatic tires are widely used in construction and agriculture, but their design requires the expertise of skilled professionals with years of experience, and finding structured rules is difficult, making full or partial automation difficult [5]. Therefore, we propose to design non-pneumatic tires using GAN AI models.

Advances in AI technology have led to a wide range of applications in fields as diverse as image generation and conversion, speech synthesis, and natural language processing. These techniques are being utilized to solve many previously unsolvable problems by endowing computers with human-like cognitive abilities. Among them, GANs [6], generative models based on unsupervised learning, are making breakthroughs in generating images. For example, NVIDIA's StyleGAN is a technique for creating realistic human faces based

on GANs [7]. These techniques have applications in design [8], art [9], biotechnology [10], and the gaming industry [11]. Although GAN technology has exhibited high performance in image generation, it still has many limitations, including Nash equilibrium [12], gradient vanishing [13], mode collapse [14], mode dropping [15], a lack of diversity [16], and internal covariate shift [17]. To overcome these limitations, various GAN-based AI models, such as DCGAN [18], StarGAN v2 [19], StyleGAN v2-ADA [20], and ProjectedGAN [21], have been proposed.

To evaluate these GAN-based AI models, Inception Score (IS) [14], which measures diversity and quality; Fréchet Inception Distance (FID) [22], which measures the difference between generated and true image distributions; learned perceptual image patch similarity (LPIPS) [23], which measures the similarity of images by mimicking human visual perception; and precision and recall (PR) [24], which measures the precision and recall between generated and true images, have been utilized.

In this study, 2000 pattern images were used to train a GAN-based AI model. For the objective evaluation of GAN, DCGAN, StarGAN v2, StyleGAN v2-ADA, and ProjectedGAN, we used FID and LPIPS, which are commonly used evaluation indicators, and conducted quantitative performance evaluations of the models and qualitative design evaluations such as patterns and shape circles. From this, we conducted a comparative analysis of the objectivity inherent in evaluation indicators and the proficiency of the models in generating target images. This study is relevant to designers of non-pneumatic tires, who will be able to complete design work very efficiently and rapidly.

2. Test Methods

2.1. Acquiring Data

In this paper, we classified the shapes of non-pneumatic tires into four main categories using original images from the Korea Automotive Technology Institute (Katech) and various papers [25–28]. All images used for training were generated, not resized versions of original images or images found in papers. The images provided by Katech can be seen in Figure 1.

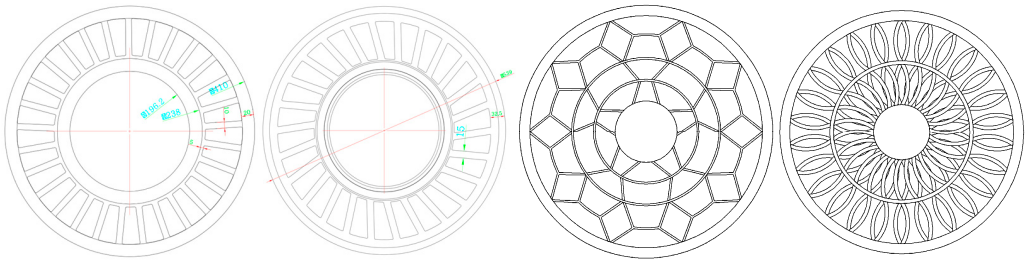
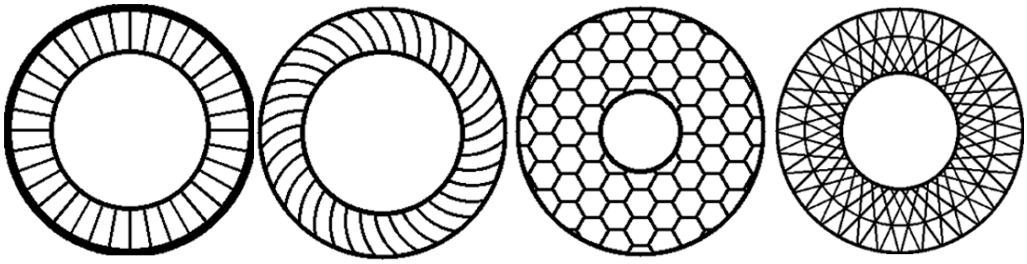


Figure 1. Original images from the Korea Automotive Technology Institute (Katech).

To compare the performance of the GAN models, we used OpenCV to generate 2000 images of the data to be trained. A total of 500 images were generated for each spoke shape, as shown in Figure 2.



(**a**) (**b**) (**c**) (**d**)

Figure 2. Four types of non-pneumatic tires created with OpenCV: (**a**) plate spoke; (**b**) curve spoke; (**c**) honeycomb spoke; (**d**) triangle spoke.

Figure 2 shows sample data from a spoke design generated using OpenCV. Four types of non-pneumatic tire spokes were selected and generated: plate spokes, curved spokes, honeycomb spokes, and triangular spokes. The plate spoke in Figure-(a) is a straight, columnar structure that uses vertical stiffness to support weight. The curved spoke in Figure-(b) is a curved cell structure with an ammonite pattern that has a high compressive strength-to-volume ratio. The honeycomb spoke in Figure-(c) has a hexagonal cell structure that is flexible in both the vertical and shear directions and is currently the most commonly researched. The triangular spoke in Figure-(d) has a triangular cell structure and similar characteristics to the honeycomb spoke. We generated 500 images of each spoke design, corresponding to a total of 2000 images for training.

The initial image of the tire that was generated was 1024 × 1024, but it required a significant amount of time and resources for training. Additionally, resizing it to 256 × 256 did not yield significantly different results. Therefore, we unified the extension of each data point to PNG for easy training and resized the image to 256 × 256 pixels. We then compared the similarities of the generated images by calculating the differences in luminance, contrast, structure, and pixel values using SSIM [29] and MSE [30] algorithms: SSIM compares the luminance, contrast, and structure of the original image and the comparison image, whereas MSE shows the measured value of the difference in each pixel value. Similar images were removed based on the comparison of the similarity of the images, and the remaining images were used as data for training.

2.2. GAN

GANs [6] have received continuous attention from deep learning researchers due to their emphasis on high-quality image generation, diversity of generated images, and stability of training. This is because GANs are models that can be applied to a variety of fields, including computer vision, natural language processing, time series synthesis, and semantic segmentation, but they have been particularly successful in generating synthetic image data.

