Handwriting Digital Image Generation based on GAN: A Comparative Study of Basic GAN and CGAN Models

Hongzhi Zeng

Guangdong University of Technology, Guangzhou City, Guangdong Province, China

Abstracts. The vast application of artificial intelligence in numerous fields—image generation being one of them—has been made possible by the quick development of deep learning. Generative Adversarial Networks (GAN) can generate high-quality images through an adversarial training mechanism. The use and performance of GAN and its conditional variation, CGAN, in the field of handwritten digital image generation, are thoroughly examined in this research. The basic GAN and CGAN models, based on the PyTorch deep learning framework and the Modified National Institute of Standards and Technology (MNIST) dataset, are applied to generate handwritten digital images respectively. To assess and compare the variations between the two models concerning the fineness of image generation, the loss changes, and other relevant factors, the generation outcomes and the loss changes that occur during the training phase are documented. The experimental results demonstrate that, compared with the basic GAN, CGAN exhibits notable advantages in terms of image quality stability, the avoidance of model collapse, and the control of image categories. Furthermore, an investigation of other cutting-edge generating models indicates that there is still room for optimization in the CGAN network structure to improve its performance for increasingly intricate generative tasks.

1. Introduction

One particular use for image generation, which is crucial to computer vision and artificial intelligence, is the generation of handwritten digits. The process of generation is achieved by training a generative model that converts random noise vectors into realistic handwritten digit images. Handwritten digit generation has shown its value in several application scenarios, and this technique has been extensively applied in the domains of data expansion and enhancement, model training and evaluation, system recognition, and other fields, and provides important support for scientific research and practical applications [1].

Recent years have witnessed a notable advancement in the area of image generation due to the quick growth of deep learning. In contrast to conventional image-generating techniques, deep learning can capture the complex patterns and details of images, resulting in realistic and diverse images. In particular, Generative Adversarial Networks (GAN) can generate

Corresponding author: 3122010006@mail2.gdut.edu.cn

higher-quality handwritten digital images through an adversarial training mechanism that does not rely on complex feature design or additional label information [2].

As early as October 2014, Ian J. Goodfellow et al. proposed GAN [3]. GANs consist of generators that attempt to generate realistic images and discriminators that attempt to distinguish the authenticity of those images. This adversarial structure of GANs makes them superior in generating images. Based on this advantage of GAN, many scholars have applied it to handwritten digital image generation. Chen et al. proposed Information Maximizing Generative Adversarial Networks (InfoGAN) to further enhance the control and interpretation of generated images by maximizing the interpretable information in the generated images so that the generated handwritten digits can be controlled according to specific latent variables [4]. Odena et al. proposed that Semi-Supervised Generative Adversarial Networks (SGAN), which receives training from both a huge volume of unlabeled data and a small amount of annotated data, can generate higher-quality handwritten numbers and improve the classification accuracy of 0.802 for 25 small samples, which is better than the accuracy of convolutional neural network (CNN) of 0.750 [5]. Song et al. used GAN to generate handwritten digital images, demonstrating the excellent accuracy and robustness of GAN in this regard, and pointed out that the accuracy of image recognition can be significantly improved by forming new adversarial datasets and adjusting network parameters [6].

Despite its good performance in handwritten digit generation, GAN suffers from problems such as easy pattern collapse during training, inconsistent quality of generated images, and difficulty in controlling the categories of generated images [2, 7]. To control the categories of the generated images and somewhat increase their quality, Mirza et al. devised a Conditional Generative Adversarial Network (CGAN), which introduces conditional information into the generator and discriminator [8]. A great deal of study has been done on using GANs to generate images, but most of them are based on the most common basic GAN models. CGAN has shown potential in some areas, but its effectiveness in generating handwritten digital images has not been fully investigated [9].

In this study, based on the PyTorch deep learning framework and the same dataset MNIST, basic GAN is used to generate handwritten digital images, and the loss changes and generated results during the training process are recorded. Subsequently, CGAN was introduced to compare the performance of the two in terms of loss change and generated image quality, and the advantages of CGAN in generating high-quality handwritten digital images were discussed and verified, especially in the ability to effectively avoid model collapse, control the generated image category and ensure the stability of generated quality, which provided a reference for subsequent research. The results of this study are not only of great significance for the generation of handwritten digital images but can also be applied to other scenarios that require the generation of controllable images, such as image enhancement and style transfer [10].

