DC-Art-GAN: Stable Procedural Content Generation using DC-GANs for Digital Art

Rohit Gandikota[1] and Nik Bear Brown[1]

Northeastern University, Boston, MA 02115, USA

Abstract. Digital Art is an artistic method of using digital technologies as a part of the generative or creative process. With the advent of digital currency and NFTs (Non-Fungible Token), the demand for digital art is growing aggressively. In this manuscript, we advocate the concept of using deep generative networks with adversarial training for a stable and variant art generation. The work mainly focuses on using the Deep Convolutional Generative Adversarial Network (DC-GAN) and explore the techniques to address the common pitfalls in GAN training. We compare various architectures and designs of DC-GANs to arrive at a recommendable design choice for a stable and realistic generation. The main focus of the work is to generate realistic images that do not exist in reality but are synthesised from random noise by the proposed model. We provide visual results of generated animal face images (some pieces of evidence showing a blend of species) along with recommendations for training, architecture and design choices. We also show how training image preprocessing plays a massive role in GAN training.

Keywords: Procedural Data · Generative Adversarial Networks · Convolutional Networks · Digital Art.

1 Introduction

A procedural content generation is an approach to generating content from random noise after passing it through a model. In this work, we choose the content of digital art images and the model as DC-GAN. Art is a crucial part of the design system in video games and simulations. The character design to small background details requires a lot of creative skill and design. With the help of procedural art generators like our model, a game designer can be assisted with various ideas with control over physical and textural characteristics. However, the training and modelling of such a generator are tricky; this makes it difficult even for experts to provide a realistic, creative, and variant content generator.

The main challenge in providing a realistic content generator lies within the training helm of the process. GAN training is a min-max game that has to reach a Nash equilibrium; the training process is susceptible to architectural design, training parameters and input preprocessing. The work by [3] does talk about the architectural changes compared to [2], and the benefits with it, but do not address the training sensitivity towards the training parameters and preprocessing.

This work aims to close the gap between the design of a realistic content generation and a stable, more consistent training process. The paper talks about the challenges faced with the training and possible solutions. We propose an architecture and training process for a realistic and more reliable content generation of images. Finally, we strengthen our proposal with visual results on various animal face generations inspired by [6].

Generative Adversarial Networks were introduced as generative models, but in recent times, many works have used their ability in input-output correlation applications. [4] have used GANS to segment clouds and water from satellite images. [9] applied GAN in data hiding to encrypt audio inside images. The work by [5] has proposed a generic model for pixel-to-pixel conversion. With the advent of these applications of GAN, the research on generation has relatively declined. [7] have proposed the latest version of GAN that can generate high-quality face images using progressively growing networks. [8] has proposed techniques to improve the training of GAN [2]. They inspire our work to provide not only training procedures but architectural choices and preprocessing choices for improving DC-GAN [3].

2 Method

2.1 Problem Formulation

As depicted in 1, the generator G takes a random 1D noise signal as input and generates 3D content. The discriminator D evaluates a 3D image to classify either a real sample or a fake sample generated by the G. This classification is done at a patch level, classifying each patch in an image as fake or real. The generator improves itself, intending to fool the discriminator to classify the generated as real optical image patches. The min-max game settles at the Nash equilibrium. The generator produces realistic semantic images with the help of the patch level discrimination and imparted training tricks explained in the latter part of the manuscript. We adhere to the classical loss function of binary cross-entropy that showed promising results compared to MSE or hinge loss.

2.2 Model Architecture Details

The generator network has 5 intermediate hidden layers with 1024 dense input layers followed by 4096 dense to expand the features. A reshape layer is added to bring the domain to 3D for image generation, followed by 3 convolutional transpose layers in expanding decoder (128, 128, 128 filters) and stride 2 to expand the dimensions. We used Leaky ReLU activation with alpha 0.2 after every convolutional in the generator; dropout or batch norm layers are avoided in the generator. This is done to avoid complexity in generator learning as the discriminator already dominates the min-max game due to the easier learning task in the discriminator. We also randomly initialized the weights sampling from a normal distribution of 0 mean and 0.02 standard deviation.

