Title
Application of Generative Adversarial Network on Image Style Transformation and Image Processing

Author
Wang, Anshu

Publication Date
2018

Peer reviewed|Thesis/dissertation

UNIVERSITY OF CALIFORNIA

Los Angeles

Application of Generative Adversarial Network on Image Style Transformation and Image Processing

A thesis submitted in partial satisfaction
of the requirements for the degree
Master of Science in Statistics

by

Anshu Wang

2018

ABSTRACT OF THE THESIS

Application of Generative Adversarial Network
on Image Style Transformation and
Image Processing

by

Anshu Wang
Master of Science in Statistics
University of California, Los Angeles, 2018
Professor Ying Nian Wu, Chair

Image-to-Image translation is a collection of computer vision problems that aim to learn a mapping between two different domains or multiple domains. Recent research in computer vision and deep learning produced powerful tools for the task. Conditional adversarial networks serve as a general-purpose solution for image-to-image translation problems. Deep Convolutional Neural Networks can learn an image representation that can be applied for recognition, detection, and segmentation. Generative Adversarial Networks (GANs) has gained success in image synthesis. However, traditional models that require paired training data might not be applicable in most situations due to lack of paired data.

Here we review and compare two different models for learning unsupervised image to image translation: CycleGAN and Unsupervised Image-to-Image Translation Networks (UNIT). Both models adopt cycle consistency, which enables us to conduct unsupervised learning without paired data. We show that both models can successfully perform image style translation. The experiments reveal that CycleGAN can generate more realistic results, and UNIT can generate varied images and better preserve the structure of input images.

The thesis of Anshu Wang is approved.

Hongquan Xu

Mark Stephen Handcock

Ying Nian Wu, Committee Chair

University of California, Los Angeles

2018

To my parents . . .

who taught me that

Success will come and go

But Integrity is forever

TABLE OF CONTENTS

LIST OF FIGURES

ACKNOWLEDGMENTS

I want to express my sincere gratitude to my advisor Dr. Ying Nian Wu, who motivated my interest in learning and inspired my life attitude.

I want to acknowledge Erik Nijkamp for exposing me to recent advancements in science.

I want to thank Jiajie Chen, for setting an example for me, making me realize I can always be a better person.

CHAPTER 1

Introduction

Image-to-Image translation is a collection of computer vision problems. From photo editing Apps that transfer photos to stylistic images, to the emerging application of autonomous driving where we need a segmentation network to detect roads, cars, and pedestrians, there are more and more tasks that aim to learn a mapping between two different domains or multiple domains.

For different purposes, the objective of Image-to-Image translation may vary. For photo editing, we want the photo to look fancy, but at the same time realistic. To create maps from satellite images, we emphasize the accuracy of the translated map and the correct coloring of the map.

Recent research in computer vision and deep learning produced powerful tools for the task. Conditional adversarial networks serve as a general-purpose solution for image-to-image translation problems. Deep Convolutional Neural Networks can learn an image representation that can be applied for recognition, detection, and segmentation. However, traditional models that require paired training data might not be applicable in most situations since it is too expansive to collect paired data. Besides the powerful neural networks, we need an appropriate approach to learn from unpaired data, in another word, how to learn the mapping in an unsupervised setting.

Generally, most image-to-image translation has to obey the following rules: 1) the transformed images should be indistinguishable from both domains. That is to say, if you turn a satellite image into a map, it looks like other maps. 2) The image transformation should be invertible. Let's say you want to turn an apple image into an orange image, and you can satisfy the first rule by always mapping the apple image to the same orange image. We also want the mappings to be well-behaved, that is to say, the distribution learned does not fall into single modes.

In the thesis, we mainly discuss two approaches for learning unsupervised image to image translation: CycleGAN and Unsupervised Image-to-Image Translation Networks (UNIT). Both models adopt cycle consistency, which is proposed to solve the second rule mentioned above. We can see that UNIT and CycleGAN share some similarities in the models. CycleGAN generates images that are more indistinguishable while UNIT, restricted by the VAE generators, are more picky on training sets but provides more varied results.

CHAPTER 2

Prior Art

The task of Image-style transformation is to learn a mapping between images from two domains. Previous work has shown the capacity of convolution Neural Network (CNN) on handling image problems. Paper [3] extracts the texture and content representation from the images via a deep neural network for texture and style transfer. Context Encoder [13] apply CNNs to generate the contents of an arbitrary image region conditioned on its surroundings. It is very surprising how CNNs perform on image classifications and how the learned deep image representations demonstrate potential for high-level image synthesis and manipulation.

Recently, Generative Adversarial Networks (GANs) have achieved impressive results on realistic image generation and representation learning. Instead of defining a loss function like CNNs, GANs use a discriminator to automatically learn the loss for outputting images that can not be told from the training images. Since GANs learn the loss that adapts the data, they are more flexible than traditional models and can be applied to a broader range of problems. Some recent research in the stability and convergence of GANs [4][1] make GANs more applicable. Conditional GANs (CGANs) can be viewed as GANs that learn a conditional generative model. Most of image-to-image translation models are built on CGAN since we can condition on the input image to generate an output image.

One successful work is pix2pix [6], which led to a lot of applications, such as image super-resolution [9] and image coloring. Later, Cycle-GAN proposes cycle-consistency loss so that we don't need paired data for training. The idea of cycle-consistency is not new. In

natural language processing, we can use "back translation" to verify and improve machine translation or to augment the existing data in different languages. Cycle consistency loss has been leveraged in Cycle-GAN, Disco-GAN, Dual-GAN. Most models use l_1 loss for cycle consistency loss because l_2 loss tends to give a blurry result.

