Name: SAHIL

Email address: sahiltinky94@gmail.com

Contact number: 9711308110

Anydesk address:

Date: **03rd May 2020**

Self Case Study -2: Image2Image Translation

Overview

*** Write an overview of the case study that you are working on. ***(MINIMUM 200 words)*** ***

Image to Image translation is one of the computer visions where most of practical applications used. But doing with the hand task takes a lot of times. For example, I have an image, I want to recreate same image with different style (like into sketch drawing).

Various researcher perform on different domain specific (like translate image from daytime to nighttime, translate original image into the context of artistic style, etc.) and some of the models work perfect in some domain and some other domain doesn't work. The key challenge is that they able to created well but the time is the key. A lot of research trying to optimizing each of the model with different objective in order to achieve this task.

So, basically, you have two inputs: one is content image which you want to translate into different style and another is style image which you wish to want to translate it to content image.

There is no dataset available for this. However, we will limit to some domain images as there are varieties of images (like nature, art, logo, landscape, places, etc). Of course some people have used dataset, but we will download the images from the available openly in Internet.

There are various dataset that performed and are available in (: https://people.eecs.berkeley.edu/~taesung_park/CycleGAN/datasets/). So, we will focus on **cezanne2photo, monet2photo, vangogh2photo and ukiyoe2photo** dataset.

There are two train data folder: A and B where one folder is for context and other folder is for style representation.

trainA images (artist style)


trainB images (content/ photo)


Dataset	trainA (art)	trainB (photo)	testA (art)	testB (photo)
Cezanne2photo	525	6287	58	751
Monet2photo	1072	6287	121	751
Ukiyoe2photo	562	6287	263	751
Vangogh2photo	400	6287	400	751

trainA and testB have same images present in it.

Research-Papers/Solutions/Architectures/Kernels

*** Mention the urls of existing research-papers/solutions/kernels on your problem statement and in your own words write a detailed summary for each one of them. If needed you can include images or explain with your own diagrams. ***

1. Texture Synthesis Using Convolutional Neural Network [Leon A. Gatys et al.] (2015)

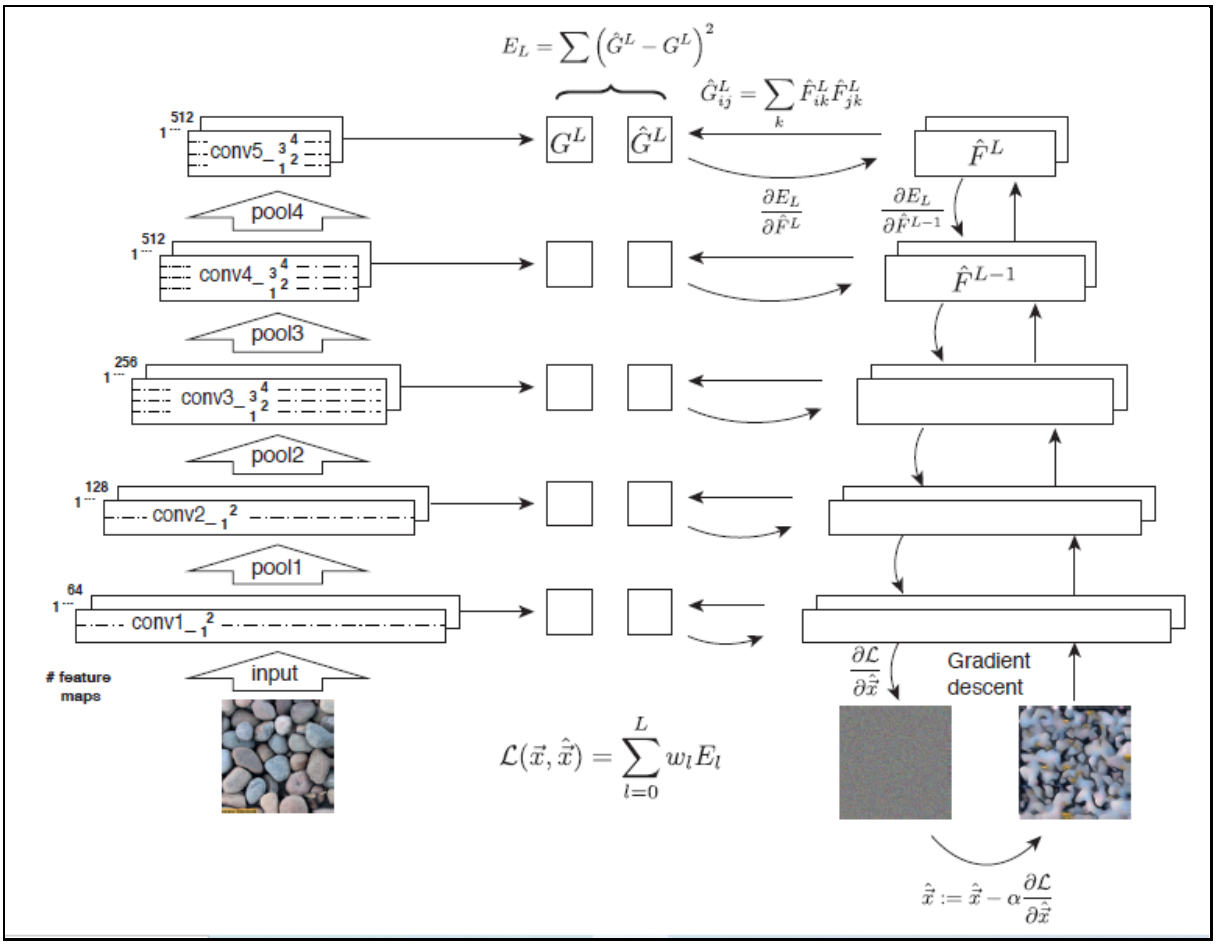
Goal and Evaluation:

The goal of visual texture synthesis is to infer a generating process from an example texture, which then allows to produce arbitrarily many new samples of that texture. The evaluation criterion for the quality of the synthesised texture is usually human inspection and textures are successfully synthesised if a human observer cannot tell the original texture from a synthesised one.

Proposal Work:

It used a convolutional neural network as the foundation for our texture model. Then combined the conceptual framework of spatial summary statistics on feature responses with the powerful feature space of a convolutional neural network that has been trained on object recognition.

Architecture and Working:



Configuration Of Network:

It used VGG19 Network which consisted of 16 convolutional and 5 pooling layers. No Fully Connected Used. ReLu activation function with kernel size 3x3xk where k in number of input feature maps. Fixed the strides (s) and padding (p) equal to 1.

Initially performed using MaxPooling with kernel size 2x2. However, AveragePooling gives the better result than MaxPooling while experimental.

Texture Model:

1. Starting from the left side of the architecture, feed input original texture to networks.
2. Compute activations for each layer in the network.
3. At each layer, its compute 'Gram Matrix' (which is inner product of two feature map i and j in layer l)

$$G^l_{ij} = \sum_k F^l_{ik} F^l_{jk}.$$

Texture Generation:

1. Now, on the right side of the architecture, To generate a new texture on the basis of a given image, it uses gradient descent from a white noise image to find another image that matches the Gram Matrix of the original image.
2. Observe left and right, both networks, at each layer l, compute G(l) and G'(l). It calculates the total loss of each contribution layer.

$$E_l = \frac{1}{4N_l^2 M_l^2} \sum_{i,j} \left(G_{ij}^l - \hat{G}_{ij}^l \right)^2$$

$$\mathcal{L}(\vec{x}, \hat{\vec{x}}) = \sum_{l=0}^{L} w_l E_l$$

where w_l are weighting factors of the contribution of each layer to the total loss.

3. For gradient w.r.t activation layer 'l' (for right network only, because we need to update their weights, not for left network architecture)

$$\frac{\partial E_l}{\partial \hat{F}_{ij}^l} = \begin{cases} \frac{1}{N_l^2 M_l^2} \left((\hat{F}^l)^{\mathrm{T}} \left(G^l - \hat{G}^l \right) \right)_{ji} & \text{if } \hat{F}_{ij}^l > 0 \\ 0 & \text{if } \hat{F}_{ij}^l < 0 \,. \end{cases}$$

4. It used an L-BFGS optimizer. (It's an old technique. Basically think of L-BFGS as a way of finding a (local) minimum of an objective function, making use of objective function values and the gradient of the objective function. https://stats.stackexchange.com/questions/284712/how-does-the-l-bfgs-work/285106)

