Asadullah Hill Galib
CSE 891-001: Deep Learning
Programming Assignment 4
December, 15

1 DCGAN

1.1 Padding

Using the following formula,

$$Output = \frac{W - K + 2P}{S} + 1$$

So,

$$P = \frac{S(Output - 1) - W + K}{2}$$

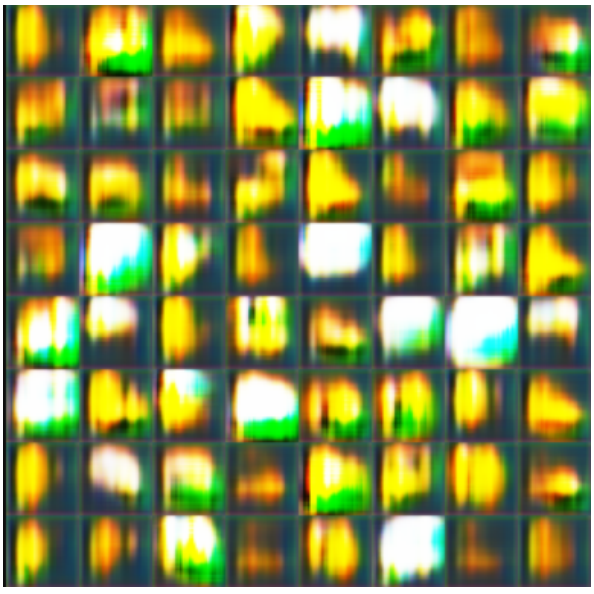
Padding: conv1 - **2**, conv2 - **2**, conv3- **2**, conv4 - **1**.

1.2 Experiments

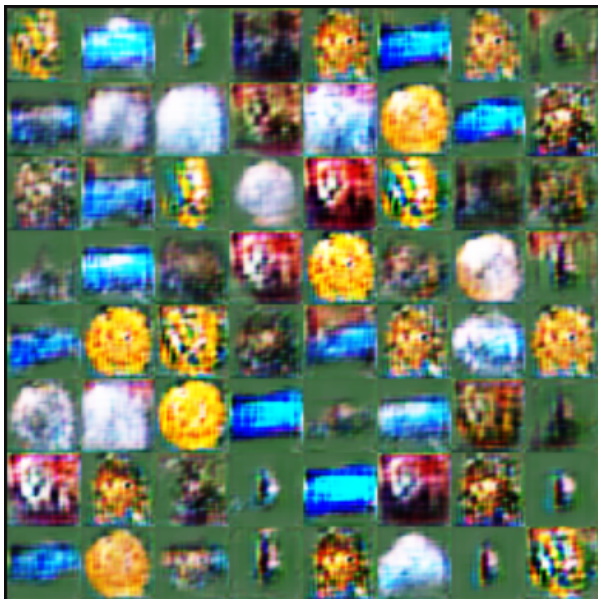
1.2.1 Training DCGAN

It is clear that the generated images quality are far better for 200 epochs than the 25 epochs. Image generation after 25 epochs does not make that much sense.

However, there are some mode collapse images - blue and the yellow repetitive ones.