Paper [12] models each domain using a VAE-GAN and connects both networks by a weight sharing constraint, which enforces a shared latent space. Say we want to generate a female face to a male face: what we already know is that all faces are alike: everyone has two eyes, one nose, and one mouth. We can use two variational auto-encoders to generate male face and female face. But what if we want to change a male face into a female face while making the female face look like the original one? The paper postulates a shared latent space assumption to combine two VAEs learned from two domains and again, add cycle consistency loss to regularize the ill-posed unsupervised image-to-image translation problem further.

CHAPTER 3

Generative Adversarial Network

3.1 Introduction

Generative models and discriminative models are the two main approaches in learning. Generative models aim to learn the distribution of the data, while discriminative models intend to model the distribution of data conditioning on the observed data. Generative models are generally more flexible since they can not only be used to calculate conditional distributions for classification tasks, but they can also generate new data from the learned distribution for data augmentation.

Training generative models also enables inference of the latent space. A variational autoencoder (VAE) forces the network to learn a mapping from an independent Gaussian space to the target distribution, say a set of images. Another type of generative model is Generative Adversarial Network (GAN). VAE learns by minimizing the loss between original data and compressed data, while GAN learns it in an adversarial way: A generative network is trained to fool the discriminator while the discriminator learns to tell the fake data generated by the generator.

3.2 Generative Adversarial Network

A Generative Adversarial Network consists of a generator network and a discriminator network. We want the generator to generate data that resembles the training data. It is not easy to measure how much a generated image resembles the training set. However, we can train a discriminator to tell what's the probability that the generated image belongs to the training set class.

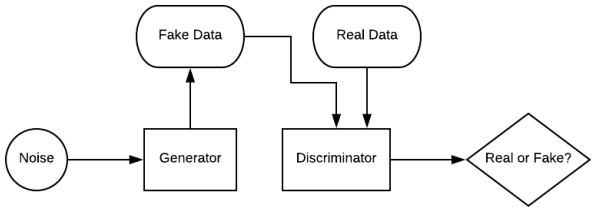

Figure 3.1: Illustration of Generative Adversarial Network

We denote the discriminator to estimate the probability of a given sample coming from the real dataset as D, and the generator to synthesize samples given a noise input z as G. The generator G aims at generating "real" data that can pass the discriminator, while the discriminator tries to tell the fake data from the real data. Mathematically, our goal can be expressed as below:

$$\begin{aligned}\min_G \max_D L(D,G) &= \mathbb{E}_{x\sim p_r(x)} log D(x) + \mathbb{E}_{z\sim p(z)} log(1 - D(G(z))) \\ &= \mathbb{E}_{x\sim p_r(x)} log D(x) + \mathbb{E}_{x\sim p_g(x)} log(1 - D(x))\end{aligned}$$

where $p_r(x)$ is the distribution of the real data and $p_g(x)$ is the distribution of the data synthesized by the generator.

From the objective function, we can tell that the training of GAN is not easy. A recent paper [16] discussed the problem with GANs gradient-descent-based training procedure. The

training process of GAN can be viewed as finding a Nash equilibrium, and there might exists multiple equilibria. A most popular way is to update the gradient of the generator and discriminator alternatively. There is no guarantee a convergence would be reached by doing so. Also the training might take a long time before it actually converges. There have been emerging studies on convergence and stability of GAN.

Another common problem with GAN is mode collapse. Mode collapse is when the generator generates a limited diversity of samples, no matter what input z is given. The cause for mode collapse is that in our objective, we only ask the generator to fool the discriminator, but we don't address diversity in data. The paper [15] points out that we can apply inception score to identify mode collapse. [10] proposes Markovian Generative Adversarial Net- works (MGANs), a method for training generative neural networks for efficient texture synthesis.

CHAPTER 4

Unpaired Image Style Transformation using Cycle-GAN

Paper [19] proposed Cycle-GAN for image to image transformation without using paired data. [6] (pix2pix) applies conditional Adversarial Networks for image to image style transformation. They penalize on the $\mathcal{L}_1$ distance between the generated image by the conditional GAN and the real image. For this model, paired data is needed for training and it's not easy to get paired data. The Cycle-GAN is built based on pix2pix [6] framework. Cycle-GAN solved this problem by introducing a Cycle Consistency Loss: the transformation between two sets of images should be reversible.

4.1 Model Definition

Suppose we are to learning the mappings between two sets of images $\{x_i\} \in X$ and $\{y_i\} \in Y$. And we denote two mappings as $F : X \to Y$ and $G : Y \to X$. We denote the discriminators for adversarial network as D_X and D_Y, which respectively tells X from $G(Y)$ and Y from $F(X)$. We have two requirements on the transformed image: First, $G(Y)$ should generate images that resemble those in X and $F(X)$ should generate images that resemble those in Y; Secondly, the transformed image should still preserve some features of the original one.

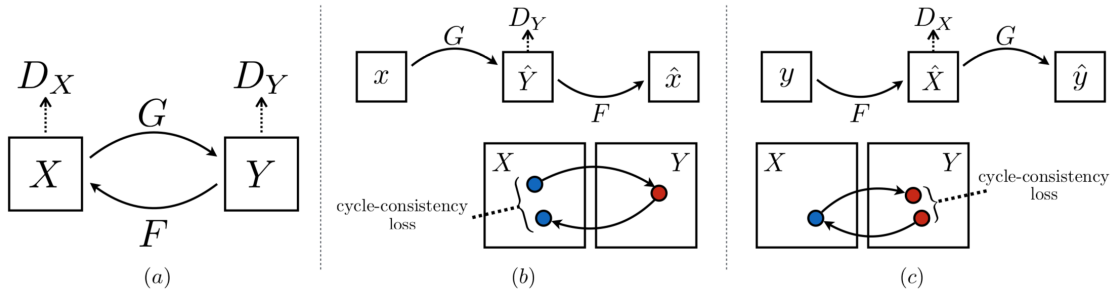

Figure 4.1: The model contains two mappings: G: $X \rightarrow Y$ and F: $Y \rightarrow X$, and two adversarial discriminators D_Y and D_X.

