Image Generation Using Generative Adversarial Networks

Amogh Chakkarwar
chakkarwar.a@husky.neu.edu
CSYE7245(1)

Mohit Arya
arya.mo@husky.neu.edu
CSYE7245(1)

Shashank Jain
jain.shash@husky.neu.edu
INFO7390(4)

Spring 2018, Northeastern University

Abstract-- Research Endeavors on realistic image generation have shown excellent results by adopting Generative Adversarial Networks, a framework corresponding to a two-player duel between a generative network (G) and a discriminative network (D). This framework is being known to work well in producing images, in which D acts as binary classifier discriminating between real and fake images. In this project we are exploring GAN and DCGAN's (Deep Convolution Generative Adversarial Network) capabilities in learning set of high dimensional features to generate images from MNIST Dataset. Our experiments demonstrate that both the model can generate good images up to an extent, but DCGAN turn out to be a winner in long run.

Keywords— GAN, DCGAN, Image Generation

I. INTRODUCTION

Generating realistic-looking images has been an enduring goal in machine learning domain. Earlier, it was believed that a generative model was as good as the degree to which it understands the visual world and hence, as a result, image generation was only used as a diagnostic tool. Due to the recent immense quality improvements over the last few years and the successes of discriminative modeling overall, image generation has become a goal on its own, with industrial applications within close reach.

II. BACKGROUND

Generative adversarial network (GAN) was introduced by Ian Goodfellow and his colleagues in 2014. GANs are a class of artificial intelligence algorithms used in unsupervised machine learning, implemented by a system of two neural networks. These two networks play against each other while co-training through the plain old backpropagation.

Despite of the success of Generative Adversarial Networks (GANs) for image generation tasks, the trade-off between image diversity and visual quality is a well-known issue. Conventional techniques achieve either visual quality or image diversity; the improvement in one side is often the result of sacrificing the degradation in the other side. Although GANs have been successful in the field of image generation, the instability of training processes such as extreme sensitivity of a network structure and parameter tuning are well-known disadvantages. This motivated us to learn more about the issues and ultimately find out better techniques for generate images using GAN.

III. IMAGE GENERATION METHODOLOGY

Since the task is to produce an image (found from database or generated), we employ simple GAN and DCGAN model that outputs images similar to the images found in our intended classes.

We have used the MNIST database of handwritten digits, available from below mentioned page, has a training set of 60,000 examples, and a test set of 10,000 examples. It is a subset of a larger set available from NIST. The digits have been size-normalized and centered in a fixed-size image.

Name	Number of Samples	Image Size
MNIST	Training set:60000 Test set: 10000	28*28*1 Source: http://yann.lecun.com/exdb/mnist/

Table 1: Dataset details

GANs are a class of artificial intelligence algorithms used in unsupervised machine learning, implemented by a system of two neural networks. These two networks play against each other while co-training through the plain old backpropagation. The two networks, viz., a generator network and a discriminator network contest with each other in a zero-sum game framework.

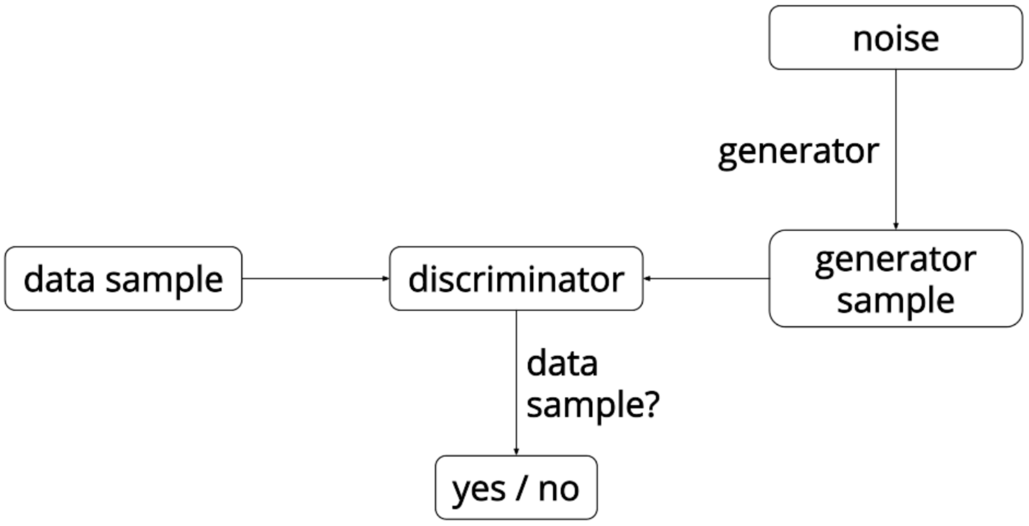

Fig. (1) GAN

Here, the discriminator tries to distinguish the real example, sampled from ground-truth images, from the samples generated by the generator. On the other hand, the generator tries to produce realistic samples that the discriminator is unable to distinguish from the ground-truth samples. This idea can be described as an adversarial loss that applied to both generator and discriminator in the actual training process, which effectively encourages outputs of the generator to be similar to the original data distribution.

