Generating Anime Avatars Using Generative Adversarial Networks
Final Report for Deep Learning Course

Wang Ao
Tsinghua University
2017010395
wa17@mails.tsinghua.edu.cn

Sun Ziping
Tsinghua University
2015013249
me@szp.io

Cui Yanfei
Tsinghua University
2017012326
929881841@qq.com

Abstract

In the past few years, computer vision has achieved rapid development, and Generative Adverarial Networks [5] have attracted wide attention since proposed in 2014. In this paper, we aimed to generate from noise and to transfer from existing pictures to anime avatars, which is an interesting and practical task. There have been many paper achieving this kind of task, such as DCGAN [12], CycleGAN [24] and CartoonGAN [1]. In this paper, we reimplement these three models and tune them to achieve better performance on this specific task. And then, we give several novel loss functions to make models perform better, such as face landmark loss. At last, we compare the different results between these three models and come to the conclusion that DCGAN is generally a good choice to generate images from noise, CycleGAN performs the best on image tansfering while CartoonGAN tends to preserve image details, which results to a poorer performance.

1. Introduction

Generally speaking, image generating and style transferring belongs to the generative problem. And deep generative models have achieved impressive success in this field. Among all deep generative models, the most outstanding ones, for the time being, are Generative Adversarial Networks (GANs) and Variational Autoencoders (VAEs). In our work, we focus on GANs.

Traditional GANs consist of two parts, the generator and the discriminator. The generator tries to generate fake outputs similar to the real data, while the discriminator tries to tell the real ones apart from the fake ones. When they trained together, the discriminator drives the generator to generate realer outputs, and meanwhile, the generated realer outputs also drive the descriminator to jugde more correctly. And finally, in order to deceive the discriminator, the generator can be well-trained for the generating task, that's exactly what we want. It is also notable that these two parts may not necessarily be trained exactly simultaneously. One part can be pretrained, or they can be trained alternatively.

As mentioned above, there are two ways to generate anime avatars:

- directly generate a anime avatar from random noise
- generate corresponding anime avatar from real human avatar

For the first approach, DCGAN is among one of the most state-of-the-art models. It is a variation of GAN, in which transposed convolutions are added. It is mainly aimed for image generating tasks. Transposed convolutions are the inverse operation of normal convolutions, and they broadcast input values through the kernels, resulting a larger output shape, which is useful for up-sampling.

As for the second approach, according to the dataset given, it can be classified to two kinds of problems. One is based on paired image transfer, while the other is unpaired. In our work, dataset is unpaired. CycleGANs perform best on this task. CycleGAN consists of two generators and two descriminators. Source images first goes through one generator resulting in fake target images and then goes through the other generator producing reconstruction source images. And for the target images, they are processed in a similar way. The origin and reconstruction images can be compared to get a cylce consistency loss, which helps the model to be trained.

We also take a look at CartoonGAN. It is suitable for image transferring but without a cyclic structure. Two novel losses are proposed: (1) a semantic content loss, which forces the features of photos and cartoons to be close, and (2) an edge-promoting loss to preserve clear edges, since edges in cartoons are generally clearer than that of photos.

However, it still remains to be a difficult question how to evaluate the performance of a generative model. So we just manually analyze the generated images. Though this accessment method is subjective, the result is generally universal.

2. Related Work

2.1. Non-photorealistic rendering (NPR)

In order to mimic specific artistic styles (including animation [14]), many automatic and semi-automatic NPR algorithms have been developed. Most of the works are animated by using a simple shadow rendering method [16]. One technique, called *cel shading*, is widely used in the creation of games, animations, and movies, saving artists a lot of time [11].

To mimic the cartoon style, people have developed various methods to create images with flat shadows. These methods either use image filtering [19] or use specific transformation formulas [21] in optimization problems. However, it is difficult to capture rich artistic styles using only simple mathematical formulas. In particular, filtering or optimizing the entire image does not create the high-level abstractions that artists typically require. For portraits, people also have special algorithms [22, 15], in which semantic segmentation can be automatically derived by detecting facial components. However, these methods cannot handle general images.

2.2. Convolutional neural networks

With the great success of convolutional neural networks [8, 9] in the field of computer vision, people are looking forward to their performance in the field of image style transfer and image generation. Compared with the traditional complex NPR algorithm [16, 11], CNN is indeed more convenient and more applicable. For example, in the task of image style transfer, the VGG network [18] has a good ability to extract picture features.

