Proceedings of the 3rd International Conference on Signal Processing and Machine Learning
DOI: 10.54254/2755-2721/4/20230438

Generate handwritten images based on DCGAN

Yu Huo

Department of Computer Science and Engineering, The Ohio State University, Columbus, OH, 43210, US

huo.125@osu.edu

Abstract. Nowadays, more and more machines are applied to artificial intelligence, and generate something itself has become more and more available. To augment databases to get a more accurate result from training a model, generating data is necessary. This paper introduces how to generate new handwritten images by training the existing database. The Generative Adversarial Networks model is used as the basic model. Deep Convolutional Generative Adversarial Networks and the MINIST database are applied as the method and training dataset respectively in this research. After 50 epochs by training 256 images each epoch, the new handwritten-number images are generated.

Keywords: Generative Adversarial Networks (GAN), MINIST, Generate Images, Deep Convolutional Generative Adversarial Networks (DCGAN), Minimax Formulation.

1. Introduction

With the improvement of technology in our current age, there are more and more machines appear in our daily life. In the beginning, machines and agents can just do insipid and single works, and almost all of them need to be operated by a human. For example, packaging cloth machines, cars, lamps, doors, etc. The human becomes to rely on machines, and operating machines to do simple work can't satisfy their demands, so more and more advanced machines and agents are needed. By those demands, machines and agents are developed more intelligently and skillful. Thus, the concept that deep learning is introduced. Deep learning (DL) a popular aspects of machine learning (ML) and artificial intelligence means that makes the machine obtain and learn knowledge like the human's brain [1].

The generative adversarial networks must be discussed when illustrating deep learning. Generative adversarial networks, also called GANs, is a model that learns itself through the competition of two neural networks. The generator and the discriminator are two neural networks in the GANs model. The purpose of the generator is to generate images from random noise to a normal image which is labeled as a generated fake image. The goal of the discriminator is to compare the ture image and generated image and distinguish which is real and which is fake. Initially, the generator can't generate a similar image to the real one, thus the discriminator can distinguish it and tell the generator the generated data is fake. Then, the generator needs to learn by itself to generate a more analogous image to confuse the discriminator. In order to find out the fake image, the discriminator also needs to learn by itself. This is how GAN learns for itself.

There are a lot of applications based on GANs. For example, generating example images for database, text-to-image translation, photograph editing, and generating similar images, to name but a

DOI: 10.54254/2755-2721/4/20230438

few [2]. Generate images is necessary in our society. Phones and computers are universal in our daily lives. Someone would like to type, but someone else would like to writing words for message. How could phones or computers recognize the script? The best answer is that training machines with a plenty of existing handwritten-number images. If there aren't enough scripts, generate handwritten-number images is necessary here. In this paper, the MINIST database [3] is likely used as training database to develop the DCGAN to produce similar handwritten-number images.

2. Related work

2.1.Generative Adversarial Networks (GAN)

There is a popular part in machine learning framework: Generative Adversarial Networks (GAN) [4]. GAN is a generative model which creates the new data instance by training existing database. In addition, GAN is an unsupervised learning which means that human gives the sample data to the machine without labels. The generator and the discriminator are two neural networks that compose the GAN. The generator is used to generate some data instances from its own 'knowledge', whereas the discriminator is operated to estimate data instances and calculate the probability of both data instances [5], the fake one and the real one. If the data is the generated one, the output probability is negative. If the data is the real one, the output probability is positive. G and D are two adversarial modules. In other words, generating the forged data to fool D is G's purpose, and the aim of the D is to point out the forged data from the G. The two modules learn from each other by calculating the loss.

