Photo-Anime Translation based on Cycle-Consistent Adversarial Networks

Yiming Hu (TSCRM) Jiarui Tan (TSCRM) Jia Fu (TMAIM)
KTH Royal Institute of Technology
Stockholm, Sweden 10044
yimingh@kth.se jiaruit@kth.se jiafu@kth.se

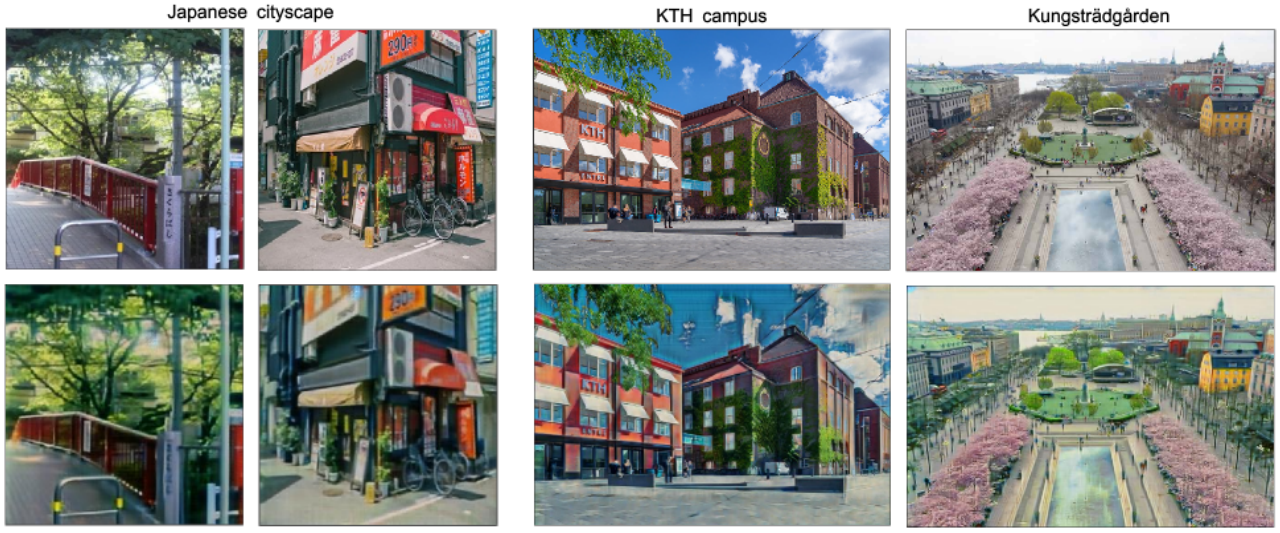


Figure 1: Our result of photo animation: Japanese landscape, Kungliga Tekniska högskolan and Kungsträdgården

Abstract

In this paper, we mainly focus on the task of photo animation, which is a popular and challenging branch of image-to-image translation problem in computer vision. To solve this task, a cycle-consistent adversarial network (CycleGAN) is implemented to transform real-world images to cartoon style counterfeit under the supervision of unpaired, small-sized training data. In order to learn geometric features and essential content from the source image, two different generator architecture are used, i.e. the U-Net architecture and ResNet architecture. Experiments have shown that we managed to stabilize the model and give quite a few exquisite photos in animation style.

1. Introduction

The twenty-first century has witness the worldwide influence of cartoon animation as a new form of artistic movies. A large amount of splendid works and inspiration was transferred from real life, like the phenomenal animation film "Your name", whose prototype was Tokyo in Japan. Besides manually drawn animation images, Generative Adversarial Networks (GANs) is capable of producing "fake" photos that look similar to cartoon images. In this paper, due to limitation caused by unpaired training data in the photo animation task, we train a cycle-consistent adversarial network [5] based on different network architectures to tune the best style transformer that generates impressive animation images from real-world photos.

The rest of the paper is organized as follows. Section 2 summarizes related research field regarding image-to-image translation and the state-of-art GANs. Section 3 introduces the dataset "Real2Anime2" used in our training session. Section 4 explains details of network architecture, the objective loss function and the optimization methods we use. Section 5 presents a few animation images produced by our model, which is further analyzed in Fourier domain and compared with results of latest work. Section 6 gives an overall summary of our work and advice for further work.

