Understanding and Improving Generative Adversarial Networks

by

Qi Li

A thesis submitted to the Graduate Faculty of
Auburn University
in partial fulfillment of the
requirements for the Degree of
Master of Science

Auburn, Alabama
May 2, 2020

Keywords: Generative Adversarial Networks, Deep Learning, Embedding Layer

Approved by

Anh Nguyen, Chair, Assistant Professor of Computer Science and Software Engineering
Daniel Tauritz, Associate Professor of Computer Science and Software Engineering
Bo Liu, Assistant Professor of Computer Science and Software Engineering

Abstract

Generative Adversarial Networks (GANs) have been under the spotlight in the machine learning field for a few years. Especially, the power that learns a data distribution in an unsupervised fashion leads GANs to be applied to various applications such as page generation, image style transformation, image attribution manipulation, and similar domains in computer vision. Despite the huge success of GANs, the difficult and unstable training process still limits the applications of GANs in the real world. Mode collapse is a well-known byproduct of the unstable GAN training. We propose to improve the sample diversity of a pre-trained class-conditional generator by modifying its class embeddings in the direction of maximizing the log probability outputs of a classifier pre-trained on the same dataset. We improved the sample diversity of state-of-the-art ImageNet BigGANs at both 128×128 and 256×256 resolutions. By replacing the embeddings, We can also synthesize plausible images for Places365 using a BigGAN pre-trained on ImageNet.

Acknowledgments

I have to thank my advisor Dr. Anh Nguyen. I knew nothing about either research or Deep Learning two years ago. He taught me programming, writing and almost everything he knows to help me to be successful. He helped me to become a junior researcher in Deep Learning and Computer Vission. I also thank Dr. Daniel Tauritz and Dr. Bo Liu as my committee members. I am gratefully indebted to their very valuable comments on this thesis. Especially, I want to thank Long Mai from Adobe Research for his support and advice to both of my research projects. I thank Michael Alcorn, Naman Bansal, Thang Pham who are my lab mates for valuable discussions and helpful feedback on the drafts. They not only helped me but also argued with me so much in different ways. Those made us become better researchers and good friends. Additionally, I want to thank Chirag Agarwal who was an intern in our lab and work with me for a wonderful summer. Finally, I thank the supports of my family. They took the responsibilities that should be mine on their shoulder when I was here for my Master's Degree. Finally, I must express my very profound gratitude to my parents, Jianhua Li and Huizhi Qi. This accomplishment would not have been possible without them. Thank you.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

Generative adversarial networks (GANs) have been a hot research topic and attracted growing interests from different background researchers. Yann LeCun said that "GANs are the most interesting idea in the last 10 years in machine learning." GANs have been applied to great effect to a variety of applications such as computer vision, natural language processing, time series synthesis, semantic segmentation, etc. GANs have the advantages that handling sharp estimated functions, generating realistic and diverse samples and eliminating deterministic bias and compatibility with the internal neural networks [2].

The GAN framework usually consists of two parts: a generator, which is learning to transform a simple distribution to a high-dimensional distribution (i.e., the natural images) and a discriminator that tells whether the input distribution is a true distribution or a synthesized one by the generator. These two parts are typically implemented by neural networks, but they can be implemented with any form of differentiable system that maps data from one space to the other. The training of GANs is a minimax optimization problem. The solution to the optimization problem is the Nash equilibrium where neither generator nor discriminator can improve unilaterally. Then, the generator can be thought to have captured the real distribution of true examples.

Generative adversarial networks have been confused with the concept of "adversarial examples" [3]. Adversarial examples are the inputs to a neural network that intentionally designed to cause the network to make a mistake i.e., misclassification, but are visually indistinguishable to some inputs cause correct outputs with the same neural network. I also contributed to [4] that introduces a novel method to generate adversarial examples.

1.1 Generative Adversarial Network

Fig. 1.1 shows a typical training process of GAN. During the training process, the generator (G) learns to make real-like data from a random noise, typically a simple distribution, and the discriminator (D) is trained to distinguish between the real data and generated data. To learn the generator's distribution $p_{\boldsymbol{g}}$ over data $\boldsymbol{x}$, we define a prior on a input noise varialbels $p_{\boldsymbol{z}}(\boldsymbol{z})$ and $G(\boldsymbol{z})$ as the samples from the distribution $p_{\boldsymbol{g}}$. $D(\boldsymbol{x})$ represents the probability that $\boldsymbol{x}$ came from the p_{data} rather than $p_{\boldsymbol{g}}$. The goal of a GAN is to learn the generator's distribution $p_{\boldsymbol{g}}$ that approximates the real data distribution p_{data}. This adversarial learning process can be formulated to a joint loss function V for D and G as shown in Equation 1.1.

$$\min_{G} \max_{D} \; V(D,G) = \mathbb{E}_{\boldsymbol{x}\sim p_{data}(\boldsymbol{x})} \left[\log D(\boldsymbol{x}) \right] \; + \; \mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})} \left[\log \left(1 - D(G(\boldsymbol{z}))\right) \right] \tag{1.1}$$

1.2 GAN Variants

There is a GAN boom after the original GAN released. The researchers work on different loss functions and the architectures of the generator and discriminator to improve the performance and stability of GAN. Several techniques such as batch normalization, stacked architecture, and multiple generators and discriminators are applied to GANs. We list out some remarkable benchmark GANs (see the comparison Fig. 1.2) here to show how the GANs developed in recent years.

1.2.1 Conditional GAN

GANs can be extended to a conditional version if both generator and discriminator are conditioned on class information $\boldsymbol{y}$. The objective function of conditional GANs [5] is as shown in Equation 1.2.

$$\min_{G} \max_{D} \; V(D,G) = \mathbb{E}_{\boldsymbol{x}\sim p_{data}(\boldsymbol{x})} \left[\log D(\boldsymbol{x} \mid \boldsymbol{y}) \right] \; + \; \mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})} \left[\log \left(1 - D(G(\boldsymbol{z} \mid \boldsymbol{y}))\right) \right] \tag{1.2}$$

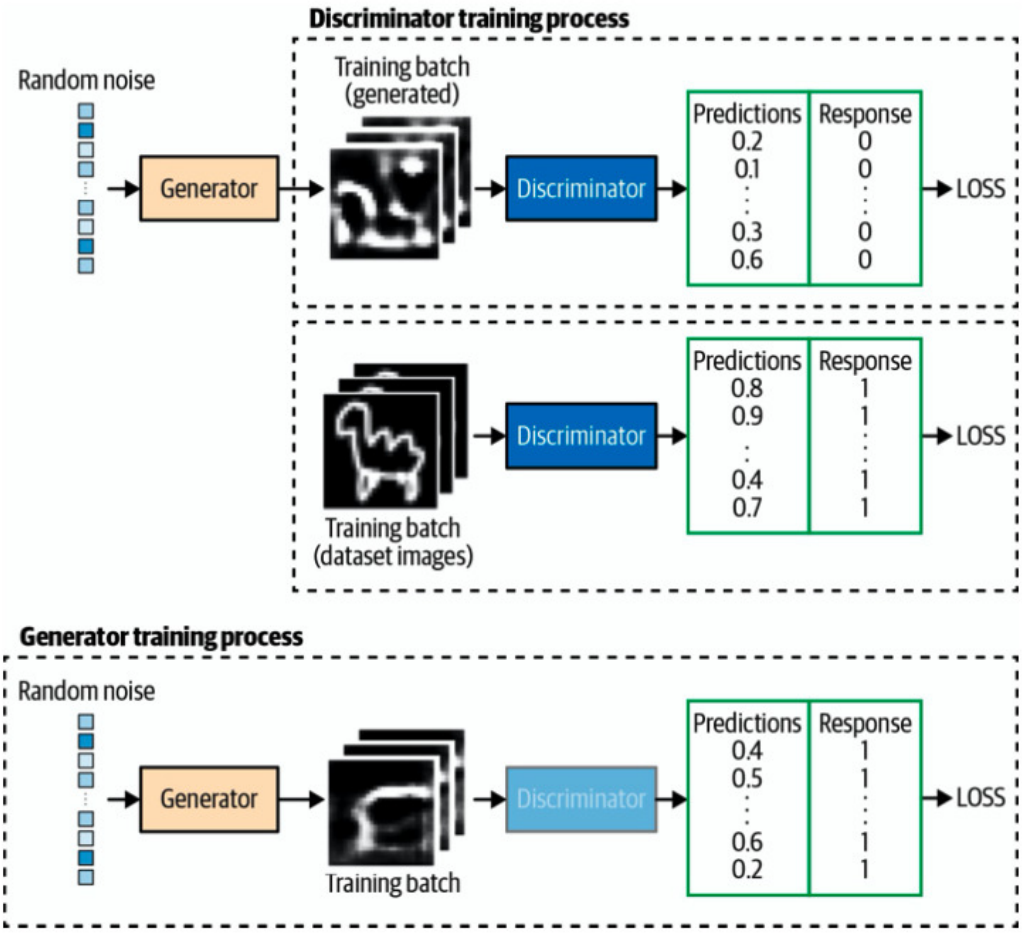


Figure 1.1: The training process of the GAN. Figure from [1]

Conditional GAN shows the potential of conditional adversarial nets and give promise for interesting and useful applications. It inspires some latest GANs to utilize the class information to make GANs generate high quality and diverse images.

