Replicating Deep Convolution Generative Adversarial Network using existing code

Jiamin He HEJM37@MAIL2.SYSU.EDU.CN

Abstract

In this paper, we replicated the result of Deep Convolution Generative Adversarial Network using an existing code. We trained the neural network on a two core CPU and compared the result during the training steps and other generative model. We also use a nearest neighbor classification algorithm to evaluate the result of the training.

1. Experiments

1.1. MNIST Dataset

The MNIST dataset is a dataset of handwritten digits which is widely used in training image processing machine learning algorithm. It contains 60,000 training images and 10,000 testing images which are always called training set and testing set. As a dataset for classification algorithm, the MNIST dataset is a well sovled dataset, and it is known as a beginning dataset for a lots of image processing algorithm. In this paper, we use the dataset to testify the convolution version of well known GAN, DCGAN. Another reason to choose this dataset is dued to the computation resources.

1.2. Implementation Details

Training setup In the training, we used Adam optimizer with $\beta_1 = 0.5$, $\beta_2 = 0.999$, and a fixed learning rate of 0.0002 for both generator and discriminator. The networks are only trains for 20K steps due to the computing resources limit. In every k step the generator updates $k-1$ times and the discriminator updates one time, here $k = 2$. We train the network with a mini-batch size of 128 using stochastic gradient descent.

Network Architecture The network structure of generator is in Figure 4 while the structure of discriminator is roughly symmetrical with the one dimensional output unit rather than 100 dimensional. Notably, we use fractional-strided convolutions in generator and strided convolutions in discriminator. The activation in generator for all layers are ReLU except the output layer which uess Tanh. And activation in discriminator are all LeakyReLU.

1.3. Compared Training Step

To inspect the learning process of DCGAN, we sampled a few generated image during the training. And we could see the generated image is becoming more confounding with the real MNIST image (Figure 6).

5k steps Figure 1 is 196 images sampled after 5k steps of training. We can see that most the images have had a basic shape of corresponding label.

Figure 1. Images sample after 5k steps of training

10k steps The images sampled after 10k steps of training show that it is easier to identify the digits on the images. But the thickness of the digits vary widely among the images. There is a great amount of digits are only partly visible.

Proceedings of the 33^{rd} International Conference on Machine Learning, New York, NY, USA, 2016. JMLR: W&CP volume 48.

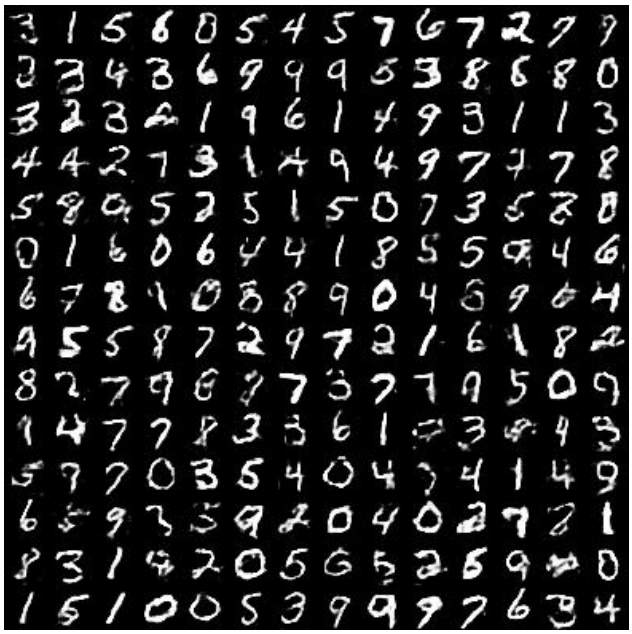

Figure 2. Images sample after 10k steps of training

20k steps After 20k steps of training, the thickness problem improved a lot compared to 10k steps of training. We can see that most of the digits are plump enough as "human written digits". But still, there are some samples that are hard to recognize the digit on it.

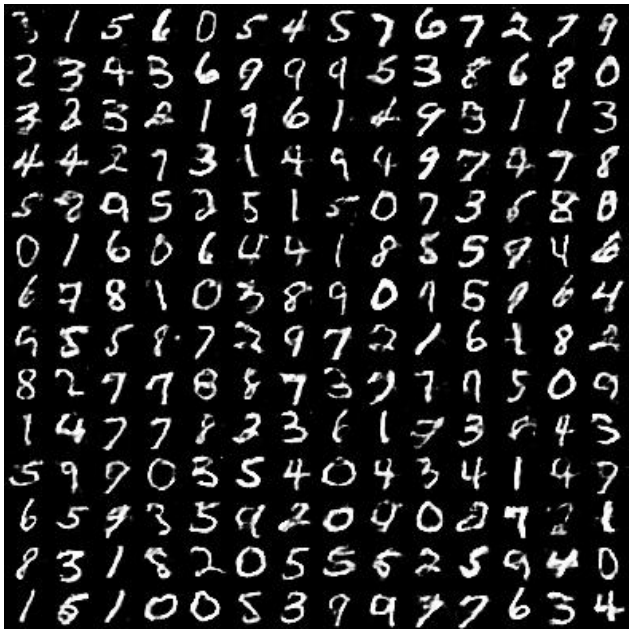

Figure 3. Images sample after 20k steps of training

1.4. Qualitative Results

We compared our result with groundtruth MNIST, a baselines GAN and the origin paper (Radford et al., 2016). From Figure 5 we can see that lacking of training, our result is still far behind the origin paper. The digit in origin paper are so well that it is hard to differ from the groundtruth MNIST. But we can also found that the digits generated by us are more plump and clear than that generated by GAN.

