Proceedings of the 2nd International Conference on Software Engineering and Machine Learning
DOI: 10.54254/2755-2721/74/20240427

Enhancing the stability of Generative Adversarial Networks: A survey of progress and techniques

Yuwei Ni

Department of Computer Science and Technology, Xiamen University Malaysia, Sepang, Selangor, 43900, Malaysia

1971564643@qq.com

Abstract. This survey aims to comprehensively review and analyze the progress in enhancing the stability of generative adversarial networks (GANs). It systematically explores the evolution of GAN architectures, loss functions, and regularization techniques, focusing on their impact on training stability. In addition, this article briefly discusses the main challenges encountered during GAN training, such as mode collapse and vanishing/exploding gradients, and synthesizes various strategies and methods developed to alleviate these problems. Finally, the survey highlights the ongoing need for innovative solutions to improve the stability of GAN training and ensure its effective and robust application in various fields.

Keywords: Generative adversarial network (GAN), Training Stability, Loss Function.

1. Introduction

Generative adversarial networks (GANs) have become a research hotspot in deep learning and computer vision since they were proposed by Goodfellow et al. in 2014[1]. GANs can generate data similar to the real data distribution through the game process of a generator and a discriminator. Due to its powerful generation capabilities, GANs have been widely used in many fields, such as image generation, style transfer, and data enhancement. However, the training stability of GANs has been a major issue limiting their development. Many researchers have proposed different variations and techniques to solve this problem. This article aims to review the basic principles and development history of GANs, as well as various factors that affect the stability of GANs training and corresponding solutions. This study will introduce the basic architecture and principles of GANs, as well as their main variants and features. Then it will delve into the main factors that affect the stability of GANs training, including mode collapse, vanishing/exploding gradients, etc., and discuss the causes and effects of these problems. Finally, the survey will review and summarize the current main methods and technologies to improve the stability of GANs training, as well as their application cases.

2. Basic information about GANs

The Generative Adversarial Network (GAN) model consists of a generator (G) and a discriminator (D), which play games with each other during the model training process [1]. The role of the generator is similar to a counterfeiter in the market, while the discriminator is like a supervisor. What is different from reality is that the counterfeiter, that is, the generator, can and can only send the return signal through the discriminator to repeatedly test whether it has succeeded in counterfeiting. In this simulated market,

DOI: 10.54254/2755-2721/74/20240427

the generator continuously attempts to generate increasingly realistic fake data, while the discriminator strives to improve its ability to distinguish between real and fake data. The generator continuously adjusts its strategy based on the feedback from the discriminator, intending to generate data that can fool the discriminator.

However, such a training process is not always ideal. First, the volume and diversity of data is an essential factor. If the training data is not rich and diverse enough, the generator may fall into overfitting and generate data that lacks diversity. Secondly, the training process of GAN is relatively abstract, and it is difficult to manually evaluate performance or determine an explicit criterion for stopping training. The original GAN uses cross-entropy as the loss function. One drawback of this loss function is the problem that can lead to mode collapse, where the generator always generates the same or similar samples. In addition, the cross-entropy loss function may also cause the vanishing gradient problem, making it difficult for the generator to obtain effective gradient updates during the training process [1].

Conditional Generative Adversarial Networks (cGANs) were proposed by Mehdi Mirza and Simon Osindero in 2014[2]. cGANs allow the introduction of labels or other conditional information as additional input variables during the generation process. This design enables the model to generate samples with specific attributes based on specified conditions. This does not directly reduce the impact of the loss function on the model, but it helps guide the model to generate data that is more consistent with specific conditions. Since the generation process of cGANs relies on conditional variables, the selection and design of conditional variables becomes particularly important. Inappropriate condition variables may affect the training stability of the model and the quality of the generated samples [3]. Therefore, the training stability of cGANs relies heavily on the quality of the data and the correlation of the conditional variables with the data.

DCGAN (Deep Convolutional Generative Adversarial Networks) was proposed by Alec Radford, Luke Metz, and Soumith Chintala in 2015[4]. DCGAN has made a series of structural improvements based on the original GAN: it uses convolutional layers and deconvolutional layers in the generator and discriminator; it uses the Leaky ReLU activation function in the discriminator; and in the generator, The ReLU and Tanh activation functions are used. These improvements have significantly improved the model performance and training stability of DCGAN when processing image data. Regarding the loss function, DCGAN still uses the loss function form of the original GAN, which minimizes the confrontation loss between the generator and the discriminator [4]. However, due to the improvement of its network structure, DCGAN is relatively more stable during the training process, alleviating the problems of gradient disappearance and mode collapse. However, as a non-convex game process, it is still difficult to avoid falling into local optimality, resulting in low quality or lack of diversity in the generated samples.

