Using Generative Adversarial Networks (GANs) and Deep Convolutional GANs (DCGANs) for Generating Cat Images

1st Yan Ru Lee
Department of Computer Science
University of National Taiwan Normal University
Taipei, Taiwan
alexlee2511@gmail.com

***Abstract*—Generative Adversarial Networks (GANs) and Deep Convolutional GANs (DCGANs) are two popular deep learning networks that have been used to generate realistic images. In this paper, we will investigate the use of GANs to generate images and compares their performance with DCGANs.**

***Index Terms*—GANs, DCGANs, convolutional layer, image, cat**

I. Introduction

GANs are a type of generative model that consists of two neural networks: a generator and a discriminator. The generator generates fake images, while the discriminator tries to distinguish between the fake images and real images. The two networks are trained together in a game-like manner, with the goal of improving the quality of the generated images over time. GANs have been successfully used for a variety of tasks, including image generation, style transfer, and data augmentation. DCGANs are a type of GAN that use convolutional layers to generate high-quality images. They were first introduced in 2015 and have since become a popular choice for image generation tasks. DCGANs use convolutional layers in both the generator and the discriminator, which allows them to learn features from the image data and generate high-quality images. In this essay, we will discuss the use of these two networks to generate cat images, and compare their pros and cons.

II. Method

A. Prepare the data

To train a GAN to generate cat images, you will need a large dataset of cat images. There are many publicly available datasets that you can use, such as the ImageNet or the Cat Dataset. In this experiment, I used the dataset 'Cats faces 64x64 (For generative models)' from Kaggle. This dataset consists of more than 15,700 cats' face images. Since generative modeling is an unsupervised learning method, hence there are no labels on these images.

B. Build the GAN, DCGAN model

The GANs model consists of two neural networks: a generator and a discriminator. The generator creates new images of cats, while the discriminator tries to distinguish between the generated images and real images. The two networks are trained together in a game-like manner, with the goal of improving the quality of the generated images over time. These two processes is typically repeated for many epochs until the GANs generates high-quality images.

1) Train the discriminator: Initialize the weights of the discriminator network and train it on real and generated images. The discriminator should learn to distinguish between real and fake images.

2) Train the generator: Initialize the weights of the generator network and use it to generate fake images. These images are then fed into the discriminator, and the generator is updated based on the feedback from the discriminator.

- Difference between GANs and DCGANs model: While a standard GAN may use fully connected layers for both the generator and discriminator, a DCGAN replaces these layers with convolutional layers to handle image data more efficiently. This allows the DCGAN to learn hierarchical representations of the input data, which can improve the quality of the generated images.

C. Optimize the networks

To train the GAN, you will need to feed it batches of cat images and update the weights of the generator and discriminator based on the loss function. This process is typically repeated for many epochs until the GAN generates high-quality images.

D. Generate new images

Once the GAN is trained, you can use the generator to create new cat images by feeding it random noise. The generator will use this noise to generate new images of cats that resemble the images in the training dataset.

III. Experiment

We conduct one Experiment referring to two different models, Generative Adversarial Networks (GANs) and Deep Convolutional GANs (DCGANs). In this experiment, we maintain other factors the same in our model, including learning

