

Image to Image Translation Using CycleGAN

Hinal Shah

California State Polytechnic University, Pomona

hnshah@cpp.edu

Abstract

This project is about implementing Image-to-image translation using Cyclic Generative Adversarial Network (CycleGAN). Image-to-image translation is a problem where model learns the mapping between an input image and an output image using a paired training dataset. But for many tasks paired data is not available. So CycleGAN algorithm trains the model to learn the mapping between source domain and target domain. This project build the model using several datasets, calculate the losses and uses tensorboard for visualizations.

1. Introduction

Cycle GAN is a type of Generative Adversarial Network which is used for cross domain style transfer. It is a method that can learn to do tasks like capturing special characteristics of one image domain and how to translate these characteristics into the other image domain, where both domains are unpaired. The model is trained to do mapping $G: X \rightarrow Y$ such that the output $\hat{y} = G(x)$, $x \in X$, is indistinguishable from images $y \in Y$ by an adversary trained to classify \hat{y} apart from y . So, the objective is to induce an output distribution over \hat{y} that matches the empirical distribution $p_{data(Y)}$. The optimal G thereby translates the domain X to a domain \hat{y} distributed identically to Y . But this translation does not guarantee that there will be any meaningful way of pairing between input x and output y . Therefore, “cycle consistent” property should be defined such that if there is one translator $G: X \rightarrow Y$ and another translator $F: Y \rightarrow X$, then G and F should be inverses of each other. And this can be implemented by training both the mapping functions, G and F simultaneously, and adding a cycle consistency loss and combining this loss with adversarial losses on domains X and Y yields [Figure 1]. Using this method one can translate an image into painting like effect and perform tasks like object transfiguration and photo enhancement to get DSLR photographs.

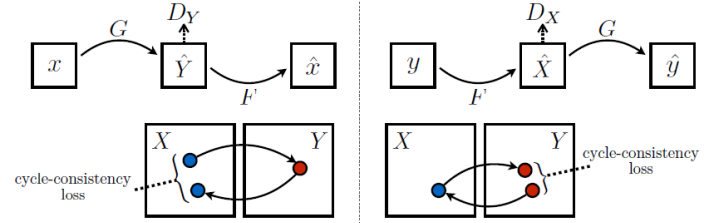


Figure 1

2. Review of Literature

Recent methods such as Neural style transfer which synthesizes image by combining the content of one image with the style of another image, but new styles cannot be tested as it is time consuming whereas this is not the case with CycleGAN. And Pix2Pix depend on the paired data available for training where the same data is available in both domains. The power of CycleGAN lies in learning such transformations without one-to-one mapping between data in source and target domains. In this, the model tries to map the source domain image to target domain and then back to the original image. This way it eliminates the need for a paired image in the target domain. So, the mapping is done using a generator network and the quality of image generated is improved by pitching the generator against a discriminator. The discriminator predicts the input image is original or it is the output of the generator.

3. Methodology

3.1. Model Architecture

To build this model, we need a generator architecture to generate fake image to fool discriminator and discriminator architecture that produces one dimensional output to make decision if the input image is real or fake. Since this model needs to work in both the direction i.e., from source domain \rightarrow target domain and from target domain \rightarrow source domain, it will have two Generators and two Discriminators.

a. Generator Architecture:

The architecture has three components: Encoder, Transformer and Decoder. Encoding is the three 2D convolution network that extracts features of input image and give feature vectors which is then passed to transformer that has nine residual blocks as image size is 256 x 256. For decoding, two layers of deconvolution and instance normalization and ReLU layer is used and tanh as activation function.

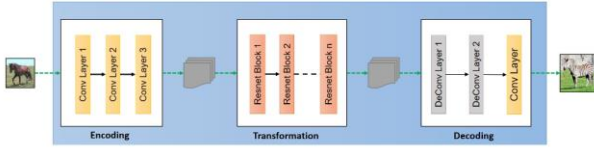


Figure 2. Generator Architecture

b. Discriminator Architecture

To build discriminator, the network consists of five convolutional layers using leaky ReLU and the last layer is to produce 1 dimensional output to decide whether these features are fake or real.

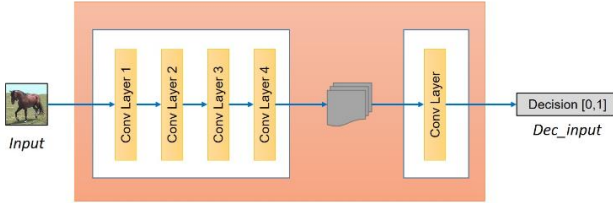


Figure 3. Discriminator Architecture

- c. For optimization, Adam optimizer is used. The cyclic loss and loss of generator and discriminator is calculated.

