GENERATING ANIME FACES FROM HUMAN FACES WITH ADVERSARIAL NETWORKS

[1]*Yu-Jing Lin (林裕景)*, [1]*Chiou-Shann Fuh (傅楸善)*

[1]Department of Computer Science and Information Engineering,
National Taiwan University, Taipei, Taiwan

ABSTRACT

The generative adversarial network has achieved a huge success in the generative algorithm. Besides the generating handwritten digits, human faces, indoor designs, and many other images from noise, more and more researchers applied the adversarial techniques on style transfer task, which is also regarded as domain transfer task, in the past three years. In this work, we aim at generating anime-style faces from human faces in the real world. We construct a Face2Anime Dataset, performed the generative adversarial learning on it, and evaluate the result at the end.

***Index Terms*—** Anime Face Generation, Style Transfer, Generative Adversarial Network, IPPR, CVGIP 2018.

1. INTRODUCTION

In 2004, Goodfellow, I. et al. [1] introduced the generative adversarial network (GAN), which is a brilliant deep learning method generating spurious data of a given distribution. Despite its huge difficulty of generation, people realized that neural networks are able to create meaningful things. Radford, A. et al. [2] proposed an improved architecture called deep convolutional generative adversarial network (DCGAN), which generates much better images. Later, Arjovsky, M. et al. [3] stabilized the training procedure of DCGAN by utilizing several tricks in training and named the new architecture as Wasserstein GAN (WGAN). The blooming development of GAN began since these hopeful techniques were introduced.

When it comes to style transfer via deep learning, Gatys, A. et al. [4] brought a deep learning-based algorithm into the world in 2015. By minimizing the content loss and the style loss of the activation outputs of inner layers between a content image and a style image, we can transfer the style of style image onto the content image. Not only had the authors revealed the power of nonlinear multi-layer perceptrons, but this method was the first big success of style transfer in the field of deep learning. Two years later, The Luan, F. et al. [5] improved the method as the delicate algorithm known as *neural style*, which achieves a much better result of styled images. Moreover, Li, Y. et al. [6] enhanced this kind of photorealistic image style transfer to have a better inference in shorter processing time.

However, this kind of methods above requires to tune parameters in order to find the best result. In 2017, Isola, P. et al. introduced the image-to-image translation [7], which is also called *pix2pix*. By utilizing U-Net on paired data from two different domains, *pix2pix* transfers an image from one domain to the other and vice versa in a robust way. U-Net performs well on paired image transfer task.

In the real world, however, it is not practical to collect a bunch of paired data for one task. The situation is that we usually have data in domain X and in domain Y respectively. In the same year of 2017, CycleGAN [8], DiscoGAN [9], and DualGAN [10] all reveal the domain-to-domain transfer via deep learning at the same time, although the Taigman, Y. et al. [11] proposed an unsupervised adversarial domain transfer network in the previous year. The ideas of Zhu, J. et al. [8], Kim, T. et al. [9], Yi, Z. et al. [10] are basically the same and simple: generate images from images by GAN and retain the consistency. By a pair of GANs, one converts images from domain X to domain Y and the other converts those from domain Y to domain X, these methods constructed a deep learning-based style transfer system successfully. The most famous example is the zebra-to-horse. There are other examples of bidirectional style transfer depicted by the authors of CycleGAN [8], such as Monet-to-photo, summer-to-winter, apple-to-orange, etc.

For anime image generation, there are also several brilliant methods via generative adversarial networks. Jin, Y. et al. [12] proposed a conditional anime character GAN based on DRAGAN [13], which is inspired by ACGAN [14]. Liu, Y. et al. used conditional GAN to generate painted colorful images from hand-drawn sketches. Zhang, L. and Ji, Y. and Lin, X. also integrated residual U-Net with ACGAN [14] to paint gray-scale sketches. All these works show the power of generative adversarial networks on anime images.