The discriminator network has 5 intermediate hidden layers with 5 convolution layers of stride 2 (64, 128, 128, 256, and 1 filter). The output is a 2D patch of size 4X4 activated with a sigmoid. This ensures that the input is evaluated at patch level (16 equal patches). We use leaky ReLU with 0.2 alpha activation in the intermediate discriminator layers and a sigmoid activation in the output. We also randomly initialized the weights sampling from a normal distribution of 0 mean and 0.02 standard deviation. Unlike the generator, we apply dropout at the output layer to avoid overfitting in the discriminator. We also designed a patch output instead of a standard single unit output to discriminate at the patch level rather than the global image level. This resulted in realistic images in zoom levels as well. The network architecture is depicted in Fig 1

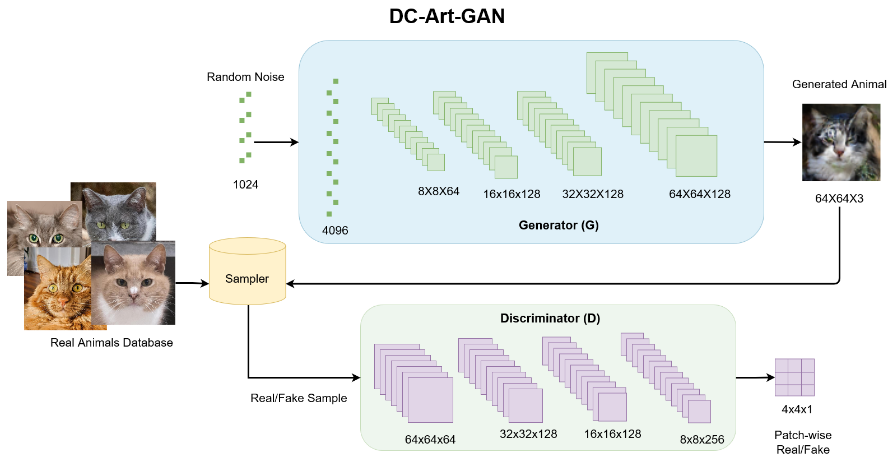

Fig. 1. The proposed network architecture of generating art. The model consists of a generator (to synthesize realistic images) and a patch discriminator (to classify each patch of the provided image as real or fake).

3 Experiment

3.1 Dataset Details

We used the open-source dataset by [6] which contains 16130 high quality (512X512) images of animal faces. There are 3 classes of cats, dogs and wild animals in the dataset, with 5000 samples of each class. The dataset has diversity in species, breeds and backgrounds, making this a preferable dataset for generative model training. These images are subsampled to 64X64 RGB images for our work, and the model is trained on all the 16k images.

3.2 Training

We use an Adam optimizer with a 0.0002 learning rate and 0.5 decay rate to estimate both networks' first moment. For every epoch of discriminator training, the generator was trained twice. The model's training took around 2 days for 30 thousand epochs on a training dataset of 16130 samples. At this stage, we used the standard GAN objective functions cross-entropy loss Eq. 1 for the discriminator to minimize.

$$D_{Loss} = \min_{D} log\mathbf{D}(Real) + log(1 - \mathbf{D}(Fake)) \tag{1}$$

$$Where\ \mathbf{D}\ is\ Discriminator$$

Generator in a GAN is designed to maximize the same loss that discriminator is minimizes (Min-Max Game). But maximizing a loss led to explosion of gradient due to which we propose to minimise the negative of the Eq. 1 as shown in Eq. 2.

$$G_{Loss} = \min_{G} log\mathbf{D}(G(Noise)) \tag{2}$$

$$Where\ \mathbf{G}\ is\ Generator$$

During this virtual min-max optimization, we chose the proportional constants in Eq. 3, η_1 and η_2, in the direct proportion of the depth of generator and discriminator respectively. This is done to achieve stable adversarial training and achieve Nash equilibrium.

$$GAN_{Loss} = \eta_1 D_{Loss} + \eta_2 G_{Loss} \tag{3}$$

4 Addressing Common Pitfalls in GAN Training

4.1 Poor Generator Performance

GAN training is the most common challenge where a generator's outputs are very distorted and poor, even with a large dataset and abundance of training epochs. To understand the root cause of this problem, we suggest analyzing the training curve of the discriminator of real and fake samples individually. It is usually the case where a discriminator learns faster than a generator, leaving minute gradients for the generator to learn, as shown in Fig. 2.

