Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Network

Phillip Harding II, ECE, Online MS

Abstract

This report describes a re-implementation of the paper "Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks" (2). Topics covered in this report include the knowledge foundation, mathematical framework, and key methods to execute unsupervised image translation. The report continues with a review of the re-implementation results compared to the baseline and original paper (2). Then concluding with an analysis and reflection of limitations and design decisions.

1. Introduction

The paper "Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks" (2) introduces a method of mapping two sets of images from different domains, translate, without having corresponding trained and unordered image pairs prior. This is referred to as unsupervised or unpaired image-to-image translation. The important emphasis is that the special characteristics of a set of images X that make it unique are being transferred to a different set of images Y that have minimal similarities by learning the relationship between the two sets. The problem aimed to be solved is that, with a lack of paired data, mapping directly is difficult to near impossible. Also, instances of "mode collapse" may occur with pair data-sets, which is where the image in set X is incorrectly mapped to an image in set Y that shares no relationship. Cycle Consistency Generative Adversarial Network (CycleGAN) is proposed as a solution. Generative Adversarial Network (GAN) is the process of taking "real" existing data and creating (generating) "fake" data that is indistinguishable from the real. Cycle Consistency (Cycle) takes the generated image and attempts to translate it back to the original. Image and video editing as well as computer vision are key examples of applications that would greatly benefit from this method because in both of these examples, the image or video alteration needed isn't always known prior to execution, and an example of said alteration does not exist. Additional detection of the needed changes could be difficult and time-consuming for the same reasons. The goal of this report is to share the understanding and results of re-implementing the paper (2).

2. Knowledge Foundation

The knowledge foundation to understand the paper (2) and the report's implementation begins with GANs. The goal of a GAN is to create new data (in our case, images) that is similar to the existing training data. A GAN's success is measured by its adversarial loss, which is the difference between the ground-truth real data and the generated predicted data. The innovation of this paper (2) is its unsupervised approach to unpaired image-to-image translation, meaning that there is no paired or target output image Y for the corresponding input image X. By using cycle consistency, we aim to improve the image-to-image translation by inverting the original translation using two separate generators and training two discriminators separately. The cycle consistency loss is the difference between the original image and the reconstruction of the original image based on the generated image. Regularization by L1, or lasso regression, will be the means to minimize adjusted loss and prevent over-fitting and under-fitting. L1 loss also known as absolute error loss is the absolute difference between a prediction and the actual value.

3. Method

3.1. Adversarial Loss

The objective of a cycle consistency generative adversarial network (CycleGAN) is to learn to map input domain data-set X to output domain data-set Y, given training samples of X referred to as x and samples of Y referred to as y. How this is uniquely accomplished is by training two mapping generator networks that are inverses of each other. Generator network G's goal is to take an input image x and generate an output image $\widehat{Y} = G(x)$, $G : x \rightarrow \widehat{Y}$. Generator network F's goal is to take an input image y and generate an output image $\widehat{X} = F(y)$, $F : y \rightarrow \widehat{X}$. Additionally, two adversarial discriminator networks are needed. Discriminator D_Y's goal is to determine if the generated image $\widehat{Y}$ is real or not. Discriminator D_X's goal is to determine if the generated image $\widehat{X}$ is real or not. Each discriminator will produce adversarial losses, the difference between the ground-truth

real image and the generated image. As stated in the paper (2), the adversarial losses were calculated utilizing the mean square error loss shown accordingly:

$$\mathcal{L}_{GAN}(G, D_Y, X, Y) = \mathbb{E}_{x \sim p_{data}(x)}[(D(G(x)) - 1)^2] + \mathbb{E}_{y \sim p_{data}(y)}[(D(y) - 1)^2] + \mathbb{E}_{x \sim p_{data}(x)}[D(G(x))^2] \quad (1)$$

$$\mathcal{L}_{GAN}(G, D_X, X, Y) = \mathbb{E}_{y \sim p_{data}(y)}[(D(G(y)) - 1)^2] + \mathbb{E}_{x \sim p_{data}(x)}[(D(x) - 1)^2] + \mathbb{E}_{y \sim p_{data}(y)}[D(G(y))^2] \quad (2)$$