2. Methodology

2.1 Datasets

The Modified National Institute of Standards and Technology (MNIST) database is one of the most classic benchmark datasets in the fields of deep learning and machine learning and is frequently used for testing and training handwritten digit generation [11]. The database was created by modifying samples from the National Institute of Standards and Technology (NIST) original database and contains 10 different categories representing handwritten numbers 0 to 9, which consists of 10,000 test samples and 60,000 training samples, each with

a 28x28 pixel grayscale image. NIST's training dataset provides half of the training set and half of the test set, while NIST's test dataset provides the remaining half of the training set and half of the test set.

Based on the PyTorch deep learning framework, this study loads and uses the MNIST dataset through the torchvision library.

2.2 Basic GAN model

The two components of a GAN are the generator and discriminator, which make up this deep learning model. Fig. 1 displays the GAN network structure flow.

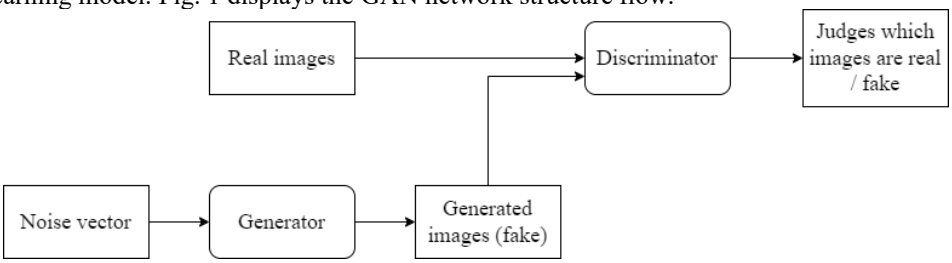


Fig. 1. Flowchart of the network structure of GAN (Photo/Picture credit: Original).

The primary function of the generator is to sample from a noise vector and generate an image through a series of deconvolution or upsampling operations. The objective is to produce synthetic data that is analogous to the distribution of actual data. The generator model typically comprises multiple layers, with each layer attempting to progressively approximate the true distribution of the data. By adjusting the loss function so that it becomes difficult for the discriminator to discriminate between the generated fake samples and the actual ones, the quality of the generated data can be improved.

Classifying the incoming data into generated samples and real samples is the discriminator's task. The process typically involves a sequence of convolutional layers to identify characteristics in the input data and then categorize them using a fully connected layer. The discriminator's objective is to maximize the likelihood that actual samples and generated samples can be accurately distinguished from one another. This drives the generator to provide more genuine data in an attempt to "trick" the discriminator. As training progresses, the discriminator is optimized to more effectively identify fake data, which in turn motivates the generator to provide samples that are more like the actual data.

Through a min-max game, in which the discriminator seeks to maximize the objective function and the generator seeks to minimize it, the entire GAN model is trained. The GAN has the following loss function:

$$\min_G \max_D V(D,G) = E_{x\sim p_{data}(x)}[\log D(x)] + E_{z\sim p_z(z)}\left[\log\left(1 - D\left(G(z)\right)\right)\right] \quad (1)$$

where x is a sample from the real data distribution p_{data} and z is random noise sampled from the noise distribution $p_z(z)$. The generator network is represented by $G(z)$ and the discriminator network by $D(x)$. The discriminator $D(x)$ tries to maximize its prediction accuracy for the real data x, i.e. $\log D(x)$, and minimize its false determination for the fake data $G(z)$, i.e. $\log\left(1 - D\left(G(z)\right)\right)$. With the ultimate goal of maximizing $D\left(G(z)\right)$, the generator $G(z)$ then makes an effort to generate false data such that the discriminator is unable to distinguish it from the genuine data [3, 6].

2.3 CGAN model

In the basic GAN, the generated results cannot be controlled due to the lack of explicit attribute identification, and the drawback becomes more obvious, especially when the amount of data is particularly large or the pixels of the images to be generated are complex. Therefore, CGAN is introduced. The network structure of CGAN is shown in Fig. 2.