As shown in Figure 3, a GAN is a game-theoretic AI model that uses two convolutional neural networks (CNNs) [31], each acting as a generator to produce an image and a discriminator to determine whether the image is real or fake and attempts to find a Nash equilibrium between the two CNNs. In a GAN, the generator and discriminator compete by generating data that resemble real data, and a sample that cannot be determined to be real or fake, considered a perfect equilibrium, is the optimal outcome. A GAN is divided into two parts: the part that identifies real images and the part that identifies false (fake) images generated by the constructor, as shown in Equation (1).

$$\min_{G}\max_{D} V(D,G) = \mathbb{E}_{x\sim p_{data}(x)}[\log D(x)] + \mathbb{E}_{z\sim p_z(z)}[\log(1 - D(G(z)))] \quad (1)$$

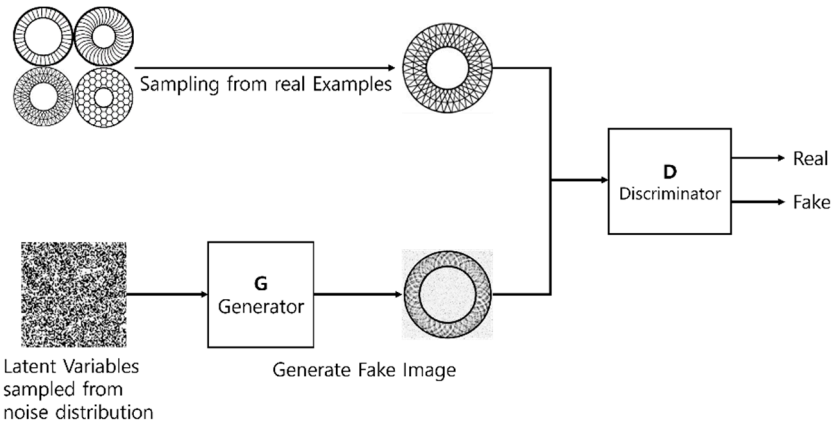


Figure 3. Flowchart of GAN operation.

$\mathbb{E}_{x\sim p_{data}(x)}[\log D(x)]$ is the part where the actual image x is put into D(x) to increase the log value so that it has a high probability value.$\mathbb{E}_{z\sim p_z(z)}[\log(1 - D(G(z)))]$ is the part where the image generated by G(x) is put into D(x) to make the log value smaller, resulting in a lower probability value. The constructor plays a min/max game, learning to produce images that are as close to the real image as possible until the discriminator fails to distinguish between the real and fake images.

For the original GAN, we used only linear layers for training because they were easier to model and faster to learn. We fixed the learning rate to 0.0002 for the discriminator and 0.0001 for the generator and applied the Adam optimizer [32] to a batch size of 128 and 256 × 256-pixel-size images for 2000 epochs. The result is shown in Figure 4. The tires and wheelbase appeared to be consistent with no distortion, but the image quality was very low, making it difficult to see the shape of the spoke design, and the FID values were not calculated due to mode collapse, which resulted in the continual production of similar images, reducing diversity.

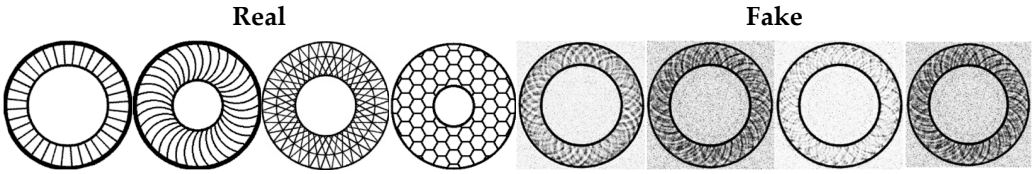


Figure 4. Comparison between real images and images generated by GAN.

2.3. DCGAN

DCGAN [18] is a model that combines the general GAN model with four learning techniques employed in CNN models: all convolutional net (All-CNN) [33], eliminating fully connected layers [34], batch normalization [35], and ReLU [36] + LeakyReLU [37].

1. All convolutional net: Introduced in 2014, All-CNN is a method that removes all deterministic spatial pooling functions [38], such as maxpolling [39], that are applied between the layers of a CNN. The layers are constructed using only strided convolutions without any pooling functions, allowing for self-downsampling.
2. Eliminating fully connected layers: This method does not utilize fully connected layers and consists of all convolutional layers. DCGAN directly connects the top convolutional features of the constructor and the discriminator to make the model stable and increase the speed at which it converges.
3. Batch normalization: Batch normalization produces a mean of 0 and a variance of 1 in each batch of data. This makes the model less sensitive to scale and, therefore, more stable during learning. This method helps to overcome the problem of poor learning due to poor initial weights or poor updating of the gradient descent method [35,40]. BatchNorm has been shown to play an important role in preventing the mode collapse of all samples to a single point, which is a common problem with GANs, and in promoting deeper generator layers. However, applying BatchNorm to all layers may result in the model repeating only a few patterns with limited samples or instability. Therefore, we did not apply BatchNorm to the output layer of the generator or the input layer of the discriminator.
4. ReLU + LeakyReLU: We used Tanh activation [41] for the output layer of the generator and ReLU activation for all other layers. For the discriminator, we used LeakyReLU activation to maintain a high resolution.

Based on these four fundamental methods, we improved the original GAN, changed the structure of the model to produce higher quality images, and used batch normalization and ReLU activation to improve the training stability and performance.

We used 2000 images of four types, created using OpenCV, for training. Because the existing DCGAN has an architecture optimized for 64 × 64 images, to perform training for the 256 × 256 image size used in this study, we added another convolutional layer to the

existing architecture, as shown in Figure 5. The architecture is based on the DCGAN code provided by Pytorch [42].