1.2.2 DCGAN

Deep convolutional generative adversarial network (DCGAN) [6] provides a significant improvement in performance and stability since the generator and discriminator are defined by deep convolutional neural networks (DCNNs). Most current GANs are at least loosely based on the DCGANs architecture (see Fig. 1.3). The architecture of DCGAN is mostly based on the convolutional net without pooling and upsampling layers. When Gneeds to increase the dimensionality of representations, it uses transposed convolution (deconvolution). The batch normalization is used for most layers of G and D.

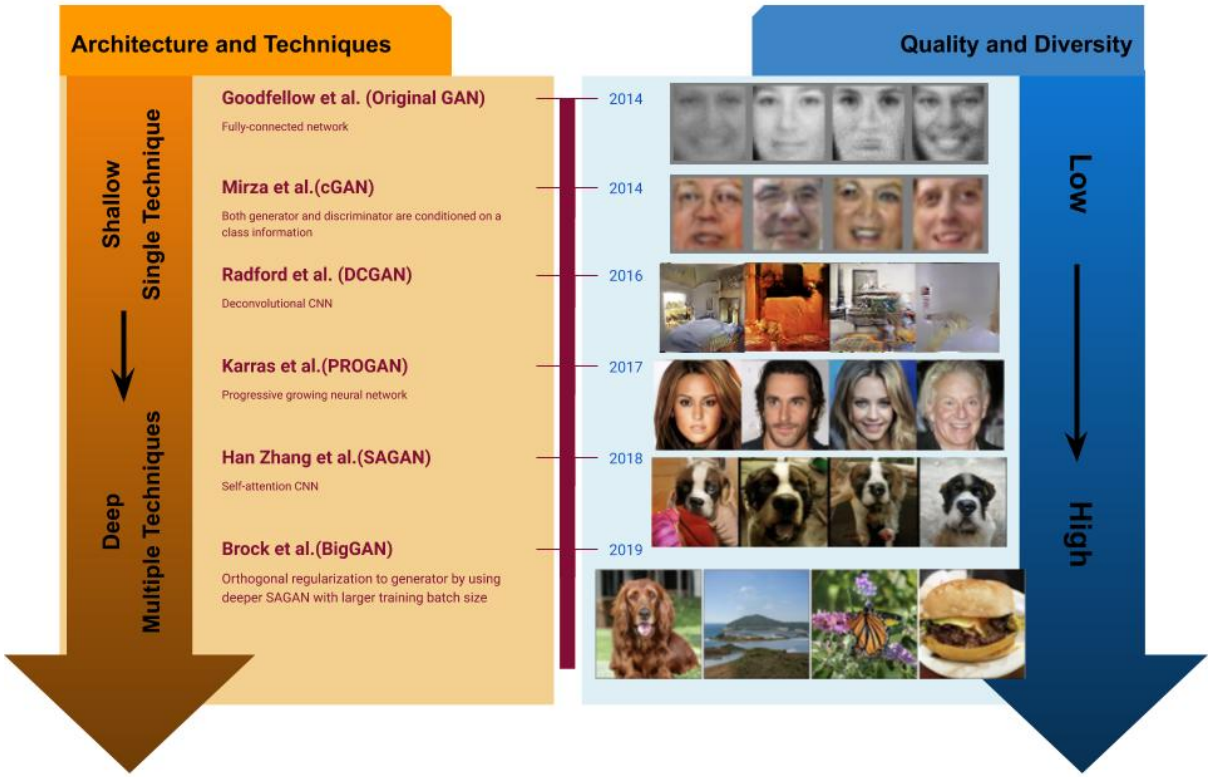


Figure 1.2: Timeline of some benchmark GANs. the orange part shows the architecture variants and techniques used in the GANs, the blue part shows the sample quality and diversity from the generators.

1.2.3 ProgressiveGAN

A new training strategy for GAN is proposed by Progressive GAN (PGGAN) [7]. The structure of PGGAN is based on progressive neural networks that is first proposed by Andrei A et al. [8] in 2016. The key idea (see Fig. 1.4) of PGGAN is to add layers to both the generator and discriminator progressively: starting from a low resolution, adding new layers that model increasingly fine details as training progresses.

1.2.4 SAGAN

Self-Attention Generative Adversarial Network (SAGAN) [9] is proposed to allow attention-driven (see Fig. 1.5), long-range dependency modeling for image generation tasks. Spectral normalization technique is used to the discriminator in SNGAN [10] firstly. SAGAN applied

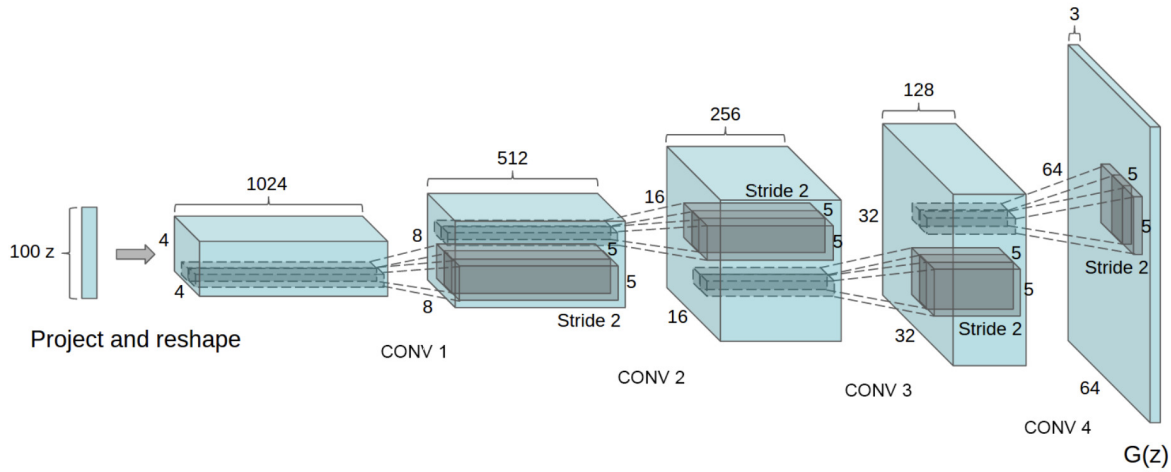


Figure 1.3: A series of four fractionally-strided convolutions then convert this high level representation into a 64×64 pixel image. No fully connected or pooling layers are used. Figure from [6]

spectral normalization for both generator and discriminator, and it improves training dynamics. They also confirmed that the two time-scale update rule (TTUR) is effective in SAGAN.

1.2.5 BigGAN

BigGANs [11] is a large scale TPU implementation of GANs, which has a similar architecture to SAGAN but scaled up greatly. BigGANs can generate realistic images with high resolution up to 512×512 pixels. The architecture of BigGAN shows in Fig. 1.6. Lucic et al. [12] show that BigGANs can be trained to perform homogeneously with fewer labels. BigBiGAN [13], based on BigGANs, extends it to representation learning by adding an encoder and modifying the discriminator. All of the variants of BigGAN show the state-of-the-art performance of generative models.

1.3 GAN Challenges

While GANs have achieved an unprecedented performance for generative tasks, they are also notoriously difficult to train. We will explore some common problems during the training of GAN framework.

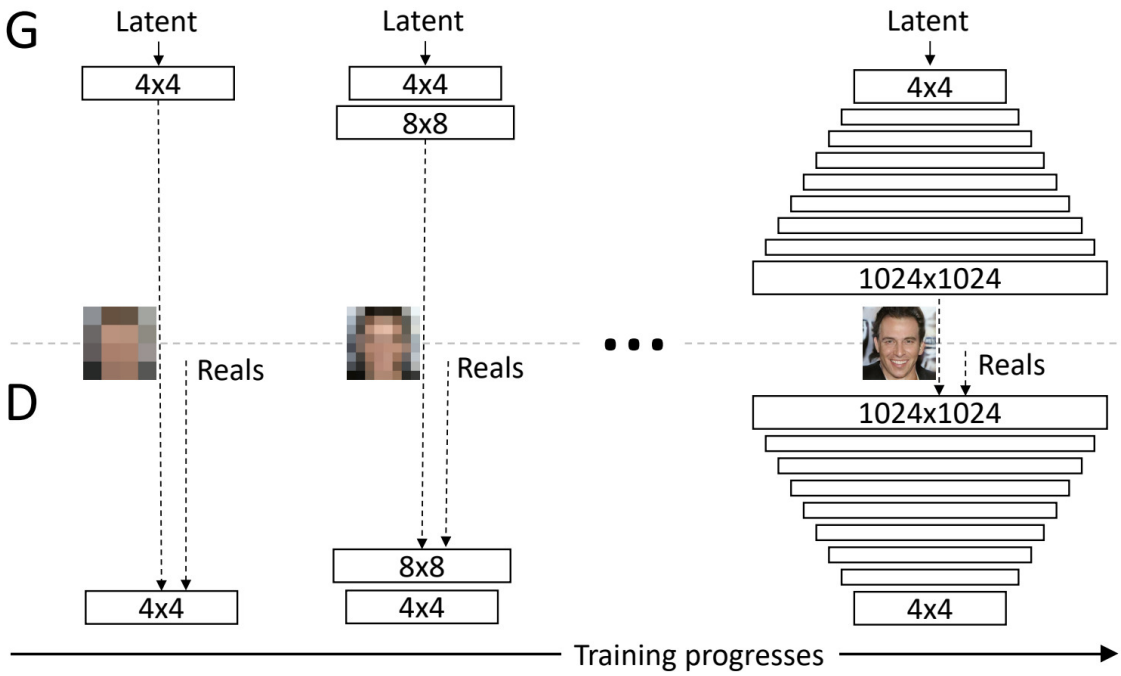


Figure 1.4: Adding layers to G and D incrementally during the training process to increase the spatial resolution of the generated images. This allows stable synthesis in high resolutions and also speeds up training considerably. Figure from [7]

