Image Generation and Transformation with GAN

Yefei Fan; Jiahui Meng; Bing Wu
University of Maryland, Robert H. Smith School of Business

Introduction

Generative Adversarial Network, Generative Adversarial Network, called GAN, is a method of unsupervised machine learning algorithms. Two neural networks contest with each other in a zero-sum game framework. This technique can generate photographs that look at least superficially authentic to human observers, having many realistic characteristics.

Motivation: Our objective is to get deep insights on GAN methods and to seek business values of GAN applications. During the earlier phase, we have learned basic theory about GAN by mimicking and implementing the classic MNIST digit image-to-image example via Python. During later phase, we then researched conditional GAN (CGAN) on Maps dataset to further explore GAN network and dig business insights.

Challenges: The computational power is limited and time costs and risk are high; dataset is not large enough; there is a gap between research and real application.

Methodology

• Highly Effective Platform --- AWS

Two instances in EC2 from Amazon Web Service (AWS), which are more powerful and time-saving.

• Proper Algorithm --- GAN

Generator

- Task: produce a fake image indistinguishable from a real image and confuse the discriminator.
- Goal: generate 'real' images as much as possible to cheat discriminator network
- Objective Function:

maximize V1 = log(D(G(Z(i))))

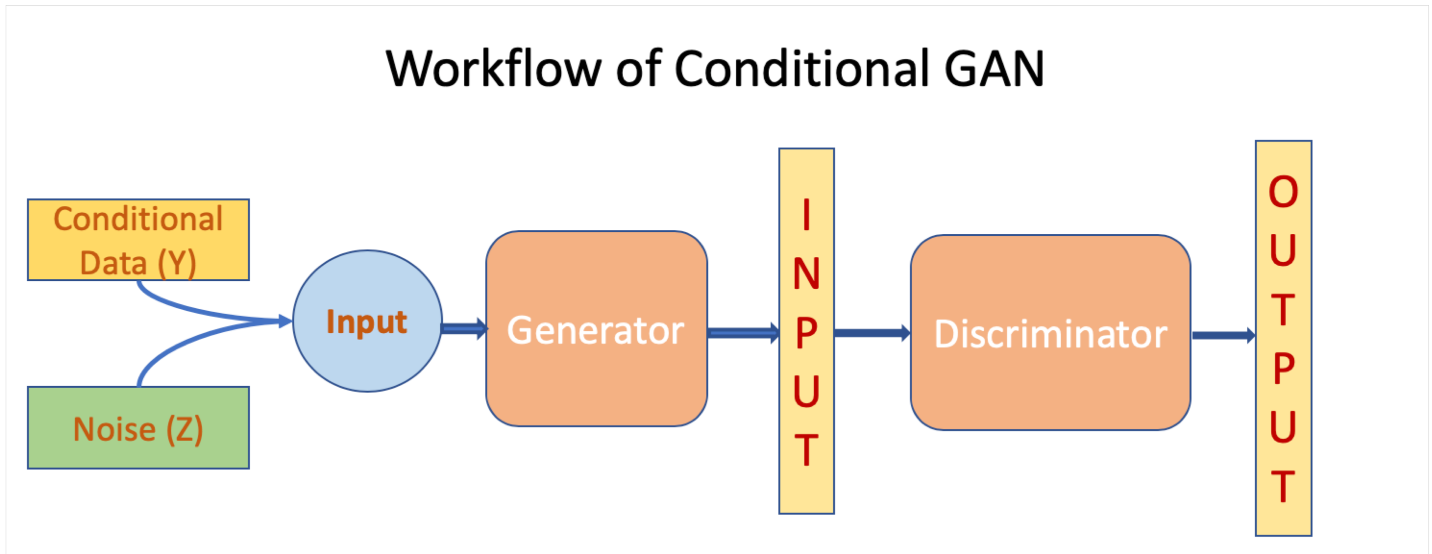
Discriminator

- Task: distinguish between real image and fake image from the generator.
- Goal: try to separate the images generated by generator from the real ones.
- Objective Function:

maximize V = 1/m * sum(log(D(X(i) + log(1 - D(G(Z(i))))

C-GAN

$$\min_G \max_D V(D,G) = \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z}\sim p_{z}(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))]$$



- Noise and conditional data are combined to create a dense code of the output image.
- The dense code is "unsampled" via deconvolution into image space.
- Images are transformed via convolution into a dense code
- *P* (*I* ~ P data | Y)

Figure 1. Conditional GAN.

