Human to Anime Face Generation using CycleGAN

Ross Bernstein
New York University
ross.bernstein@nyu.edu

Guido Petri
New York University
guidopetri@nyu.edu

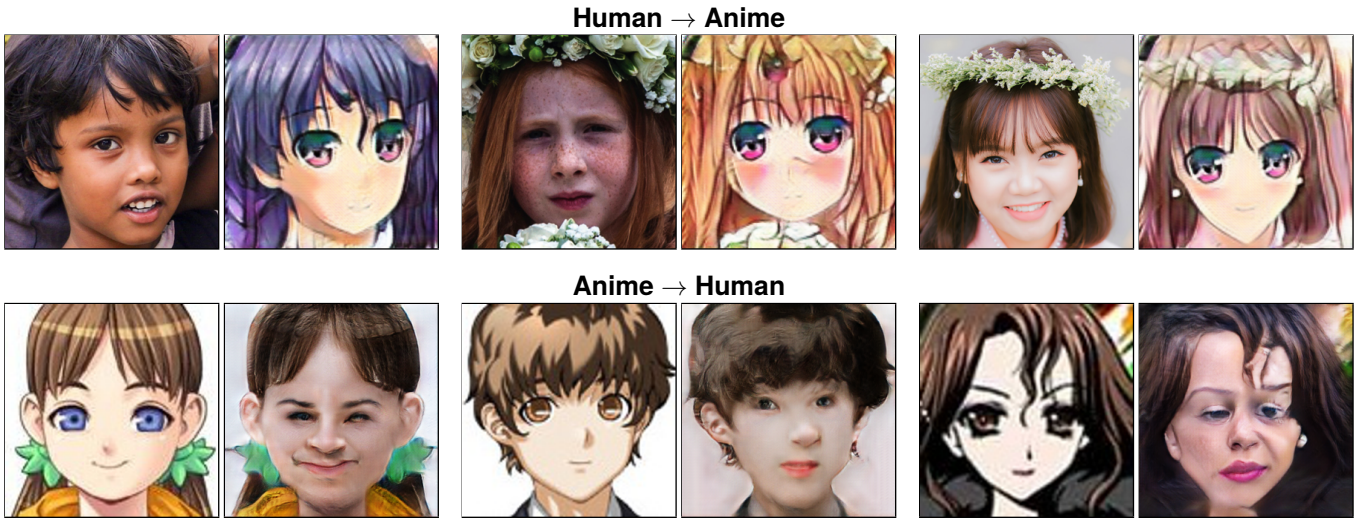

Figure 1: Our model transforms images of human faces into anime faces and vice versa

***Abstract*—In this paper, we present a method for transforming images of human faces into anime faces using CycleGAN. As GANs are notoriously difficult to train, we first discuss the implementation details that were required to produce stable results. We then experiment with changes to the original model, and evaluate the effects of different architecture choices with regards to the practical applications of anime face generation.**

I. Introduction

Technology designed to change one's appearance has become widely used on the internet, especially on social media. The mobile app Snapchat owes much of its popularity to the incorporation of filters that can change the user's face in a multitude of ways. In addition, the task of transforming images of people into animated characters has applications within the entertainment industry, where automating the motion capture process can significantly reduce the cost of producing movies and video games.

Image-to-image translation tasks have traditionally required training models on paired images. The requirement for paired image data is extremely limiting, as such data can be very difficult or impossible to obtain. CycleGAN is a technique for performing image translation without the need for paired images. With unpaired image-to-image translation, we seek to learn a mapping that can transform an image from the source domain such that the generated output is indistinguishable from a real image in the target domain. CycleGAN also learns the reverse mapping, allowing us to transform images from either domain into the other.

In this project, we implement CycleGAN and train it to transform human faces into anime faces. Our objective is to verify the reproducibility of the original paper, and to explore possible refinements to the model.

A. Related work

The original CycleGAN (Zhu et al. 2017) paper details the implementation and provides source code written in PyTorch. However, the paper also mentions that its results are not very stable. There are very few instances in which images generated by the model are truly indistinguishable from the target distribution. Additionally, the network is not generalizable. Performance on a holdout set is typically far worse than on the training set.