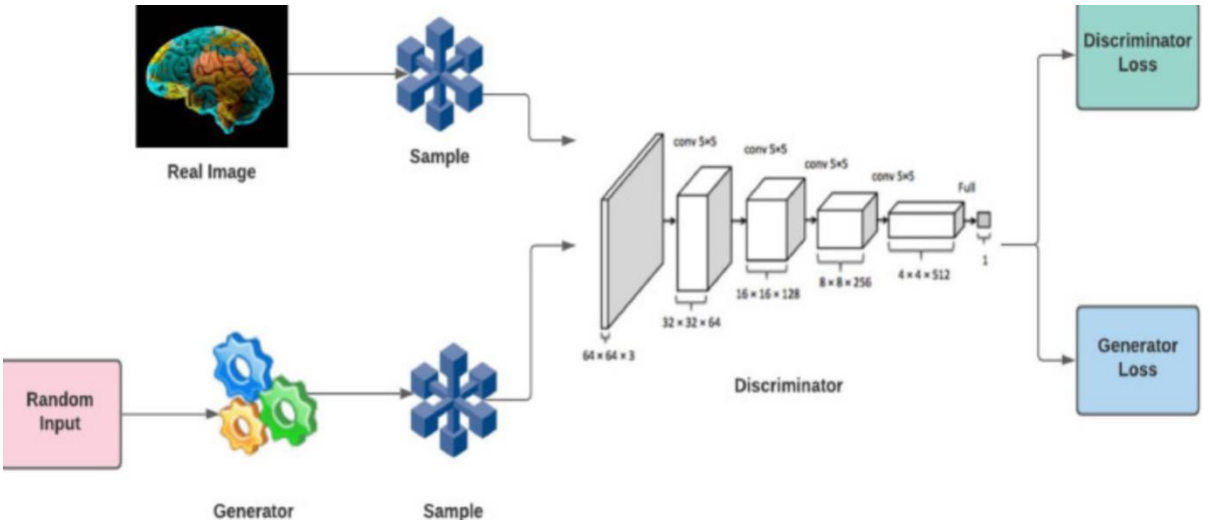

Figure 1. Diagram of Generative Adversarial Networks. [6].

2.2.Deep Convolutional Generative Adversarial Networks (DCGAN)

There is a generative adversarial network architecture called Deep Convolutional Generative Adversarial Networks (DCGAN). This network is developed by GAN and Convolutional Neural Networks (CNN). Involve CNN in the generator and the discriminator. DCGAN uses the convolutional layer in the discriminator and the transpose-convolutional layer in the generator. In the generator module, it uses the transpose convolution for upsampling [7]. The Batch Normalization function and Leaky ReLU function follow the transpose convolution. Compared with the generator, the discriminator uses convolution for downsampling followed by the Leaky ReLU function and Dropout function. In both modules, they replace all max pooling with convolutional strides [7]. The figure below shows the designed structure of the generator.

DOI: 10.54254/2755-2721/4/20230438

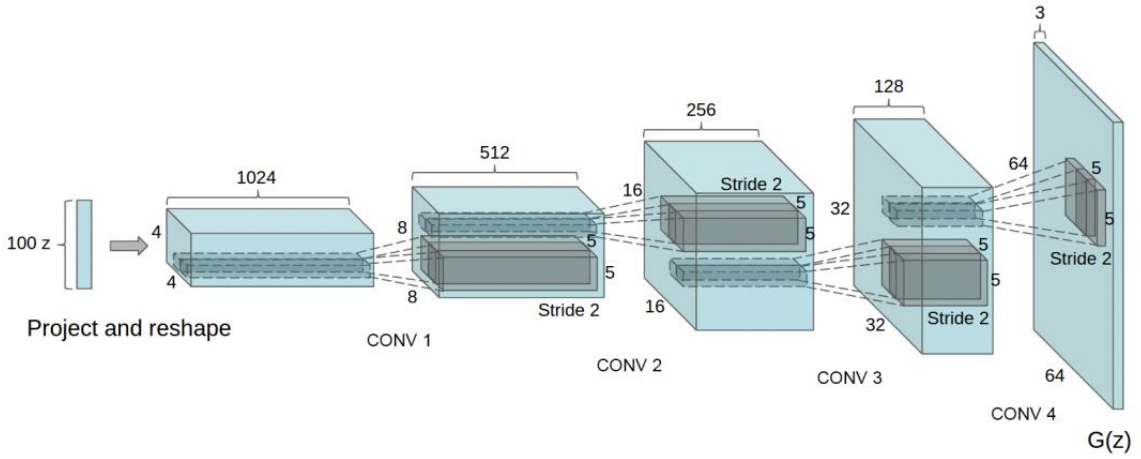

Figure 2. Designed structure of generator for DCGAN.

3. Method

3.1.Network Structure:

In the generator of DCGAN, decode a noise that is selected randomly to the normal size. A 7*7*256 random noise is added to the model as input. Normalize it and activate it by Leaky ReLU. Then the input model takes twice times transpose convolutions from 7*7*256 to 7*7*128 and 7*7*128 to 14*14*64 respectively. After that, do the last transpose convolution to switch the size of the model from 14*14*64 to 28*28*1 which is the same as the size of the original image in the MINIST [3] database.

To estimate the probability of two images, the discriminator encodes the image from a 3-dimensional array to a 1-dimensional array. It convolves twice to downsampling the images so that the shape of the image changes from 28*28*1 to 14*14*64, 14*14*64 to 7*7*128. After each convolution, the model would like to call the Leaky ReLU function to activate the model and avoid the vanishing gradients. Then the model needs to randomly drop our 20% data in order to avoid overfitting. After all the convolutions, flatten the model and add random noise to it. Finally, return the calculated loss to the generator. Return the positive value if the input of the discriminator is real. Return negative if not.

