CycleGAN and Representation Disentanglement

Yanyu Chen
ychengx@connect.ust.hk

Mingkai Tang
mtangag@connect.ust.hk

Chiu Wai Yan (AG)
cwyan@connect.ust.hk

Yanxin Valance Wang
ywanglx@connect.ust.hk

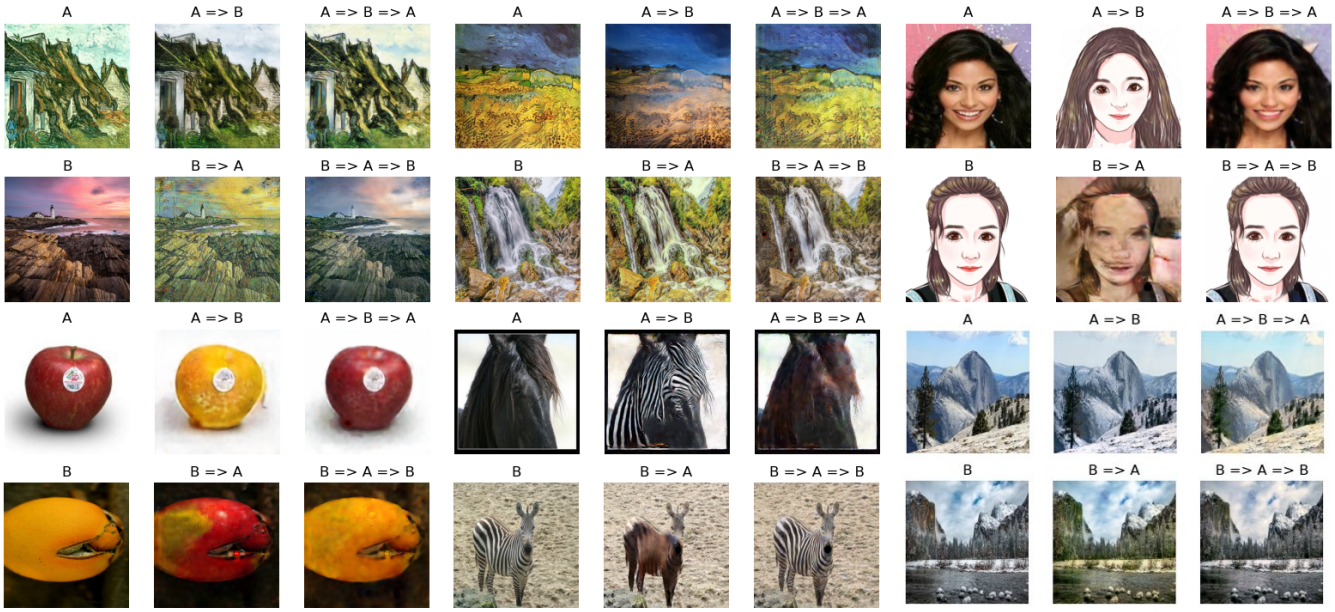

Figure 1. A subset of the output generated by our implemented CycleGAN model.

Abstract

In this project, we implemented a modified version of CycleGAN. Following the original paper, we adopted the adversarial loss, cycle-consistency loss as well as an identity loss in training. Other than that, we also investigated the capability of the CycleGAN model for representation disentanglement. Specifically, we attempt to imitate the unsupervised latent representation enforcement as introduced in InfoGAN, by defining an extra classifier to predict the latent code in the generator. The trained CycleGAN model succeeds to translate images between two domains in a realistic way, with occasional flaws. Nevertheless, for representation learning, the model failed to come up with a semantically disentangled latent codes. We regard such failure to the drawback induced by the image-to-image framework, cycle-consistent loss, and also the insufficiency of the input dataset.

Keywords: GANs, neural networks, image-to-image transfer, image generation

1 Introduction

Nowadays we have cameras, why do we still need artist to do paintings? People faced exactly the same problem in 19th century [8]. As cameras becoming more and more portable, what was the utility of artists? In response to the challenge, impressionism was invented - artists no longer captured the exact representation, in contrast, they focused on rendering a perception of the scene. That is why paintings by Monet and Van Gogh etc is important.