1.5. Quantitative Results

The mission of GAN is to train a network that the distribution of its output is simular to a given distribution. In this paper, we use the MNIST as the given distribution. We want the network to produce a distribution simular to the given data set. So same as the origin paper (Radford et al., 2016), we apply a standard classification metrics to evaluating the conditional distributions learned. We sampled 10192 images from our model (trained for 20k steps) and evaluated the models using k nearest neighbor classifier comparing real data to a set of generated conditional samples. We could see that only train the model for 20k steps is still not enough.

Method	Test Error (50K)	Test Error (10K)
Real Data	3.1%	-
GAN	6.28%	-
DCGAN (origin)	2.98%	-
Real Data (ours)	-	3.22%
DCGAN (ours)	-	5.19%

Table 1. Nearest neighbor classification results

Acknowledgments and Anouncement

We would like to thank Austin FitzGerald. This paper is creating by the author in a course in SYSU which aims to teach the students to read and write scientific paper. The knowledge and skill in the paper does not belong to the author, if anything used in this paper does not contain a proper reference, the author would be pleased to receive an notification email and get things correct.

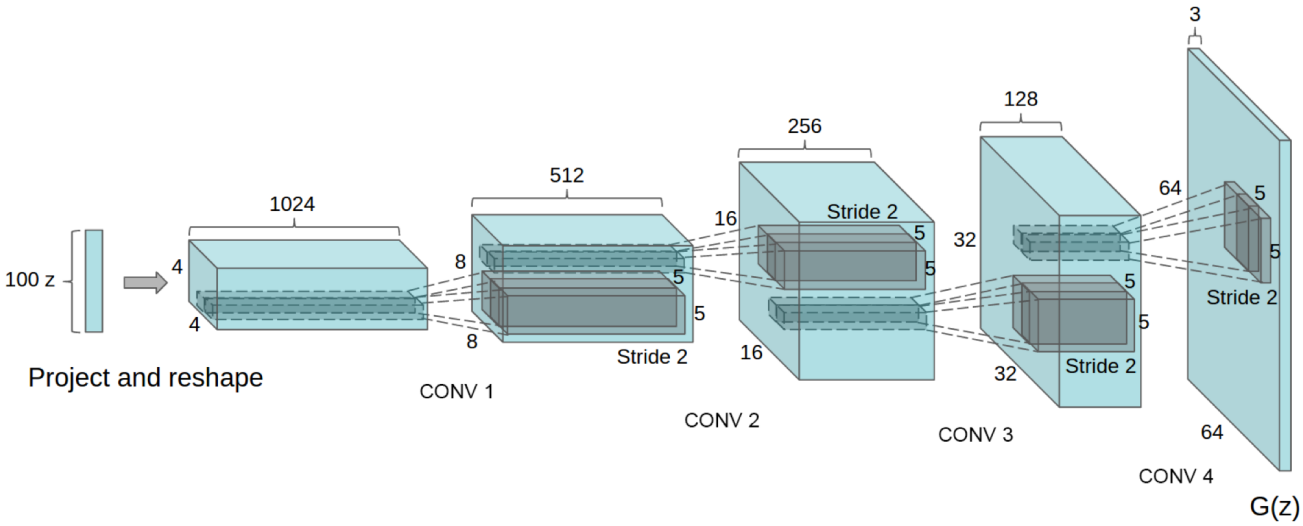

Figure 4. A 100 dimensional uniform distribution Z is projected to a small extent convolutional representation with many feature maps. A series of four fractionally-strided convolutions then convert this high level representation into a 64 x 64 pixel image. Notably, no fully connected or pooling layers are used.