Similar to GAN, the DCGAN is an architecture for learning to generate new content and just like GAN, it consists of a generator and discriminator. The only difference here is that DCGAN uses convolutional layers in both the networks, generator and discriminator.

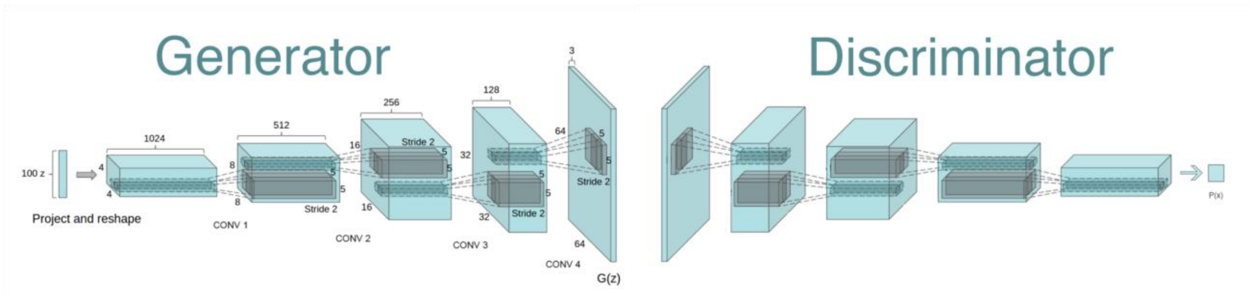

Fig (2). DCGAN

The first layer in the Generator is a fully connected layer which is reshaped into a deep and narrow layer. Then we use batch normalization and a leaky ReLU activation. Next is a transposed convolution where we would halve the depth and double the width and height of the previous layer. Again, we use batch normalization and leaky ReLU. For each of these layers, the general scheme is Convolution > Batch Normalization > Leaky ReLU.

Discriminator is basically just a convolutional classifier. Use batch normalization on each layer except the first convolutional and output layers. Again, each layer looks something like Convolution > Batch Normalization > Leaky ReLU.

Hyperparameters DCGAN:

Generator:

Operation	**Kernel**	**Strides**	**Feature Maps**	**Batch Normalization**	**Nonlinearity**
Linear	N/A	N/A	128	Y	ReLU
Conv2D	5×5	2×2	64	Y	ReLU
Conv2D	5×5	2×2	1	Y	tanh

Table 2: Generator Hyperparameters for DCGAN (MNIST Dataset)

Discriminator:

Operation	**Kernel**	**Strides**	**Feature Maps**	**Batch Normalization**	**Nonlinearity**
Conv2D	5×5	2×2	64	Y	ELU
Conv2D	5×5	2×2	128	Y	ELU
Linear	N/A	N/A	1	Y	Sigmoid

Table 3: Discriminator Hyperparameters for DCGAN (MNIST Dataset)

Our code with documentation can be found here

IV. RESULTS

GAN:

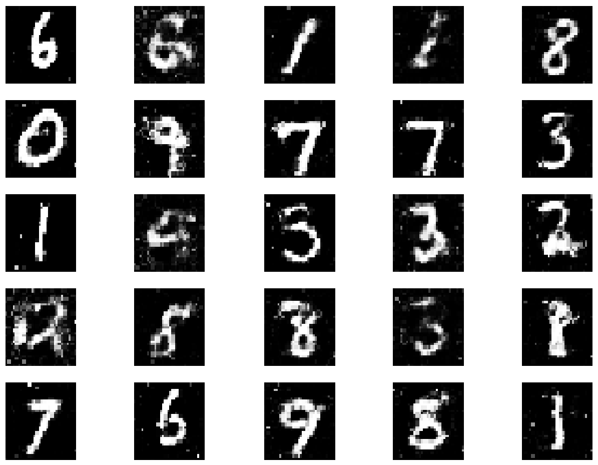

Fig (3). GAN-generated Random Numbers from MNIST, epoch 29800

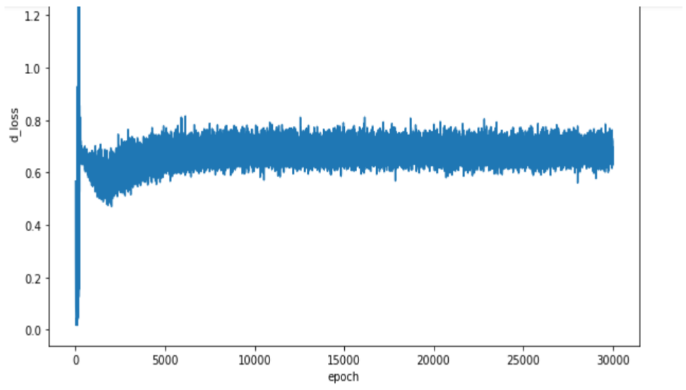

Fig. (4) Discriminator Loss VS Epochs (GAN)

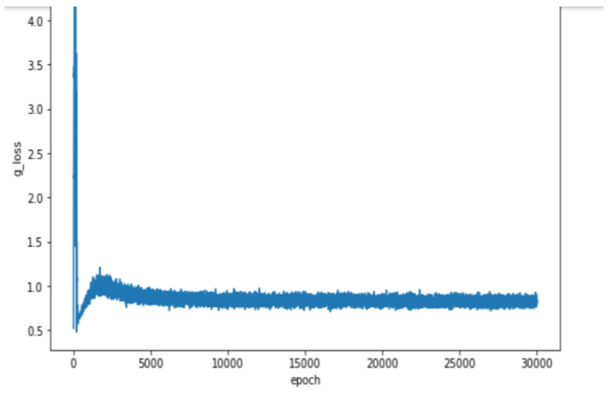

Fig. (5) Generator Loss VS Epochs (GAN)

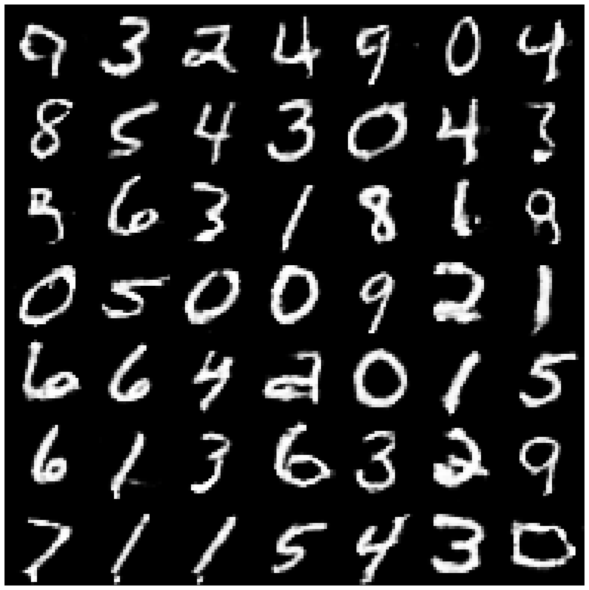

Fig. (6) DCGAN-generated Random Numbers from MNIST, epoch 54th

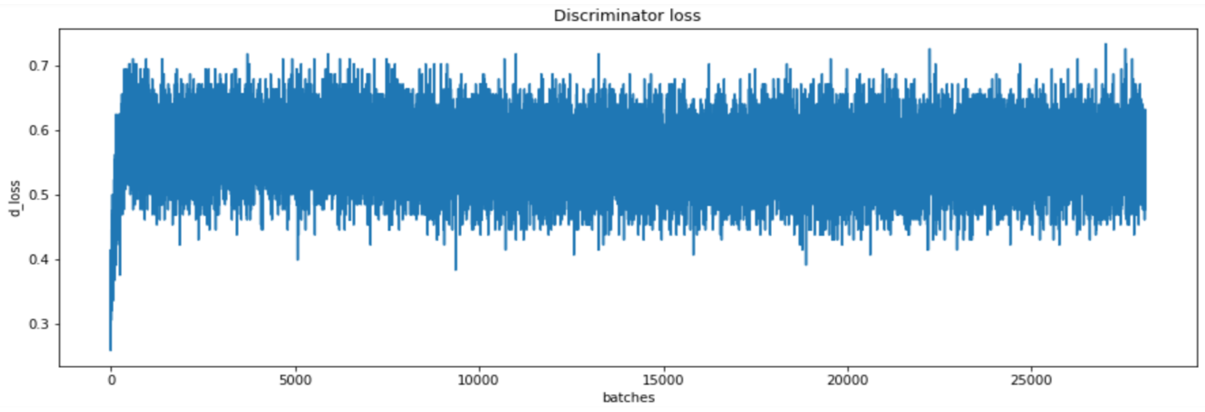

Fig. (7) Discriminator Loss (DCGAN) - 60 Epochs (28020 Batches)

