Comparison between the performance of GAN and DCGAN

Liu Fangxu
CentraleSupélec
fangxu.liu@student.ecp.fr

Sun Xiaoyu
CentraleSupélec
xiaoyu.sun@student.ecp.fr

Xu Gezheng
CentraleSupélec
gezheng.xu@student.ecp.fr

Tan Gansheng
CentraleSupélec
gansheng.tan@student.ecp.fr

ABSTRACT

Generative Adversarial Networks (GAN) are algorithms that generate content (usually images), which imitate real content that is fed to them. The idea was first introduced in 2014 by Ian Goodfellow and his group of researchers. A GAN consists of a generator, which produces images from randomly initial values, and a discriminator, which tries to distinguish real images from fake ones. In this project, we aim to test and compare the performances of a GAN algorithm and a Deep Convolutional Generative Adversarial Network (DCGAN) algorithm, using multiple datasets. The neural networks in the GAN are implemented using the *keras* package with *tensorflow* as the backend. The DCGAN algorithm is taken from a project on GitHub which focuses on generating human faces with DCGAN. This DCGAN algorithm has been tested on a dataset of anime characters, which is a reproduction of the work of another GitHub user, before being applied to our actual dataset.

The results of GAN and DCGAN on a set of car images have been compared after each algorithm has run for 200 epochs. DCGAN clearly generates images which are closer to the real ones, which is also reflected in the comparison of IS scores. However, its running time is several times of GAN's running time.

KEYWORDS

GANs, DCGANs, Comparison

Figure 1: Generated Anime Characters[5]

Figure 2: GAN style transfer : input images tranfered into the artistic styles of Monet, Van Gogh, Cezanne, and Ukiyo-e[9]

1 INTRODUCTION

GANs have plenty of potential applications. For a start, GANs can be used to generate Anime characters to save manpower in the industry[5], the transformation of a photograph into art of a certain style[9], and the generation of high-resolution images from those of lower resolution. Furthermore, GANs can also offer a new approach in medical research by proposing possible drugs for previously incurable diseases.
The project aims to understand the structure of a GAN algorithm. A GAN consists of a generator, which generates a sample (usually an image), and a discriminator which determines if the sample resembles real samples. In practice, DCGANs are almost always used, which means that both the generator and discriminator contain numerous convoluted layers and activators. In our project, we test the performance of a simple GAN algorithm and a DCGAN algorithm on a set of 5000 car images, as well as a set of 50000 images containing faces of anime characters.

2 PROBLEM DEFINITION

In our project, we shall restrict the scope of the project to the generation of images and the problem can thus be defined as follows:

Given a dataset of images (ex: a set of images of cars) **Use** GAN/DCGAN **To** generate images from the same distribution as the input and **Evaluate and compare** the results of the two algorithms

3 RELATED WORK

All generative models are based on the principle of the maximum likelihood estimation as we try to find the parameters of our model to maximize the likelihood of samples from the dataset. Hense, once we find these optimal parameters, our model is most likely to generate images that are from the same distribution as the real images. However, it is impossible and unrealistic to learn the distribution of each pixel of the images, especially when the images are large and complex. We have to find other ways. Below is a taxonomy of generative models.[2]

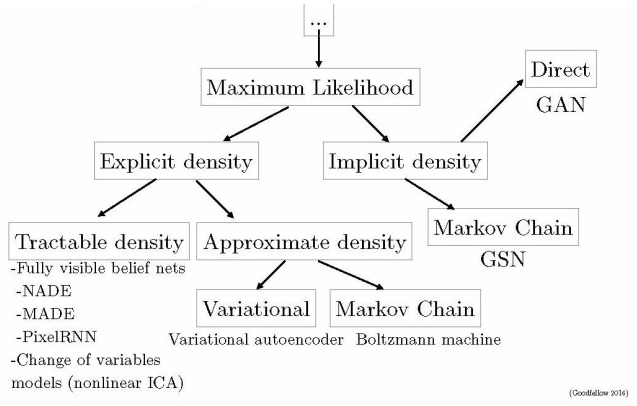

Figure 3: taxonomy of generative models

As the most successful generative model, GAN[3] use implicitly the density. We do not have to construct a Markov Chain to train our model. The principle of GAN will be detailed in the fourth part.

