Proceedings of the 4th International Conference on Computing Innovation and Applied Physics
DOI: 10.54254/2753-8818/87/2025.20348

Face Image Generation for Anime Characters Based on Generative Adversarial Network

Zihao Ning[1,a,*]

[1]*Mathematical Studies, University of Waterloo, Waterloo, Canada*
a. z7ning@uwaterloo.ca
**corresponding author*

Abstract: With the increasing demand for digital art, animation, and games, facial generation for anime characters has attracted growing research interest in recent years, which aims to build models to automatically generate unique and high-quality character images. Thanks to the rapid advancement of deep learning techniques, particularly generative adversarial networks, GAN-based image generation methods have continuously achieved breakthroughs in generation effectiveness and speed. Focusing on generating realistic anime face images, this paper proposes an anime character face image generation model based on GANs, which integrates Bath Normalization and Dropout to maintain strong stability and avoid overfitting. Comprehensive experiments show the efficacy of the proposed method, which can achieve high diversity in facial features and styles while maintaining visual coherence.

Keywords: Image generation, Anime character face generation, GANs.

1. Introduction

Anime character face generation is a specialized task in computer vision to create unique, diverse and high-quality by implementing generative models. With the increasing demand for digital art, animation and gaming, the ability to generate anime characters automatically becomes more essential. Although human artists can create more complex and imaginative designs, it is always time-consuming and labor-intensive, which cannot meet large-scale applications. To this end, the automation for this process still has shown a promising future and attracted more and more attention from academics and industry.

Traditional image generation methods, such as attribute-based representations (Farhadi et al., 2009) [1] rely on predefined characteristics of the objects being generated, which usually require domain-specific knowledge and are not easily adaptable to broader categories. In 2014, Goodfellow et al. (2014) [2] introduced Generative Adversarial Networks (GANs), which shows the ability to utilize random noise and other inputs for realistic image generation. Recently, the achievements in GANs such as human faces, natural faces even for anime character faces shown impressive results. Compared with traditional method, GANs provide a more flexible and scalable approach, allowing for the generation of diverse image outputs without the need for manual feature engineering. In addition, the model can learn complex visual patterns directly from the data by leveraging deep learning, further enhancing its ability to generate high-quality and unique anime faces. However, how to capture the detailed and exaggerated features to ensure the consistency and diversity remains an open issue in GAN-based image generation, especially for the anime character face generation task.

DOI: 10.54254/2753-8818/87/2025.20348

To alleviate this challenge, this paper focuses on developing an anime character face image generation model based on GANs, which integrates Bath Normalization [3] and Dropout [4] to maintain strong stability and avoid overfitting. To investigate the efficacy of the suggested approach, extensive experiments are implemented on the carefully selected anime face datasets, which can achieve a high degree of diversity in character faces design and style after suitable training epochs. The primary contribution of this study is the design and implementation of a simple and effective GAN architecture tailored for anime character face generation. The model achieves high diversity in facial features and styles while maintaining visual coherence, making it a valuable resource for various applications in digital art.

2. Method

The complete framework of the proposed face image generation method is illustrated in Figure 1, which encompasses three key steps: dataset collection, data processing, model designing and training. The subsequent subsections will offer a more in-depth introduction to each module.