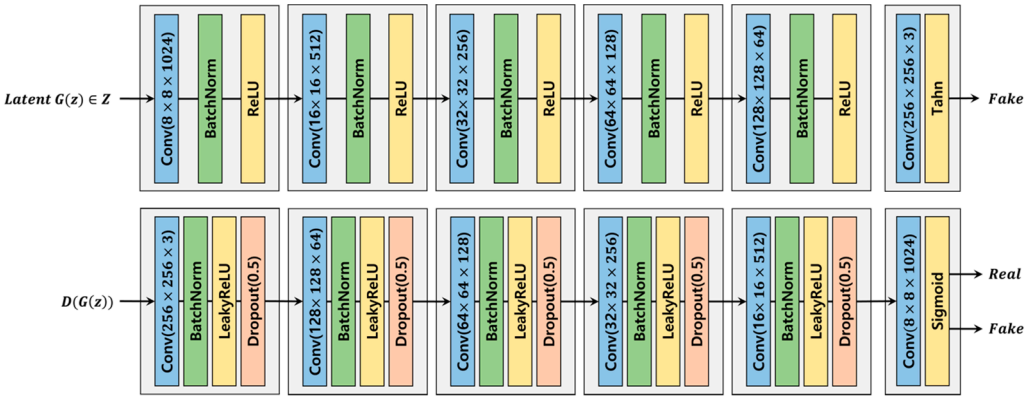


Figure 5. DCGAN architecture modified for 256 × 256 image size.

Figure 6 shows the training results of the DCGAN, which generated image sizes of 256 × 256. The DCGAN shows an increase in the quality and diversity of images compared to the original GAN. However, noise in the background and an inability to generate consistent patterns is also evident. Finally, we generated 2000 dummy images and compared them with the real images, resulting in an FID of 192.02 and an LPIPS of 0.4986.

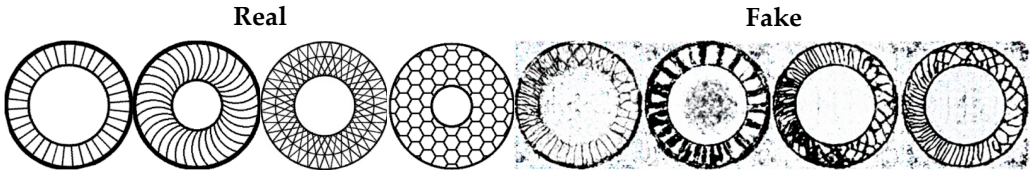


Figure 6. Comparison between real images and images generated by DCGAN.

2.4. StarGAN

An image-to-image translation model must preserve two properties: image diversity and scalability across multiple domains. However, existing models either synthesize images using only two domains or require a corresponding number of generators if there are many domains. For example, if there are K domains, existing methods require K(K − 1) generators and a training effort that depends on their number. StarGAN v2 proposes four new frameworks to meet these two requirements.

Generator: The generator (G) takes an image x as input and outputs G(x,s), reflecting a style code (s). The style code (s) is provided by a mapping network (F) or a style encoder (E). s is used in G by applying AdaIN [43] to it. This method eliminates the need to input numerous y directly into G and allows G to synthesize images from any domain. A description of these notation can be found in Table 1.

Table 1. Notation used in StarGAN v2.

Notation	Denote
G	Generator
D	Discriminator
s	Style coder
E	Style Encoder
F	Mapping Network
z	Latent code
y	domain
x	Input Image

Mapping network: The mapping network (F) takes a latent code (z) and a domain (y) as input and generates a style code $s = F_y(z)$. F_y is characterized for domain y. F consists of an MLP with multiple output branches and provides style codes for all domains. F takes a random latent vector z and samples from domain y to generate a set of style codes.

Style encoder: E takes an image x that matches y as input. E extracts the style code of x, $s = E_y(x)$. It is similar to the mapping network F, but because E takes a random number of images rather than a latent vector as input, a style code that references an image can be obtained.

Discriminator: The discriminator (D) in StarGAN v2 is a multi-task discriminator (D) consisting of several output branches. Each branch D_y learns a binary classification that distinguishes between a real image x in domain y and a fake image G(x,s) provided by the generator (G).

Images x and $\tilde{s}$ are input into G using Equation (2) and trained to output $G\left(x, \tilde{s}\right)$ with adversarial loss.

$$L_{adv} = \mathbb{E}_{x,y}\left[\log D_y(x)\right] + \mathbb{E}_{x,\tilde{y},z}\left[\log\left(1 - D_{\tilde{y}}\left(G\left(x, \tilde{s}\right)\right)\right)\right] \quad (2)$$

where $D_y(\cdot)$ denotes the output for domain y in D. In other words, if image x is in domain y, the output is fake for all other domains except the corresponding domain and real for that domain. F is trained to provide a style code $\tilde{s}$ that is likely to be domain $\tilde{y}$, and G is trained to use $\tilde{s}$ to produce a $G\left(x, \tilde{s}\right)$ that is indistinguishable from the real image of domain $\tilde{y}$.

Equation (3) is the style reconstruction loss. The loss reconstructs the original image x using the generated image $G\left(x, \tilde{s}\right)$ and the style code. It attempts to minimize the difference between the reconstructed image and the original image x. This allows G to learn from the input images x and s to produce an image of the desired domain.

$$L_{sty} = \mathbb{E}_{x,\tilde{y},z}\left[\left\|\tilde{s} - E_{\tilde{y}}\left(G\left(x, \tilde{s}\right)\right)\right\|_1\right] \quad (3)$$

Equation (4) is the diversity-sensitive loss that allows the generator to learn a variety of images.

$$L_{ds} = \mathbb{E}_{x,\tilde{y},z1,z2}\left[\left\|G\left(x, \tilde{s}_1\right) - G\left(x, \tilde{s}_2\right)\right\|_1\right] \quad (4)$$

Here, the provided latent codes z1 and z2 are used by F to generate style codes $\tilde{s}_1$ and $\tilde{s}_2$. The corresponding loss maximizes the regularization term, which encourages G to generate different styles and explore a wider image space.

Equation (5) is the cycle consistency loss, which ensures that the common characteristics of images belonging to a particular domain are preserved.

$$L_{cyc} = \mathbb{E}_{x,y,\tilde{y},z}\left[\left\|x - G\left(G\left(x, \tilde{s}\right).\hat{s}\right\|_1\right] \quad (5)$$

where $\hat{s} = E_y(x)$ is the style code of the input image x with y as the original domain of x. In other words, $\hat{s}$ generates a new style code that considers the domain of the input image rather than the style code extracted from the original domain. By using this $\hat{s}$ to reconstruct the input image x, G can perform a style-consistent transformation while preserving the original properties of x.

