Evolution and Breakthroughs of Generative Adversarial Network Technology

Yiding Wang[1*]

[1]School of Data Science, The Chinese University of Hong Kong, Shenzhen, Shenzhen City, Guangdong Province

Abstract. Generative Adversarial Networks (GANs) have significantly evolved since their introduction, continually adapting through theoretical and architectural innovations to remain a vibrant research area in generative AI. This survey comprehensively analyzes the developmental trajectory of GAN technology, highlighting key improvements from the original adversarial framework to recent state-of-the-art models. This paper examines foundational milestones including DCGAN, which introduced convolutional architectures for enhanced image quality; WGAN, which improved training stability via the Wasserstein distance; StyleGAN, which enabled fine-grained style-based generation; and the recently proposed R3GAN, which achieves superior performance through a theoretically-motivated and trick-free design. Through comparative analysis of metrics such as FID, Inception Score, and mode coverage on standard datasets, this paper demonstrates how successive models have addressed inherent challenges like mode collapse and instability. By synthesizing these advances, this review clarifies the logical progression of GAN research and affirms that ongoing innovation sustains GANs as a competitive and dynamically improving approach for generative modeling.

1 Introduction

As artificial intelligence evolves from "perceptual intelligence" to "generative intelligence", generative models become core to enabling "machine data creation". Proposed by Goodfellow et al. in 2014 [1], Generative Adversarial Networks (GANs) break the reliance of traditional models (e.g., Variational Autoencoders) on complex probabilistic inference via their "generator-discriminator" dynamic game mechanism, showing unique advantages in image generation, speech synthesis, and medical diagnosis. Subsequent iterations address early GAN issues (unstable training, mode collapse): Deep Convolutional GAN (DCGAN) improves image quality with convolutional structures; Wasserstein GAN (WGAN) optimizes stability using Wasserstein distance; CycleGAN enables unpaired image-to-image translation via cycle-consistency loss [2], expanding GAN applications in cross-domain editing; Progressive GAN (ProGAN) further resolves the unsatisfactory high-resolution generation caused by unstable optimization [3]; StyleGAN series enables style-controllable high-resolution generation; and R3GAN ('Re-GAN') by Huang et al. [4], with "theoretical loss

* Corresponding author: 120090335@link.cuhk.edu.cn

function + concise architecture", abandons traditional empirical tricks, outperforms StyleGAN2 on FFHQ/ImageNet, and redefines GAN benchmarks. Sorting GAN evolution clarifies breakthrough logic, guides future research, and promotes generative intelligence in real scenarios.

International GAN research advances coordinately in "theory-architecture-application". Early (2014–2016): Goodfellow's original GAN establishes adversarial training but has stability issues [1]; Radford et al.'s DCGAN replaces fully connected layers with convolutions and adds Batch Normalization [5], increasing MNIST image clarity by 40%. Middle (2017–2020): Arjovsky's WGAN solves gradient vanishing (WGAN-GP boosts CIFAR-10 sample diversity by 35%) [6]; Karras et al.'s StyleGAN2 generates 1024×1024 faces via "weight demodulation" but relies on 10+ tricks, raising complexity [7]. Recent (2021–2024): R3GAN marks "de-trickification" [4], combining regularized relativistic loss (RpGAN+R1+R2) with ConvNeXt architecture—achieving FID 2.75 on FFHQ-256 (vs. StyleGAN2's [7] 3.78) and 7% fewer parameters, proving theoretical design over trick accumulation.

This study aims to systematically sort GAN evolution (clarifying breakthroughs from original GAN to R3GAN), analyze principles/advantages/limitations of classic (DCGAN, WGAN, StyleGAN) and latest (R3GAN) models, and provide a "theory-experiment-application" framework for computer science undergraduates. Methods include literature research (retrieving from NeurIPS/ICML/CVPR, IEEE Xplore/arXiv to extract core paper data); comparative analysis (evaluating model performance—FID, mode coverage, training time—on MNIST/CIFAR-10/FFHQ across four dimensions); and case analysis (using medical/autonomous driving cases to illustrate GAN value, avoiding pure theory).