The full objective can be expressed as

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

where $L_{cyc}(G, F)$ is the cycle consistency loss. The goal is to solve G, F, D_X, D_Y such that

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

The adversarial loss L_{GAN} is the same as the loss in pix2pix:

$$L_{GAN}(G, D_X, X, Y) = E_x[logD_X(X)] + E_y[1 - logD_X(G(Y))]$$
$$L_{GAN}(F, D_Y, X, Y) = E_y[logD_Y(Y)] + E_x[1 - logD_Y(F(X))]$$

The training process of Cycle-GAN is similar to that of a regular GAN. We start with training the discriminator first, then freeze discriminator and train Generator for each iteration. Note that instead of understanding Cycle-GAN as two GANs, we can also view it as two auto-encoders. We are learning an "encoder" $F \cdot G$ and another "encoder" $G \cdot F$. The cycle-consistency loss enforces that $G(F(X)) \approx X$ and $F(G(Y)) \approx Y$.

One advantage of introducing cycle-consistency loss is that it prevents mode collapse. The network has to generate image not only looks like a specific set of images but also preserves the feature of the original image.

4.2 Other Models inspired by cycle-consistency

By introducing cycle-consistency, a lot of problems that originally can only be trained on paired data are free of that constraint. Disco-GAN [7], Dual-GAN [18], Star-GAN [2] can also be applied to unpaired image-to-image style translation with slightly different applications.

DualGAN[18]: Dual-GAN adopts the same network architecture as pix2pix, which is a symmetric U-Net as the generator. DualGAN also use WGAN instead of GAN for training.

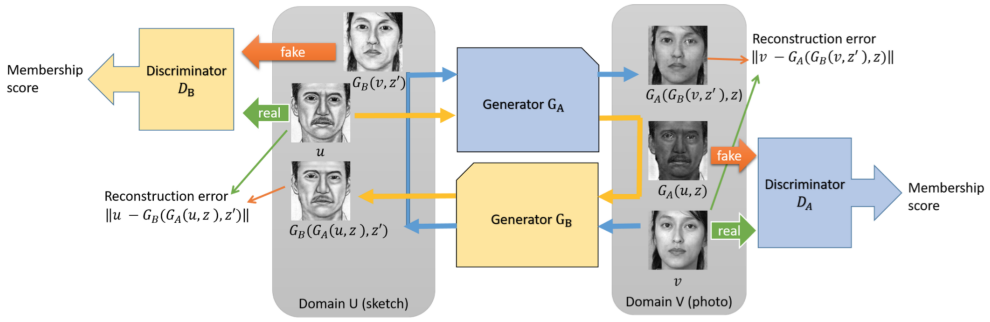

Figure 4.2: Network architecture and data flow chart of DualGAN for image-to-image translation

DiscoGAN[7] :The main difference between DiscoGAN and CycleGAN is the penalty on cycle-consistency. Unlike other methods, Disco-GAN penalizes on the l_2. Usually, l_1 loss works better than l_2 in that it generates images with higher resolution and l_1 loss penalize harder on the cycle inconsistency. One interesting example of Disco-GAN is that they create a corresponding pair of shoes based on the input handbag image.

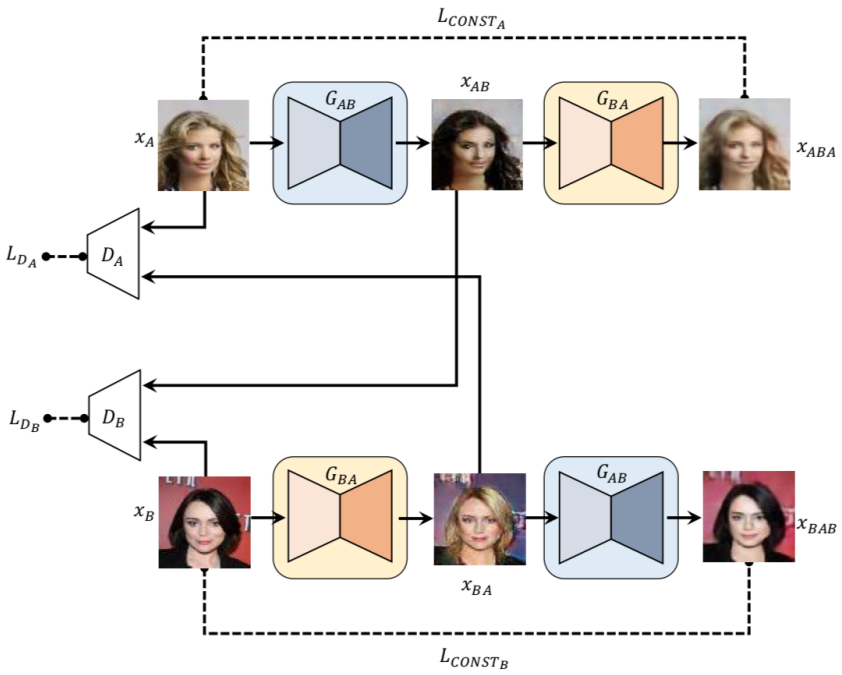

Figure 4.3: Network architecture of DiscoGAN for cross domain image translation

Star-GAN[2]: A star-GAN can be viewed as a particular type of Cycle-GAN that transform an image from one domain to multiple domains. Say we have four different domains of images, we need to train $4 \times (4-1) = 12$ different models if we use CycleGAN. The natural idea is to train a model that only translates images from one domain to all other domains. StarGAN's name comes from the Star-like shape of the network.

