Discriminator Synthesis: On reusing the other half of Generative Adversarial Networks

Diego Porres
Computer Vision Center (CVC), Universitat Autònoma de Barcelona
Barcelona, Spain, 08193
dporres@cvc.uab.es

Abstract

Generative Adversarial Networks have long since revolutionized the world of computer vision and, tied to it, the world of art. Arduous efforts have gone into fully utilizing and stabilizing training so that outputs of the Generator network have the highest possible fidelity, but little has gone into using the Discriminator after training is complete. In this work, we propose to use the latter and show a way to use the features it has learned from the training dataset to both alter an image and generate one from scratch. We name this method Discriminator Dreaming, and the full code can be found at https://github.com/PDillis/stylegan3-fun.

1 Introduction

Generative adversarial networks (GAN) [9] have, irrevocably, altered the course of machine-aided art. Some architectures are better suited for the task [23, 4, 28], mainly due to the eventual ease of use and evaluation of these models, as well as the availability of data and compute. However, the main focus in the literature has been to increase the fidelity and stabilization of the images being generated [12–15] and, while the Discriminator has to also increase its capability in order for these larger networks to be trained, it is usually left unexplored if not outright discarded.

While previous efforts have gone into using the Discriminator as part of an optimization process, either to use it in order to generate images it deems to be fake [3] or to use it alongside another classifier to update the weights of the Generator [2], more effort can be done here, especially to use these large networks in an artistic manner. These networks have learned unique features from both the real and fake dataset, and they are simply being thrown away.

This work has two main sources of inspiration. The first is by Robbie Barrat who used the Discriminator from a trained GAN to optimize the tile placements of a nude portrait [1]. The second is the recent global effort to use large pre-trained models such as CLIP [24], which would not have been able to be trained by an individual researcher or artist. Two notable examples are the Big Sleep [21] and Control The Soul [6], who use CLIP alongside BigGAN and StyleGAN, respectively, to generate images from prompts.

Even though the features learned by a Discriminator may not be used for other downstream classification tasks (e.g., such as the models trained on ImageNet [7] which are then fine-tuned for other tasks), it is still worth exploring the learned features, especially for artistic endeavors. Indeed, what warrants as a 'fake' or 'real' image for the network itself will not translate to logical human sight features per se, but nevertheless these may still hold beauty.

We can use the Discriminator in a variety of ways. For example, we could use it in lieu of a VGG network [26] for style transfer [8], or when projecting an image using the Generator of the same network (or even using another Generator altogether). Staying within StyleGAN, we could use it as a

35th Conference on Neural Information Processing Systems (NeurIPS 2021), Sydney, Australia.

Figure 1: Discriminator Dreaming with StyleGAN2 models of different resolutions. From left to right and top to bottom: (1024 resolution) MetFaces and MinecraftGAN; (512 resolution) Huipiles, FFHQ-512, Cars, and AFHQ-Cat; (256 resolution) Doors, FFHQ-256, LSUN Horse, and LSUN Church. Results shown to scale. Best viewed in color.

sort of feedback loop with the Generator, as the shape of the fully-connected layer before the final output (that is, the output shape of `D.b4.fc`) has the same shape as the latent space $\mathcal{Z}$.

As a proof of concept, we use StyleGAN2's Discriminator instead of the Inception network [27] for Inceptionism/DeepDream [20]. We provide a starter code on how to extract the intermediate features of the Discriminator (with residual [10] connections) of StyleGAN2, in particular, its ADA (PyTorch) version, but is also compatible with StyleGAN3's [16] repository and models. Our code is built upon the DeepDream-PyTorch repository [19] and we name our algorithm Discriminator Dreaming.

2 Discriminator Dreaming

The following are some preliminary results we have obtained, though the final code is subject to change. For this work, we used a vast collection of available pre-trained models, some trained by third parties or provided by the authors of the original papers, others trained by ourselves. These official models were introduced in StyleGAN2 and StyleGAN2-ADA papers, and can be found in their respective GitHub repositories. These include: at 1024 resolution, FFHQ and MetFaces; at 512 , FFHQ-512, Cars, and AFHQ-Cat; and at 256, FFHQ-256, LSUN Horse, and LSUN Church.

Other models include, at 1024 resolution, MinecraftGAN, a model trained with images taken from a first-person POV in the videogame Minecraft throughout different weathers and biomes [11]. At 512 resolution, the Huipiles model was trained with images from huipiles from Guatemala and Mexico [22]. At 256 resolution, Doors is a model trained on art nouveau doors found in Barcelona [17].