Literature Review

Pros of GAN (Compared with traditional network)

- GAN seems to produce better samples ;
- No need to follow any kind of factorization to design the model
- GAN takes less time to generate samples.
- GANs generates one sample at a time.
- GANs is asymptotically consistent

Pros of Conditional GAN (Compared with GAN)

- Possible to direct the data generation process;
- Improvement for GAN from unsupervised to supervised methods;
- Add flexible conditions to generate various results;
- Practical methods for transform images from on domain to another domain;
- Learn a loss function to train the mapping from input image to output image;
- Make it possible to apply the same generic approach to problems that traditionally would require very different loss formulations.

Result

Dataset:

• MNIST dataset

42000 digit images in the training set and 28000 digit images in the testing set. Each image is a black-and-white picture and is with 28 pixels in height and 28 pixels in width, for a total of 784 pixels in total.

• Maps dataset

1097 training image pairs, 1098 validation image pairs, and 1098 test image pairs. Each image composed of RGB channels and of size 600 pixels *600 pixels.

Digit Generator:

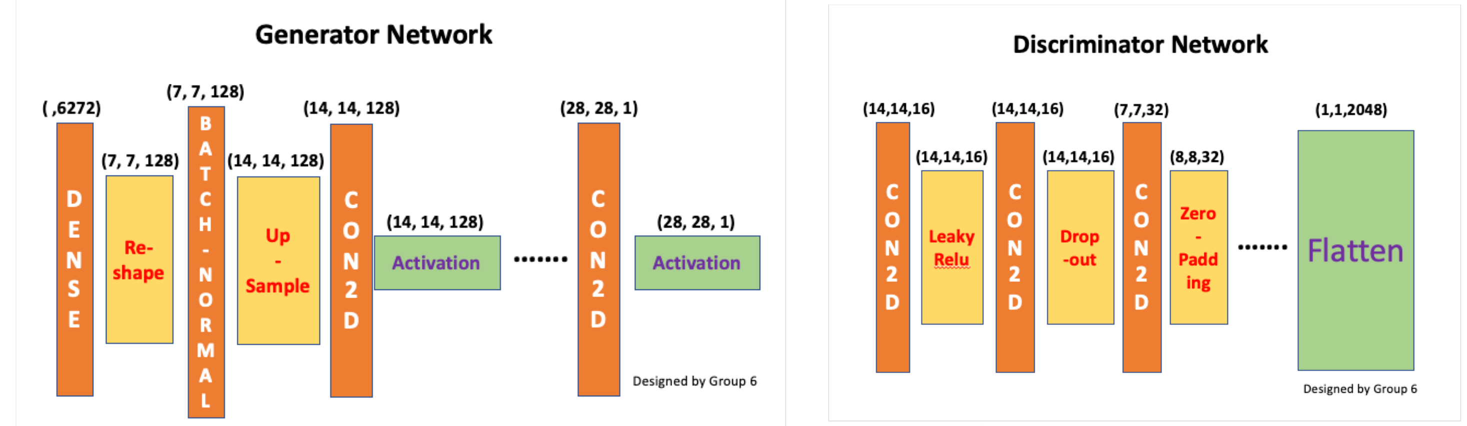


Figure 2. Network Structure.

Epoch 200 Epoch 8000

	Measurement	Index
acc_max	0.890625	291
op_acc_max	1.00	2489
D_loss_min	0.69602346	11511
G_loss_min	0.6542347	7904

Figure 3. Output Summary.

Map Transformation:

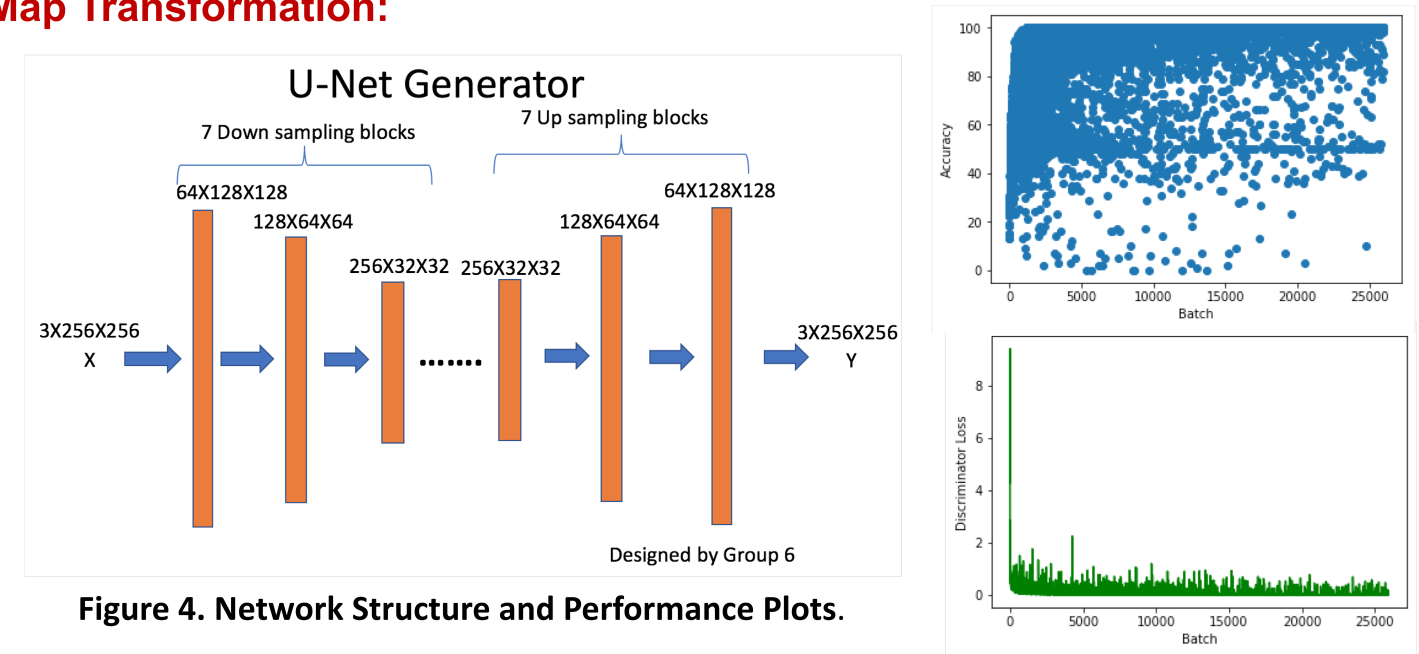


Figure 4. Network Structure and Performance Plots.

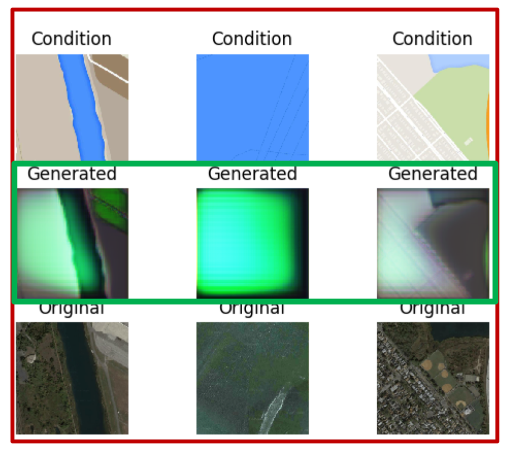


Figure 5. epoch 0.

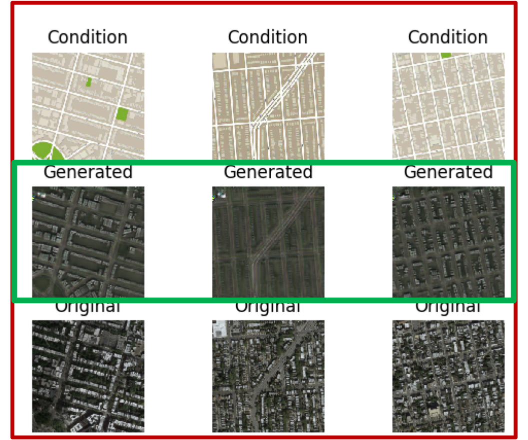


Figure 6. epoch 10.