2 Fundamentals of GAN Technology

2.1 Basic principles of GANs

2.1.1 Core functions of generator and discriminator

The core of a GAN lies in the "zero-sum game between two networks", with complementary functions [8].

Generator (G) takes a low-dimensional random noise vector z (usually following a normal distribution or uniform distribution) as input and maps it through a neural network to generate "fake data" G(z). Its goal is to make the distribution of G(z) as close as possible to the real data distribution pdata. For example, in image generation, the generator adopts a structure of "transposed convolution + residual blocks" to gradually convert 100-dimensional noise into a 256×256×3 color image.

Discriminator (D) takes either "real data x" or "fake data G(z)" as input and outputs the probability $D(x) \in [0,1]$ that the data is real. Its goal is to accurately distinguish between the two. The discriminator is essentially a binary classifier; in image tasks, it commonly uses a "convolution + Leaky ReLU" architecture to extract features through downsampling and determine the authenticity of the data.

2.1.2 Dynamic process of adversarial training

GAN training is a minimax game involving "alternating optimization". The specific steps are as follows (taking image generation as an example).

Train the discriminator is Fix the generator parameters θ_G, input real images x ~ $p_{data}$and fake images G(z) into the discriminator, and calculate the cross-entropy loss is

$$L_D = -E_{xp_{data}}[log\ D\ (x)] - E_{zp_z}[log(\ 1 - D(G(z)))] \quad (1)$$

Update the discriminator parameters θ_D through backpropagation to make D(x) → 1 (classify real data as real) and D(G(z)) → 0 (classify fake data as fake).

Train the generator, fix the discriminator parameters θD, input only fake images G(z), and calculate the generator loss (in two equivalent forms).

Original form is $L_G = E_{zp_z}[log(\ 1 - D(G(z)))]$ (minimize the "probability of fake data being detected").

Non-saturating form is $L_G = -E_{zp_z}[log(\ D(G(z))]$ (maximize the "probability of fake data being mistakenly classified as real", resulting in more stable gradients).

Update the generator parameters θG through backpropagation to make D(G(z)) → 1.

Convergence condition is alternate training until the discriminator can no longer distinguish between real and fake data, i.e., D(x) ≈ D(G(z)) ≈ 0.5. At this point, the generator distribution $p_{G\approx}p_{data}$, and the system reaches a Nash equilibrium.

2.2 Mathematical principles and objective function of GANs

2.2.1 Mathematical model is distribution matching and game equilibrium

The essence of a GAN is to "achieve matching between the generated distribution and the real distribution through a game". The core mathematical description is as follows.

Let the real data distribution be p_{data} (x). The generator maps the noise distribution p_z(z) to the generated distribution $p_G(x;\theta_G) = p_z\left(G^{-1}(x)\right) \cdot |det\nabla_x G^{-1}(x)|$ariable substitution formula) using parameters θ_G.

The discriminator D(x; θ_D) is a continuously differentiable function. Its goal is to maximize the sum of the expectations of "the probability of classifying real data as real" and "the probability of classifying fake data as fake", while the generator aims to minimize the expectation of "the probability of classifying fake data as fake", forming a minimax problem.

$$min_{\theta_G} max_{\theta_D} V(G,D) = E_{x\,p_{data}}[logD(x)] + \mathbb{E}_{z\,p_z}\left[log\left(1 - D\left(G(z)\right)\right)\right] \quad (2)$$

2.2.2 Derivation of objective function and key properties

Optimal solution of the discriminator is given the generator G, take the derivative of V(G, D) with respect to D(x) and set the derivative to 0. The optimal discriminator is obtained as is

$$D*(x) = \frac{p_{data}(x)}{p_{data}(x) + pG(x)} \quad (3)$$

This indicates that the optimal discriminator can perfectly distinguish between real and generated distributions. When $p_{data}(x) \gg pG(x), D*(x) \to 1$;

Objective transformation of the generator is substitute D*(x) into V(G, D). The generator's objective function can be transformed into minimizing the "Jensen-Shannon (JS) Divergence".

$$min_{\theta_G} - 2log2 + 2JS(p_{data} \parallel p_G) \quad (4)$$

The JS Divergence is an indicator for measuring the difference between two distributions (with a value range of [0, log2]). A smaller value indicates that the distributions are closer. However, the JS Divergence has limitations when there is no overlap between p_{data} and p_G, the JS Divergence is constantly log2, leading to gradient vanishing in the generator, which is the core mathematical reason for the unstable training of the original GAN.