In Figure 1, we can see the result of Discriminator Dreaming using different layers for various models. We use, as a starting image, `00012.png` from the FFHQ dataset introduced in StyleGAN. This is using a single frame, so for a video results, refer to Appendix A.1. As per this section, for future work, we could set the zoom, rotation, and translation to be variable instead of fixed, by e.g. using an audio-reactive code [5].

2.1 Limitations

Due to the iterative nature of this algorithm, it is slow to synthesize images, so perhaps more efficient algorithms could be used. Resizing the octaves (see Appendix A) usually makes the code run twice as slow on a RTX 2070, making it less energy-efficient as a trained Generator. However, if we do not resize the octaves, then the image loses its sharpness, resulting in fuzzy images, as well as sometimes giving us vastly different results. Thus, a lot of tweaking and exploration will be needed in order to get the desired result. See Figures 2 and 3 for a difference when resizing or not the octaves.

Acknowledgments and Disclosure of Funding

Diego Porres acknowledges the financial support to perform his Ph.D. given by the grant PRE2018-083417.

Ethical implications

This work is meant for purely artistic use. We have limited our work to using StyleGAN2 and StyleGAN3's Discriminator, for which usually the available pre-trained models are based on publicly-available datasets. However, the trained Discriminator may have learned the distribution of images it was trained on for which we might not have the rights to. Fully utilizing the Machine Learning models we train is what this work strives towards, as the environmental implication of training each from scratch and then discarding half of the network should not be so easily dismissed.

References

[1] Robbie Barrat. Untitled (GAN imagery without using the generator), 2019. URL https://twitter.com/videodrome/status/1158419262463258624.

[2] Terence Broad and Mick Grierson. Transforming the output of gans by fine-tuning them with features from different datasets. *CoRR*, abs/1910.02411, 2019. URL http://arxiv.org/abs/1910.02411.

[3] Terence Broad, Frederic Fol Leymarie, and Mick Grierson. Amplifying the uncanny. *CoRR*, abs/2002.06890, 2020. URL https://arxiv.org/abs/2002.06890.

[4] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale GAN training for high fidelity natural image synthesis. *CoRR*, abs/1809.11096, 2018. URL http://arxiv.org/abs/1809.11096.

[5] Hans Brouwer. Audio-reactive latent interpolations with stylegan. In *Proceedings of the 4th Workshop on Machine Learning for Creativity and Design at NeurIPS 2020*, December 2020. URL https://jcbrouwer.github.io/assets/audio-reactive-stylegan/paper.pdf.

[6] Katherine Crowson. Control The Soul, 2021. URL https://chainbreakers.kath.io/.

[7] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[8] Leon A. Gatys, Alexander S. Ecker, and Matthias Bethge. A neural algorithm of artistic style. *CoRR*, abs/1508.06576, 2015. URL http://arxiv.org/abs/1508.06576.

[9] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks, 2014.

[10] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015. URL http://arxiv.org/abs/1512.03385.

[11] Jeff Heaton. Minecraft GAN, 2020. URL https://github.com/jeffheaton/pretrained-gan-minecraft.

[12] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of gans for improved quality, stability, and variation. *CoRR*, abs/1710.10196, 2017. URL http://arxiv.org/abs/1710.10196.

[13] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. *CoRR*, abs/1812.04948, 2018. URL http://arxiv.org/abs/1812.04948.

[14] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of stylegan. *CoRR*, abs/1912.04958, 2019. URL http://arxiv.org/abs/1912.04958.

[15] 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. URL https://arxiv.org/abs/2006.06676.

[16] Tero Karras, Miika Aittala, Samuli Laine, Erik Härkönen, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Alias-free generative adversarial networks. *CoRR*, abs/2106.12423, 2021. URL https://arxiv.org/abs/2106.12423.

[17] Vasily Korf. Modernisme meets StyleGAN, 2020. URL https://vasilykorf.com/doors-stylegan/.

[18] Henry Lim. Earth View by Google, 2016. URL https://github.com/limhenry/earthview.

[19] Erik Linder-Norén. PyTorch-Deep-Dream, 2019. URL https://github.com/eriklindernoren/PyTorch-Deep-Dream.

[20] Alexander Mordvintsev. Inceptionism: Going Deeper into Neural Networks, 2015. URL https://ai.googleblog.com/2015/06/inceptionism-going-deeper-into-neural.html.

[21] Ryan Murdock. The Big Sleep, 2021. URL https://twitter.com/advadnoun/status/1351038053033406468?lang=en.

[22] Diego Porres. Threaded History, 2020. URL http://www.aiartonline.com/art-2020/diego-porres-2/.

[23] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. In Yoshua Bengio and Yann LeCun, editors, *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, 2016. URL http://arxiv.org/abs/1511.06434.

[24] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. *CoRR*, abs/2103.00020, 2021. URL https://arxiv.org/abs/2103.00020.

[25] Derrick Schultz. Freagan (StyleGAN2 trained on Frea Buckler's artwork), 2020. URL https://twitter.com/dvsch/status/1255885874560225284?s=20.

[26] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition, 2015.

[27] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott E. Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. *CoRR*, abs/1409.4842, 2014. URL http://arxiv.org/abs/1409.4842.

[28] 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. URL http://arxiv.org/abs/1703.10593.

A Appendix

Discriminator architecture A full breakdown of StyleGAN2's Discriminator architecture is beyond the scope of this work, though we refer the interested reader to its respective paper. In short, we will be mostly interested in the two main convolutional layers per block (e.g., `D.b1024.conv0` and `D.b1024.conv1` for the block at 1024×1024 scale). There will be $\log_2 (\texttt{D.img\_resolution}/4)$ blocks in total, and the last one at 4×4 scale will have a minibatch standard-deviation layer, a convolutional layer, and two fully-connected layers, the latter one giving the actual output of the Discriminator (i.e., the same as if passing `D(input_image, None)` to an unconditional model).

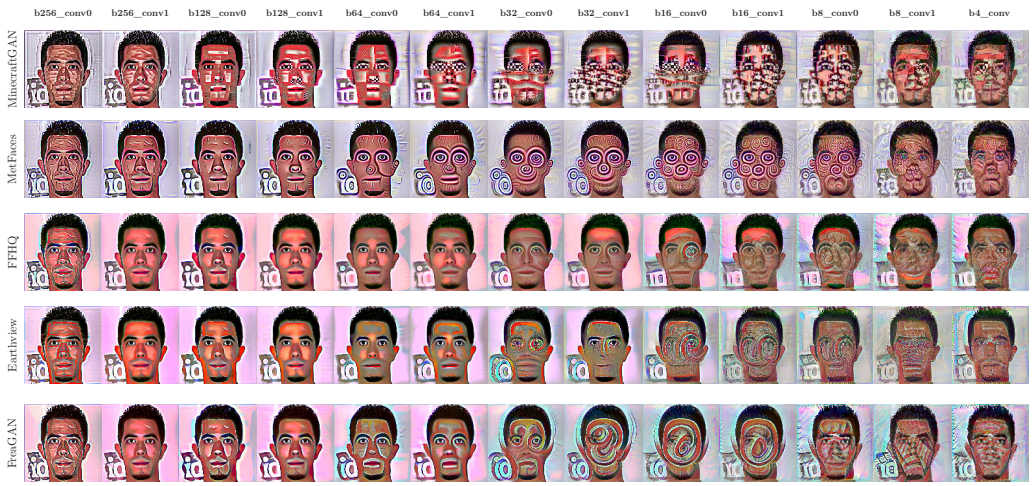


Figure 2: Discriminator Dreaming with different models of 1024×1024 resolution, without resizing the octaves. Columns denote the individual layer we are using to do the dreaming (starting from block 256×256 down to 4×4) and rows the models used. We use image `00012.png` as a starting image. Zoom in for details. Best viewed in color.

Discriminator Dreaming with intermediate layers We show the preliminary results of Discriminator Dreaming with a StyleGAN2 Discriminator. In Figures 2 and 3 we show results using the intermediate layers: from blocks at scales from 256×256 down to 4×4. In the selected models, the blocks at higher scale generally did not produce meaningful results, and the last fully-connected layers added little of interest, compared to the convolutional layer in the block at scale 4×4. However, all of the available layers can be accessed and used by the user if so is desired.

We show results with two additional models at 1024×1024 resolution. Earthview is a model trained with images from Earth View from Google, which are pre-selected satellite images of the most beautiful landscapes in Google Earth [18]. FreaGAN is a model by Derrick Schultz which was trained on Frea Buckler's art [25].

For both of these networks, we can see in Figures 2 and 3 that they were trained via transfer learning from FFHQ, as the results when using the convolutional layers of the larger blocks are quite similar. This makes sense, as the training of these models was limited compared to that of FFHQ, mainly due to a difference in the size of each dataset, so the gradients were mostly concentrated in the lower resolution blocks.