1.3.1 Oscillating Loss

Typically, there is some small perturbation of the loss between batches during the training of deep neural network, but the loss should become stable or gradually increases or decreases, rather than fluctuating in long term, to ensure your GAN converges and improves over time. During the training of GAN, the loss of the discriminator and generator can oscillate wildly, rather than exhibiting long-term stability. Fig. 1.7 illustrate the changing of loss of the discriminator and generator during the training has started to be out of control, at around batch 1,300. It is difficult to establish if or when this might occur as vanilla GANs are prone to this kind of instability.

1.3.2 Hyperparameters

There are a large number of hyperparameters to tune even with simple GANs. Based on different architectures of GANs, the hyperparameters such as batch size, the number of epoch, learning rete, the batch normalization, dropout, activation layers, convolutional filters, kernel

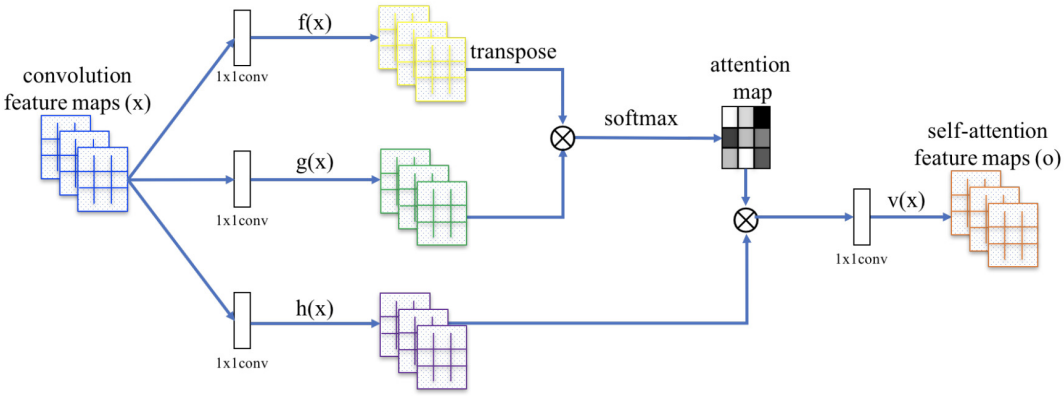


Figure 1.5: The proposed self-attention module for the SAGAN. The $\otimes$ denotes matrix multiplication. The softmax operation is performed on each row. Figure from [9]

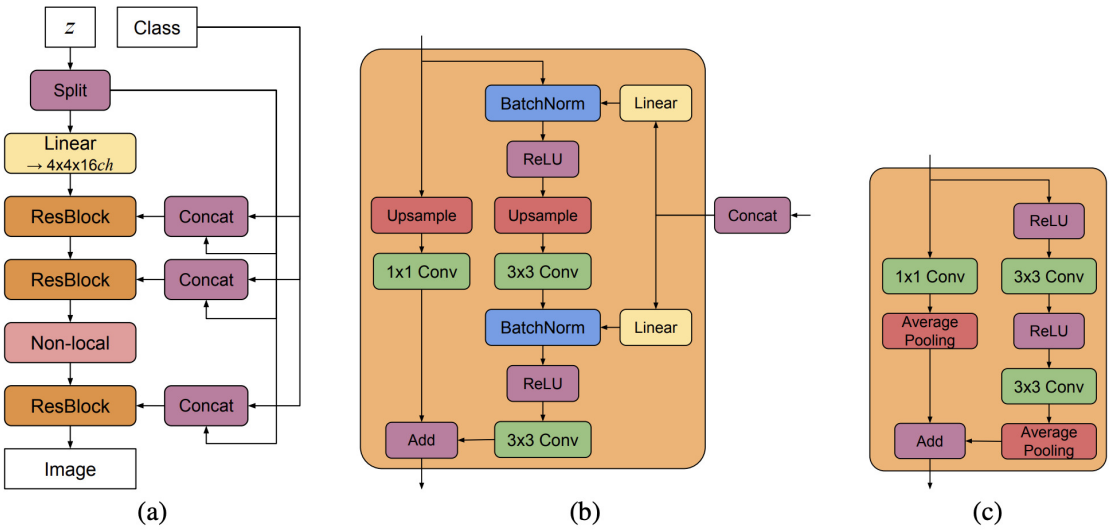


Figure 1.6: (a) A typical architectural layout for BigGAN's G. (b) A Residual Block (ResBlock up) in BigGAN's G. (c) A Residual Block (ResBlock down) in BigGAN's D. Figure from [11]

size, etc should be considered. GANs are also very sensitive to some slight changes in all of these parameters. To find a group of parameters that works well is often a process of empirical trial and error, rather than following an established set of guidelines.

1.3.3 Mode collapse

Mode collapse occurs when the generator finds a small number of samples that fool the discriminator and therefore isn't able to produce any examples other than this limited set. Suppose we train the generator over several batches without updating the discriminator in between. The generator would be inclined to find a single observation (also known as a mode) that always

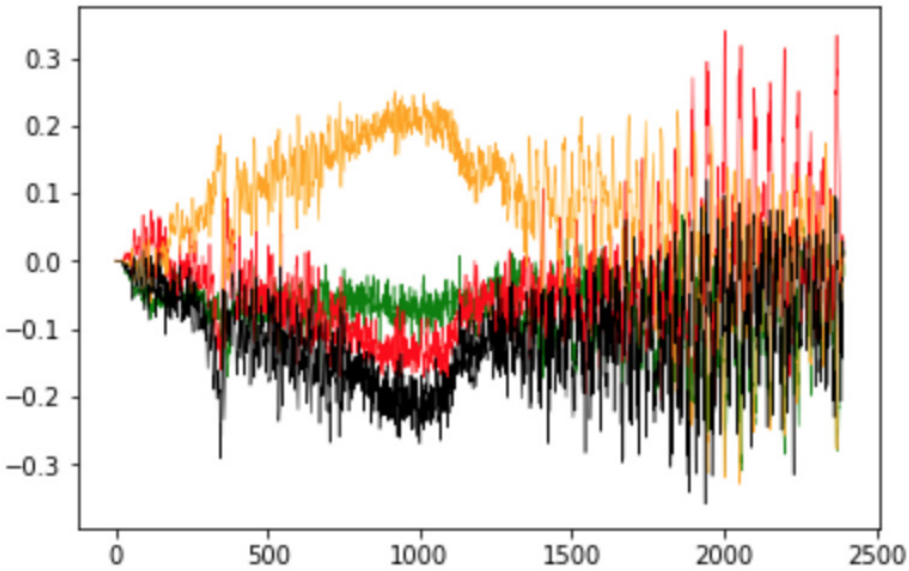


Figure 1.7: Oscillating loss. Figure from [1]

fools the discriminator and would start to map every point in the latent input space to this observation. This means that the gradient of the loss function collapses to near 0. Even if we then try to retrain the discriminator to stop it being fooled by this one point, the generator will simply find another mode that fools the discriminator, since it has already become numb to its input and therefore has no incentive to diversify its output. [1]

In this work, we focus on the mode collapse problem of GANs and try to tackle this problem of the state-of-the-art GAN from a new perspective. This work is still under peer review at a machine learning conference and most of the content is similar to [14].

Chapter 2

Improve the diversity of the generated images of GANs

2.1 Problem statement

Generative Adversarial Networks (GANs) [2] have achieved great success in generating high-fidelity images [7] and enabled a wide range of image synthesis applications [15]. However, they have a known problem of *mode collapse* i.e., the generated distribution does not capture all modes of the true distribution [16]. Therefore, synthesizing images to match the 1000-class ImageNet dataset [17] has been a grand challenge to GANs whose samples were often far less diverse than the real data. The recent class-conditional BigGAN [11] has reached an unprecedented state-of-the-art image quality and diversity on ImageNet by using large networks and batch sizes. However, interestingly, we observed that BigGAN samples from a set of $\sim$50 classes exhibit substantially lower diversity than samples from other classes. For example, generated images for daisy mostly show white flowers on green grass, but the training data includes images of daisies with a variety of colors and backgrounds (Fig. 2.1). Furthermore, samples for window screen not only have low diversity but also poor realism (see Fig. 4.5 for more low-diversity BigGAN samples). This phenomenon is intriguing given that BigGAN synthesizes photo-realistic images for many classes [11] i.e., the generator is *already capable* of painting a wide variety of images.

Why do we observe this stark contrast in BigGAN sample diversity for window screen or daisy vs. the other classes? Due to the notorious GAN training instability [16], BigGAN authors trained the generator until its performance collapsed and took the previous best snapshot as the final model. Therefore, the inferior sample diversity for a class y_c (e.g., window screen) may

be because as BigGAN training collapsed, the generator's parameters were corrupted in a way that degraded the capability of synthesizing the visual features needed for class y_c.

As it remains a mystery how the synthesis capability degraded, improving the sample diversity for a mode-collapse class is non-trivial. First, re-training BigGANs requires expensive computation—the original 256×256 model took 48 hours of training on 256 Google Cloud TPUs. On more modest hardware of 8 $\times$ V100 GPUs [18], the training is estimated to take 3–5 weeks and has not been found to match the results in [11]. Second, re-training or finetuning is likely to still cause a set of classes to collapse as we observed in the BigGAN-deep model released by [11].

2.2 Our appoach to Improve the diversity of the generated images of GANs

In this work[1], we found that, for many classes, mode collapse can be substantially ameliorated (Fig. 2.1) by only modifying the class embeddings (i.e., keeping the generator unchanged). We improved the diversity by iteratively searching for an embedding input to a pre-trained BigGAN generator that yields random samples that maximize the probability scores by a pre-trained image classifier (Fig. 2.2).

Let P be a pre-trained ImageNet classifier [19] that maps an image $\boldsymbol{x} \in \mathbb{R}^{256\times256\times3}$ onto a softmax probability distribution over 1,000 output classes.

Let G be a class-conditional generator, here a BigGAN pre-trained by [11], that takes a class embedding $\boldsymbol{c} \in \mathbb{R}^{128}$ and a latent vector $\boldsymbol{z} \in \mathbb{R}^{140}$ as inputs and outputs an image $G(\boldsymbol{c}, \boldsymbol{z}) \in \mathbb{R}^{256\times256\times3}$. The embedding matrix was learned during GAN training. In this study, we test improving sample diversity by only changing the embeddings.

Diversity regularization Intuitively, we search for an input class embedding $\boldsymbol{c}$ for the generator G such that the set of output images $\{G(\boldsymbol{c}, \boldsymbol{z}^i)\}$ is diverse with random latent vectors $\boldsymbol{z}^i \sim \mathcal{N}(0, I)$. Specifically, we encourage small changes in the latent variable to cause large changes in the output image [20] by maximizing:

[1] All code and data will be available on https://github.com/qilimk/biggan-am

(a) ImageNet images (b) BigGAN [11] (c) AM (ours)

LPIPS: 0.73 LPIPS: 0.59 LPIPS: 0.66

Figure 2.1: 256×256 BigGAN samples for some classes, here, daisy (b) are far less diverse than the real data (a). By changing *only* the class embeddings of BigGAN while keeping the latent vectors constant, our AM method (c) substantially improved the diversity, here reducing the LPIPS diversity gap by 50%. This result interestingly shows that the BigGAN generator itself was already capable of synthesizing such diverse images but the originally learned embeddings limited the diversity. See more comparison figures in Figs. 4.12—4.16.

$$\max_{\boldsymbol{c}} \; L_{\mathrm{D}}(\boldsymbol{c}) = \mathbb{E}_{\boldsymbol{z}^i, \boldsymbol{z}^j \sim \mathcal{N}(0,I)} \frac{\|\phi(G(\boldsymbol{c}, \boldsymbol{z}^i)) - \phi(G(\boldsymbol{c}, \boldsymbol{z}^j))\|}{\|\boldsymbol{z}^i - \boldsymbol{z}^j\|} \tag{2.1}$$

where $\phi(.)$ is a feature extractor. In [20], $\phi(.)$ is an identity function to encourage pixel-wise diversity. We also tested with $\phi(.)$ being outputs of the conv5 layer and the output softmax layer of AlexNet.

Via hyperparameter tuning, we found maximizing the above objective via 10 unique pairs of $(\boldsymbol{z}^i, \boldsymbol{z}^j)$ selected from $\mathcal{Z}$ to be effective (full hyperparameter details are in Sec. 3.3).

Activation maximization When a class embedding changes, it is important to keep the generated samples still realistic and in the given class. To do that, we also move the class embedding $\boldsymbol{c}$ of the generator G such that the output image $G(\boldsymbol{c}, \boldsymbol{z})$ for any random $\boldsymbol{z} \sim \mathcal{N}(0, I)$ would cause the classifier P to output a high probability for a target class $\boldsymbol{y}$ (Fig. 2.2). Formally, we maximize the following objective given a pre-defined class y_c:

$$\max_{\boldsymbol{c}} \; L_{\mathrm{AM}}(\boldsymbol{c}) = \mathbb{E}_{\boldsymbol{z} \sim \mathcal{N}(0,I)} \log P(\boldsymbol{y} = y_c \mid G(\boldsymbol{c}, \boldsymbol{z})) \tag{2.2}$$

We try to solve the above Activation Maximization (AM) problem [21] via mini-batch gradient descent. That is, we iteratively backpropagate through both the classifier P and the

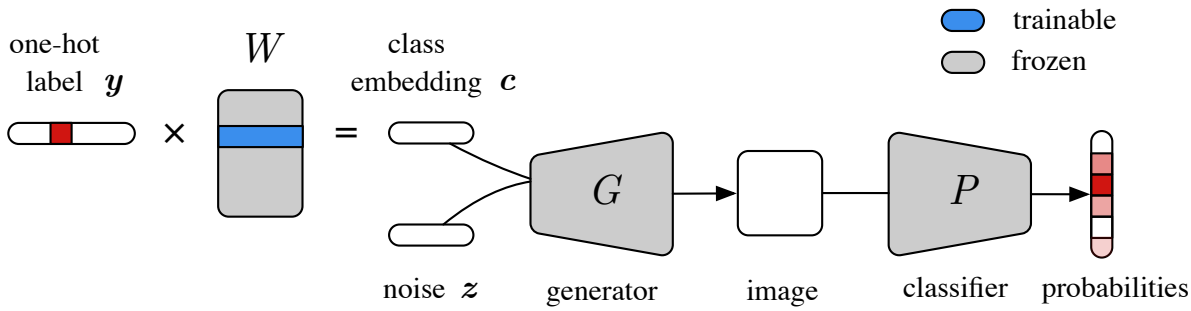


Figure 2.2: To improve the samples for a given target class represented by a one-hot vector $\boldsymbol{y}$, we iteratively take steps to find an embedding $\boldsymbol{c}$ (i.e., a row in the embedding matrix W) such that all the generated images $\{G(\boldsymbol{c}, \boldsymbol{z}^i)\}$, for different random noise vectors $\boldsymbol{z} \sim \mathcal{N}(0, I)$, are (1) classified as the target class $\boldsymbol{y}$; and (2) diverse i.e., yielding different softmax probability distributions. We backpropagate through both the frozen, pre-trained classifier P and generator G and perform gradient descent to maximize the target-class probability of the generated samples over a batch of random latent vectors $\{\boldsymbol{z}^i\}$.

generator G and change the embedding $\boldsymbol{c}$ to maximize the expectation of the log probabilities over a set $\mathcal{Z}$ of random latent vectors.

In sum, we encouraged the samples to be diverse but still remain in a target class $\boldsymbol{y}$ via the full objective function below:

$$\max_{\boldsymbol{c}} \; L_{\text{AM-D}}(\boldsymbol{c}) = L_{\text{AM}} + \lambda L_{\text{D}} \tag{2.3}$$

where λ is a hyperparameter to be tuned.

2.3 Datasets

While the generators and classifiers were pre-trained on the full 1000-class ImageNet 2012 dataset, we evaluated our methods on a subset of 50 classes (hereafter, ImageNet-50) where we qualitatively found BigGAN samples exhibit the lowest diversity. The selection of 50 classes was informed by two diversity metrics (see below) but decided by humans before the study.

2.4 Evaluation metrics

Because there is currently no single metric that captures the multi-dimensional characteristics of an image set [22], we chose a broad range of common metrics to measure the diversity and realism of samples separately.

Diversity We measured the intra-class diversity by randomly sampling 200 image pairs from an image set and computing the MS-SSIM [23] and LPIPS [24] scores for each pair. For each method, we computed a mean score across the 50 classes $\times$ 200 image pairs.

Realism To measure sample realism, we used the Inception Score (IS) with 10 splits [25], the Fréchet Inception Distance (FID) [26], and the Inception Accuracy (IA) [23]. These three metrics were computed for every set of 50,000 images = 50 classes $\times$ 1000 images.

2.5 Networks

Classifiers Our default image classifier is AlexNet [19] pre-trained on the 1000-class ImageNet 2012 dataset. Note that other ImageNet classifiers can also be used as shown in Sec. 3.3.

Generators We used two pre-trained ImageNet BigGAN generators [11], a 256×256 and a 128×128 model, released by the authors in PyTorch. For the purpose of studying diversity, all generated images in this paper were sampled from the full, non-truncated prior distribution [11].