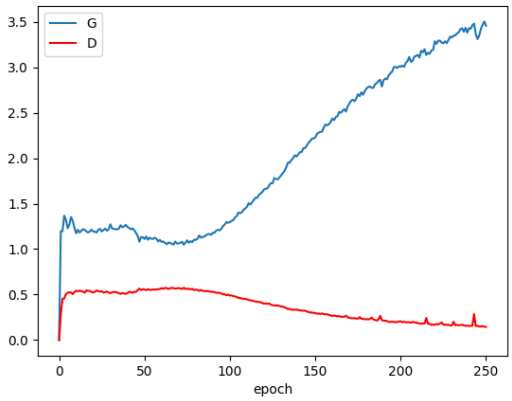

Fig. 1. GAN model - Generator and discriminator loss

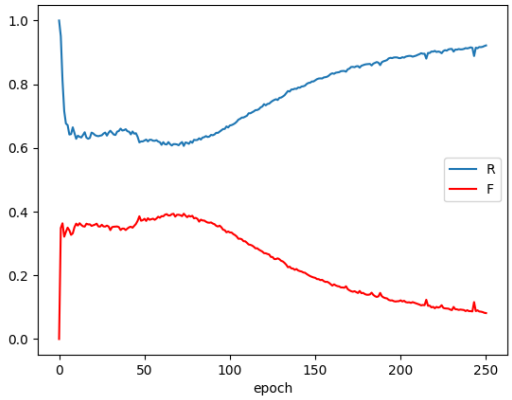

Fig. 2. GAN model - Discriminator Real image's and fake image's score

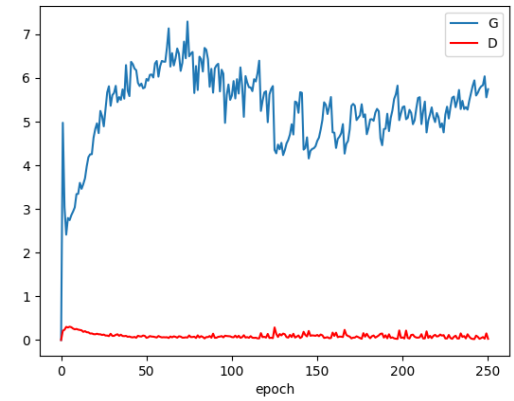

Fig. 3. DCGAN model - Generator and discriminator loss

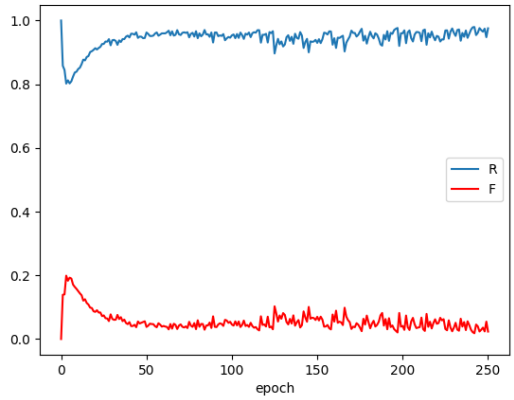

Fig. 4. DCGAN model - Discriminator Real image's and fake image's score

rate at 0.002, batch size at 64, epoch times at 250, Binary Cross Entropy Loss as our loss function, and Adam as our model's optimizer.

A. GAN Model:

In Fig 1, we can see that GAN model's Generator loss increase severely after 100 epochs, while discriminator loss decrease continuously. This may indicate that the model is suffering from a phenomenon called "mode collapse", which the generator produces a limited set of outputs that all look similar, regardless of the input noise. In this case, the discriminator can easily learn to distinguish between the generated images and the real images, causing the generator's loss to increase. In Fig 2, we can see that the real images' score and fake image's score has a sharp turn after 100 epochs, showing the rising performance of the discriminator. However, the generator is not producing diverse outputs, leading to a lack of progress in the learning process.

B. DCGAN Model

In Fig 3, we can see that though DCGAN model's Generator loss oscillate during the training process, the loss function curve eventually converge at an acceptable standard. The discriminator's loss function curve doesn't have any severe increase or decrease during the training, which shows that the model is successful. In Fig 4, we can see that the real images' score and fake image's score is gradually close to 1 and 0 without any rapid changes, showing that the training process works well and the model is successful.