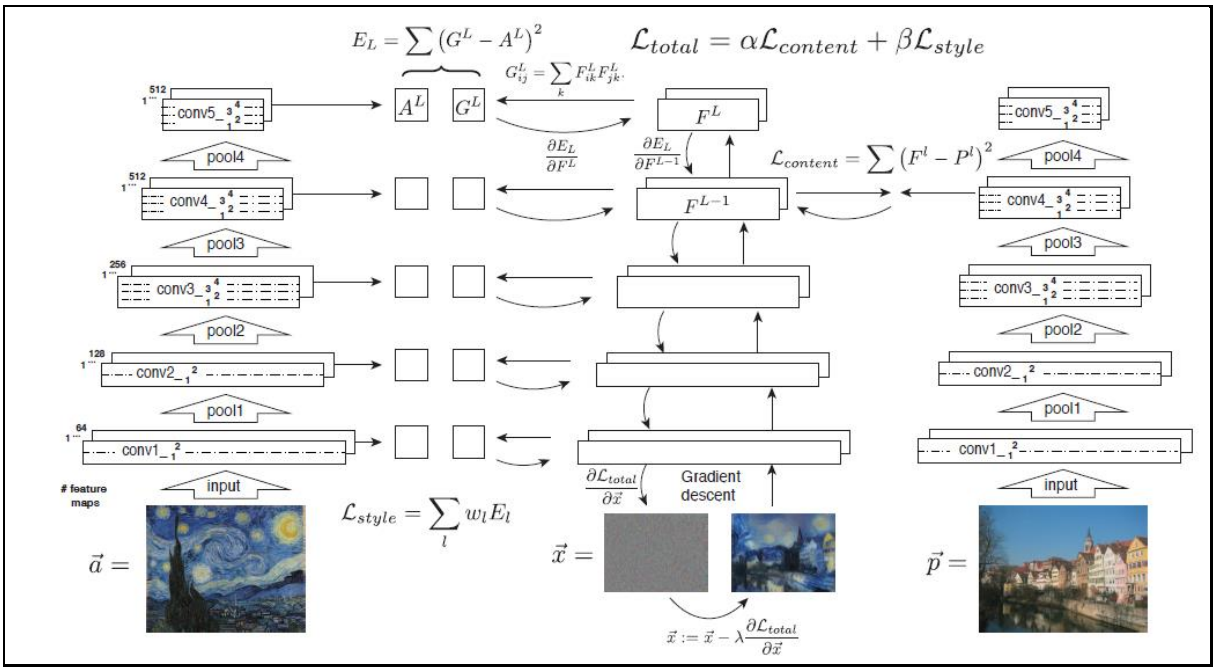
2. Image Style Transfer Using Convolutional Neural Networks [Leon A. Gatys et al.] (2016)

Transferring the style from one image onto another can be considered a problem of texture transfer.

Goal:

To synthesize a texture from a source image whole constraining the texture synthesis in order to preserve the semantic content (like object) of a target image.
Conceptually, it is a texture transfer algorithm that constrains a texture synthesis method by feature representation from convolutional neural network..

Architecture and Working:



If you notice that the left and middle network is exactly the same as we discuss in previous ones. It extended the target image on the right side of the network.

The <u>configuration of the network</u> is the same as the previous one as we discussed.

<u>Content Representation:</u>

1. Variables p and x are original image and image generated and P(l) and F(l) are their feature representation with respective laye 'l'.
2. Define the squared-error loss between the two feature representations.

$$\mathcal{L}_{\text{content}}(\vec{p}, \vec{x}, l) = \frac{1}{2} \sum_{i,j} \left(F^l_{ij} - P^l_{ij}\right)^2 .$$

3. The gradient w.r.t feature in layer 'l'.

$$\frac{\partial \mathcal{L}_{\text{content}}}{\partial F^l_{ij}} = \begin{cases} \left(F^l - P^l\right)_{ij} & \text{if } F^l_{ij} > 0 \\ 0 & \text{if } F^l_{ij} < 0 , \end{cases}$$

Update gradient w.r.t x can be computed by standard error-backpropagation.

These refer to the feature responses in higher layers of the network as a 'content representation'.

Style Representation:
Just the same as previous as we discussed.