Chapter 3

Experiments and Results

3.1 Semantically meaningful BigGAN class embeddings

We observed via t-SNE visualizations [27] that the class embeddings learned by BigGAN accurately reflect the semantics of ImageNet classes. That is, we projected 1000 class embeddings $\boldsymbol{c}^i \in \mathbb{R}^{128}$ onto a 2-D t-SNE space. Interestingly, the embeddings for the low-diversity ImageNet-50 classes are far from random, i.e., they were located in the neighborhoods of related concepts (Fig. 4.18; the daisy embedding is near other flowers and plants). The semantically meaningful t-SNE arrangements of the BigGAN class embeddings motivated us to search in the neighborhood of the original embeddings to find a new vector that yields more diverse images (see the following sections).

3.2 Adding noise to or finetuning the class embeddings did not improve diversity

Adding noise A naive approach to improving sample diversity is adding small random noise to the embedding vector of a low-diversity class. Across 50 classes, we found that adding small noise $\sim \mathcal{N}(0, 0.1)$ had negligible effects on image quality and diversity (Fig. 3.1; Noise-S) while adding larger noise $\sim \mathcal{N}(0, 0.3)$ degraded the samples on both criteria (Fig. 3.1; Noise-L). For example, daisy samples gradually turned into human-unrecognizable rubbish images as we increased the noise (Fig. 4.4).

Finetuning Another strategy to improve sample diversity is to finetune BigGANs. However, how to finetune a BigGAN to improve its sample diversity is an open question. As reported in [11], the BigGAN pre-trained model would start to degrade if one continued training it using the original hyperparameters.

To minimize the GAN training instability and compare with other approaches in this paper, we only finetuned one embedding at a time, keeping the other embeddings and all parameters in the generator and discriminator frozen.

Because [11] only released the discriminator for their 128×128 generator but not for the 256×256 model, we only finetuned the 128×128 model. For each class, we added a small amount of noise $\sim \mathcal{N}(0, 0.1)$ to the associated embedding vector and finetuned it using the original BigGAN training objective for 10 iterations until the training collapsed. Across 50 classes $\times$ 5 trials, quantitatively, finetuning did not improve the sample diversity but lowered the realism (Fig. 3.2; purple Δ vs. green $\square$).

3.3 Activation Maximization was effective in improving 256×256 sample diversity

The previous results show that modifying the embeddings following random directions (i.e., adding noise) or the gradients from BigGAN discriminators (i.e., finetuning) failed to improve sample diversity. Here, we propose to update an embedding using the gradient from an image classifier to maximize its log probabilities (Fig. 2.2).

We found two strategies to be effective: (1) searching within a small region around the original embeddings (AM-S); (2) searching within a large region around the mean embedding (AM-L).

Hyperparameters For AM-S, we randomly initialized the embedding within a Gaussian ball of radius 0.1 around the original embedding. we used a learning rate of 0.01. For AM-L, we randomly initialized the embedding around the mean of all 1,000 embeddings and used a larger learning rate of 0.1. For both settings, we maximized Eq. 2.2 using the Adam optimizer and its default hyperparameters for 200 steps. We re-sampled a set $\mathcal{Z} = \{\boldsymbol{z}^i\}_{20}$ every 20 steps. Every step, we kept the embeddings within $[-0.59, 0.61]$ by clipping. To evaluate each trial, we used the embedding from the last step and sampled 1,000 images per class. We ran 5 trials per

class with different random initializations. We used 2 to 4 $\times$ V100 GPUs for each optimization trial.

Classifiers In the preliminary experiments, We tested four ImageNet classifiers: AlexNet, Inception-v3 [28], ResNet-50 [29], and a ResNet-50 [30] that is robust to pixel-wise noise. By default, we resized the BigGAN output images to the appropriate input resolution of each classifier. With Inception-v3, we achieved an FID score that is (a) substantially better than those for the other three classifiers (Table 4.1; 30.24 vs. 48.74), and (b) similar to that of the original BigGAN (30.24 vs. 31.36). The same trends were observed with the Inception Accuracy metrics (Table 4.1). However, we did not find any substantial qualitative differences among the samples of the four treatments. Therefore, we chose AlexNet because of its fastest run time.

Results Across 50 classes $\times$ 5 trials, we found that both AM-S and AM-L produced samples of higher diversity than the original BigGAN samples. For both MS-SSIM and LPIPS, on average, our AM methods reduced the gap between the original BigGAN and the real data by $\sim$50% (Fig. 3.1a; AM-S and AM-L vs. BigGAN). For all 50 classes, we always found at least 1 out of 10 trials (i.e., from both methods combined) that yielded samples that match the real data in MS-SSIM or LPIPS scores (Fig. 3.1a; AM-max vs. ImageNet-50). The statistics also align with our qualitative observations that AM samples often contain more diverse object poses, object shapes, and backgrounds than the BigGAN samples (see Figs. 4.9–4.11).

In terms of IA and FID scores, AM samples have lower realism than BigGAN samples (Table 3.1). Given the known inflation issues with IS scores [31], our IS scores (Fig. 3.1b) suggest that AM did not improve the BigGAN sample realism. However, for some classes e.g., window screen, AM was able to turn the original rubbish images into a diverse set of recognizable samples (Fig. 4.10c).

3.4 Explicitly encouraging diversity yielded worse sample realism

Inspired by [20], here, we attempted to improve the sample diversity further by incorporating a diversity regularizer into the previous two AM-S and AM-L methods (Sec. 2.2) producing two new variants, AM-D-S and AM-D-L. We tested encouraging diversity in the (1) image

space; (2) conv5 feature space; and (3) softmax outputs of AlexNet, and found they can bias the optimization towards different interesting spaces of diversity.

While the addition of the regularizer quantitatively improved sample diversity, sample quality was considerably lower (Fig. 3.1b AM-S vs. AM-D-S and AM-L vs. AM-D-L). Similarly, the IA scores of the AM-D methods were consistently lower than those of the original AM methods (Table 3.1).

We also found that in $\sim$2% of the AM-S and AM-L trials, the optimization converged at a class embedding that yields similar images for different random latent vectors. Here, we try to improve the sample diversity further by incorporating a specific regularization term into the AM formulation (as described in Sec. 2.2).

Experiments In the preliminary experiments, we tested encouraging diversity in the (1) image space; (2) conv5 feature space; and (3) softmax outputs of AlexNet. We observed that the pixel-wise regularizer can improve the diversity of background colors (Fig. 4.1) and tends to increase the image contrast upon a high λ multiplier (Fig. 4.1c). In contrast, the impact of the conv5 diversity regularizer is less noticeable (Fig. 4.2). Encouraging diversity in the softmax output distribution can yield novel scenes e.g., growing more flowers in monarch butterfly images (Fig. 4.3c).

While each level of diversity has its own benefits for specific applications, here, we chose to perform more tests with the softmax diversity to encourage samples to be more diverse *semantically*. That is, we re-ran the AM-S and AM-L experiments with an additional softmax diversity term (Eq. 2.3) and a coefficient $\lambda = 2$ (see Fig. 4.3). We call these two AM methods with the diversity term AM-D-S and AM-D-L.

Results We found that the addition of the regularizer did not improve the diversity substantially but lowered the sample quality (Fig. 3.1b AM-S vs. AM-D-S and AM-L vs. AM-D-L). Similarly, the IA scores of the AM-D methods were consistently lower than those of the original AM methods (Table 3.1).

Method	IS (10 splits) (higher=better)	FID (lower=better)	Inception Accuracy (higher=better)	MS-SSIM (lower=better)	LPIPS (higher=better)
1. ImageNet-50 (real)	6.49 ± 0.63	N/A	0.90	0.43 ± 0.04	0.70 ± 0.08
2. BigGAN	6.03 ± 0.87	24.34	0.87	0.46 ± 0.05	0.61 ± 0.09
3. Noise-S	6.53 ± 0.86	28.75	0.82	0.46 ± 0.05	0.61 ± 0.09
4. Noise-L	7.67 ± 0.95	84.61	0.36	0.46 ± 0.05	0.49 ± 0.04
5. AM-S					
a. Best LPIPS trial	7.33 ± 0.73	40.82	0.72	0.44 ± 0.05	0.64 ± 0.08
b. Average	7.03 ± 0.71	38.39	0.74	0.44 ± 0.05	0.63 ± 0.08
6. AM-L					
a. Best LPIPS trial	7.49 ± 0.81	47.25	0.64	0.44 ± 0.04	0.65 ± 0.08
b. Average	7.22 ± 0.79	46.86	0.68	0.44 ± 0.05	0.63 ± 0.08
7. AM-D-S					
a. Best LPIPS trial	7.62 ± 0.90	45.61	0.66	0.44 ± 0.04	0.65 ± 0.08
b. Average	7.32 ± 0.80	43.78	0.68	0.44 ± 0.05	0.64 ± 0.08
8. AM-D-L					
a. Best LPIPS trial	7.58 ± 0.84	50.94	0.64	0.44 ± 0.04	0.65 ± 0.08
b. Average	7.43 ± 0.85	52.68	0.61	0.44 ± 0.05	0.64 ± 0.08

Table 3.1: We compared Activation Maximization (AM) samples with the BigGAN samples and the real ImageNet-50 images on two diversity metrics (MS-SSIM and LPIPS) and three realism metrics, Inception Score (IS), Fréchet Inception Distance (FID), and Inception Accuracy (IA). ImageNet-50 is a subset of ImageNet that contains 50 classes where BigGAN samples exhibit limited diversity (see Sec. 2.3). For each AM method, we ran 50 classes $\times$ 5 trials and reported here (a) the trial with the best LPIPS score and (b) the average across 5 runs. In MS-SSIM and LPIPS, all AM trials consistently produced more diverse samples than the BigGAN samples. However, FID and IA scores indicated that AM samples are worse in realism compared to the original BigGAN samples. See Fig. 3.1 for some graphical plots of this table.