3.2. Cycle Consistency Loss

An additional key method for this implementation is cycle consistency, which constrains the generators. Cycle consistency prevents the learned mapping of generators G and F from conflicting with each other, which minimizes the adjusted loss. In short, we are regularizing the mapping to ensure that the reconstructed images from the Y output domain are similar to the original X domain input images. Producing a network that learns meaningful and bijective mapping with accurate translations. Building on what was mentioned so far, we now take our generated images $\widehat{X}$ and $\widehat{Y}$ and apply them to the opposite generator from which they originated. Since the generators are inverses of each other, the result should be a reconstruction of the original image, which we will call $\widehat{x}$ and $\widehat{y}$. The cycle consistency losses that are produced are the difference between the original images x and y and the reconstruction of the original images $\widehat{x}$ and $\widehat{y}$. $F(G(x)) = F(\widehat{Y}) = \widehat{x}$ and $G(F(y)) = G(\widehat{X}) = \widehat{y}$. The losses are calculated utilizing L1 norm to preserve the original image characteristics such as color and tint, by taking the difference pixel by pixel. This is shown accordingly:

$$\mathcal{L}_{cycle}(G, F) = \mathbb{E}_{x \sim p_{data}(x)}[||F(G(x)) - x||_1] + \mathbb{E}_{y \sim p_{data}(y)}[||G(F(y)) - y||_1]] \quad (3)$$

3.3. Identity Loss

In the original paper (2), identity loss was used for a few data- sets. The decision to incorporate it into the implementation in an attempt to improve the results, which was successful. As introduced in the paper (2), identity learning is a method of preserving the background or non-regions of interest (non-ROI) and only translating the subject of the paper (2), horses and zebras, for the data-set. Identity learning is implemented by taking a real image of the intended output and passing it through the generator, which is trained to generate the intended output image. For example, by taking the real image y and producing a generated image $\widehat{Y}$, theoretically, there should be no changes or alterations made. As a result, the identity loss difference between y and $\widehat{Y}$ in this example should be zero. The identity loss ensures that the generators don't change the color, tint, or style of an image when there is no need to do so. The L1 norm was also used to calculate the identity loss, as shown accordingly

$$\mathcal{L}_{identity}(G, F) = \mathbb{E}_{y \sim p_{data}(y)}[||(G(y)) - y||_1] + \mathbb{E}_{x \sim p_{data}(x)}[||(F(x)) - x||_1]] \quad (4)$$

3.4. Full Objective

The total loss objective function is the sum of each of the generator's adversarial losses plus the cycle consistency loss times the cycle loss weight, λ_{cycle}, plus the identity loss times the identity loss weight, $\lambda_{identity}$. Combining the losses completes the objective of unpaired image-to-image translation. The calculations are shown accordingly:

$$\mathcal{L}(G, F, D_X, D_Y) = \mathcal{L}_{GAN}(G, D_Y, X, Y) + \mathcal{L}_{GAN}(G, D_X, X, Y) + \lambda_{cycle}\mathcal{L}_{cycle}(G, F) + \lambda_{identity}\mathcal{L}_{identity}(G, F) \quad (5)$$

4. Experiment

The approach to executing the project implementation was to review an alternative implementation of the original paper (2) to better understand the execution of this method through the similarities and differences. Early on into re-implementation it was realized that the challenges faced would be due to the limited computing power, long run-times, and limited background and programming skills possessed, but more on that in the conclusion section. The implementation that made the most sense and was able to followed was Aladdin Persson's execution of CycleGAN (1). This is what was used as a baseline in order to compare their results with the original paper's (2) results and the results shown in this report.

The hyper-parameters specified per the paper (2) that were used were a batch size of 1, a learning rate of 0.0002, a cycle loss weight (λ_{cycle}) of 10, and an identity loss ($\lambda_{identity}$) weight of 0.5 * λ_{cycle}. Lastly, due to computation limits and long run-times, it was deemed ideal to limit the learning rate to 100 epochs instead of the 200 specified by the paper (2). Also, a learning rate decay was not implemented.