In our Face2Anime, we are going to introduce a way to generate anime-style faces from real human faces. We take advantage of the ability of generalization from GAN for this kind of style transfer on unpaired images. We first gathered

numerous human faces from public face datasets and anime faces from the Internet. Then we applied the CycleGAN on these data to train a pair of generators. The one from real to anime is our target generator. And we will show the generated faces from faces in the datasets as well as unseen faces in the section of experiments.

2. RELATED WORKS

Face2Anime is related to the generative adversarial network, especially the CycleGAN. We are going to go through the architectures and the objective functions they try to minimize.

2.1. Generative Adversarial Network

The generative adversarial network (GAN) is comprised of a pair of networks: generator (G) and discriminator (D). As depicted in Figure 1, the generator outputs images from random noise while the discriminator tries to determine whether the input image is real or fake, which is generated by G, by giving a score to the image. The score from D is range from 0 (fake) to 1 (real). Generator and discriminator compete against each other and get improved iteratively. In the end, the generator will be able to generate images similar to the images in the training dataset and the discriminator cannot tell them from the real ones.

The following equations show the objective function of GAN. The discriminator wants to minimize both the discriminating loss $\mathcal{L}_D$ (Equation 1) and the generating loss $\mathcal{L}_G$ (Equation 2) while the generator tries to maximize $\mathcal{L}_G$.

$$\mathcal{L}_D = \mathbb{E}_{x\sim p_{data}(x)}[\log D(x)] \tag{1}$$

$$\mathcal{L}_G = \mathbb{E}_{z\sim p_z(z)}[\log(1 - D(G(z)))] \tag{2}$$

The total objective function is the sum of discriminating loss and generating loss:

$$\min_G \max_D V(D,G) = \mathcal{L}_D + \mathcal{L}_G \tag{3}$$

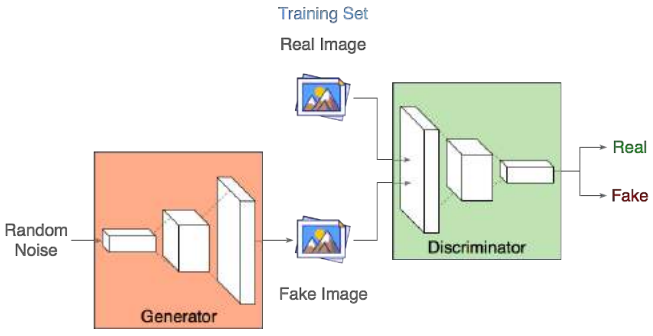

Fig. 1: Generative adversarial network.

In the training of GAN, D iteratively takes real images from the dataset and fake images from G, and then we update the network according to the above equation via backpropagating the gradients through trainable parameters in the whole network.

2.2. CycleGAN

In CycleGAN, there are a pair of GAN, G_{XY}, G_Y and G_{YX}, G_X, where X and Y are two different domains. CycleGAN works in the way Figure 2 shows. The G_{XY} generates fake images in domain Y from those in domain X and D_Y evaluates the images in domain Y. For G_{YX} and D_X, they work in an inverse way. CycleGAN takes not only one objective function into consideration in this respect.

2.2.1. Adversarial Loss

Firstly, since CycleGAN is a kind of generative adversarial networks, we have the typical GAN loss called adversarial loss:

$$\begin{aligned}&\mathcal{L}_{GAN}(G_{XY}, D_Y, X, Y)\\ &= \mathbb{E}_{y\sim p_{data}(y)}[\log D_Y(y)]\\ &+ \mathbb{E}_{x\sim p_{data}(x)}[\log(1 - D_Y(G_{XY}(x)))]\end{aligned} \tag{4}$$

$\mathcal{L}_{GAN}$ is actually the same as the summation of Equation 1 and Equation 2 described in Section 2.1.

