Study of Generative Adversarial Networks (GANs)

Hui Kwat Kong
20123133

1. Introduction

Generative adversarial networks (GANs), formed in 2014[1], is a state of the art deep neural network with many applications. Unlike the traditional machine learning in unsupervised learning (it does not require a targeted label), GANs is a generative model which generates new content by given data.

It is interesting to see the generated images of MNIST (handwritten digit database) first by GANs at the right side from its original paper[1]:

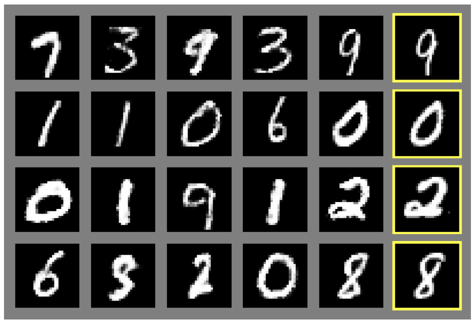

Figure 1. Generated images by GANs of MNIST

2. Principle

The analogy of GANs is known as a fake-currency detection game between a counterfeiter and police[1]. According to the tutorial of GANs by Goodfellow[2], GANs consists of two characters, namely, the generator (counterfeiter) and the discriminator (police).

The counterfeiter tries to produce fake money and deceive the police (discriminator) by looking at the real banknote. The work of the discriminator is to distinguish whether the given money is real or not. Firstly, the money generated by the counterfeiter is too coarse that can be distinguished easily. Based on these failures, now the counterfeiter tries hard to produce more sophisticated money. At the same time, the police is now more experienced to distinguish the money.

As repeating this process many times, both parties learn from the opposite party (that is the reason it is called as "adversarial") and therefore become mature and sophisticated

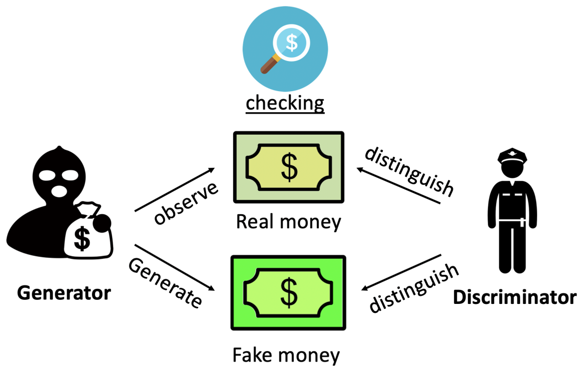

Figure 2. Analogy of the game between counterfeiter and police

enough. Now, the fake money produced by the counterfeiter is very realistic to other people except the police.

3. Objective function for GANs

Recall that an objective function, is a function to be optimized during training (usually to maximize it, if we want to minimize it, usually it is called as loss function). There are several methods to find the optimal point of the objective function, such as Maximum Likelihood Estimation (MLE) and different ways of gradient ascending/descending etc.

GANs consists of two deep neural networks, the generator network (denoted as G) and the discriminator network (denoted as D). The objective of G is to output the fake data $G(z)$ to fool D from the distribution p_{data}. Oh the other hand, D outputs a probability of a real data, where its objective is to maximize the probability of real label and minimize the probability of fake label (with distribution p_z).

During training, G and D update their parameters θ_g and θ_d based on the below minmax objective function $V(G,D)$:

$$\min_{\theta_g}\max_{\theta_d} V(G,D) = \min_{\theta_g}\max_{\theta_d}\Big[\mathbb{E}_{x\sim p_{\text{data}}}\log D_{\theta_d}(x) + \mathbb{E}_{z\sim p(z)}\log\left(1 - D_{\theta_d}\left(G_{\theta_g}(z)\right)\right) \tag{1}$$

The discriminator should give a high value to true image ($D_{\theta_d}(x)$), which is same as taking logistic function for it (($logD_{\theta_d}(x)$)). It should also give a low value to fake image ($G_{\theta_g}(z)$), which is the same as by subtracting it and taking the logistic function ($log(1 - D_{\theta_d}(x))$). In short, it can be achieved by maximizing both part by above objective function $V(G, D)$ for θ_d.

For generator, it is desirable to fool the generator, which can be achieved if $D_{\theta_d}(x)$ is close to 1, which is same as minimizing $log(1 - D_{\theta_d}(x))$. There is a difference for the genrator, that is to say, the first part in the minmax function $V(G, D)$ does not depend on the generator. (In the analogy of the game between counterfeiter and police, whether a police can recognize the real money has nothing to do with the counterfeiter.) In short, it is same as minimizing the objective function $V(G, D)$ for θ_g.

