Extension of GAN and VAE to model Multimodal Distribution

Yijie Zhao

University of Pennsylvania
zhaoyij@seas.upenn.edu

Abstract. Many image-to-image translation models only support a one-to-one or many-to-one mapping between the input and output domain, which is oftentimes ambiguous given the set of potential images that an input image can be translated to in many image translation tasks such as gray-scale to RGB conversion. In this project, we reproduce BicycleGAN, which models a *distribution* of possible outputs in a conditional generative modeling setting by learning a bijection between a low-dimensional latent space and the output space. We evaluate the model performance both qualitatively and quantitatively using metrics such as FID score and LPIPS against the a baseline CycleGAN model that learns a forward and inverse mapping between the two domains that the unpaired input images reside. We notice a clear improvement in both photo-realisticity and diversity among the images generated by BicycleGAN.

Keywords: Multimodal image-to-image translation, Generative Adversarial Networks (GAN)

1 Introduction

Image-to-image translation is a class of vision and graphics problems that aims to learn a mapping between an input image and an output image. Similar to how a script can be written in different languages but convey the same message, an image can also be rendered in various formats, such as graphically in edge map, RGB, or gray-scale; geometrically in 2D or 3D; and semantically as a labeled map, etc. Different types of neural network architectures are used to study this topic, among which GANs have been one of the most popular choices. For example, CycleGAN trains unsupervised image translation models via the GAN architecture using unpaired collections of images from two different domains [13]. Pix2Pix, as another example, is an image-to-image translation GAN model that uses paired training samples and investigates with various different configurations within the GAN model [5]. There are also many other GAN models that target the translation of images rendered in certain specific formats [1] [2] [3].

However, most techniques aforementioned focus only on generating a single result. That is, one input image often only gets mapped to a single output image. As seen in Figure 1, in many scenarios considered in image-to-image translation tasks, there can be many possible output images translated from a given

input image. In light of this observation, in this project we aim to reproduce BicycleGAN, which models a *distribution* of potential outputs in a conditional generative modeling setting with the goal of producing results that are both photorealistic and diverse [14].

BicycleGAN learns a mapping from a high-dimensional input to a distribution of high-dimensional outputs by learning a lower-dimensional latent space that encodes the multimodality of the output space, thus drastically reducing the difficulty of the learning problem. To ensure diversity of the mapped output space, the model encourages a bijection between the latent and output space by jointly learning an encoder that maps generated outputs back to the latent space in addition to learning a generator that does the reverse mapping from the latent space (plus inputs) to the output space. At inference time, the generator takes in an input image as well as a random sample from the latent space and returns an output image. The two main components of BicycleGAN, `cVAE-GAN` (Conditional Variational Autoencoder GAN) and `cLR-GAN` (Conditional Latent Regressor GAN) that together achieve the functionalities of BicycleGAN aforementioned will be discussed in detail in later sections.

We perform quantitative evaluations on the model performance using metrics including FID score and LPIPS [1]. We also perform qualitative evaluations through demonstrations of various generated outputs from the same input. Our code is available at https://github.com/lisazhao9897/CIS680BicycleGAN.

Fig. 1. Multimodal image-to-image translation using BicycleGAN: given the edge map of a shoe, BicycleGAN models a distribution of potential RGB images of the corresponding shoe. A few selected outputs are displayed above.

[1] FID is a measure of similarity between two datasets of images. LPIPS evaluates the distance between image patches.

2 Related Work

Pix2Pix uses a conditional GAN (cGAN) model to learn a function that maps from an input image to an output image. It is the starting block that BicycleGAN builds upon and has previously shown to produce good results for many image-to-image translation tasks [5]. More specifically, Pix2Pix trains a generator that is conditioned on the input image using two losses: a regression loss between the generated output and the ground truth image as well as a learnt discriminator loss that encourages photo-realisticity.

Similar to BicycleGAN, Pix2Pix is generic and not speicific to certain classes of images. However, as seen in Table 1, unlike BicycleGAN, Pix2Pix is not multi-modal. That is, the model can only map an input image to a single output. As discussed in [14], when an additional latent code z randomly drawn from a simple distribution is fed to the generator in addition to the input image as an attempt to transform the model towards multi-modal, it is observed that there is oftentimes very little incentive for the generator to actually make use of the noise vector z. Thus, the network is unable to achieve effective multi-modality. BicycleGAN, on the other hand, learns a bijection in between the latent and output space, thus forcing the generator to take advantage of the latent code and the encoder to embed information into the latent space.

One advantage of Pix2Pix over BicycleGAN is that the training process of the latter is slower as it learns to discover multiple modes.

CycleGAN is an unsupervised image translation model via the GAN structure using unpaired collections of images from two different domains [13]. More specifically, the model learns a mapping $G : X \rightarrow Y$ such that the distribution of images from $G(X)$ becomes indistinguishable from the distribution Y using an adversarial loss. To ensure that the learnt mapping would not ignore the input and simply map all to a single element in Y, the model also learns an inverse mapping $F : Y \rightarrow X$ and introduces cycle consistency losses to encourage $F(G(X)) \approx X$ (and vice versa) as shown in Figure 2.

Compared to BicycleGAN, CycleGAN as an unsupervised algorithm requires only unpaired images from two domains, which is a less strict requirement as compared to that of the prior. However, similar to Pix2Pix, CycleGAN is also uni-model and maps an input image to only a single output as seen in Table 1.