(a) After 25 epochs

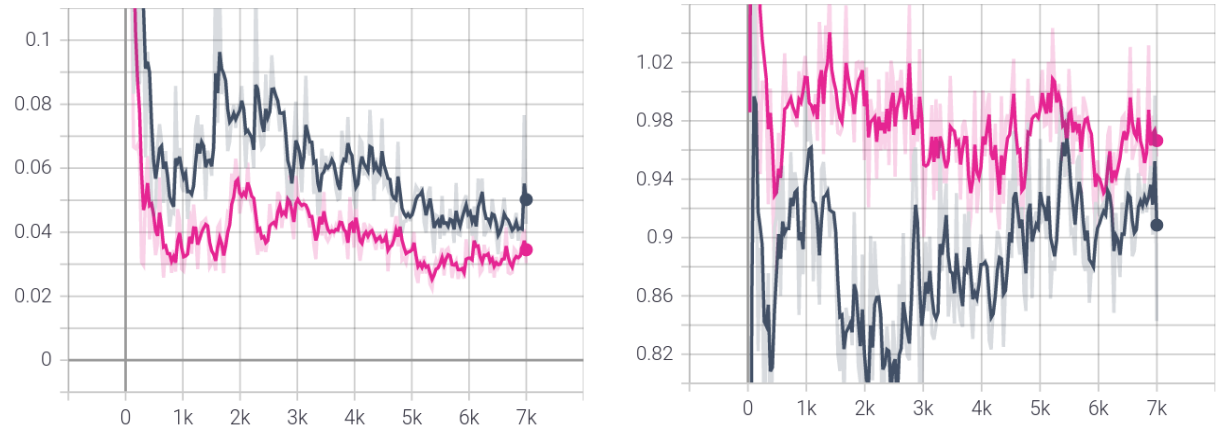


(b) After 200 epochs

Figure 1: Improvement over epochs

1.2.2 Gradient Penalty

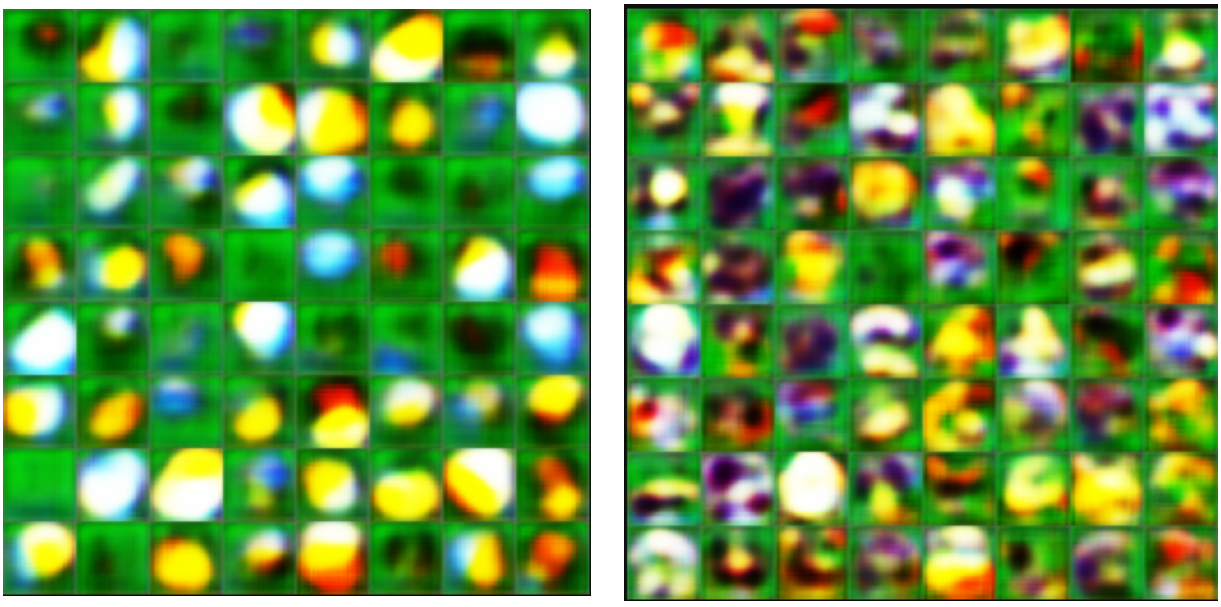
After using the gradient penalty, the training is stabilized. And, there is not any mode collapse issues according to the samples. However, the generated images are quite blurry. The loss curve is quite smoother than the previous one (Fig. 2: without gradient penalty, the curve is more unstable, however the loss is smaller for it.