For the style and content of the image, Gatys et al. [4] first proposed a CNN-based neural style transfer (NST) method that can transfer the style of a picture from one picture to another. They use a feature map of a pre-trained VGG network to extract picture content and optimize the resulting image, so that it can match the corresponding texture information described by the global Gram matrix [3] while retaining the original content of the image. However, such operations caused a serious loss of the edge information and shadow information of the picture.

2.3. Image synthesis with GANs

The other method using GANs [5] has achieved great imporvement. It has achieved the state of the art results in the fields of text-to-image translation [13], image inpainting [23], and image super-resolution [10], etc. The key idea of the GAN model is to train two networks (generator and discriminator). Iteratively, the adversarial loss provided by the discriminator transforms the generated image into a target manifold [23].

Some literatures [2, 6, 7] provided solutions using GANs for pixel-level image synthesis problems. However, these methods require paired data sets during the training process, but such high-quality data sets are difficult to obtain and therefore cannot be used for our training.

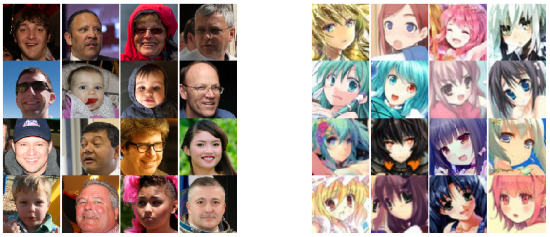
3. Approach

3.1. Prepare Dataset

We downloaded three datasets. One is real person photos, and the other two belongs to anime avatars. The datasets are shown in Table 1, and some examples drawn from datasets are shown in Figure 1.

name	category	no of samples
Anime	labeled anime avatars	115087
Thumbs	unlabeled real photos	3666
Faces	unlabeled anime avatars	51223

Table 1: The dataset we use in our project. We will refer to these dataset using their name in this paper.



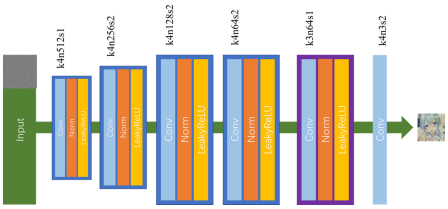
(a) Examples from Thumbs (b) Examples from Faces

Figure 1: Some example images from datasets

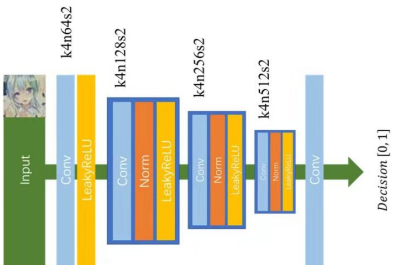
3.2. Generate from Noise Using DCGAN

The architecture of our DCGAN model is shown in Figure 2. The input of the generator is a noise of length 100 drawn from normal standard normal distribution. Then the input is expanded to the shape of $512 \times 4 \times 4$ and goes through another 3 transposed convolution layer upsampling to the shape of $64 \times 32 \times 32$, each of which reduce the input channels to half and expand the input size twice. And then the input is processed by a size-preserving convolution layer called extra layers by us, which changes the channel number to 3. Finally the input goes thought a transposed convolution layer, which doubles the size. Every convolution layer except the last one is followed by a batch normalization layer and a leakly ReLU activiation layer. The last convolution layers outputs directly to the Tanh activiation layer. Thus, the generated image has a size of $3 \times 64 \times 64$, of which elements are in range $(-1, 1)$.

The discriminator is like a reverse version of the generator. The input size is $3 \times 64 \times 64$, it is first converted



(a) Generator Architecture of DCGAN



(b) Discriminator Architecture of DCGAN

Figure 2: Architecture of DCGAN. Here the number of k represents kernel size, n is the output size and s is stride

to the size of $64 \times 32 \times 32$ by a convolution layer. And then it undergoes 3 convolution layers, each of which doubles the channel number and cut the size to half. Then it goes through a channel-and-size-preserving convolution layer, and the output is of size $512 \times 4 \times 4$. Finally, the data is processed by one convolution layer, of which the output is of size $1 \times 1 \times 1$. It is notable that the descriminator does not use any fully connected layer.