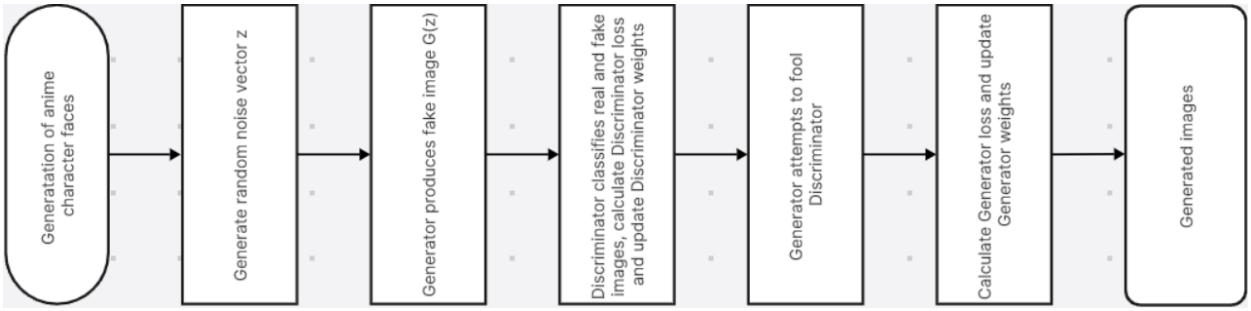
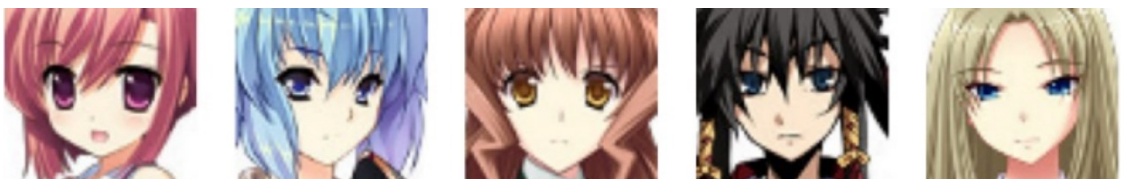


Figure 1: The framework of the proposed image generation method

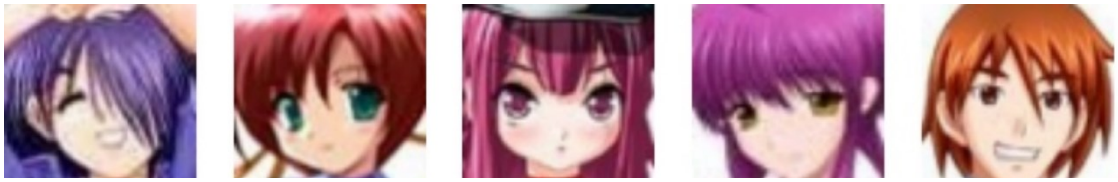
2.1. Dataset

In this project, 3 datasets are adopted for model training and test. The first one is the Anime Faces [5], which includes 20 thousand images scraped from an anime website [6] and each scraped images will be resized to 64 × 64 pixels [7] as the input. Additionally, there are some bad cropping and non-human faces in this dataset, it would be the challenge for the training process. The second one is the Anime Faces dataset [8] containing more than 60 thousand of anime character faces in different styles and with various features. Properly preparing this data ensures that it can be efficiently fed into the GAN model for training. One of the strengths of the Anime Faces Dataset is its diversity in terms of the facial characteristics of the anime characters. This diversity leads the GAN learns a wide variety of features, leading to more generalized and versatile models. Some of the key areas of diversity such as hairstyle and eye shape. The third dataset is the Anime GAN Lite [9], which consists of 25 thousand images generated by Style GAN-2 in a dimensional of 512 × 512 pixels. Since the images are generated by a GAN model, the diversity of this dataset might not be as rich as real images. We visualize the images from different datasets, as shown in Figure 2.



(a) Anime Faces

DOI: 10.54254/2753-8818/87/2025.20348



(b) Anime Faces Dataset

(c) Anime GAN Lite

Figure 2: The sample images from 3 different dataset

2.2. Data processing

Although the Anime Faces Dataset is well-created, some preprocessing steps were significant to assure data is appropriate for GAN training. The quality of the generated images is directly influenced by the quality of the input data, so any irregularities should be removed carefully. For better training this mode, some functions in PyTorch [10] are used, such as PyTorch dataset loader.

The initial step in preprocessing involves resizing all images to a standard resolution of 64 × 64 pixels. The changed resolution allows the generator and discriminator networks to process the images uniformly. Image resizing makes sure that each input image conforms to the same dimensionality, which is necessary for both computational efficiency and consistency in output. After resizing, the images' pixel values were normalized to the interval of [-1, 1]. Since the output of the generator utilizes a Tanh activation function that produces values within the same range, normalization is essential to match the input distribution with the model's output normalization is crucial for aligning the input distribution with the model's output. This normalization step helps improve the convergence speed during training by making the data more suitable for the GAN architecture.