Convolutional neural networks (CNN) is a class of deep neural networks, most commonly applied to analyzing visual imagery. Compared to neural networks, CNN uses one or multiple convolutional layers which help to extract better 2-dimentional features. By applying different convolutional kernels,it is possible to extract different features from the images(ex: horizontal or vertical edges).

DCGAN[7, 1] is a natural combination of CNN and GAN. The core to the DCGAN architecture uses a standard CNN architecture on the discriminative model. For the generator, convolutions are replaced with deconvolutions, so the representation at each layer of the generator is actually successively larger, as it maps from a low-dimensional latent vector onto a high-dimensional image.

Here are the architecture guidelines for stable Deep Convolutional GANs:

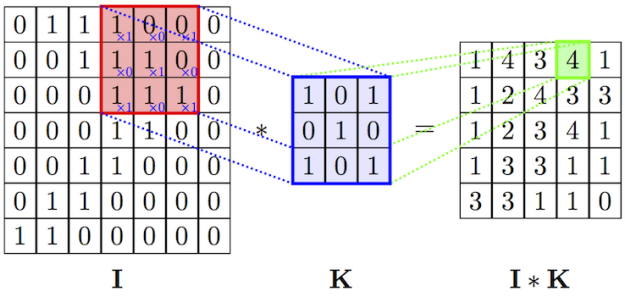

Figure 4: Convolutional operation

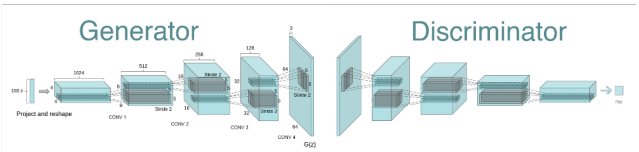

Figure 5: DCGAN architecture

- Replace any pooling layers with strided convolutions (discriminator) and fractional-strided convolutions (generator).
- Use batchnorm in both the generator and the discriminator.
- Remove fully connected hidden layers for deeper architectures.
- Use ReLU activation in generator for all layers except for the output, which uses Tanh.
- Use LeakyReLU activation in the discriminator for all layers.

4 METHODOLOGY

The specific steps that we have taken are summarized below.

(1) As the project mainly aims to understand the structure of a GAN algorithm, we began the project with existing algorithms. We have selected 2 algorithms. The first is a simple GAN algorithm[4] for producing handwritten digits, using images from the MNIST database, whereas the second algorithm is a DCGAN algorithm that has been used for generating new anime characters. The dataset for DCGAN, which contains anime characters, have been collected using a crawler on an anime website, konachan.net, and using a face detection classifier of OpenCV to perform a face clipping. The algorithms mainly involve the construction and training of neural networks.

(2) Find a suitable database of images. In our project, we have selected 5000 car images from the CIFAR-10 dataset collected by Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. The dataset initially contained 50000 images stored as arrays and divided into 10 classes. An