3.2.Minimax formulation:

To train the generator and the discriminator, the minimax formulation needs to be introduced. The formulation below is the Minimax objective function from Slides of Stanford University.

Minimax function:

$$\min_{\theta_g}\max_{\theta_d}\left[\mathbb{E}_{x\sim p_{data}}\log D_{\theta_d}(x)+\mathbb{E}_{z\sim p(z)}\log\left(1-D_{\theta_d}\left(G_{\theta_g}(z)\right)\right)\right] \quad (1)$$

$\log D_{\theta d}(x)$ is the output of the discriminator for true data x. For generated data/fake data G(z), the discriminator will output $D_{\theta d}(G_{\theta g}(z))$ as result. In this model, the Discriminator (θ_d) wants to augment D(x) close to 1 which means the discriminator can distinguish the real data with a high probability and D(G(z)) approximate to 0 [8], the discriminator can determine the generated fake data as the real data with a very low probability. For the probability, 1 means the image is exacting (100%) the real data, and 0 means the image is absolutely fake (0%) data. By contrast, the purpose of the generator (θ_g) is to increase D(G(z)) approximately to 1 [8]. By the way of explanation, cheat the discriminator into thinking the forged output is the real one with a high probability.

DOI: 10.54254/2755-2721/4/20230438

3.3.Loss Function

When training the generative adversarial network, the model generates new data at first [9]. Then the discriminator estimates the generated data and the real one and outputs a probability. In order to train the discriminator, the generator needs to be fixed temporarily. Training the discriminator with a batch of real & fake data for a few epochs, to minimize the loss function is needed:

$$\mathcal{J}(D) = \mathbb{E}_{x\sim p_{data}}[-\log D(x)] + \mathbb{E}_z\left[-\log\left(1 - D\left(G(z)\right)\right)\right] \quad (2)$$

In order to minimize the loss function of the discriminator, D(x) is expected nearly to 1, and D(G(z)) is expected approaximately to 0. Since in the log(x) function, between 0 and 1, the closer the x to 1, the smaller the -log(x) will be.

After training the discriminator, training the generator is required. Fix the discriminator is demanded when training the generator. The loss function of the generator should be maximized:

$$\mathcal{J}(G) = \mathbb{E}_z\left[-\log\left(1 - D\left(G(z)\right)\right)\right] \quad (3)$$

In order to maximize the loss function of the generator, D(G(z)) is expected nearly to 1. In log(1-x) function, the closer the x to 1, the larger output the -log(1-x) will generate [10].

4. Experiment

4.1.Dataset

In order to generate handwritten-number images, the dataset MINIST [3] would be a good choice. The images in the dataset are similar but have their own specializations. The table below displays the categories and counts of sample images in each dataset in the MINIST, and there are two examples from the dataset:

Table 1. Number of data in database, training, and testing.

	Number of categories	Number of images
Database	10 (0, 1, 2, 3, 4, 5, 6, 7, 8, 9)	70,000
Training set	10 (0, 1, 2, 3, 4, 5, 6, 7, 8, 9)	60,000
Testing set	10 (0, 1, 2, 3, 4, 5, 6, 7, 8, 9)	10,000

Figure 3. Two examples in MINIST database.

4.2.Experiment process

Since the picture is a little large, it needs convolution or transpose-convolution twice in both discriminator and generator. In training the model, for each image, it would like to generate data at first. Then put real data and generated fake data into the discriminator respectively and return their flattened output respectively. After getting the output, calculate the loss which is computed by the

DOI: 10.54254/2755-2721/4/20230438

cross_entropy() function and gradients of generator and discriminator. Apply the gradients to the optimizer of generator and discriminator respectively. For each epoch, the code will generate 12 examples for the generated images.