We use Anime dataset as the input of DCGAN. The input is normalized to standard normalization distribution, because the fake images also obey this distribution. The training procedure consists of two phases. In the first phase, the generated fake images and the real images are feed to discriminator and measured with binary cross entropy (BCE) loss. In the second phase, the loss function tries the make the generator generate images that trick the descriminator into thinking they are real.

3.3. Transfer from Real Photos Using CycleGAN

Figure 3 shows the architecture of out CycleGAN. CycleGAN consists of two generators and two discriminators. The generators use two down-sampling convolution layers, 4 ResNet blocks and two up-sampling convolution layers. Since the model is a bit too large to fit in our computer's GPU memory, the number of channels is made smaller than DCGAN. At first, we use one down-sampling and one up-sampling, but preliminary experiment show that shallower down-sampling and up-sampling layers lead to poorer generization ability of the networks. So we change the numbers of down-sampling and up-sampling layers to two. And it seems the networks work much more better.

The discriminators are similar to the ones of DCGAN. But the outputs contain not only one element. The discriminators are binary classification models, so the output shapes do not matter, as the real labels and fake labels can be any shape of ones and zeros.

We use Thumbs dataset and Faces dataset as the input of CycleGAN. Hereinafter, we will refer these two dataset as A and B, two generators as G_{A2B} and G_{B2A}, and two discriminators as D_A and D_B. Similar to DCGAN, the training process of CycleGAN also has two steps. One is updating generators, and the other is updating discriminators. In the updating generator step, there existing six losses of three kinds. They are:

- Identity loss: $G_{A2B}(B)$ should be similar to B, and $G_{B2A}(A)$ should also be similar to A. This kind of losses is measured by L1.
- GAN loss: $D_B(G_{A2B}(A))$ and $D_A(G_{B2A}(B))$ need to be classified as real. This kind of losses is measured by MSE. It can be measured by BCE, but experiments show MSE is better.
- Cycle loss: $G_{B2A}(G_{A2B}(A))$ should be similar to A, and $G_{A2B}(G_{B2A}(B))$ should be similar to B. This is measured by L1.

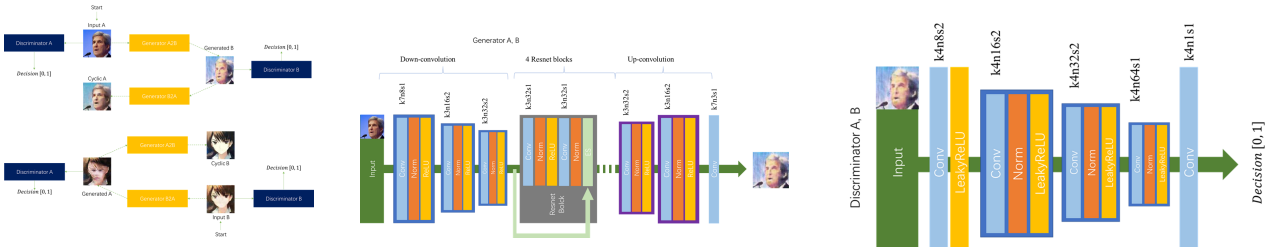
These three losses are sumed up at the ratio of 5:1:10. As for the updating descriminator step, the losses try to make $D_B(G_{A2B}(A))$ and $D_A(G_{B2A}(B))$ classified as fake and $D_A(A)$, $D_B(B)$ classified as real. Furthermore, we use a fake image pool [17] to help the descriminator remember its historical decision.

3.4. Transfer from Real Photos Using CartoonGAN

CartoonGAN has no cyclic structure. As shown in Figure 4, its generator consists of two down-sampling convolution layers, 8 ResNet blocks and two up-sampling convolution layers. Its discriminator consists of 7 convolution layers including two down-sampling layer.

The datasets CartoonGAN uses are the same as the ones CycleGAN uses. But there is one special preprocessing step, which creates another dataset based on Faces (referred as B) through a process called edge promoting. We will refer this generated dataset as E hereinafter. During the process, first of all, edges are detected, and then the pixels near the edges are convolved with Gaussian kernel. This makes the edges in the pictures become smoother, in other words, less sharper. The idea is that the picture of which edges are smooth should be classified as real images, aka fake anime images, because cartoon images generally have sharper edges. In addition, CartoonGAN uses a pretrained VGG19 as feature extraction model for calculating content loss. We will refer it as VGG.