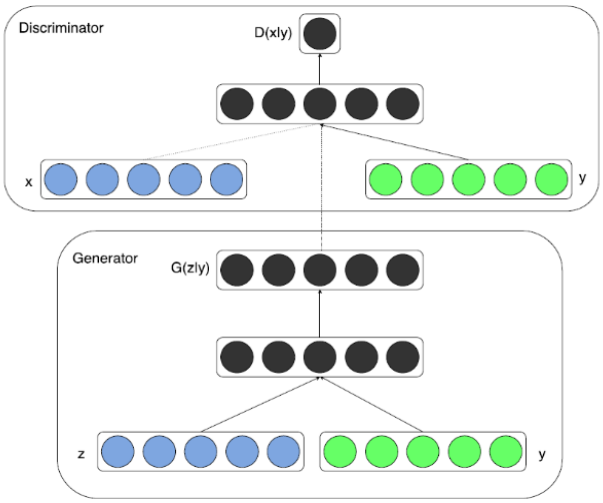


Fig. 2. Schematic diagram of the network structure of CGAN [8]

CGAN differs from GAN in the input of data. CGAN needs to add additional conditional information y to the inputs of generator G and discriminator D to make unsupervised learning become supervised learning. During training, CGAN typically needs to optimize the generator once and the discriminator multiple times to ensure that high-quality data is generated. Fig. 3 shows the network structure flow of CGAN.

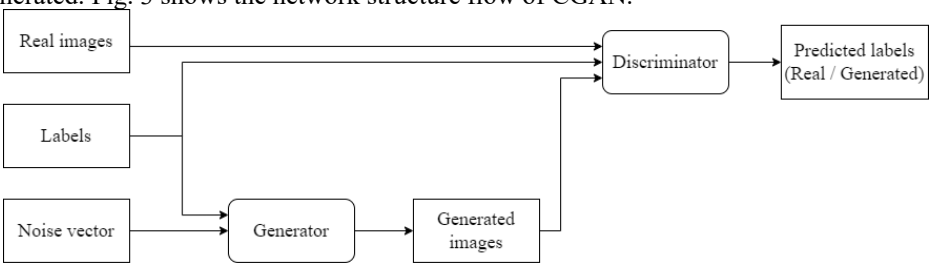


Fig. 3. Flowchart of the network structure of CGAN (Photo/Picture credit: Original).

In the training phase, the conditional variable y and the random noise z are inputted into the generator G together. The resultant generated data $G(z|y)$ as closely as feasible obeys the real data distribution p_{data}. Discriminator D receives the probability $D(x|y)$ of the data sample x for the real data under the given condition y and the data $G(z|y)$ generated by generator G, and finally outputs a scalar that estimates if the data used for input originates from the real data. The objective function is:

$$\min_G \max_D V(D,G) = E_{x\sim p_{data}(x)}[\log D(x|y)] + E_{z\sim p_z(z)}\left[\log\left(1 - D\left(G(z|y)\right)\right)\right] \quad (2)$$

Generator G seeks to maximize the likelihood that discriminator D will make a mistake in evaluating the input data, while discriminator D seeks to accomplish binary distinction

of data sources. After continuous confrontation and optimization, when discriminator D cannot correctly determine the authenticity of data, this indicates that generator G has become proficient in real data distribution, and the two have reached a state of equilibrium [8, 9].

2.4 Experimental design

In this study, the basic GAN model is parameterized and Table 1 displays the specific parameters.

Table 1. GAN model parameters

Parameter	Value
Number of training rounds (n_epochs)	200
Batch size (batch_size)	64
Learning rate (lr)	0.0002
First-order momentum decay coefficient of Adam's optimizer (b1)	0.5
Second-order momentum decay coefficient of Adam's optimizer (b2)	0.999
Latent space dimension (latent_dim)	100
Image size (img_size)	32
Number of channels	1
Number of iterations per saved generated image (sample_interval)	400

In terms of loss function, the basic GAN model uses Binary Cross Entropy Loss (BCELoss) to measure the performance of the discriminator and generator. The basic idea of GAN is to set up the generator and discriminator as an adversarial network, where the discriminators are trained to be binary classifiers. BCELoss is a loss function designed specifically for this type of dichotomous problem.

The generator in the GAN model is made up of several fully connected layers, each of which maps the input vectors to a feature space with a higher dimension. The Batch Normalization (BN) and Leaky ReLU activation functions inside each layer stabilize the training process. The generator transfers the high-dimensional features to an image with the given dimensions following a sequence of non-linear alterations. This is accomplished by using the Tanh activation function, which generates the image by standardizing the output pixel values to the interval [-1, 1]. Its loss function:

$$\mathcal{L}_G = -E_{z\sim p_z(z)}\left[\log D\left(G(z)\right)\right] \tag{3}$$

where z is the noise vector sampled from the latent space and p_z is the distribution in the latent space.$D\left(G(z)\right)$ represents the prediction of the generator's output image $G(z)$ made by discriminator.