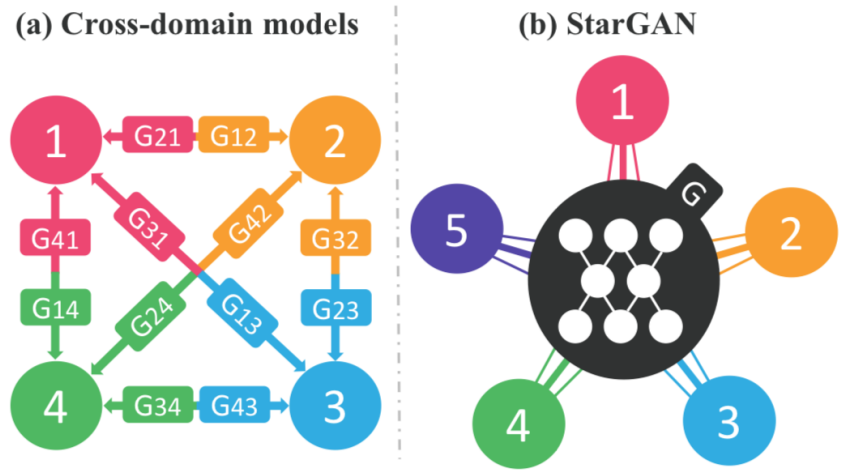

Figure 4.4: Difference between traditional cross-domain models and StarGAN: a) cross-domain models train a pair of model for every pair of image domains b) StarGAN is capable of learning mappings among multiple do- mains using a single generator.

Co-GAN[11] Co-GAN learns two GAN generators for each domain with tied weights on the first few layers for shared latent representation.

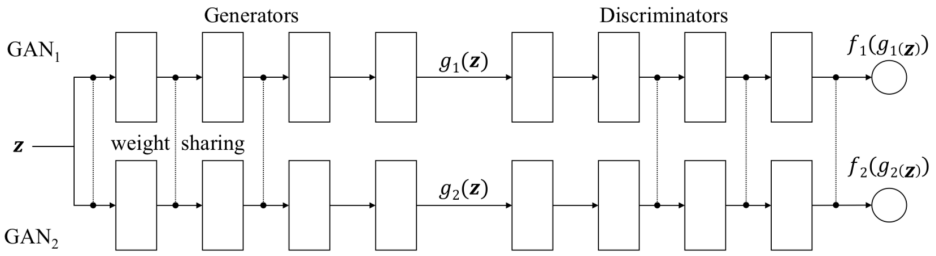

Figure 4.5: Network structure of CoGAN

CHAPTER 5

UNIT: Image-to-Image Translation Networks based on VAE-GAN

5.1 Variational Auto-encoders (VAEs)

Variational Autoencoders (VAEs) [8], like what the name indicates, is a generative model that automatically encode the input data. a variational autoencoder consists of an encoder, a decoder, and a loss function. The encoder is a neural network that takes in the data and outputs a hidden representation z that is usually of a much smaller dimension. The decoder is another neural network whose input is some random vector z. The decoder reconstructs the data using the input vector.

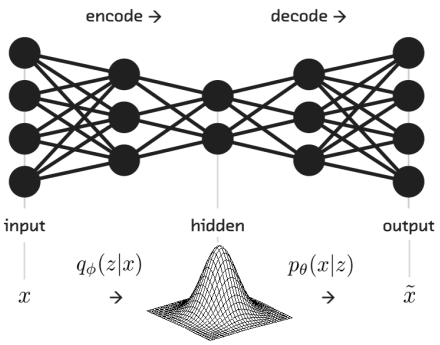

Figure 5.1: Example of a Variational Autoencoder

In probability terms, the encoder network serves as an inference network that parametrizes the approximate posterior of the latent variables z. The inference network outputs parameters to the distribution $Q_\phi(z|x)$. Similarly, a decoder network outputs parameters to the likelihood distribution $p_\theta(x|z)$. A VAE learns to approximate the intractable posterior $p_\theta(z|x)$ with $Q_\phi(z|x)$ by optimizing the variational lower bound:

$$\mathcal{L}(\theta, \phi, x) = -D_{KL}[Q_\phi(z|x)|p_\theta(z|x)] + \mathbb{E}_{Q_\phi(z|x)}[logp_\theta(x|z)]$$

However, we can not conduct traditional back-propagation on the latent variable z to minimize the object. In this chapter, we first go through Evidence lower bound(ELBO), which is important in forming the objective of Variational Autoencoder, then an important trick that allows us to perform variational inference via stochastic backpropagation.

5.1.1 Evidence Lower Bound

In the variational family of algorithms, inference is to cast as an optimization problem using variational calculus. Suppose we are given an intractable posterior probability distribution $P_\theta(Z \mid X)$, variational techniques will try to solve an optimization problem over a family of tractable distributions $Q_\phi(Z \mid X)$

Assume we choose $Q_\phi(Z \mid X)$, say Gaussian family, then we want to adjust the parameters ϕ such that some measure of divergence between $Q_\phi(Z \mid X)$ and $P_\theta(Z \mid X)$ is minimized. Mean-field variational Bayes employs the reverse Kullback-Leibler divergence as such divergence metric between two distributions.

Kullback-Leibler divergence is defined as

$$D_{KL}(Q\|P) = \int q(z \mid x) \log \frac{q(z \mid x)}{p(z \mid x)} dx$$

where Q and P denote two probability distributions. This measure of divergence allows us to formulate variational inference as optimization problem.