Style Transfer:

1. To transfer the style of an artwork ‘a’ onto photograph ‘p’, it synthesis a new image that simultaneously matches the content representation ‘p’ and the style representation ‘a’.
2. It jointly minimizes the distance of the feature representation of a white image noise from the content representation of the photograph **in 1 layer only**. And the style representation of the painting is defined on **a number of layers** of network.
3. Loss function we minimize

$$\mathcal{L}_{\text{total}}(\vec{p}, \vec{a}, \vec{x}) = \alpha \mathcal{L}_{\text{content}}(\vec{p}, \vec{x}) + \beta \mathcal{L}_{\text{style}}(\vec{a}, \vec{x})$$

4. Used L-BFGS optimizer.

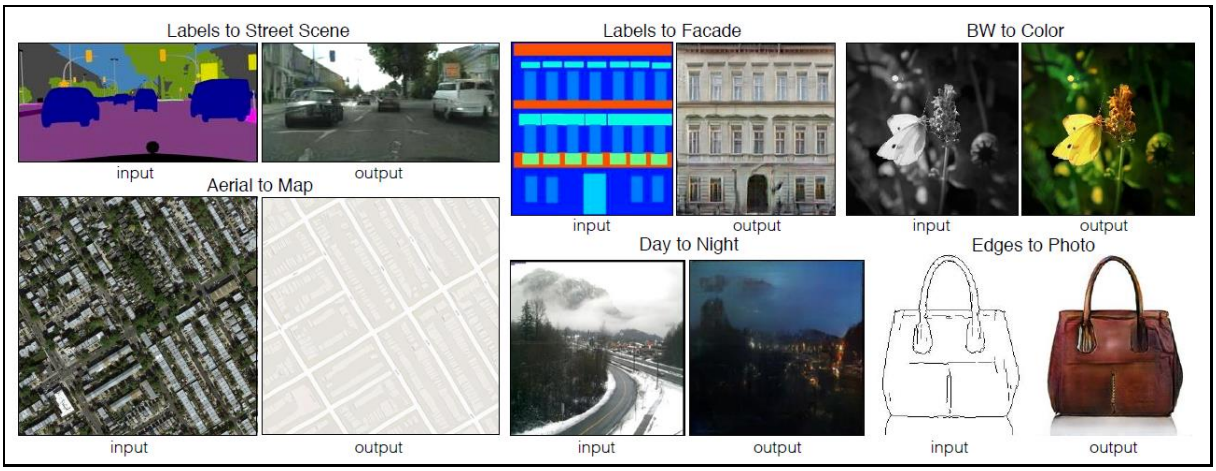
They have done several hyperparameter on trade off between content and style matching: $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ and shown the result. ALso they also showed the result of different layers output with style and content.

Limitation:

1. Most limiting factor is resolution of the synthesis images. So, speed of processing depends on resolution of an image. Like in this paper, 512x512 resolution takes one hour on an Nvidia K0 GPU.
2. Synthesised images are sometimes subject to some low-level noise. The problem becomes more apparent when both, content and style images, are photographs and the photorealism of the synthesised image is affected. Noise is very characteristic and appears to resemble the filters of units in the network. Thus it could be possible to construct efficient de-noising techniques to post-process the images after the optimisation procedure.

3. Image-to-Image Translation with Conditional Adversarial Network [Phillip Isola et al.] (2018)

In analogy to automatic language translation, it defines automatic image-to-image translation as the task of translating one possible representation of a scene into another, given sufficient training data. (As shown figure below)



Contribution:

1. Demonstrate using Conditional GAN
2. Simple framework sufficient to achieve good results and analyse several architecture choices.

Architecture

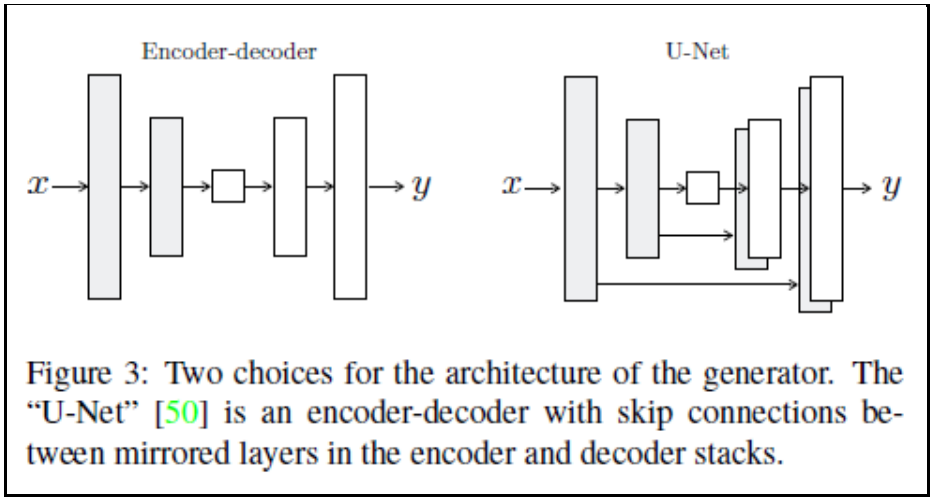


Figure 3: Two choices for the architecture of the generator. The "U-Net" [50] is an encoder-decoder with skip connections between mirrored layers in the encoder and decoder stacks.