3.5 Humans rated AM samples more diverse and similarly realistic

Because quantitative image evaluation metrics are imperfect [22], we ran a human study to compare the AM vs. original BigGAN samples. For each class, across all 20 embeddings from 5 trials $\times$ 4 methods (AM-S, AM-L, AM-D-S, and AM-D-L), we manually chose one embedding that qualitatively balanced between diversity and realism to sample images to represent our AM method in this study. As a reference, this set of AM images was more diverse and less realistic than BigGAN samples according to the quantitative metrics (Fig. 3.1; AM-human vs. BigGAN).

Experiments We created two separate online surveys for diversity and realism, respectively (see Figs. 3.3–3.6. a panel of 8×8 BigGAN images and asked participants to rate the panel that was more diverse on a scale of 1–5. A 1 or 5 indicates the left or right panel is clearly more diverse, respectively, while a 3 indicates both sets are similarly diverse. For each class,

the AM and BigGAN panels were randomly positioned on the left or right. The realism survey was a duplicate of the diversity survey except that each panel only showed 3×3 images so participants could focus more on the details.

Results For both tests, we had 52 participants who were mostly university students and do not work with machine learning or GANs. On average, AM samples were rated more diverse and similarly realistic compared to BigGAN samples. That is, AM images were given better than the neutral score of 3, i.e., 2.24 ± 0.85 in diversity and 2.94 ± 1.15 in realism. Also, AM samples were rated more diverse in 42/50 classes and more realistic in 22/50 classes. Example comparisons can be found in Figs. 4.9–4.11.

3.6 AM embeddings still capture semantics and enable realistic interpolations

While the embeddings found by our AM methods changed the generated samples entirely for many classes e.g., window screen, we observed that interpolating in the latent or embedding spaces still yields realistic intermediate samples (Fig. 3.7). See Figs. 4.22– 4.24 for more interpolation examples between $\boldsymbol{z}$ pairs and between $\boldsymbol{c}$ pairs (i.e., classes).

In addition, when projected onto a 2-D t-SNE space, the 50 embeddings found by AM still reflect class semantics like the original BigGAN embeddings (see Fig. 4.21 for side-by-side comparisons).

3.7 Generalization to a 128×128 BigGAN

To test whether our method generalizes to a different GAN at a lower resolution, we applied our AM-S method (see Sec. 3.3) to a pre-trained 128×128 BigGAN released by [18]. As in previous experiments, we ran 50 classes $\times$ 5 trials in total. To evaluate each trial, we used the last-step embedding to sample 1000 images per class.

Consistent with the results for the 256×256 resolution, here, AM-S improved the diversity over the pre-trained model on both MS-SSIM and LPIPS (Fig. 3.2a; 138k). In terms of quality, FID and IS showed a mixed result of whether AM-S sample realism is lower or higher. See Fig. 4.17 for random side-by-side image comparisons.

3.8 Generalization to different training snapshots of 128×128 BigGAN

We have shown that BigGAN sample diversity can be substantially improved by only changing the embeddings (Sec. 3.3) which indicates that the generator was actually capable of synthesizing those diverse images. Here, we test how much sample diversity and quality can be improved by AM as the BigGAN training gradually collapses, which might impair not only the embeddings but also the generator's parameters.

Experiments We took the pre-trained 128×128 BigGAN model (saved at the 138k-th iteration) and continued training it for an additional 9000 iterations using the same hyperparameters as in [18]. We applied the AM-S method using the same hyperparameters as in Sec. 3.7 to four BigGAN snapshots captured at the 140k, 142, 144k, and 146k iteration, respectively.

Results AM-S consistently improved the sample diversity of all snapshots. For some classes, AM qualitatively improved both sample diversity and quality (Figs. 3.8 and 4.6–4.8). However, the diversity and realism of both AM-S and the original BigGAN samples gradually dropped together (Fig. 3.2; AM-S vs. BigGAN). The result suggests that, as the GAN training gradually collapsed, the generator weights might have converged to a local minimum that changing the class embeddings alone is not sufficient to significantly improve the samples.

3.9 BigGAN trained on ImageNet can synthesize scene images for Places365

Our previous experiments show that the BigGAN generator pre-trained on ImageNet is able to synthesize a wider variety of images than one might expect. Here, we test whether the same ImageNet generator can synthesize images for an entirely different target dataset of Places365 [32], which contains 365 classes of scene images. For evaluation, we randomly chose 50 out of 365 classes in Places365 (hereafter, Places-50).

Mean initialization We ran the AM-L algorithm for 5 trials per class using the same hyperparameters as in Sec. 3.3 but with a ResNet-18 classifier [29] pre-trained on Places365.

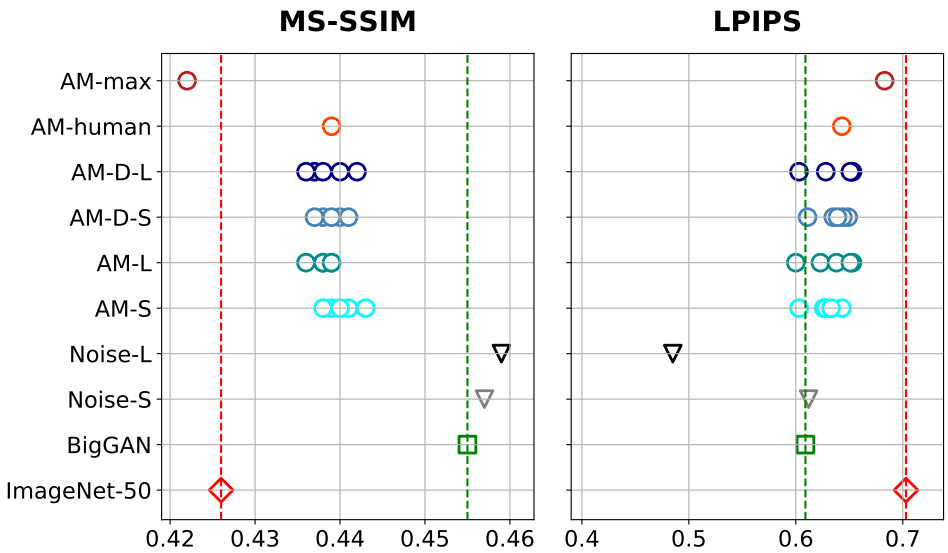
Top-5 initialization We also tested initializing from the top-5 embeddings, i.e., five class embeddings whose 10 randomly generated samples were given the highest average accuracy scores by the Places365 classifier. For example, for the hotel room class from Places365, the

embedding for quilt (from ImageNet) generates images with the highest accuracy (Fig. 3.9). We ran 5 AM-L trials where each trial was initialized with a unique embedding from the top-5.

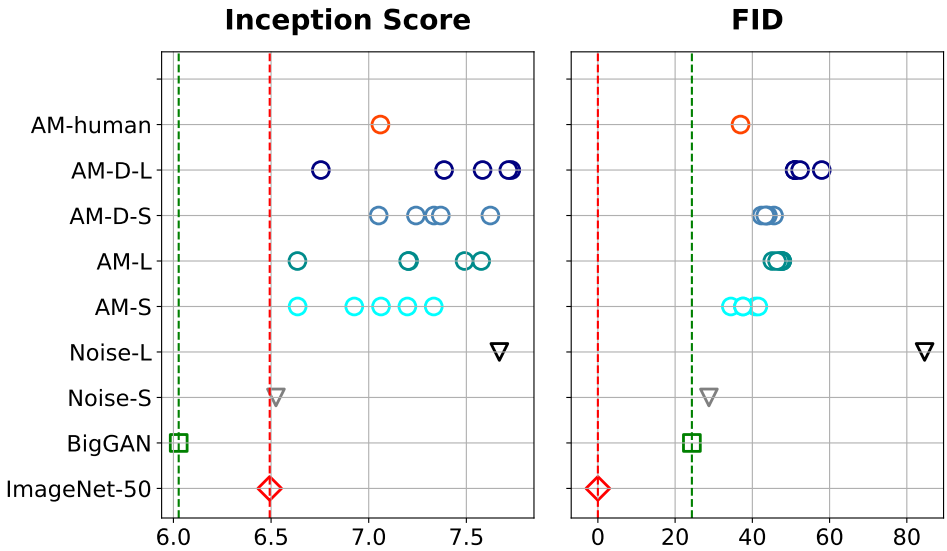
Baseline As a baseline, we used samples generated from the unmodified BigGAN conditioned on the top-1 ImageNet class embedding found from the top-5 initialization procedure described above.

