Anime Character Generation For Virtual Livestreams

Shuyang Cui Meng Zhang Shuangyang Xie
Electrical and Computer Engineering
University of California San Diego
La Jolla, CA 92093
{s3cui, mez010, shx017}@ucsd.edu

Abstract

The continuing expansion of virtual live-streaming services result in increased demand for virtual live-streamers (Vtubers), and anime characters have been one of the most popular choices for Vtuber images. By automatically generating anime Vtuber images for users, the live-streaming industry and community can be significantly benefited. In this project, we explore the deep generative models that provide an enriched variety of anime characters for live-streaming, implement four GAN variants to generate new anime Vtuber images, and compare their performances. Based on experiment results, we discuss the novelty and significance of our project, while providing advice for future research. Our *PyTorch implementations* are shared at the end of this report.

1 Introduction

The large number of virtual live-streamers, also known as Vtubers, have dramatically increased the demand of virtual characters, which are used to replace human broadcasters in livestream scenes. As more people start to livestream, rich options in the appearance of Vtubers that can help users customize their own virtual figures are expected. Anime characters are one of the most popular choices for virtual figures, and they become increasingly attractive for livestreamers as well. Automatic generation of anime characters can significantly increase the variety of virtual anime characters, provide users of livestreaming services with starting points and references of creating and customizing their own virtual anime image for livestreaming.

Driven by this objective, our work aims to implement generative models that produces new anime characters which can be used as Vtubers in livestream scenes. Our own anime generating models will be realized using four different variants of Generative Adversarial Network (GAN). We train the models over a collection of traditional anime character portraits, and will be used to generate modern-looking and high-quality Vtuber figures. Since the Vtuber image dataset is significantly smaller than the traditional anime character portrait dataset, we first generate extra Vtuber images using the larger dataset to expand the smaller Vtuber dataset using unpaired domain adaptation (UDA), which is a key functionality of CycleGAN. After the dataset expansion, we use both datasets to generate new Vtuber images. The performances of GAN variants are evaluated and compared.

The remaining content of this report is organized as follows: *Section 2* gives a general review on previous works related to our project design; *Section 3* presents our proposed methods to implement and test the generative models for anime Vtuber images; *Section 4* elucidates our experiment processes and results; In *Section 5*, we discuss the novelty and significance of our project work; In *Section 6*, we summarize our project work and give advice on future research.

37th Conference on Neural Information Processing Systems (NeurIPS 2023).

2 Related Works

2.1 Generative Adversarial Network

Generative Adversarial Network (GAN) was introduced by Goodfellow et al. in [1], where a novel framework of adversarial training between a generator and a discriminator network proposed. The architecture has achieved impressive results in image generation [2][3], image editing [4] and representation learning [3][5][6]. One of the main challenges in training the GANs under the original architecture is the instability of the training process, and much effort has been made to address this issue. Radford et al. [3] introduced a deep convolutional GAN (DCGAN) that uses convolutional neural networks (CNNs) to improve the quality of the generated images. Mirza & Osindero [7] proposed a variant called the conditional GAN, which can generate images conditioned on specific inputs, such as class labels. Another notable modification to the GAN framework is the use of Wasserstein distance. Arjovsky et al. [8] introduced a variant called the Wasserstein GAN (WGAN), which uses Wasserstein distance to stabilize the training process and improve the quality of the generated data. Karras et al. [9] introduces StyleGAN, which improved image quality and control by separating the style and content representations in GANs. The authors continued to improve the performance of StyleGANs and has presented two revised versions of the original StyleGAN architecture, i.e. StyleGAN2-ADA [10] and StyleGAN3-R [11], and they respectively feature the addition of an adaptive discriminator augmentation mechanism that significantly stabilizes training given only limited training data, and the introduction of signal processing techniques into the generative model framework.