Let $Q_\phi \in \mathcal{F}$ denote a distribution in family $\mathcal{F}$ with variational parameters ϕ and P_θ a target distribution with fixed θ. Then

$$Q_\phi = \underset{\phi}{argmin}\, D_{KL}(Q_\phi \| P_\theta).$$

We substitute the definition of the conditional distribution and distribute

$$\begin{aligned}
D_{KL}(Q_\phi \| P_\theta) &= \int q_\phi(z \mid x) \log \frac{q_\phi(z \mid x)}{p_\theta(z \mid x)} dz \\
&= \int q_\phi(z \mid x) \log \frac{q_\phi(z \mid x) p_\theta(x)}{p_\theta(z, x)} dz \\
&= \int q_\phi(z \mid x) \left(\log \frac{q_\phi(z \mid x)}{p_\theta(z, x)} + \log p_\theta(x) \right) dz \\
&= \int q_\phi(z \mid x) \log \frac{q_\phi(z \mid x)}{p_\theta(z, x)} dz \\
&\quad + \log p_\theta(x) \int q_\phi(z \mid x) dz \\
&= \int q_\phi(z \mid x) \log \frac{q_\phi(z \mid x)}{p_\theta(z, x)} dz + \log p_\theta(x).
\end{aligned}$$

$$\begin{aligned}
Q_\phi &= \underset{\phi}{argmin}\, D_{KL}(Q_\phi \| P_\theta) \\
&= \underset{\phi}{argmin} \int q_\phi(z \mid x) \log \frac{q_\phi(z \mid x)}{p_\theta(z, x)} dz \\
&= \underset{\phi}{argmin}\, \mathbb{E}_Q \left[\log q_\phi(z \mid x) - \log p_\theta(x \mid z) - \log p_\theta(z) \right] \\
&= \underset{\phi}{argmax}\, \mathbb{E}_Q \left[-\log q_\phi(z \mid x) + \log p_\theta(x \mid z) + \log p_\theta(z) \right] \\
&= \underset{\phi}{argmax}\, \mathbb{E}_Q \left[\log \frac{p_\theta(z)}{q_\phi(z \mid x)} + \log p_\theta(x \mid z) \right] \\
&= \underset{\phi}{argmax}\, \mathcal{L}(\phi).
\end{aligned}$$

Note, if the terms $p_\theta(z)$, $p_\theta(x \mid z)$ and $q_\phi(z \mid x)$ are tractable, then $\mathcal{L}(\phi)$ is comutationally tractable. We can further rewite $\mathcal{L}(\phi)$ into it's commonly used form

$$
\begin{aligned}
\mathcal{L}(\phi) &= \mathbb{E}_Q \left[\log \frac{p_\theta(z)}{q_\phi(z \mid x)} + \log p_\theta(x \mid z) \right] \\
&= \mathbb{E}_Q \left[\log \frac{p_\theta(z)}{q_\phi(z \mid x)} \right] + \mathbb{E}_Q \left[\log p_\theta(x \mid z) \right] \\
&= \mathbb{E}_Q \left[\log p_\theta(x \mid z) \right] - \mathbb{E}_Q \left[\log \frac{q_\phi(z \mid x)}{p_\theta(z)} \right] \\
&= \mathbb{E}_Q \left[\log p_\theta(x \mid z) \right] - D_{KL} \left(q_\phi(z \mid x) \| p_\theta(z) \right).
\end{aligned}
$$

In the literature, $\mathcal{L}(\phi)$ is referred to as *variational lower bound*.

$$
\log p_\theta(x) - \mathcal{L}(\phi) = D_{KL}(Q_\phi \| P_\theta)
$$

which by non-negativity of $D_{KL}(Q_\phi \| P_\theta)$ yields

$$
\log p_\theta(x) \leq \mathcal{L}(\phi)
$$

5.1.2 Black-Box Variational Inference

Recall that in the Variational Inference we aim to maximize the ELBO

$$
\mathcal{L}(\phi, \theta) = \mathbb{E}_Q \left[\log p_\theta(x \mid z) \right] - D_{KL} \left(q_\phi(z \mid x) \| p_\theta(z) \right) \geq \log p_\theta(x)
$$

over the space of all q_ϕ. The ELBO satisfies

$$
\log p_\theta(x) = D_{KL}(q_\phi(z \mid x) \| p_\theta(z \mid x)) + \mathcal{L}(\phi, \theta)
$$

In order to optimize over $q_\phi(z \mid x)$ we maximize the ELBO over ϕ by gradient descent. Therefore, $q_\phi(z \mid x)$ must be differentiable with respect to ϕ. In contrast to traditional Variational Inference, we perform gradient descent jointly over both ϕ and θ. This will have two effects: (1) Optimization over ϕ will ensure that the ELBO stays close to $\log p(x)$, (2) Optimization over θ will increase the lower bound and thus $\log p(x)$.

To perform black-box variational inference, we need to compute the gradient on the ELBO

$$\nabla_{\theta,\phi}\mathbb{E}_Q\left[\log p_\theta(x,z) - \log q_\phi(z)\right].$$

In most cases we cannot express the expectation with respect to q_ϕ in closed form. Instead, we may draw Monte-Carlo samples from q_ϕ to estimate the gradient by Ergodicity Theorem. For the gradient with respect to p_θ we change the order of gradient and expectation and estimate by Monte-Carlo

$$\mathbb{E}_Q\left[\nabla_\theta \log p_\theta(x,z)\right] \overset{MC}{\approx} \frac{1}{N}\sum_{i=1}^{N} \nabla_\theta \log p_\theta(x,z_i)$$

5.1.3 Reparameterization Trick

As we have seen earlier, maximizing the ELBO requires a good estimate of the gradient.[17] provides a low-variance gradient estimator based on the so-called *reparametrization trick*. Assume we can express $q_\phi(z \mid x)$ as a two-step generative process: (1) Sample noise $\epsilon \sim p(\epsilon)$ from a simple distribution e.g. $\mathcal{N}(0, I)$, (2) apply a deterministic transformation $g_\phi(\epsilon, x)$ that shapes the random noise into an arbitrary distribution $z = g_\phi(\epsilon, x)$.

As an example, we can apply this formulation to Gaussian random variables:

$$z \sim q_{\mu,\sigma}(z) = \mathcal{N}(\mu,\sigma),$$
$$z = g_{\mu,\sigma}(\epsilon) = \mu + \epsilon \odot \sigma$$

where $\odot$ denotes the Hadamard product.

Then we may change the order of expectation and gradient in (**??**) by applying this reformulation. More generally, for any f it holds

$$\begin{aligned}\nabla_\phi \mathbb{E}_{z\sim q_\phi(z|x)}\left[f(x,z)\right] &= \nabla_\phi \mathbb{E}_{\epsilon\sim p(\epsilon)}\left[f(x, g_\phi(z,\epsilon))\right] \\ &= \mathbb{E}_{\epsilon\sim p(\epsilon)}\left[\nabla_\phi f(x, g_\phi(z,\epsilon))\right] \\ &\overset{MC}{\approx} \frac{1}{N}\sum_{i=1}^{N} \nabla_\phi f(x, g_\phi(z,\epsilon)).\end{aligned}$$

Finally, we may use Monte-Carlo integration to estimate the expectation of the gradient with low variance [14].

5.2 Problem Definition

Let's say the two domains are X and Y. We assume that images from two domains follow a joint distribution of $P_{X,Y}(x, y)$. Since we don't have paired images, what we observe is just the marginal distribution of each domain, namely $P_X(x)$ and $P_Y(y)$. The shared latent space assumption suggests that there exist a z such that we can use z to recover a image in both domains. We define the mapping from z to x and y as G_X and G_Y, that is,

$$x = G_X(z), y = G_Y(z)$$

We can also infer the latent space z from x and y via some encoders E_X and E_Y,

$$z = E_X(x), z = E_Y(y)$$

Recall that our goal is to translate image x from domain X to domain Y. Given an image x, we can first encode x using the encoder E_X and then decode the latent vector using G_Y,

$$F_{X\to Y}(x) = G_Y(E_X(x))$$

$$F_{Y\to X}(y) = G_X(E_Y(y))$$

We can also define cycle-consistency over $F_{X\to Y}$ and $F_{Y\to X}$,

$$x = F_{Y\to X}(F_{X\to Y}(x)) = G_X(E_Y(G_Y(E_X(x)))) = G_X(E_Y(y)) = x$$

$$y = F_{X\to Y}(F_{Y\to X}(y)) = G_Y(E_X(G_X(E_Y(y)))) = G_Y(E_X(x)) = y$$

We can see from above that the shared-latent space assumption implies the cycle-consistency assumption. The shared-latent space assumption is stronger than cycle-consistency.

5.3 Shared weight VAE-GAN for Image-to-Image translation

The framework of the model, as illustrated in the last chapter, is based on variational auto-encoders(VAEs) and generative adversarial networks (GANs). It consists of two encoder networks, two generator networks and

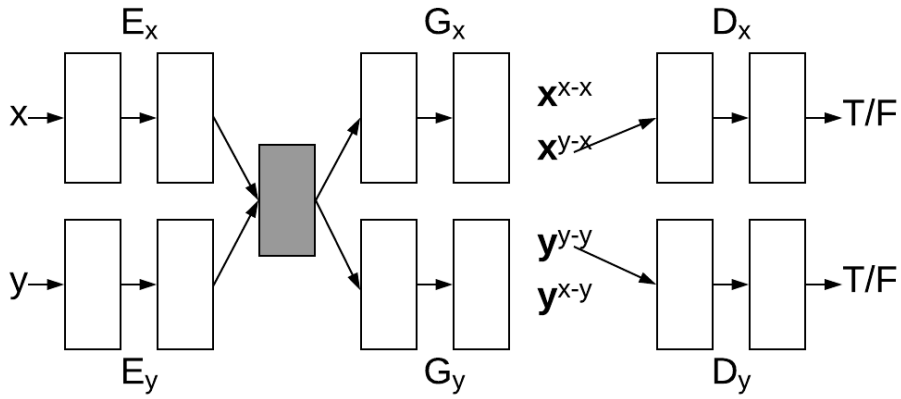

Figure 5.2: Illustration of the VAE-GAN model that share latent space

As illustrated in the figure above, if we ignore the shared latent space (the gray block), then it is exactly two variational auto-encoders. The loss can be decomposed into six parts: VAE_X, VAE_Y,GAN_X, GAN_Y and two cycle consistency loss.

$$\min_{E_X,E_Y,G_X,G_Y} \max_{D_X,D_Y} \mathcal{L}_{VAE_X}(E_X, G_X) + \mathcal{L}_{VAE_Y}(E_Y, G_Y) + \mathcal{L}_{GAN_X}(E_X, G_X, D_X)$$
$$+ \mathcal{L}_{GAN_Y}(E_Y, G_Y, D_Y) + \mathcal{L}_{CC_1}(E_X, G_X, E_Y, G_Y)$$
$$+ \mathcal{L}_{CC_2}(E_X, G_X, E_Y, G_Y)$$

Although the shared latent space assumption already implies cycle consistency, we still add the cycle consistency loss to further regularize the ill-posed unsupervised image-to-image translation problem. The cycle consistency loss is calculated using a VAE-like objective function

$$\mathcal{L}_{CC_1}(E_X, G_X, E_Y, G_Y) = KL(q_x(z_1|x)|p_z(z)) + KL(q_y(z_2|x^{x\to y})|p_z(z))$$
$$+ \lambda \mathbb{E}_{z\sim q_y(z_2|x^{x\to y})} log p_{G_1}(x|z_2)$$

The first two KL divergence terms penalize on $q_x(z_1|x)$ deviating from the prior distribution while the third term guarantees that the transformation from $X \to Y \to X$ resembles the identity mapping.