The four formulas can be summarized as follows:

$$\min_{G,F,ED} \max = L_{adv} + \lambda_{sty} L_{sty} - \lambda_{ds} L_{ds} + \lambda_{cyc} L_{cyc} \quad (6)$$

In Equation (6), $\lambda_{sty}, \lambda_{ds}, \lambda_{cyc}$ are the hyperparameters for each loss. When training the model, instead of using only latent vectors, we further trained it by using images to generate style codes. This approach yielded better style transfer.

In conclusion, StarGAN v2 is an image transformation model that can handle different domains in a single dataset and constructor through the four processes described above. The domain here refers to a set of data with the same image feature values. Compared to the existing models, StarGAN v2 is more effective in terms of computational efficiency and image quality.

Figure 7 shows the training results after changing only the settings of num_domains and the directory under the default settings of the training networks in StarGAN v2-ADA on Github [44]. Figure 7 shows the output of the synthesis of the first and second rows in the third row. Noticeably, the shape of the lines are represented for the spoke- and triangle-type tires, whereas for the curve- and honeycomb-type tires, the features of the domain were not learned, and the image disappeared. Because the qualitative evaluation yielded meaningless results, we did not proceed with the quantitative evaluation, FID.

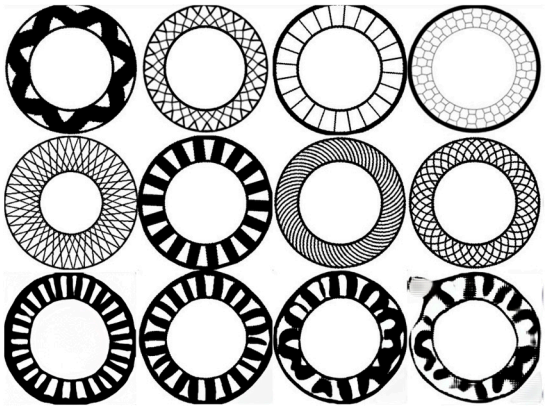


Figure 7. Image generation results of StarGAN v2.

2.5. StyleGAN v2-ADA

StyleGAN v2-ADA is an adaptive discriminator augmentation mechanism for solving the problem of overfitting with less data [45]. To train for modern high-resolution high-quality images, 10^5 to 10^6 images are required. If overfitting occurs with fewer training images, the feedback from the discriminator becomes meaningless, and the learning results begin to diverge. To solve this problem, data augmentation is used during deep learning to prevent overfitting. For example, rotating an image or applying noise to it to train an image classifier increases the invariance of the image to semantic-preserving distortions and increases the performance of the image classifier. These image variations can be used to make the images generated by full GAN more realistic and diverse. However, augmentation leakage can occur where the distribution of the augmented images follows the distribution of the original dataset, producing images with different characteristics. To avoid the above problems, StyleGAN v2-ADA proposes four solutions.

Stochastic discriminator augmentation: StyleGAN v2-ADA is a model with an architecture similar to that of bCR [46]. However, in the StyleGAN v2-ADA architecture, we removed the CR loss term from bCR and modified the model so that the discriminator uses only the images with augmentation applied. The generator also learns using only the images with augmentation applied. We call this approach stochastic discriminator augmentation.

Design augmentations that do not leak: If the data are augmented or corrupted to allow an invertible transformation, the discriminator can find the distribution of normal images from the corrupted images [47]. This can be achieved by making the images with normal distributions non-leaking with an appropriate fraction ($p < 1$) of the augmented images.

Adaptive discriminator augmentation: ADA is a method of dynamically automating the control based on the degree of overfitting to reduce the effort and time for learning by manually adjusting the hyperparameters. The basic method of measuring overfitting is using a separate validation set and observing the difference from the training set. This

method has the disadvantage of having to split the validation set even if small amounts of training data are available. However, using Equation (7), ADA can quantify overfitting without a validation set.

$$r_v = \frac{\mathbb{E}[D_{train}] - \mathbb{E}[D_{validation}]}{\mathbb{E}[D_{train}] - \mathbb{E}\left[D_{generated}\right]} \quad r_t = \mathbb{E}[\text{sign}(D_{train})] \tag{7}$$

Our enhancement pipeline: Given the diversity of images, we have organized our pipeline into 18 transformation types, which can be grouped into six categories: pixel blitting (x-flips, 90° rotations, and integer translations), general geometric transformations, color transformations, image-space filtering, additive noise, and cropping. We applied the transformations in a fixed order. The strength of the augmentation was controlled by a scalar p. We found that as long as p is kept below a safety threshold, the generator produces only clean images.

Figure 8 is the result of training for 2000 epochs using the Training New Networks command of StyleGAN v2-ADA with the hyperparameters set to default. When trained with StyleGAN v2-ADA, the image quality is significantly improved compared to that of the original GAN and DCGAN, and the pattern shape is relatively diverse. However, the circles were squashed, the tire shape was not properly visible, and the shape was not consistently generated. We used the FID calculation code built into StyleGAN v2-ADA and obtained an FID of 42.88 and an LPIPS of 0.3806.

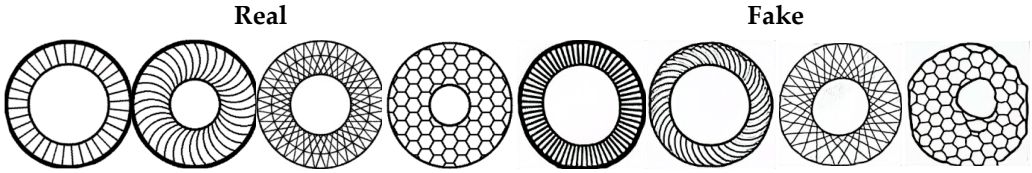


Figure 8. Comparison between real images and images generated by StyleGAN v2-ADA.