2. Related Work

I2I translation A few approaches to solve I2I tasks on the early stage were led by non-parametric texture synthesis like sampling from probability distribution [3] or Markov random field [12], where the patch mapping is found based on local similarity. A modern technique for the search of representative feature and content uses Convolution Neural Network (CNN) to learn a compact representation to preserve the original patterns [10]. The down-sampling layers in CNN aims to compress the input to extract key information, while a transposed convolution up-samples the signal by its local features but is likely to cause checkerboard artifacts [8]. In animation style transform, significant breakthrough and impressive works was made by advanced generative adversarial network [1],[2],[6].

GANs The emergence of GANs by Isola *et al.* [4] presents an alternative method to attack the I2I task, simulating the zero-sum game between two agents, i.e. the discriminator and generator. The objective cost function follows in equation 1:

$$\min_G \max_D L(D,G) = E_{x\sim p_{data}(x)}[log(D(x)] + E_{z\sim p_z(x)}[log(1-D(G(x))] \quad (1)$$

Afterwards, they motivated to add conditional information y to guide the generator[5]. This method known as conditional GAN (cGAN) has been proven effective in attack multiple tasks in the field of I2I translation.

While most deep neural network required a matched dataset for training, Zhu *et al.* [11] proposed an unpaired I2I approach using a cycle-consistent adversarial network (CycleGAN), aiming not only for learning a mapping G from domain X to Y, but also a converse mapping F from Y to X. This work contributes as the most important reference for our work.

3. Data

The "Real2Anime2" dataset used in our experiment is originated from [1], which is a popular dataset for animation style transfer. The source domain includes 6,656 real-world photos of natural landscape, cityscape and living creatures and the target domain covers three unique styles of different Japanese animation master:

- 1,792 animation images from Miyazaki Hayao's fictionalized story movie "The wind rises".
- 1,650 animation images from Makoto Shinkai's romatic fantasy film "Your name".
- 1,553 animation images from Kon Satoshi's psychological thriller anime film "Paprika".

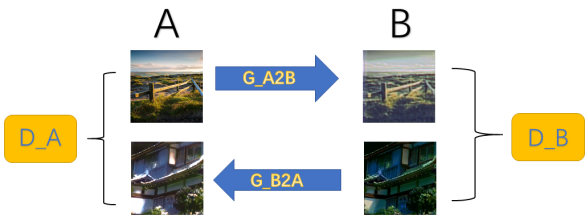


Figure 2: An illustration of the basic architecture of our CycleGAN model

Each style has several similar training images due to data augmentation, which includes "random flip", "resize", "random crop", etc.

Our "Real2Anime2" aims to demonstrate a light batch of animation style transfer dataset. The rule of tailoring is to preserve non-repetitive, unique animation images and randomly select real-world photos. The final training data includes 545 real-world photos and 372 animation image mixture of three styles.

4. Methods

4.1. Network architecture

The CycleGAN model we implemented consists of two generator networks and two discriminator networks, as shown in Figure 2.

Generator For the generator, we use a typical encoder-decoder architecture to map the real images to animation domain. It begins with three two-dimenasional downsampling convolution block with stride 2 aiming for detecting the spatial feature across the source domain. Then the output tensor goes through a nine-layer residual network (ResNet) which is responsible for complementing the missing key elements. Finally, two upsampling convolution blocks with stride 1/2 and one convolution block with filter size of 7×7 decode the information and reconstruct it as the same shape of the input image .

Discriminator In practice, we use a vanilla discriminator to distinguish real images and fake images. It extracts features by several 3×3 convolutional layers in a row and ends with a fully-connected layer compress 512 channels to one classication output.

U-Net To extend our work, we use an advanced neural network called U-Net [9] to enhance the generator to a semantic processing level, thus pushing its opponent to be also "stronger". U-Net architecture is similar to the encoder-decoder as mentioned before but has direct connections between each high resolution layers. We have expected U-Net to improve cycle consistency in our model by understanding semantic features in different scales.

4.2. Loss function

The loss function for the generator is composed of three parts, adversarial loss, cycle consistency loss and identity loss. Each loss is defined as either Manhattan distance loss (L_1) or Mean Squared Error loss (L_2):

$$L_1(\boldsymbol{y}, \hat{\boldsymbol{y}}) = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{2}$$

$$L_2(\boldsymbol{y}, \hat{\boldsymbol{y}}) = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{3}$$

Adversarial loss is computed using L_2 function which is more efficient than cross entropy loss in training GANs. [7] Here the generator and discriminator for mapping $X \rightarrow Y$ are denoted as G and D_y respectively, the adversarial loss L_{GAN} is given in equation 4. For generator F and discriminator D_x with respect to mapping $Y \rightarrow X$, the definition is similar. A smaller L_{GAN} means that the generator has a stronger ability to deceive the corresponding discriminator.