We will refer the generator as G and the discriminator as D. During the training step of D, the BCE losses drive D to classify $D(B)$ as real, $D(G(A))$ and $D(E)$ as fake. During

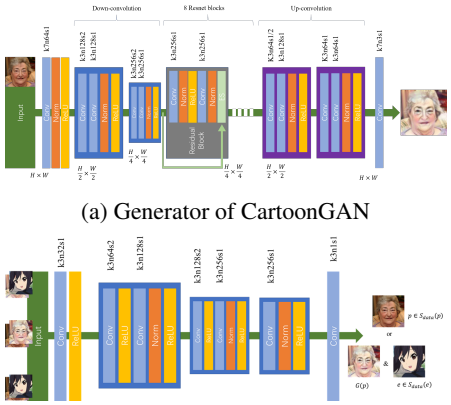


(a) Whole Architecture of CycleGAN

(b) Generator of CycleGAN

(c) Discriminator of CycleGAN

Figure 3: Architecture of CycleGAN



(a) Generator of CartoonGAN

(b) Discriminator of CartoonGAN

Figure 4: Architecture of CartoonGAN

the training step of G, three losses are added together:

- GAN loss: $D(G(A))$ should be classified as real.
- Content loss: $VGG(A)$ and $VGG(G(A))$ should be similar measured by L1 loss, because the content of the source and generated images should be same.
- Style loss: $Gram(VGG(B))$ and $Gram(VGG(A))$ should be similar measured by L1 loss. Here $Gram$ means the Gram matrix. The idea is that the style of the target and generated images should be same.

Another notable thing is that generator goes through a pre-trained process minimizing the content loss.

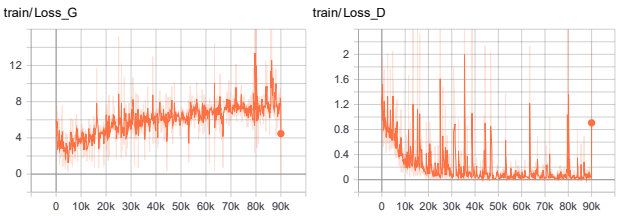
In addition to the origin work, we also tried to add some loss functions to help train the networks. These losses are:

- Face landmark loss: Give high priority to the key points on faces, so that the important features on faces can be preserved. This is inspired by [20].
- Grayscale loss and color loss: Proposed by someone, criticized and abandoned by me.

4. Experiment

4.1. DCGAN

We trained DCGAN with batch size 128, learning rate 0.0003 and epoch 100. The losses is shown in Figure 5. From Figure 5, we can draw the conclusion that at first both the generator and the discriminator develop well, but soon the descriminator gets the upper hand leading to the increase of generator loss.



(a) Generator Loss

(b) Discriminator Loss

Figure 5: Training loss curve for DCGAN

Figure 6 shows anime avatars generated by DCGAN from random noises, from continuous noices and at different epochs. We can see that DCGAN works and generates images similar to anime avatars, but the generated images are less clear and confusing. Especially the faces features of intermediate transition images are strange. From the generated images at different epochs, we can see that the generator reaches a well-trained start at a very early stage, and subsequent training only has minor effects.

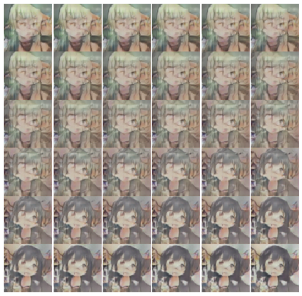
Nevertheless, when compared to our other models, DCGAN produced really good results.

4.2. CycleGAN

CycleGAN is trained with 128 batch size and 300 epochs. During the first 120 epochs, the learning rate is 3×10^{-4}, and after it, the learning rate is decreased to 3×10^{-5}.

Figure 7 illustrates the training loss trends over time. It is obvious that the identity losses and cycle losses decreases

(a) Avatars generated from random noises



(b) Avatars generated from continuous noises

(c) Avatars generated from different epochs. The leftmost is selected from epoch 10, and the rightmost is selected from epoch 100

Figure 6: Avatars generated by DCGAN

gradually, while the GAN losses increases and seems to be unstable. The discriminator losses is nearly to be zero. All these losses indicate two facts: (1) the discriminator completely defeats the generator, (2) the generator is good at preserving inputs and poor at transferring, which means it behaves like an identity map, and that's not what we want.