2.6. Projected GAN

Projected GAN adds a feature projector, P_l, to the original GAN, as shown in Equation (8). P_l maps the real image and the generated image into the input space of the discriminator.

$$\min_G \max_{\{D_l\}} \sum_{l \in L} (\mathbb{E}_x[\log D_l(P_l(x))] + \mathbb{E}_z[\log(1 - D_l(P_l(G(z))))]) \tag{8}$$

D_l works on any independent feature projection. P_l is fixed, and only the parameters of G and D_l are optimized. By fixing P_l, no competition among the producers occurs. Thus, the variation in the producers is small, resulting in stable learning. The projected GAN proposed three main methods to speed up the convergence of learning to make it comparable to that of existing GANs and solve the problem of the discriminator not fully utilizing the pre-trained features of deeper layers.

Multi-scale discriminators: Multi-scale discriminators extract features from four layers, L_l, of a pre-trained feature network F. Each layer, L_l, is applied to a separate discriminator, D_l, using a simple convolutional architecture that uses spectral normalization [48]. We used a simple convolutional architecture, where each discriminator D_l utilizes spectral normalization [48]. We found that the best performance was achieved by applying convolution until each discriminant had a scale of 4, 4 and then selecting logit. Finally, the logit values generated by each of the four discriminators were summed and used as the loss.

Random projections: We identified two strategies to make information about all features available to the discriminator. The first is cross-channel mixing (CCM) and the second is cross-scale mixing (CSM). CCM is a cross-channel mixing method that applies a 1×1 convolution with the same number of inputs, outputs, and channels so that the input is more informative. Increasing the number of output channels improves performance because the image information can be better preserved. CSM uses 3×3 convolution and

bilinear interpolation to mix the CCM output with the CCM output. These two methods allow us to preserve large amounts of information with fewer layers and the performance is better than that of traditional GANs.

Pre-trained feature network: In this scheme, we experimented with different feature networks. The first, EfficientNets [49], allows for direct control of model size and performance; that is, by changing or removing the structure of the model, the size of the model can be reduced or increased, and the performance of the model changes accordingly. EfficientNets is an image classification model trained with ImageNet [50] that models the tradeoff between accuracy and computation. Second, we used ResNets [51] of different sizes. We also used ResNets trained on the R50-CLIP dataset, which is a dataset of image–text pairs, instead of ImageNet, to analyze how models pre-trained with features from ImageNet affect GAN. Finally, we used the vision transformer architecture (ViTBase) [52] and its successor (DeiT small distilled) [53]. We excluded the inception network [54] because it is used to extract feature maps from FID metrics, and the generated images can be correlated with FIDs.

Unlike traditional GAN models, Projected GAN is a method that trains a GAN model using a pre-trained situation. This increases data efficiency and removes unnecessary constraints to improve image quality, training speed, and sample utility. Although GAN produces high-quality images, it requires numerous datasets, high computing power, and careful handling of hyperparameters to perform smooth training. However, Projected GAN solves these problems by fixing the image data generated in the pre-training environment and the actual image data for training. Projected GAN exhibits high performance with very low FID values compared to other GAN models.

Figure 9 shows the results of training Projected GAN. As the number of epochs increased, the image quality and circle distortion improved, and the spoke design became denser and more detailed. Compared to other GANs, the projected GAN model performed the best among the GANs used so far, ideally generating multiple images with a variety of features, such as differences in the size of the circles on the spokes, the design of the spokes, and the thickness of the lines. The final FID was 13.41, which was the highest among the five models.

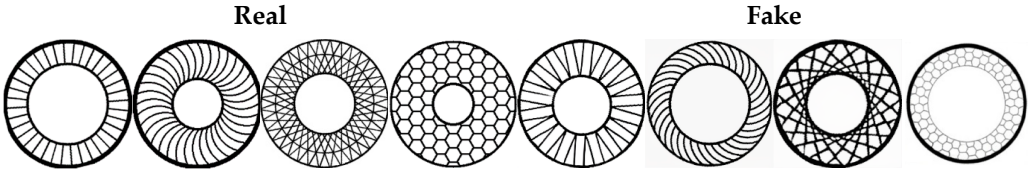


Figure 9. Comparison between real images and Projected GAN-generated images.

3. Validate Image Similarity

Metrics exist for evaluating the performance of the GAN model while verifying image similarity, such as IS, FID, and LPIPS. IS is a metric that evaluates the quality of the images generated by the GAN model, and this metric aims to evaluate how qualitatively good and diverse the generated images are. The better the IS, the higher the value is. In Equation (9), x in $x \sim p_g$ denotes the sample of p_g. IS consists of $\mathbb{E}_{x \sim p_g}$, which is the expected value; $p(y|X)$, the conditional probability distribution; and $p(y)$, the marginal class distribution. D_KL denotes the KL divergence (Kullback–Leibler divergence) [55] and measures the difference between two probability distributions, $p(y|x)$ and $p(y)$. $p(y|x)$ is the conditional probability of a class and denotes the probability distribution of the class on the input data. In $p(y|x)$, y refers to a class, and x refers to the input data. Therefore, it calculates the probability of an image being y given x. It calculates the probability of an image being in each class and places x in the class with the highest probability. $p(y) = \int p(y|x)p_g(x)$ is the marginal class distribution. Marginal probability distributions are used to remove the effects of variables from a multivariate probability distribution and calculate the distribution of some variables. Marginal probability distributions are used to

test for independence between variables and collapse variables, and determine the overall properties of a probability distribution.

$$\mathrm{IS}(G) = \exp\left(\mathbb{E}_{x\sim p_g} D_{KL}(p(y|X)||p(y)\right) \tag{9}$$

FID was developed to compensate for the shortcomings of IS and facilitate the evaluation of GAN performance. The drawback of IS is that it evaluates GANs by only considering the statistics of the generated images rather than the statistics of the real-world samples. In other words, IS only evaluates the quality and diversity of the generated images and does not directly compare the similarity between the generated and real-world images. Equation (10), on the other hand, measures the statistical similarity between the quality of the generated images and the real image by measuring the distance between two probability distributions. For FID, the lower the evaluation score, the better the performance of the model.

$$\mathrm{FID} = \left\|\mu_r - \mu_g\right\|^2 + T_r\left(\sum_r + \sum_g - 2\left(\sum_r \sum_g\right)^{\frac{1}{2}}\right) \tag{10}$$