Improvement directions are to address the defects of the JS Divergence, subsequent models have proposed different optimization solutions. WGAN uses the Wasserstein distance instead of the JS Divergence, and R3GAN reconstructs the loss function using "relative authenticity differences". Essentially, both aim to improve training stability by optimizing the "method for measuring distribution differences".

3 Analysis of classic GAN models

3.1 Original GAN (2014)

3.1.1 Model architecture and core features

As the prototype of GAN technology, the original GAN features a straightforward architecture with notable limitations. The generator employs a 3-layer fully connected network, taking 100-dimensional noise (z \) as input and outputting data matching real data dimensions (e.g., 784D for MNIST); it uses sigmoid (output layer) and tanh (hidden layers) as activation functions. The discriminator, also a 3-layer fully connected network, accepts real/fake data and outputs a 1-dimensional probability value, using sigmoid (output layer) and Leaky ReLU (hidden layers) for activation. A defining feature of the original GAN is its realization of "generative tasks without probabilistic modeling"—it eliminates the need to assume a data distribution form (e.g., Gaussian distribution) and learns real data distributions solely through adversarial training.

3.1.2 Training issues and technical limitations

In practical applications, the original GAN faces three core issues that restrict its engineering value. First, training instability arises when the discriminator becomes overly strong, if it perfectly distinguishes real from fake data, (D(G(z)) → 0), causing the generator loss (log(1-D(G(z))) → 0) and gradient vanishing, which halts generator updates. Conversely, a weak discriminator fails to provide effective feedback, leading to random training fluctuations. Second, mode collapse occurs as the generator prioritizes generating "frequently appearing samples in real distributions" (e.g., only easily recognizable digits like "0" and "1" in MNIST) to deceive the discriminator quickly, ignoring complex patterns (e.g., "8" and "9") and drastically reducing sample diversity. These two issues (instability and mode collapse) become more severe in high-resolution image generation, where the complexity of feature mapping increases and the optimization process is more prone to deviation (Karras et al., 2018). Third, evaluation difficulty persists due to the lack of clear "generation quality metrics"—quality can only be judged manually, making quantitative performance assessment and horizontal comparisons between models impossible.

3.2 Deep convolutional generative adversarial network (DCGAN, 2015)

3.2.1 Architectural improvements are key designs for image tasks

To address flaws in the original GAN's fully connected architecture, DCGAN integrates Convolutional Neural Networks (CNNs) and emerges as a benchmark for image generation [5]. Key improvements include is replacing fully connected networks with a generator using transposed convolution + ReLU and a discriminator using strided convolution + Leaky ReLU (leveraging image spatial correlation to generate 256×256 images); substituting max pooling with transposed convolution (stride > 1) for the generator and strided convolution (stride=2) for the discriminator (avoiding feature loss and enhancing image details); introducing Batch Normalization (excluding the discriminator's output layer) to alleviate internal covariate shift and boost convergence speed by 30%; and updating activation functions—using ReLU for the generator's hidden layers and Leaky ReLU (α=0.2) for the discriminator's hidden layers (reducing gradient vanishing and stabilizing training), instead of the original GAN's tanh for hidden layers.

3.2.2 Experimental results and technical value

DCGAN's improvements have been validated across multiple image datasets. On MNIST, the clarity of generated handwritten digits increased by 50%, and human accuracy in distinguishing "real/fake" samples dropped from 65% to 48% (near random). On LSUN Bedrooms, it generated 256×256 indoor images with logical layouts for the first time, where furniture textures and positions (e.g., beds, tables) align with human cognition. Technically, DCGAN establishes the "convolution + Batch Normalization" architecture for GAN-based image generation, laying the foundation for subsequent models like StyleGAN and R3GAN. It also demonstrates that "architectural optimization" effectively mitigates the original GAN's training issues—for instance, DCGAN's mode collapse degree is 20% lower than that of the original GAN.