This can be addressed with a few possible measures,

- Increase learning rate of generator for faster learning (not preferable as this may lead to an overshoot of global minima).
- Reduce the complexity of generator architecture for relatively faster convergence. (may lead to depreciation in output quality as we are reducing the model complexity to map higher-dimensional features)
- Removing any complex training layers like dropouts and batch norm layers (preferable as this does not compromise on convergence or quality)
- Ensuring the discriminator is set as non-trainable when training the generator.

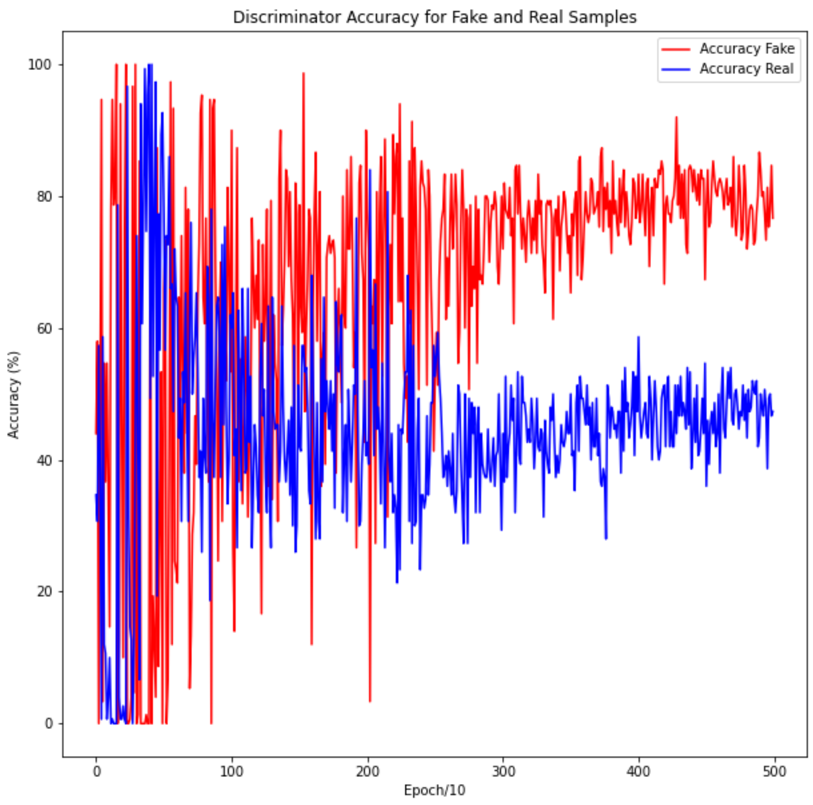

Fig. 2. The training losses for discriminator when evaluating fake and real samples separately showing the unfair min-max game. The red curve represents the discriminator's performance in accurately classifying fake samples (higher accuracy representing bad generation). The blue curve represents the performance of real samples. Ideally, the generator should synthesize realistic output and confuse the discriminator by forcing the curves to converge at 50%. This case shows a degradation in the generator's output.

4.2 Mode Collapse

Another common challenge in GAN training is Mode Collapse. It is the state where the generator outputs the same image for every random input. This occurs when the discriminator converges at local minima, and the generator takes advantage of the situation to settle at a local state of generating the same image as shown in Fig. 3. Mode collapse has to be primarily addressed at the discriminator level and then generator on a secondary level.

- Adding Leaky ReLU activation at the discriminator will help boost the classification training as negative activations can be perceived as punishments bringing a reward-punishment based training system.
- Reducing the learning rate of the discriminator to avoid local minima convergence.
- Increasing the batch size of training with batch training to bring more diversity to the training process, pushing the discriminator out of converged local minima.