In 21st century, we can not only have paintings of impressionism, but create new impressionistic paintings using artificial intelligence. Thanks to technology of neural style transfer [9], we can now convert photos to synthetic paintings which are in the style of different artists.

The contribution of our work is three-fold: first, we introduce a neural network for style transfer - CycleGAN; second, we show that this model can entertain image-to-image translation with dataset that lacks a paired, one-to-one relationship; third, we explore whether CycleGAN system can demonstrate representation disentanglement.

In Section 2 we review related streams of work. In Section 3 we derive formulation of CycleGAN. In Section 4 we show

some experiment result. In Section 5 we summarize points for discussion and conclusion. In Section 6 we summarize individual contribution to the term project.

2 Related Works

2.1 Generative Adversarial Network (GAN)

Generative Adversarial Networks (GANs) have achieved a great success in image generation tasks. Composing of two sub-networks, one generator and one discriminator, GANs are able to generate realistic image that is indistinguishable by the discriminator, and even human. An abundant amount of GAN variations, in terms of model architecture, loss function and training method have been proposed recently. For example, conditional GAN (cGAN) [7] makes use of a random vector as input to simulate non-deterministic behavior. The CycleGAN [9] model that we attempt to reproduce is another typical adaptation of the GAN framework.

2.2 Image-to-image translation

In an image-to-image translation task, we would like to let the model learn a mapping function to transfer an image in one domain to another. Typically, variation auto-encoders (VAE) [3] can be employed to perform this task. CoGAN [6] learns the representation between two domains with two weight-sharing GANs. [5] further extends CoGAN to learn a common latent space between domains. Another recent work that achieves great success is pix2pix [2], which makes use of a cGAN architecture on image domain mapping. All of the aforementioned models perform a one-way translation on the image, which is usually irreversible. However, our implemented CycleGAN is able to revert the translation and convert the image back to the original domain.

2.3 Representation Disentanglement

With the ability to generate images, one type of research work focuses on investigating whether the generative models are capable of breaking down the image into latent attributes that is similar to human understanding. More and more works focus on such behavior during image generation. DC-IGN [4] proposed a training procedure to encourage latent space neurons to represent a specific transformation, in a weakly-supervised setting. InfoGAN [1] defined loss terms for an effective unsupervised learning of categorical and continuous attributes. Although the ability to disentangle representation is hardly covered in the original CycleGAN paper, in this work, we applied a simplistic adaptation of InfoGAN on the cycle-consistency framework to evaluate the performance of segregating latent code in semantics that is understandable by human.

3 Implementation

3.1 Problem formulation and loss function

For training, we prepared two unpaired dataset A and B. Our goal is to train the network to translate pictures in domain A to B. In addition, the network should learn to approximate a translation function, mapping domain A to B. For CycleGAN, the datasets don't have to label features as we target to let the network learn it automatically.

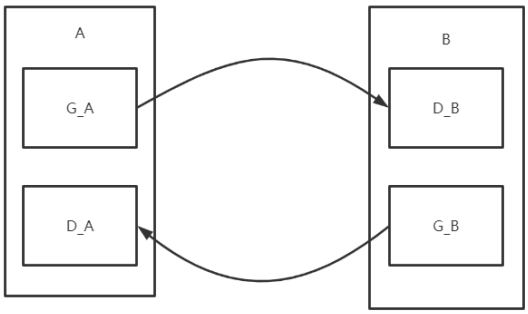

Figure 2. Network structure

As shown in Fig.2, in our network, there are two generators named G_A and G_B and two discriminators named D_A and D_B. G_A is to generate dataset A to dataset B. G_B is to generate dataset B to dataset A. D_A is to estimate the similarity between generated image and image from dataset A. D_B is to estimate the similarity between generated image and image from dataset B.

There are three kinds of loss in the loss function. The first is the adversarial loss which is common used in most GAN. It is written as

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

It represent how different between the generated image and the real data from dataset Y

The second is cycle consistency loss which is only used in CycleGAN. We have an image from dataset A, we want to the image remain the same after it pass through G_A and then pass through G_B because when image from dataset A pass G_A, it will produce a image with the style similar with B, and if we put the result into G_B, it can translate into the style similar with A. In the ideal cases, it should be same with the original image. So do image from dataset B. Using this loss, the generated image should remain the main feature of the original image, otherwise it's impossible to restore to the original image. Cycle consistency loss is defined as:

$$L_{cyc}(G_A, G_B, A, B) = \mathbb{E}_{a \sim p_{data}(A)}\left[||G_B(G_A(a)) - a||_1\right] + \mathbb{E}_{b \sim p_{data}(B)}\left[||G_A(G_B(b)) - b||_1\right] \quad (2)$$

The third loss is called identity loss. Adding in implementation can increase stability. It is to let image from A pass through G_B, it should remain the same, because G_B is to translate image to style A and the input image is already style A, it should not change to other style different from A. The formula of identity loss is:

$$\begin{aligned} L_{id}(G_A, G_B, A, B) =& \lambda_{idA}\mathbb{E}_{a\sim p_{data}(A)}\left[||G_A(a) - a||_1\right] \\ &+ \lambda_{idB}\mathbb{E}_{b\sim p_{data}(B)}\left[||G_B(b) - b||_1\right] \end{aligned} \quad (3)$$

The objective function is:

$$\begin{aligned} L(G_A, G_B, D_A, D_B, A, B) =& L_{GAN}(G_A, D_B, A, B) \\ &+ L_{GAN}(G_B, D_A, B, A) \\ &+ \lambda_{cyc} * L_{cyc}(G_A, G_B, A, B) \\ &+ \lambda_{id} * L_{id}(G_A, G_B, A, B) \end{aligned} \quad (4)$$

We need to solve:

$$G_A^*, G_B^*, D_A^*, D_B^* = \arg \min_{G_A, G_B} \max_{D_A, D_B} L(G_A, G_B, D_A, D_B, A, B) \quad (5)$$

In each epoch, we train D_A and D_B first. After that we fix the parameters of D_A and D_B and train G_A and G_B.

3.2 Generator

We use UNet as the generator. The stucture is shown in Fig.3 UNet is an architecture that is similar to ResNet. In UNet, the "residual" connection is performed symmetrically, for example, the first 2^{nd} layer connects with the last 2^{nd} layer. The "residual" connection enables a gradient flow that is less likely to be influenced by the vanishing gradient problem. On top of the original UNet, we further adopt some fine adjustments. On one hand, we change the size of layers to fit the dataset. On the other hand, we change some layers' type. For example, we change "ConvTranspose2d" to "Upsample + Conv2d" to alleviate the checkerboard effect caused by the deconvolution layer.

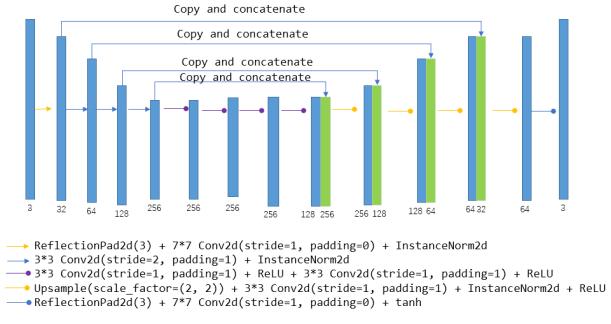

Figure 3. Generator structure

3.3 Discriminator

For the discriminator, it's the same with the CycleGAN paper. It is shown in Fig.4. There are 4 stacks of "conv2d + InstanceNorm2d + LeakyRelu(0.2)" in the discriminator,

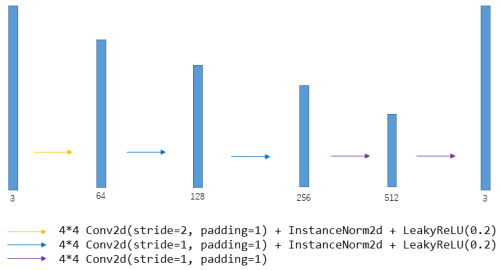

Figure 4. Discriminator structure

3.4 Additional Adaptation to Representation Learning

In this subsection, we will discuss how to adjust the network to let it learn about the represtation. We follow the idea of InfoGAN to perform unsupervised representation disentanglement. However, the greatest challenge is that InfoGAN generates image from noise vector, CycleGAN generates from another image. We should combine their technique by some tricky method. For the discriminators, we add a component called Q, to predict a continuous n-vector. Q is same as discriminator (and share weights) except the last layer. The last layer of Q is "4*4 conv2d(stride=2, padding=1) + Linear". For the generators, in the middle of UNet, after 8^th layer, we add a linear projection that maps the feature space into attribute space, so that this attribute vector acts as the ground-truth label of prediction of Q. By this method, the network can use continuous latent vector to do the representation disentanglement. However, in our work, discrete attribute is not yet implemented.

4 Experiments

In this study, CycleGAN is applied into various database including monet2photo, horse2zebra, and also photo2cartoon, a custom data set prepared by us. Unless otherwise specified, we trained all models for 200 epochs. We applied Adam optimizer, with a fixed learning rate of 2×10^{-4} for the first 100 epochs while we decay the learning rate linearly down to 0 in the remaining epochs. For the weighting factors, we set $\lambda_{cyc} = 10$, $\lambda_{id} = 0.5$ and $\lambda_{attr} = 0.1$.

4.1 Qualitative Result

We feed training dataset A into generator A, and obtain the translated fake image B. After that, the fake image is

Figure 5. Result of Monet2photo datasets

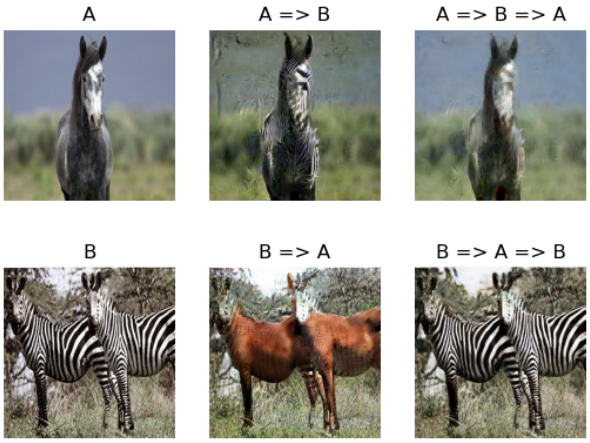

Figure 6. Result of Horse2zebra datasets

converted B back to the original domain A, by generator B. This helps verify whether the CycleGAN successfully learned the translation in both of the ways. For monet2photo dataset, the model is expected to switch the style of the picture but not too much content. The style in monet2photo, as observed by human, should consist of multiple information of the image, including saturation, acutance, color histogram and edge. The result of monet2photo database is shown in the Fig 5. The figure is formed by 6 subplots. The first row describes the cycle conversion generated by generator A and reconstruction by generator B. The second row describes in the other way round. The resolution of plots in Monet2photo are 128×128. From the process from A to B, though a sharp picture can be produced, it still does not looks like a photo captured by real camera or by human eyes. However, the saturation, color histogram, etc, have undergone an obvious change. Thus, it indicates that our implemented CycleGAN is functional and could learn some of the features in both domain A and B.

In another dataset, horse2zebra, it is observed that the model could extract and replace the features of horse and zebra, without changing the picture style in a large extent. The result of horse2zebra is shown in the Fig 6. The first row illustrates the program is trying to convert a front view horse with black and white skin pattern to a picture of zebra. Although the "A to B" is a little bit vague, the zebra pattern and the shape is not mussy. However, the "B to A" mess up the shape of horse head and the left hand side one contains the zebra's head and horse body. But at "B to A to B", the plot looks almost no change with the initial one. Thus, we conclude that the program has the ability to extract the major features of horse and zebra, although there are some minor noise and error in the converted picture.

Fig 7 shows one of the output generated from the photo2cartoon dataset. On the whole, the translations are not particularly successful. As we can see, the machine does extract some

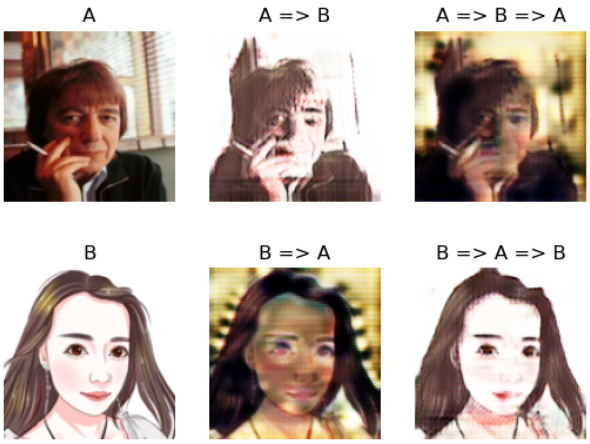

Figure 7. Result of photo2cartoon dataset

features from the cartoon pictures, such as the overall whiteness. Also, it extracts certain features from camera caught photo, such as the color of the person's skin. But we can see at a glance that these images are particularly unnatural, no-mater it's "A to B" or "B to A". We will further discuss and explain the result of such translation in later section regarding representation disentanglement.

4.2 Quantitative Analysis and Representation Disentanglement

Table 1 and Table 2 is a recap of the AMT and FCN-score in the original CycleGAN paper. We can see that the performance of CycleGAN is much better than most of remaining architecture. The only exception is that CycleGAN performs relatively poor comparing to pix2pix, the basis of CycleGAN, since the cycle-consistency makes the model unable to modify too much. For example, in photo2cartoon, the model cannot modify the shape, as it is likely to be irreversible,

Loss	Map to Photo	Photo to Map
CoGan	0.6% ± 0.5%	0.9% ± 0.5%
BiGan/ALI	2.1% ± 1.0%	1.9% ± 0.9%
Pixel+Gan	0.7% ± 0.5%	2.6% ± 1.1%
Featureloss+Gan	1.2% ± 0.6%	0.3% ± 0.2%
CycleGAN	**26.8% ± 2.8%**	**23.2% ± 3.4%**

Table 1. AMT "real vs fake" test on maps ↔ aerial photos.

Loss	Per-pixel acc.	Per-class acc.	Class IOU
CoGan	0.40	0.10	0.06
BiGan /ALI	0.19	0.06	0.02
Pixel+Gan	0.20	0.10	0.04
Featureloss+Gan	0.06	0.04	0.01
CycleGAN	**0.52**	**0.17**	**0.11**
pix2pix	0.71	0.25	0.18

Table 2. FCN-scores for different methods, evaluated on Cityscapes labels ↔ photos.

which makes an real face photo hardly convert to a cartoon version, in shape and texture (and vice versa).

Fig 8 presents the loss for horse2zebra during the training process. From the figure, it can be discovered that loss G and D from generator and discriminator are approaching each other as model converges.

For representation disentanglement, this study follows the implementation in InfoGAN [1], but the CycleGAN architecture could not segregate the information well as InfoGAN does. As the result showing in Fig 9, no matter for photo to cartoon or backward, the variation of the latent code c does not alter one specific feature clearly. One possible reason is that InfoGAN generates image from noise, but CycleGAN generates image from image. The image information collected by the encoder gives prior information, which is more difficult to edit. Furthermore, InfoGAN has the freedom to modify the latent vector since they are initially random, but CycleGAN's latent code largely come from the input image. Another cause deteriorating the performance is the lack of clarity in the datasets. In the samples of photo2cartoon, the skin color and background is having a very similar color, and there is almost no texture at the middle, which causes convolution filters that detect change in color to fail.

5 Conclusion and Discussion

In this project, we implemented a modified version of CycleGAN, a GAN model that utilizes the cycle-consistency loss to learn to interchangeably convert images from one domain to another. Under a well trained environment, CycleGAN is able to generate realistic images, while some of the best-case result could even fool human. However, flaws of the generated image can usually be observed, probably due to the small and polluted training dataset. We also noticed a