The learned perceptual image patch similarity (LPIPS) metric was created to address the inability of L2/PSNR [56], SSIM, and FSIM [57] to reflect human perceptual abilities. LPIPS utilizes the image classification models SqueezeNet [58], AlexNet [49], and VGG [59] to attempt to measure similarity based on human perception, in contrast to IS and FID. Given an image x, x_0 in Equation (11), a pre-trained network of ImageNets [49] is fed an activation map $\hat{y}^l, \widehat{y_0^l} \in \mathbb{R}^{H_l \times W_l \times C_l}$ at layer l. The activation map is then fed to the network in Equation (11). The activation map is scaled by ω_l, and the Euclidean distance [60] is then calculated. Finally, the LPIPS can be calculated by spatially averaging and summing per channel. The lower the LPIPS, the more similar they are.

$$d(x, x_0) = \sum_l \frac{1}{H_l W_l} \sum_{h,\omega} \left|\left|\omega_l \odot \left(\hat{y}^l_{hw} - \hat{y}^l_{0hw}\right)\right|\right| \tag{11}$$

4. Results

The quantitative metric FID is an evaluation metric that verifies the statistical similarity of images by comparing the distance between the distribution of the actual image and the distribution of the generated image, and LPIPS is a method that compares the similarity of the characteristics of images through a model that reflects human perceptual abilities. Lists the qualitative and quantitative evaluation of five GAN models in Table 2: GAN, DCGAN, StarGAN, StyleGAN v2, and ProjectedGAN, including the image synthesis model.

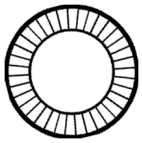
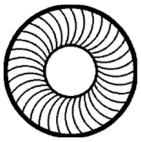
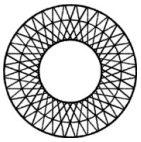
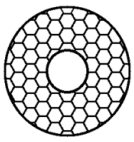
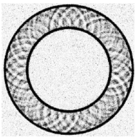
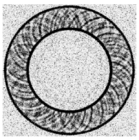
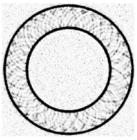
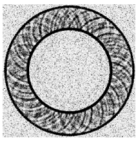
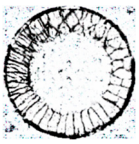
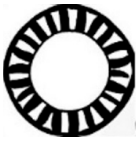
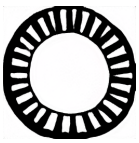
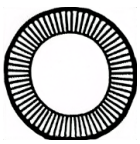
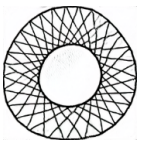
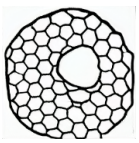
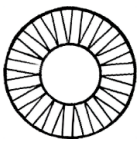
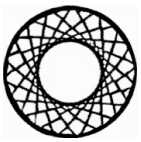
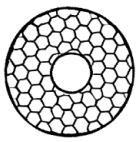
In the case of GAN, large amounts of noise were generated in the background image, and the shape of the pattern was barely visible. However, with less training data, mode collapse could be observed. We did not measure FID and LPIPS separately because the qualitative evaluation yielded meaningless results.

For DCGAN, we had the same background noise as GAN, but no mode collapse occurred, and we saw that it learned a variety of patterns. However, we had a problem mixing multiple patterns. The FID was 192.02, and the LPIPS was 0.4986.

StarGAN v2 failed to learn all domain types except for spoke, as listed in Table 2. For the curve type, the spoke appears to bend a little, but not in the form of a pattern. For the triangle and honeycomb types, no images were generated at all. We did not measure quantitative evaluation metrics because they were deemed meaningless through the qualitative evaluation.

StyleGAN v2-ADA generated high-quality, noise-free images with less data. However, it was unable to maintain the circle, resulting in distortion, and the patterns were distorted accordingly. However, compared to GAN and DCGAN, it produced more realistic images. The evaluation metrics FID and LPIPS were 42.88 and 0.3806, respectively, exhibiting a significant reduction compared to DCGAN.

Table 2. Training results and evaluation metrics based on GAN models.

	Spoke	Curve	Triangle	Honeycomb	FID	LPIPS	Notes
Real					-	-	-
GAN					-	-	Mode collapse
DCGAN					192.02	0.4986	-
StarGAN v2					-	-	Mode collapse and failure to learn the domain
StyleGAN v2-ADA					42.88	0.3806	-
Projected GAN					13.41	0.3477	-

The image generation from ProjectedGAN exhibited no noise in the background, and the resolution of the image was kept high. It also showed that the circle was not distorted, and the pattern was consistently drawn. In terms of qualitative evaluation, the FID was 13.41, exhibiting a reduction of approximately 30 compared to StyleGAN v2-ADA, and LPIPS was reduced to 0.3477, corresponding to the best result among the GANs used in this paper.

5. Conclusions

This study presents a survey of GAN models and their improvements and provides a detailed analysis of the application of patterns to GANs based on primitive shapes. At its core, this study addressed the fact that GAN models trained on complex image data such as people and Pokémon, which have been covered in previous studies, exhibit good performance in generating realistic images, but a large performance gap exists in generating images with simple shapes and regular patterns based on GAN models [61]. That is because achieving perfect symmetry requires simple shapes and regular patterns, which can be more challenging than training using humans and Pokémon. In practice, we found that GAN and DCGAN retained the circle but failed to learn the pattern domain and generated large amounts of noise in the background. StarGAN v2 learned the plate spoke domain but failed to learn the remaining domains. StyleGAN v2-ADA seemingly learned the patterns well but was unable to maintain circularity and suffered from distortion. In conclusion, we demonstrate that the image generated by ProjectedGAN best maintains the patterns based

on primitive shapes, and the tire is neither circular nor distorted compared to other GANs. In keeping with the qualitative evaluation, the quantitative measures FID and LPIPS were 13.41 and 0.3477, respectively, which were superior among the four models.

Non-pneumatic tires are widely used in construction and agriculture, but their design requires expertise from skilled professionals with years of experience, and finding structured rules is difficult, making full or partial automation a challenge. In this study, we conducted a wide range of research on how to generate basic images of pneumatic tires in the most effective way using various techniques of artificial intelligence, which will greatly increase the efficiency of pneumatic tire design.