4. Training GANs

Instead of the original objective function, there is a modification of the generator part for practical reason in training:

$$\max_{\theta_g}[\mathbb{E}_{z\sim p(z)} \log\left(D_{\theta_d}\left(G_{\theta_g}(z)\right)\right)] \tag{2}$$

The reason is that at the beginning of training, the generated fake images are too obvious (i.e. $D_{\theta_d}(x)$ is close to zero) and therefore the loss is almost zero and with zero gradient.

For the new objective function, the gradient is higher than the original one when $D_{\theta_d}(x)$ is small:

$$\nabla_{\theta_g} \log\left(D_{\theta_d}(G_{\theta_g}(z^{(i)}))\right) = \frac{\nabla_{\theta_g} D_{\theta_d}(G_{\theta_g}(z^{(i)}))}{D_{\theta_d}(G_{\theta_g}(z^{(i)}))} \tag{3}$$

but for the one gradient:

$$\nabla_{\theta_g} \log\left(1 - D_{\theta_d}(G_{\theta_g}(z^{(i)}))\right) = \frac{-\nabla_{\theta_g} D_{\theta_d}(G_{\theta_g}(z^{(i)}))}{1 - D_{\theta_d}(G_{\theta_g}(z^{(i)}))} \tag{4}$$

The denominator in (3) is much small than the denominator in (4) when $D_{\theta_d}(x)$ is small.

The original paper states the optimal point of discriminator D and the training procedure, which are skipped here to avoid the mathematical details and direct copying, but providing informal idea for illustration here:

- The training takes place in **minibatch** of noise samples and real examples. The objective of **minibatch** is to accelerate the training process since training a DNN by all data is extremely heavy.
- The original stochastic gradient descent of generator is usually replaced by stochastic gradient ascend of equation (2).

5. Comparison

We are going to compare the other two generative models PixelCNN[12] and Variational Autoencoders (VAEs)[5] with GANs:

5.1. PixelCNN

Convolutional Neural Network (CNN) is a deep neural network for analyzing visual imagery, which typically consists of convolution layer, pooling layer and fully connected layer, connected by the activation functions (output of the node by a set of given input in a neural network).

PixelCNN is an CNN-based explicit density model, which decomposes the probability of an given image x into a product of distribution of the pixels. The training objective is to maximize the following likelihood, which can be achieved by Maximum Likelihood Estimation (MLE)).

$$p(\mathbf{x}) = \prod_{i=1}^{n^2} p\left(x_i | x_1, \ldots, x_{i-1}\right) \tag{5}$$

Comparing with GANs, PixelCNN is able to output the explicit data likelihood, where GANs is an implicit density model approach, and hence training PixelCNN is more stable than training GANs. However, PixelCNN is slower in training due to sequential generation and its generative quality is worse than GANs.

5.2. Variational Autoencoders

Before we go through Variational Autoencoders (VAEs), it is necessary to discuss autoencoders first.

Autoencoders is an unsupervised neural network which aims to reconstruct the input data x. Autoencoders consists of the encoder and decoder network, where the encoder compresses the original information from input x as z, and the decoder reconstructs the data $\hat{x}$ given by the compressed features z. The loss function is calculated by the original input x and the reconstructed $\hat{x}$ through backpropagation.

Unlike encoding and decoding the explicit single point, VAEs encodes the distribution of x over the latent space. Then, a point from the latent space can be sampled from that latent distribution. Finally the output is decoded by the decoder network (distribution $p(x|z)$) and error can be measured.

In practice, the distribution is assumed to be normal, so that now we need to find out the optimal mean ($\mu_{z|x}$) and variance ($\sum_{z|x}$) of the latent distribution in encoder and the mean ($\mu_{x|z}$) and variance ($\sum_{x|z}$) of latent distribution in decoder.

Same with PixelCNN, VAEs learns the approximate distribution of the given data explicitly, which may be useful for feature compression and representation for other purposes. However, it is known that the samples generated by VAEs are blurrier than GANs. (The mathematical reason has been explained in detail in our lecture note 13)

6. Special topic in GANs: domain transfer

In this section, we would like discuss the recent development in GANs.

6.1. Conditional GANs

The original GANs trains the generator and discriminator with no supplementary information. Therefore, Conditional GANs(cGANs)[8] adds some condition constraints to GANs such that the model is more capable to handle different contextual information.