2.2.2. Cycle Consistency Loss

The critical part of unpaired domain transfer is to use a pair of GAN. For a generated fake image in domain Y, the $G_Y X$ is supposed to have the ability of converting it back. To make sure G_{XY} and G_{YX} converts an image to domain Y and back to domain X. The cycle consistency loss is introduced as Figure 3 shows and as the following equation:

$$\begin{aligned}&\mathcal{L}_{cyc}(G_{XY}, G_{YX})\\ &= \mathbb{E}_{x\sim p_{data}(x)}[||G_{YX}(G_{XY}(x))||_1]\\ &+ \mathbb{E}_{y\sim p_{data}(y)}[||G_{XY}(G_{YX}(y))||_1]\end{aligned} \tag{5}$$

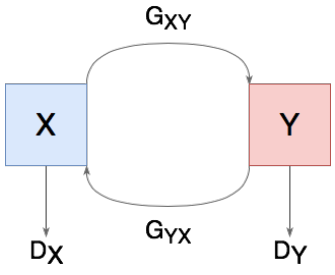

Fig. 2: CycleGAN.

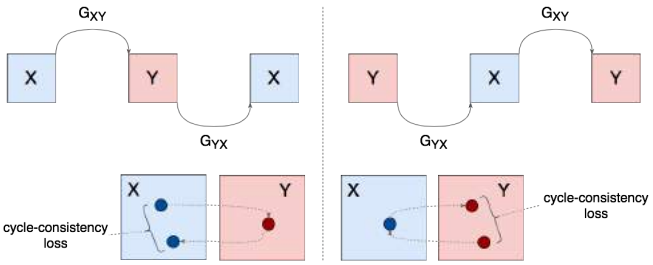

Fig. 3: Cycle-consistency loss.

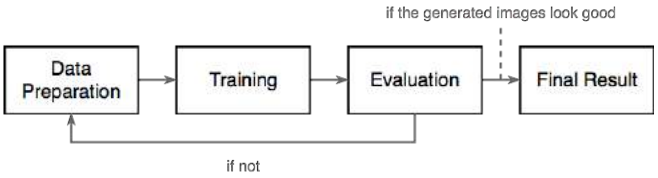

Fig. 4: High level procedure of Anime2Face.

2.2.3. *Full Objective Function*

The full objective function of the whole network is, therefore, a summation of these two kinds of loss functions. The parameter λ controls the influence of cycle consistency in training.

In face, λ is the critical parameter in CycleGAN training. A network with too small λ is hard to generate samples consistent with the given data; a network with too large λ, however, will have difficulty to impose changes to the data.

$$\begin{aligned}&\mathcal{L}(G_{XY}, G_{YX}, D_X, D_Y)\\&= \mathcal{L}_{GAN}(G_{XY}, D_Y, X, Y)\\&+ \mathcal{L}_{GAN}(G_{YX}, D_X, Y, X)\\&+ \lambda\mathcal{L}_{cyc}(G_{XY}, G_{YX})\end{aligned} \tag{6}$$

D_X and D_Y attempt to maximize the total loss while G_{XY} and GYX's goals are minimizing it. The parameters of the whole network are then updated according to the following objective function:

$$G^*_{XY}, G^*_{YX} = \arg\min_{\substack{G_{XY},\\ G_{YX}}} \max_{\substack{D_X,\\ D_Y}} \mathcal{L}(G_{XY}, G_{YX}, D_X, D_Y) \tag{7}$$

After numerous iterations, G^*_{XY} and G^*_{YX} are the final powerful domain transfer generators.

3. FACE2ANIME

We applied the well-designed CycleGAN on generating anime faces from real faces. Although the CycleGAN works for transferring images in both ways, it is still difficult to generate real faces from anime ones. Therefore, we focus on only the one from real to anime.

3.1. Face2Anime Dataset

The data are the most important to this task. Without proper data, there are no ways to learn a set of parameters for our CycleGAN. As a result, we first constructed our Face2Anime Dataset [15], including **anime face dataset**, **cropped CelebA dataset**, and **SCUT-FBP5500 dataset**. Then we trained the CycleGAN as Figure 5 shows.