Results AM-L found many class embeddings that produced plausible images for Places365 scene classes using the same ImageNet BigGAN generator. For example, to match the hotel room class which does not exist in ImageNet, AM-L synthesized bedroom scenes with lights and windows whereas the top-1 class (quilt) samples mostly consist of beds with blankets (Fig. 3.9). See Figs. 4.25, 4.26, 4.27, 4.28 for more image comparisons.

Compared to the baseline, AM-L samples have substantially higher realism in FID (41.25 vs. 53.15) and ResNet-18 accuracy scores (0.49 vs. 0.17). In terms of diversity, AM-L and the baseline performed similarly, and both were slightly worse than the real images in MS-SSIM (0.42 vs. 0.43) and LPIPS (0.65 vs. 0.70). See Table 4.2 for more detailed quantitative results.

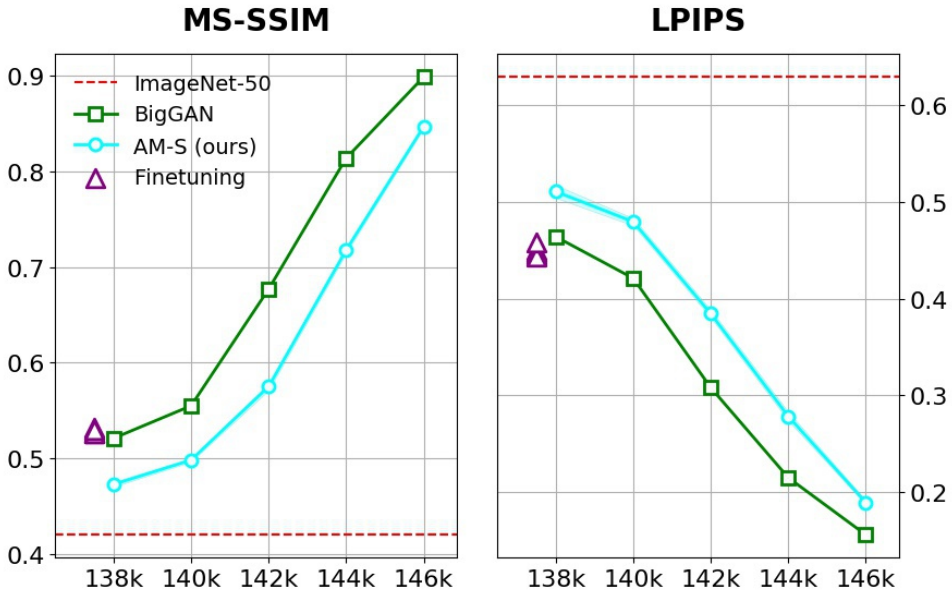


(a) Diversity comparison in MS-SSIM and LPIPS metrics.

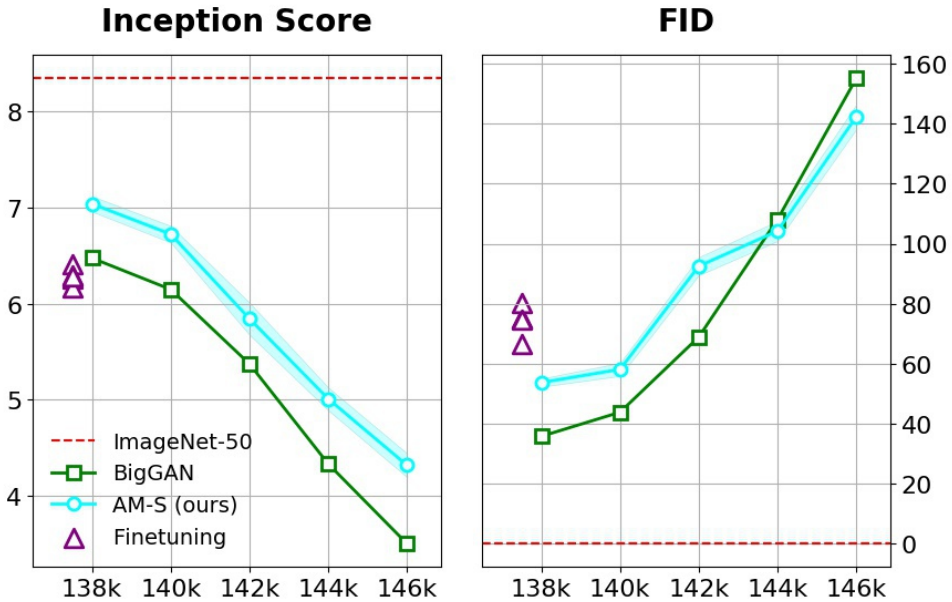


(b) Realism comparison in IS and FID metrics.

Figure 3.1: Each point in the four plots is a mean score across 50 classes from one AM optimization trial or one BigGAN model. The ultimate goal here is to close the gap between the BigGAN samples (- - - -) and the ImageNet-50 distribution (- - - -) in all four metrics. Naively adding noise degraded the embeddings in both diversity (MS-SSIM and LPIPS) and quality (IS and FID) scores i.e., the black and gray ∇ actually moved away from the red lines. Our optimization trials, on average, closed the *diversity* gap by $\sim$50%, i.e., the AM circles are halfway between the green and red dashed lines (a). However, there was a trade-off between diversity vs. quality i.e., on the IS and FID metrics, the AM circles went further away from the red line (b).



(a) Diversity comparison in MS-SSIM and LPIPS metrics.



(b) Realism comparison in IS and FID metrics.

Figure 3.2: Each point in the four plots is a mean score across 50 classes and five AM-S trials or one 128×128 BigGAN model. Finetuning the 138k snapshot neither improved the sample diversity nor realism (purple △ vs. green □). Optimizing the embeddings via AM-S consistently improved the diversity in both MS-SSIM and LPIPS (a). IS and FID metrics disagree on whether AM-S (cyan ○) sample quality is better or worse than that of the BigGAN samples. See Fig. 3.8 for a side-by-side comparison of the samples from these five snapshots.

Figure 3.3: The instruction of online survey for diversity comparison.

An example comparison

We show below two panels (LEFT and RIGHT) that contain images of Rock Beauty fish.

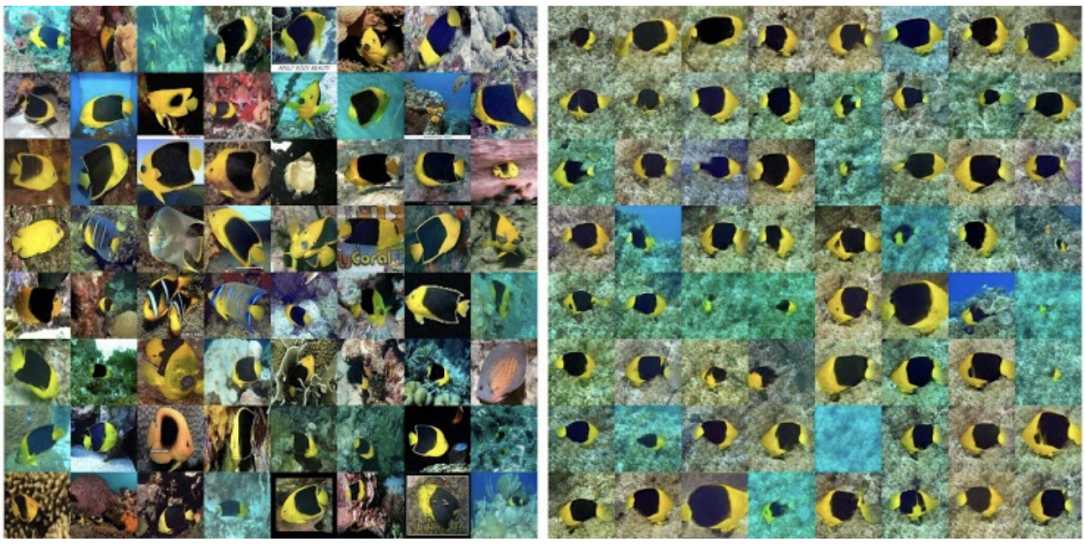
Question: Which panel is more diverse?

To answer that, for each panel, please consider the main object (e.g. colors, textures, sizes, poses) and the background (e.g. lighting, colors, scenes).

Here, the correct answer here is "1" i.e. LEFT panel is clearly more diverse because of the following observations:
- The LEFT panel contains images of rock beauty fish that are captured in different sizes, poses & lighting conditions and the backgrounds vary from corals, blue water to rocks.
- In contrast, the RIGHT panel has almost the same background and the fish are almost always horizontal to the ground and in front of the camera.

Please choose your answer and click 'NEXT' to start the formal survey.

Rock Beauty



1 2 3 4 5

LEFT panel is more diverse. ○ ○ ○ ○ ○ RIGHT panel is more diverse.

Figure 3.4: The example of online survey for diversity comparison.