Nonetheless, several works have advanced CycleGAN. In our project's field of face generation, (Wang et al. 2019) uses a CycleGAN based network to generate faces belonging to different age groups; (Lu, Tai, and Tang 2018) expands CycleGAN to add covariates, calling this Conditional CycleGAN, and uses the network to generate human faces with specific attributes; (Jin, Qi, and S. Wu 2018) uses CycleGAN to generate facial expressions transferred from a different person. In all these examples, it is clear that CycleGAN does produce some outputs that look realistic, but for the most part, they are not realistic enough to fool humans.

(R. Wu et al. 2019) approaches a very similar problem to ours: they generate cartoon characters from human faces. In addition to the standard CycleGAN network, they also use landmark generation in order to assist the network in generating cartoon faces. The outputs of the network are far

superior to those of plain CycleGAN.

(Kodali et al. 2017) and (Chu, Minami, and Fukumizu 2020) discuss convergence options for GANs in general, creating new network architectures that are more stable than standard GANs. (Shrivastava et al. 2017) shows that updating the discriminator network with a history of generated images can significantly improve training stability.

II. Method

A. Model

Our model is composed of two architectures: the generator, which is adopted from (Johnson, Alahi, and Fei-Fei 2016), and the discriminator, which utilizes a *PatchGAN* architecture. CycleGAN requires two instances of each model, since we need a generator to map human to anime faces, one to map anime to human faces, and separate discriminators for both anime faces and human faces as well.

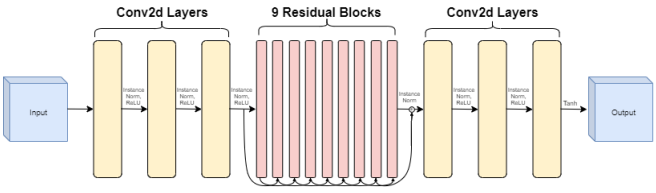

Fig. 1. Generator architecture

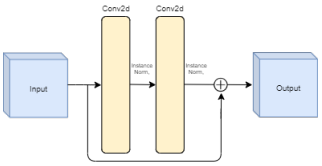

Fig. 2. A single residual block

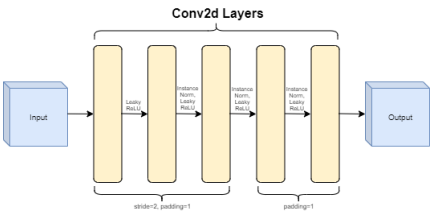

Fig. 3. Discriminator architecture

The generator was composed of convolutional layers interspersed with ReLU nonlinearities and instance norm layers, as well as residual blocks composed of the same. The discriminator was composed similarly, except it used LeakyReLU with a negative slope of 0.2 as a nonlinearity, and did not have any residual blocks.

Initially, we implemented these architectures from scratch, solely based on the formulations presented in the original paper. However, we later incorporated several implementation details to improve stability, as recommended by (Zhu et al. 2017). The main difference between our first implementation and the original implementation is the final layer of PatchGAN, where we used a sigmoid function and the original implementation used average pooling to reduce the input to a single element per sample. Average pooling allows for different sized PatchGANs to be used with the exact same code, whereas our implementation necessitated a different convolution as the final layer before the activation to accept different sized inputs.

B. Data

Fig. 4. Samples from each dataset used to train the model

We utilized two datasets for our project. The first one consisted of human faces, fake and real; it can be found on Kaggle under "Real and Fake face detection"[1]. We only utilized the real faces from this dataset for our training procedure. Our second dataset was composed of anime faces and can be found on Github[2]. This dataset had to be slightly cleaned, since there were several images with zero byte size, and several thousand images that were too small for our networks that had to be filtered out. As a result, our combined dataset was composed of 1,081 real human faces, with resolution 600x600, and 22,805 anime faces of varying resolution between 82x82 and 220x220. The faces were centered in the images, but were set at different angles.

Various transforms were applied to the data. First, all images were resized to 287x287 with bicubic interpolation to standardize the inputs to the network. Next, a random 256x256 crop was applied, followed by a random horizontal flip with probability 0.5. These random transforms were used to increase the generalizability of the model and prevent overfitting. Finally, all images were normalized.

Since we are training a generative network with no specific target, we did not use a validation or testing set during training.

[1]https://www.kaggle.com/ciplab/real-and-fake-face-detection
[2]https://github.com/bchao1/Anime-Face-Dataset

C. Model training

Training was performed with the ADAM optimizer, and momentum parameters $(0.5, 0.999)$. Later, to help stabilize training, we used a buffer of real images with which to train the discriminator (Shrivastava et al. 2017). The model weights were initialized randomly, sampling from $\mathcal{N}(0, 0.02)$ for the convolution layers, and sampling from $\mathcal{N}(1, 0.02)$ for the batch norm layers. The bias values for the batch norm layers were initialized at zero.