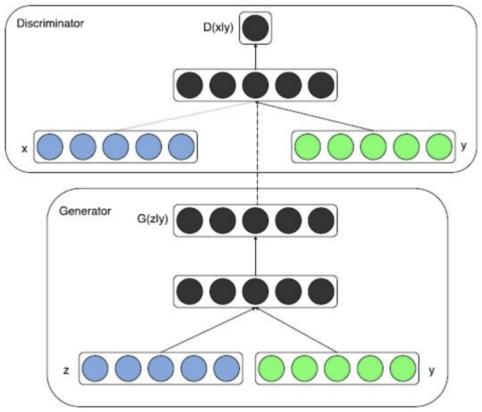

Figure 3. The structure of cGANs from the original paper

According to Figure 3 from the original paper[8], both the generator and discriminator in cGANs include the additional condition y. For example, the paper shows that cGANs trained from MNIST images (database of handwritten digits) with its digit labels. The updated objective function is as follow:

$$\min_{\theta_g}\max_{\theta_d}\left[\mathbb{E}_{x\sim p_{\text{data}}}\log D_{\theta_d}(x|y)\right.\\ \left.+\mathbb{E}_{z\sim p(z)}\log\left(1-D_{\theta_d}\left(G_{\theta_g}(z|y)\right)\right)\right. \tag{6}$$

After the foundation of cGAN, there was a popular paper, "Image style transfer using convolutional neural networks"[6], which introduced the application of style transfer by GANs. Later, there were several GANs methods for domain transfer, such as ix2pix GANs, Cycle GANs, Perceptual Adversarial Networks and Star GANs etc.

In the next two subsection, Cycle GANs and pix2pix will be discussed.

6.2. pix2pix

Pix2pix[10] is a cGANs based model, which aims to perform "Image-to-Image" translation. One of the drawbacks mentioned in the other generated models is that they tend to generate blurry images, because the image is generated pixel by pixel. Instead, pix2pix considers the whole structure of the image.

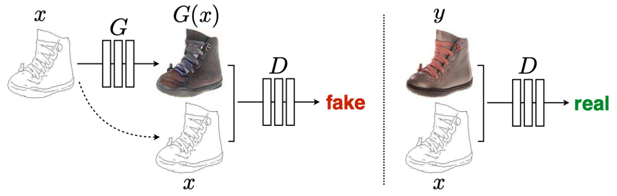

Figure 4. The methodology of pix2pix from the original paper

Pix2pix requires a paired training data. For example in Figure , both the generator and discriminator pix2pix requires both the real shoes (y) and the framework of that shoe (as x). That is to say, the generator do not start from scratch, but starting with the help of the picture framework (additional condition in cGANs). Oh the other hand, the discriminator will also consider the framework x as well.

For formal explanation, the objective function is as follow:

$$\begin{aligned}\mathcal{L}_{cGAN}(G,D)=&\mathbb{E}_{x,y}[\log D(x,y)]+\\ &\mathbb{E}_{x,z}[\log(1-D(x,G(x,z))]\\ \mathcal{L}_{L1}(G)=&\mathbb{E}_{x,y,z}\left[\|y-G(x,z)\|_1\right.\end{aligned}$$

Here $\mathcal{L}_{cGAN}(G,D)$ describes the cGANs loss as mentioned, where $\mathcal{L}_{L1}(G)$ states the L1 loss (absolute distance) of the real image with the generated image $G(x,z)$. Apart from fooling the discriminator D, it adds one more requirement that the generated image by its framework should be identical to the original image. To avoid blurring, L1 is preferable than L2[10].

Then then final minmax objective is as follow:

$$\arg\min_G\max_D\mathcal{L}_{cGAN}(G,D)+\lambda\mathcal{L}_{L1}(G), \tag{7}$$

where λ is a learned parameter.

Finally, it is worth mentioning that pix2pix applies the "U-Net" [9] in the network. In the original network, there is a down-sample (to compress the feature) and up-sample (to decompress it) part in the network. In U-net, there are some skip-connections between the encoder and decoder, in order to preserve the structural integrity and reduce distortion of the image.

6.3. Cycle GANs

Cycle GANs[3] is a GAN model aims to transfer different style and property. As shown in Figure 6 [11], the idea of Cycle GANs is to let the image A (a horse) to transform as image Generated_B (a zebra) by Generator G ($G:A\rightarrow B$, denoted as GeneratorA2B in Figure 6) with Discriminator B (D_B) distinguishing B and Generated_ (a real zebra or not). The "cycle" part takes place when Generator F ($F:B\rightarrow A$, denoted as GeneratorB2A in Figure 6) transforms back from Generated_B to Cycle_A with Discriminator A distinguishing A and Cycle_A (a real horse