CHAPTER 6

Experiments and Results

In this section, we conducted experiments of image-to-image translation using two models discussed in the previous two chapters. Both models took a long time to train.

6.1 Datasets

We apply the models to cartoonize real people. The anime images we use is a subset of Danbooru2017 dataset, which is a large-scale crowdsourced and tagged anime illustration dataset. This dataset is a collection of images uploaded by users who are part of a highly active community tagged by different keywords. The training set consist of 3,528 anime images randomly sampled from Danbooru2017.

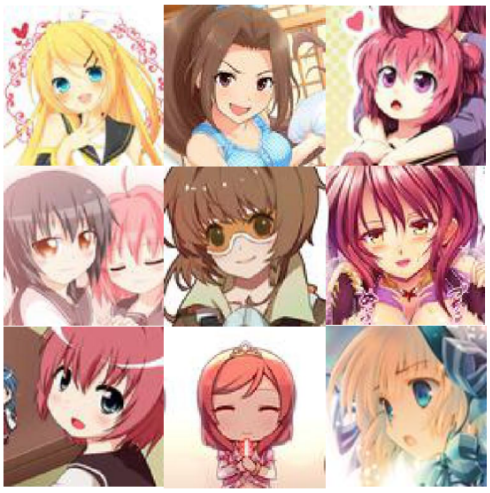

Figure 6.1: Examples of images in Danbooru2017 dataset

Another dataset we use is MS-Celeb-1M[5], a large scale real world face image dataset to public. Again, we randomly sampled 5,866 images from this dataset as training data.

Figure 6.2: Examples of images in MS-Celeb-1M dataset

6.2 Implementation

6.2.1 Implementation of CycleGAN

The input images are all resized to 256×256 before training. The generator network starts with two convolution layers with strides two. Then the network is followed by 9 residual blocks, and two fractionally-strided convolutions with stride $\frac{1}{2}$. The discriminator consists of three CONV-RELU layers which classify whether the image belongs to a certain domain or not.

6.2.2 Implementation of UNIT

For the network architecture of UNIT, the encoders consisted of 3 convolutional layers and 4 basic residual blocks. The final layer of two encoders is shared. The generators consisted of 4 basic residual blocks and 3 transposed convolutional layers. The first layer of the generator is shared. The discriminators comprised six convolutional layers. For all the networks, LeakyReLU is used for nonlinearity.

6.3 Results

The first two images are image translation using CycleGAN. The photo images on the lower part are the original images, and CycleGAN generates the upper ones. We can see that CycleGAN well preserves the structure feature of the original images. There is almost no difference in the coloring and the texture. It seems that CycleGAN can identify where eyes are and turn them into cartoonish eyes.

Figure 6.3: Examples of photo to anime results by Cycle-GAN: lower: testing images from MS-Celeb-1M; upper: anime images generated by CycleGAN

Figure 6.4: Examples of photo to anime results by Cycle-GAN: lower: testing images from MS-Celeb-1M; upper: anime images generated by CycleGAN

Despite the amazing results above, CycleGAN does not always perform perfectly. The figure below is an example of failure. Due to lack of side faces in the training set, the CycleGAN turned a side face to hair by mistake.

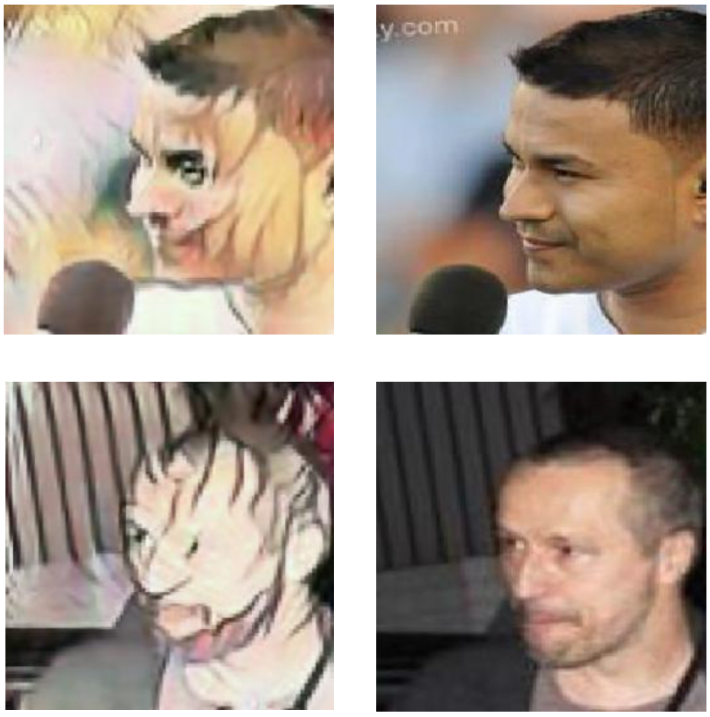

Figure 6.5: CycleGAN turned a side face to hair by mistake

The results below are results generated by UNIT. Note that unlike CycleGAN, UNIT outputs indeterministic results. Below are three different sets of results generated. For each set of them, the sampled latent space is the same. We can see from the results that, within the same group, UNIT tends to generate images with similar coloring style and face features.