CycleGAN, an important network in our project work, was introduced by Jun-Yan Zhu et al. [12]. The original paper presents a novel method for performing image-to-image translations without the need for paired training data. It achieves this by utilizing a cycle consistency loss to ensure that translating an image to a different domain and then back again results in the original image. This approach opened up possibilities for many tasks where paired data is difficult to obtain. Building upon the CycleGAN's concept, Choi et al. [13] proposed a single model that can handle image-to-image translation for multiple domains, significantly broadening the scope of applications. This is a step forward in reducing the complexity and computation cost when dealing with multiple domains. Alami et al. [14] introduced an attention mechanism to the CycleGAN architecture. The addition of the attention mechanism allows the model to focus on specific, relevant regions of the image during the translation process. This enhancement leads to improvements in the quality of translated images, especially where detailed or nuanced changes are required. Chu et al. [15] discovered that CycleGANs were unintentionally hiding information about the source image in the noise patterns of generated images. Effectively, the CycleGAN was acting as a steganography tool. This unexpected property raises questions about the interpretability and robustness of CycleGAN models. Fu et al. [16] proposed an extension to CycleGAN, adding a geometric consistency loss to better handle deformations between the source and target domains. The enhancement preserves geometric structures and shapes during the translation process, increasing the quality of generated images in scenarios involving significant shape or structure changes.

Along with revised variants of GANs, several important improvements for training GANs are also presented. Salimans et al. [5] introduces several training techniques for stabilizing GAN training, including feature matching and minibatch discrimination. Karras et al. [17] introduced progressive growing, a training method that gradually increases the resolution of generated images, leading to improved image quality.

2.2 Anime Character Generation

For generating anime characters in specific, two early attempts of generating anime characters were made by Mattya [18] and Rezoolab [19], who explored the use of newly proposed Deep Convolutional Generative Adversarial Network (DCGAN). Hiroshiba [20] proposed a conditional generation model for anime character faces, and codes such as IllustrationGAN and AnimeGAN have been made available online for anime face generation. Despite these efforts, the generated images often have blurring and distortion issues, making it difficult to generate industry-standard facial images for anime characters. To generate facial images of anime characters, Jin et al. [21] have explored the training of GAN models using a specialized dataset and proper application of DRAGAN resulting in a stable and high-quality model.

3 Methodology

3.1 Proposed Solution

We use two datasets for our project design. The first dataset (Anime Character Dataset) consists of 5,000 high resolution images of anime girl faces with a resolution of 256×256[†]. The other dataset (Vtuber Dataset) contains 500 existing anime images that are used as Vtubers [‡].

To start our implementation, we train the CycleGAN model using the larger Anime Character Dataset to generate extra Vtuber images that fit the data format in the Vtuber Dataset, thus expanding the smaller dataset as we described earlier. The images in first dataset are multi-angle anime portraits of traditional styles, while those in the second dataset are more modern-looking anime tachies and have been made suitable for livestreaming scenes. Since the images from both datasets are all stylish anime characters, they share similarities in their domains, so we can utilize unpaired domain adaption (UDA) technique, which is enabled by CycleGAN, to augment the smaller Vtuber dataset with the portraits from the larger Anime Character Dataset.

After the dataset expansion, we train all four generative models on the expanded Vtubder Dataset to generate new Vtuber images. Four GAN variants, i.e. DCGAN, WGAN, StyleGAN2-ADA and CycleGAN, are implemented and tested on the datasets. We use the first three GANs as baseline models and mainly focus on improving the performance of the CycleGAN model. The training takes at least 1000 epochs, and sample images generated by the models being trained, along with relevant training data such as loss curves. We will also examine the performances of these image generation models using subjective metrics, that is evaluating the visual and aesthetic qualities.

At last, the performances of these models are evaluated and compared in terms of FID and Inception Scores. Based on the results, we discuss and make comments on the implemented models. The overall project implementation is summarized as the flow diagram in Figure 1.

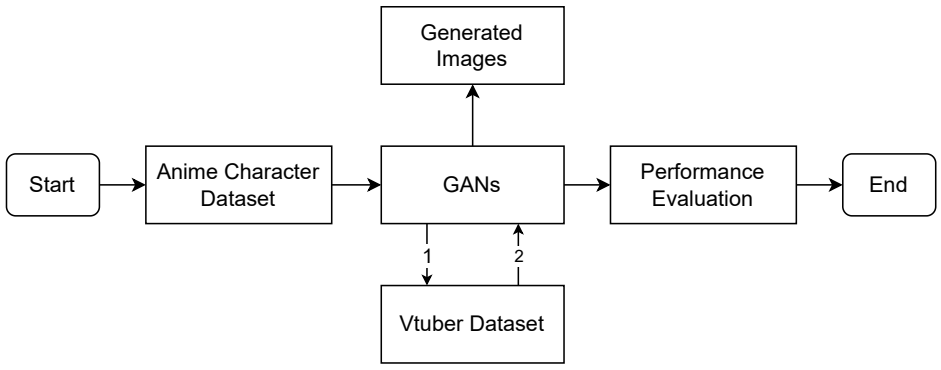


Figure 1: Flow diagram of project implementation. Notice we will expand the smaller Vtuber Dataset using the generated images (1), then train the models using the expanded dataset (2).