Several fully connected layers make up the discriminator's layers as well, which extract and classify key features through layer by layer decreasing the input image's dimensionality. The Leaky ReLU activation function is used in each layer to hold onto negative information, and the Sigmoid activation function at the final layer generates a probability value ranging from 0 to 1 to represent the possibility that the input image is real. By continuously changing the weights, the discriminator enhances its capacity to discriminate between authentic and fraudulent images. Its loss function consists of two components: one for the real image and one for the fake image.

For real images:

$$\mathcal{L}_{\text{real}} = -E_{x\sim p_{\text{data}}(x)}[\log D\,(x)] \tag{4}$$

For fake images:

$$\mathcal{L}_{\text{fake}} = -E_{z\sim p_z(z)}\left[\log\left(1 - D\big(G(z)\big)\right)\right] \tag{5}$$

The average of the losses of the real and fake images yields the total discriminator loss:

$$\mathcal{L}_{\text{D}} = -\frac{1}{2}\left(E_{x\sim p_{data}(x)}[\log D\,(x)] + E_{z\sim p_z(z)}\left[\log\left(1 - D\big(G(z)\big)\right)\right]\right) \tag{6}$$

Where x represents the real image and p_{data} is the distribution of the real image. D(x) represents the output of the discriminator to the real image x. The meanings of z, p_z and $D\big(G(z)\big)$ are as mentioned above [2].

As for the CGAN model, the parameter settings are the same as GAN, the only difference is the addition of the number of categories parameter n_classes, whose default value is 10, to control the categories of the generated images.

In CGAN, the discriminator and generator loss functions use Mean Square Error Loss (MSELoss) to quantify the discrepancy between the target label and the discriminator output's validity. MSELoss is more sensitive to changes in output values, effectively reducing the occurrence of pattern crashes and in some cases leading to a more stable training process.

In contrast to the basic GAN where the inputs are only noise vectors, the generator in the CGAN model encodes the category labels into inputs combined with the noise vectors through an additional embedding layer, and the network structure includes a series of linear layers, BN layers, and Leaky ReLU activation functions, and finally outputs a Tanh-activated image. Its loss function:

$$\mathcal{L}_G = E_{z\sim p_z, y\sim p(y)}[\text{MSE}(D(G(z,y),y),1)] \tag{7}$$

Expand the MSE item:

$$\text{MSE}(D(G(z,y),y),1) = \frac{1}{m}\sum_{i=1}^{m}(D(G(z_i,y_i),y_i) - 1)^2 \tag{8}$$

Where z_i is the $i-$th noise vector. y_i is the $i-$th conditional label. $G(z_i,y_i)$ is the image generated by the generator. $D(G(z_i,y_i),y_i)$ is the output of the discriminator to the resulting image. The target value is 1 (which means "real").

The discriminator in CGAN also adds processing of category labels. It splices the input image with the corresponding category labels, and then performs the discriminant through multiple linear layers and the Leaky ReLU activation function, and finally outputs a binary classification result activated by Sigmoid. Compared with the discriminator in the basic GAN, the discriminator of CGAN considers not only the image itself but also the corresponding category information, thus enabling a more detailed judgment of the generated image. Its loss function also consists of two components:

$$\mathcal{L}_D = \frac{1}{2}\left[E_{x,y\sim p_{data}}[\text{MSE}(D(x,y),1)] + E_{z\sim p_z, y\sim p(y)}[\text{MSE}(D(G(z,y),y),0)]\right] \tag{9}$$

Expand the MSE item:

For real images:

$$\text{MSE}(D(x,y),1) = \frac{1}{n}\sum_{i=1}^{n}(D(x_i,y_i) - 1)^2 \tag{10}$$

For fake images:

$$\mathrm{MSE}(D(G(z,y),y),0) = \frac{1}{m}\sum_{i=1}^{m}(D(G(z_i,y_i),y_i) - 0)^2 \tag{11}$$