Let's take a look at the generated results shown in Figure 8. The origin images and the reconstructed ones look like exactly the same, which indicates the reconstruction ability of our CycleGAN is quite good. The generated images, however, are not in good form. Those generated from real photos get a little cartoon style but too colorful, and the face features of those images are distorted. Those images generated from anime avatars are patched unnaturally with a human face. The directions of the origin faces and the patched ones are not consistent.

We can conclude that CycleGAN prove the ability to transfer between different domains, but not in a good manner. It may be because the domains of real photos and anime avatars have too much difference or because we place too much weight on the cycle loss limiting the transferring ability of the networks. Further experiments may be needed to analyze the root cause of the poor performance.

4.3. CartoonGAN

5. Conclusion

References

[1] Y. Chen, Y.-K. Lai, and Y.-J. Liu. Cartoongan: Generative adversarial networks for photo cartoonization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9465–9474, 2018.

[2] V. Dumoulin, I. Belghazi, B. Poole, O. Mastropietro, A. Lamb, M. Arjovsky, and A. Courville. Adversarially learned inference. *arXiv preprint arXiv:1606.00704*, 2016.

[3] L. A. Gatys, A. S. Ecker, and M. Bethge. Texture synthesis and the controlled generation of natural stimuli using convolutional neural networks. *arXiv preprint arXiv:1505.07376*, 12:4, 2015.

[4] L. A. Gatys, A. S. Ecker, and M. Bethge. Image style transfer using convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2414–2423, 2016.

[5] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[6] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1125–1134, 2017.

[7] L. Karacan, Z. Akata, A. Erdem, and E. Erdem. Learning to generate images of outdoor scenes from attributes and semantic layouts. *arXiv preprint arXiv:1612.00215*, 2016.

[8] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[9] S. Lawrence, C. L. Giles, A. C. Tsoi, and A. D. Back. Face recognition: A convolutional neural-network approach. *IEEE transactions on neural networks*, 8(1):98–113, 1997.

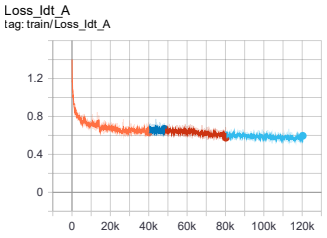
[10] C. Ledig, L. Theis, F. Huszár, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang, et al. Photo-realistic single image super-resolution using a generative adversarial network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4681–4690, 2017.

[11] R. R. Luque. The cel shading technique. Technical report, Citeseer, 2012.

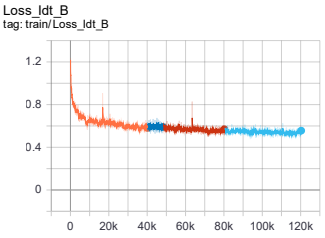
[12] A. Radford, L. Metz, and S. Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. In *International Conference on Learning Representations*, 2016.

[13] S. Reed, Z. Akata, X. Yan, L. Logeswaran, B. Schiele, and H. Lee. Generative adversarial text to image synthesis. *arXiv preprint arXiv:1605.05396*, 2016.

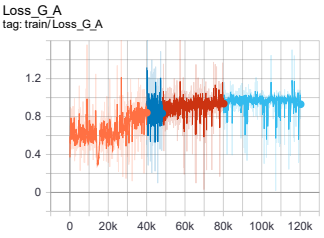
[14] P. Rosin and J. Collomosse. *Image and video-based artistic stylisation*, volume 42. Springer Science & Business Media, 2012.