Author Contributions: J.Y.S.—data collection and experimentation; S.-m.J. and D.-h.C.—software development and analysis; S.L. (Seungjae Lee)—providing research direction and writing papers; S.L. (Sungchul Lee)—review and editing. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the Sun Moon University Research Grant.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data used in this paper can be made available by contacting the corresponding author, subject to availability.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Gent, A.N.; Walter, J.D. Pneumatic Tire. In *Mechanical Engineering Faculty Research*; National Highway Traffic Safety Administration: Washington DC, USA, 2006; p. 854.
2. Mykola, K.; Prentkovskis, O.; Skačkauskas, P. Comparison Analysis Between Pneumatic and Airless Tires by Computational Modelling for Avoiding Road Traffic Accidents. In Proceedings of the International Conference on Reliability and Statistics in Transportation and Communication, Riga, Latvia, 20–21 October 2022; pp. 295–305.
3. Kim, K.W.; Kwark, C.W. Introduction to Technology Trends, Problems and Solutions of Non-Pneumatic. *J. Korean Soc. Automot. Eng.* **2019**, *41*, 26–31.
4. Chavan, S.S.; Avhad, S.P.; Chavan, S.R. Study of tweel non-pneumatic tires. *Int. J. Res. Appl. Sci. Eng. Technol.* **2022**, *10*, 1047–1051. [CrossRef]
5. Sardinha, M.; Reis, L.; Ramos, T.; Vaz, M.F. Non-pneumatic tire designs suitable for fused filament fabrication: An overview. *Procedia Struct. Integr.* **2022**, *42*, 1098–1105. [CrossRef]
6. Goodfellow, I.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; Bengio, Y. Generative adversarial networks. *Commun. ACM* **2020**, *63*, 139–144. [CrossRef]
7. Karras, T.; Laine, S.; Aila, T. A Style-Based Generator Architecture for Generative Adversarial Networks. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 15–20 June 2019; pp. 4401–4410.
8. Gatys, L.A.; Ecker, A.S.; Bethge, M. A neural algorithm of artistic style. *arXiv* **2015**, arXiv:1508.06576. [CrossRef]
9. Mao, H.; Cheung, M.; She, J. Deepart: Learning joint representations of visual arts. In Proceedings of the 25th ACM International Conference on Multimedia, Mountain View, CA, USA, 23–27 October 2017; pp. 1183–1191.
10. Donovan-Maiye, R.M.; Brown, J.M.; Chan, C.K.; Ding, L.; Yan, C.; Gaudreault, N.; Theriot, J.A.; Maleckar, M.M.; Knijnenburg, T.A.; Johnson, G.R. A deep generative model of 3D single-cell organization. *PLoS Comput. Biol.* **2022**, *18*, e1009155. [CrossRef]
11. Kim, S.W.; Zhou, Y.; Philion, J.; Torralba, A.; Fidler, S. Learning to simulate dynamic environments with gamegan. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 14–19 June 2020; pp. 1231–1240.
12. Ratliff, L.J.; Burden, S.A.; Sastry, S.S. Characterization and computation of local Nash equilibria in continuous games. In Proceedings of the 51st Annual Allerton Conference on Communication, Control, and Computing (Allerton), Monticello, IL, USA, 2–4 October 2013; IEEE: Piscataway, NJ, USA, 2013.
13. Goodfellow, I. Nips 2016 tutorial: Generative adversarial networks. *arXiv* **2016**, arXiv:1701.00160.
14. Salimans, T.; Goodfellow, I.; Zaremba, W.; Cheung, V.; Radford, A.; Chen, X. Improved techniques for training gans. *Adv. Neural Inf. Process. Syst.* **2016**, *29*, 03498. [CrossRef]
15. Arjovsky, M.; Chintala, S.; Bottou, L. Wasserstein generative adversarial networks. In Proceedings of the International Conference on Machine Learning, Sydney, Australia, 6–11 August 2017; pp. 214–223.
16. Zhao, S.; Liu, Z.; Lin, J.; Zhu, J.Y.; Han, S. Differentiable augmentation for data-efficient gan training. *Adv. Neural Inf. Process. Syst.* **2020**, *33*, 7559–7570.