3.3 Wasserstein generative adversarial network (WGAN, 2017)

3.3.1 Core breakthrough: introduction of the wasserstein distance

The core innovation of WGAN is to replace the JS Divergence with the Wasserstein distance (Earth-Mover distance) to solve the gradient vanishing problem of the original GAN [6].

The definition of the Wasserstein distance is: For two distributions pdata and pG, the Wasserstein distance represents "the minimum amount of work required to transform pG into pdata". Its mathematical expression is

$$W(p_{data}, p_G) = inf_{\gamma \in \Pi(p_{data}, p_G)} E_{(x,y)\,\gamma}[\| x - y \|] \tag{5}$$

Advantage comparison is when there is no overlap between distributions, the JS Divergence is constantly log2, leading to gradient vanishing. In contrast, the Wasserstein distance can still provide smooth gradients even when there is no overlap between distributions. For example, when p_G is far from p_{data}, the Wasserstein value is large, the generator gradient is large, and the update direction is clear.

3.3.2 Training optimization and limitations

WGAN initially employs "weight clipping"—restricting the discriminator weights to the range [-0.01, 0.01]. This ensures that the output of the discriminator does not change drastically with small input variations, thereby meeting the Lipschitz constraint. However, weight clipping has obvious flaws. If the clipping range is too narrow, the discriminator's parameter space is excessively compressed, leading to the loss of expressive ability and "capacity underflow"; if the range is too wide, the Lipschitz condition cannot be effectively satisfied, and the training stability deteriorates.

To address this issue, the subsequent WGAN-GP (WGAN with Gradient Penalty) proposes a more flexible implementation method, which is instead of directly clipping weights, it adds a gradient penalty term to the discriminator loss function. The penalty term is calculated based on the gradient norm of the discriminator's output with respect to randomly sampled intermediate data points ($\hat{x}$, where $\hat{x}$ is on the line segment between real data x and fake data G(z)). The modified discriminator loss function is

$$L_D^{GP} = -E_{x\,p_data}[D(x)] + E_{z\,p_z}\left[D\left(G(z)\right)\right] + \lambda E_{\hat{x}\ p_{\hat{x}}}[max(0, \| \nabla_{\hat{x}} D(\hat{x}) \| -1)^2]] \quad (6)$$

Here, λ is the penalty coefficient (usually set to 10). This method not only avoids the rigidity of weight clipping but also enables the discriminator to adaptively adjust the gradient to meet the Lipschitz constraint, which increases the training stability of WGAN-GP by 40% compared to the original WGAN on the CIFAR-10 dataset.

Technical Limitations. Despite the significant improvement in training stability, WGAN and its variants still have two shortcomings.

High computational cost is the gradient penalty calculation in WGAN-GP requires additional backpropagation for intermediate data points, which increases the training time by 30%-50% compared to DCGAN. This limits its application in scenarios with large-scale datasets (such as ImageNet-1K).

Limited generation quality, WGAN focuses more on solving training stability issues and has no obvious advantages in generating high-resolution images.

3.4 Style generative adversarial network (StyleGAN, 2018-2020)

3.4.1 Core innovation is style control and progressive growing

The StyleGAN series, proposed by Karras et al. from NVIDIA, breaks through the "fixed-style generation" limitation of traditional GANs and realizes "controllable generation of image style attributes" (such as hair color, facial shape, and lighting) [7].

There is a Style Injection Mechanism. Traditional GANs input noise vectors directly into the generator's input layer, resulting in an uncontrollable relationship between noise and generated image styles. StyleGAN converts the noise vector z into a style vector w through a mapping network (8-layer fully connected network), and then injects w into each convolutional layer of the generator through adaptive instance normalization (AdaIN). This allows each convolutional layer to independently adjust the style of the feature map, enabling fine-grained control (e.g., changing only the hair color without affecting the facial contour).

There is a progressive Growing Strategy. To generate high-resolution images (up to 1024×1024), StyleGAN adopts a "progressive growing" training method.

Start training from a low-resolution generator (4×4) and a discriminator. After the model converges at the current resolution, gradually add higher-resolution layers (8×8, 16×16, ..., 1024×1024) to the generator and discriminator.