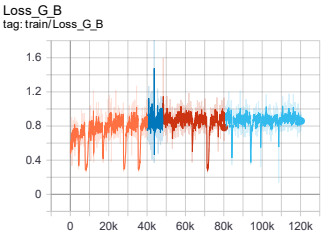
(a) Identity loss: similarity loss between $G_{A2B}(B)$ and B



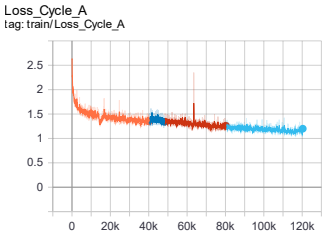
(b) Identity loss: similarity loss between $G_{B2A}(A)$ and A



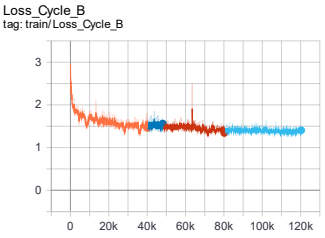
(c) GAN loss: BCE loss of $D_B(G_{A2B}(A))$ and real label



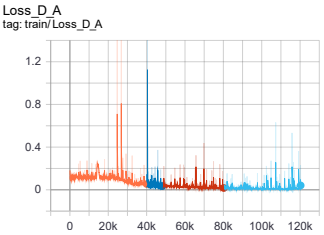
(d) GAN loss: BCE loss of $D_A(G_{B2A}(B))$ and real label



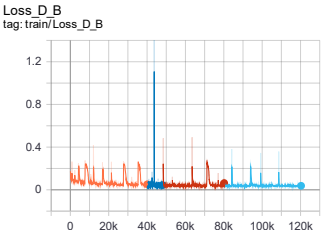
(e) Cycle loss: similarity loss between $G_{B2A}(G_{A2B}(A))$ and A



(f) Cycle loss: similarity loss between $G_{A2B}(G_{B2A}(B))$ and B

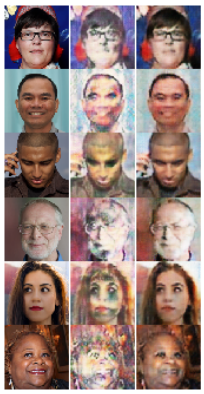


(g) Discriminator loss on A

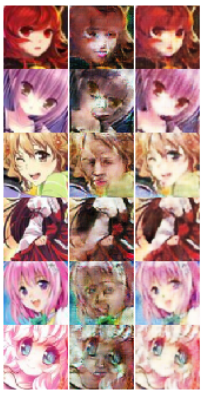


(h) Discriminator loss on B

Figure 7: Training loss curve for CycleGAN. The training process is interrupted three times and resumed from the checkpoints



(a) From left to right are A, $G_{A2B}(A)$, $G_{B2A}(G_{A2B}(A))$



(b) From left to right are B, $G_{B2A}(B)$, $G_{A2B}(G_{B2A}(B))$

Figure 8: CycleGAN generated images: origin, generated and reconstructed ones

[15] P. L. Rosin and Y.-K. Lai. Non-photorealistic rendering of portraits. In *Proceedings of the workshop on Computational Aesthetics*, pages 159–170. Eurographics Association, 2015.

[16] T. Saito and T. Takahashi. Comprehensible rendering of 3-d shapes. In *ACM SIGGRAPH Computer Graphics*, volume 24, pages 197–206. ACM, 1990.

[17] A. Shrivastava, T. Pfister, O. Tuzel, J. Susskind, W. Wang, and R. Webb. Learning from simulated and unsupervised images through adversarial training. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2107–2116, 2017.

[18] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[19] H. Winnemöller, S. C. Olsen, and B. Gooch. Real-time video abstraction. In *ACM Transactions On Graphics (TOG)*, volume 25, pages 1221–1226. ACM, 2006.

[20] R. Wu, X. Gu, X. Tao, X. Shen, Y.-W. Tai, and J. Jia. Landmark assisted cyclegan for cartoon face generation. *arXiv preprint arXiv:1907.01424*, 2019.

[21] L. Xu, C. Lu, Y. Xu, and J. Jia. Image smoothing via l 0 gradient minimization. In *ACM Transactions on Graphics (TOG)*, volume 30, page 174. ACM, 2011.

[22] M. Yang, S. Lin, P. Luo, L. Lin, and H. Chao. Semantics-driven portrait cartoon stylization. In *2010 IEEE International Conference on Image Processing*, pages 1805–1808. IEEE, 2010.

[23] R. Yeh, C. Chen, T. Y. Lim, M. Hasegawa-Johnson, and M. N. Do. Semantic image inpainting with perceptual and contextual losses. *arXiv preprint arXiv:1607.07539*, 2:3, 2016.

[24] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networkss. In *Computer Vision (ICCV), 2017 IEEE International Conference on*, 2017.