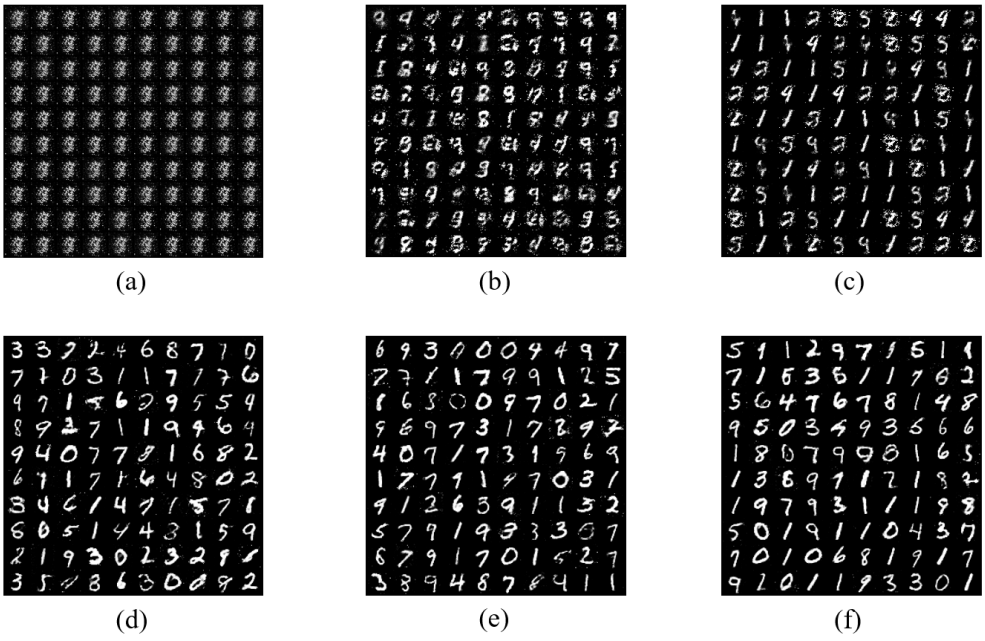
Where x_i is the $i-$th real image. $D(x_i,y_i)$ represents the output of the discriminator to the real image. The meanings of z_i, y_i and $D(G(z_i,y_i),y_i)$ are as mentioned above [9].

Based on the same dataset MNIST, the basic GAN and CGAN models were used to generate handwritten digital images, and the loss changes and generated results during the training process were recorded. By contrasting the two performed with respect to image quality and loss change, the advantages of CGAN in generating high-quality handwritten digital images are discussed and verified and can be applied to other scenarios that need to generate controllable images.

3. Results

3.1 Visualisation results analysis

In this study, there are 200 epochs in the model training round. The training results of the basic GAN model are shown in Fig. 4. As the number of training rounds increases, the handwritten digit images generated by the GAN become clearer and clearer, and very clear images are generated at epoch=200.



(a) (b) (c)

(d) (e) (f)

Fig. 4. Training process of Basic GAN: (a) epoch=0, (b) epoch=25, (c) epoch=50, (d) epoch=100, (e) epoch=150, (f) epoch=200 (Photo/Picture credit: Original).

Visualize the difference between the real image and the generated image during training. To assess the generator's performance and the quality of the images it produced, several real images were chosen at random from the test set and compared to the images the generator produced. Fig. 5 is an example of a comparison between the real images of the test dataset and the images generated by the GAN model training.

Fig. 5. Comparison of real images of the test dataset for GAN and images generated by GAN model training. (a) real image, (b) generated image (Photo/Picture credit: Original).

As can be observed from the comparison, although similar styles of the original image can be simulated by the GAN model, the GAN-generated image is random and cannot be specified to generate a specific class of handwritten numbers. The generated image is blurry and distorted, with unstable image quality.

For the CGAN model, there are also 200 epochs in the training round. The CGAN training results are shown in Fig. 6. As CGAN training progresses, the handwritten digital image becomes clearer, and with the input of label information, the CGAN can generate a specific handwritten digital image. Each column is shown here arranged from 0 to 9.

