Collection of Conditional Generative Adversarial Network

MU-HSUAN LI[a]

[a]*Department of Civil Engineering, National Taiwan University, Taipei, Taiwan*

Keywords: machine learning, deep learning, generative adversarial network

1. Introduction

Nowadays, generative adversarial networks or GANs, are wisely used to generate synthetic images in computer vision, and it has a variety of architectures that can be applied to different situations using deep learning methods.

One of these is conditional GAN(CGAN), Mirza and Osindero [1] modify simple GANs structure let the generator and discriminator are conditional during the training process so that the generator can specify what label you want the generator produce.

However, the simple CGAN suffered from several issues such as Mode collapse and convergence instability problems like simple GAN.When the generator produces limited varieties of samples, this GAN failure is called model collapse. And convergence instability means GAN convergence is often a fleeting, rather than stable state.

In this project, we want to implement some different training strategy or model structure in CGAN to see if it can improve these problems. Both the training loss function curve and synthetic images generate by different generator will be analyze and discussion.

Email address: r09521245@ntu.edu.tw (MU-HSUAN LI)

2. Methodology

In this project,we will implement two well-known GAN models that can significantly improve the performance of generated images with CGAN training process, namely DCGAN and WGAN for short.

Mirza and Osindero [1] perform the conditioning by feeding y into the both the discriminator and generator as additional input layer. Fig 1 illustrates the structure of a simple conditional adversarial net. The objective function of two adversarial network of CGAN would be as (1).

$$\min_G \max_D V(D,G) = \mathbb{E}_{\mathbf{x}\sim p_{data}(\mathbf{x})}[\log D(\mathbf{x}|\mathbf{y})] + \mathbb{E}_{\mathbf{z}\sim p_z(\mathbf{z})}[\log(1 - D(G(\mathbf{z}|\mathbf{y})))] \tag{1}$$

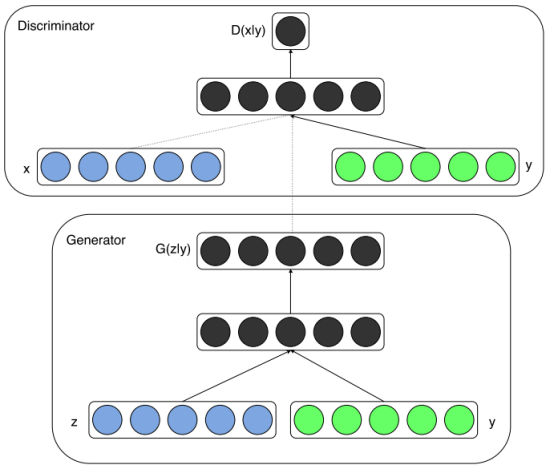


Figure 1: CGAN structure [1]

The deep convolutional generative adversarial network is presented by Radford et al. [2],which showed us how to effectively use convolutional techniques with GANs (DCGAN) to create images that improve fake image quality. Fig 2 shows the generator architecture with convolution kernel in DCGAN.

