Pretraining technology of adversarial generative networks based data selection

Ziyi Zhao

Department Mathematics, Ludwig-Maximilian University, Munich, 80809, Germany

Ziy.Zhao@campus.lmu.de

Abstract. The effectiveness of adversarial generative networks (GANs) has been shown in many works. However, the convergence of the GANs should be granted mathematically with proper hyperparameters and some network structures. Considering the difficulties of constructing a large and clean dataset, GANs are required in many cases to extract enough information from a relatively small dataset. This could result in bad coverage performance of GANs, and one way to tackle this problem is pretraining. Since this method relies on importing external information from the pretraining dataset, this paper does a necessary discussion on the effect that the pretraining dataset has on the model and makes a simple assumption on the standard of choosing a pretraining dataset. Specifically, this paper focuses on the case where the target is to generate animation-style characters, and it has been proven that when providing a pretraining dataset with higher diversity, the performance of the GAN is better. The result of this paper should reveal the possibility that using a specially chosen pretraining dataset can improve the result of GAN.

Keywords: Machine Learning, Neural Network, Pretraining, GAN, Computer Vision.

1. Introduction

Since introduced, generative adversarial neural networks have been the most effective machine learning paradigm for generative tasks [1]. Given a set of samples that represents the distribution of the generation target in the sample space (e.g., in $\mathbb{R}^{3\times64\times64}$ for 64*64 colored picture generation), the GANs can find a map from the distribution of a latent noise to a distribution in the sample space (the generated distribution), which minimizes the difference between the generated distribution and the sample distribution. Based on the idea of a "zero-sum" game between two neural networks, the generator and the discriminator, this network structure pointed out a brand-new way of the generative model. Soon after, the good nature of GANs in generative works was revealed, and numerous studies are using GANs and their variants for generative tasks [2].

Apart from improvements made on the network, some tricks have also been conducted to boost the training process of GANs, among which the pretraining trick is what these paper focuses on. Pretraining may be widely used in many machine learning studies for better performance, e.g., accelerating convergence [3, 4]. However, the goal of most of these studies is classification rather than generation, and the application of pretraining on GAN architectures seems to be less considered. Only a few studies have been done in this field. For example, one has shown the effectiveness of pretraining, especially when the dataset is small [5]. More recently, another research proved that both training of generator and discriminator could benefit from pretraining [6]. Nevertheless, more detailed information remains unknown, e.g., the selection for an appropriate pretraining checkpoint and the standard for choosing a pretraining dataset.

Unlike the mentioned works, this paper investigates the relationship between the pretraining dataset and the result's quality. By building pretraining datasets from different base datasets, it is possible to control the pretraining dataset's diversity roughly. Using this technique, two experiments using two different GAN architectures are conducted. Each goes in 2 significant steps: First, the GAN is trained with a pretraining dataset several times, then switched to the training set and continuously trained for a long time. After that, save some samples for comparison. Second, initialize the parameters, train the GAN with an altered pretraining dataset, and repeat the same operations as step 1. In the end, the outcomes of these two steps are compared, and the result shows that when the pretraining dataset is altered to have better diversity, the performance of GAN is better.

The remainder of this paper is written in the following structure: Section 2 contains information on generative adversarial networks, including some problems with GANs. Section 3 gives a brief introduction to 3 different GAN variants, which are used in this study. Section 4 demonstrates the approach used to compare the "power" of different features and the result with discussions. Section 5 gives a conclusion of this paper.

2. The generative adversarial networks

2.1. The Structure of GANs

The GANs consists of 2 parts, the generator G and the discriminator D. G is the generative part of the model. It generates fake samples in high-dimensional space with given low-dimensional latent noise vectors (usually uniform or Gaussian). In this way, the generator maps the distribution of the latent noise to distribution in a high-dimensional space. By contrast, the goal of D is to distinguish the generated samples from the real ones. The judgment of the discriminator usually comes out in the form of real numbers. It is used to construct the loss function. During the training process, these two networks are trained alternately. The structure of GANs is given in Figure 1.