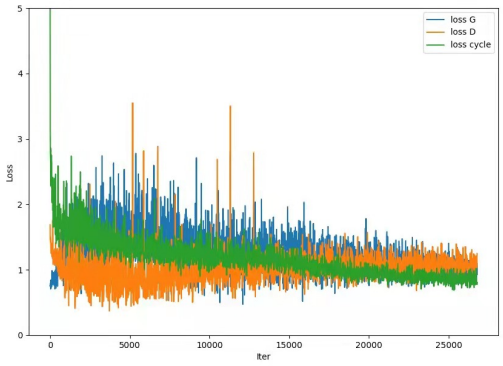

Figure 8. A plot on the losses of the model trained on horse2zebra

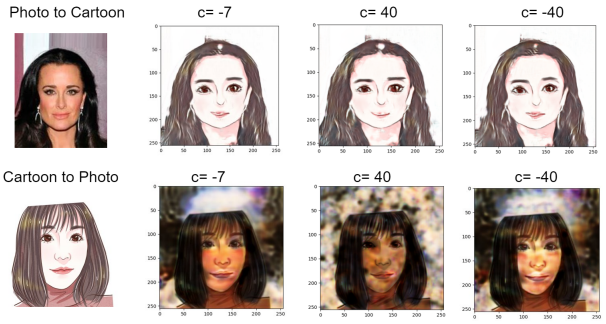

Figure 9. A comparison of output with different first latent value c. The default latent code determined by the generator is $c = -7$.

general decay in performance in comparison to other GAN infrastructure such as pix2pix. We attribute such drawback to the cycle-consistency loss that discourages the output to deviate too much from the original image, refraining the model to apply modifications that are likely to be irreversible to the images, such as the overall shapes, texture details, etc.

We also augmented the reproduced CycleGAN architecture with latent value prediction, one of the key concept in InfoGAN, so as to learn representation of the image attributes that are meaningful to human. Although we are able to transplant similar ideas into our implementation, the ability to disentangle latent information is still limited. It may be because of the complexity of the CycleGAN dataset, together with the distinct nature of input between CycleGAN and InfoGAN models. We leave the exploration on possible improvement to such enforcement of disentangled latent code to future works.

6 Individual Contribution

Task	Sub-task	Personel
Literature review	Literature review	Valance
Coding up the model	Coding up the model	Mingkai, AG
	Check the model	Valance, AG
Find dataset	Find dataset	Valance
Train CycleGAN	Train CycleGAN	Valance, AG
Making plots	Making plots	Yanyu, AG

Table 3. Individual task assignment

Report and Presentation	Personel
Introduction	Valance
Related works	AG
Design and implementation	Mingkai
Experiment	Yanyu
Discussion and conclusion	AG

Table 4. Report and presentation task assignment

References

[1] Xi Chen, Yan Duan, Rein Houthooft, John Schulman, Ilya Sutskever, and Pieter Abbeel. 2016. InfoGAN: Interpretable Representation Learning by Information Maximizing Generative Adversarial Nets. arXiv:1606.03657 [cs.LG]

[2] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. 2017. Image-to-Image Translation with Conditional Adversarial Networks. *CVPR* (2017).

[3] Diederik P Kingma and Max Welling. 2013. Auto-Encoding Variational Bayes. In *ICLR*.

[4] Tejas D Kulkarni, William F. Whitney, Pushmeet Kohli, and Josh Tenenbaum. 2015. Deep Convolutional Inverse Graphics Network. In *Advances in Neural Information Processing Systems*, C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett (Eds.), Vol. 28. Curran Associates, Inc. https://proceedings.neurips.cc/paper/2015/file/ced556cd9f9c0c8315cfbe0744a3baf0-Paper.pdf

[5] Ming-Yu Liu, Thomas Breuel, and Jan Kautz. 2017. Unsupervised Image-to-Image Translation Networks. In *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (Eds.), Vol. 30. Curran Associates, Inc. https://proceedings.neurips.cc/paper/2017/file/dc6a6489640ca02b0d42dabeb8e46bb7-Paper.pdf

[6] Ming-Yu Liu and Oncel Tuzel. 2016. Coupled Generative Adversarial Networks. In *Advances in Neural Information Processing Systems*, D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett (Eds.), Vol. 29. Curran Associates, Inc. https://proceedings.neurips.cc/paper/2016/file/502e4a16930e414107ee22b6198c578f-Paper.pdf

[7] Mehdi Mirza and Simon Osindero. 2014. Conditional Generative Adversarial Nets. arXiv:1411.1784 [cs.LG]

[8] Wikipedia. 2021. Impressionism — Wikipedia, The Free Encyclopedia. http://en.wikipedia.org/w/index.php?title=Impressionism&oldid=1020526026. [Online; accessed 07-May-2021].

[9] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. 2017. Unpaired Image-To-Image Translation Using Cycle-Consistent Adversarial Networks. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*.