Comparing the interpretability of different GAN architectures

Henry Tu
University of Toronto

Seel Patel
University of Toronto

1 Abstract

A Generative Adversarial Network (GAN) uses an adversarial process between two models which are simultaneously trained to estimate a generative model.[1] There are many variants of GAN architectures, such as COCO-GAN[2] and StyleGAN[3], which both have the ability to generate synthetic images which mimic real images.

We are exploring methods of comparing the quantitative performance of these architectures with their interpretability. Our quantitative measures include C2ST[4], image quality measures[4], Maximum Mean Discrepancy (MMD)[4], etc.

In order to analyze these networks qualitatively, we will use methods such as SmoothGrad[5], Latent Space Exploration[6] and nearest neighbour tests[4] to decipher what the networks have learned.

By combining these two forms of analysis, we hope to gain insight into the relationship between performance metrics and the generated output of the models.

2 Introduction

StyleGAN and COCO-GAN generate images using different techniques to accomplish two different goals: StyleGAN is designed such that high level features of the output image can be finely tuned and adjusted (e.g. lighting, hair colour, etc.)[3]. On the other hand, COCO-GAN generates each part of the image separately before stitching it together in order to simulate the human perception of vision[2]. As a result, we expected each model to generate images with different qualities that relate to their designed tasks.

Although we tried to analyze the models using all the metrics listed in the Abstract, there were a few challenges involved with getting the desired results.

3 Analysis Dataset Generation

Both GAN models were trained on the LSUN Bedroom dataset[7] at a resolution of 256×256. Due to time constraints, we used pre-trained models to generate images for our analysis.

3.1 StyleGAN

NVIDIA Research Projects' Official TensorFlow Implementation of StyleGAN pretrained to the LSUN Bedroom dataset[8] was used to generate 5000 images for analysis. 5000 latent code vectors $\mathbf{z} \in \mathbb{R}^{512}$ drawn from a standard normal distribution which each correspond to an output image.

3.2 COCO-GAN

4 Quantitative Analysis

4.1 C2ST

Classifier Two-Sample Tests (C2ST) is used to predict if two samples came from the same distribution[9]. In the context of evaluating GANs, we will use C2ST to quantitatively measure how well the models mimic real images. We trained a deep convolutional neural network to perform binary classification based on the recommended discriminator architecture from *Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks*[10]. The goal is to compare the margin by which each GAN model is able to trick the classifier as a measure of image generation quality.

To avoid introducing bias, all three datasets (training, validation, test) were equally balanced in real and fake images. The 5000 fake images were further divided in half between images from StyleGAN and COCO-GAN . Out of the combined pool of 10000 images, 5000 were allocated for training, 2500 validation, and 2500 testing.

The classifier was implemented using Pytorch with the following network architecture:

```
test
```

After tuning hyper parameters and performing early stopping with the validation dataset, the following accuracies were attained:

```
Training Accuracy: 82.1% of 5000
Validation Accuracy: 66.8% of 2500
Test Accuracy (Real): 58.4% of 1250
Test Accuracy (StyleGAN): 62.4% of 625
Test Accuracy (COCO-GAN): 86.4% of 625
```

As seen here, the model failed to accurately discriminate between the real and generated images. This suggests that either the generated images are so good that it can't be distinguished from the real images, or the classifier is ineffective.

4.2 Image Quality

asd

4.3 MMD

asd

5 Qualitative Analysis

5.1 SmoothGrad

asd

5.2 Latent Space Exploration

asd

5.3 Nearest Neighbours

asd

References

[1] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. (2014) *Generative Adversarial Networks*

[2] Chieh Hubert Lin, Chia-Che Chang, Yu-Sheng Chen, Da-Cheng Juan, Wei Wei, Hwann-Tzong Chen. (2019) *COCO-GAN: Generation by Parts via Conditional Coordinating*

[3] Tero Karras, Samuli Laine, Timo Aila. (2018) *A Style-Based Generator Architecture for Generative Adversarial Networks*

[4] Brownlee, Jason. (2019) *How to Evaluate Generative Adversarial Networks.*

[5] Daniel Smilkov, Nikhil Thorat, Been Kim, Fernanda Viegas, Martin Wattenberg. (2017) *SmoothGrad: removing noise by adding noise*

[6] Tom White. (2016) *Sampling Generative Networks*

[7] Yu, Fisher and Zhang, Yinda and Song, Shuran and Seff, Ari and Xiao, Jianxiong. (2015) *LSUN: Construction of a Large-scale Image Dataset using Deep Learning with Humans in the Loop*

[8] NVLabs. (2019) *StyleGAN — Official TensorFlow Implementation*

[9] David Lopez-Paz, Maxime Oquab. (2016) *Revisiting Classifier Two-Sample Tests for GAN Evaluation and Causal Discovery*

[10] Alec Radford, Luke Metz, Soumith Chintala. (2015) *Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks*