Exploration and Optimization of Deep Convolutional Generative Adversarial Networks

Pragnya Upendra Pathak
University of California, San Diego
A69027339

Krish Mehta
University of California, San Diego
A59026465

Abstract

This project aims to generate anime character images. We start with implementing a baseline DCGAN (Deep Convolutional Generative Adversarial Networks) adapted from Radford et al. [1]. The generator and discriminator architectures of DCGAN are trained on a custom dataset consisting of around 60,000 anime character images. We identify areas of improvement for the model and optimize the architecture to obtain diverse and rich image collections by implementing Spectral Normalization inspired from Miyato et al. [2] on top of the vanilla DCGAN.

1 Introduction

In this project, our primary aim is to explore the generative capabilities of DCGANs and utilize them to generate high-quality anime character images by training a DCGAN model on a repository of images consisting of pre-existing amine character images. The face generation problem using DCGAN has been implemented extensively for human faces. We aim to move beyond real human faces to anime character faces and explore the architecture's suitability for this unique image style.

Anime character faces are the most fundamental part of anime character design. Hence, the ability to generate synthetic, innovative, and diverse anime characters can find applications in character creation and animation for manga, anime, and anime-inspired games.

To solve the problem statement, we first implemented the original DCGAN model as demonstrated by Radford et al. [1]. This model was trained to generate images of anime faces from random noise fed as input to the system. The potential of DCGANs is evident right from the get go, as the recipe suggested originally in Radford et al. [1] is quite well performing by itself. However, it has some limitations.

The images suffered from some degree of distortion, lack of uniqueness, and low image quality. To overcome this we experimented with various architectural modifications and were able to obtain good quality and diverse images of anime characters. We also dip into some enhancements to the initial DCGAN formula, by introducing spectral normalization [2].

2 Related Work

Generative Adversarial Models (GANs) were first introduced by Goodfellow et al. [3] in "Generative Adversarial Networks". In essence they are a system of networks that can model a distribution that mimics a distribution given to them during training. Although a breakthrough, the GANs architecture used basic convolutional and fully connected layers. In "Unsupervised Representation Learning With Deep Convolutional Generative Adversarial Networks", Radford et al, [1] then modified this architecture by incorporating changes such as using strided convolutions, batch normalization, ReLU, LeakyReLU, and Tanh activations.

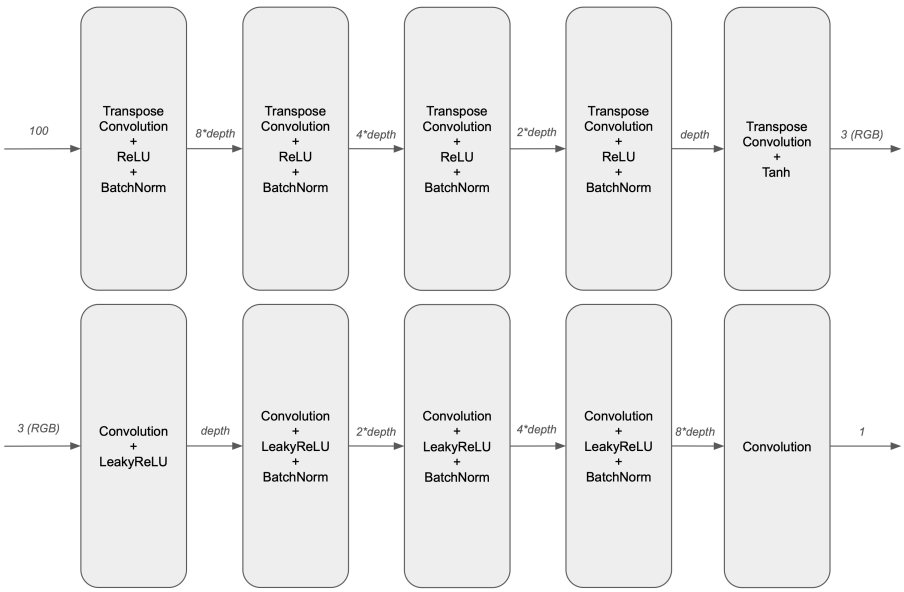

Figure 1: *Generator Network (top) and Discriminator Network (bottom) architecture. Numbers over the arrows indicate the number of channels*

DCGANs however, may suffer from mode collapse and training instability, i.e. they may over-generate one kind of image and thus ignore the diversity of the dataset. One method to overcome this was shown by Miyato et al. [2] in "Spectral Normalization for Generative Adversarial Networks". Spectral Normalization is applied to the convolution layers of the model to stabilize its performance.

3 Method

We designed the network architecture as per the principles demonstrated by Radford et al. [1]. Their work is built upon the foundation of GANs by Goodfellow et al. [3]. The adversarial framework is the backbone of this architecture.