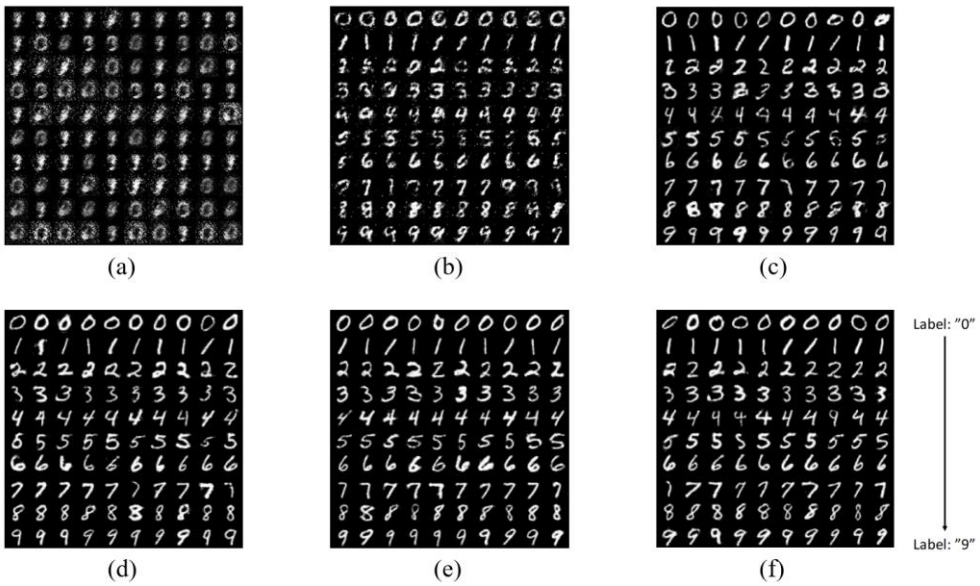


Fig. 6. Training process of CGAN: (a) epoch=0, (b) epoch=25, (c) epoch=50, (d) epoch=100, (e) epoch=150, (f) epoch=200 (Photo/Picture credit: Original).

Fig. 7 shows an example of a comparison between real images from the test dataset and images generated by CGAN model training. The comparison shows that the images produced by CGAN have a quality that is more consistent and exhibits less distortion and blurring than the images produced by the basic GAN. They also resemble the real images in the test dataset considerably better than the latter.

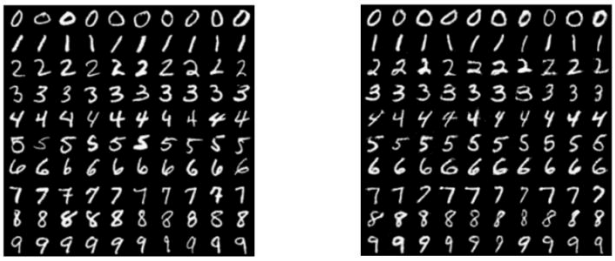


Fig.7. Comparison of real images of the test dataset for CGAN and images generated by CGAN model training. (a) real image, (b) generated image (Photo/Picture credit: Original).

3.2 Performance metrics

This study also presented the accuracy of the model in epoch graphically and performed statistical analysis. Fig. 8 and Fig. 9 are the loss curves of the trained GAN and CGAN models, respectively. The ordinate denotes the loss value, while the abscissa shows how many training cycles the model has had. Discriminator loss (D loss, orange line) shows a generally low and relatively stable trend, indicating an effective distinction between images that have been generated and those that are authentic. Generator loss (G loss, blue line) exhibits a high degree of variability, reflecting the fact that the generator has been trying to generate an image that convincingly deceives the discriminator.

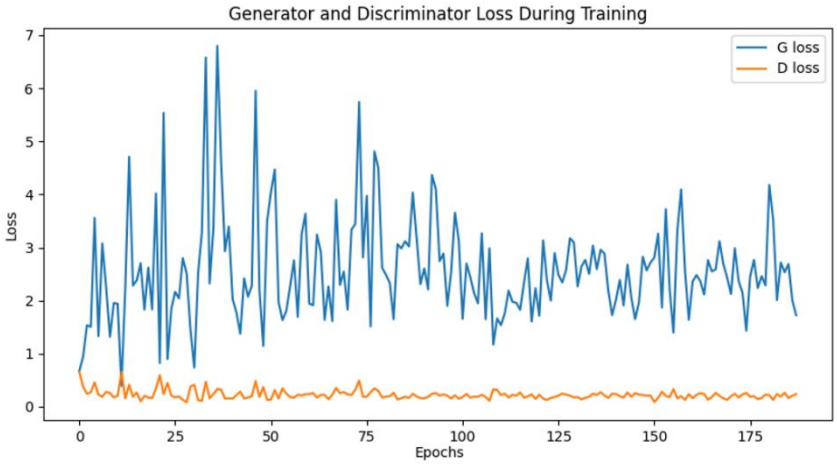


Fig. 8. Change in Loss of Training GAN (Photo/Picture credit: Original).