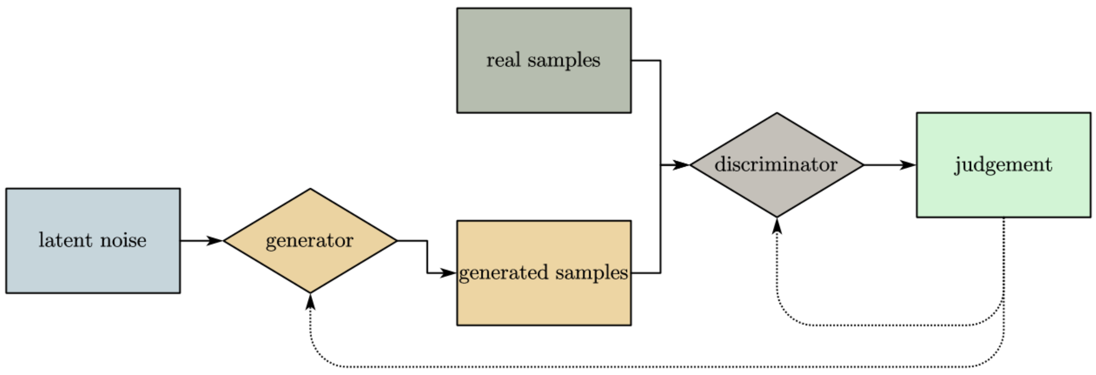

Figure 1. The structure of GAN.

In the original GAN, the discriminator gives the predicted probability that the sample belongs to the real dataset. Denote by $\mathcal{D}_{\text{data}}$ the distribution of the dataset and $\mathcal{D}_{\text{z}}$ the distribution of the generated samples. If the loss function is chosen to be the binary entropy loss, then G and D can be seen as two player of the minimax game with value function $V(D, G)$:

$$\min_G \max_D V(D,G) = \mathbb{E}_{x\sim\mathcal{D}_{\text{data}}}[\log D(x)] + \mathbb{E}_{z\sim\mathcal{D}_z}\left[\log\left(1 - D\big(G(Z)\big)\right)\right] \tag{1}$$

It is also proved that if D is optimal with respect to a fixed G, then the goal to minimize the value function can be viewed as to minimize the Jensen–Shannon divergence between the $\mathcal{D}_{\text{z}}$ and $\mathcal{D}_{\text{data}}$.

2.2. Some Problems of GANs

Although the GANs are suitable for generative tasks, they (especially the original GAN) still suffer serious problems. Some of them are listed as follows:

2.2.1. Vanishing Gradients. One problem is the absence of a gradient of the generator. The GAN relies on the gradient descent method to update the discriminator and the generator and let these two players of the minimax game reach the Nash equilibrium. However, as mentioned in a related study, in some cases where the discriminator can do the classification perfectly, Jensen–Shannon divergence cannot provide valid gradient information [7]. This may happen when the dataset and the generated distribution are disjoint, or their support is a low-dimensional manifold in the sample space. In these 2 cases, the measure for their support is zero. Moreover, the dataset and the generated distribution are low-dimensional manifolds, which means this problem will happen if the discriminator is “too good” [8]. Therefore, one must consider carefully how often the discriminator should be trained.

2.2.2. Model collapse. Another problem of GANs is called model collapse. Sometimes the generator only produces the same output (or a small set of outputs), which is plausible for the discriminator. Theoretically, the strategy of the discriminator, in this case, is to reject all samples in this set to force the generator to learn other patterns of outputs. Nevertheless, the discriminator may not be optimized enough or be stuck at some local minima. Hence, the discriminator cannot always perform the ideal strategy, and the generator continues to produce the same outputs to fool the discriminator.