Figure 6.6: Examples of photo to anime results by UNIT: upper: Original image from MS-Celeb-1M; middle: Reconstructed Face Image via VAE; lower: transformed anime images based on UNIT

Figure 6.7: Examples of photo to anime results by UNIT: upper: Original image from MS-Celeb-1M; middle: Reconstructed Face Image via VAE; lower: transformed anime images based on UNIT

Figure 6.8: Examples of photo to anime results by UNIT: upper: Original image from MS-Celeb-1M; middle: Reconstructed Face Image via VAE; lower: transformed anime images based on UNIT

Figure 6.9: Examples of anime to photo results by UNIT: upper: Original image from danbooru2017; middle: Reconstructed Anime Image via VAE; lower: transformed face images based on UNIT

Figure 6.10: Examples of anime to photo results by UNIT: upper: Original image from danbooru2017; middle: Reconstructed Anime Image via VAE; lower: transformed face images based on UNIT

Figure 6.11: Examples of anime to photo results by UNIT: upper: Original image from danbooru2017; middle: Reconstructed Anime Image via VAE; lower: transformed face images based on UNIT

We evaluate the empirical result depicted in Figure 6.9 to Figure 6.11. As a result of VAE, UNIT can fill in different facial features, and the generated image coloring vary from sample to sample. We can conclude that the shared latent space learned by two VAEs well encodes the face features. Similar to CycleGAN, UNIT preserves the structure of the original image during image to image translation.

Compared to CycleGAN, UNIT can better identify face and hair. But the translated image is less satisfactory. One disadvantage of using cycle consistency is that it is hard to change the shape of the image (due to l_1 loss). Another limitation of CycleGAN is that it only learns one-to-one mappings.

CHAPTER 7

Conclusion and Future work

We compared two models for image-to-image translation using CycleGAN and UNIT. The experiments showed that both models could translate an image from one domain to another without any paired images in two domains. However, one limitation of CycleGAN is that: The mapping is deterministic based on the input image. For the UNIT model, the model is restricted due to the Gaussian latent space assumption, which is a possible reason for some abnormal translated images. Both models are hard to train because of the saddle point searching problem.

The cycle consistency idea provides a powerful tool for unpaired image-to-image translation. But the transformation is only limited to color, texture and slight changes in shape.

We plan to address these issues in the future work. Say we can apply more sophisticated GAN structure, say WGAN[1]; We can also add stochastic input for one-to-many mapping when applying CycleGAN; Cycle consistency is an example of preserving information during translation, we can also research on another kind of consistency and loss functions which can generalize to more general problems.

REFERENCES

[1] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein gan. *arXiv preprint arXiv:1701.07875*, 2017.

[2] Y. Choi, M. Choi, M. Kim, J.-W. Ha, S. Kim, and J. Choo. StarGAN: Unified Generative Adversarial Networks for Multi-Domain Image-to-Image Translation. *ArXiv e-prints*, November 2017.

[3] Leon A. Gatys, Alexander S. Ecker, and Matthias Bethge. Image style transfer using convolutional neural networks. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[4] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. Improved training of wasserstein gans. In *Advances in Neural Information Processing Systems*, pages 5769–5779, 2017.

[5] Y. Guo, L. Zhang, Y. Hu, X. He, and J. Gao. MS-Celeb-1M: A Dataset and Benchmark for Large-Scale Face Recognition. *ArXiv e-prints*, July 2016.

[6] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. *arXiv preprint*, 2017.

[7] T. Kim, M. Cha, H. Kim, J. K. Lee, and J. Kim. Learning to Discover Cross-Domain Relations with Generative Adversarial Networks. *ArXiv e-prints*, March 2017.

[8] D. P Kingma and M. Welling. Auto-Encoding Variational Bayes. *ArXiv e-prints*, December 2013.

[9] Christian Ledig, Lucas Theis, Ferenc Huszár, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, et al. Photo-realistic single image super-resolution using a generative adversarial network. *arXiv preprint*, 2017.

[10] Chuan Li and Michael Wand. Precomputed real-time texture synthesis with markovian generative adversarial networks. In *European Conference on Computer Vision*, pages 702–716. Springer, 2016.

[11] M.-Y. Liu and O. Tuzel. Coupled Generative Adversarial Networks. *ArXiv e-prints*, June 2016.

[12] Ming-Yu Liu, Thomas Breuel, and Jan Kautz. Unsupervised image-to-image translation networks. In *Advances in Neural Information Processing Systems*, pages 700–708, 2017.

[13] D. Pathak, P. Krahenbuhl, J. Donahue, T. Darrell, and A. A. Efros. Context Encoders: Feature Learning by Inpainting. *ArXiv e-prints*, April 2016.

[14] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. *arXiv preprint arXiv:1401.4082*, 2014.

[15] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen. Improved Techniques for Training GANs. *ArXiv e-prints*, June 2016.

[16] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. In *Advances in Neural Information Processing Systems*, pages 2234–2242, 2016.

[17] Tim Salimans and Diederik P Kingma. Weight normalization: A simple reparameterization to accelerate training of deep neural networks. In *Advances in Neural Information Processing Systems*, pages 901–909, 2016.

[18] Z. Yi, H. Zhang, P. Tan, and M. Gong. DualGAN: Unsupervised Dual Learning for Image-to-Image Translation. *ArXiv e-prints*, April 2017.

[19] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. ICCV, 2017.