Unsupervised Image-to-Image Translation Networks (UNIT) is an unsupervised image translation model also via the GAN structure using unpaired collections of images from two domains [6]. As seen in Figure 3, the model makes the assumption that each domain has an auto-encoder that can encodes an input image into a latent code in a latent space this is assumed to be shared across the two domains such that two analogous images are mapped to the same latent

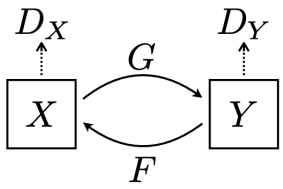


Fig. 2. CycleGAN contains two mapping functions $G : X \rightarrow Y$ and $F : Y \rightarrow X$ and associated adversarial discriminators D_X and D_Y.

code. Thus, to transfer an image from one domain to another, we can simply encode it with the encoder of the source domain and denote it with the decoder of the target domain.

As compared to CycleGAN aforementioned, UNIT has a stronger assumption: while the prior assumes only cycle consistency, the latter assumes a shared latent space, which is a lot stronger. As compared to BicycleGAN, while UNIT is unsupervised and does not require paird images for training, it is not multi-modal and is unable to map an input image to multiple outputs as BicycleGAN.

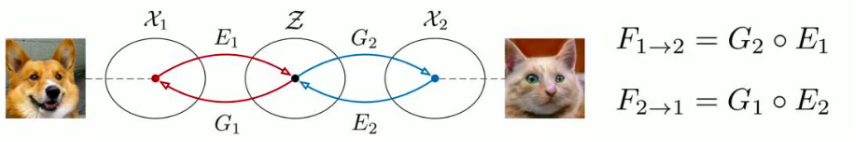


Fig. 3. UNIT architecture with shared latent space

Method	BicycleGAN [14]	Pix2Pix [5]	CycleGAN [13]	UNIT [6]
Multi-modal	✓			
Unsupervised			✓	✓

Table 1. Comparisons between several GAN-based image translation models

3 Method

BicycleGAN consists of two components: Conditional Variational Autoencoder GAN (`cVAE-GAN`) and Conditional Latent Regressor GAN (`cLR-GAN`). In this section, we will first briefly discuss the architecture and objective functions of each component and how BicycleGAN as a hybrid model combines the two. Then

we will dive deeper into detailed architectures of the sub-modules (generators, discriminators, and encoders) used in each component. Lastly, we will highlight some of the key training details.

3.1 cVAE-GAN

As seen in Figure 4, `cVAE-GAN` first encodes the ground truth image B into the latent space using encoder **E**. Then generator **G** takes in an input image A as well as the latent code Z, which is the encoded ground truth image, returning a generated output image $\hat{B}$. The flow of `cVAE-GAN` is $B \rightarrow Z \rightarrow \hat{B}$.

Objective functions associated with `cVAE-GAN` are three-folds:

$$\mathcal{L}_1^{image}(G) = \mathbb{E}_{A,B\sim p(A,B),z\sim E(B)} \left\| B - G(A,z) \right\|_1 \tag{1}$$

$$\begin{aligned} \mathcal{L}_{GAN}^{VAE}(D,G,E) = {} & \mathbb{E}_{A,B\sim p(A,B)}[log(D(A,B))] \\ & + \mathbb{E}_{A,B\sim p(A,B),z\sim E(B)}[log(1 - D(A,G(A,z)))] \end{aligned} \tag{2}$$

$$\mathcal{L}_{KL}(E) = \mathbb{E}_{B\sim p(B)}[\mathcal{D}_{KL}(E(B) \mid\mid \mathcal{N}(0,1))] \tag{3}$$

As seen in Figure 4, $\mathcal{L}_1^{image}$ as described in Equation 1 is an l_1 loss between the target image B and the generated output image $\hat{B}$ that constraints the transformed image $G(A,z)$ to be similar to the target. $\mathcal{L}_{GAN}^{VAE}$ in Equation 2 symbolizes the minimax game played between the generator and discriminator, where the discriminator attempts to differentiate between real and fake (generated) samples, while the generator aims to fool the discriminator. Here, $D(A,\cdot)$ is an indicator function ruling whether the discriminator believes $\cdot$ to be the real image associated with input A. To achieve their respective goal aforementioned, a discriminator aims to maximize $\mathcal{L}_{GAN}^{VAE}$, while a generator aims to minimize it. We will follow the original BicycleGAN paper and build our model on the Least Square GANs (LSGANs) variant [8], which uses a least-squares objective instead of a cross entropy loss. That is, we will use MSE loss for $\mathcal{L}_{GAN}^{VAE}$. Lastly, $\mathcal{L}_{KL}$ in Equation 3 enforces the latent space to be a compact Gaussian distribution, which we can then take samples from at inference time. $\mathcal{L}_{KL}$ could be derived to the following form. The procedure is omitted here since it is provided in the GAN assignment.

$$\mathcal{L}_{KL}(E) = 0.5 \cdot \lambda_{KL} \sum_{zdim} (exp(log(\sigma^2(x))) + \mu(x)^2 - 1 - log(\sigma^2(x))) \tag{4}$$

, where $log(\sigma^2(x))$ is the log variance of the output from the VAE encoder E.

3.2 cLR-GAN

As seen in Figure 5, `cLR-GAN` first passes the input image A as well as a randomly sampled latent vector z through the generator **G**. The generated output $\hat{B}$ is then passed through an encoder **E**, which tries to regain the latent vector from the output image. The flow of `cLR-GAN` is $Z \rightarrow \hat{B} \rightarrow \hat{Z}$.