(a) Discriminant Loss

(b) Generator Loss

Figure 2: Loss without gradient penalty (purple) and with gradient penalty (black)



(a) After 25 epochs

(b) After 200 epochs

Figure 3: With Gradient Penalty

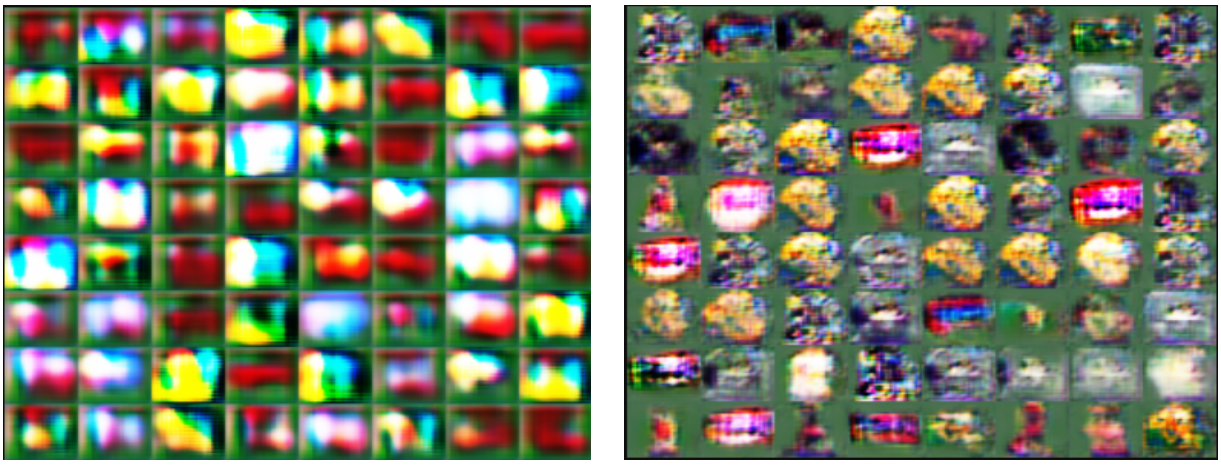
According to Gulrajani et al. [1], a Gradient Penalty is a softer variant of the Lipschitz constraint, which arises from the fact that functions are 1-Lipschitz if their gradients have a maximum norm of 1. The gradient penalty is the squared difference from norm 1. Alexia et al. [2] suggest that using gradient penalties resulted in a large-margin classifier (thus, a large-margin discriminator in GANs). They show how anticipated margin maximization can help

GANs decrease vanishing gradients at fake (fabricated) samples, which is a well-known problem.

1.2.3 Other hyperparameters

The following figures show the effects of spectral norm, learning rate, and d_lr_factor.

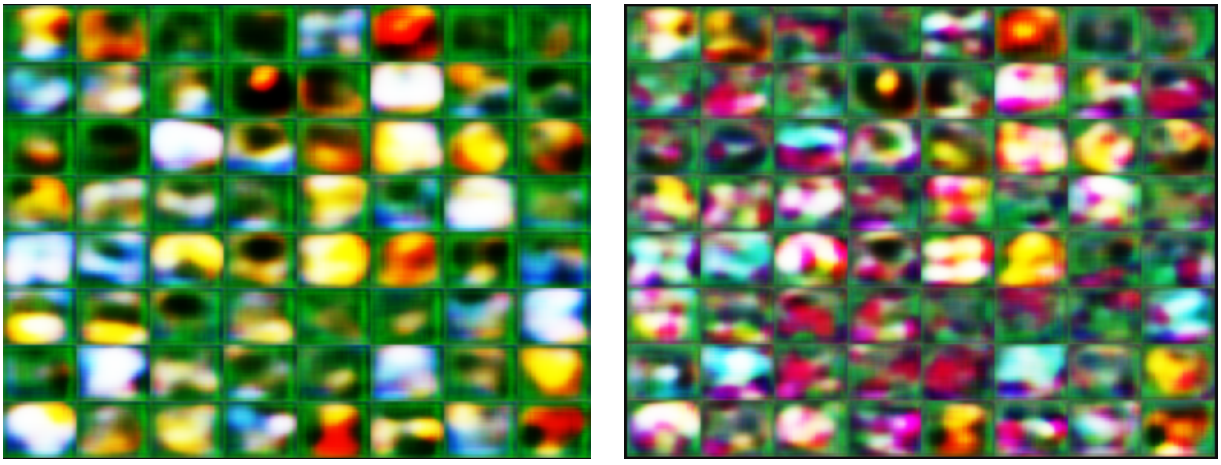
After setting d_lr_factor =5 with default parameters, the images look sharp though there are some mode collapse issues.



(a) After 25 epochs (b) After 200 epochs

Figure 4: with d_lr_factor = 5 and default parameters

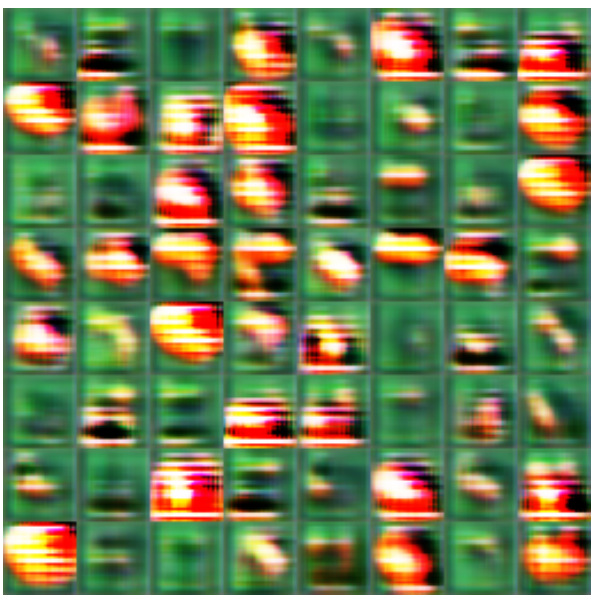
After applying spectral norm only, the images do not look good.



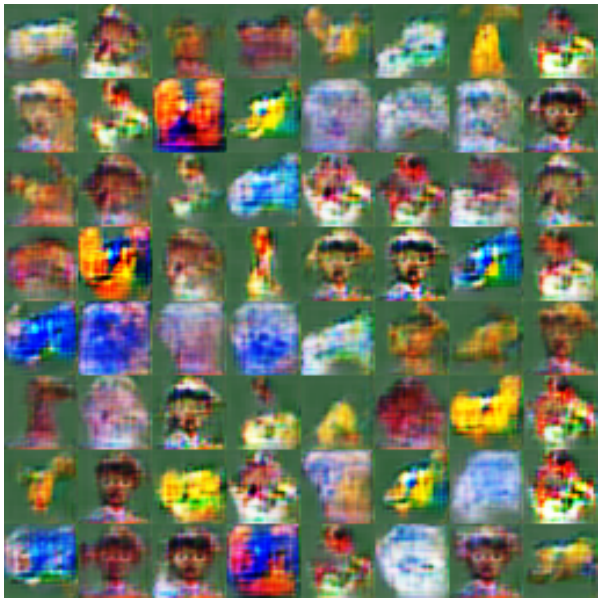
(a) After 25 epochs (b) After 200 epochs

Figure 5: with spectral norm and default parameters

After incorporating spectral norm and setting d_lr_factor = 5 and learning rate = 0.0001, the output images are quite good.



(a) After 25 epochs



(b) After 200 epochs

Figure 6: With spectral norm, d_lr_factor = 5, learning rate = 0.0001

How the measures help: The discriminator's training is stabilized by spectral normalization, which is a weight normalization. It regulates the discriminator's Lipschitz constant to prevent explosive gradients and mode collapse. Decreasing learning rate is often a good hack for performance improvement. It helps to converges but slowly. And, by increasing d_lr_factor, it helps stabilizing the model by giving discriminator more turns to catch fake images by generator per generator turn, so the generator is getting more stable feedback from the discriminator and able to learn and generate more stable outputs.

2 CycleGAN

2.1 Training CycleGAN

(a) X to Y

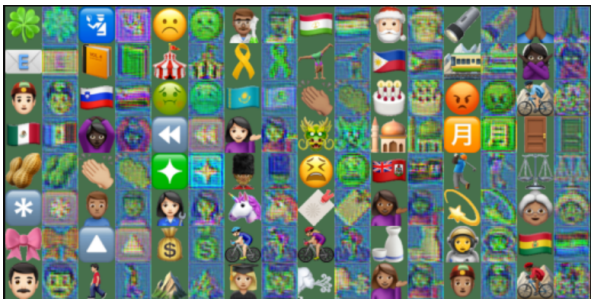
(b) Y to X

Figure 7: CycleGAN outputs after 200 epochs

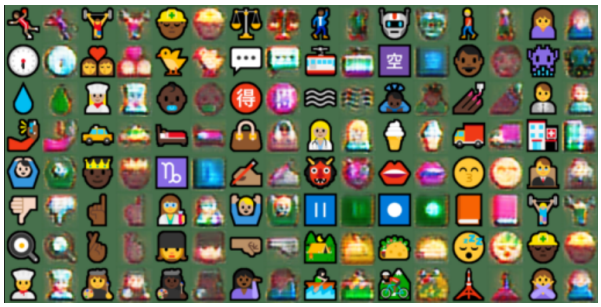
2.2 Different Random Seeds

With random seed 0 and 23, the quality of the generated images are different. It looks like the generations are better for random seed 0. The Y to X generations are quite reddish and X to Y generations are quite purplish for random seed 23. Also, their sharpness is better for random seed 0. I think the color representation and sharpness are the most noticeable differences.

The reason is for different random seed the network as well as the noise initializes differently. Also, training is GAN is different from normal optimization problem: here's generator and discriminator play a minmax game and both try to optimize. So, due to the discrepancy in random seed, the model optimizes differently epoch to epoch.

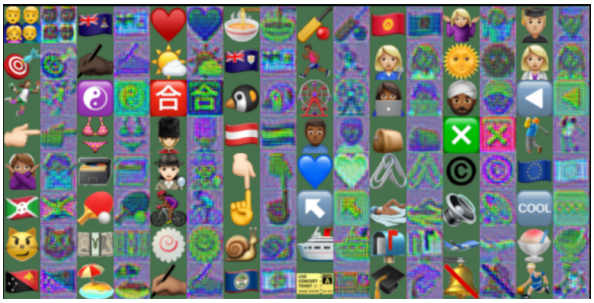


(a) X to Y



(b) Y to X

Figure 8: CycleGAN outputs after 200 epochs with random seed = 0



(a) X to Y

(b) Y to X

Figure 9: CycleGAN outputs after 200 epochs with random seed = 23

2.3 Different lambda_cycle

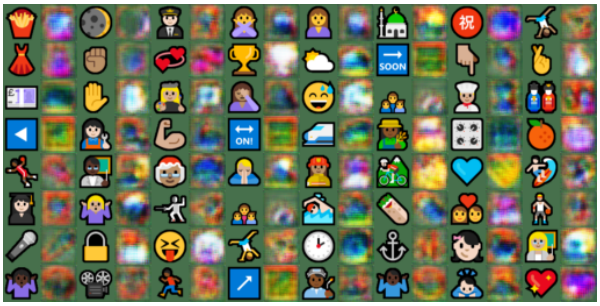
With different lambda_cycle, the results after 25 epochs and 200 epochs are shown in the following figures.

Results without cycle consistency is poorer than the results with cycle consistency. Without cycle consistency, the outputs are quite rectangular in all cases, and less colorful, quite blurry. With incorporating cycle consistency, the outputs look reasonable, more colorful, and sharp.

But, with the cycle consistency values greater than 0, the results do not look much different. I think the results with default value is slightly better.

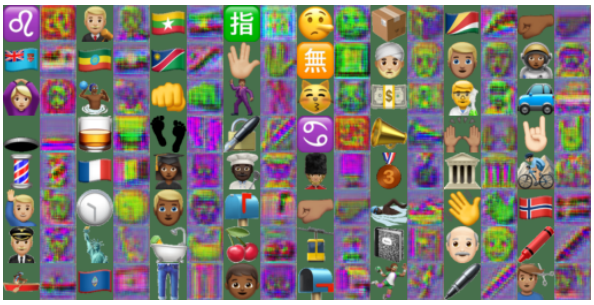
The reason for poor outputs is: without cycle consistency, the model generates quite different outputs from the inputs because the mean squared error between the inputs and outputs is not incorporated in the loss. So, the model generates outputs, which are less related to the original images. By incorporating cycle inconsistency loss, the model can learn to produce relevant outputs to inputs.

(a) X to Y



(b) Y to X

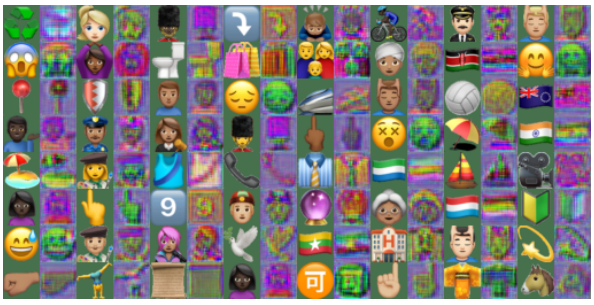
Figure 10: CycleGAN outputs after 25 epochs with lambda_cycle = 0.0



(a) X to Y

(b) Y to X

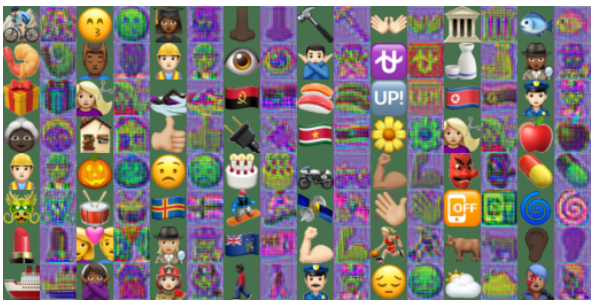
Figure 11: CycleGAN outputs after 200 epochs with lambda_cycle = 0.0



(a) X to Y

(b) Y to X

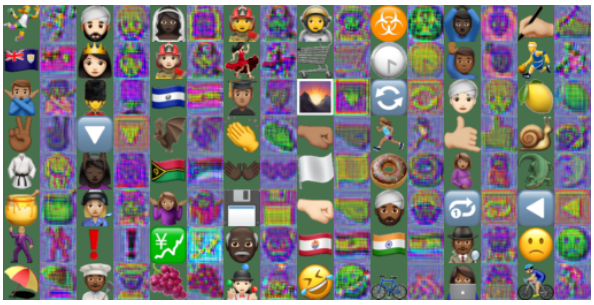
Figure 12: CycleGAN outputs after 25 epochs with lambda_cycle = 0.015



(a) X to Y

(b) Y to X

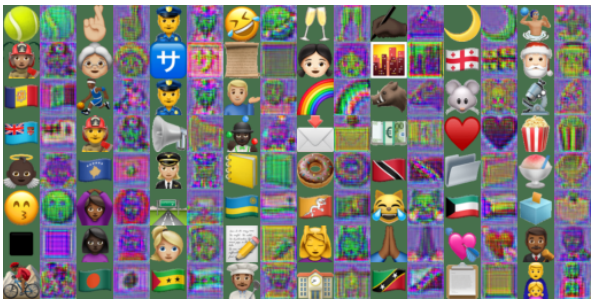
Figure 13: CycleGAN outputs after 200 epochs with lambda_cycle = 0.015



(a) X to Y

(b) Y to X

Figure 14: CycleGAN outputs after 25 epochs with lambda_cycle = 0.3



(a) X to Y

(b) Y to X

Figure 15: CycleGAN outputs after 200 epochs with lambda_cycle = 0.3

References

[1] Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., & Courville, A. (2017). Improved training of wasserstein gans. arXiv preprint arXiv:1704.00028.

[2] Jolicoeur-Martineau, A., & Mitliagkas, I. (2019). Gradient penalty from a maximum margin perspective. arXiv preprint arXiv:1910.06922.