2.2.3. Convergence Status is Hard to Monitor. For the supervised optimization problems, the loss can be viewed in real-time. By contrast, the GAN, as an unsupervised learning architecture, does not have such trait. Although in WGAN models, the output of the discriminator can be seen as an estimator of the EM distance and therefore be used for monitoring convergence, this metric still correlates with the discriminator’s structure. It is meaningless when comparing different models [7].

3. The gAN architectures

Despite the problems mentioned above, the generative adversarial network has been a well generative model, and many of its variants have been introduced to tackle these problems. Two variants used in the experiment part of this paper are listed here.

3.1. DCGAN

Deep convolutional GAN (DCGAN) is a classical variant of GANs. This GAN architecture changes the original GAN's MLP layers to convolutional ones. It adds batch normalization layers in between [9]. Besides, the ReLu function is used after every convolutional layer in the generator. LeakyReLu is used in the discriminator to prevent parse parameters. Compared to MLP layers in the original GAN, convolutional layers are more capable of extracting patterns or features from the input signal, especially in computer vision. Moreover, the batch normalization layers make the model more stable. Therefore, DCGAN is a successful GAN architecture and has been used in many other studies, mainly in computer vision [10-12].

3.2. WGAN-GP

Wasserstein GAN (WGAN) also makes modifications to the loss function. It solves the vanishing gradient problem and makes the training more stable. WGAN uses the Wasserstein distance instead of the JS-divergence to measure the loss between the generated and data distribution [7]. This could solve the vanishing gradient problem, as Wasserstein distance can always provide reliable gradients for training. In addition, this model also solves the model collapse problem to some extent, as the WGAN model allows the discriminator (or the “critic”) to be trained to perfection. In other words, a WGAN model solves the following minimization problem:

$$\underset{D}{min} V_{\mathrm{LSGAN}}(D) = E_{z\sim p_{\mathrm{z}}(z)}\left[D\big(G(z)\big)\right] - E_{x\sim p_{\mathrm{data}}(x)}[D(x)] \tag{2}$$

$$\min_{G} V_{\mathrm{LSGAN}}(G) = -E_{z\sim p_{z}(z)}\left[D\left(G(z)\right)\right] \tag{3}$$

The only problem with Wasserstein distance is that it is a much weaker condition compared to JS-divergence. Hence, the discriminator has to satisfy the precondition of Lipschitz continuity to guarantee the convergence of the network. In WGAN-GP, the gradient penalty method is introduced to maintain the Lipschitz property by penalizing significant gradients of the discriminator.

4. Experiments

This paper train GAN models with images, as graphic generation is one of the most popular applications of GANs. The experiments are implemented in the following steps: In the pretraining phase, the GAN model is trained on a dataset for some time, and the pre-trained model is saved and passed to the training phase. In the training phase, the model has trained on the target dataset for a longer time to learn to generate target pictures. The experiment is repeated with different GAN models (those mentioned in the previous section) and pretraining datasets, so different combinations are tested, returning a series of results.

4.1. Datasets

The experiment in this paper uses relatively small datasets. One reason for this is that the pretraining trick is usually used to help train on a small dataset, so the experiment is made to simulate such situation. Besides, smaller datasets are more controllable, i.e., it is easy to make significant changes in a small dataset by adding a limited number of pictures, which is essential for the experiment method. For the pretraining datasets, one choice is around 30000 bedroom pictures randomly chosen from the LSUN-Bedroom dataset [13]. Since the original LSUN dataset is too big and pretraining only lasts for a limited time, using this truncated dataset should be more cost-effective. Another choice is a mixed dataset containing pictures from LSUN-Bedroom and pictures of cats and human faces. In the remaining part of this paper, the former is referred to as” the bedroom dataset,” and the latter is referred to as ”the mixed dataset .”Some samples from these datasets are displayed in Figure 2.