To efficiently load and process images during training, a custom PyTorch dataset loader was created. This loader not only reads the images from the dataset directory but also applies transformations (resizing and normalization) on-the-fly as images are loaded into memory in batches.

2.3. Model Design

The GAN architecture is composed of two main components, the generator and the discriminator. Both models are equipped with convolutional layers to efficiently capture the features required for producing and assessing high-quality anime faces.

2.3.1. Generator

The generator starts with a latent vector of size 128, sampled from a Gaussian distribution. This latent space acts as the "seed" within the generator to create anime face images. The randomness in the latent space ensures variety in the produced images. To unsampled the latent vector into an image, the generator employs a sequence of transposed convolutional layers (also referred to as deconvolutions). The spatial dimensions of the input are gradually increased by these layers, while the depth is reduced, eventually outputting a 64 × 64 × 3 image (three channels representing RGB

DOI: 10.54254/2753-8818/87/2025.20348

values). Batch normalization and a ReLU activation function follow each transposed convolutional layer to promote stable training and prevent problems like mode collapse. A Tanh activation function is utilized in the final layer, producing pixel values within the range of [-1, 1]. This aligns with the normalized input data and guarantees that the generated images exhibit realistic color distributions.

2.3.2. Discriminator

The discriminator aims to distinguish real and generated (fake) anime faces. It could be seen as a binary classifier, outputting the probability of whether the input image is real. The discriminator is also a deep convolutional network, but it is opposite to generator, as it progressively downs sample the input image through convolutional layers. The discriminator takes input which is 64×64 RGB images and passes them through a series of convolutional layers, each of the convolutional layers reduces the spatial dimensions and increases the depth. This process helps the discriminator to extract complex features from the image, which are essential for determining its reality. Unlike the generator, the discriminator uses a LeakyReLU activation function after each convolutional layer. This helps avoid the issue of dying ReLUs, and the gradient flow could be maintained even if the activations are close to zero. The final layer of the discriminator uses a sigmoid activation function to convert the output into a probability value ranging from 0 to 1, signifying whether the input image is real or fake. A value close to 1 means the image is likely real, while a value near 0 suggests it is fake. This binary classification is essential for the discriminator to discern between genuine images from the training data and the fake images created by the generator, thus aiding the training process of the generative adversarial network (GAN).

2.4. Loss function

Training a GAN involves a delicate balance between the generator and the discriminator. If one model becomes too dominant compared with the another, the training process can become unstable. To maintain this balance, the generator and the discriminator should be trained with alternating updates. The loss function utilized in GANs is binary cross-entropy (BCE) loss, which is widely employed for binary classification tasks. The goal of the generator is to maximize the discriminator’s likelihood of classifying generated images as real, then the discriminator aims to minimize this likelihood for generated images while maximizing it for real images. The discriminator aims to maximize the likelihood of correctly labeling real and fake images. Therefore, the loss is computed as the sum of the binary cross-entropy losses for the classifications of real and fake images. The generator loss is calculated by using binary cross-entropy on the discriminator's output, like evaluating how well a forger has imitated a painting based on the critic's judgment., but the labels are reversed, this means that the generator tries to let the discriminator to recognize fake images as real. The goal of the generator is to curtail this loss, and proficiently improve its capability to generate realistic images.

2.5. Training details

Adam [11] was used to optimize both the generator and the discriminator, with initial learning rates and betas set to stabilize the training process. The Adam optimizer could adjust the learning rate for each parameter and make it suitable for handling the complexity of the GAN's dynamic training. The learning rate was carefully tuned to balance stability and speed of convergence. During the training process, the loss values for both the generator and discriminator were monitored after each epoch. Additionally, generated images were saved periodically to visually assess the progress of the GAN. This allowed for early detection of problems like mode collapse or imbalance between the generator and discriminator.