Resizing octaves We can apply the algorithm to a single image, but that would rarely produce significant amount of alteration. In order to affect different layers of granularity of the image, we can apply Discriminator Dreaming at different scales by reducing the size of the image and adding this loss to the overall image. Each scale of the image is referred to as an octave.

However, given that our Discriminator has pretty aggressive downscaling per block (each block lowers the resolution by a factor of 2), we might eventually encounter errors of size mismatch. To solve this, we can simply resize each octave back to the original resolution. This will make the code run slower, but we also obtain different results, as can be seen in Figures 2 and 3 (using the same parameters, except whether or not we resize the octaves).

Random image In lack of a starting image, as per the original DeepDream results, we can start from a random noise image (initialized with a seed), to see what the network generates. Figure 4 shows the result of Discriminator Dreaming with two different layers, starting from a random image with seed 0.

Using more than one layer The user may also use more than one layer to play with. One of the layers may overtake the other, so we can normalize each by either the number or square root

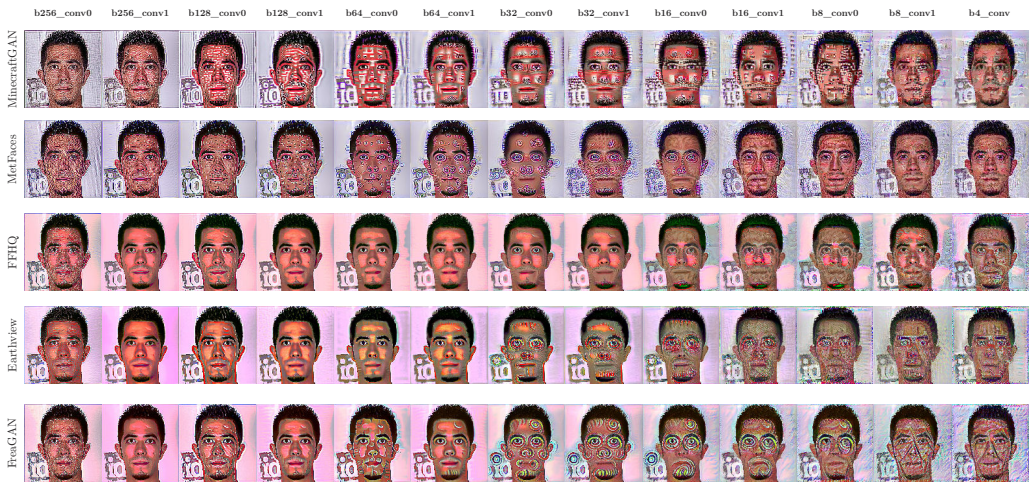


Figure 3: Discriminator Dreaming with different models of 1024×1024 resolution, plus resizing the octaves. Columns denote the individual layer we are using to do the dreaming (starting from block 256×256 down to 4×4) and rows the models used. We use image `00012.png` as a starting image. Zoom in for details. Best viewed in color.

of elements of elements per layer. There is no limit on the number of layers to use, though it is recommended to start with a lower amount for better control of the final result.