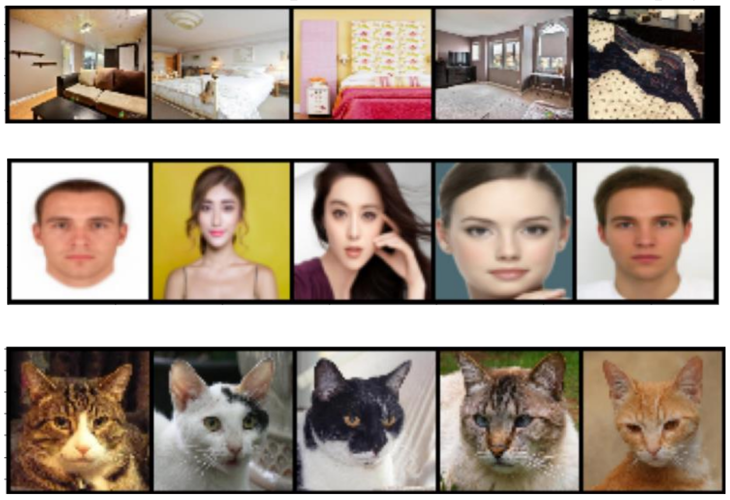

Figure 2. Some Pictures from the datasets used in pretraining.

A dataset containing Japanese animation-style girl faces was used for the training dataset. This choice is because this kind of picture is less studied, so it is hard to find an existing extensive dataset for training. Therefore, the pretraining trick should be meaningful in this situation. Another reason is that animation style faces differ strongly from the pretraining datasets. Hence, the only difference between these two pretraining choices is the diversity. Again, some samples of the training dataset are displayed in Figure 3.

Figure 3. Some pictures from the dataset used in training.

4.2. Results and Discussions
They are starting with the experiment on the DCGAN model. Generally, as shown in Figure 2, the network produces better pictures when the pretraining is done on the mixed dataset. Specifically, when pre-trained with the mixed dataset, the GAN model learns the shape of the face better, and a wider variety of colors can also be observed in the generated pictures (see Figure 4). This may result from features of the bedroom dataset: Most contour lines in bedroom pictures are straight, and a lot of them has some part that is plain white, e.g., the bed or the wall. These two features are different from animation style faces, affecting the network's performance. By contrast, adding the cat and human face pictures in pretraining disallows the model to fit the bedroom dataset. In other words, this change prevents the model from being trained too well in the pretraining phase. A further experiment shows that even with a doubled pretraining time, the mixed dataset still performs well in pretraining (see Figure 5). Hence, using a "dirty" or more diverse dataset for pretraining may help improve the performance of the pretraining trick and make the pretraining trick more stable towards the change of pretraining time.

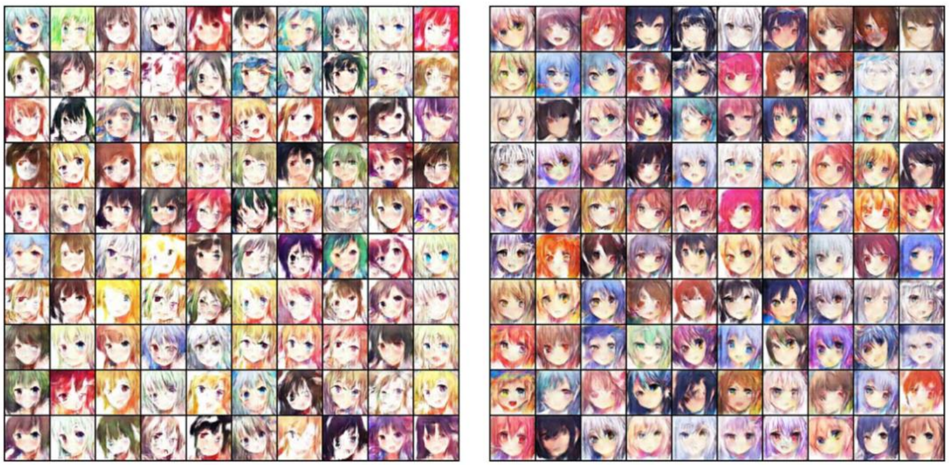
Figure 4. Generated pictures by DCGAN on an animation style girl face set.