3.2 Model Description

3.2.1 DCGAN

DCGAN (Deep Convolutional Generative Adversarial Networks) is a generative model based on CNN architecture. It improves image quality by using strided convolutions instead of pooling, eliminating fully connected layers, and incorporating batch normalization. Strided convolutions enable the network to learn spatial downsampling and upsampling. By removing fully connected layers, the model stability is increased, and batch normalization normalizes input for improved learning and gradient flow in deep models. These changes enhance the performance and training efficiency of DCGANs significantly.

[†]The dataset is published by *TIANBAIYU-TOBY* at *Anime Girl Faces*.

[‡]The dataset is accessible at *panic3d-anime-reconstruction*, GitHub repository published by *ShuhongChen*.

3.2.2 WGAN

Wasserstein GAN (WGAN) features a new objective function based on Wasserstein distance to address issues with traditional GANs, such as mode collapse and training instability. The Wasserstein distance measures the distance between two probability distributions in terms of how much "work" is required to move one distribution into the other, and it enabled WGAN to better capture the structure of the data distribution and produce higher quality and more diverse samples. WGAN also introduced the use of a weight clipping technique to enforce the Lipschitz constraint on the discriminator, which ensures that the gradients of the discriminator are not too large and cause instability during training. However, weight clipping has limitations, and other variants of WGAN, such as WGAN-GP, introduced gradient penalty instead of weight clipping to address this issue.

3.2.3 StyleGAN

The input of traditional GAN generator is a random variable or hidden variable $\mathbf{z}$, therefore lacks the ability to control over image attributes. In order to diversify the generator from input latent space, we consider follow the work of StyleGAN proposed by Karras et al. in [9]. StyleGAN uses a more complex and sophisticated generator architecture that incorporates multiple layers of style information to understand the latent space and generate high-quality, diverse, and controllable images.

Style-based generator There are two components in the style-based generator: *mapping* network and *synthesis* network. The mapping network decouples the given latent code $\mathbf{z}$ from input hidden space $\mathcal{Z}$ to an intermediate latent space $\mathcal{W}$. The mapping network consists of 8 fully connected layers, and $\mathbf{w}$ is obtained from $\mathbf{z}$ through a series of affine transformations. The synthesis network consists of 18 layers which upsample resolution from 42 to 10242. The nonlinear mapping in the mapping network is also known as the learned affine transformation, and customizes $\mathbf{w}$ to the style $\mathbf{y} = (y_s, y_b)$, and they are used to control adaptive instance normalization (AdaIN) after each convolutional layer in the synthesis network g. The basic idea behind AdaIN is to take the feature maps from a particular layer of the generator network and apply them to the style information learned using the mapping network from a chosen style image. The AdaIN operation is given by

$$\operatorname{AdaIN}(\mathbf{x}_i, \mathbf{y}) = \mathbf{y}_{(s,i)} \frac{\mathbf{x}_i - \mu(\mathbf{x}_i)}{\sigma(\mathbf{x}_i)} + \mathbf{y}_{(b,i)}, \tag{1}$$

where each feature map x_i is normalized to zero mean and unit-variance and then the corresponding scalar $\mathbf{y}_{(s,i)}$ is standard deviation of the style information, and a bias $\mathbf{y}_{(b,i)}$ is mean of the style information. Therefore, the dimensionality of $\mathbf{y}$ is twice the number of feature of input. Unlike style transfer, StyleGAN is computed from the vector $\mathbf{w}$ instead of the style image. Finally, a way to generate random (diversity) details is provided by introducing noisy inputs to the generators. The noise input is a single channel of data consisting of uncorrelated Gaussian noise, which is fed to each layer of the generative network.

The property of StyleGAN enables it to mix multiple styles. By inputting latent code from different latent space at different levels in the synthesis network, images combing different details can be generated. The network can output images with random and natural detail by adding noise at each level of the synthesis network. A new indicator, perceptual path length, is also proposed to measure whether the image changes it style on the shortest path in latent space. The calculation is given by

$$l_{\mathcal{Z}} = \mathbb{E}\left[\frac{1}{\epsilon^2} \cdot d\Big(G\big(\operatorname{slerp}(\mathbf{z}_1, \mathbf{z}_2; t)\big), G\big(\operatorname{slerp}(\mathbf{z}_1, \mathbf{z}_2; t+\epsilon)\big)\Big)\right], \tag{2}$$

where $\mathbf{z}_1, \mathbf{z}_2, t$ are random variables: $\mathbf{z}_1, \mathbf{z}_2 \sim P(\mathbf{z})$, $t \sim U(0,1)$, $G(\dots)$ denotes the generator, $d(\dots)$ evaluates the perceptual distance between the resulting images, and slerp is linear interpolation.

StyleGAN2-ADA The training data of anime characters is limited, so we adopt StyleGAN2-ADA proposed by Karras et al. in [10]. StyleGAN2-ADA is an improved version of the original StyleGAN framework proposed in [9]. Benefitting from adaptive discriminator augmentation mechanism, the StyleGAN2-ADA can improve its generalization and robustness to different image styles and types. In our experiment, we use pixel blitting and geometric transforms, which are validated benefits

improving the quality of generated images. The augmentation strength is adjusted dynamically according to the overfitting heuristics by

$$r_v = \frac{\mathbb{E}\left(D_{\text{train}}\right) - \mathbb{E}\left(D_{\text{validation}}\right)}{\mathbb{E}\left(D_{\text{train}}\right) - \mathbb{E}\left(D_{\text{generated}}\right)}, \tag{3}$$

$$r_t = \mathbb{E}\left(\operatorname{sgn}\left(D_{\text{train}}\right)\right), \tag{4}$$

where D_{train}, $D_{\text{validation}}$, $D_{\text{generated}}$ are discriminator outputs for the training set, validation set and generated images, and sgn is the sign function. The outputs of multiple consecutive batches of each dataset are averaged individually and adjust p when r increase.

3.2.4 CycleGAN

CycleGAN is designed to learn a mapping function between two data domains, X and Y, without paired examples, which we have referred to as *unpaired domain adaptation (UDA)*. It uses two mapping functions $G : X \to Y$ and $F : Y \to X$, and associated adversarial discriminators, which are respectively denoted by D_X and D_Y. As is detailed in [12], the training process of CycleGANs involves two types of terms: *adversarial losses* [1] for matching the distribution of generated images to the data distribution in the target domain, and *cycle consistency losses* to prevent the learned mappings G and F from contradicting each other.

Adversarial Loss Adversarial loss ensures that the mapped images look similar to the target domain. For the mapping function $G : X \to Y$ and its discriminator D_Y, we try to solve the following min-max game, which is similar to the standard GAN loss. The loss is given by

$$\mathcal{L}_{GAN}(G, D_Y) = \mathbb{E}_{y \sim p_{\text{data}}(y)}\left[\log D_Y(y)\right] + \mathbb{E}_{x \sim p_{\text{data}}(x)}\left[\log\left(1 - D_Y(G(x))\right)\right]. \tag{5}$$

Notice in the equation above that G tries to generate images $G(x)$ that look similar to images from Y, while D_Y aims to distinguish between translated samples $G(x)$ and real samples y. A similar loss is defined for $F : Y \to X$ and its discriminator D_X.

Cycle Consistency Loss The main innovation of CycleGAN is the introduction of a cycle consistency loss to capture the intuition that if we translate from one domain to the other and back again, we should arrive at where we started. This is defined as

$$\mathcal{L}_{CYC}(G, F) = \mathbb{E}_{x \sim p_{\text{data}}(x)}\left[\|F(G(x)) - x\|_1\right] + \mathbb{E}_{y \sim p_{\text{data}}(y)}\left[\|G(F(y)) - y\|_1\right], \tag{6}$$

where $F(G(x))$ should be close to x and $G(F(y))$ should be close to y. This loss creates a link between the two mapping functions even in the absence of explicit paired data.

Final Objective The final objective of CycleGAN is to solve

$$G^*, F^* = \arg\min_{G,F} \max_{D_X, D_Y} \mathcal{L}(G, F, D_X, D_Y) \tag{7}$$

$$= \arg\min_{G,F} \max_{D_X, D_Y} \left[\mathcal{L}_{GAN}(G, D_Y) + L_F(F, D_X) + \lambda \cdot \mathcal{L}_{CYC}(G, F)\right], \tag{8}$$

where G^* and F^* are the optimal generators, and λ controls the relative importance of the cycle consistency loss compared to the adversarial loss. This objective function can be optimized using stochastic gradient descent (SGD) or one of its variants.

4 Experiments

We conduct two experiments using the two datasets and four GAN variants. The first experiment (Exp.I) trains all models to generate anime girl faces, the second experiment (Exp.II) focuses on CycleGAN and generates Vtuber images using both datasets as training sets.

4.1 Exp.I: Anime Girl Faces

4.1.1 DCGAN

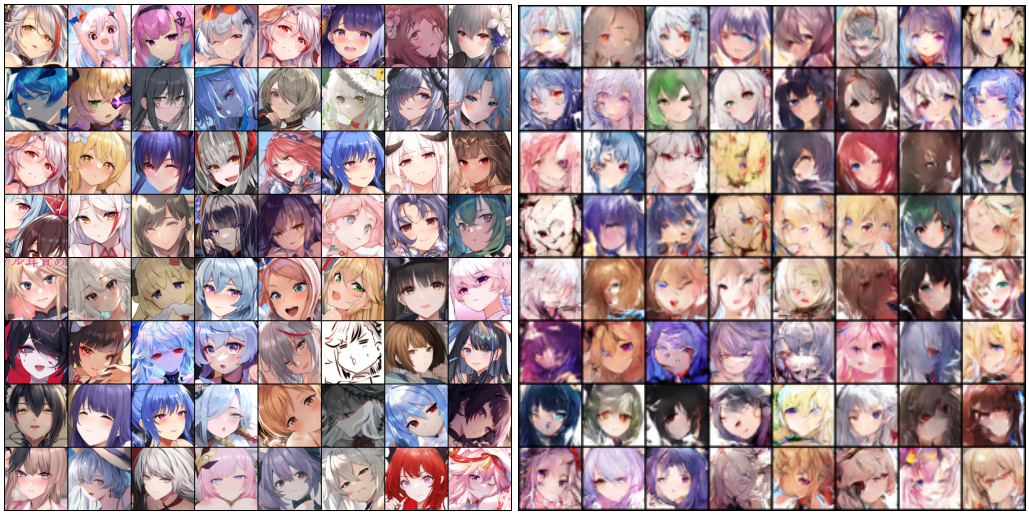
The new anime face images generated by DCGAN is illustrated in Figure 2. We can see that the overall image clarity is not satisfactory, and we can observe a significant level of noises present in the generated images. This indicates that the generated images may be very distant from the original dataset images, resulting in low FID and Inception scores.

(a) Original anime face images (b) Generated anime face images

Figure 2: Anime girl faces generated by the DCGAN model

4.2 WGAN

The new anime face images generated by WGAN is illustrated in Figure 3. Compared to DCGAN, the results are slightly better in terms of the presence of noise. However, the overall resolution of generated images is still low, and this is likely to cause low FID and Inception scores.



(a) Original anime face images (b) Generated anime face images

Figure 3: Anime girl faces generated by the WGAN model

4.2.1 StyleGAN2-ADA

The new anime face images generated by StyleGAN2-ADA is illustrated in Figure 4. As can see that the general resolution of the generated images is excellent, but the structural outlines of anime faces get blurred. This way, we expect the FID and Inception scores to be better than the previous two models, but still there may not be a significant improvement due to the structural issues. However, StyleGAN is usually used for much higher resolution images, like 128×128, 256×256, or 1024×1024 pixels, where the Inception score may not be the most representative metric to evaluate the performance, so we only take the score as one aspect of the system performance and the evaluation should be done comprehensively.