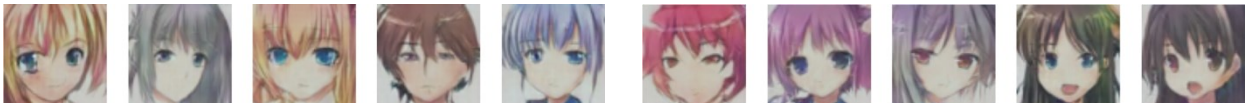
DOI: 10.54254/2753-8818/87/2025.20348

3. Experiment

3.1. Qualitatively Image Generation

Two neural networks, the Generator and the Discriminator, make up a GAN. These models are trained in opposition, with the Generator aiming to produce convincing images, the discriminator, on the other hand, functions to distinguish between real images and those that are generated (fake). The image size is set as 64 × 64 pixels for comparison with each dataset, and the latent size is 128 for balancing diversity and complexity. The batch size is 128 to ensure the data can be enough trained for every epoch, and there is a total of 100 epochs to avoid less training and monitoring when overfitting. We visualize the generation results in Figure 3 on three datasets, where some images with similar good or bad features are selected. The images from the first dataset look harsher, and the part of distortion more clearly, especially on the eyes, such as the size and color of the eyes. Anime Faces Dataset generates softer and clearer, there is less distortion happening, but there are still some unbalances on the face. The last dataset output is more perfect, as the input images are resized, so the images look not very clear, there are still some disadvantages such as extra hair and generated images that look similar.

(a) Anime Faces



(b) Anime Faces Dataset

(c) Anime GAN Lite

Figure 3: Visualization of the proposed method on three different datasets

3.2. Analysis for Real and Fake Score

The Real-Fake Score in GANs refers to a metric that indicates how well the generator is performing in producing realistic samples by comparing them with the real data. It is derived from the output of the discriminator, which is assigned the role of distinguishing between real data samples and the fake samples generated by the generator. Those three graphs show the real score keeps stability increasing especially on the few dozen epochs, the following data like a wave. Thus, it could be concluded that after the first few dozen epochs, the generated images would be performed well as closer to real images. A hyperparameter known as the learning rate defines the degree of adjustments applied to the model based on the estimated error during each update of the model's weights.

DOI: 10.54254/2753-8818/87/2025.20348

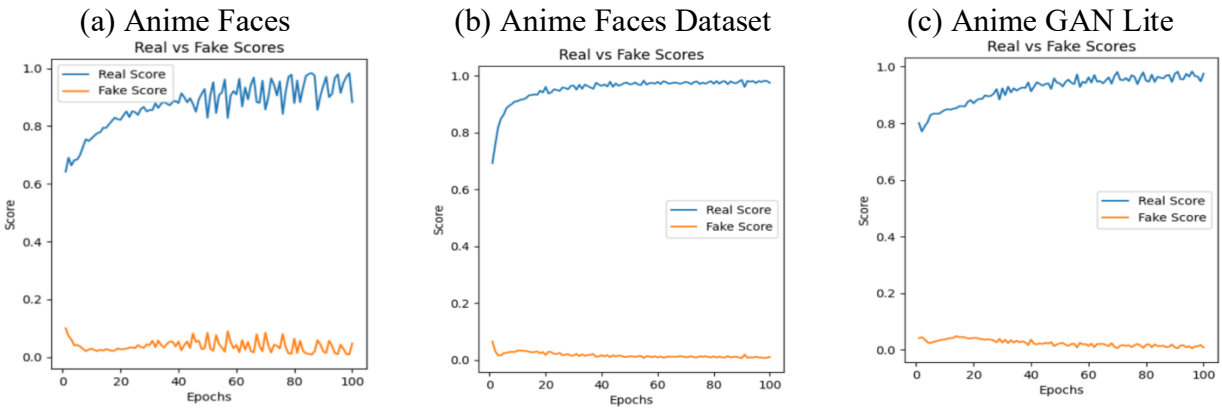


Figure 4: The real-fake score over epochs

3.3. Effectiveness of Learning Rate

The learning rate is a key setting that controls how much the model's weights change based on the error during updates. A higher learning rate makes training faster but can cause mistakes, while a lower rate allows for careful adjustments but slows down training. It's important to pick the right learning rate for good results. For controlling the variables, only the Anime Faces Dataset is chosen, and 5 different learning rates are compared, including 0.0001, 0.0005, 0.001, 0.003, and 0.005. As shown in Figure 5, large or small learning rate cause unstable loss curve, especially for the large results surge loss curve.