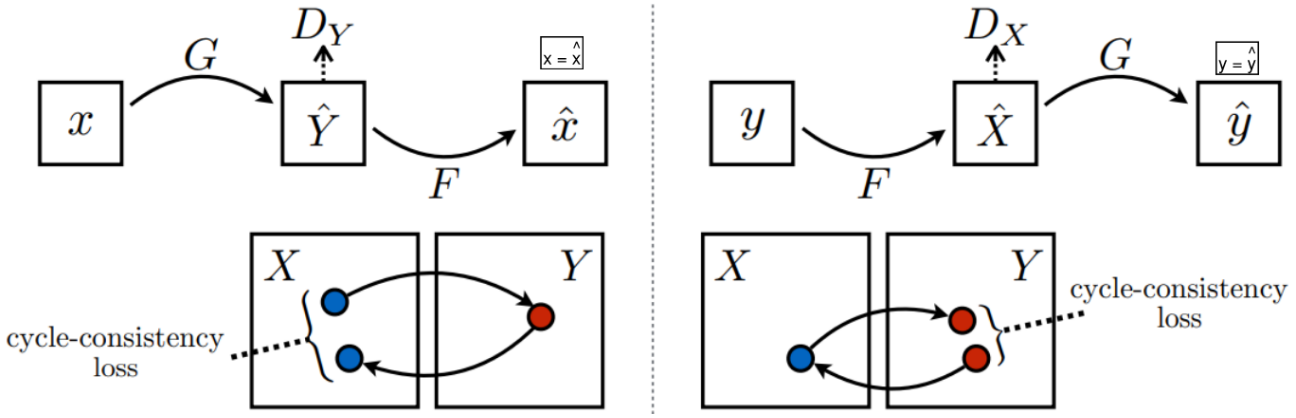

Figure 1. Cycle Consistency Block Diagram

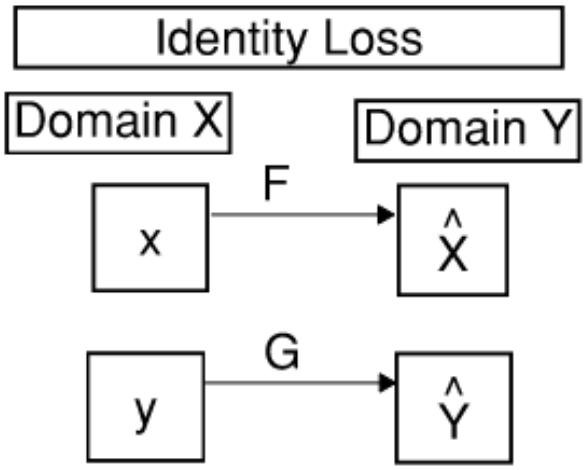

Figure 2. Identity Loss Block Diagram

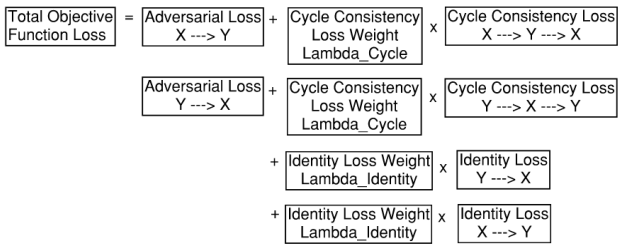

Figure 3. Generator Loss Block Diagram

The data set used was the horses (domain X) to zebras (domain Y) data-set, which was one of the data-sets used in the original paper (2). Identity learning and loss were not used in evaluating this data-set in the original paper (2); however, through multiple interactions, the solution was to incorporate identity loss to improve the results, especially since the number of epochs was halved. The paper (2) states that the horse-to-zebra data-set was a collection of 939 horse images and 1176 zebra images scaled to 256 x 256 pixels. The generator network architecture used has four convolutional layers and nine residual blocks to address images that are 256 x 256, as described in the paper (2).