CycleGAN was proposed in 2017 by Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros [5]. The model contains two generators and two discriminators. The two generators are responsible for the conversion from one domain to another, and the two discriminators are used to determine whether the conversion results are realistic. At the same time, in addition to using traditional adversarial losses, CycleGAN also adds cycle consistency losses. The adversarial loss ensures that the generated images are realistic in the target domain, while the cycle consistency loss ensures that the transformation is reversible and consistent. The combination of these two losses enables CycleGAN to achieve sample transformation between different domains without supervision. However, although CycleGAN dramatically reduces dependence on data due to cycle consistency loss and special model structure, its generation quality may not be as good as models using paired training data. Especially in some complex transformation tasks, CycleGAN may have problems with the generated results lacking logic or losing details.

WGAN (Wasserstein GAN) was proposed by Martin Arjovsky, Soumith Chintala, and Léon Bottou in 2017[6]. It introduces a new loss function based on Wasserstein distance (also known as Earth Mover's Distance), which measures the distance between two probability distributions. Unlike traditional GAN, the discriminator of WGAN uses a linear layer without an activation function as the output layer. This means that the discriminator is no longer just distinguishing between real and

DOI: 10.54254/2755-2721/74/20240427

generated samples, but trying to find a way to best fit the Wasserstein distance between the two distributions. Therefore, the training process of WGAN is relatively more stable, and its loss function value has a more precise physical meaning, which can be used as an effective indicator to evaluate model performance. In addition, WGAN clips the weights of the discriminator during the training process to satisfy the Lipschitz constraint. This design makes WGAN perform well in solving the mode collapse problem in traditional GAN, and the generated samples have better diversity. However, it also limits the representation ability of the model, which reduces the stability of training and the quality of generated samples [6].

3. Factors that affect stability

By browsing the development history of the GAN model, it can be observed that scholars' improvements to the model are largely based on improving the stability of the model training process. Next, this survey will list and discuss some common instability factors.

The first is the most basic data dependency. The most common problem is that the data is too small, resulting in model overfitting. The characteristic of GAN is that it can generate samples by itself. However, if the data is too small, the model will have a meager generalization ability, making the model meaningless in practical applications [7]. In addition, due to the nature of the generator to create "realistic" data, if the input data contains a lot of noise or outliers, the GAN model may constantly try to copy these meaningless features during the training process, making the final model confusing and inapplicable. Similarly, if the data type is too homogeneous and lacks diversity, the generator may only generate certain patterns in the data set and produce overly similar outputs, leading to pattern collapse. In certain types of GAN models, such as conditional GAN, the training of the model relies too much on data labels or other condition information. If these labels or condition information are inaccurate or inconsistent, the stability of model training will be significantly reduced [2].

Overfitting is also an influencing factor at the data level. Overfitting usually occurs when the complexity of the model is so high that it starts to learn the noise and unimportant details in the training data instead of capturing the true distribution of the data [8]. Specifically, the model performs extremely well on the training data because it can even remember random noise and outliers. However, model performance often drops significantly when applied to test new, unseen data. This is mainly because the model is too complex or the training time is too long, causing the model to become too sensitive to small changes in the training data and lack sufficient generalization ability when facing new data [9].

The resistance to minor changes or noise in the input data mentioned above has one word to describe it, and that is robustness. Robustness is an essential indicator of model stability, especially in training GANs. A GAN model with low robustness will have difficulty maintaining the stability of the model output when faced with small changes or noise in the input data and may cause considerable changes in the model output due to small fluctuations in the input. Improving robust habits not only helps to improve the generalization ability of the model so that it can perform well on unseen data but also helps to improve the stability of model training and reduce oscillations and instability during the training process [10].

Oscillation and non-convergence are usually mentioned together when discussing when GAN ends training. When training GANs, it is usually observing the changes in the loss function and the quality of the generated samples to judge whether the model has been trained sufficiently or whether training should be stopped. Issues of oscillation and non-convergence can make this judgment particularly difficult [11]. This problem usually occurs due to improper selection of model architecture or improper setting of hyperparameters (such as learning rate). The generator and discriminator in the GANs model are trained alternately [1], so if the balance between the two is broken, it may produce "one-sided" or alternately "one-sided" results. For example, suppose a generator is much more potent than another discriminator. In that case, it may cause the discriminator to be unable to distinguish true from false, causing the generator to continue to reduce the quality of its generation. Then the discriminator can always successfully distinguish true from false, and the cycle repeats.

DOI: 10.54254/2755-2721/74/20240427

Mode collapse is also one of the common problems when training GANs. It can be regarded as an alternative manifestation of the oscillation problem. Mode collapse occurs when the generator starts generating highly similar or almost identical samples, ignoring the diversity of the input data. In other words, the generator found a "shortcut" to deceive the discriminator and only generated a few samples that the discriminator considered to be "real" without capturing the diversity and richness of the real data distribution, falling into a local optimum. This makes it difficult for the model to reach a good balance point, making the training process difficult to converge [12].