Figure 4: Anime girl faces generated by StyleGAN2-ADA (right) and original images (left)

4.2.2 CycleGAN

The new anime face images generated by StyleGAN2-ADA is illustrated in Figure 5. We can see that the overall image clarity and textures in the generated anime faces is optimal among all. However, we cannot identify clear structures of these faces by direct visual evaluation, which means the images are not aesthetically high-quality. This model may have good FID and Inception scores, but the visual effect is not satisfactory.

(a) Real Vtuber images (b) Generated Vtuber images

Figure 5: Anime girl faces generated by the CycleGAN model

4.3 Exp.II: Vtuber Images

Our early experiments with only the existing Vtuber images have produced unsatisfactory outcomes, and the generated images are excessively similar to each other. To compensate for the small size

of the Vtuber dataset, we use the UDA technique, that is essentially using our larger anime face dataset to generate new Vtuber images to expand the dataset. Using both datasets as training sets in CycleGAN allows us to generate new Vtuber images, expanding the smaller Vtuber dataset. After the expansion, the total number of Vtuber images increases from 500 to 2,000. Meanwhile, since the model uses the anime face dataset, which has more diverse styles of anime portraits, it takes advantage of its data domain and can create greater variation in the generated images.

This experiment results in the new Vtuber images shown in Figure 6. We can see that the generated images have high structural definition, and the characters have different postures, hairstyles and clothing styles. However, the color diversity present in the original Vtuber images has deteriorated in the generated images. At the same time, the colored backgrounds are absent in the generated images. Despite the shortcomings mentioned above, we expect the performance of CycleGAN when using two datasets as training sets to be optimal among all of the implemented models, due to the dataset expansion and the simultaneous use of both datasets.

(a) Real Vtuber images (b) Generated Vtuber images

Figure 6: Vtuber images generated by the CycleGAN model utilizing both training datasets

4.4 Evaluation

After generating images using our trained models, we can calculate the FID and Inception scores. The results are displayed in Table 1. We can clearly see that CycleGAN demonstrates optimal performance on anime girl face images in terms of theoretical metrics compared to its counterparts. However, as we have mentioned earlier, visual evaluation on the generated anime faces indicates the results are not good enough although they have the highest clarity among all generated images. For Vtuber images, CycleGAN delivers even better performance with both datasets providing training data. This proves that the UDA technique contributes to the enhanced quality of Vtuber image generation. (Notice that we did not train the other models on the Vtuber dataset due to the time limit of this course project.)

	Animeface		Vtuber	
	FID	Inception	FID	Inception
DCGAN	183.6	4.7	-	-
WGAN	143.7	4.3	-	-
StyleGAN2-ADA	97.7	8.1	-	-
CycleGAN	75.4	10.9	63.6	12.3

Table 1: FID and Inception scores of implemented GAN variants

5 Contribution

Our team has been aware of the limitation of the project due to the scope of this course, implementing our models and conducting experiment based on many other relevant works. Still, we have endeavored to add innovative and constructive modifications to our implemented architecture. Our contributions can be summarized as the following three points.

Dataset Expansion With UDA As we have been repeatedly mentioning in earlier sections of this report, the Vtuber dataset has a much smaller size compared to the anime face dataset. UDA, a special technique enabled by the CycleGAN architecture, allows us to use both datasets to train the generative model, while expanding the smaller Vtuber dataset with high-quality images generated by the model itself. The data domains of the datasets overlaps with each other, while having different data distributions. This brings greater variations to the generated images. We take advantage of this nature and the outcome has been excellent on Vtuber images.

Comprehensive Use of GAN Variants Implementing various GAN variants on a single dataset is essential for comparing their performance, understanding trade-offs, and identifying the optimal model for the specific data, i.e. the anime face and Vtuber datasets in our case. It allows for optimizing results through hyperparameter tuning, assessing stability and training dynamics, and gaining insights into the strengths and weaknesses of each model. Furthermore, this experimentation is helpful for obtaining practical experience, exploring ensemble methods, evaluating the robustness and generalization of generated samples, and potentially uncovering novel applications for the synthesized data.

Automatic Anime Character Generation As far as we are concerned, there is not much research in the practical application of anime character generation, and little data is available for any further academic practices. By implementing generative models to automatically generate anime characters or Vtuber images, we hope to provide people who are interested in such application with references and basis to start their own experiments. At the same time, the experiment design and our discussions may also help guide future efforts on this field of research to obtain better results.