Objective functions associated with `cLR-GAN` are two-folds:

$$\begin{aligned}\mathcal{L}_{GAN}(D,G) = &\mathbb{E}_{A,B\sim p(A,B)}[log(D(A,B))] \\ &+ \mathbb{E}_{A\sim p(A),z\sim p(z)}[log(1-D(A,G(A,z))]\end{aligned} \tag{5}$$

$$\mathcal{L}_1^{latent}(G,E) = \mathbb{E}_{A\sim p(A),z\sim p(z)} \left\| z - E(G(A,z)) \right\|_1 \tag{6}$$

As seen in Figure 5, $\mathcal{L}_{GAN}$ described in Equation 5 is an adversarial GAN loss similar to Equation 2. It helps ensure the generated images are as photo-realistic as possible. Similar as before, we will use a least-square objective to represent this component of the GAN loss. $\mathcal{L}_1^{latent}$ in Equation 6 is an l_1 loss between the randomly sampled latent code and the Encoder output. It ensures that the generated image $\hat{B}$ can be encoded back into the learned latent space of B, which is a normal distribution after optimizing over the KL divergence loss in `cVAE-GAN` aforementioned. This helps achieve a bijection between the output and latent space and thus encouraging diversity among the output images. It's worth noting that no l_1 loss between ground truth output image B and the generated output $\hat{B}$ is used since the noise vector is randomly drawn, and thus $\hat{B}$ does not necessarily need to replicate exactly the ground truth image.

3.3 BicycleGAN

The objective function of BicycleGAN is a combination of the two components aforementioned. In the first component, while the latent code is produced from ground truth outputs, it may not necessarily result in photo-realistic images to be generated. On the other hand, while the latent code is sampled from a simple distribution in the second component, the generator is not seeing the input-target output image pairs. As a hybrid model, BicycleGAN takes advantages from both GAN models. The overall objective function of BicycleGAN can be summarized as follows:

$$\begin{aligned}G^*, E^* = arg \min_{G,E} \max_{D} &\mathcal{L}_{GAN}^{VAE}(D,G,E) + \lambda \mathcal{L}_1^{image}(G) + \mathcal{L}_{GAN}(D,G) \\ &+ \lambda_{KL}\mathcal{L}_{KL}(E) + \lambda_{latent}\mathcal{L}_1^{latent}(G,E),\end{aligned} \tag{7}$$

where the hyper-parameters λ, λ_{KL}, and λ_{latent} controls the relative importance of each term. We will reveal our choice of these hyper-parameters in later sections.

3.4 Sub-Modules

The sub-modules used in the above two GAN components include generators, discriminators, and encoders. We use the same architecutre for each of the sub-modules across the two components of BicycleGAN.

[2] Figures taken from https://junyanz.github.io/BicycleGAN/

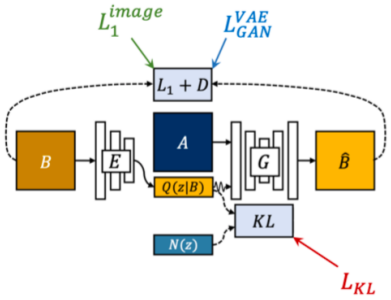


Fig. 4. cVAE GAN architecture

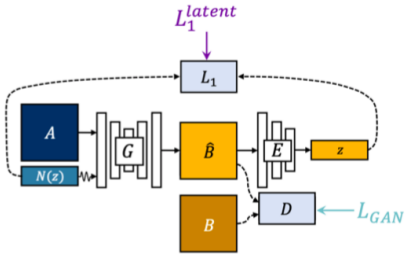


Fig. 5. cLR GAN architecture[2]

Generator For generator G, we follow the original BicycleGAN paper and use U-Net [10], which contains an encoder-decoder architecture with symmetric skip connections. This architecture has been shown to have a good performance in unimodal image-to-image translations, especially when there exists a spatial correspondence in between the input and output pair [14]. As shown in Figure 6, while we maintain the skipped connections across distant layers as well as the contraction and expansion path as the original U-Net architecture, we have also made slight modifications based on our empirical observations on model performances. For instance, we choose to use convolution layers with kernel size $= (4 \times 4)$, stride $= 2$, padding $= 1$; we use leaky ReLU with a slope of 0.2 to connect the double convolution layers; we add in an additional instance normalization layer after all intermediary convolution layers to deal with internal covariate shifts, etc. Again, detailed architecture can be seen in Figure 6.

Discriminator For discriminator D, we follow the original BicycleGAN paper and use PatchGAN, which penalizes structure at the scale of local image patches. It tries to classify if each $N \times N$ patch in an image is real or fake and runs convolutionally across the image, taking the average of all responses as the final output. Here, we use two scales of patches: 30×30 and 14×14. The input image is first passed through a sequence of convolution layers, leaky ReLUs, instance / batch normalization layers and form a 30×30 patch. Simutaneously the first image is also shrunk via an average pooling layer and undergoes the same sequence layer as defore to form another 14×14 patch. A list of patches of different scales is returned as output from the discriminator. Detailed architecture can be seen in Figure 7.

Encoder For encoder E, we use ResNet18 as seen in Figure 8. Through the introduction of "identity shortcut connection" that skips one or more layers, ResNet-18 (with 18 layers) is able to combat the vanishing gradient problem that hinders deep neural networks from learning [4]. The architecture as shown in the figure has been proven to have a well performance in many computer vision tasks and is therefore chosen here as the encoder architecture.

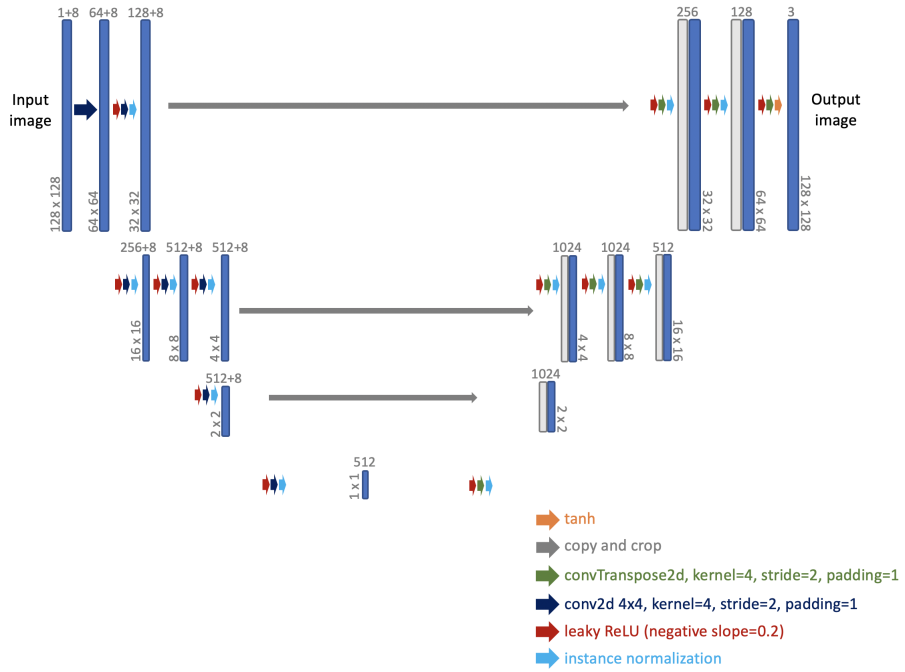


Fig. 6. Modified U-Net architecture for BicycleGAN generator.

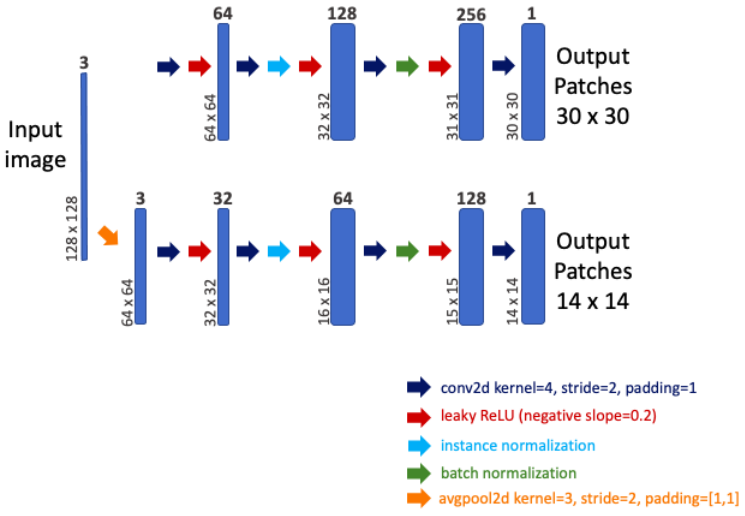


Fig. 7. PatchGAN architecture for BicycleGAN discriminator.

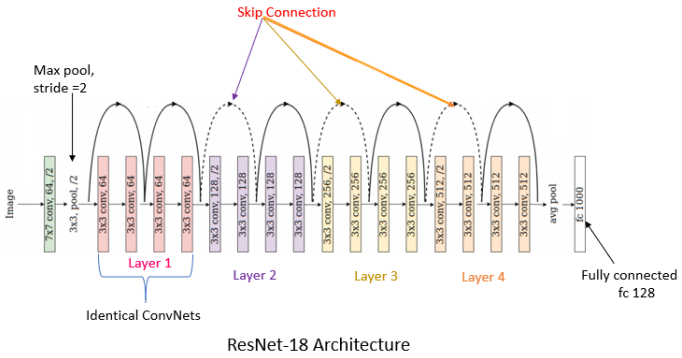


Fig. 8. ResNet18 architecture for BicycleGAN encoder.[3]

3.5 Training Details

Pre-processing we normalize input tensors of shape $(H, W, C) = (128, 128, C)$ in the range of $[-1, 1]$ with $\mu = \sigma = 0.5$. Here $C = 3$ for RGB image tensors and $= 1$ for edge maps.

Hyper-parameters As mentioned before, hyper-parameters λ, λ_{KL}, and λ_{latent} controls the relative importance of each term in the overall loss function. We use $\lambda = 10.0, \lambda_{KL} = 0.01$, and $\lambda_{latent} = 0.5$, same as the original BicycleGAN paper. We choose the dimension of the latent space to be 8.

Training procedure There are four sub-modules for the hybrid BicycleGAN model: encoder, generator, and two discriminators. We choose to use two separate discriminators for the two components `cVAE-GAN` and `cLR-GAN` because it yields slightly better visual results compared to sharing weights [14]. However, weights are shared across the two components for the generator and the encoder. The general flow of computation has been described in detail before.

The inputs of the generator G is a latent code $z \in \mathbb{R}^{zdim} = \mathbb{R}^8$ and a real images A (edge maps in the case of this project) of shape $(128 \times 128 \times 3)$. The output of G is a generated RGB image of shape $(128 \times 128 \times 3)$. The input of the encoder E is a real RGB image of shape $(128 \times 128 \times 3)$ and the output of E is the sufficient statistics of a normal distribution: $\mu, log(\sigma^2) \in \mathbb{R}^{zdim} = \mathbb{R}^8$, using which we can sample a latent code $z = \mu + \epsilon \cdot \sigma$, where the standard deviation σ can be calculated using the log-variance via $\sigma = e^{0.5 \cdot log(\sigma^2)}$. Lastly the input of the discriminator D is an RGB image, either real or fake, of shape $(128 \times 128 \times 3)$. D returns a list of patches of different scales, (30×30) and (14×14). Each patch

[3] Image taken from https://www.pluralsight.com/guides/introduction-to-resnet

is then compared with a target label of either all ones (a 'real' label) or all zeros (a 'fake' label) to calculate the GAN loss, which as mentioned before is an MSE loss since we choose to build out model on the Least Square GANs (LSGANs) [8].

When back-propagating the networks, we first freeze the two discriminators and back-propagating the encoder E and generator G together. The losses that are back-propagated include the l_1 loss on the generated image, corresponding to Equation 1, the KL divergence loss on the latent space as described in Equation 3, and lastly the GAN loss on the generator in both components of Bicycle-GAN, which as described before is an MSE loss between generator G's output images and a target 'real' label. The fed-in target label is all ones ('real') since the goal of the generator is to fool the discriminator. Then we freeze the encoder E and back-propagate the generator G on the l_1 loss for the latent code as described in Equation 6. Lastly, we update the two discriminators based on the MSE GAN loss, which consists of two parts: the loss on real images and the loss of fake images. The discriminator is expected to classify real images as real and fake as fake.

We use an Adam optimizer with an initial learning rate $= 0.0002$ with a scheduled decay by a factor of 0.1 every 100 steps. We set betas $= (0.5, 0.999)$. We choose a batch size of 16 and train on Google Colab for 50 epochs. Each epoch takes roughly $20 - 30$ minutes.

4 Experiments / Analysis

4.1 Visualization of Training

Losses Below are the loss curves across epochs. The losses below include the generator GAN loss and the discriminator GAN loss in `cVAE-GAN` and `cLR-GAN` respectively, corresponding to Equation 2 and Equation 5; KL loss from Equation 3, l_1^{image} loss from Equation 1; l_1^{latent} loss from Equation 6; and finally the total loss, which is the sum of all the loss components aforementioned.

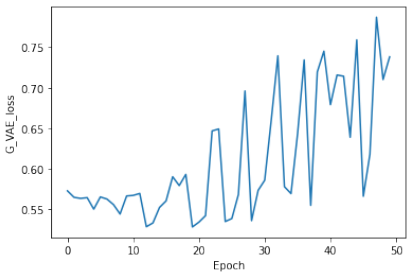


Fig. 9. `cVAE GAN` generator loss

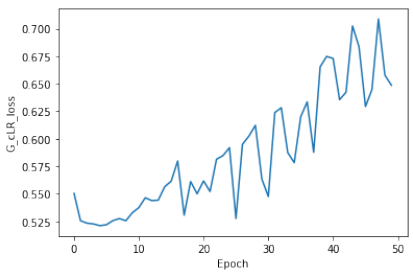


Fig. 10. `cLR GAN` generator loss

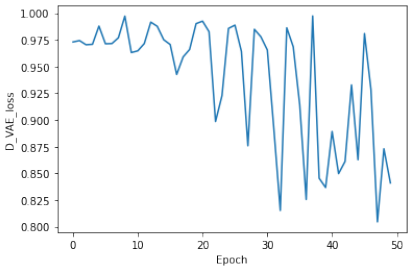


Fig. 11. `cVAE GAN` discriminator loss

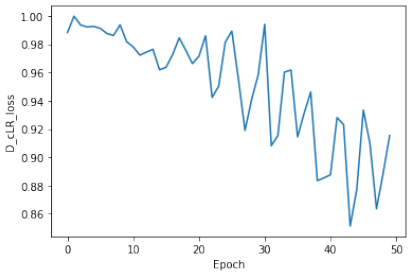


Fig. 12. `cLR GAN` discriminator loss

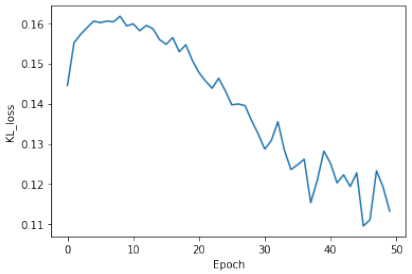


Fig. 13. KL loss

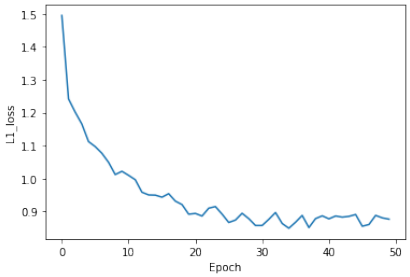


Fig. 14. l_1 image loss

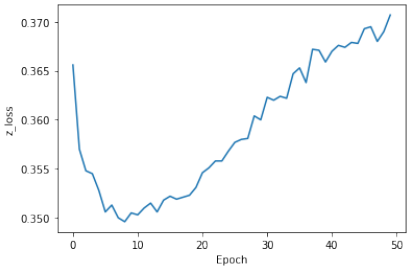


Fig. 15. l_1 latent loss

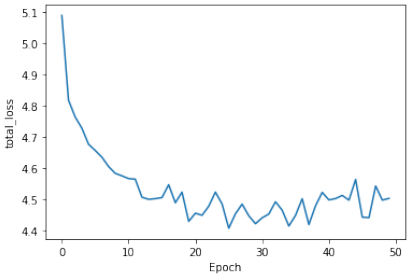


Fig. 16. total loss

.

As seen in Figure 10, 9, 12, 11, which represent the minimax game played in between the generator as well as the discriminator in the classification as well as generation of the output images. We notice clear ups and downs across epochs, which is desirable since it suggests the 'fighting process' between the two components, which can lead to more photo-realistic images generated by the generator. Furthermore, as seen in Figure 13, we notice that the KL divergence between the latent space outputted by the encoder and $\sim \mathcal{N}(0, 1)$ first increases, then decreases. This behavior could be attributed to the initialization being close to the standard normal that we wish to approximate, while the initial weight changes could deviate a bit from the target, eventually the weights will be readjusted, causing the KL divergence loss to decrease. As seen in Figure 14, the l_1^{image} loss decreases consistently, which coincides with the overall good quality of the generated images that we observe. Furthermore, as seen in Figure 15, we notice first a decrease, then an increase on the l_1^{latent} loss, which helps encourage a bijection between the latent and the output space to be learnt. The increase in l_1^{latent} loss could potentially explain the increasingly lack of diversity among the generated images as training progresses.

Generated Images with Fixed Inputs we fix 2 pairs of images as well as 6 latent codes, each $\in \mathbb{R}^{zdim} = \mathbb{R}^8$ and is drawn independently from $\sim \mathcal{N}(0, I)$. For every 10 epochs we feed the fixed set of inputs to the generator and record the output images, which are displayed below. We notice that the generated images have visually become very similar to the real ones starting around the 20^{th} epoch. We also notice that, especially in the case of the second fixed image, the diversity of the set of generated shoes that we visualize seem to decrease, converging towards a similar color as training progresses. This could coincide with the increasing l_1^{latent} loss that we observe in Figure 15, suggesting that the model didn't quite learn a bijection between the latent and output space.

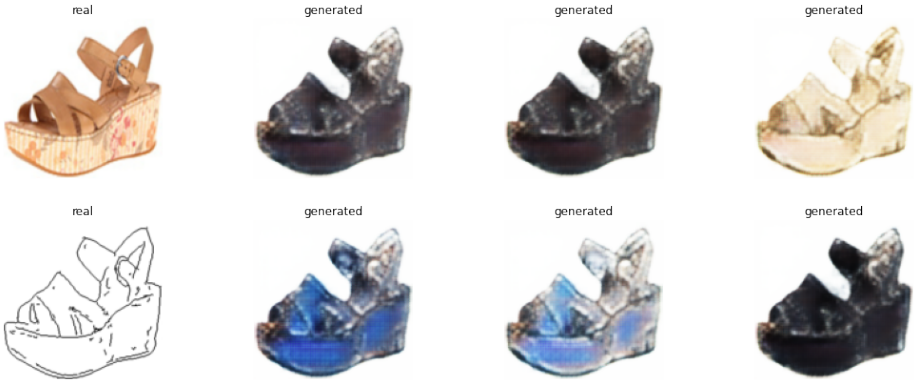


Fig. 17. Image 1 @ epoch 1

Fig. 18. Image 1 @ epoch 11

Fig. 19. Image 1 @ epoch 21

Fig. 20. Image 1 @ epoch 31

Fig. 21. Image 1 @ epoch 41

Fig. 22. Image 1 @ epoch 50

Fig. 23. Image 2 @ epoch 1

Fig. 24. Image 2 @ epoch 11

Fig. 25. Image 2 @ epoch 21

Fig. 26. Image 2 @ epoch 31

Fig. 27. Image 2 @ epoch 41

Fig. 28. Image 2 @ epoch 50

4.2 Quantitative Evaluation

Benchmark Model We use the CycleGAN model trained on the same data set as the BicycleGAN model aforementioned from the GAN assignment as our benchmark for comparison as well as analysis on the performance of BicycleGAN. It is worth noting that the CycleGAN model is trained only for 2 epochs, while the BicycleGAN model is trained for 50 epochs. The difference could result in potential analytical bias.

FID score We will evaluate the photo-realistic quality of the generate images using FID score between three sets of data sets:

1. two sets of real RGB images A, B from the validation set, each of size 100
2. real RGB image set A aforementioned & images generated using **CycleGAN** from the edge map corresponding to A
3. real RGB image set A aforementioned & images generated using **BicycleGAN** from the edge map corresponding to A and latent code z randomly sampled from a standard Normal

FID score, or the Frechet Inception Distance, captures the photo-realistic quality of generated images by measuring the distance between real image sets and generated image set. We embed a set of generated images into a feature space given by the output of final avgpool layer of Inception Net v3. Viewing the embedding as a continuous multivariate Gaussian, the mean and covariance matrix are estimated for both the generated data and the real data. The Frechet distance between these two Gaussian distributions is used to quantify the quality of generated samples.

$$FID(r, g) = \|\mu_1 - \mu_2\|_2^2 + T_r(\Sigma_1 + \Sigma_2 - 2(\Sigma_1\Sigma_2)^{\frac{1}{2}}) \tag{8}$$

From Equation 8, a lower FID value means the compared image sets have a closer distribution in the feature space [4].

The first two pairs of data sets provide us with a good comparison baseline for the FID score between the real image set A and the BicycleGAN generated images. Our hypothesis is that the FID score of the first pair to be the lowest, followed by the FID score of the third pair (BicycleGAN), then the second (CycleGAN) given that CycleGAN is unsupervised and thus the 'quality' of its inputs are not as high as that of BicycleGAN, which have paired input images.

From Table 2, we see the results are consistent with our hypothesis. The FID scores of the three pairs of image sets from low to high are: FID(real image sets) < FID(real A, BiycleGAN outputs) < FID(real A, CycleGAN outputs). This suggests that the set of images generated by BicycleGAN is more **photo-realistic** as compared to that of CycleGAN.

[4] Definition and formula taken from GAN assignment instruction.

Image Set	FID Score
real A & real B	63.30969
real A & CycleGAN gen.	131.35600
real A & BicycleGAN gen.	93.71946

Table 2. FID scores across three pairs of image sets.

LPIPS Score We will quantify the diversity of BicycleGAN using the Learned Perceptual Image Patch Similarity (LPIPS) metric, which evaluates the distance between image patches. A higher LPIPS score means a larger difference, whereas a smaller LPIPS score means a stronger similarity across a set of images (less diverse) [12]. We calculate the averaged pairwise distance across three sets of data:

1. the entire validation set of size 200
2. images generated using **CycleGAN** from the edge maps of the validation set
3. images generated using **BicycleGAN** from the edge maps of the validation set and latent code z randomly sampled from a standard Normal

The first two sets of data provide us with a good comparison baseline for the averaged pairwise LPIPS Score across the set of images. We expect the real image set to be the most diverse, followed by the one generated by BicycleGAN, and lastly the one by CycleGAN. Thus, our hypothesis is that the validation set has the highest averaged pairwise LPIPS Score, then BicycleGAN, and lastly CycleGAN.

From Table 3, we see the results are consistent with our hypothesis. The LPIPS scores of the three pairs of image sets from low to high are: CycleGAN generated images, BicycleGAN generated images, the validation set. This suggests that the set of images generated by BicycleGAN is more **diverse** as compared to that of CycleGAN.

Image Set	LPIPS Score
Validation Set	0.427595
CycleGAN gen.	0.284115
BicycleGAN gen.	0.341292

Table 3. LPIPS scores across three pairs of image sets.

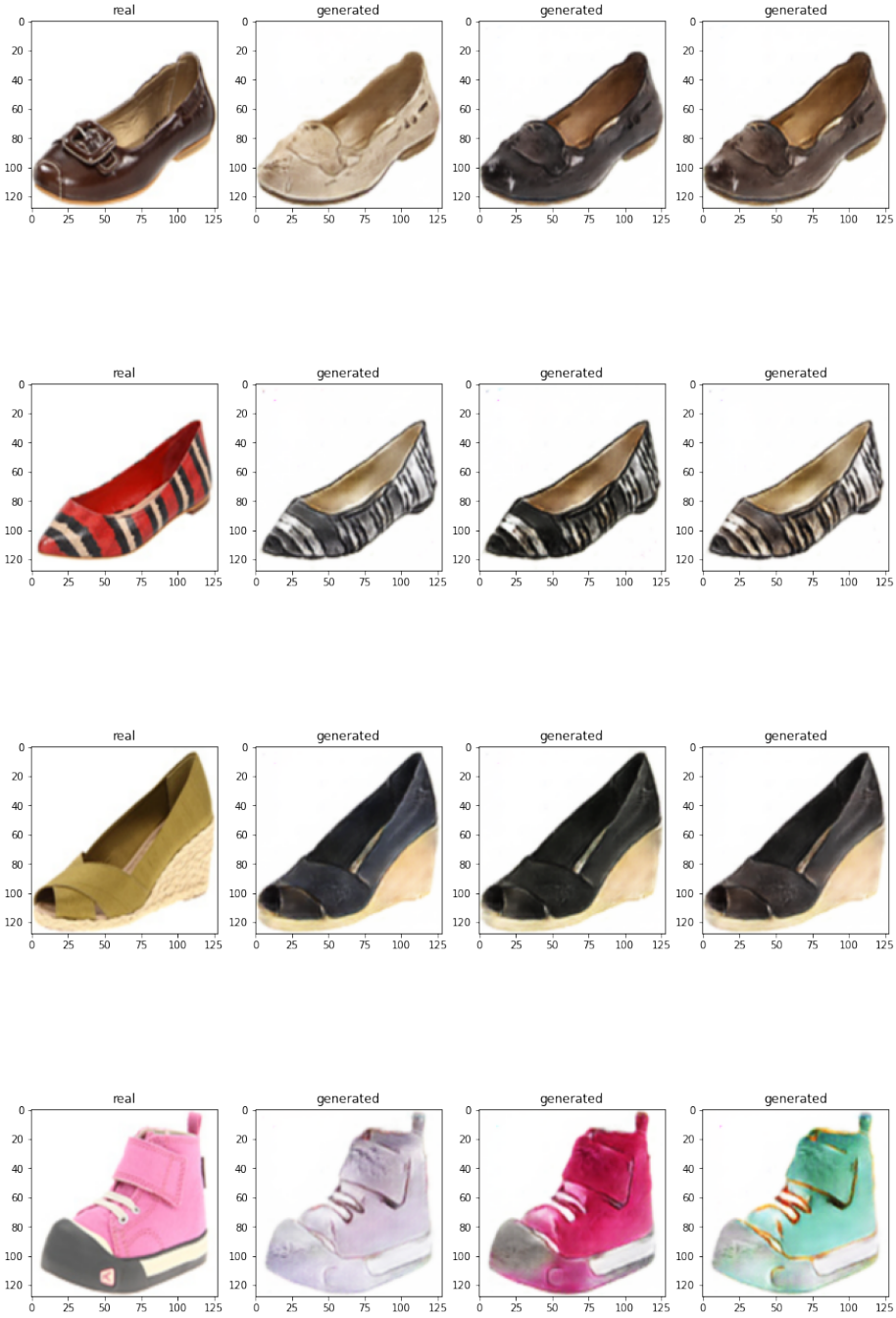
4.3 Qualitative Evaluation

We first use the CycleGAN model from the previous GAN assignment to generate Figure 4.3 as a baseline visual comparison. Then, we randomly selected 10 images from the validation set, for each of which we display below three generated images with randomly sampled latent code. The first image of each row is the real image and the following three are generated.

As compared to the results from CycleGAN, we notice a clear diversity and photo-realisticity among the images generated by BicyleGAN, which is consistent to our qualitative evaluation results above. We notice that the BicyleGAN generated results are mostly very photo-realistic, many of which can even fool some people we ask to identify which shoes are real. It's surprising that the model is able to capture even the reflection on leather shoes as shown in one example below. While we see a decent diversity among the generated results, we do notice that a large number of the generated shoes have a brownish-gray color, which is likely the color of the majority of the training set shoes.

Fig. 29. CycleGAN generated Images & the corresponding real images


real
generated
generated
generated
real
generated
generated
generated
real
generated
generated
generated
real
generated
generated
generated
real
generated
generated
generated


real
generated
generated
generated
real
generated
generated
generated
real
generated
generated
generated
real
generated
generated
generated

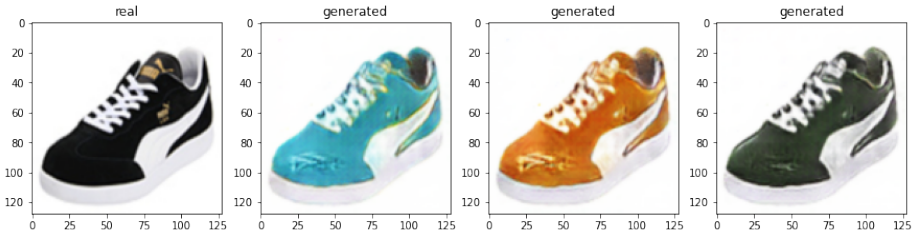


Fig. 30. BicycleGAN generated Images & the corresponding real images

5 Discussion / Conclusion

The experience as well as evaluation above verifies our hypothesis that BicycleGAN generated images are not only more photo-realistic but more diverse as compared to those generated by CycleGAN.

The improvement in photo-realisticity of BicycleGAN as compared to CycleGAN could be potentially attributed to the better 'quality' input images that the prior model has: BicycleGAN requires paired images of two domains, while CycleGAN requires unpaired images, which can be easier to obtain but increase the learning difficulty as well. Furthermore, the difference in training epochs between the two models we use for evaluation could also attribute to the performance difference here: while the BicycleGAN model has been trained for 50 epochs, the CycleGAN model is trained only for 2 epochs. As suggested by the author of the BicycleGAN paper in https://github.com/junyanz/BicycleGAN/issues/12, the performance in terms of photo-realisticity between BicycleGAN and CycleGAN should be similar.

On the other hand, the improvement in diversity of BicycleGAN as compared to CycleGAN is significant. By learning a bijection between the latent and output space, BicycleGAN generator is able to effective utilize the latent code and generate a set of diverse outputs, validated both qualitatively and quantitatively above.

Future work on this project can be split to two parts. First, as noted above, there exists a bias in model performance evaluation resulted from an imbalanced training between the baseline and target model. Thus, it's worthwhile to train the baseline CycleGAN model for 50 epochs and compared its performance with BicycleGAN for a better evaluation. Second, it is worthwhile to explore various tricks to stabilize GAN training or improve image quality, such as instance normalization [11], spectral normalized GAN [9], etc. Furthermore, we could also investigate MSGAN [7] to enforce diversity.

References

1. Choi, Y., Choi, M., Kim, M., Ha, J.W., Kim, S., Choo, J.: Stargan: Unified generative adversarial networks for multi-domain image-to-image translation (2018)
2. Efros, A.A., Freeman, W.T.: Image quilting for texture synthesis and transfer. In: Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques. p. 341–346. SIGGRAPH '01, Association for Computing Machinery, New York, NY, USA (2001). https://doi.org/10.1145/383259.383296, https://doi.org/10.1145/383259.383296
3. Emami, H., Moradi Aliabadi, M., Dong, M., Chinnam, R.B.: Spa-gan: Spatial attention gan for image-to-image translation. IEEE Transactions on Multimedia **PP**, 1–1 (02 2020). https://doi.org/10.1109/TMM.2020.2975961
4. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. CoRR **abs/1512.03385** (2015), http://arxiv.org/abs/1512.03385
5. Isola, P., Zhu, J.Y., Zhou, T., Efros, A.A.: Image-to-image translation with conditional adversarial networks. CVPR (2017)
6. Liu, M.Y., Breuel, T., Kautz, J.: Unsupervised image-to-image translation networks. In: Proceedings of the 31st International Conference on Neural Information Processing Systems. p. 700–708. NIPS'17, Curran Associates Inc., Red Hook, NY, USA (2017)
7. Mao, Q., Lee, H., Tseng, H., Ma, S., Yang, M.: Mode seeking generative adversarial networks for diverse image synthesis. In: Proceedings - 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2019. pp. 1429–1437. Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, IEEE Computer Society, United States (Jun 2019). https://doi.org/10.1109/CVPR.2019.00152
8. Mao, X., Li, Q., Xie, H., Lau, R.Y.K., Wang, Z.: Multi-class generative adversarial networks with the L2 loss function. CoRR **abs/1611.04076** (2016), http://arxiv.org/abs/1611.04076
9. Miyato, T., Kataoka, T., Koyama, M., Yoshida, Y.: Spectral normalization for generative adversarial networks. In: International Conference on Learning Representations (2018), https://openreview.net/forum?id=B1QRgziT-
10. Ronneberger, O., Fischer, P., Brox, T.: U-net: Convolutional networks for biomedical image segmentation. CoRR **abs/1505.04597** (2015), http://arxiv.org/abs/1505.04597
11. Ulyanov, D., Vedaldi, A., Lempitsky, S.V.: Instance normalization: The missing ingredient for fast stylization. arXiv: Computer Vision and Pattern Recognition (2016)
12. Zhang, R., Isola, P., Efros, A.A., Shechtman, E., Wang, O.: The unreasonable effectiveness of deep features as a perceptual metric. In: CVPR (2018)
13. Zhu, J.Y., Park, T., Isola, P., Efros, A.A.: Unpaired image-to-image translation using cycle-consistent adversarial networkss. In: Computer Vision (ICCV), 2017 IEEE International Conference on (2017)
14. Zhu, J.Y., Zhang, R., Pathak, D., Darrell, T., Efros, A.A., Wang, O., Shechtman, E.: Toward multimodal image-to-image translation. In: Advances in Neural Information Processing Systems (2017)