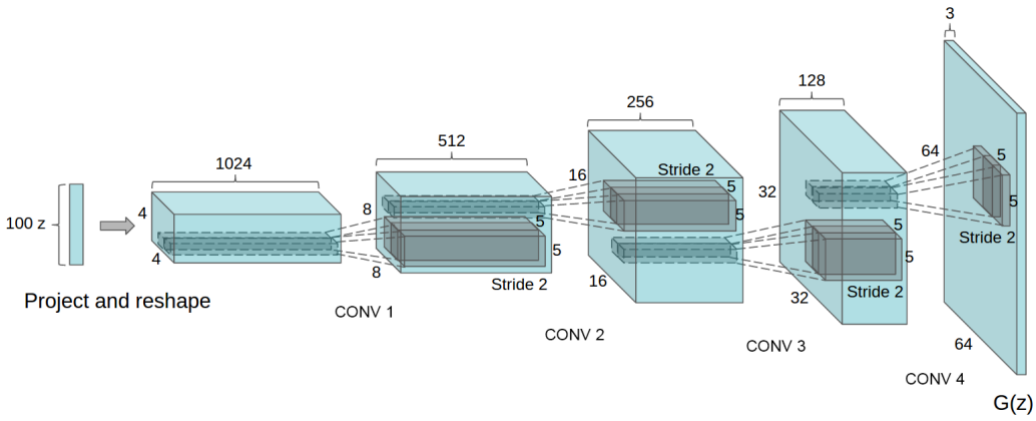


Figure 2: DCGAN generator [2]

Arjovsky et al. [3] use the Wasserstein distance instead of Jensen-Shannon divergence(JS divergence) in a simple GAN to calculate the distance between real images and synthetic images distributions is called WGAN. It successfully improved the gradient issue can lead to unstable training. (2) is the objective function of WGAN, from the Wasserstein distance.

$$\max_{w \in W} \mathbb{E}_{x \sim \mathbb{P}_r}[f_w(x)] - \mathbb{E}_{z \sim p(z)}[f_w(g_\theta(z))] \tag{2}$$

Frechet Inception Distance, or FID for short, was proposed and used by Heusel et al. [4], is a metric for evaluating the quality of generated images, developed specifically for evaluating the performance of generative adversarial networks. The FID score is calculated by a famous pre-trained Inception v3 model, but the output layer of the model is removed and outputs the 2048-dimensional vectors from the previous layer:

$$\text{FID}(g, r) = ||\mu_g - \mu_r||_2^2 + Tr(\Sigma_g + \Sigma_r - 2(\Sigma_g \Sigma_r)^{\frac{1}{2}})) \tag{3}$$

In (3), the μ_1 and μ_2 refer to the feature-wise mean of the real and generated images, e.g. 2,048 element vectors where each element is the mean feature observed across the images. the Σ_g and Σ_r are the covariance matrix for the real and generated feature vectors.

The FID represents the distance between the feature vector of the generated image and the feature vector of the real image. The closer the distance is, the better the generated model is, i.e., the image have high quality and rich in diversity.

Finally, combine the DCGAN convolution kernel framework with the CGAN training process both in discriminator and generator, which objective function is (1), become deep convolutional conditional generative adversarial network(DCCGAN). In addition, replace original objective function (1) with (2) but still conditioning in DCCGAN, called WCGAN.

In conclusion, there are three different GAN in total of this project as shown in Table 1. In the end, makes every generator to produce images about 100 samples each label and evaluate them by FID score after fully training.

Model	**Structure**	**Objective function**
CGAN	fully connected layer	JS divergence
DCCGAN	convolutional layer	JS divergence
WCGAN	convolutional layer	Wasserstein distence

Table 1: Different GAN

3. Datasets

Fashion-MNIST is a very popular dataset of Zalando's article images, consisting of training set of 60,000 samples and testing set of 10,000 samples. Each sample is a 28x28 grayscale image, associated with a label from 10 classes.

Fig 3 shows some examples for each classes in Fashion-MNIST dataset. Combine all training set and testing set total 70,000 samples to train each GAN in this project.

Figure 3: Fashion-MNIST samples

4. Results and Discussions

For data augmentation, simply resize the image to 32x32 and convert the pixels from [0,255] to [-1,1] for all data. In the generator of all GAN, a noise prior z with dimensionality 100 is drawn from a uniform distribution.The model are trained using Adam optimization algorithm for CGAN and DCCGAN, RMSprop optimization algorithm for WCGAN. All of them with mini-batches of size 128 and learning rate 0.0001.The more detail of model structure and results are as following:

4.1. CGAN

The conditional extra information y are mapped to 10 dimensions both in discriminator and generator. For the generator, the fully connected layer input size is image size $100 + 10 = 110$, then connect to three hidden layer, with sizes of 256, 512 and 1024 respectively, the output size is $32 \times 32 = 1024$ with Tanh activation function.