The anime faces took us a lot of work to prepare because there are few suitable anime face dataset. We turned to collect a bunch of images from anime image sites, such as *Danbooru*, by Fährmann, M.'s tool *gallery-dl* [16]. Then we detect the anime faces in each image and crop them in a proper size to form an anime face dataset. The last step was to clean the dataset because there were some misdetected faces, which are just a part of a face or even not a face. In the end, we built a dataset [15] with 5,025 anime faces of size 64x64.

For human faces, we utilize the existing public face dataset: CelebFaces Attributes Dataset [17] and SCUT-FBP5500 Dataset [18]. CelebFaces Attributes Dataset, or CelebA, is a large-scale dataset of face images with annotated attributes. The size of images in CelebA is 178x218 so we **cropped** only the faces and resized them as 64x64. Moreover, we only took 40,000 of 202,599 images as our human face training data. Apart from CelebA, we also use SCUT-FBP5500 Dataset (FBP5500), which is a dataset for facial beauty prediction, collected by South China University of Technology. FBP5500 is in 64x64 and the faces are located in a proper location (center) in the 5,000 images. In our experiments, we are going to not only evaluate the performance of CycleGAN on style transfer but the difference between two human face datasets.

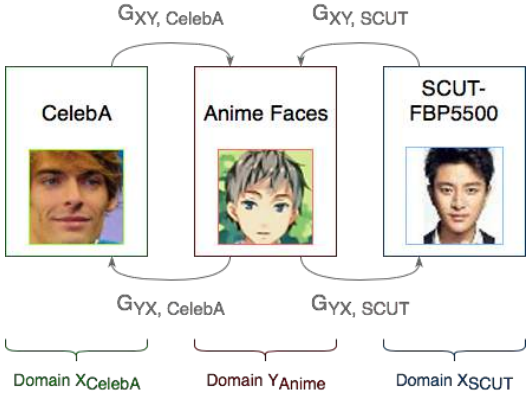

Fig. 5: Face2Anime dataset.

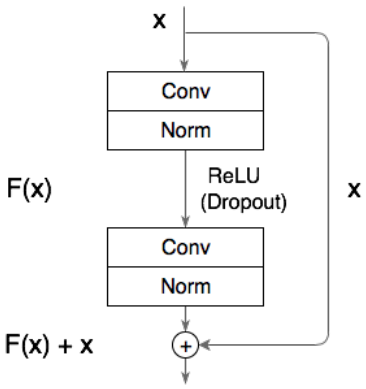

Fig. 6: Residual block.

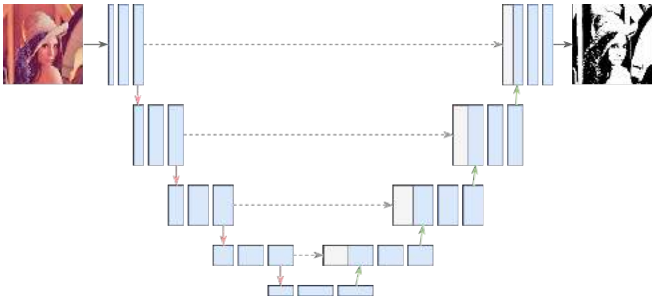

Fig. 7: U-Net.

3.2. Model Architecture

For tasks such as image generating or style transfer via generative adversarial networks, the architecture of generator and discriminator also matters. A good model comprehends the data and thus is able to learn to generate plausible data. There are also various parameters we can set to train a CycleGAN: learning rate, cycle consistency factor λ, the hidden size of each layer, etc. We in this work, however, not going to discuss too many of them but only the difference between data sources and between model architectures, which are the most important two factors in deep learning.

3.2.1. Residual Block Generator

We first chose the residual block generator, which consists of 9 residual blocks (Figure 6), proposed by Zhu, J. et al. [8] in the original paper.

Residual block generator is basically a sequence of residual blocks. A residual block in Figure 6 is inspired from the residual network proposed by He, K. et al. [19]. It takes a nonlinear transformation F of input x and then sums the output $F(x)$ and original x, known as *short-cut path*, together. The residual architecture keeps the information of inputs and prevents hidden units from dying (always output zeros). In our Face2Anime, we apply an instance normalization after each convolutional layer.