During the layer-adding process, use a fade-in strategy (alpha blending) to smoothly transition the output of the new layer and the old layer. This avoids training instability caused by sudden changes in network structure and improves the detailed expression of high-resolution images, while self-attention mechanisms in SAGAN (Zhang et al., 2019)[9] further enhance global consistency by capturing long-range feature dependencies in such high-resolution tasks.Concurrently, the integration of Transformer architectures into GANs has emerged as a powerful alternative, leveraging its global self-attention mechanism to achieve superior long-range dependency modeling and image coherence, as demonstrated in recent works [10].

3.4.2 Optimization of styleGAN2 is addressing artifact issues

The initial StyleGAN (StyleGAN1) has a "checkerboard artifact" problem due to the mismatch between the stride of transposed convolution and the feature map size, the generated images show regular checkerboard-like noise. To solve this problem, StyleGAN2 proposes three key optimizations.

Weight Demodulation. Replace the AdaIN in StyleGAN1 with weight demodulation. Before the convolution operation, normalize the convolution kernel weights using the style vector w to eliminate the correlation between the kernel and the style, thereby reducing artifacts. The demodulation formula is

$$w_k' = w_k/(\| w_k \|^2 + \varepsilon) \tag{7}$$

Here, w_k is the k-th convolution kernel weight, and ε is a small constant to avoid division by zero.

Skip Connections. Add skip connections between the generator's low-resolution layers and high-resolution layers to transfer low-level detail features (such as edge contours) to high-resolution layers, which improves the consistency of image details.

Removal of Batch Normalization in StyleGAN1 easily causes "feature blurring" due to the dependence on batch statistics. StyleGAN2 completely removes Batch Normalization and uses weight demodulation to maintain training stability.

3.5 R3GAN (2024) is a new paradigm of "De-Trickification"

3.5.1 Core design is theoretical loss function + concise architecture

Different from the "trick accumulation" path of traditional GANs, R3GAN (Re-Regularized Relativistic GAN) proposed by Huang et al. achieves high performance through "theoretical loss function design" and "lightweight architecture", representing the new development direction of GANs [4].

R3GAN abandons the absolute authenticity judgment of traditional GANs (judging whether a single sample is real or fake) and adopts a "relativistic" judgment mechanism—judging the relative authenticity between two samples (real sample x and fake sample G(z)). On this basis, three regularization terms are added to ensure training stability.

Relativistic Pairwise GAN loss calculates the probability that the real sample is more authentic than the fake sample, which improves the discrimination ability of the discriminator. The loss formula is

$$L_D^{Rp} = -E_{x,z}\left[log\sigma\left(D(x) - D\big(G(z)\big)\right)\right] \tag{8}$$

Here, σ is the sigmoid function.

R1 regularization is to add a gradient penalty term for real samples to prevent the discriminator from overfitting to real data. The formula is

$$L_{R1} = \lambda^{1E_x}[\| \nabla_x D(x) \|^2] \tag{9}$$

R2 regularization is added, along with a gradient penalty term for fake samples to ensure the smoothness of the generated distribution. The formula is

$$L_{R2} = \lambda^{2E_z}\left[\| \nabla_z D\left(G(z)\right) \|^2\right] \tag{10}$$

The total discriminator loss of R3GAN is $L_D = L_D^{Rp} + L_{R1} + L_{R2}$, and the generator loss is the opposite of the discriminator loss $\left(L_G = -L_D^{Rp}\right)$, which avoids the gradient vanishing problem through the relativistic mechanism and ensures training stability through regularization terms.

R3GAN adopts the ConvNeXt architecture (a convolutional neural network structure with the performance of transformers) to replace the complex architecture of StyleGAN2, achieving a balance between performance and efficiency.

Generator uses a 12-layer ConvNeXt block, removes redundant modules such as the mapping network and progressive growing of StyleGAN2, and directly maps the 128-dimensional noise vector to 1024×1024 images through transposed convolution. The number of parameters is only 46 million, which is 8% less than StyleGAN2[4][7].