$$L_{GAN}\left(G, D_y, X\right) = L_2\left(D_y\left(G\left(X\right)\right), \mathbf{1}\right) \tag{4}$$

Cycle loss consists of two L_1 functions, as shown in equation 5. This loss function adds constraint between the input and output. Thus, the generated output images will be similar to the input ones. A small cycle loss means that the consistency is well kept.

$$\begin{aligned} L_{\text{cycle}}\left(G, F, X, Y\right) = {} & L_1(F(G(X)), X) \\ & + L_1(G(F(Y)), Y) \end{aligned} \tag{5}$$

Identity loss is defined in equation 6. This constraint helps preserve color composition between the input and output when generating photos from paintings. [11]

$$L_{\text{idt}}(G, F, X, Y) = L_1(F(X), X) + L_1(G(Y), Y) \tag{6}$$

$$\begin{aligned} \text{Loss} = {} & w_{GAN}[L_{GAN}(G, D_y, X) + L_{GAN}(F, D_x, Y)] \\ & + w_{cycle}\, L_{\text{cycle}}(G, F, X, Y) \\ & + w_{idt}\, L_{\text{idt}}(G, F, X, Y) \end{aligned} \tag{7}$$

Finally, the total loss function for generator is given in equation 7. w_{GAN}, w_{cycle} and w_{idt} are weights of different parts which are considered as hyper-parameters in our implementation. In addition, the loss function for discriminator D_x is defined in equation 8, similarly for D_y.

$$\begin{aligned} L_{D_x}(D_x, F, X, Y) = {} & \frac{1}{2} L_2(D_x(X), \mathbf{1}) \\ & + \frac{1}{2} L_2(D_x(F(Y)), \mathbf{0}) \end{aligned} \tag{8}$$

4.3. Optimization method

In our experiments, we use Adam optimizers to train all generators and discriminators. The coefficients for computing gradient average and its square are set to $(0.5, 0.999)$. Moreover, we also use a scheduler to decrease the learning rate after 100 epochs. We set the batch size to 1 and the initial learning rate to 2×10^{-4}.

5. Experiments

It turns out that our CycleGAN can transfer real-world photos into animation-style images with an unpaired training set. As shown in Figure 3, the U-net architecture is capable of producing high-contrast, "compressed" animation photos while the Residual architecture tends to generate more detailed, bright-colored images.

In Section 5.1, we discuss stabilizing training under the condition of using a small-sized dataset. In order to study the shift of geometric characteristics during the transfer, we use Fourier transform on test image "Kungsträdgården" in Section 5.2. Finally in Section 5.3, we compare our result with the state-of-art GANs for animation style transfer.

(a) real (b) UNet G (c) ResNet G

Figure 3: Image style transform generated by our CycleGAN: (a) real images from test data and Stockholm city (b) Fake images by U-net generator (c) Fake images by ResNet generator

5.1. Stabilizing training session

Training GAN on unpaired datasets is more likely to cause mode collapse, where generator happens to confuse

the discriminator by producing only one certain kind of output. Consequently, we used a pre-trained discriminator to make the training session more stable.

Another problem in our case is that the generator occasionally produces artifacts at regions of black color as Figure 4 shows. Since it is a mode collapse problem caused by fast dynamics of the generator optimizer, we have slowed down the training pace by decreasing the learning rate and increasing the cycle consistency loss and manage to solve this problem.

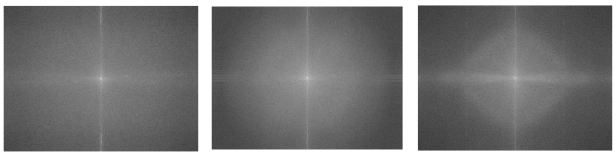
(a) test (b) artifacts (c) smooth

Figure 4: Training with different learning rate and loss coefficient: (a) Japanese cityscape, test image (b) $lr = 1e{-}3, w_{cycle} = 10$ (c) $lr = 1e{-}4, w_{cycle} = 25$

5.2. Fourier domain analysis

To further study the effect of different generators on image translation, we transformed images from spatial domain to Fourier domain using fast Fourier transform (FFT) and shift the zero-frequency component to the center. For an image of size M $\times$ N, two dimensional discrete FFT is given by equation 9.

$$F(u,v) = \frac{1}{\sqrt{MN}} \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} f(m,n) e^{-2\pi i (\frac{mu}{M} + \frac{uv}{N})} \quad (9)$$



(a) real (b) UNet (c) ResNet

Figure 5: Fourier transform of the real "Kungsträdgården" and its two counterfeits

As shown in Figure 5, U-Net compresses the geometric features of the source image from high frequency into low frequency and preserved the original direction. While ResNet discards some high frequency components and creates more features at frequency around half of the maximum. Hence, we may conclude the U-Net performs better to keep detailed features.