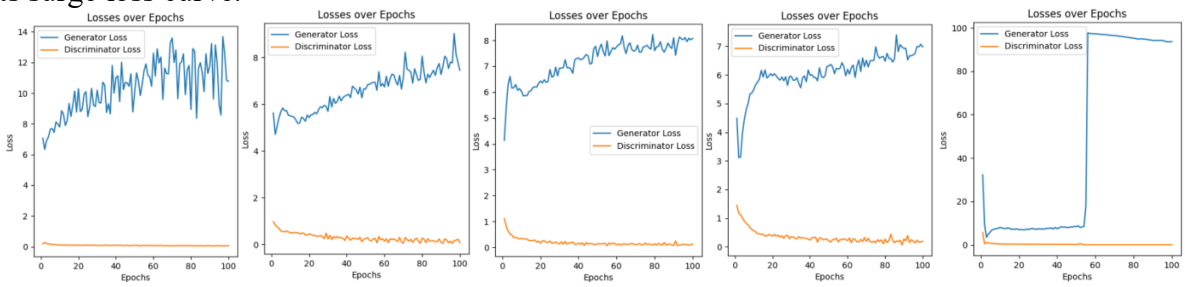


Figure 5: Loss curve based on different learning rate

4. Conclusion

This paper proposes a simple but effective anime face image generation model built upon the generative adversarial network, which has been verified on different datasets. Although some of the generated images showed some distortion and were unclear, the proposed model significantly reduced the training time. As there are some non-human and bad images in Anime Face, the quality of generated images is inferior to the Anime Faces Dataset and Anime GAN Lite. As there are some non-human and bad images in Anime Face, the quality of generated images not good as the Anime Faces Dataset and Anime GAN Lite. In addition, the style of the original images causes the quality of generated images, by comparing with Anime Faces Dataset and Anime GAN Lite, as Anime GAN Lite's image from another GAN model, so the style of images is similar, the second time to use GAN would output similar images. However, as the Anime Faces Dataset is created by humans, the details of images such as eye and facial expressions are different, after training by the GAN model, some bad images would be generated.

DOI: 10.54254/2753-8818/87/2025.20348

References

[1] *Farhadi, A., Endres, I., Hoiem, D., & Forsyth, D. (2009). Describing objects by their attributes. In 2009 IEEE Conference on Computer Vision and Pattern Recognition (pp. 1778-1785). IEEE. DOI: 10.1109/CVPR.2009.5206772*

[2] *Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., & Bengio, Y. (2014, June 10). Generative Adversarial Networks. arXiv.org. https://arxiv.org/abs/1406.2661*

[3] *Ioffe, S., & Szegedy, C. (2015, March 2). Batch normalization: Accelerating deep network training by reducing internal covariate shift. arXiv.org. https://arxiv.org/abs/1502.03167*

[4] *Liu, Z., Xu, Z., Jin, J., Shen, Z., & Darrell, T. (2023, May 31). Dropout reduces underfitting. arXiv.org. https://arxiv.org/abs/2303.01500*

[5] *Rakshit, S. (2019b, May 16). Anime faces. Kaggle. https://www.kaggle.com/datasets/soumikrakshit/anime-faces*

[6] *Getchu (n.d.). https://www.getchu.com/*

[7] *Nagadomi. (n.d.). lbpcascade_animeface. GitHub. https://github.com/nagadomi/lbpcascade_animeface*

[8] *Churchill, S. (2019a, October 13). Anime face dataset. Kaggle. https://www.kaggle.com/datasets/splcher/animefacedataset*

[9] *Kottarathil, P. (2021, January 15). Anime Gan Lite. Kaggle. https://www.kaggle.com/datasets/prasoonkottarathil/gananime-lite*

[10] *Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., & Chintala, S. (2019). PyTorch: An imperative style, high-performance deep learning library. Advances in Neural Information Processing Systems, 32, 8024–8035.*

[11] *Kingma, D. P., & Ba, J. (2017, January 30). Adam: A method for stochastic optimization. arXiv.org. https://arxiv.org/abs/1412.6980*