The left is from the experiment pre-trained with the bedroom dataset, and the right one is from the experiment using a mixed dataset. The left one contains more white parts, and its colors are also not as good as the right one: Colors in the left picture are too dark or too bright. Besides color, the experiment using the bedroom dataset also performs poorly in drawing the shape of the face: In the left picture, several samples failed to create a character with a round face, while this failure is rare for the right one. Moreover, using the bedroom dataset for pretraining may lead to model collapse, as the blonde hair samples in the left picture are similar.

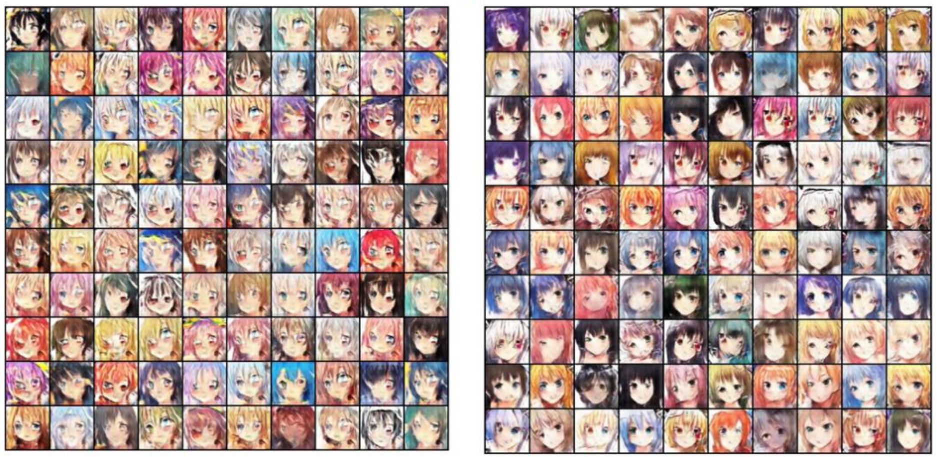
Figure 5. Generated pictures using a doubled pretraining time.

Obviously, using a doubled training time has a negative effect on the performance, and when using the mixed dataset, such decrease is less remarkable.

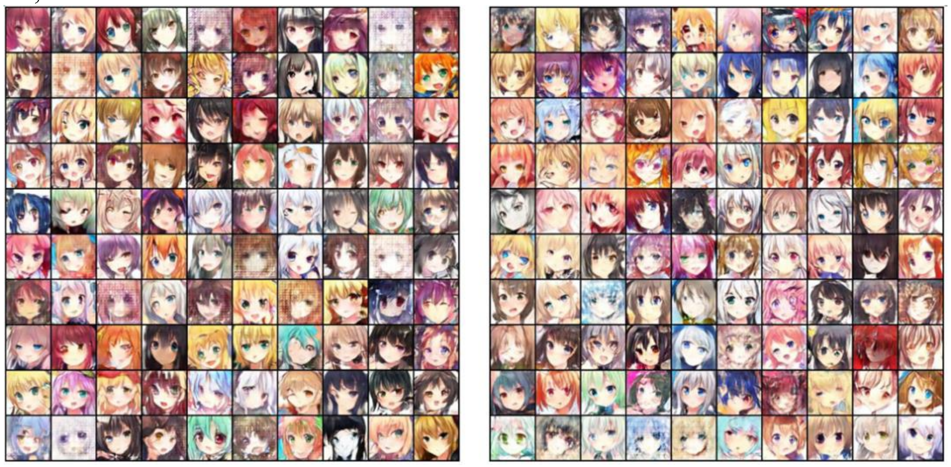
Figure 6. Generated picture of WGAN-GP model.

The left is from the experiment pretrained with the bedroom dataset, and the right one is from the experiment using mixed dataset. Like Figure 2., the mixed dataset performs also better here, especially in matching the round shape of character faces and eyes.