The best of the translation results are shown below. A visual inspection is sufficient to determine the success of the implementation, given the nature of this specific project. A successful translation is defined as the subject of the image being converted with minimal to no conversion of the background. Of the largest sample of images that were reviewed, the ”success” rate of horse-to-zebra translation was 18.65%, or 25 out of 134 images. Of the zebra to horse translation, 15.51% (18 out of 116) was the success rate. The horse-to-zebra translation was more successful due to the zebra pattern being black and white, which often contrasts well with the earthy background colors and brown fur of the horses The zebra-to-horse translation has to overcome replacing the original zebra pattern with brown fur, which is often not completely accomplished. Prior to incorporating the identity loss, none of the reviewed images from prior implementations preserved the background; there were often significant alterations, often translating the pattern of the zebra to the background. The hypothesis is that the due to the earthy background in the photos is similar to the brown horse fur. This still persisted in the final run, but far less. Whenever there is more than one subject in an image and the subjects overlap, it is difficult to separate the overlapping subjects in the translation of the image, specifically in the

case of the generated zebra images.

I recognize and acknowledge that minimal changes were made when compared to Aladdin Persson's implementation (1); however, I truly believe this was the simplest and most digestible implementation of the several reviewed.

5. Conclusion

The paper "Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks" enabled image-to-image translation using unsupervised learning that mapped between domains without relying on paired data. Unpaired data is the reality of the majority of datasets and is often difficult to work with. However, a benefit is that unpaired data requires less categorization than paired data, which allows for an increase in training sample size as well as the freedom to make any trained pairing. Image translation is when the input is an image and the output is a different version of the input image that is changed according to guidelines. This paper accomplished these feats by using two discriminative GANs that are equivalent inverses of each other but learn and improve together. Each network consists of a generative network that tries to create a realistic image and a discriminator network that tries to learn the difference between real images and fake generated images. To regularize and constrain the GANs and improve the quality of the image translation, a cycle consistency loss function that converts input to output back to the original input image was incorporated per GAN. An additional identity loss regularization term further improved the quality of the translation. A strength of the paper's implementation is changing the color and texture of the subject in the image. A weakness is changing the shape of the subject in the image.

A few questions and difficulties arose during the re-implementation of this report. First, I doubted the need to divide the total loss objective function by 2, because intuitively, it did not make sense how that would slow the rate at which the discriminators learned. Only after several iterations with and without dividing by two was it noticed that the translation was overfitting with the dividing the objective by 2. Second, it is assumed that increasing the batch size would improve the results; however, due to the limitations of computation capacity and runtime execution, exploring this option was not feasible for this report, and the same can be assumed for the original paper. Incorporating density loss did improve the final results, especially the background pixels of the images. The intent was not to do so because the original paper did not include the horse-to-zebra dataset. After doing so, the problem that arose was that initially the identity loss weight was incorrectly set to 10, the cycle consistency loss weight. The correction was to set the identity loss weight equal to half the cycle loss weight of 10 per the paper.

The results obtained from this re-implementation were conclusively worse than the baseline and the original paper. To remedy this, more epochs are needed to improve learning and reduce total losses. For future attempts, outputing the losses to see if there is improvement as the epochs increase would provide more insight. Solving the runtime issue would be difficult because the iterations per epoch were 1300 plus and took 5-7 hours for 100 epochs while utilizing the Google Colab Premium GPU. Reducing iterations to improve runtime could improve performance by making the code more efficient; however, a solution was not obtained.

The code implementation can be found here:
https://github.com/PHTheSecond/ECE-50024-Final-Project-CycleGAN

6. Acknowledgements

As mentioned earlier, Aladdin's simple and readable implementation of CycleGAN (1) was used as the baseline and basis for executing this project. Lecture and course material, as well as the original paper "Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks," provided the foundation for the understanding and explanations of this paper (2). OpenAI's MLGPT was used to polish the written text.

References

[1] Aladdin Persson. "CycleGAN implementation using PyTorch." *GitHub repository*, 2022. https://github.com/aladdinpersson/Machine-Learning-Collection/tree/master/ML/Pytorch/GANs/CycleGAN

[2] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. "Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks." In *Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2223–2232, 2017.

[3] MLGPT. "Large Language Model Trained by OpenAI." *OpenAI Blog*, 2023. https://openai.com/blog/gpt-3-5-billion/

Figure 4. Horse to Zebra Results

Figure 5. Zebra to Horse Results