For the discriminator, the input size is $32 \times 32 + 10 = 1034$, then connect to three hidden layer, with sizes of 1024, 512 and 256 respectively, the output size is 1 with Sigmoid activation function.

After training CGAN 250 epochs, Fig 4 shows CGAN learning curve every epoch, and Fig 5 are some generative images during the training.

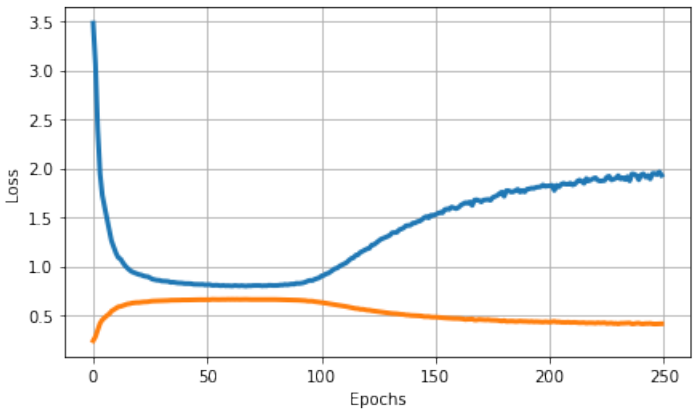


Figure 4: CGAN learning curve

(a) (b) (c) (d)

Figure 5: CGAN generative image (a) 20 epoch (b) 50 epoch (c) 120 epoch (d) 250 epoch

4.2. DCCGAN

In DCCGAN, all weights are initialized from a zero-centered Normal distribution with standard deviation 0.02, and no fully connected or pooling layers are used refer to [2].

A 100 dimensional uniform distribution Z and 10 dimensions extra information y are projected to a small spatial convolutional representation with $64 \times 4 = 256$ feature maps, and concatenate to $256 \times 2 = 512$ feature maps convolutional representation. A series of three fractionally-strided convolutions with sizes of $64 \times 4 = 256$, $64 \times 2 = 128$ and 1 feature maps, then convert this high level representation into a 32x32 pixel grayscale image with Tanh activation function for the generator. For the discriminator, input image convolutional representation concatenate with extra information y convolutional representation into $64 \times 2 = 128$ feature maps. Convert these representation by 3 convolutional layers with sizes of $64 \times 4 = 256$, $64 \times 8 = 512$, the output size is 1 with Sigmoid activation function.

After training DCCGAN 80 epochs, Fig 6 shows DCCGAN learning curve every epoch, and Fig 7 are some generative images during the training.

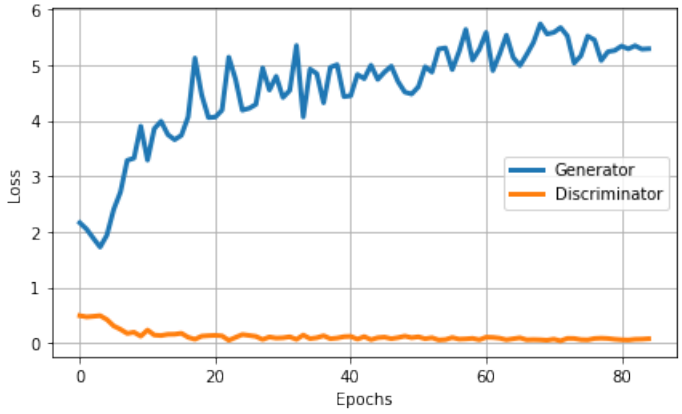
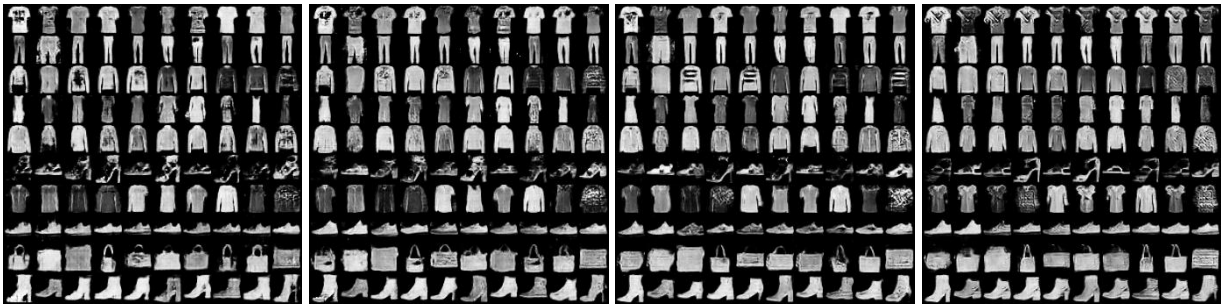