A.1 Video

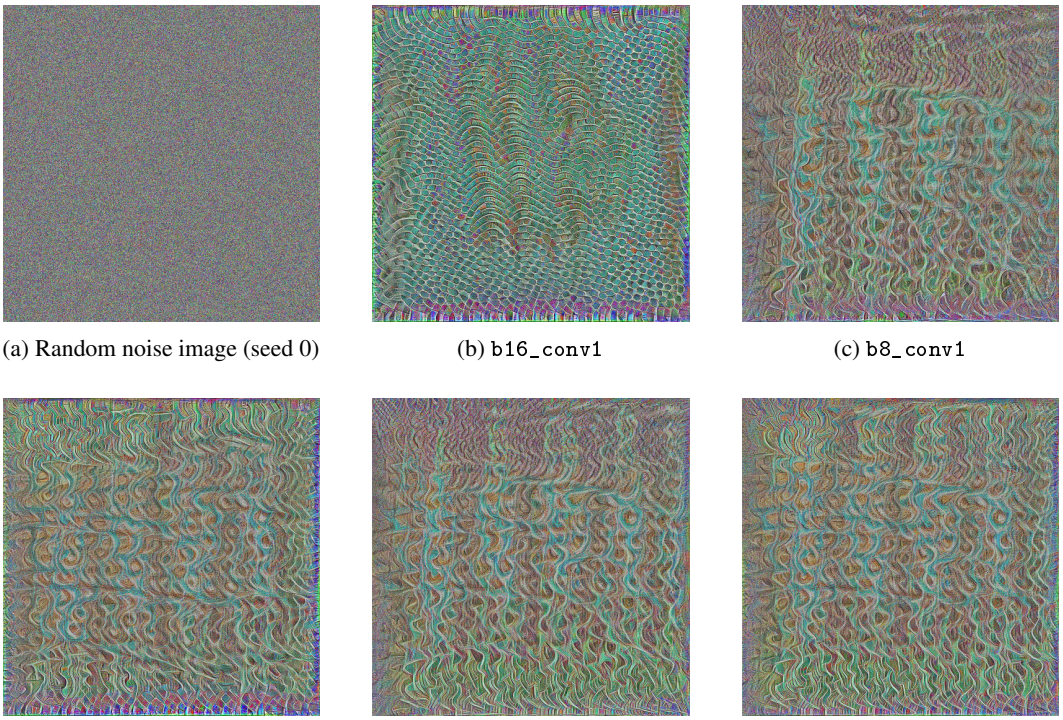
Applying the algorithm iteratively on the outputs, as per the original DeepDream, we can create some interesting video outputs. In each case, the user can set the frames per second (FPS) and length of the video, learning rate (so as to better control the effect per frame), the number of octaves, the octave scale, which layers to use from the Discriminator, should each layer be normalized, etc. Each frame will be saved in the resulting directory, allowing for the code to save as a video in the end or for the users to use each individual frame as they please. The full code and list of available settings can be found at `https://github.com/PDillis/stylegan3-fun`.

We showcase some examples of zoom, rotation, and translation. For each, if the image is zoomed or rotated, the background will be black by default, but this can be changed by the user. Note that each of the following can also be 0, so the generated video will be iteratively applying the effect to a static image.

Zoom We zoom the dreamed image after each iteration by a fixed amount of pixels and then we resize it back again to the Discriminator's expected image size, forming a loop. A sample video of zooming in (by one pixel) can be found at `https://youtu.be/HAO3Kj2Y2Ok`. In it, we use the layer `b16_conv1` of the MetFaces model, starting from image `00012.png` of FFHQ. We use 5 octaves whilst resizing them, a learning rate of 5×10^{-3}, for a total of 30 seconds at 30 FPS, with 10 iterations per frame. Note that the number of pixels to zoom by can be of arbitrary magnitude and sign, so a zoom-out result can also be obtained if it is set to e.g. -1.

Rotation We rotate each dreamed image by a fixed angle counterclockwise, usually low, before feeding it back to the Discriminator (without expanding the image). A sample video of rotation (rotating by $0.2\,\text{deg}$ counter-clockwise) can be found at `https://youtu.be/0QD1O_g-VcE`. In it, we use layer `b8_conv1` of the MinecraftGAN model, using 5 octaves without resizing, a learning rate of 5×10^{-3}, for a total of 60 seconds at 30 FPS, with 20 iterations per frame.

The rotation angle can be higher (or negative), it can be set so that the user gets the desired result. Another sample video using both rotation (by $0.1\,\text{deg}$ counter-clockwise) and zoom (by -1 pixel) can be found at `https://youtu.be/hEJKWL2VQTE`. We use the same settings as the previous video, but an FPS of 25 as well as reversing the final video, resulting in a fake zoom-in.



(a) Random noise image (seed 0) (b) `b16_conv1` (c) `b8_conv1`

(d) `b16_conv1` and `b8_conv1`, no normalization (e) `b16_conv1` and `b8_conv1`, with normalization (f) `b16_conv1` and `b8_conv1`, with square-root normalization

Figure 4: Results of Discriminator Dreaming starting from a random noise image, using the Earthview model and two different layers. We use 10 octaves (and resize them), with an octave scale of 1.4, learning rate of 0.01, for a total of 20 iterations. When using multiple layers, we can normalize each by either its number of elements or square-root of the number of elements, so as to let each layer have equal effect on the final image.

Translation We can translate the image in both horizontal and vertical axes in an independent manner, giving a panning result. The image is translated from left to right and from top to bottom when using positive values of translation. As an example, a sample video using both horizontal and vertical translation can be found at `https://youtu.be/_WCyhrymo-0`. We use the `b16_conv1` layer of the MinecraftGAN model and translate both axes by 1 pixel per frame, a learning rate of 5×10^{-3} with 20 iterations, using 5 octaves without resizing.