3.1 Network Architecture

The Generator network building block is a Transpose Convolution which is passed through a non-linear ReLU activation followed by Batch Normalisation. Transpose Convolutions are known to perform well for segmentation applications but here they are used to amplify the number of channels and increase the number of feature maps so the model has enough complexity to learn various representations. Normalization helps bring all inputs to a zero mean and unit variance. This will help with the gradient vanishing problem.

The last layer is slightly different, with just a transpose convolution followed by the Tanh activation function. This bounded activation was observed to converge more quickly over the wide color gamut.

A notable feature of this architecture is that the conventional format of attaching Fully Connected layers to the output of convolution layers is dropped in favor of connecting the highest channel layers to the input and output of the generator and discriminator respectively.

3.2 Training

After having learned conventional neural networks, the training process of GANs was a new concept. The generator's goal is to learn from the training data and produce images with similar features such that the discriminator might mis-classify it. The discriminator's goal is to learn to classify the images between real and fake, in the sense of whether it is a generated image or a real image from the dataset. The equilibrium or the back-and-forth of this training loop is what leads to the generator producing such high-quality images.

Figure 2: *Some random examples of input images from the AnimeFaces dataset*

4 Experiments

Our goal with the implementation and further experiments was to recreate the image-generation capabilities of DCGANs demonstrated by Radford et al. [1]. However, we also wanted to gain a deeper understanding of the architecture so we also tweaked the architecture and observed how it affected the generated images.

- One major drawback of DCGAN is its training instability and non-convergence. To understand the implications and resolution of this problem, we looked into a new normalization technique called spectral normalization and applied it to the model's convolution layers.
- We also tried different hyperparameters to enable faster and better convergence of the model. Eventually, we found a set of hyperparameter values. Different combinations of optimizers we experimented such as [Adam, Adam], [Adam, RMSProp], [RMSProp, Adam], and [RMSProp, RMSProp] were used to train the model. The best performance was attained when Adam optimizer was used for both the generator and discriminator.
- We attempted to objectively analyze the classification properties of the discriminator network and understand the quality of images produced by the generator, so we used a set of real and fake images to train a classifier and evaluate its accuracy in classifying real and fake images. It did show classification potential but in our implementation it was not performing up to the level of dedicated specialized clustering or classification algorithms like KNN.

4.1 Dataset

In our experiments we have used the AnimeFaces dataset. This contains roughly 60,000 pre-processed images of the faces of anime characters. This is a suitable dataset because there is a wide variety of features, expressions, and colors in anime character faces, but they are also simpler than human faces and can potentially give us better results faster.

4.2 Architecture and Training

We trained the model with a learning rate of 0.0002 as suggested by Radford et al. [1]. We tried to increase it in case it helps our use case but this seemed to be the best value for stable yet reasonably fast convergence. We used β_1 of 0.5 and β_2 of 0.999 in the Adam optimizer for both generator and discriminator. For our loss function, we used a combination of a Sigmoid layer and the BCELoss.

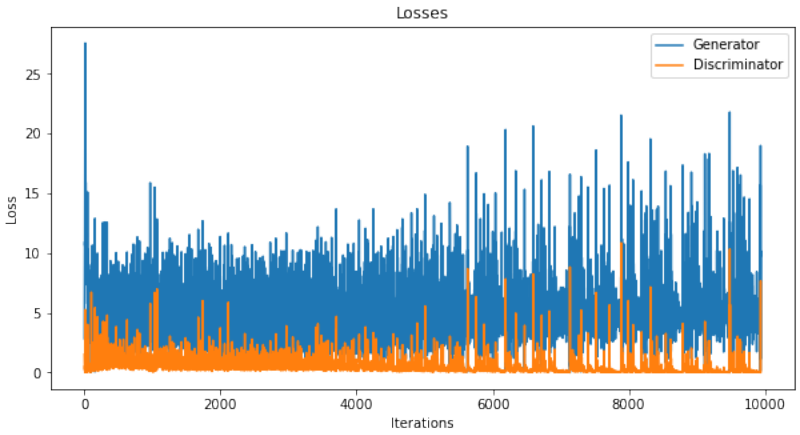

Figure 3: *Loss vs Iterations for the Generator and Discriminator network*

Figure 4: *Images generated after 1 epoch of training (Baseline DCGAN)*

4.3 Results

The DCGAN architecture by itself has a very strong performance potential. In just 1 epoch, the model learns the high level features, like the overall placement of the face, eyes and hair. Naturally though, it is quite noisy, because the generator is just beginning to learn how to create meaningful representations out of the random noise given to it.

By the time we reach 10 epochs in the baseline DCGAN, the model irons out most of the noise. However, at that stage none of the images are quite right in terms of the facial features, there are some noisy features in most of the images that the human eye can instantly spot.

Figure 5: *Images generated after 10 epochs of training (Baseline DCGAN)*

Figure 6: *Images generated after 20 epochs of training (Baseline DCGAN)*

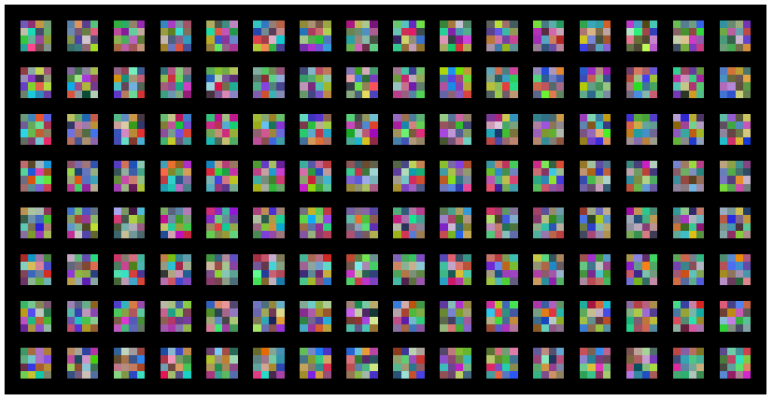

Figure 7: *Visualization of the kernels of the first discriminator layer from a baseline DCGAN trained for 20 epochs*

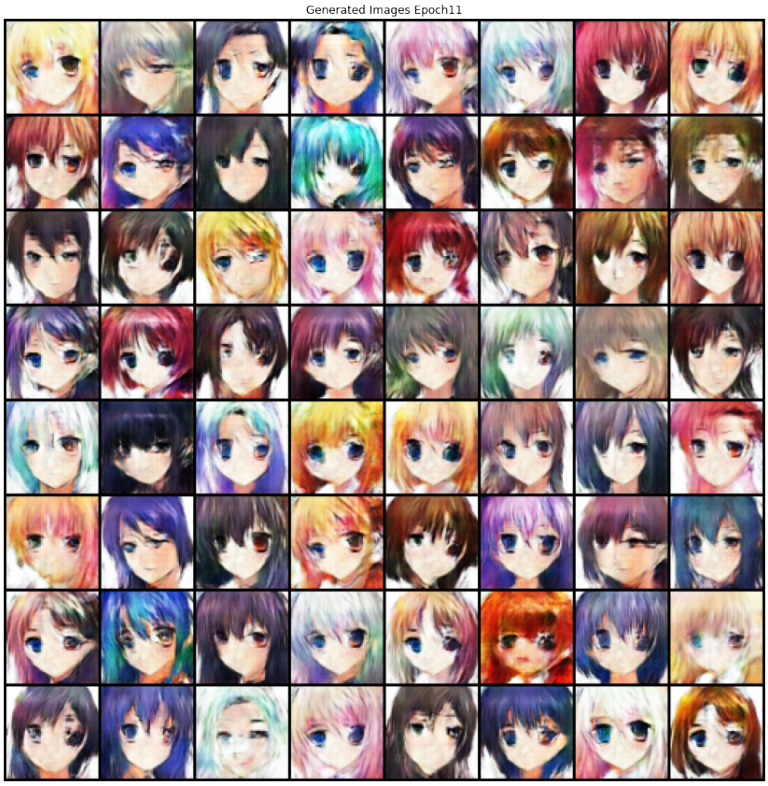

Figure 8: *Images generated after 11 epochs of training (with Spectral Normalization)*

As we reach around 20 epochs in the baseline DCGAN, the model starts to produce some images which are quite consistent and true to the training data in terms of overall facial features.

5 Supplementary Material

A video presentation (linked here) of this work is available to view for University of California, San Diego network. Contributions by team members are mentioned below.

Figure 9: *Images generated after 24 epochs of training (with Spectral Normalization). There is a certain diversity to the kind of images spectral normalized network is able to generate*

Pragnya Pathak:

- DCGAN Literature Review
- Training runs, Hyperparameter tuning, and convergence/stability testing
- Spectral Normalization
- Report Sections 1, 2, 4, 4.2

Krish Mehta:

- DCGAN Literature Review
- Dataset setup/preprocessing
- Baseline PyTorch Implementation of Generator and Discriminator Networks
- Report Sections 3, 4.1, 4.3

References

[1] Alec Radford, Luke Metz, Soumith Chintala. "Unsupervised Representation Learning With Deep Convolutional Generative Adversarial Networks", ICLR 2016, `https://arxiv.org/pdf/1511.06434.pdf`

[2] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, Yuichi Yoshida. "Spectral Normalization For Generative Adversarial Networks", ICLR 2018, `https://arxiv.org/pdf/1802.05957.pdf`

[3] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, Yoshua Bengio. "Generative Adversarial Nets", `https://arxiv.org/pdf/1406.2661.pdf`