Deep Convolutional Generative Adversarial Networks (DCGAN)-Based Anime Face Generation

Xunxiong Ou

Aberdeen Institute of Data Science and Artificial Intelligence, South China Normal University, Guangdong, 528225, China
202138030043@m.scnu.edu.cn

Abstract. This study delves into the realm of anime face generation with the aim of empowering individuals to create their own anime characters and easing the burden on artists. Employing Deep Convolutional Generative Adversarial Networks (DCGAN), the research focuses on generating anime face images. The DCGAN model consists of a generator and a discriminator, each designed and trained for their respective roles. The generator employs a convolutional transpose structure, while the discriminator utilizes a convolutional neural network structure. Through simultaneous training of the generator and discriminator using a diverse dataset of anime face images, a comprehensive DCGAN model is developed. Leveraging the Kaggle dataset, the study evaluates the training progress of the model through loss change curves of the generator and discriminator, alongside the final generated anime face images. Comparing different loss change curves and generated images across varying epochs and batch sizes reveals superior performance with 60 epochs compared to 30 epochs, facilitating clearer facial features. Moreover, a batch size of 32 outperforms 256, attributed to its more stable loss change curve. These findings contribute valuable insights to the domain of anime face generation research.

Keywords: Anime Face Generation, Deep Convolutional Generative Adversarial Networks (DCGAN), Training progress.

1 Introduction

As technology evolves, deep learning is being applied in various industries. One area where it's making an impact is in anime face generation, which helps automate character creation in the anime industry. In today's world, many people love anime characters and even use it as avatars. However, creating new characters can be tough due to the difficulty of drawing skills [1]. Plus, artists often find the process of creating characters time-consuming [2]. This has led to a big demand for anime faces, which can be met with the technology of anime face generation. By providing users with new anime faces, this technology is driving the entertainment industry forward.

Applications for Generative Adversarial Networks (GANs) include image synthesis, super-resolution, image-to-image translation, classification and regression, and

Y. Wang (ed.), *Proceedings of the 2024 2nd International Conference on Image, Algorithms and Artificial Intelligence (ICIAAI 2024)*, Advances in Computer Science Research 115,
https://doi.org/10.2991/978-94-6463-540-9_86

more [3]. Using self-attention modules within a GAN architecture and a multiscale patch discriminator learning technique, Chandaliya and Nain provide a kid's face age-progression and regression system that produces photo-realistic face images with intact identity [4]. This leads to notable improvements in face recognition, ranking, and age estimation, both qualitatively and quantitatively. Islam and Zhang proposes an approach utilizing GAN to generate synthetic medical images, specifically brain Positron Emission Tomography (PET) images representing three different stages of Alzheimer's disease to address the challenge of limited annotated datasets for medical image analysis [5]. In order to achieve considerable gains over previous approaches, Bulat et al. offer a two-stage process for image and face super-resolution [6]. First, a High-to-Low GAN is used to imitate real-world picture degradation and downsampling, and then a Low-to-High GAN is used for super-resolution. In addition to the practical applications of GAN in various fields, optimization of GAN continues to be pursued. Bao et al. Proposes recurrent regression dual discriminator generative adversarial network (R2D2GAN) [7]. Its goal is to learn from previous stochastic processes to identify patterns for multiple agents, thereby generating spatiotemporal data for these agents. This technology uses classification and regression discriminators to depict diverse data features, which helps stabilize the training process of generative adversarial networks.