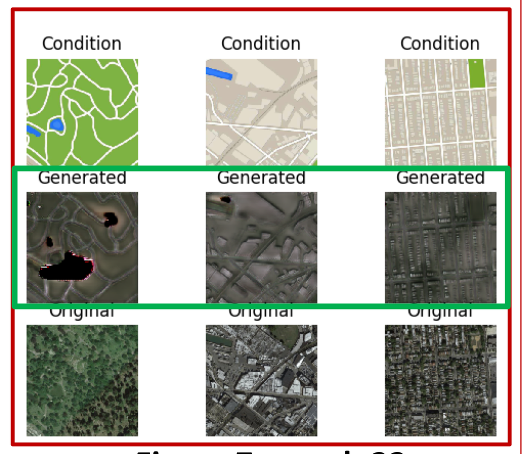


Figure 7. epoch 23.

As we can see from Figure 4, discriminator loss decreases as batches are processed. When the batch reaches 15000th, the loss tend to become very low and closer to 0 and the accuracy can quickly reach to 100%, during each epoch.

We run 23 epochs with 200 batches each. We will choose the three representative outputs for process demonstration. Figure 5 is very blurry so that we cannot find any information from generated images. Figure 6 is better and the maps generated in the middle is recognizable. Figure 7 is clear and the generated maps are quite desirable. We get the highest discriminator accuracy, 100%.

Implications and Limitations

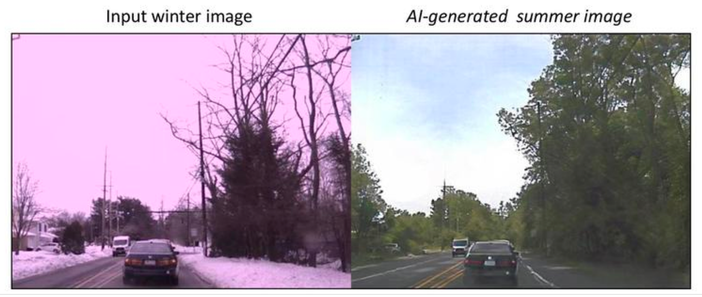
Business values: We think that our model could be applied in several industries. For example:

- **Automotive Industry**

Our project could be implied in automotive industry. GAN method are used for imitation learning and inverse reinforcement learning. This could be applied to studying the road conditions and provide information for human drivers and self-drive cars. Thus, GAN method can be used for generating realistic environments in simulation for training self-driving car policies.

- **Fashion Industry**

Our project can be implied in fashion industry. Apparel companies focus on designing unique and fashionable products for customers. From skirts to coats, hats and handbags and boots, the variety and style are dazzling. Our model can help generate unlimited apparel picture samples, including different shapes, colors, characters, textures, so designers can use them for inspirations.



Conclusions

Generative adversarial networks perform very well at generating simple images such as digits. To give control of the process of generating images, using conditional GAN is an efficient solution for translating image from one visual domain to another. Our U-Net based networks perform well in aerial-to-map translation with a high accuracy.

The applications of **conditional GAN** are not limited to aerial-to-map translation. Conditional-GAN can be applied to translations between other domains of images such as style transformation and change image backgrounds.

However, **limited data resources** is always a problem. We believe our model would perform better with more data. Moreover, with more computation power, we would have chance to try more possible solutions.

Future work includes exploring **residual based network** for both generator and discriminator, and experimenting with dynamic training frequency to allow generator train more often than discriminator in the beginning and gradually slow down.

Contact

Yefei Fan, Master in Business Analytics, yefei.fan@rhsmith.umd.edu
Jiahui Meng, Master in Business Analytics, jiahui.meng@rhsmith.umd.edu
Bing Wu, Master in Business Analytics, bing.wu@rhsmith.umd.edu

References

1. M. Arjovsky, and L. Bottou: Towards Principled Methods for Training Generative Adversarial Networks (2017)
2. M. Arjovsky, S. Chintala, and L. Bottou: Wasserstein GAN (2017)
3. Gulrajani , F. Ahmed, M. Arjovsky, V. Dumoulin, and Aaron Courville: Improved Training of Wasserstein GANs (2017)
4. Ian Goodfellow: NIPS 2016 Tutorial: Generative Adversarial Networks (2016)
5. Luke de Oliveira, Augustus Odena, Christopher Olah, Jonathon Shlens: Conditional Image Synthesis with Auxiliary Classifier GANs 5. Keras ACGAN implementation (2016)
6. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial nets (2014)
7. Radford, A., Metz, L., Chintala, S.: Unsupervised representation learning with deep convolutional generative adversarial networks (2016)
8. Mehdi Mirza, Simon Osindero: Conditional Generative Adversarial Nets (2014)