3.2 Datasets Details

The model is trained with learning rate 0.0002 and slowly decay the rate after epoch 50. Also, in loss calculation, λ was set to 10. The model is trained on several datasets: Sunflower \leftrightarrow Daisy, Horse \leftrightarrow Zebra, Face \leftrightarrow Cartoon, all with images size 256 x 256. Links to datasets are found in Reference section [7].

3.3 Formulation

a. Adversarial Loss

For the mapping function $G: X \rightarrow Y$ and its discriminator D_Y , loss is defined as:

$$\mathcal{L}_{GAN}(G, D_Y, X, Y) = \mathbb{E}_{y \sim p_{data}(y)} [\log D_Y(y)] + \mathbb{E}_{x \sim p_{data}(x)} [\log(1 - D_Y(G(x)))]$$

where G tries to generate images that look similar to domain Y images, while D_Y aims to distinguish between generated samples $G(x)$ and real samples y . So, generator aims to minimize this objective against an adversary D that tries to maximize it. Similarly, adversarial loss is defined for the mapping function $F: Y \rightarrow X$ and its discriminator D_X as well.

$$\text{For } G, \text{ minimize } \mathbb{E}_{x \sim p_{data}(x)} [(D(G(x)) - 1)^2] \\ \text{For } D, \text{ minimize } \mathbb{E}_{y \sim p_{data}(y)} [(D(y) - 1)^2] + \mathbb{E}_{x \sim p_{data}(x)} [D(G(x))^2]$$

b. Cycle consistency Loss

Adversarial training helps to learn mappings G and F and produce outputs identically distributed as target domains Y and X . However, with large enough data, it might happen that a network maps the same set of input images to any random permutation of images in the target domain, where the learned mappings can induce an output distribution that is identical to the target distribution. Thus, adversarial losses alone cannot guarantee that the learned mapping function can translate an individual input image, x_i to a desired output image, y_i . Hence, cycle consistency loss is defined to check if for each image x from domain X , the image translation cycle is succeeded to bring x back to the original image. Cycle consistency Loss is defined as follow for both Generators (G) and (F).

$$\mathcal{L}_{cyc}(G, F) = \mathbb{E}_{x \sim p_{data}(x)} [\|F(G(x)) - x\|_1] + \mathbb{E}_{y \sim p_{data}(y)} [\|G(F(y)) - y\|_1]$$

c. Full Objective

Hence, the final objective function is defined as follows:

$$\mathcal{L}(G, F, D_X, D_Y) = \mathcal{L}_{GAN}(G, D_Y, X, Y) + \mathcal{L}_{GAN}(F, D_X, Y, X) + \lambda \mathcal{L}_{cyc}(G, F)$$

where λ controls the relative importance of the two objectives.

4. Code and Results

4.1 Resources Required

For developing this project, following software and hardware are used:

- a. GPU based PC/Laptop
- b. Python 3.6
- c. Libraries- TensorFlow 1.0, NumPy









4.2 Code Link

<https://github.com/shahhinal/DLPrj>

4.3 Results









Below are some images generated by this model:

a. Datasets Used: Sunflower to Daisy

Input Image	Generated Image
	
	
	
	



b. Datasets Used: Horse to Zebra

Input Image	Generated Image
	
	
	
	



However, there were some failed cases while translating images from one domain to another:

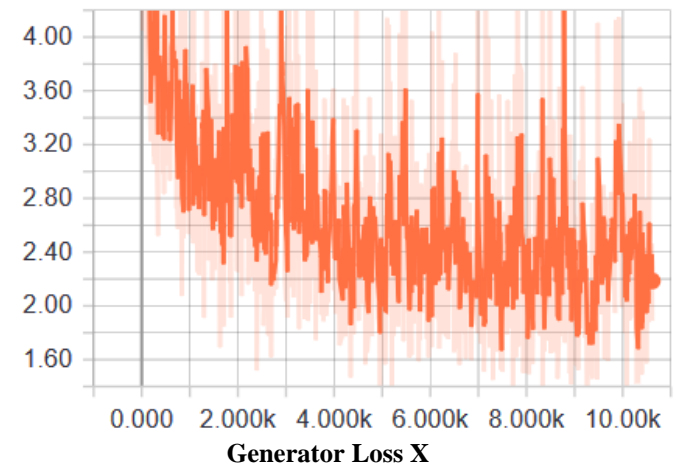
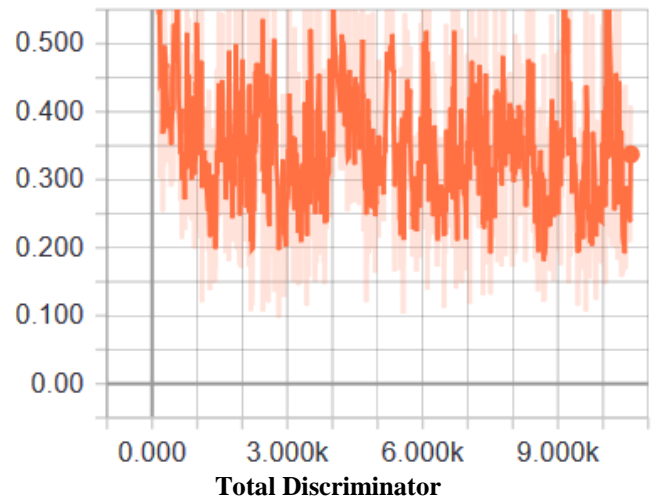
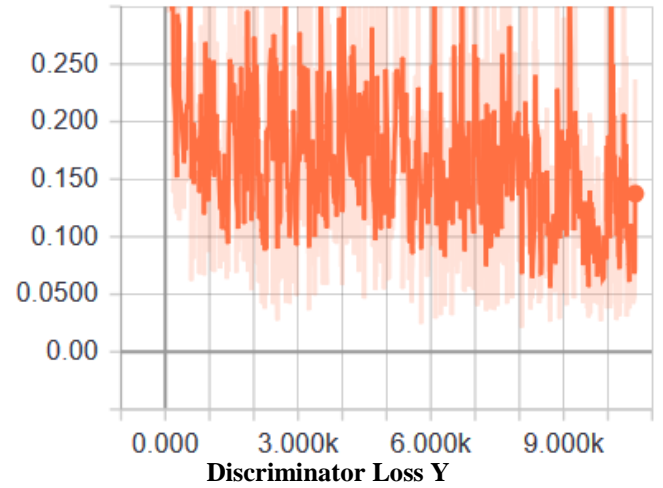
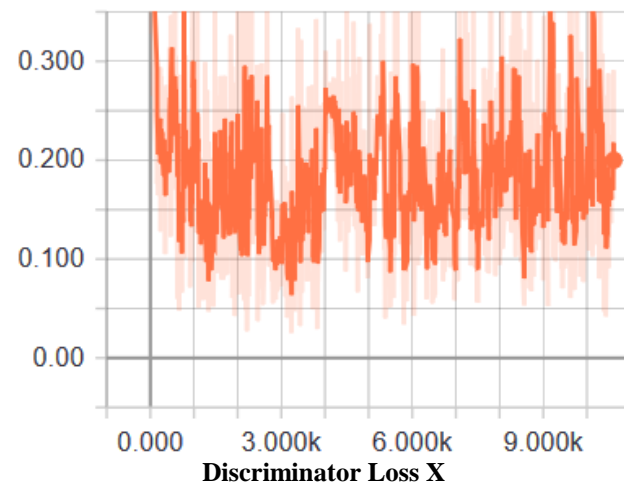
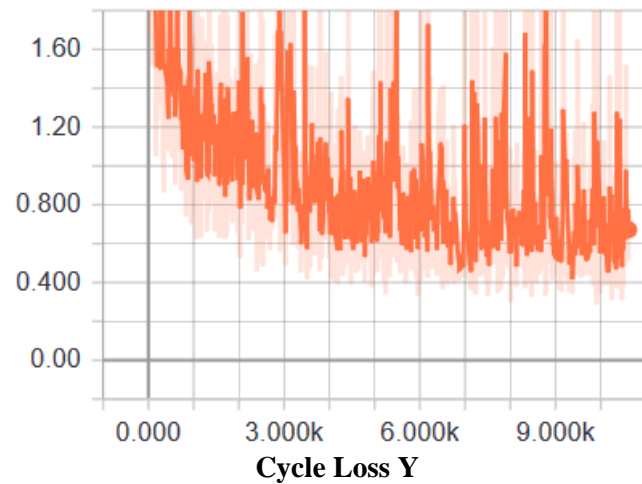
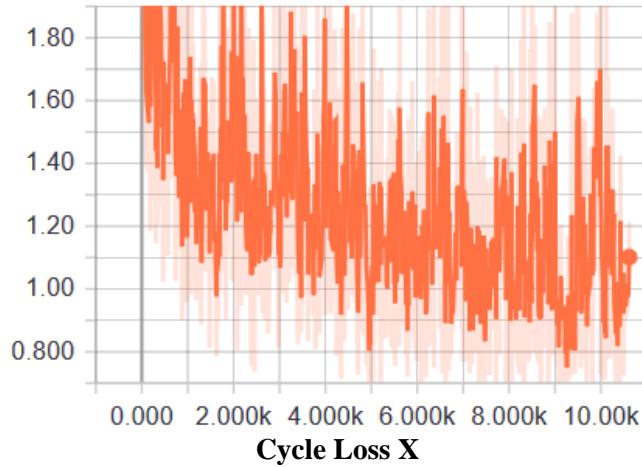
Input Image	Generated Image

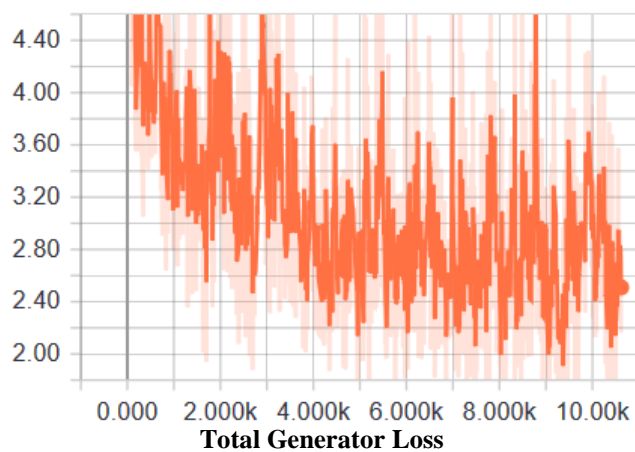
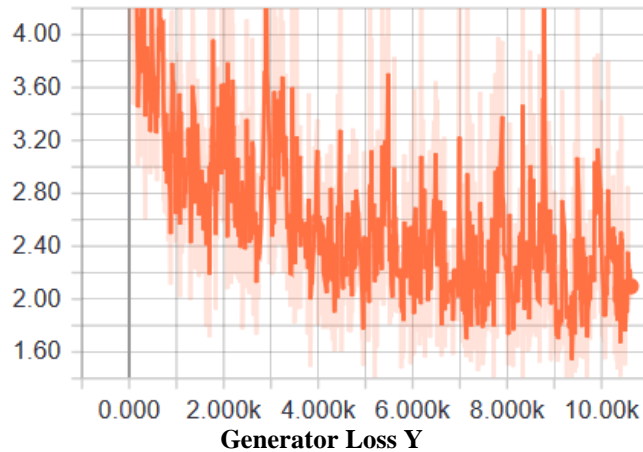
d. Datasets Used: Face to Cartoon

Input Image	Generated Image

5 Visualizations

For visualizations, tensorboard was used.
Below are some screenshots for visualizing losses:
For Dataset used: Horse to Zebra





6 Conclusions and future work

- a. It has been observed that tasks that involve color and texture changes, this method often succeeds. But in tasks that require geometric changes, this method did not perform well. For example, task of face → cartoon transfiguration, the learned translation degenerates into making minimal changes to the input.
- b. Some failure also occurred while translating horse to zebra which might be caused by the distribution characteristics of the training datasets because the model was trained on the wild horse and zebra images which does not contain person in the images.
- c. Handling more varied and extreme transformations, especially geometric changes, is an important problem for future work.

7 References and Links

- [1] Zhu, J., Park, T., Isola, P., & Efros, A. A. (2017). Unpaired Image-to-Image Translation Using Cycle-Consistent Adversarial Networks. *2017 IEEE International Conference on Computer Vision (ICCV)*. doi:10.1109/iccv.2017.244
- [2] <https://github.com/vanhuyz/CycleGAN-TensorFlow>
- [3] <https://github.com/xhujoy/CycleGAN-tensorflow>
- [4] <https://www.tensorflow.org/tensorboard>
- [5] https://people.eecs.berkeley.edu/~taesung_park/CycleGAN/datasets/
- [6] https://medium.com/@jonathan_hui/gan-cycle-gan-6a50e7600d7
- [7] <https://hardikbansal.github.io/CycleGANBlog/>