17. Ulyanov, D.; Vedaldi, A.; Lempitsky, V. Instance normalization: The missing ingredient for fast stylization. *arXiv* **2016**, arXiv:1607.08022.
18. Radford, A.; Metz, L.; Chintala, S. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv* **2015**, arXiv:1511.06434.
19. Choi, Y.; Uh, Y.; Yoo, J.; Ha, J.W. Stargan v2: Diverse image synthesis for multiple domains. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 14–19 June 2020; pp. 8188–8197.
20. Karras, T.; Aittala, M.; Hellsten, J.; Laine, S.; Lehtinen, J.; Aila, T. Training generative adversarial networks with limited data. *Adv. Neural Inf. Process. Syst.* **2020**, *33*, 12104–12114.
21. Sauer, A.; Chitta, K.; Müller, J.; Geiger, A. Projected gans converge faster. *Adv. Neural Inf. Process. Syst.* **2021**, *34*, 17480–17492.
22. 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. *Adv. Neural Inf. Process. Syst.* **2017**, *30*, 6629–6640.
23. 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 IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 586–595.
24. Brock, A.; Donahue, J.; Simonyan, K. Large scale GAN training for high fidelity natural image synthesis. *arXiv* **2018**, arXiv:1809.11096.
25. Rhyne, T.B.; Cron, S.M. Development of a non-pneumatic wheel. *Tire Sci. Technol.* **2006**, *34*, 150–169. [CrossRef]
26. Deng, Y.; Wang, Z.; Shen, H.; Gong, J.; Xiao, Z. A comprehensive review on non-pneumatic tyre research. *Mater. Des.* **2023**, *227*, 111742. [CrossRef]
27. Ju, J.; Kim, D.M.; Kim, K. Flexible cellular solid spokes of a non-pneumatic tire. *Compos. Struct.* **2012**, *94*, 2285–2295. [CrossRef]
28. Sim, J.; Hong, J.; Cho, I.; Lee, J. Analysis of vertical stiffness characteristics based on spoke shape of non-pneumatic tire. *Appl. Sci.* **2021**, *11*, 2369. [CrossRef]
29. Wang, Z.; Bovik, A.C.; Sheikh, H.R.; Simoncelli, E.P. Image quality assessment: From error visibility to structural similarity. *IEEE Trans. Image Process.* **2004**, *13*, 600–612. [CrossRef]
30. Chemane, L.; Mapsanganhe, S. Distributed Government e-Mail Service: Mozambique GovNet Case Study. In Proceedings of the 2010 IST-Africa, Durban, South Africa, 19–21 May 2010; pp. 1–9.
31. O'Shea, K.; Nash, R. An introduction to convolutional neural networks. *arXiv* **2015**, arXiv:1511.08458.
32. Kingma, D.P.; Ba, J. Adam: A method for stochastic optimization. *arXiv* **2014**, arXiv:1412.6980.
33. Springenberg, J.T.; Dosovitskiy, A.; Brox, T.; Riedmiller, M. Striving for simplicity: The all convolutional net. *arXiv* **2014**, arXiv:1412.6806.
34. Lin, M.; Chen, Q.; Yan, S. Network in network. *arXiv* **2013**, arXiv:1312.4400.
35. 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, 6–11 July 2015; pp. 448–456.
36. Nair, V.; Hinton, G.E. Rectified linear units improve restricted boltzmann machines. In Proceedings of the 27th International Conference on Machine Learning (ICML-10), Haifa, Israel, 21–24 June 2010.
37. Maas, A.L.; Hannun, A.Y.; Ng, A.Y. Rectifier nonlinearities improve neural network acoustic models. In Proceedings of the the 30th International Conference on Machine Learning, Atlanta, GA, USA, 16–21 June 2013; p. 3.
38. LeCun, Y.; Bottou, L.; Bengio, Y.; Haffner, P. Gradient-based learning applied to document recognition. *Proc. IEEE* **1998**, *86*, 2278–2324. [CrossRef]
39. Bengio, S.; Bengio, Y.; Cloutier, J.; Gecsei, J. On the optimization of a synaptic learning rule. In *Optimality in Biological and Artificial Networks?* Routledge: London, UK, 2013; pp. 281–303.
40. Gholamalinezhad, H.; Khosravi, H. Pooling methods in deep neural networks, a review. *arXiv* **2020**, arXiv:2009.07485.
41. Nwankpa, C.; Ijomah, W.; Gachagan, A.; Marshall, S. Activation functions: Comparison of trends in practice and research for deep learning. *arXiv* **2018**, arXiv:1811.03378.
42. DCGAN TUTORIAL. Available online: https://pytorch.org/tutorials/beginner/dcgan_faces_tutorial.html (accessed on 2 June 2023).
43. Huang, X.; Belongie, S. Arbitrary style transfer in real-time with adaptive instance normalization. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 1501–1510.
44. Stargan-v2. Available online: https://github.com/clovaai/stargan-v2 (accessed on 2 June 2023).
45. Srivastava, N.; Hinton, G.; Krizhevsky, A.; Sutskever, I.; Salakhutdinov, R. Dropout: A simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.* **2014**, *15*, 1929–1958.
46. Zhao, Z.; Singh, S.; Lee, H.; Zhang, Z.; Odena, A.; Zhang, H. Improved Consistency Regularization for GANs. *Proc. AAAI Conf. Artif. Intell.* **2021**, *35*, 11033–11041. [CrossRef]
47. Bora, A.; Price, E.; Dimakis, A.G. AmbientGAN: Generative models from lossy measurements. In Proceedings of the International Conference on Learning Representations, Vancouver, BC, Canada, 30 April–3 May 2018.
48. Miyato, T.; Kataoka, T.; Koyama, M.; Yoshida, Y. Spectral normalization for generative adversarial networks. *arXiv* **2018**, arXiv:1802.05957.
49. Tan, M.; Le, Q. Efficientnet: Rethinking model scaling for convolutional neural networks. In Proceedings of the International Conference on Machine Learning, Long Beach, CA, USA, 9–15 June 2019; pp. 6105–6114.

50. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. Imagenet classification with deep convolutional neural networks. *Commun. ACM* **2017**, *60*, 84–90. [CrossRef]
51. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
52. Dosovitskiy, A.; Beyer, L.; Kolesnikov, A.; Weissenborn, D.; Zhai, X.; Unterthiner, T.; Dehghani, M.; Minderer, M.; Heigold, G.; Gelly, S.; et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv* **2020**, arXiv:2010.11929.
53. Touvron, H.; Cord, M.; Douze, M.; Massa, F.; Sablayrolles, A.; Jégou, H. Training data-efficient image transformers & distillation through attention. In Proceedings of the International Conference on Machine Learning, Virtual, 18–24 July 2021; pp. 10347–10357.
54. Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; Rabinovich, A. Going deeper with convolutions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 1–9.
55. Shlens, J. Notes on kullback-leibler divergence and likelihood. *arXiv* **2014**, arXiv:1404.2000.
56. Korhonen, J.; You, J. Peak signal-to-noise ratio revisited: Is simple beautiful? In Proceedings of the 2012 Fourth International Workshop on Quality of Multimedia Experience, Melbourne, Australia, 5–7 July 2012; pp. 37–38.
57. Zhang, L.; Zhang, L.; Mou, X.; Zhang, D. FSIM: A feature similarity index for image quality assessment. *IEEE Trans. Image Process.* **2011**, *20*, 2378–2386. [CrossRef]
58. Iandola, F.N.; Han, S.; Moskewicz, M.W.; Ashraf, K.; Dally, W.J.; Keutzer, K. SqueezeNet: AlexsNet-level accuracy with 50× fewer parameters and <0.5 MB model size. *arXiv* **2016**, arXiv:1602.07360.
59. Simonyan, K.; Zisserman, A. Very deep convolutional networks for large-scale image recognition. *arXiv* **2014**, arXiv:1409.1556.
60. Dokmanic, I.; Parhizkar, R.; Ranieri, J.; Vetterli, M. Euclidean distance matrices: Essential theory, algorithms, and applications. *IEEE Signal Process. Mag.* **2015**, *32*, 12–30. [CrossRef]
61. Sauer, A.; Schwarz, K.; Geiger, A. Stylegan-xl: Scaling stylegan to large diverse datasets. In Proceedings of the ACM SIGGRAPH 2022 Conference Proceedings, Vancouver, BC, Canada, 7–11 August 2022.

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.