Fig. 8 demonstrates that the GAN model loss remains highly variable, ranging from 0.19 to 6.91. This reflects the fact that GAN exhibits greater training instability in generating handwritten digital images, and is prone to loss fluctuations and mode collapse, resulting in inconsistent quality of the generated images.

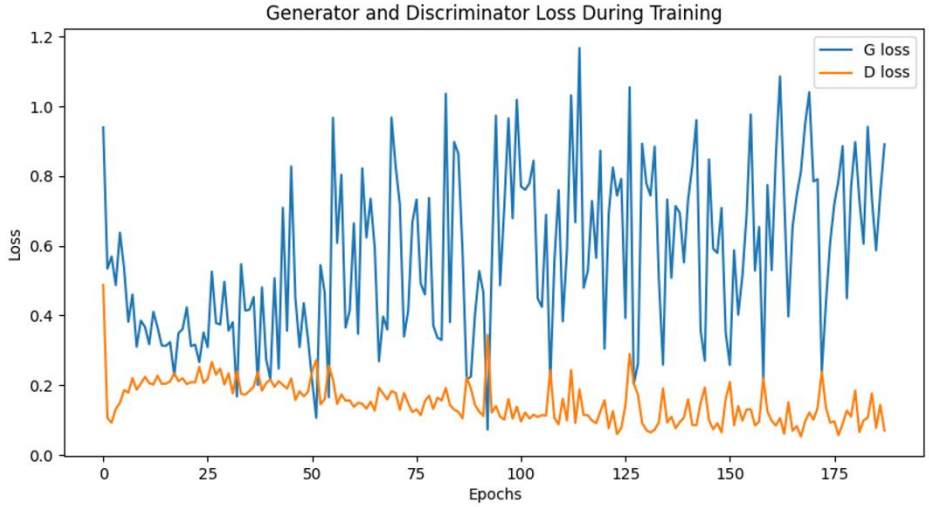


Fig. 9. Change in loss of training CGAN (Photo/Picture credit: Original).

Compared with Fig. 8 and Fig. 9, the CGAN loss change is significantly smaller than that of GAN, which is only in the range of 0.06 to 1.17. This reflects the higher training stability of the CGAN model, which produces higher quality and more stable images.

4. Discussion

4.1 Model comparison

Although the architecture of the basic GAN model is relatively simple, the training is intuitive, and the absence of conditional constraints allows the GAN to produce a more diverse output of handwritten digital images, there are some obvious problems when generating handwritten digital images using the basic GAN. The training of GAN models is unstable, and it is difficult to maintain a balance in the adversarial training between generators and discriminators, which tends to focus on several similar patterns, where mode collapse during the model's training may easily result. The quality of the resulting images also fluctuates greatly, sometimes appearing blurry, unsharp, and even with noise or artifacts. In addition, the GAN model cannot control the generated image categories, and it cannot generate handwritten digital images of the specified categories by inputting specific conditions.

Comparatively, due to the introduction of conditional variables, the CGAN model has a greater advantage in generating handwritten digital images. This paper summarizes three advantages: Firstly, CGAN training is more stable. The adversarial training of generators and discriminators is more orderly, which reduces the frequency of model instability and mode collapse, and CGAN can converge in a shorter time and maintain a more stable generation effect. Secondly, the image quality generated by CGAN is higher. CGAN can learn different classes of features more efficiently, and the resulting handwritten digital images outperform GAN in terms of clarity, detail richness, and consistency. Thirdly, CGAN has higher controllability. CGAN can accurately generate handwritten numbers of specific classes, avoiding the randomness of the generated results.

4.2 Model improvements

In terms of generating handwritten digital images, the CGAN model is a big improvement over the basic GAN model. However, there is also a lot that can be improved.