Gradient disappearance and gradient explosion are extreme problems encountered during the training process that affect stability. In GAN, the goal of the discriminator is to distinguish real samples from generated samples, while the generator tries to generate samples that can fool the discriminator [13]. If the discriminator is trained too well, the gradient of the generator may disappear because the gradient of the discriminator becomes very small during backpropagation. This causes the generator's weights to update very slowly, making it difficult for the generator to improve the samples it generates, which in turn causes model training to stagnate and the generator to be unable to generate convincing fake samples. On the contrary, during the training process of GAN, if the balance between the generator and the discriminator is broken, it may cause a gradient explosion. For example, if the generator generates poor-quality samples, the discriminator may easily differentiate between real and generated samples, causing the gradient to become very large during backpropagation. This will cause the model weights to be updated too drastically, making the model training unstable, which will, in turn, make the training process of the generator and discriminator very difficult, and even make the model unable to converge [14].

4. Techniques and applications to improve stability

Certainly, whether in the past or now, scholars are constantly seeking to explore new technologies in order to increase the stability of the GANs training process and obtain better-performing models.

Network architecture is one of the first technologies to be thought of and applied in innovation, and the proposal of DCGAN is based on this. This is a landmark model that significantly improves the performance of GAN on image generation tasks through a series of innovative network architecture designs. DCGAN uses convolutional layers and deconvolutional layers, with special emphasis on the application of convolutional neural networks (CNN) in image generation [15]. This enables DCGAN to take advantage of CNN to better capture and generate the hierarchical structure and spatial information of images. DCGAN also introduces Batch Normalization (BN), a technology that can alleviate the vanishing gradient problem and enable deeper training of the model. By standardizing the data of each mini-batch, BN can maintain a relatively stable distribution of the input of each layer in the network, which helps to improve the training stability and convergence speed of the model [4].

DCGAN also simplifies the model structure and reduces the number of parameters of the model by removing the fully connected layers in the generator and discriminator. This simplification not only helps mitigate the risk of overfitting but also makes the model easier to train and optimize. In the choice of activation function, DCGAN uses Leaky ReLU in the discriminator and ReLU and Tanh in the generator [4]. The Leaky ReLU function allows small negative values to pass, which means that even if the input value is negative, Leaky ReLU will have a non-zero output. Therefore, even when the input value is negative, the gradient does not disappear completely, thus maintaining the flow of information. The ReLU function in the generator is used as the activation function of the hidden layer. The ReLU function directly outputs the value when the input value is positive, and outputs zero when the input value is negative. This design also helps alleviate the vanishing gradient problem. The Tanh function is used in the generator output layer to compress output values to between -1 and 1, which helps the model's output match the common range of image data. These network architecture innovations enable DCGAN to exhibit higher stability during training and generate higher-quality samples, laying the foundation for subsequent GAN research and development [4].

The improvement of the loss function also plays a crucial role in the training stability of the GAN model. In particular, Wasserstein GAN (WGAN) brings significant improvements to the training

DOI: 10.54254/2755-2721/74/20240427

stability of the GAN model by introducing Wasserstein distance as the loss function. Wasserstein distance, as a measure of the difference between two probability distributions, has smoother gradient characteristics in GAN training than the cross-entropy loss function. In traditional GANs, the cross-entropy loss function is commonly used. When the samples generated by the generator are significantly different from the real samples, the discriminator can easily distinguish between true and false samples [1]. As a result, during backpropagation, the generator receives a weak gradient signal. This may lead to the vanishing gradient problem, making the training of the generator challenging. The Wasserstein distance can provide meaningful gradients throughout the training process. Even when the generator performance is poor, Wasserstein distance can provide effective gradient information to guide the generator to update and learn more effectively. This characteristic makes WGAN more stable and easier to converge during the training process [6].

In addition, in WGAN, the discriminator is no longer a binary classifier but becomes a critic function [6]. In traditional GAN, the goal of the discriminator is to distinguish whether the input sample is a real sample or a generated sample and output a probability value. In WGAN, the task of the discriminator (critic) is to fit the Wasserstein distance between the real data distribution and the generated data distribution as closely as possible. This design change reduces oscillations and instability during model training because the gradient information provided by the critic is more direct and stable. In addition, since critic is no longer limited by the cross-entropy loss function, it can provide outputs that vary within a wider range, which also helps provide richer and more effective gradient information. Moreover, in WGAN, critic training is constrained, and it is necessary to ensure that the critic function is 1-Lipschitz continuous, which is usually achieved through weight clipping or gradient penalty, which helps ensure effective and accurate estimation of Wasserstein distance. WGAN also effectively alleviates the mode collapse problem. Since the Wasserstein distance provides richer gradient information, the generator can better capture the diversity and richness of the data distribution and avoid generating too similar samples [6]. These improvements enable WGAN to exhibit better stability and convergence during training.