Figure 6: DCCGAN learning curve



(a) (b) (c) (d)

Figure 7: DCCGAN generative image (a) 10 epoch (b) 20 epoch (c) 50 epoch (d) 80 epoch

4.3. WCGAN

In WCGAN, the generator and discriminator (or critic in WGAN) structure are the same as DCCGNA in 4.2, but remove the Sigmoid function in output layer of discriminator. After training DCCGAN 80 epochs and clamp the weights to a fixed box [-0.01,0.01] after each gradient update for weight clipping.

After training WCGAN 80 epochs, Fig 8 shows WCGAN learning curve every epoch, and Fig 9 are some generative images during the training.

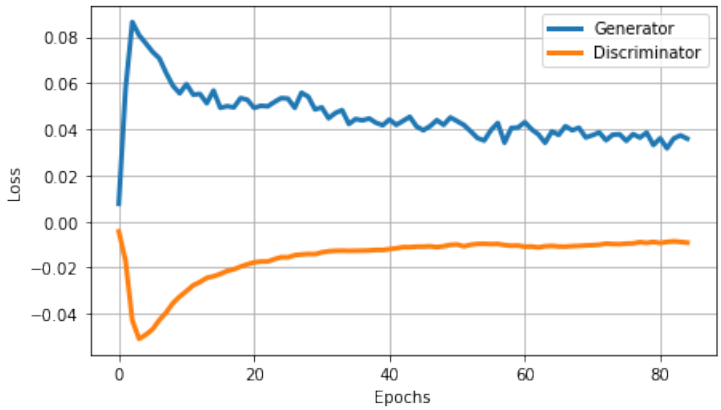
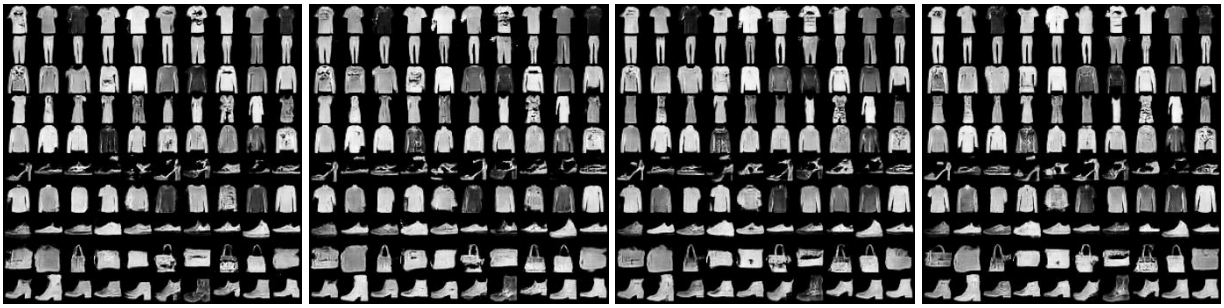


Figure 8: WCGAN learning curve



(a) (b) (c) (d)

Figure 9: WCGAN generative image (a) 10 epoch (b) 20 epoch (c) 50 epoch (d) 80 epoch

4.4. FID score

Evaluate each generator of different GAN by producing 100 images every classes in Fashion-MNIST, and calculate the FID score with real Fashion-MNIST also 100 images every classes, Table 2 shows all GAN final FID score.