The overview architecture

Both generator and discriminator use modules of the form convolution-BatchNorm-ReLu.

- Generative with skips connections: Generator in GAN architecture is modified U-Net. Within the U-Net architecture, one of them is encoder where each block is (Conv -> Batchnorm -> Leaky ReLU) and for other is decoder where each block is is (Transposed Conv -> Batchnorm -> Dropout(applied to the first 3 blocks) -> ReLU). And apply skip connection between encoder and decoder.

- Markovian discriminator (PatchGAN): Discriminator in GAN is PatchGAN. Each block in the discriminator is (Conv -> BatchNorm -> Leaky ReLU). From this paper, the shape of the output after the last layer is (batch_size, 30, 30, 1). Each 30x30 patch of the output classifies a 70x70 portion of the input image (such an architecture is called a **PatchGAN**). As usual, a discriminator model receives 2 inputs: real and generated image.

Conclusion:

1. U-Net architecture allows low-level information to shortcut across the network. However, encoder-decoder is unable to learn to generate realistic images in this experiment.
2. The advantage of U-Net appears not to be specific conditional GANs: when both U-net and encoder-decoder are trained with an L1 loss, and by the result shown that U-Net again achieves the superior results.

4. Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks [Jun-Yan Zhu et al.] (2018)

They present an approach for learning to translate an image from a source domain X to a target domain Y in the absence of paired examples. Their goal is to learn a mapping G : X → Y such that the distribution of images from G(X) is indistinguishable from the distribution Y using an adversarial loss. Because this mapping is highly under-constrained, we couple it with an inverse mapping F : Y → X and introduce a cycle consistency loss to enforce F(G(X)) ≈ X (and vice versa). It exactly called CycleGAN

Below 2 figures shown: Fig2. Illustrate the example of paired examples and unpaired example while training. Fig3. Illustrate the represent of CycleGAN.

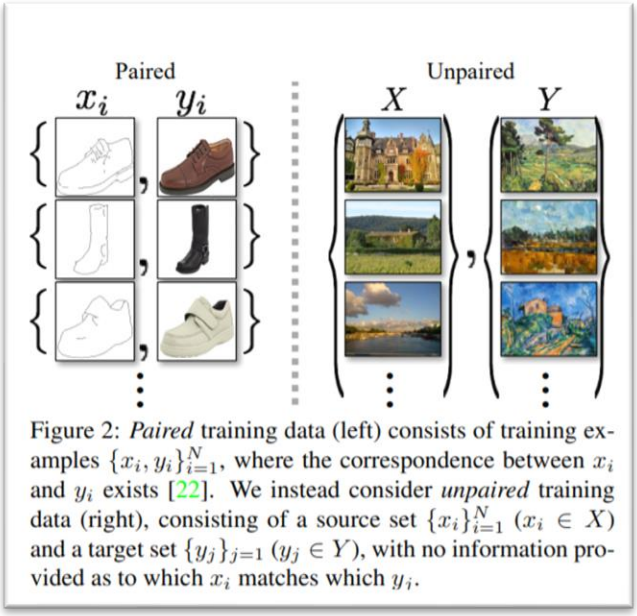


Figure 2: *Paired* training data (left) consists of training examples $\{x_i, y_i\}_{i=1}^N$, where the correspondence between x_i and y_i exists [22]. We instead consider *unpaired* training data (right), consisting of a source set $\{x_i\}_{i=1}^N$ $(x_i \in X)$ and a target set $\{y_j\}_{j=1}$ $(y_j \in Y)$, with no information provided as to which x_i matches which y_i.