Spectral Normalization is a regularization technique used for stable training in Generative Adversarial Networks (GANs), proposed by Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida in 2018 [16]. This technology is mainly applied to the weight matrix of the discriminator. Limiting the spectral norm of the weight matrix (that is, the maximum singular value of the weight matrix), prevents the gradient explosion problem and ensures the stability of the training process. The basic principle of spectral normalization is to modify the weight matrix in the network so that its spectral norm does not exceed 1. For each weight matrix, spectral normalization divides its maximum singular value, so that the processed weight matrix has better stability properties. This processing method helps balance the competitive relationship between the generator and the discriminator, prevents the discriminator from becoming too powerful prematurely, and allows the generator to obtain useful gradient information in the early stages of training. It therefore helps avoid mode collapse problems, allowing the generator to generate more diverse samples.

5. Conclusion

It is foreseeable that the stability of the GANs training process will still be an urgent problem to be solved in the future. However, it can be seen from the discussion that the current technology has become more and more refined in improving stability. I call on more scholars to focus on new data processing functions, especially the functions used by the discriminator, to obtain a more stable GAN model. From the earliest changes to the model architecture to changing the loss function to proposing new training methods, this is a progressive process, and this article explores this process.

References

[1] Goodfellow I, Pouget-Abadie J, Mirza M, Xu B, Warde-Farley D, Ozair S, Courville A and Bengio Y 2014 "Generative adversarial nets" in Proceedings of *the 27th International Conference on Neural Information Processing Systems* - Volume 2 (Montreal, Canada) pp 2672–76.

DOI: 10.54254/2755-2721/74/20240427

[2] Mirza M and Osindero S 2014 "Conditional generative adversarial nets" arXiv preprint arXiv:1411.1784.
[3] Isola P, Zhu JY, Zhou T and Efros AA 2017 "Image-to-image translation with conditional adversarial networks" in Proceedings of *the IEEE Conference on Computer Vision and Pattern Recognition* (Honolulu, HI) pp 1125–34.
[4] Radford A, Metz L and Chintala S 2015 "Unsupervised representation learning with deep convolutional generative adversarial networks" arXiv preprint arXiv:1511.06434.
[5] Zhu JY, Park T, Isola P and Efros AA 2017 "Unpaired image-to-image translation using cycle-consistent adversarial networks" in Proceedings of *the IEEE International Conference on Computer Vision* pp 2223–32.
[6] Arjovsky M, Chintala S and Bottou L 2017 "Wasserstein generative adversarial networks" in Proceedings of *the 34th International Conference on Machine Learning* - Volume 70 (ICML'17) (JMLR.org) pp 214–23.
[7] Wang Z, Zheng H, He P, Chen W and Zhou M 2022 "Diffusion-GAN: Training GANs with diffusion" arXiv preprint arXiv:2206.02262.
[8] Yazici Y, Foo CS, Winkler S, Yap KH and Chandrasekhar V 2020 "Empirical analysis of overfitting and mode drop in GAN training" in *2020 IEEE International Conference on Image Processing (ICIP)* (October 2020) pp 1651–55.
[9] Gulrajani I, Raffel C and Metz L 2020 "Towards GAN benchmarks which require generalization" arXiv preprint arXiv:2001.03653.
[10] Chrysos GG, Kossaifi J and Zafeiriou S 2020 "RocGAN: Robust conditional GAN" *International Journal of Computer Vision* - Volume 128 pp 2665–2683.
[11] Thanh-Tung H and Tran T 2020 "Catastrophic forgetting and mode collapse in GANs" in *2020 International Joint Conference on Neural Networks (IJCNN)* (July 2020) pp 1–10.
[12] Liu K, Tang W, Zhou F and Qiu G 2019 "Spectral regularization for combating mode collapse in GANs" in Proceedings of *the IEEE/CVF International Conference on Computer Vision* pp 6382–6390.
[13] Creswell A, White T, Dumoulin V, Arulkumaran K, Sengupta B and Bharath AA 2018 "Generative adversarial networks: An overview" *IEEE Signal Processing Magazine* - Volume 35 no 1 pp 53–65.
[14] Tao S and Wang J 2020 "Alleviation of gradient exploding in GANs: Fake can be real" in Proceedings of *the IEEE/CVF Conference on Computer Vision and Pattern Recognition* pp 1191–1200.
[15] Tsimenidis S 2020 "Limitations of deep neural networks: A discussion of G. Marcus' critical appraisal of deep learning" arXiv preprint arXiv:2012.15754.
[16] Miyato T, Kataoka T, Koyama M and Yoshida Y 2018 "Spectral normalization for generative adversarial networks" arXiv preprint arXiv:1802.05957.