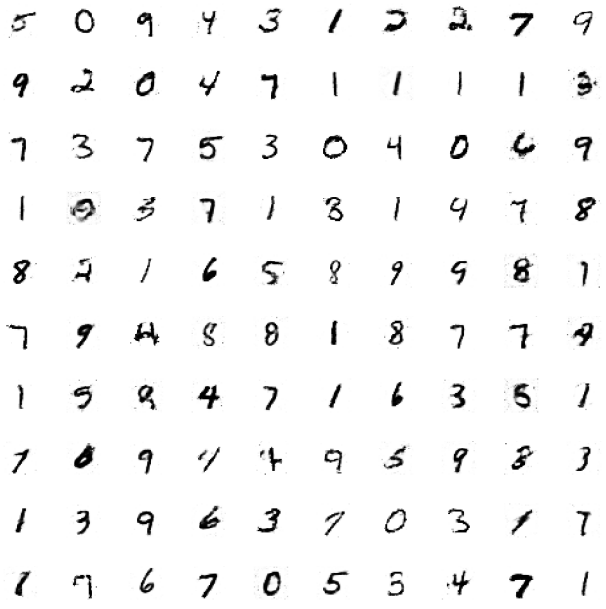

Figure 6: Hand-written digits by GAN after 120 epoches

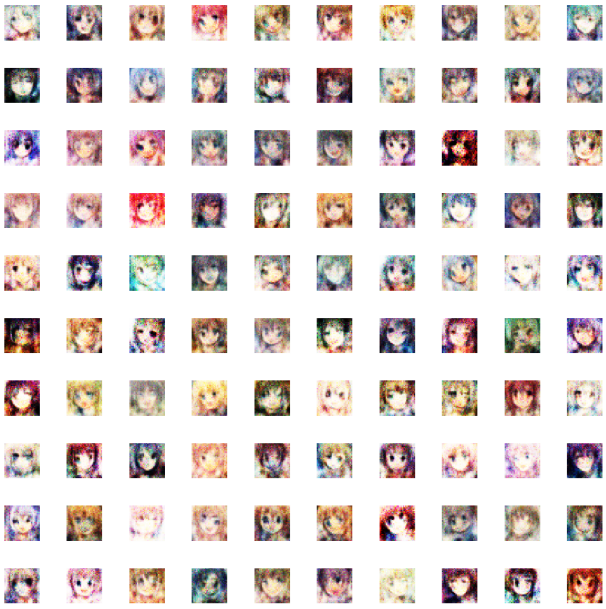

Figure 7: Anime characters by GAN after 40 epoches

algorithm was written to extract images only from the "car" class. In order to adapt to DCGAN, we have also written an algorithm transforming arrays images into JPG form.

Figure 8: Anime characters by DCGAN after 30 epoches

(3) Test the performance of these 2 algorithms on this database. Our main limitation was in terms of computational power. The experiment was performed on two laptops of our team members which do not possess enough CPU power for such problems. As a result, it takes the simple GAN algorithm 2 minutes to train one epoch and generate images of hand-written digits with size 24 x 24. The running time gets even longer when used on images of larger size or when using the DCGAN algorithm, which includes the convolution operation. Surprisingly, the memory is not a limitation since we have been using highly reduced images. We tried to install the GPU support on a machine from the CVN lab but we did not succeed due to its complexity. We regret that because of this high cost of computation, we have not done enough experiment as expected to do a better comparison.
(4) Compare and evaluate our results.

5 EVALUATION

The car images of size 32 x 32 generated using GAN and DCGAN after 200 epochs are shown in 9 and 10 respectively. In the images generated using the GAN algorithm, although we do observe a blurred object resembling a car, the quality of the image is much worse compared to the images generated with the DCGAN algorithm, where specific details of a car can be identified, albeit distorted. However, each epoch of the DCGAN algorithm takes a much longer time to run than the GAN algorithm. The GAN algorithm took about 4 hours for

the 200 epochs whereas the DCGAN algorithm took about a full day.

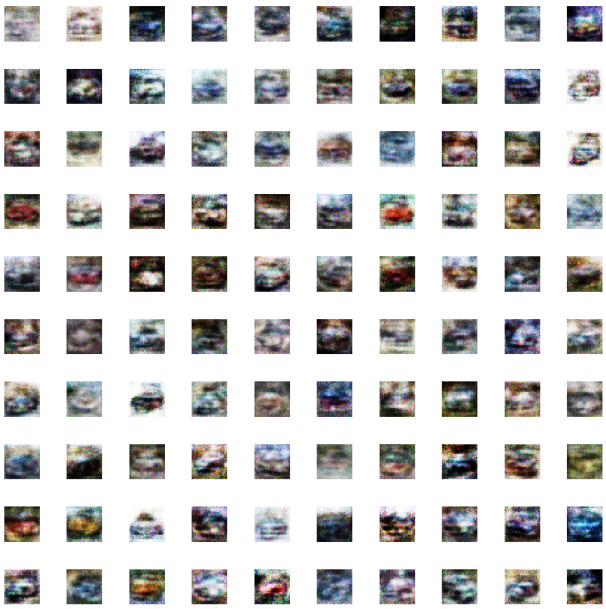

Figure 9: Cars generated by GAN after 200 epoches

Figure 10: Cars generated by DCGAN after 200 epoches

According to [6], we can use Inception Score and Fréchet Inception Distance to compare results from different GAN models of one model with different datasets. Firstly, the inception score uses the quality of the generated images and their diversity to measure the performance. Quality refers to high predictability, which is, given an image, we should know the object type easily. One way to do that is use [8] to predict the conditional probability $p(y|x)$ (X=generated) of the generated images which reflects the quality of the images. Secondly, we will measure the diversity by calculating the marginal probability $p(y) = \int_z p(y|x = G(z))dz$. If the distribution for y is uniform, then the generated images are diverse. Finally, we compute their KL-divergence to compute IS, where

$$\mathrm{IS}(G) = \exp\left(\mathbf{E}_{x\sim p_a} D_{KL}(p(y|x)||p(y))\right)$$

Here we use logistic regression classifier to compare the results between GAN11 and DCGAN12. Firstly, we slice the final output, which consists of multiple images, into individual images and transform them into arrays. Then we plot the figure in excel according to their probabilities. As we can see in 11 and 12, the x-axes are different in the two figures. The DCGAN is less random, that is to say, the generating pictures by DCGAN are more predictable. However, there is no doubt that these two methods are great because most of the picture are located in the right side.