We utilized the L1 loss function for the cycle loss and identity loss components of our loss function, and mean squared error loss for the adversarial loss component.

All models were trained with a learning rate of 0.0002 for the first 100 epochs, with a linear decay for the following 100 epochs.

D. Evaluation Metrics

GANs are notoriously difficult to evaluate, particularly when they are used for artistic endeavors. As will be discussed in the next section, we found the results of our experiments to be highly subjective. For this reason, we decided to forgo using quantitative metrics in favor of relying on human judgement, with an in-depth analysis that considered various practical applications of the model.

III. Experiments

A. Reproducing the original paper

Our first experiment was a strict reproduction of the original CycleGAN, based on the mathematical formulations described by (Zhu et al. 2017). We implemented both the CycleGAN and PatchGAN architectures as described in the paper's appendix sections. The paper did not have many details for the architectures, and so we had to use our best judgment for the following items:

- The residual block architecture is not strictly defined. We implemented the addition operation before the final activation function.
- We used ReLU as the model's final layer, since the description of the model in the original paper's appendix implied this usage.
- The final layer of PatchGAN was not clearly defined. We used a sigmoid operation to reduce the input to the output space $[0, 1]$, since we wanted to classify whether an image was real or fake.
- We used the described 70x70 input for PatchGAN for our network, using a final kernel size that would yield a single element output.

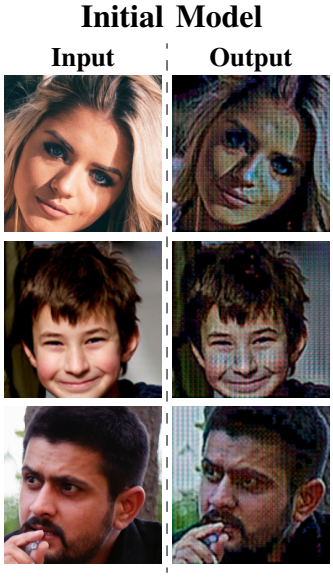

Fig. 5. Results of the first model

As shown in Figure 5, the initial implementation failed to produce desirable results. After several attempts, it became clear that adopting some of the training strategies utilized in other papers would be crucial to the success of the model. The following changes were made:

- Using least-squares GAN loss rather than negative log-likelihood
- Using an image buffer to update the discriminator, as recommended (Shrivastava et al. 2017)

As shown in Figure 6, incorporating the aforementioned strategies led to much better results.

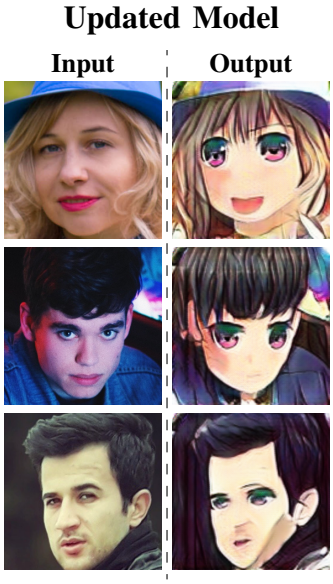

Fig. 6. Results after utilizing strategies to stabilize training

B. Using smaller input PatchGAN

After achieving a stable implementation, our first experiment was to reduce the size of the patches fed to the

discriminator. We trained a model that sampled 32x32 patches of the input image. This was done in order to measure the impact of using a smaller-scale discriminator to train our models. We expected the smaller discriminator to help with the generation of small structures, but not with the overall image, similar to how Markov chains used for text generation can make sense in a local context but not a global context. However, when using a patch size this small, the network became unstable and was unable to learn at all.

C. Ablation Study

We conduct ablation studies by reducing the number of residual blocks used in the generator. Our previous model used 9 residual blocks, as recommended by (Zhu et al. 2017). Additional models were trained with 3 and 6 residual blocks. Reducing the number of residual blocks has a significant effect on the number of trainable parameters in the model. The model with 3 residual blocks has just 38% of the amount of trainable parameters as the model with 9. Thus, these experiments were performed with the expectation that the model's capacity would decrease, producing less realistic results.