5.3. Comparison with latest works

We compare our result with the state-of-art cartoon style transfer GANs in figure 5.3, where we cite figures in [1] and focus on the same regions of interest as marked with red boxes.

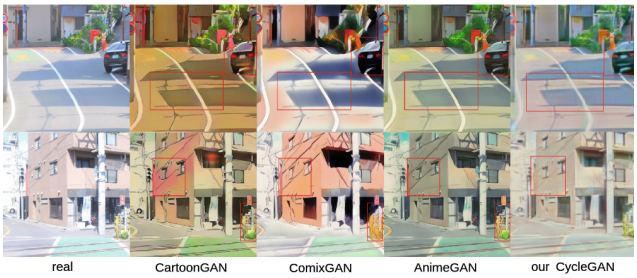


Figure 6: Comparison with latest animation style transfer network[1]

It can be seen in the figures that CartoonGAN produces colorful artifacts in some regions and loses part of the source content. While the images produced by ComixGAN always has large areas of homogeneous patterns like "thewhite shadow"in the figure on the first row and "black window" in the one on the second row. The AnimeGAN preserves most of the features from the source image. However, it produces images with turbid colors that is rarely seen in cartoon images. Our CycleGAN which uses ResNet architecture keeps the majority of the features and stylized it more towards vivid and painted style that looks like animation. As we analyzed in Section 5.2, it tends to drop the most delicate details to make it more like a drawn cartoon image. The most valuable part in our CycleGAN is that it achieves good results with a tiny dataset (10 times smaller than AnimeGAN).

6. Conclusion and Future Work

In this report, we implemented a cycle-consistent adversarial network for photo animation. The main contribution of our work is (1) implement a U-Net architecture for the generator and qualitatively analyze its functionality by Fourier transform (2) organize a small-size dataset "Real2Anime2" for fast, unpaired training and propose methods to stabilize the model and eliminate artifacts. Our experiment gives a quite impressive result on real-world landscapes as shown in Figure 1.

For the future work, we look forward to deal with the "blur" effect caused by our network which we believed can be solved by trained with a bunch of "homogeneous" training data (not a mixture of cartoon images). Furthermore, we would like to try different network architecture and loss function to find the most splendid animation style transformer.

References

[1] Jie Chen, Gang Liu, and Xin Chen. Animegan: A novel lightweight gan for photo animation. In Kangshun Li, Wei Li, Hui Wang, and Yong Liu, editors, *Artificial Intelligence Algorithms and Applications*, pages 242–256, Singapore, 2020. Springer Singapore. 2, 4

[2] Yang Chen, Yu-Kun Lai, and Yong-Jin Liu. Cartoongan: Generative adversarial networks for photo cartoonization. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9465–9474, 2018. 2

[3] A.A. Efros and T.K. Leung. Texture synthesis by non-parametric sampling. In *Proceedings of the Seventh IEEE International Conference on Computer Vision*, volume 2, pages 1033–1038 vol.2, 1999. 2

[4] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks, 2014. 2

[5] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros. Image-to-image translation with conditional adversarial networks, 2018. 1, 2

[6] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks, 2019. 2

[7] Xudong Mao, Qing Li, Haoran Xie, Raymond YK Lau, Zhen Wang, and Stephen Paul Smolley. Least squares generative adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2794–2802, 2017. 3

[8] Augustus Odena, Vincent Dumoulin, and Chris Olah. Deconvolution and checkerboard artifacts. *Distill*, 2016. 2

[9] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation, 2015. 2

[10] Ryutaro Tanno, Daniel E. Worrall, Aurobrata Ghosh, Enrico Kaden, Stamatios N. Sotiropoulos, Antonio Criminisi, and Daniel C. Alexander. Bayesian image quality transfer with cnns: Exploring uncertainty in dmri super-resolution, 2017. 2

[11] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks, 2020. 2, 3

[12] Song Chun Zhu, Yingnian Wu, and D. Mumford. Frame: filters, random fields, and minimax entropy towards a unified theory for texture modeling. In *Proceedings CVPR IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 686–693, 1996. 2