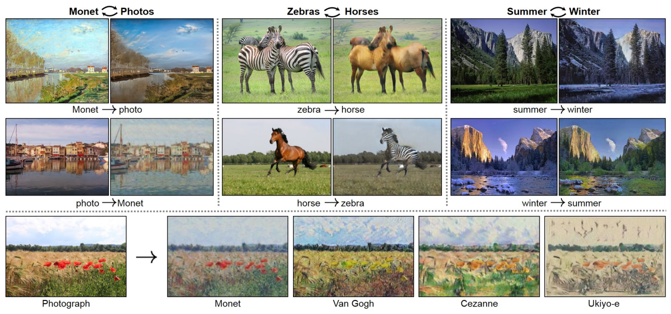

Figure 5. Image translation by Cycle GANs from the paper

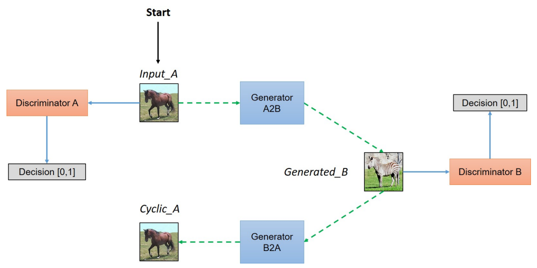

Figure 6. The network illustration of Cycle-GANs

or not). There is another network starting from B under the same methodology. The loss function is as follow: [3]

$$\begin{aligned}\mathcal{L}_{\mathrm{GAN}}\left(G, D_B, A, B\right) = &\ \mathbb{E}_{b \sim p_{\text{data}}(b)}\left[\log D_B(b)\right] \\ &+ \mathbb{E}_{a \sim p_{\text{data}}(a)}\left[\log\left(1 - D_B(G(x))\right)\right] \\ \mathcal{L}_{\text{cyc}}(G, F) = &\ \mathbb{E}_{a \sim p_{\text{data}}(a)}\left[\|F(G(a)) - a\|_1\right] \\ &+ \mathbb{E}_{y \sim p_{\text{data}}(b)}\left[\|G(F(b)) - b\|_1\right]\end{aligned}$$

Where $\mathcal{L}_{\mathrm{GAN}}\left(G, D_B, A, B\right)$ is the original GANs loss from image A to B. $\mathcal{L}_{\text{cyc}}(G, F)$ is the cyclic loss from image A to Cycle_A, where the loss is measured under L1 distance. The L1 loss makes the change the generating image as less as possible from the original one. (The reason of using L1 instead of L2 is mentioned in last section 6.2.)

The total loss is:

$$\begin{aligned}\mathcal{L}(G, F, D_A, D_B) = &\ \mathcal{L}_{\mathrm{GAN}}(G, D_B, A, B) \\ &+ \mathcal{L}_{\mathrm{GAN}}(F, D_A, B, A) \\ &+ \mathcal{L}_{\text{cyc}}(G, F)\end{aligned}$$

The optimization is still a minmax game:

$$\arg\min_{G,F}\max_{D_F, D_B} \mathcal{L}\left(G, F, D_A, D_B\right) \tag{8}$$

To compare with pix2pix, cycle GANs is more flexible sin it does not require paired training data.

7. Conclusion And Future Work

In this paper, we have discussed GANs, a breakthrough in deep learning approach. GANs is still a fast-growing and active research area, such as more stable in training, better objective functions, Conditional GANs, and with many applications: style transfer[6], generating realistic photographs, image-to-image/text-to-image translation[10], human pose generation[7] and fashion swapping[4].

References

[1] Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu,D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In NIPS, 2014.

[2] Goodfellow. Nips 2016 tutorial: Generative adversarial networks. arXiv preprint arXiv:1701.00160, 2016.

[3] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. International Conference on Computer Vision, 2017.

[4] Jetchev N , Bergmann U . The Conditional Analogy GAN: Swapping Fashion Articles on People Images[J]. 2017.

[5] Kingma, Diederik P and Welling, Max. Auto-Encoding Variational Bayes. arXiv preprint arXiv:1312.6114, 2013.

[6] L. A. Gatys, A. S. Ecker, and M. Bethge. Image style transfer using convolutional neural networks. CVPR, 2016.

[7] L. Ma, X. Jia, Q. Sun, B. Schiele, T. Tuytelaars, and L. Van Gool. Pose guided person image generation. In Advances in Neural Information Processing Systems (NIPS), 2017.

[8] M. Mirza and S. Osindero. Conditional generative adversarial nets. arXiv preprint arXiv:1411.1784, 2014.

[9] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. arXiv preprint arXiv:1505.04597, 2015.

[10] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial networks. In CVPR, 2017.

[11] Understanding and Implementing CycleGAN in TensorFlow. (n.d.). Retrieved December 6, 2019, from https://hardikbansal.github.io/CycleGANBlog/.

[12] van den Oord, Aaron, Kalchbrenner, Nal, Vinyals, Oriol, Espeholt, Lasse, Graves, Alex, and Kavukcuoglu, Koray. Conditional image generation with PixelCNN decoders. arXiv preprint arXiv:1406.2661, 2016.