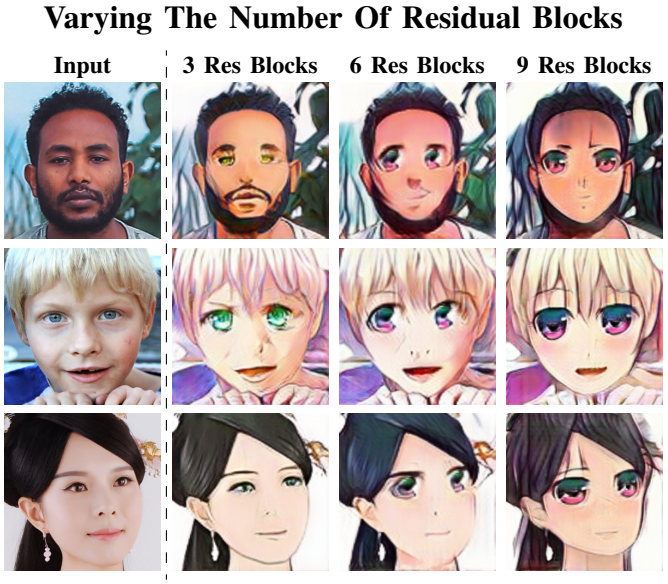

Fig. 7. A comparison of the results produced by models with generators containing 3, 6 and 9 residual blocks

An example of the outputs from each model is shown in Figure 7. Varying the number of residual blocks appears to change the output in a few distinct ways. With 3 residual blocks, the output looks more similar to the real human face. This is most apparent in the size of the eyes and mouth, and the texture of the hair. The overall shape of the face also becomes rounder. In many ways, increasing the amount of residual blocks leads to results that are aesthetically simpler, despite having a much higher capacity. This is likely due to the fact that stylistically, anime faces are much simpler than human faces, and the model with 9 residual blocks produces outputs that are closer to the target distribution of anime faces.

In order to properly evaluate the results of these models, we must consider their intended usage. Such a tool might be used in the entertainment industry to streamline the production of animated content, or by individuals, simply for the novelty of seeing themselves as an anime character. In either scenario, the model should be able to generalize well enough that it can be used by many different people, and produce realistic anime faces that still resemble the user. We proceed by testing the models with various inputs that may reveal any strengths or weaknesses of their ability to function in a practical setting.

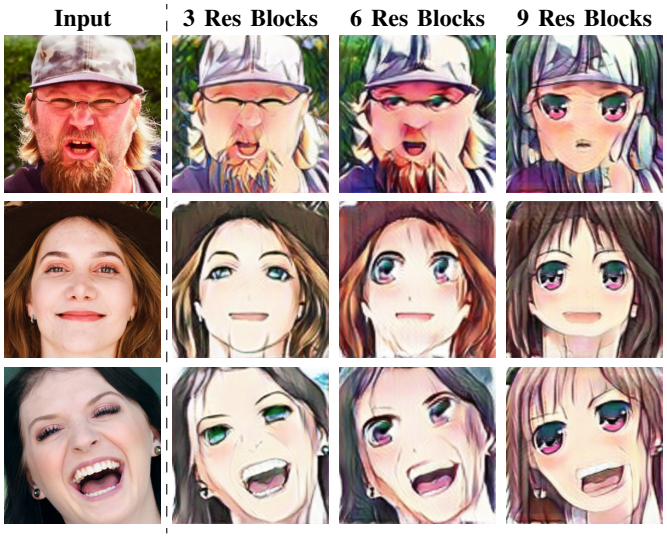

Fig. 8. Comparing each model's ability to generate faces that can convey human emotions

It is important for the generated anime faces to be able to portray basic human expressions such as anger or happiness. To test this, we ran the model on some more expressive faces. We can see in Figure 8 that the anime faces generally do retain the emotion of the input. Since the models with 3 and 6 residual blocks look more human, they perform better at this task than the model with 9 residual blocks. The man in the first row appears to be angry, and this look is retained by every model. With 9 residual blocks, the eyes all become large, even if the person's eyes are closed. Despite this, the sense of anger is still apparent because the anime face appears to have tensed eyebrows. Conversely, large eyes are a problem for the laughing woman in the third row. In the second and third columns, it is difficult to tell whether she is laughing or frightened.

It is crucial that an anime character in one scene looks like the same character in another scene. Figure 9 shows the results of running the models on two images of the same person, but with a different pose and lighting in each image. The 3 and 6-block models have some issues with the shadows on the woman's face in the second row, but still resemble the same character they generated in the first row. It is worth noting that the improper shading has soft streaks in it. This indicates that these models tend to treat dark regions on the face as if they

are hair. The higher capacity 9-block model performs the best here, the faces it generated look the most similar across the change of pose.

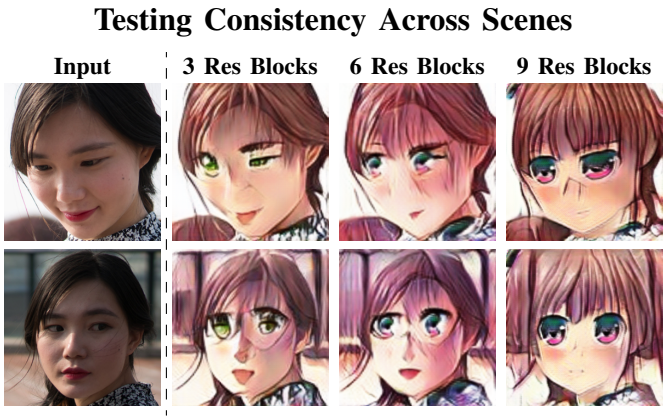

Fig. 9. Test to determine the consistency of an anime character's look across different scenes, with a different pose or lighting

Fig. 10. Results for images of people with heavy facial hair

The models should still work properly for people with different facial features. None of the anime faces in our data contained facial hair. Thus, we expect that the models would not generalize well to people with beards. However, the results from Figure 10 are surprisingly good. In each case, the generated anime faces all clearly retain facial hair. In this case, models with less residual blocks produce hair with a more realistic texture, but it is not necessarily the case that anime hair should have a similar texture to human hair.

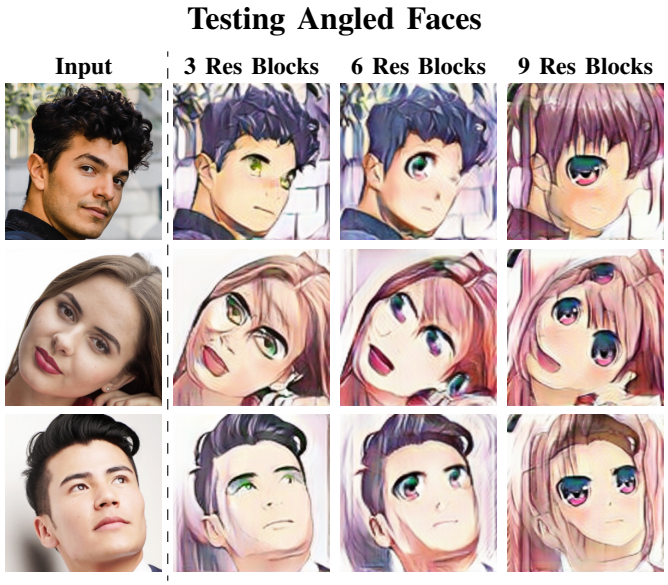

Fig. 11. Comparing each model's ability to reproduce faces at sharper angles to the camera

It is apparent that the generated anime faces look the most realistic when the inputs to the models are images of people looking straight at the camera. An experiment was conducted to determine how much worse the results become when the orientation of the human face has a more extreme angle to the camera. In Figure 11, we can see that the models do struggle more on this test. The results are worse for models with more residual blocks. In the first row, the 6 and 9-block models are both missing an eye, and the 9-block model is also missing a mouth and nose. In the second row, every model produces streaks across the woman's forehead in the direction that her hair might fall if she were angled more upright. Additionally, the orientation of the eyes in the 6 and 9-block models still looks upright.