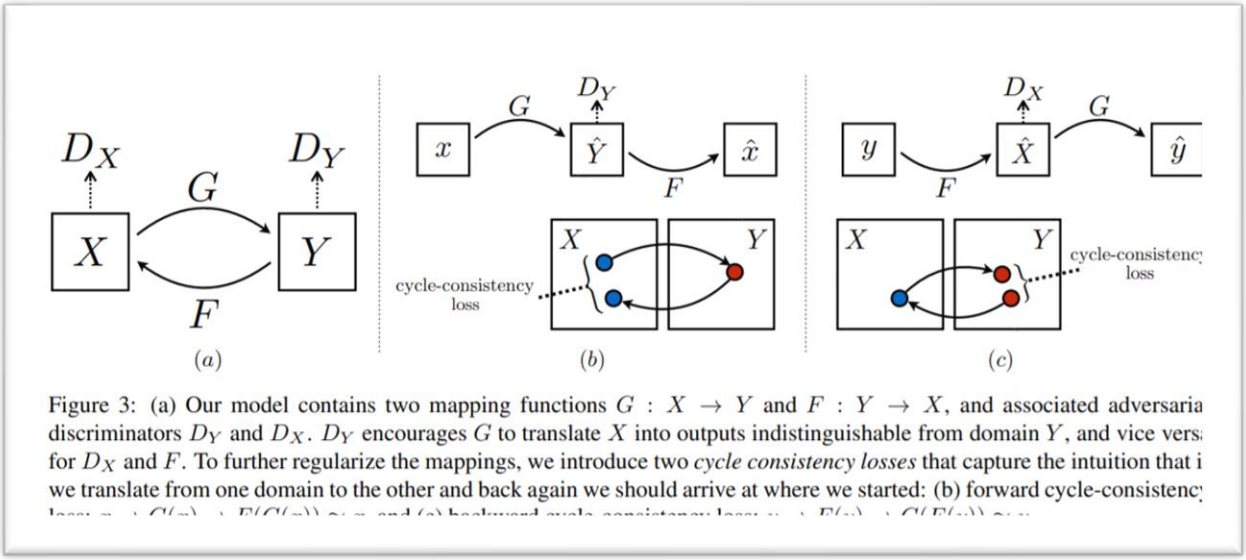


Figure 3: (a) Our model contains two mapping functions $G : X \rightarrow Y$ and $F : Y \rightarrow X$, and associated adversaria discriminators D_Y and D_X. D_Y encourages G to translate X into outputs indistinguishable from domain Y, and vice vers for D_X and F. To further regularize the mappings, we introduce two *cycle consistency losses* that capture the intuition that i we translate from one domain to the other and back again we should arrive at where we started: (b) forward cycle-consistenc

- This blog (https://machinelearningmastery.com/what-is-cyclegan/) explained very well about CycleGAN with example intuitively.

There is objective contains two types of terms: adversarial losses for matching the distribution of generated images to the data distribution in the target domain; and cycle consistency losses to prevent the learned mappings G and F from contradicting each other.

A. Adversarial Losses
   For the mapping function G : X → Y and its discriminator DY , we express the objective as:

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

   Similarly same for F: Y→X with discriminator DX.

B. Cycle Consistency loss
   In Figure 3 (b), for each image x from domain X, the image translation cycle should be able to bring x back to the original image, i.e., x → G(x) → F(G(x)) ≈ x. We call this forward cycle consistency. Similarly, as illustrated in Figure 3 (c), for each image y from domain Y , G and F should also satisfy backward cycle consistency: y → F(y) → G(F(y)) ≈ y. We incentivize this behavior using a cycle consistency loss:

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

Full Objective:

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

where λ controls the relative importance of the two objectives. We aim to solve:

$$G^*, F^* = \arg\min_{G,F} \max_{D_x, D_Y} \mathcal{L}(G, F, D_X, D_Y).$$

Network Architecture:

1. For Generative Model, it used the Resnet Model with some modified layer. It used instance normalization instead of batch normalization

2. For discriminative Model, it used PatchGAN (same as used in research paper by Phillip Isola (2018)

Limitation: The distribution characteristics of the training datasets. For example, our method has got confused in the horse → zebra example

---

First Cut Approach

*** Explain in steps about how you want to approach this problem and the initial experiments that you want to do. ***(MINIMUM 200 words)*** ***

1. Most of the research paper follows and compare with the common research paper by Gatys et al. So, we will consider this as an existing paper.

2. We will try compare our proposal model with Gatys's proposal (VGG model used) and Phillip Isola's proposal (CycleGAN Model used)