One may note that the quality of all the generated pictures is not so good. This is because this experiment does not focus on the quality but on comparing different datasets. Thus, the hyperparameters and the DCGAN structure are not well-tuned, and the GAN model was not trained for a long time. However, as displayed above, this experiment still provides enough information for comparison.

Similar results can also be observed in experiments involving the WGAN-GP or BEGAN models. Although the performance of WGAN-GP is generally worse than DCGAN, it is still apparent that using the mixed dataset in pretraining improves the quality of the generated pictures, especially in catching the shape of the face (see Figure 3). This result supports the above discussion and indicates that the conclusion may be widely applicable to various GAN models (see Figure 6).

5. Conclusion

To summarize, this paper takes results from a series of simple experiments to investigate the performance of the pretraining trick in generative neural networks. It is observed that more diverse datasets are better for pretraining. A possible explanation for this phenomenon is that more diverse datasets are harder to learn, so using this kind of dataset prevents the well-fitting of the pre-trained model. Some further experiments show that diverse datasets can make the pretraining process less sensitive to pretraining time. As there is still no tool to determine a suitable pretraining time, this paper attempts to enhance the robustness of the pretraining trick against the selection of pretraining time by using a more diverse dataset. Moreover, the results indicate that when there is already some a priori expectation for the generation target, adding data matching these expectations to pretraining might also improve the quality of training. Specifically, pretraining with human faces and cat images, in this case, helps train animation character faces. Since GAN models and the datasets used in this paper tend to be simple, there is still much room for further studies.

References

[1] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, “Generative adversarial nets,” Advances in neural information processing systems, vol. 27, 2014.

[2] M. Lucic, K. Kurach, M. Michalski, S. Gelly, and O. Bousquet, “Are gans created equal? a large-scale study,” 2017.

[3] D. Marmanis, M. Datcu, T. Esch, and U. Stilla, “Deep learning earth observation classification using imagenet pretrained networks,” IEEE Geoscience and Remote Sensing Letters, vol. 13, no. 1, pp. 105–109, 2015.

[4] B. Kieffer, M. Babaie, S. Kalra, and H. R. Tizhoosh, “Convolutional neural networks for histopathology image classification: Training vs. using pre-trained networks,” in 2017 Seventh International Conference on Image Processing Theory, Tools and Applications (IPTA), pp. 1–6, IEEE, 2017.

[5] M. Zhao, Y. Cong, and L. Carin, “On leveraging pretrained GANs for generation with limited data,” in Proceedings of the 37th International Conference on Machine Learning (H. D. III and A. Singh, eds.), vol. 119 of Proceedings of Machine Learning Research, pp. 11340–11351, PMLR, 13–18 Jul 2020.

[6] T. Grigoryev, A. Voynov, and A. Babenko, “When, why, and which pretrained gans are useful?,” 2022.

[7] M. Arjovsky, S. Chintala, and L. Bottou, “Wasserstein gan,” 2017.

[8] M. Arjovsky and L. Bottou, “Towards principled methods for training generative adversarial networks,” 2017.

[9] A. Radford, L. Metz, and S. Chintala, “Unsupervised representation learning with deep convolutional generative adversarial networks,” 2015.

[10] Q. Wu, Y. Chen, and J. Meng, “Dcgan-based data augmentation for tomato leaf disease identification,” IEEE Access, vol. 8, pp. 98716–98728, 2020.

[11] Q. Li, H. Qu, Z. Liu, N. Zhou, W. Sun, S. Sigg, and J. Li, “Af-dcgan: Amplitude feature deep convolutional gan for fingerprint construction in indoor localization systems,” IEEE Transactions on Emerging Topics in Computational Intelligence, vol. 5, no. 3, pp. 468–480, 2019.

[12] D. Berthelot, T. Schumm, and L. Metz, “Began: Boundary equilibrium generative adversarial networks,” 2017.

[13] F. Yu, Y. Zhang, S. Song, A. Seff, and J. Xiao, “Lsun: Construction of a largescale image dataset using deep learning with humans in the loop,” arXiv preprint arXiv:1506.03365, 2015.