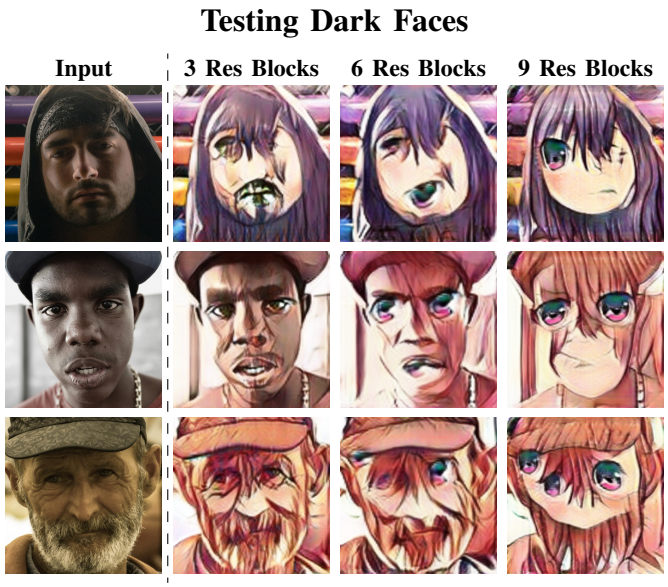

Fig. 12. Comparing each model's ability to handle dark images

We have seen that the models struggle with shadows, and tend to turn them into hair. For our last experiment, we test the models with dark images. The results of this experiment can be seen above in Figure 12. As expected, these images produce the worst results of all. As was shown previously, the models are turning shadows into hair. In the first and third rows, the models also have trouble generating the eyes properly.

IV. Conclusions

The experiments we performed highlighted the strengths and weaknesses of the models, and provided insights into how they might be improved. We observed that using fewer residual blocks allowed models to generalize better, but produced less realistic anime faces. The models struggled with producing eyes properly, handling faces set at sharp angles to the camera, and with faces containing dark shadows. Some of these issues might be remedied by performing more data augmentation, or by training on a different dataset.

In general, CycleGAN excels at transforming the pattern and texture of objects, but has difficulty with changing an object's shape. Human and anime faces have very different geometries, so the quality of anime faces generated by our models was still quite impressive.

Beyond changing our dataset or tuning the base CycleGAN model further, one promising approach involves adding new mechanisms to the model that are specialized to deal with animated face generation. (R. Wu et al. 2019) achieves impressive results in cartoon face generation by incorporating a landmark-guided discriminator. This improves the placement of generated facial features and guides the training to focus more on important regions of the face.

References

Chu, Casey, Kentaro Minami, and Kenji Fukumizu (Feb. 2020). "Smoothness and Stability in GANs". In: *arXiv:2002.04185 [cs, stat]*. arXiv: 2002.04185. URL: http://arxiv.org/abs/2002.04185 (visited on 12/17/2020).

Jin, Xiaohan, Ye Qi, and Shangxuan Wu (July 2018). "CycleGAN Face-off". In: *arXiv:1712.03451 [cs]*. arXiv: 1712.03451. URL: http://arxiv.org/abs/1712.03451 (visited on 12/17/2020).

Johnson, Justin, Alexandre Alahi, and Li Fei-Fei (2016). *Perceptual Losses for Real-Time Style Transfer and Super-Resolution*. arXiv: 1603.08155 [cs.CV].

Kodali, Naveen et al. (Dec. 2017). "On Convergence and Stability of GANs". In: *arXiv:1705.07215 [cs]*. arXiv: 1705.07215. URL: http://arxiv.org/abs/1705.07215 (visited on 12/17/2020).

Lu, Yongyi, Yu-Wing Tai, and Chi-Keung Tang (Nov. 2018). "Attribute-Guided Face Generation Using Conditional CycleGAN". In: *arXiv:1705.09966 [cs, stat]*. arXiv: 1705.09966. URL: http://arxiv.org/abs/1705.09966 (visited on 12/17/2020).

Shrivastava, Ashish et al. (July 2017). "Learning from Simulated and Unsupervised Images through Adversarial Training". In: *arXiv:1612.07828 [cs]*. arXiv: 1612.07828. URL: http://arxiv.org/abs/1612.07828 (visited on 12/17/2020).

Wang, Zipeng et al. (Sept. 2019). "How Old Are You? Face Age Translation with Identity Preservation Using GANs". In: *arXiv:1909.04988 [cs]*. arXiv: 1909.04988. URL: http://arxiv.org/abs/1909.04988 (visited on 12/17/2020).

Wu, Ruizheng et al. (July 2019). "Landmark Assisted CycleGAN for Cartoon Face Generation". In: *arXiv:1907.01424 [cs]*. arXiv: 1907.01424. URL: http://arxiv.org/abs/1907.01424 (visited on 12/17/2020).

Zhu, Jun-Yan et al. (Aug. 2017). "Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks". In: *arXiv:1703.10593 [cs]*. arXiv: 1703.10593. URL: http://arxiv.org/abs/1703.10593 (visited on 12/17/2020).