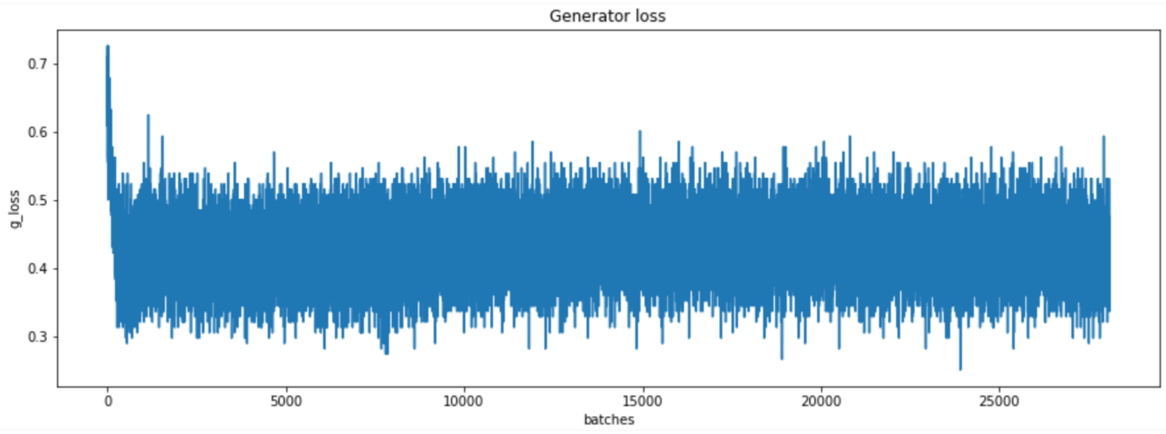

Fig. (8) Generator Loss (DCGAN) - 60 Epochs (28020 Batches)

V. DISCUSSION

The GAN directly passes images through its layers without any preprocessing such as upsampling or downsampling whereas this is the main functionality of the DCGAN. Due to this upsampling of images, the discriminator could critic generator in a better way and helps to generate images with better quality. In DCGAN upsampling and downsampling is done by the deconvolutional and convolutional layers respectively which takes a lot of processing power, consequently taking more time to generate images. But this issue can be resolved by using GPU to make the process quicker.

For GANs the losses are very non-intuitive. Mostly it happens due to the fact that generator and discriminator are competing against each other, hence improvement on the one means the higher loss on the other until this other learn better on the received loss, which breaks its competitor. From Fig. 4, 6 and 7 we can observe that both discriminator and generator losses are converging to some permanent numbers. This loss convergence normally signifies that the GAN model found some optimum, where it can't improve more, which also mean that it has learned well enough. The losses sometimes bounce around a bit which is normal and it's just the evidence of the model trying to improve itself. From Fig. 6 and 7, when the generator gets better at generating better images the loss of discriminator increases.

VI. CONCLUSION

We started working on this topic by referring different research papers and technical articles on GAN and DCGAN, and we learned that the DCGAN generates images with better quality as compared to the vanilla GAN. To test this, we began to learn and understand the detailed architecture and working of GAN and DCGAN, ultimately implementing it on the MNIST dataset. Furthermore, we also studied and compared the results of both the networks using graphs and output images. Since the evaluation of the quality of generated images is subjective, we will let ourselves be human discriminators and make statements on whether the generated images look recognizable and elegant to us. Thus, by comparing results i.e. images generated by both the networks, we observed that the images generated by the DCGAN have overall better quality than generated by GAN.

REFERENCES

1. Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, Yoshua Bengio. Generative Adversarial Nets https://arxiv.org/abs/1406.2661
2. Dixee Kimball, Chenduo Huang, and Karen Yang. Generative Adversarial Networks for Multiclass Image Generation https://web.stanford.edu/class/cs221/2017/restricted/p-final/dkimball/final.pdf
3. https://medium.com/@awjuliani/generative-adversarial-networks-explained-with-a-classic-spongebob-squarepants-episode-54deab2fce39
4. https://gist.github.com/awjuliani/1d21151bc17362bf6738c3dc02f37906
5. https://towardsdatascience.com/gan-by-example-using-keras-on-tensorflow-backend-1a6d515a60d0
6. https://deeplearning4j.org/generative-adversarial-network
7. https://towardsdatascience.com/activation-functions-neural-networks-1cbd9f8d91d6