3.2.2. U-Net Generator

In the U-Net, described in Figure 7, the output features from the decoder are passed to and concatenated with the input of the encoder. Although U-Net generator results in mode collapse, which is stated by Jin, X. et al. [20] according to their experiment. There are still some drastically successful cases trained by U-Net generator in other works for style transfer, e.g. *pix2pix* [7], so we also utilized the U-Net with 256 hidden units as our alternative generative model.

3.2.3. Discriminator

We construct a basic 3-layer convolutional neural network as our discriminator. The output of the discriminator is a 4x4x1 score map which represents the realistic scores of a given image. By calculating the binary cross entropy loss function between each score with the ground truth label (1 for real image; 0 for fake image), the discriminator learns how to distinguish images.

3.3. Training Details

In order to have high-quality results, we studied several techniques to improve GAN training [21] and utilized on our model, such as the dropout layer [22] and the normalization layer. The dropout layers in generator prevent it overfit in training data so as to create various fake images. While the Kim, T. et al. [9] of DiscoGAN and Salimans, T. et al. [21] of the technique report suggest using the batch normalization [23], we adopted the other normalization method - instance normalization [24] since the Zhu, J. et al. [8] reached awesome results in CycleGAN. Besides, we randomly flipped the images while training for data augmentation.

4. EXPERIMENTS

We conducted four experiments of the following settings:

1. Residual block generator on cropped CelabA
2. Residual block generator on SCUT-FBP5500
3. U-Net generator on cropped CelebA
4. U-Net generator on SCUT-FBP5500

The following figures illustrate the result of each setting. For each set of images, the upper row contains source images and the lower row contains generated images.

Fig. 8: Faces generated by residual block generator trained on cropped CelebA. Residual block generator turns human faces into anime faces. The results (in the second row) look like artistic style paints.

Fig. 9: Faces generated by residual block generator trained on SCUP-FBP5500. Residual block generator on SCUP-FBP5500 works as well as on CelebA, implying that it is feasible to do adversarial generating data on both human face datasets.

Fig. 10: Faces generated by U-Net generator trained on cropped CelebA. U-Net generator also generates artistic style images. Moreover, we can see that the shapes of output images look more similar to the original ones, showing that the U-Net imposes the constraints on shapes of the content and only change the texture of an image. To our surprising, the results look like statues, which are made up of polyhedron.

Fig. 11: More generated samples on SCUT-FBP5500. Some of them look crazy but cool. The left faces are typical images from FBP5500. The images on the right side are framed by round borders, which are probably from profile pictures. Although some of the training images are not square, the Face2Anime CycleGAN still works well.

Fig. 12: Novel faces which are unseen during training. The generators are able to generate anime-style faces from unseen human faces. This result shows the generalization of CycleGAN. However, some faces are only slightly changed in my experiment.

Fig. 13: Some samples of anime-to-human faces. Although it is quite difficult for a generator to generate real human faces from anime faces. There are still some successful samples from the generated testing images. Figure 13 demonstrates some of them. The generator from anime to real do learn some human face textures, such as smoother skin, smaller eyes, lower contrast hair color, a straight nose, etc.

5. CONCLUSION

We demonstrate a human-to-anime style transfer on faces via CycleGAN called Face2Anime. We successfully create a bunch of anime-style faces from human faces by Face2Anime. There are no much differences in training on the CelebA dataset and anime face dataset compared to on the SCUT-FBP5500 dataset despite part of the images in FBP5500 are round-framed. The images created by the residual block generator diverge from those generated by the U-Net generator. The former generated an artistic style paints while the latter only changes the textures and preserve the edges of original contents.

The quality of the generated images, however, is not ideal enough. The results shown in Section 4 are merely a little part of all generated images, and most of them actually do not look like a human face or an anime face. Also, the stability of training Face2Anime is poor since it is common that the CycleGAN generates only "abstract paintings" at the end. Therefore, we will keep improving the quality and stability of Face2Anime in the future.