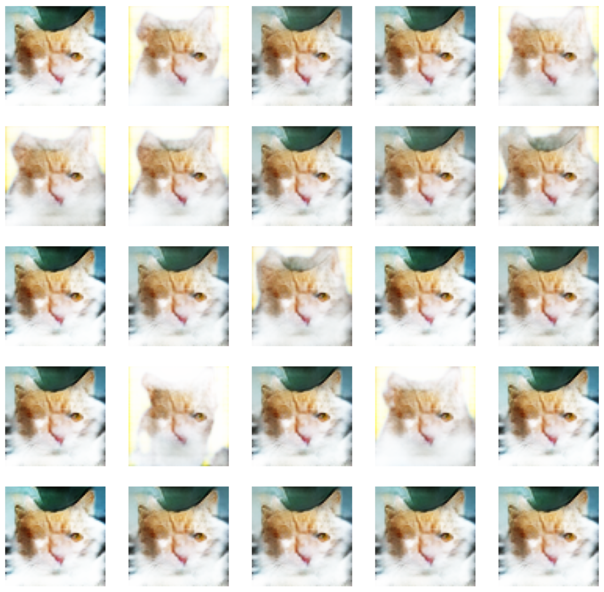

Fig. 3. Mode collapsed samples after training. The samples have zero variance in the foreground's structural and texture features. This state of the generator is called Mode Collapse. Mode collapse may occur in the foreground, background, or the entire image. This particular sample is a foreground mode collapse.

4.3 Saturation of Output Under Good Min-Max Curves

Even though the training curves show a fair min-max game, the generator output saturates due to the input space pre-processing during training, as shown in Fig. 4. It is essential to understand the pre-processing in the information flow domain. When global pre-processing is done, each image is normalized based on the statistical parameters of the entire dataset. This means the information of the dataset is biasing that information of a sample. This plays a huge role, especially in GAN training, as the generator learns information a transformation from random noise to the distribution of the dataset rather than a pixel to pixel mapping.

Due to the reasons mentioned above, sample-based normalization independent of dataset parameters will hugely improve the quality of output and more artistic variations, as shown in Fig. 5.

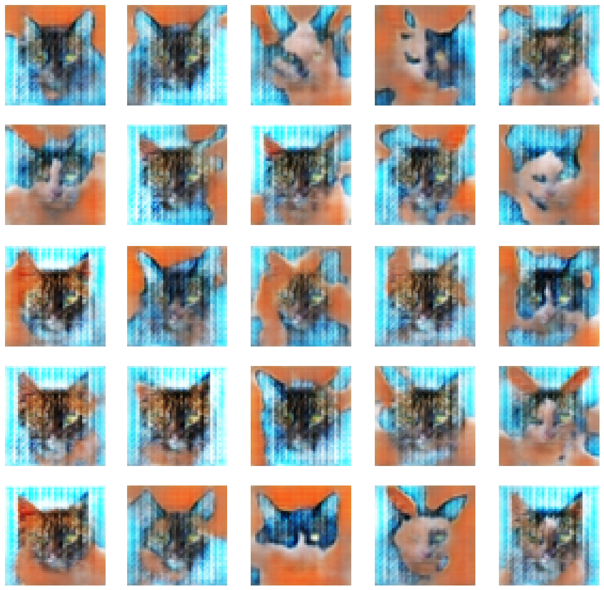

Fig. 4. Saturated samples after training. The samples have variance and display animal features but are saturated. This is evidence of a good min-max game, but the radiometry of the image is not represented properly by the generator.

5 Future Work

This work can be extended to any dataset, and we plan to explore various other datasets like paintings, statues, architecture, etc. We would like to progressively increase the output resolution generated using a temporal progression with the soft transfer. This method was proved on beta-VAEs and we plan to explore the same on DC-GANs. Style-GAN[11] can be explored to add additional variance by infusing random noise in each layer, improving the variance of outputs. VAE-GAN[10], an architecture where a decoder of a VAE being used as a generator, will be explored to denoise the output and improve the quality. We also plan to explore disentanglement in DC-GANs with particular emphasis on controllability.