Model	FID score
CGAN	75.766
DCCGAN	71.119
WCGAN	51.412

Table 2: FID score

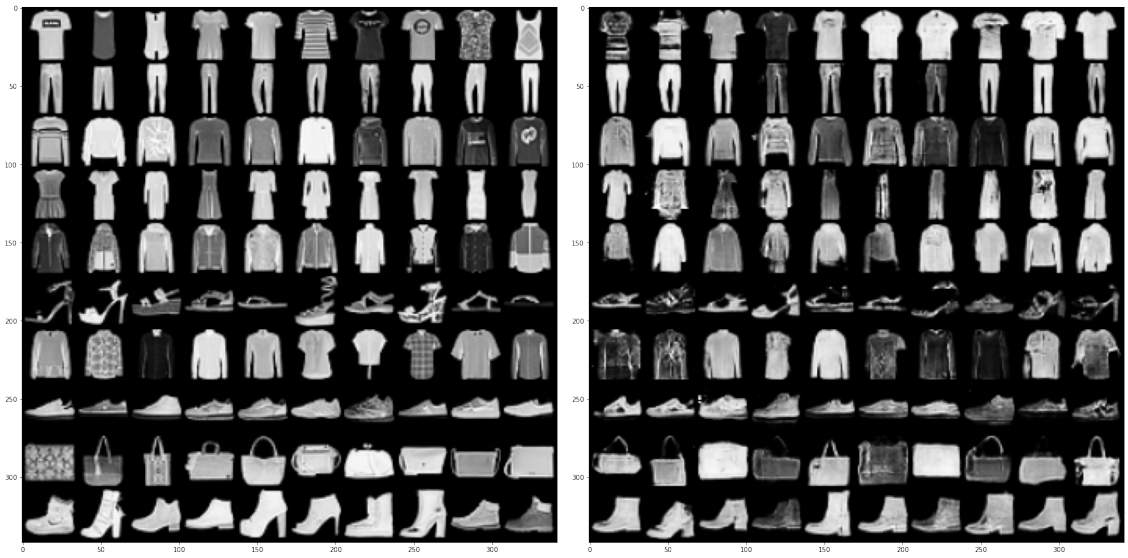
5. Conclusion

We propose collection of conditional generative adversarial networks with different model structure and training strategy in this project. There are some conclusion we can make after observing all GAN results:

1. CGAN gets the worst FID score, and we can also observe that the fake images it produces are less clear than the other generators.
2. we observe some mode collapse in DCCGAN produced images in Fig 7, it may lead the difference between DCCGAN FID score and CGAN FID score is not significant.

3. The learning curve of DCCGAN in Fig 6 may have encountered the problem of unstable convergence, i.e., discriminator dominance.
4. Using Wasserstein distence of WCGAN can effectively improve the unstable convergence issue and mode collapse problem, and the FID score indicates that the generated images have the highest quality and richness from others.

Ultimately, Fig 10 provides a deeper look at the comparison of the WCGAN-generated images with the real images of the Fashion-MNIST dataset.



(a) (b)

Figure 10: (a) Real images (b) WCGAN generated images

References

[1] Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. arXiv preprint arXiv:1411.1784, 2014.

[2] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. arXiv preprint arXiv:1511.06434, 2015.

[3] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In International conference on machine learning, pages 214–223. PMLR, 2017.

[4] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, Advances in Neural Information Processing Systems, volume 30. Curran Associates, Inc., 2017. URL `https://proceedings.neurips.cc/paper/2017/file/8a1d694707eb0fefe65871369074926d-Paper.pdf`.