Included in the CGAN network structure is the standard BN layer, which has the main role of normalizing the data in each mini-batch to have a mean of 0 and variance of 1 to ensure that the generated intermediate features maintain a stable distribution. This reduces problems such as vanishing or exploding gradients. However, BN in CGAN does not take into account conditional information, and during the normalization process, all samples share the same mean and variance parameters, and the processed features are not adjusted for specific conditions, which may lead to the mixing of different classes of features and reduce the generated images' quality and controllability. The introduction of conditional batch normalization (CBN), can effectively improve such problems. Wang et al. combined CGAN with CBN to make full use of the category labels to batch process each category of data, and the mean and variance parameters are dynamically adjusted according to the input category labels, which allows different categories of information to be better transmitted and utilized, enhances model stability and convergence speed, and improves the quality of the generated images [1].

It is also possible to better optimize the model's network structure. The network structure of the Conditional Batch Normalized Conditional Generative Adversarial Network (CBN-CGAN) model studied by Wang et al. and the Class-Conditional Deep Convolutional Generative Adversarial Networks (C-DCGAN) model studied by Ding used convolutional layers instead of traditional fully connected layers. CNN possess the potent capacity to extract local features in an image layer by layer through convolution operations, enabling the generator to generate clearer and more detailed images, while the discriminator can more effectively judge the authenticity and category of the image [1, 12]. Adopting this structure helps the CGAN model learn the image's characteristics more effectively, generates images of higher quality, and increases the model's training efficiency. At the same time, using a convolutional layer rather than a fully connected layer can help lower the model's parameter count, reduce the risk of overfitting, accelerate the training convergence speed, and further enhance the stability and adaptability of the model.

5. Conclusion

This study mainly explores the performance differences between basic GAN and CGAN in handwritten digital image generation tasks. In this study, experiments were conducted on the MNIST dataset using the PyTorch deep learning framework to compare the generated image quality and the loss change during the training phase of the two models and assess the model's performance. Experimental results show that CGAN has advantages in generating handwritten digital images compared with basic GAN. The basic GAN model with a simple architecture can generate a variety of handwritten digital images, but it faces challenges such as mode collapse, unstable quality of generated images, and lack of category control during the training process. In contrast, by introducing conditional labels, the stability of CGAN training is improved, which not only enhances image production quality but also controls the category of generated images, thereby addressing the drawbacks of basic GAN.

The results of this study verify the potential of CGAN in high-quality and controllable image generation and provide a theoretical basis and practical reference for subsequent research. In the future, it will focus on optimizing the CGAN network structure (such as introducing CBN and using a convolutional layer in place of the conventional fully connected layer), enhancing the model's stability, accelerating the rate of training convergence, and boosting the output's image quality. The research focuses on applying the improved model

to more complex datasets and generation tasks, especially for scenarios that require controllable image generation, such as image enhancement and style transfer.

References

1. Wang, D. Xue, H. Wu, M. Wang. Handwritten digit recognition based on conditional generative adversarial networks. Liquid Crystal and Display, 12, 1284-1290 (2020).
2. K. Cheng, R. Tahir, L. K. Eric, M. Li. An analysis of generative adversarial networks and variants for image synthesis on MNIST dataset. Multimedia Tools and Applications, 1 (2020).
3. I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, et al. Generative adversarial nets. MIT Press (2014).
4. X. Chen, Y. Duan, R. Houthooft, J. Schulman, I. Sutskever, P. Abbeel. InfoGAN: Interpretable representation learning by information maximizing generative adversarial nets. (2016).
5. A. Odena. Semi-supervised learning with generative adversarial networks. arXiv (2016).
6. M. Song, Q. Su, M. Zhang. Handwriting digit generation based on GAN model. In 2021 3rd International Conference on Machine Learning, Big Data and Business Intelligence (MLBDBI), 144-148 (2021).
7. T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, X. Chen. Improved techniques for training GANs. (2016).
8. M. Mirza, S. Osindero. Conditional generative adversarial nets. Computer Science, 2672-2680 (2014).
9. Z. Qin, Y. Shan. Generation of handwritten numbers using generative adversarial networks. Journal of Physics: Conference Series, 1827(1), 012070 (10pp) (2021).
10. Y. Tiwari, A. Rasool, G. Hajela. Machine learning with generative adversarial network. In 2020 Second International Conference on Inventive Research in Computing Applications (ICIRCA) (2020).
11. Yann LeCun, Corinna Cortes, Chris Burges. The MNIST database of handwritten digits. 2024-08-15. https://yann.lecun.com/exdb/mnist/.
12. Z. Ding. A comparative analysis of handwritten digit generation models based on generative adversarial networks. Modern Industrial Economics and Information Technology, 04, 263-265 (2023).