C. Results

In Fig 5, it shows that the images generated by GAN model present the classic result of mode collapse, which produces a limited set of outputs that all look similar. The resolution of the images are much lower and shape of the cat faces are vague. While in Fig 6, the images generated by DCGAN model are much clearer and distinct. Although some of the samples are considered faked due to strange colors, most of the images are close to real cat images and hard to tell the differences. In Table 1, we can see that discriminator loss in both models are low, showing that both model's discriminator perform well. Unexpected, GAN model's generaotor loss are lower than DCGAN model's, though the result show that GAN model works poorly than DCGAN model. This fact indicates

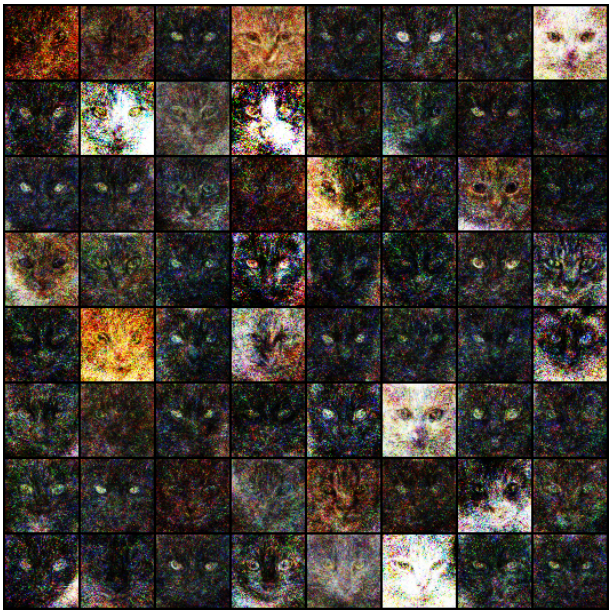

Fig. 5. GAN model - image

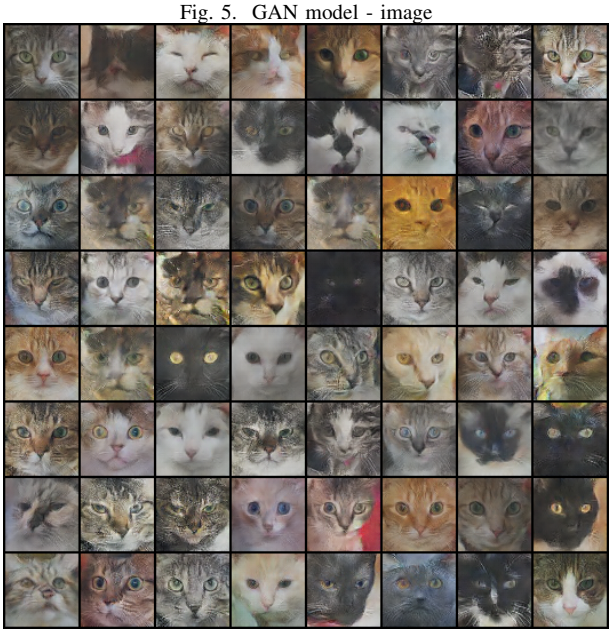

Fig. 6. DCGAN model - image

TABLE I
MODEL LOSS

Model	Generator loss	discriminator loss
GAN	3.220	0.133
DCGAN	6.412	0.008

that loss value during training may not fully expressed the performance of a model, whereas the loss function curve is equally important for analyzing the model.

D. Conclusion

In conclusion, our study demonstrates that DCGANs are a more effective and efficient variant of GANs for generating high-quality images of cats. The use of convolutional layers in both the generator and discriminator allows DCGANs to learn features from the image data and generate more realistic images. Future studies can explore the use of DCGANs for other image generation tasks and compare their performance with other types of generative models. Furthermore, with the experience of comparing GANs and DCGANs models, we can try conducting some methods to detect mode collapse, such as using specific metrics, ”Inception Score” and the ”Frechet Inception Distance” (FID), to improve our GAN model and apply to other various fields of tasks.

REFERENCES

[1] Project-Two-Tutorial.ipynb.
[2] https://chih-sheng-huang821.medium.com/pytorch.
[3] https://towardsdatascience.com/getting-started-with-gans-using-pytorch-78e7c22a14a5.
[4] https://www.topbots.com/step-by-step-implementation-of-gans-part-2/.
[5] https://mofanpy.com/tutorials/machine-learning/gan/gan.
[6] https://www.kaggle.com/code/bunnyyy/gan-implementation-from-scratch-using-pytorch.
[7] https://datascience.stackexchange.com/questions/32671/gan-vs-dcgan-difference.