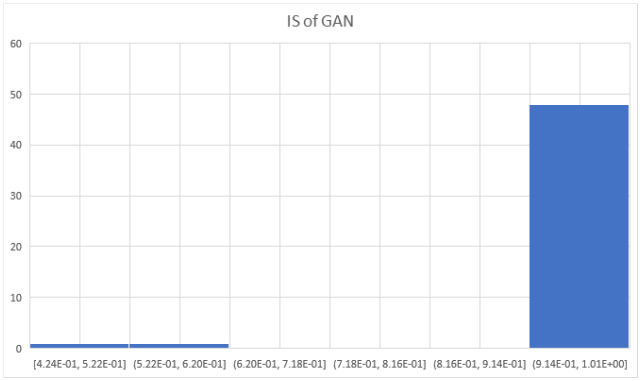

Figure 11: IS score of GAN performance

We can also consider using the concept of Fréchet Inception Distance (FID) to evaluate the performance of a GAN. In calculating the FID, features are extracted from intermediate layers. The values mu and Sigma are then calculated by plotting a distribution from these features. The FID can then be obtained from the following formula:

$$\mathrm{FID}(x, g) = \left\|\mu_x - \mu_g\right\|_2^2 + \mathrm{Tr}(\Sigma_x + \Sigma_g - 2\sqrt{\Sigma_x \Sigma_g})$$

A lower FID value implies better image quality and diversity.

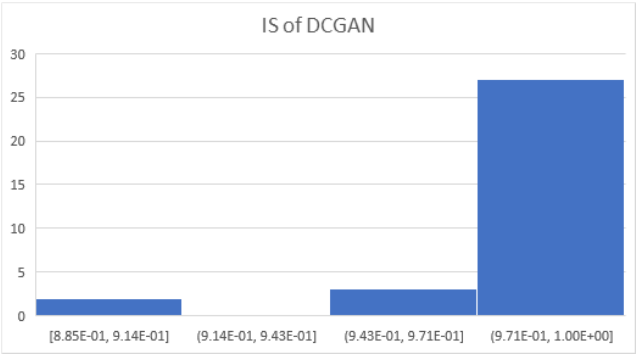

Figure 12: IS score of DCGAN performance

6 CONCLUSIONS

Upon a simple comparison with the naked eye, the DCGAN algorithm evidently produces images of better quality. This result is also supported by the IS scores of the 2 algorithms, where the distribution of DCGAN is much more closer to 1. However, a drawback of the DCGAN algorithm is that the computational time is much longer. We also notice that the quality of generated image highly depends on the quality of real images. Since the car images have worse chop and less cleaner backgrounds than anime images, we did not get the as satisfying results on the car images as that of anime character images even after more epochs of training.

REFERENCES

[1] Realization of dcgan on an anime characters dataset. https://blog.csdn.net/liuxiao214/article/details/74502975.

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

[3] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[4] Stefan Hosein. Demystifying generative adversarial nets (gans). https://www.datacamp.com/community/tutorials/generative-adversarial-networks.

[5] Yanghua Jin, Jiakai Zhang, Minjun Li, Yingtao Tian, Huachun Zhu, and Zhihao Fang. Towards the automatic anime characters creation with generative adversarial networks. *arXiv preprint arXiv:1708.05509*, 2017.

[6] Jonathan. Gan- how to measure gan performance? https://medium.com/@jonathan_hui/gan-how-to-measure-gan-performance-64b988c47732, jun 18,2018.

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

[8] Bharath Raj. A simple guide to the versions of the inception network. https://towardsdatascience.coma-simple-guide-to-the-versions-of-the-inception-network-7fc52b

[9] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Computer Vision (ICCV), 2017 IEEE International Conference on*, 2017.