On the other hand, in this work, we achieved the success on style transfer between realistic faces and anime faces. The next step is to generate real anime faces, which are not just involved in changing textures.

REFERENCES

[1] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio, "Generative adversarial nets," in *Advances in neural information processing systems*, 2014, pp. 2672–2680.

[2] Alec Radford, Luke Metz, and Soumith Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," *arXiv preprint arXiv:1511.06434*, 2015.

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

[4] Leon A Gatys, Alexander S Ecker, and Matthias Bethge, "A neural algorithm of artistic style," *arXiv preprint arXiv:1508.06576*, 2015.

[5] Fujun Luan, Sylvain Paris, Eli Shechtman, and Kavita Bala, "Deep photo style transfer," *CoRR, abs/1703.07511*, vol. 2, 2017.

[6] Yijun Li, Ming-Yu Liu, Xueting Li, Ming-Hsuan Yang, and Jan Kautz, "A closed-form solution to photorealistic image stylization," *arXiv preprint arXiv:1802.06474*, 2018.

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

[8] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros, "Unpaired image-to-image translation using

cycle-consistent adversarial networks," *arXiv preprint arXiv:1703.10593*, 2017.

[9] Taeksoo Kim, Moonsu Cha, Hyunsoo Kim, Jungkwon Lee, and Jiwon Kim, "Learning to discover cross-domain relations with generative adversarial networks," *arXiv preprint arXiv:1703.05192*, 2017.

[10] Zili Yi, Hao Zhang, Ping Tan, and Minglun Gong, "Dualgan: Unsupervised dual learning for image-to-image translation," *arXiv preprint*, 2017.

[11] Yaniv Taigman, Adam Polyak, and Lior Wolf, "Unsupervised cross-domain image generation," *arXiv preprint arXiv:1611.02200*, 2016.

[12] Yanghua Jin, Jiakai Zhang, Minjun Li, Yingtao Tian, Huachun Zhu, and Zhihao Fang, "Towards the automatic anime characters creation with generative adversarial networks," *arXiv preprint arXiv:1708.05509*, 2017.

[13] Naveen Kodali, Jacob Abernethy, James Hays, and Zsolt Kira, "On convergence and stability of gans," *arXiv preprint arXiv:1705.07215*, 2017.

[14] Augustus Odena, Christopher Olah, and Jonathon Shlens, "Conditional image synthesis with auxiliary classifier gans," *arXiv preprint arXiv:1610.09585*, 2016.

[15] Yu-Jing Lin, "Face2Anime dataset," `https://drive.google.com/open?id=1X3QUrTI6629vSOJepbE3-8M2dKAkIbjp`, 2018.

[16] Mike Fährmann, "gallery-dl: Command-line program to download image galleries and collections from pixiv, exhentai, danbooru and more," `https://github.com/mikf/gallery-dl`, 2014.

[17] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang, "Deep learning face attributes in the wild," in *Proceedings of International Conference on Computer Vision (ICCV)*, 2015.

[18] Lingyu Liang, Luojun Lin, Lianwen Jin, Duorui Xie, and Mengru Li, "Scut-fbp5500: A diverse benchmark dataset for multi-paradigm facial beauty prediction," 2018.

[19] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[20] Xiaohan Jin, Ye Qi, and Shangxuan Wu, "Cyclegan face-off," *arXiv preprint arXiv:1712.03451*, 2017.

[21] 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*, 2016, pp. 2234–2242.

[22] Geoffrey E Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors," *arXiv preprint arXiv:1207.0580*, 2012.

[23] Sergey Ioffe and Christian Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *arXiv preprint arXiv:1502.03167*, 2015.

[24] Dmitry Ulyanov, Andrea Vedaldi, and Victor S. Lempitsky, "Instance normalization: The missing ingredient for fast stylization," *CoRR*, vol. abs/1607.08022, 2016.