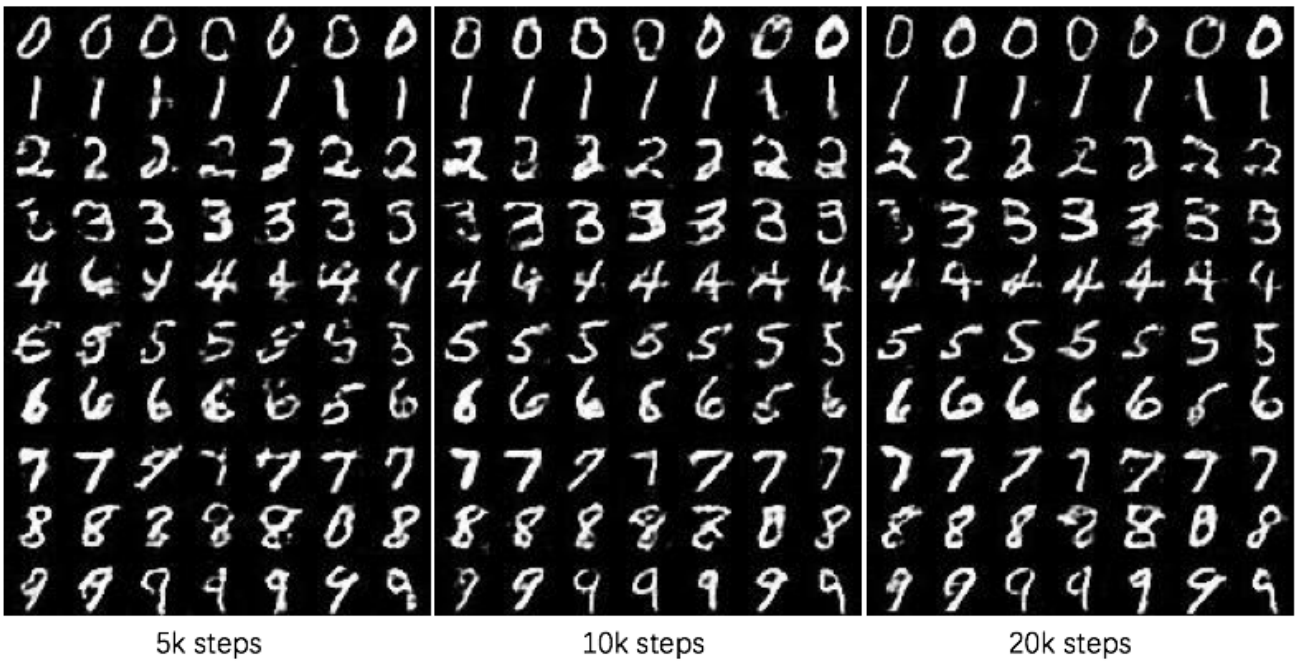

Figure 5. Side-by-side illustration of (from left-to-right) generations after 5k steps, generations after 10k steps and generations after 20k steps.

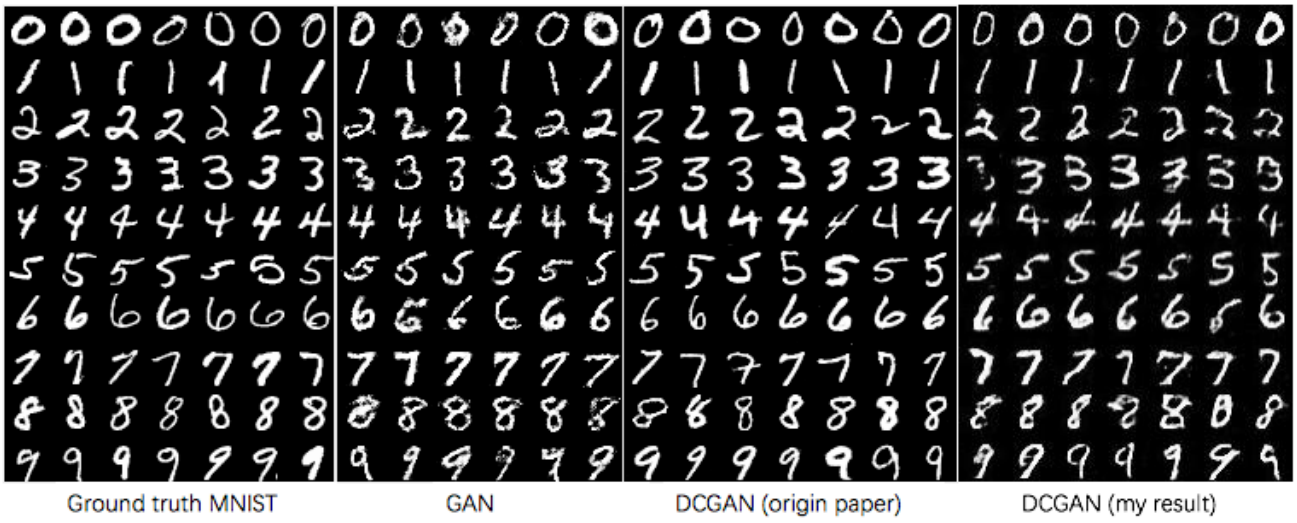

Figure 6. Side-by-side illustration of (from left-to-right) the MNIST dataset, generations from a baselines GAN, generations from origin DCGAN (Radford et al., 2016) paper and generations of our implementation.

References

Radford, Metz, Chintala. Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks *arXiv preprint arXiv:1511.06434*, 2015.

Yihui. GAN-MNIST URL https://github.com/yihui-he/GAN-MNIST/tree/master/mnist