4.3.Experiment result

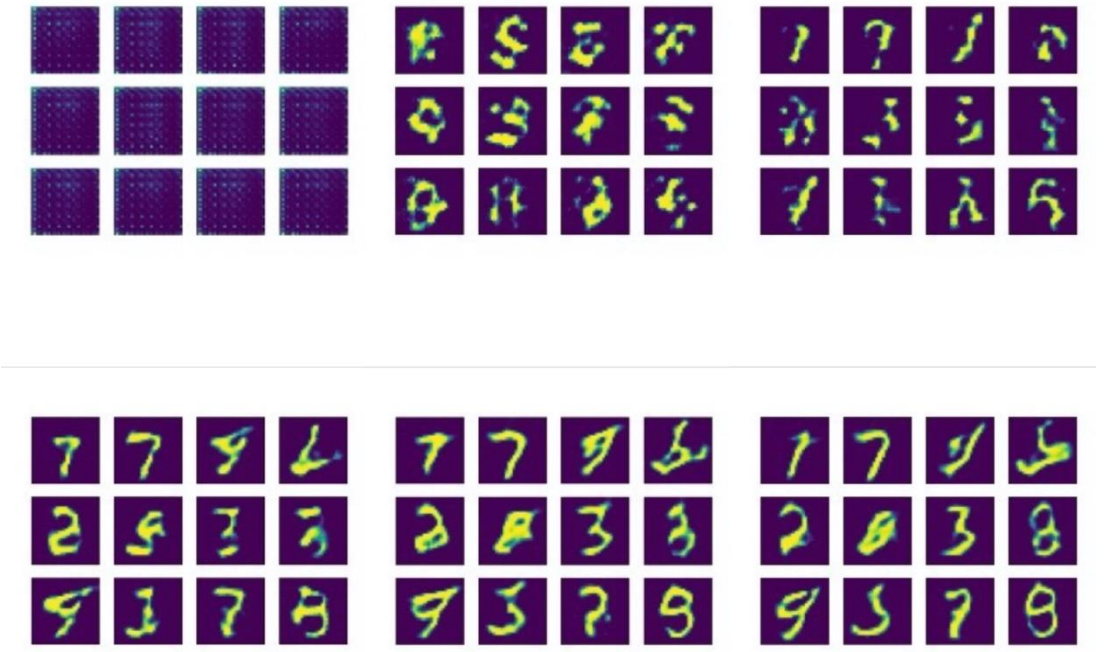

Figure 4. Outputs for Epoch1, Epoch10, Epoch20, Epoch30, Epoch40, Epoch50 (From left to right, top to bottom).

5. Conclusion

In this research, the generative adversarial network (GAN) is discussed mainly. At the beginning, the background of machine learning, GAN, and some applications based on GAN are introduced. More and more advanced machines are needed to satisfy human's demands. The purpose of the whole experiment is to generate the handwritten-number image based on GAN by training several existing handwritten-number images. In the related work part, the fundamental architecture GAN and the method DCGAN which is the architecture that is used in the experiment are illustrated. In DCGAN, it utilizes the Convolutional Neural Networks in both the generator and the discriminator. In the method part, the network structure, mathematical formulation, and loss function are introduced. In the Minimax function, the generator is intended to increase D(G(z)) whereas the discriminator is more likely to augment D(z) and diminish D(G(z)). The most crucial part is the experiment. In this part, the database and the process of the experiment are introduced and displayed.

References

[1] Burns, E., & Brush, K. (2021, March 29). What is deep learning and how does it work? SearchE nterpriseAI. Retrieved August 19, 2022, from https://www.techtarget.com/searchenterpriseai

DOI: 10.54254/2755-2721/4/20230438

/definition/deep-learning-deep-neural-network

[2] Brownlee, J. (2019, July 12). 18 impressive applications of generative adversarial networks (Gans). Machine Learning Mastery. Retrieved August 19, 2022, from https://machinelearningmastery.com/impressive-applications-of-generative-adversarial-networks/

[3] Wikimedia Foundation. (2022, July 31). MNIST database. Wikipedia. Retrieved August 24, 2022, from https://en.wikipedia.org/wiki/MNIST_database

[4] Wikimedia Foundation. (2022, August 19). Generative Adversarial Network. Wikipedia. Retrieved August 19, 2022, from https://en.wikipedia.org/wiki/Generative_adversarial_network

[5] Liu, Y., Qin, Z., Luo, Z., & Wang, H. (2017, May 7). Auto-painter: Cartoon image generation from sketch by using conditional generative Adversarial Networks. arXiv.org. arXiv.1705.01908, 2017

[6] Aggarwal, A., Mittal, M., & Battineni, G. (2021). Generative Adversarial Network: An overview of theory and applications. International Journal of Information Management Data Insights, 1(1), 100004. https://doi.org/10.1016/j.jjimei.2020.100004

[7] Hui, J. (2018, June 24). Gan - DCGAN (deep convolutional generative adversarial networks). Medium. Retrieved August 19, 2022, from https://jonathan-hui.medium.com/gan-dcgan-deep-convolutional-generative-adversarial-networks-df855c438f

[8] Lecture 13: Generative Models - Stanford University. (n.d.). Retrieved August 20, 2022, from http://cs231n.stanford.edu/slides/2017/cs231n_2017_lecture13.pdf

[9] Lauzon, F. Q. (2012, July). An introduction to deep learning. In 2012 11th International Conference on Information Science, Signal Processing and their Applications (ISSPA) (pp. 1438-1439). IEEE.

[10] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., & Bengio, Y. (2020). Generative Adversarial Networks. Communications of the ACM, 63(11), 139–144. https://doi.org/10.1145/3422622