This study's main objective is to use Deep Convolutional Generative Adversarial Networks (DCGAN) to generate unique anime face images while looking at several aspects that affect model training. Firstly, an extensive dataset comprising high-quality anime face images showcasing diverse facial expressions, hairstyles, and accessories is employed to ensure the model captures the intricate features and variations inherent in anime faces. Secondly, image augmentation techniques are applied to bolster model robustness, mitigate overfitting risks, and enhance overall performance. Thirdly, this study trains both the generator and discriminator: the former utilizes a convolutional transpose structure to convert low-dimensional random noise vectors into high-dimensional image data, thus generating convincing fake anime face images, while the latter employs a convolutional neural network structure to discern between real and fake images. More realistic final images are produced as a result of constant improvement in the discriminator's ability to make judgment calls and the generator's ability to produce realistic images throughout training. Post-training, the batch of images generated allows for visual assessment of their resemblance to real anime face images. Furthermore, the training results are evaluated by plotting changes in the loss values of the generator and discriminator. Experimental findings indicate that the generated anime face images exhibit commendable performance. The study highlights the significance of the generator and discriminator structures, underscoring their impact on training outcomes. Additionally, the number of epochs proves crucial, with optimal performance achieved at a specific epoch count.

2 Methodology

2.1 Dataset Description and Preprocessing

This study's dataset was obtained from Kaggle [8]. It comprises 63,632 anime face images, commonly employed as training data for anime face generation. Preprocessing of the images is conducted as follows. Initially, image data is loaded and converted into NumPy arrays for visualization purposes. Subsequently, image augmentation is performed in two steps. These involve scaling the pixel values of the images to the range [0, 1] and horizontally flipping it. These procedures facilitate the improved adaptation of images for training neural network models, reducing the likelihood of overfitting and bolstering the model's capacity for generalization. Finally, the images are batch processed, resizing the loaded images to 64×64 pixels, with each batch set to load 32 images. This research also involves transforming the pixel values of the images generated by the generator, converting the generated images into a format suitable for display.

2.2 Proposed Approach

The objective of this study is to train a DCGAN model using anime face images from the training dataset and utilize it to generate new anime face images. The procedure of this experiment is depicted in the flowchart. Firstly, images of size 256×256×3 from the training set are enhanced to 64×64×3 images. A generator and a discriminator are introduced in this research. The generator takes noise of dimension 300 and generates fake images of size 64×64×3 through a convolutional transpose structure. On the other hand, the discriminator receives 64x64x3 images, both real and fraudulent, using a convolutional neural network structure. It then produces a value between 0 and 1, which indicates how likely it is that the input image is real. The next step involves training the DCGAN, where the training process involves computing the losses of the generator and discriminator respectively, updating the parameters of the discriminator and generator models to minimize their respective losses. Finally, after training for a certain number of epochs, the generator is capable of producing a batch of anime face images that are convincing enough to be mistaken for real ones. The pipeline is shown in the Fig. 1.

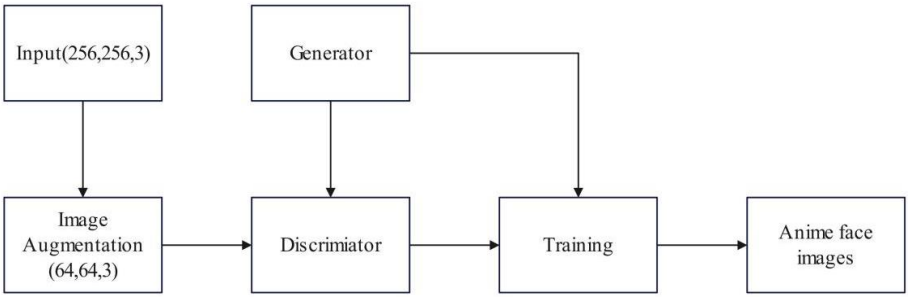

Fig. 1. The pipeline of the model.

DCGAN. This research incorporates convolutional operations into the traditional GAN model to form the DCGAN model, which aids in learning advanced features and spatial structures of images, thereby generating more realistic images [9]. Moreover, DCGAN demonstrates good adaptability to complex image distributions [10]. Like traditional GAN models, DCGAN also requires defining the generator and discriminator. Below is a detailed description of the structure of the DCGAN model used in this research.

