StyleGAN2-ADA Behaviour with Limited Datasets

Joshua McCord and Greg Turk

1 Introduction

Nvidia's StyleGAN2-ADA [NVl20] is normally trained using thousands of images, exemplified in their initial testing of the FFHQ dataset. In this write-up, we wanted to explore the behavior of the StyleGAN2-ADA using various forms of a limited dataset. Limited in this context implies a small number of images in addition to a heavy imbalance of a certain type of image.

2 Goals

An unbalanced and uneven dataset can be defined as a dataset that has various sub-categories in which one or more of those sub-categories occupies a significant portion of the dataset. In practice, a researcher may be forced to use one of these unbalanced and uneven datasets. When facing this situation, it's possible for a GAN to be trained inefficiently and result in image synthesis that isn't a reflection of the researchers requirements. Nvidia's initial StyleGAN2-ADA solves this problem partially by the addition of the Augmentation Pipeline. The Augmentation Pipeline is capable of augmenting batches of photos at a time using any combination or subset of: blit, geom, color, filter, noise, cutout. This initial approach works well when training on smaller datasets (Nvidia shows a significant quality increase with datasets less than ∼30k training images). The issue of category distribution within this constrained dataset is still present, as all images get funneled through the Augmentation Pipeline regardless. This write-up explores the various ways to selectively augment subsets of the larger dataset and determine if selective augmentation generates images of a higher quality.

3 Description of Experiment

3.1 Dataset Descriptions

To minimize the number of variables in the dataset we were testing, we used a custom dataset of roses which was composed of two categories:

1. Red Roses
2. White Roses

We gathered these photos using Flickr, Google Images, and Bing through custom scripts or browser plug-ins. After scraping the data, we processed each photo through a small pipeline. This pipeline would first crop the photo to be a square. Next it would resize the photo to 256x256 pixels. It would then store the photo in a selected folder depending on the color of the flower. Finally, at the end of the pipeline, we ran a script to remove duplicate photos to minimize any bias. Each of the following datasets contained the same 400 red roses throughout each, but varied the white roses in each. The following describe the white roses in each dataset:

3.1.1 Unbalanced Dataset

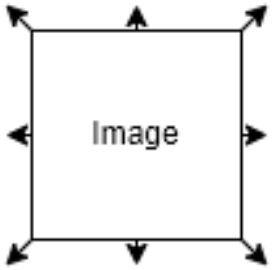
For the unbalanced dataset we used only 50 white roses. There was no particular form, angle, or characteristic.

3.1.2 Balanced Dataset

For the Balanced dataset we used 400 unique images of white roses. The starting point for this dataset was the original 50 white roses from the unbalanced dataset.

3.1.3 Balanced and Manually Augmented Dataset

For the Balanced and Manually Augmented dataset, we utilized the 50 images of the Unbalanced dataset, but performed an augmentation on those images. For each white rose, we executed a translation augmentation in the 8 directions shown in the figure below. In total, this resulted in a Balanced Dataset.



3.1.4 Balanced and Duplicate Dataset

For the Balanced and Duplicate Dataset, we utilized the 50 images of white roses from the Unbalanced dataset and created 8 copies of each photo resulting in a Balanced Dataset.

3.2 StyleGAN2-ADA Setup and Training

Every training session was run on an Nvidia P100 GPU.
The actual StyleGAN2-ADA we used for the experiments came from Derrick Schultz's implementation [Dvs21] on GitHub using the Google Colab platform. We trained each dataset starting from Nvidia's pre-existing 'ffhq1024' with initial augmentation strength equal to 0, train count equal to 0, and allowing X-axis mirroring (but not Y-axis mirroring). We started with a Gamma Value of 50.0, allowed blitting and geom augmentations, configured for 11gb-gpu, and output snapshots every 4 ticks.
Once the network was configured, we trained for a total of 224 ticks and then used the output .pkl file to generate the sample images.

4 Results

4.1 Distribution from Unbalanced Set

During the early tests, one of our first datasets was the Unbalanced Dataset (3.1.1) that contained 400 red roses and 50 white roses. After training for 224 ticks, we generated 100 images (using the random seeds 0-99), and counted the distribution of white roses to red roses. From this initial test, the GAN generated 11 white roses and 89 red roses. This generated distribution ($\frac{11}{100}$) matches the same distribution from the training set ($\frac{50}{450} \approx \frac{11}{100}$).

4.2 Generated Images

Each image below was generated using their respective .pkl files output from the GAN. Each image was generated using the following seeds (in order of their appearance in each section): 0, 50, 100, 150, 200.

4.2.1 Unbalanced Dataset

4.2.2 Balanced Dataset

4.2.3 Balanced and Manually Augmented Dataset

4.2.4 Balanced and Duplicate Dataset

5 Findings

After discovering the results from (4.1), we output the same seed values from the Balanced and Duplicate Dataset (3.1.4) and saw immediately that the distribution was closer to a 50/50 split than the unbalanced. This seems to show that if an even output of images is a requirement, then some form of selective dataset augmentation may need to take place under StyleGAN2-ADA's current configuration as the initial tests show the output probability is connected to the training set distribution.
We also found that output from the Balanced and Manually Augmented Dataset (3.1.3) generated slightly more varied results than the Balanced and Duplicate Dataset (3.1.4). Under dataset (3.1.4) it seemed as if some of the generated white roses were simply copies of one another.

6 Future Work

First objectives for future work would be to isolate all variables even further. For example, turning off augmentation completely in all tests and then training using our manual augmentation.This would even include expanding on which augmentations we perform manually. Isolating each form that StyleGAN2-ADA uses and recreating them for our own tests would help aid this issue.
Another approach that was attempted unsuccessfully was the modification of the StyleGAN2-ADA Augmentation Pipeline or dataset preparation process. We hypothesized that by modifying the GAN's own Augmentation Pipeline to only augment specifically labeled data could lead to better results and smoother process for the researcher. Although this did not come to fruition during this round of experimentation, we do believe this could be a strong path to follow.

7 StyleGAN3

Towards the end of this experimentation, Nvidia released an early version of their StyleGAN3 model [NVl21]. Although we were not able to go fully in depth with it as the StyleGAN2-ADA, we did get around to training on the same datasets. Here is our configuration and the outputted images:

7.0.1 StyleGAN2-ADA Setup and Training

All StyleGAN3 training was also completed on an Nvidia P100 GPU.
The StyleGAN3 implementation is very specific about which arguments you provide with the corresponding network. We had to configure cbase equal to 16384 (this had to match the original network exactly), cfg equal to stylegan3-t (translational SG3), batch equal to 16, and gamma equal to 2. We started each session from the stylegan3-t-ffhqu-256x256.pkl file.

7.0.2 Unbalanced Dataset

7.0.3 Balanced Dataset

7.0.4 Balanced and Manually Augmented Dataset

7.0.5 Balanced and Duplicate Dataset

References

[Dvs21] Dvschultz. Dvschultz/stylegan2-ada-pytorch: Stylegan2-ada - official pytorch implementation, Jan 2021.

[NVl20] NVlabs. Nvlabs/stylegan2-ada: Stylegan2 with adaptive discriminator augmentation (ada) - official tensorflow implementation, Oct 2020.

[NVl21] NVlabs. Nvlabs/stylegan3: Official pytorch implementation of stylegan3, Oct 2021.