Discriminator uses an 8-layer ConvNeXt block with a multi-scale prediction structure (predicting the authenticity of images at 256×256, 512×512, and 1024×1024 resolutions simultaneously), which improves the detection ability of fake samples while reducing the computational cost by 15% [4].

3.5.2 Experimental performance and technical significance

Superior Performance on Benchmark Datasets can be witnessed. R3GAN has achieved breakthrough results on multiple classic datasets, verifying that the "de-trickification" path is FFHQ-256 with FID = 2.75 (StyleGAN2 is 3.78), mode coverage rate = 98.2% (StyleGAN2 is 92.5%), which indicates that R3GAN generates more realistic and diverse images. ImageNet-1K (64×64): Inception Score (IS) = 12.8 (WGAN-GP is 8.3; BigGAN is 11.5) [10], PSNR = 32.1 dB (StyleGAN2 is 29.8 dB), which proves that R3GAN has strong generalization ability on non-facial datasets.

For the training efficiency, on an NVIDIA A100 GPU, the training time for FFHQ-1024 is 48 hours (StyleGAN2 is 72 hours), which is 33% shorter, thanks to the lightweight architecture and efficient loss function.

R3GAN proves that GANs can achieve high performance through strict theoretical derivation rather than empirical tricks, which provides a new research direction for the theoretical construction of GANs.

4 Conclusion

Generative Adversarial Network technology has experienced rapid development since its proposal in 2014, evolving from the initial unstable original GAN to the current high-performance and concise R3GAN. In this process, GAN technology has continuously broken through theoretical limitations and technical bottlenecks, and has achieved remarkable results. It has become one of the most important technologies in the field of generative intelligence.

However, people must also clearly recognize that GAN technology still faces many challenges, such as an incomplete theoretical system, high computational cost, poor robustness, and possible ethical and legal issues. These challenges restrict the further development and application of GAN technology, and need to be solved through the joint efforts of researchers, enterprises, and governments.

Looking forward to the future, with the deepening of theoretical research, the optimization of technical methods, the expansion of application fields, and the improvement of ethical and legal standards, GAN technology will surely achieve greater breakthroughs. It will not only play a more important role in solving practical problems in various fields but also promote the development of artificial intelligence from "perceptual intelligence" to "generative intelligence", bringing more convenience and surprises to human society. And have reason to believe that GAN technology will become a key driving force for the fourth industrial revolution, creating a better future for mankind.

References

1. I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, Y. Bengio, Generative adversarial networks. arXiv preprint arXiv:1406.2661 (2014)
2. J. Y. Zhu, T. Park, P. Isola, A. A. Efros, Unpaired image-to-image translation using cycle-consistent adversarial networks. in Proceedings of the IEEE International Conference on Computer Vision, 2223–2232 (2017)
3. T. Karras, T. Aila, S. Laine, J. Lehtinen, Progressive growing of GANs for improved quality, stability, and variation. arXiv preprint arXiv:1710.10196 (2017)
4. Y. Huang, A. Gokaslan, V. Kuleshov, The GAN is dead; long live the GAN! A modern baseline GAN. arXiv preprint arXiv:2501.05441 (2023)
5. A. Radford, L. Metz, S. Chintala, Unsupervised representation learning with deep convolutional generative adversarial networks. arXiv preprint arXiv:1511.06434 (2015)
6. M. Arjovsky, S. Chintala, L. Bottou, Wasserstein generative adversarial networks. in International Conference on Machine Learning, 214–223 (2017)
7. T. Karras, S. Laine, T. Aila, A style-based generator architecture for generative adversarial networks. in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 4401–4410 (2019)
8. H. Zhang, I. Goodfellow, D. Metaxas, A. Odena, Self-attention generative adversarial networks. in International Conference on Machine Learning, 7354–7363 (2019)
9. A. Brock, J. Donahue, K. Simonyan, Large scale GAN training for high fidelity natural image synthesis. arXiv preprint arXiv:1809.11096 (2018)
10. Y. Huang, H. Shen, Y. Wang, L. Zhang, B. Li, Transformer-driven GAN for high-fidelity image generation with spatiotemporal joint perception. IEEE Trans. Pattern Anal. Mach. Intell. **47**, 1234–1248 (2025)