Firstly, the author introduces the structure of the generator. The first step involves an input layer that receives one-dimensional random noise with a dimension of 300. Secondly, a dense layer is set up in this research to transform the one-dimensional random noise into a one-dimensional vector of length 32768. Thirdly, the model's non-linear expressiveness is improved by using the Rectified Linear Unit (ReLU) activation function. Fourthly, a reshape layer is set up to reshape the output of the dense layer into a three-dimensional tensor of size (8, 8, 512), preparing it for subsequent convolution operations. Fifthly, three transpose convolution layers are set up, with filter sizes set to (4, 4), strides set to (2, 2), maintaining the same feature map size between input and output, and activated by ReLU activation function. These three transpose convolution layers progressively reduce the output channels, changing from 512 to 256 to 128 to 64. These three transpose convolution layers gradually enlarge the feature maps, aiming to generate high-resolution images. Finally, an output layer is set up to receive the feature maps outputted from the previous transpose convolution layer and outputs Red, Green and Blue (RGB) images through a filter size of (4, 4), with the sigmoid function still being used to constrain the output values between 0 and 1 to match the pixel value range of real images. The pipeline is shown in the Fig. 2.

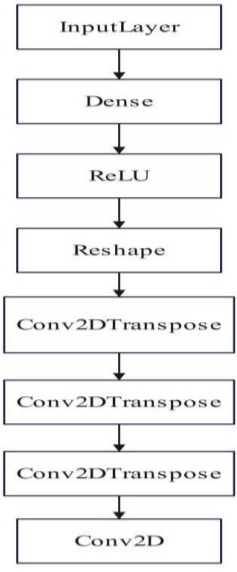

Fig. 2. The pipeline of the generator.

Next, the structure of the discriminator is introduced. Firstly, an input layer is set up to receive images with a size of (64, 64, 3). Secondly, a convolutional layer is set up, with the first convolutional layer's kernel size set to (3, 3), containing 64 convolutional kernels, and the stride defaulting to (1, 1). This layer uses the Leaky Rectified Linear Unit (LeakyReLU) activation function to activate the convolutional results, introducing nonlinearity and enhancing the model's expressiveness. The convolutional layer extracts feature from the input image through this operation. Thirdly, a pooling layer (MaxPooling2D) is set up, where the pooling kernel size is (2, 2), and the stride defaults to (1, 1). By sliding the pooling kernel on the feature map and calculating the maximum value within each window, this layer reduces the feature map size by half. Fourthly, following the above method, convolutional and pooling layers are repeated twice to further extract and compress the image's features. The change in convolutional kernel for each layer is 64 to 128 to 256. Fifthly, to receive the output from the last pooling layer and convert it into a one-dimensional vector for connecting with the next fully connected layer, a flatten layer is created. In this study, a fully connected layer is built to feed a flattened one-dimensional vector into 256 neurons. Finally, a completely connected layer is configured to produce a value between 0 and 1 based on the output of the preceding fully connected layer, which represents the probability of assessing the image as real. The pipeline is shown in the Fig. 3.

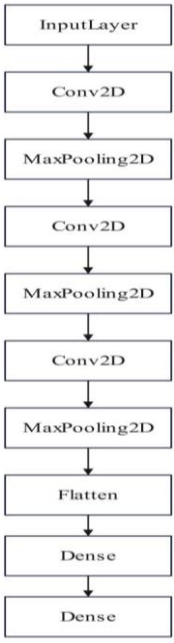

Fig. 3. The pipeline of the discriminator.

After establishing the generator and discriminator, this research simultaneously trains to ultimately generate a high-quality DCGAN model.

Loss Function. Selecting a suitable loss function is essential to this study. In this study, the binary cross-entropy loss function was chosen. This is due to the fact that the discriminator in a GAN must be able to differentiate between generated and real data, which is a task that comes under binary classification. For these kinds of binary classification jobs, a good match is found with the binary cross-entropy loss function.

$$L(q, \hat{q}) = -\frac{1}{k}\sum_{i=1}^{K}[q_i \, log(\hat{q}_i) + (1 - q_i) \, log(1 - \hat{q}_i)] \quad (1)$$

The above expression represents the binary cross-entropy loss function. Here, q represents the image's actual label, $\hat{q}$ is the image's expected label as determined by the discriminator, K denotes the total sample size, The i-th image's true label is denoted by q_i, and the discriminator's predicted label for the same image is represented by $\hat{q}_i$. In this research, the labels for real images are set to 1, and the labels for fake images are set to 0. These labels are then passed to the loss function along with the discriminator's predicted values.

2.3 Implementation Details

This research is conducted on Kaggle notebook and utilizes the Graphics Processing Unit (GPU) P100 available on Kaggle for training. Python is the programming language used in this study. In the hyperparameters section, the generator has an initial learning rate of 0.0003, while the discriminator's initial learning rate is 0.0001. The exponential decay rate is 0.5. The optimizer selected is Adam, as it converges to local optima faster compared to traditional models and exhibits better stability. The model is trained for 30 epochs with a batch size of 32.

3 Results and Discussion

This section focuses on analyzing the impact of changes in epoch and batch size on model training. Firstly, this research alters the epoch and evaluates the results by analyzing the changes in loss values. Secondly, changes are made to the batch size, and the results are also assessed by analyzing the changes in loss values.

In this research, the epoch is initially set to 60, and the batch size to 32, resulting in the generation of Fig. 4, illustrating the changes in the generator and discriminator losses. It is evident from the graph that the loss curve of the generator is not ideal, showing an overall upward trend. Additionally, it can be observed from the graph that the generator's loss experiences greater fluctuations when the epoch exceeds 30. Therefore, this research decides to maintain the batch size unchanged and set the epoch to 30. A comparison is made between 18 generated anime faces from two experimental groups (Fig. 5 and Fig. 6). It is found that the images generated by the experiment with an epoch of 60 exhibit better performance compared to those with an epoch of 30, featuring clearer facial features on the anime faces. This research attributes this to the generator requiring more epochs to learn how to generate clearer facial features.

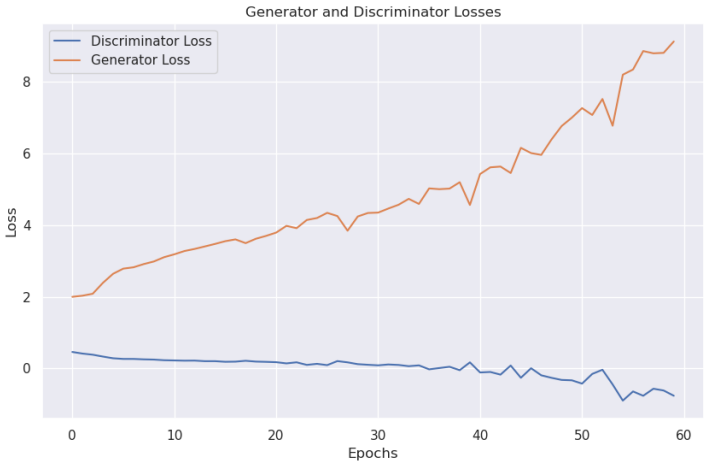

Fig. 4. Loss curve of epoch60 batchsize32 .

Fig. 5. Outcome of epoch60 batchsize32.

Fig. 6. Outcome of epoch30 batchsize32.

Given that altering the experiment's epoch does not lead to improved image performance, this research decides to change the batch size. This research, while maintaining the epoch at 60, increases the batch size. The batch size is increased to 256, resulting in the loss function curve as shown in Fig. 7. In contrast, the generator's loss experiences a discernible drop followed by a rise when the batch size is increased to 256. This research compares some images generated under these two settings (Fig. 5 and Fig. 8) and finds that the images generated with a batch size of 32 have clearer facial features, while those generated with a batch size of 256 exhibit issues such as distorted or unclear features in some images. This research believes that the reason for this phenomenon is the significant fluctuation in the generator loss when the batch size is 256, indicating the instability of the generator's loss function. In contrast, when the batch size is 32, even though the generator's loss function shows an upward trend, it is more stable, thus producing higher-quality images.

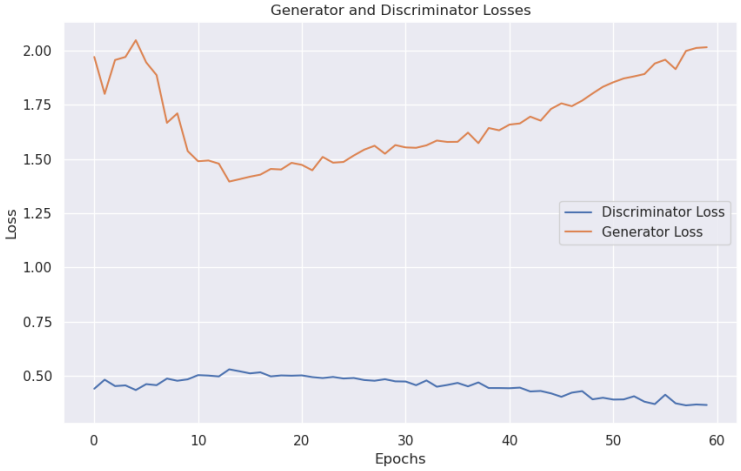

Fig. 7. Loss curve of epoch60 batchsize256.

Fig. 8. Outcome of epoch60 batchsize256.

4 Conclusion

This study's primary objective is to use the DCGAN model to produce unique anime face images. Through the establishment of a generator and discriminator within the DCGAN framework, the model undergoes comprehensive training to achieve this goal. Results from extensive experiments conducted to evaluate the proposed method reveal significant findings. Notably, maintaining a batch size of 32 and training for 60 epochs yields superior image quality compared to models trained for 30 epochs, indicating that extended training duration facilitates the learning process for generating clearer facial features. Additionally, results demonstrate that when maintaining a 60-epoch training regimen, images generated with a batch size of 32 outperform those generated with a batch size of 256, attributed to the more stable loss curve associated with the smaller batch size. Moving forward, the study aims to explore the integration of Super-Resolution Generative Adversarial Networks (SRGAN) to enhance the spatial resolution of generated images. This enhancement strategy aims to augment the details and clarity of the anime face images, ultimately yielding images of higher resolution and finer detail.

References

1. Fang, Z., Jin, Y., Li, M., Tian, Y., Zhang, J., Zhu, H.: Towards the automatic anime characters creation with generative adversarial networks. arXiv preprint :1708.05509 (2017).
2. Anjana, M. S., & Dhanya, N. M.: Anime Face Generation using Generative Adversial Networks in Deep Learning. Mathematical Statistician and Engineering Applications, 71(3s), 335-342 (2022).
3. Arulkumaran, K., Bharath, A.A., Creswell, A., Dumoulin, V., White, T., Sengupta, B.: Generative adversarial networks: An overview. IEEE signal processing magazine, 35(1), 53-65 (2018).
4. Chandaliya, P. K., & Nain, N.: Child face age progression and regression using self-attention multi-scale patch gan. In 2021 IEEE International joint conference on biometrics (IJCB), pp. 1-8. IEEE (2021).
5. Islam, J., Zhang, Y.: GAN-based synthetic brain PET image generation. Brain informatics, 7(1), 3 (2020).
6. Bulat, A., Tzimiropoulos, G., Yang, J.: To learn image super-resolution, use a gan to learn how to do image degradation first. In Proceedings of the European conference on computer vision (ECCV), pp. 185-200, (2018).
7. Bao, P., Chen, Z., Dai, D., Wang, J.: Multiple agents' spatiotemporal data generation based on recurrent regression dual discriminator GAN. Neurocomputing, 468, 370-383 (2022).
8. Kaggle dataset. https://www.kaggle.com/datasets/splcher/animefacedataset, last accessed 2023/8/10.
9. Liu, Y., Wang, Z., Zhang, J., Zhao, T.: Reconstruction of the meso-scale concrete model using a deep convolutional generative adversarial network (DCGAN). Construction and Building Materials, 370, 130704 (2023).
10. Fan, X., Liu, B., Luo, J., Lv, J., & Zou, T.: Application of an improved dcgan for image generation. Mobile Information Systems 2022, 14 (2022).