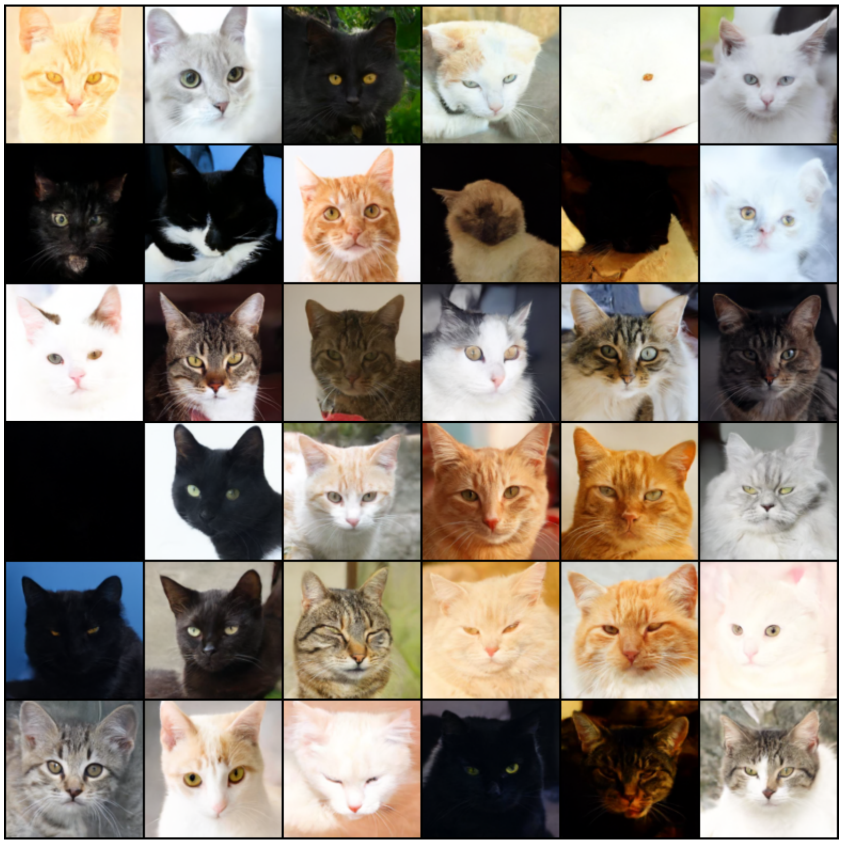

Fig. 5. Samples of generated images from the final model of DC-Art-GAN. The samples include both realistic and distorted images. The model tends to fail for some samples, especially for the classes with less number of samples provided during training. The outputs are generated in low resolution at 64X64 size.

6 Conclusion

This work proposes an architectural setting for a consistent and stable procedural content generation using DC-GANs. Common pitfalls are addressed, and an analysis of training curves has been discussed. We also proposed the use of patch wise discrimination in DC-GAN that improved the local quality of generated images. An information theory point-of-view is explored to address the considerable role input pre-processing plays in GAN training. This manuscript enables DC-GAN, light and easy to use architecture, to generate realistic and consistent images for digital art using training tricks and architectural recommendations.

References

1. Karras, T., Laine, S., Aila, T. (2019). A style-based generator architecture for generative adversarial networks. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (pp. 4401-4410).
2. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., ... Bengio, Y. (2014). Generative adversarial nets. Advances in neural information processing systems, 27.
3. Radford, A., Metz, L., Chintala, S. (2015). Unsupervised representation learning with deep convolutional generative adversarial networks. arXiv preprint arXiv:1511.06434.
4. Gandikota, R., Sharma, A., ManjuSarma, M., Bothale, V. M. (2020). RTC-GAN: Real-time classification of satellite imagery using deep generative adversarial networks with infused spectral information. In IGARSS 2020-2020 IEEE International Geoscience and Remote Sensing Symposium (pp. 6993-6996). IEEE.
5. Isola, P., Zhu, J. Y., Zhou, T., Efros, A. A. (2017). Image-to-image translation with conditional adversarial networks. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 1125-1134).
6. Choi, Y., Uh, Y., Yoo, J., Ha, J. W. (2020). Stargan v2: Diverse image synthesis for multiple domains. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (pp. 8188-8197).
7. Brock, A., Donahue, J., Simonyan, K. (2018). Large scale GAN training for high fidelity natural image synthesis. International Conference on Learning Representations.
8. Salimans, T., Goodfellow, I., Zaremba, W., Cheung, V., Radford, A., Chen, X. (2016). Improved techniques for training gans. Advances in neural information processing systems, 29.
9. Gandikota, R., Mishra, D. (2019, December). Hiding Audio in Images: A Deep Learning Approach. In International Conference on Pattern Recognition and Machine Intelligence (pp. 389-399). Springer, Cham.
10. Yu, X., Zhang, X., Cao, Y., Xia, M. (2019, August). VAEGAN: A Collaborative Filtering Framework based on Adversarial Variational Autoencoders. In IJCAI (pp. 4206-4212).
11. Karras, T., Laine, S., Aila, T. (2019). A style-based generator architecture for generative adversarial networks. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (pp. 4401-4410).