6 Conclusion

In our project, we researched, implemented and tested four GANs to generate stylish anime Vtuber images using two different datasets. With the UDA technique enabled by the CycleGAN model, we expanded the smaller Vtuber Dataset from 500 images to 2,000 images. With both datasets providing training data, CycleGAN generates high-quality Vtuber images and achieves the highest FID and lowest Inception scores among all experiments. However, visual evaluation on the images does not align with the theoretical measurement of image qualities, and further improvements to the system and experiment design are expected. Since the effect of longer training time and a larger training dataset has been significant in our early experiments, we suggest future efforts based on our project work may focus on training the same models on a much larger dataset over a larger number of epochs. Besides, due to the time limit of this course project, we did not manage to implement another modern approach to generate high-quality images, that is denoising diffusion probabilistic model (DDPM). Should a DDPM be implemented, it could achieve better performance under the same conditions.

***Access PyTorch Implementations**

We have created a public directory via Google Drive to share the code we use to implement the GAN variants. The reader may click the following italicized text to access our shared files: *Anime Character Generation For Virtual Livestreams*.

References

[1] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *Communications of the ACM*, 63(11):139–144, 2020.

[2] Emily L Denton, Soumith Chintala, Rob Fergus, et al. Deep generative image models using a laplacian pyramid of adversarial networks. *Advances in neural information processing systems*, 28, 2015.

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

[4] Jun-Yan Zhu, Philipp Krähenbühl, Eli Shechtman, and Alexei A Efros. Generative visual manipulation on the natural image manifold. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part V 14*, pages 597–613. Springer, 2016.

[5] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. *Advances in neural information processing systems*, 29, 2016.

[6] Michael F Mathieu, Junbo Jake Zhao, Junbo Zhao, Aditya Ramesh, Pablo Sprechmann, and Yann LeCun. Disentangling factors of variation in deep representation using adversarial training. *Advances in neural information processing systems*, 29, 2016.

[7] Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014.

[8] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *International conference on machine learning*, pages 214–223. PMLR, 2017.

[9] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. *CoRR*, abs/1812.04948, 2018.

[10] Tero Karras, Miika Aittala, Janne Hellsten, Samuli Laine, Jaakko Lehtinen, and Timo Aila. Training generative adversarial networks with limited data. *CoRR*, abs/2006.06676, 2020.

[11] Tero Karras, Miika Aittala, Samuli Laine, Erik Härkönen, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Alias-free generative adversarial networks. *Advances in Neural Information Processing Systems*, 34:852–863, 2021.

[12] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. *CoRR*, abs/1703.10593, 2017.

[13] Yunjey Choi, Min-Je Choi, Munyoung Kim, Jung-Woo Ha, Sunghun Kim, and Jaegul Choo. Stargan: Unified generative adversarial networks for multi-domain image-to-image translation. *CoRR*, abs/1711.09020, 2017.

[14] Youssef Alami Mejjati, Christian Richardt, James Tompkin, Darren Cosker, and Kwang In Kim. Unsupervised attention-guided image-to-image translation. *Advances in neural information processing systems*, 31, 2018.

[15] Casey Chu, Andrey Zhmoginov, and Mark Sandler. Cyclegan, a master of steganography. *CoRR*, abs/1712.02950, 2017.

[16] Huan Fu, Mingming Gong, Chaohui Wang, Kayhan Batmanghelich, Kun Zhang, and Dacheng Tao. Geometry-consistent generative adversarial networks for one-sided unsupervised domain mapping. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2427–2436, 2019.

[17] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of gans for improved quality, stability, and variation. *arXiv preprint arXiv:1710.10196*, 2017.

[18] Mattya. Chainer implementation of deep convolutional generative adversarial network. `https://github.com/mattya/chainer-DCGAN`, 2015.

[19] Mattya. Making illustration in computer with chainer. `https://qiita.com/rezoolab/items/5cc96b6d31153e0c86bc`, 2015.

[20] Hiroshiba. Girl friend factory. `http://qiita.com/Hiroshiba/items/d5749d8896613e6f0b48`, 2016.

[21] 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.