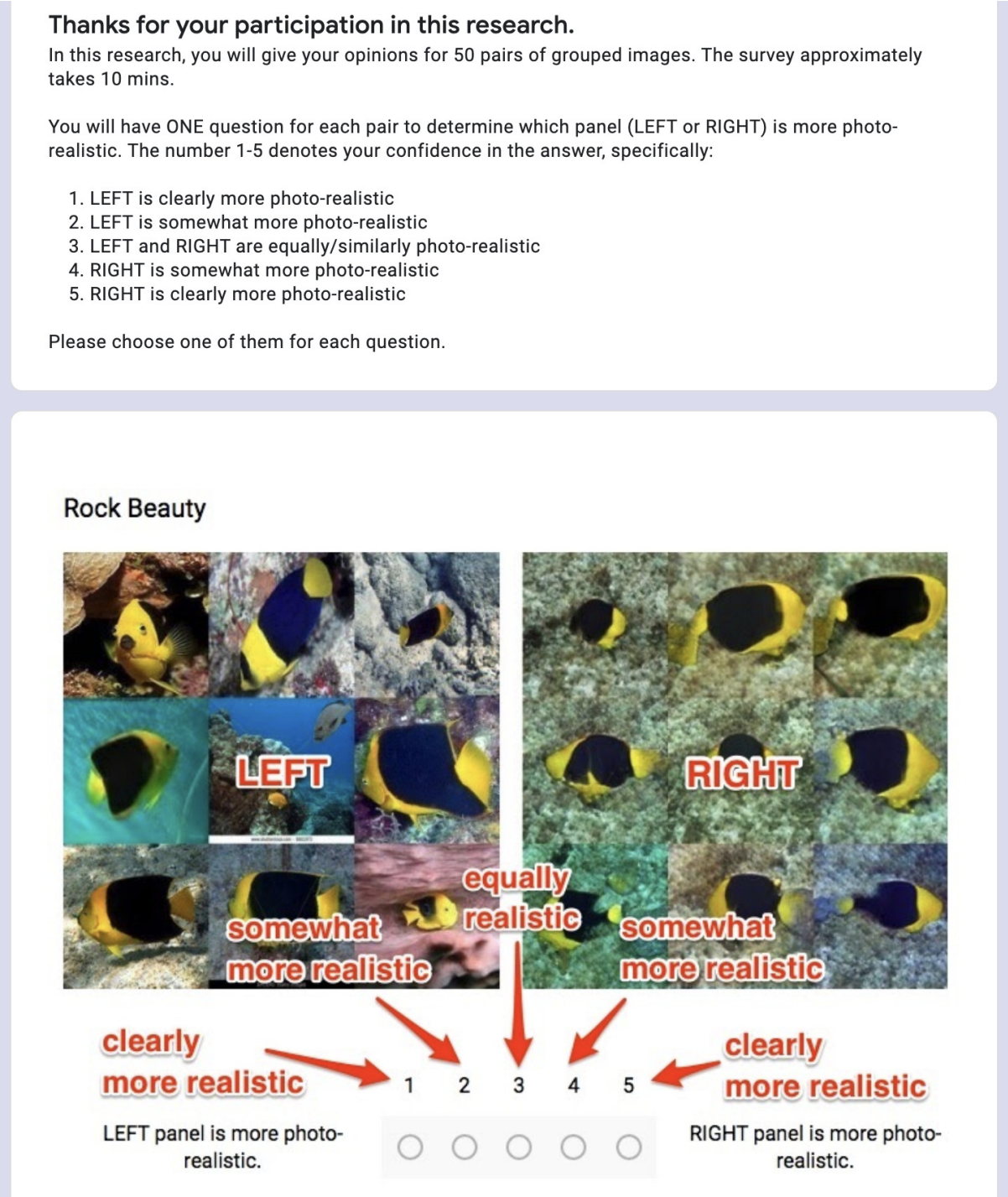


Figure 3.5: The instruction of online survey for quality comparison.

An example comparison

We show below two panels (LEFT and RIGHT) that contain images of Rock Beauty fish.

Question: Which panel is more photo-realistic?

To answer that, for each panel, please consider whether you can recognize what the object is (i.e. here, Rock Beauty fish) from the images and which panel contains images that appear like they were taken straight out of your camera.

Here, the correct answer is "2" i.e. LEFT panel is somewhat more realistic because of the following observations:
- Both panels contain images that can be recognized as photos of a type of black-and-yellow fish under water.
- For a few images in the LEFT panel, there are more details e.g. the fish eyes that help with your recognition that they are fish or even Rock Beauty fish (if you know this kind of fish).
- The LEFT images are also sharper while the RIGHT images are a bit more blurry.

Please choose your answer and click 'NEXT' to start the formal survey.

Rock Beauty

	1	2	3	4	5	
LEFT panel is more photo-realistic.	○	○	○	○	○	RIGHT panel is more photo-realistic.

Figure 3.6: The example of online survey for quality comparison.

Figure 3.7: Interpolation between a z pair in the window screen class using the original BigGAN embedding (top) yields similar and unrealistic samples. The same interpolation with the embedding found by AM (bottom) produced realistic intermediate samples between two window screen images.

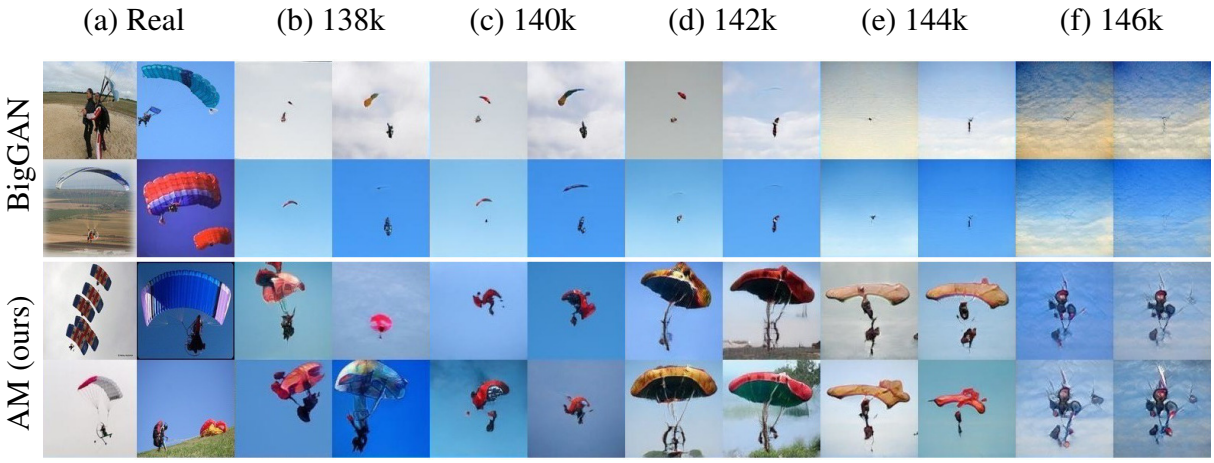


Figure 3.8: For the parachute class, the original 128×128 BigGAN samples (top panel) mostly contained tiny parachutes in the sky (b) and gradually degraded into images of only blue sky (c–f). AM (bottom panel) instead exhibited a more diverse set of close-up and far-away parachutes (b) and managed to paint the parachutes for nearly-collapsed models (e–f). The samples in this figure correspond to the five snapshots (138k—146k) reported in the quantitative comparison in Fig. 3.2. See Figs. 4.6, 4.7, 4.8 for more qualitative comparisons.

(a) Places365 images (b) Top-1 baseline (BigGAN) (c) AM-L (ours)

plaza parking meter plaza

hotel room quilt hotel room

Figure 3.9: AM-L generated plausible images for two Places365 classes, plaza (top) and hotel room (bottom), which do *not* exist in the ImageNet training set of the BigGAN generator. For example, AM-L synthesizes images of squares with buildings and people in the background for the plaza class (c) while the samples from the top-1 ImageNet class, here, parking meter, shows parking meters on the street (b). Similarly, AM-L samples for the hotel room class have unique lighting, lamps, and windows (c) that do not exist in the BigGAN samples generated using the quilt class embedding (b). The latent vectors are held constant for corresponding images in (b) and (c). See Figs. 4.25, 4.26, 4.27, and 4.28 for more side-by-side image comparisons.

Chapter 4

Discussion & Conclusion

4.1 Discussion

We showed that the low sample diversity of pre-trained GAN generators can be improved by simply changing the class embeddings without modifying the generator. Note that one could "recover" the missing modes using our AM methods and improve the sample quality further by sampling from a truncated prior distribution [11]. Compared to finetuning or re-training BigGANs from scratch, our method is more tractable even when considering the five 200-step optimization trials necessary to find a desired class embedding. There are some other researches related to our work.

4.1.1 Latent space traversal

Searching in the latent space of a GAN generator network to synthesize images is known to be effective for many tasks including (1) in-painting [33]; (2) image editing [34]; (3) creating natural adversarial examples [35]; and (4) feature visualization [36]. While all prior work in this line of research optimized the latent variable $\boldsymbol{z}$, instead optimize the class embeddings $\boldsymbol{c}$ of a class-conditional generator over a set of random $\boldsymbol{z}$ vectors.

Our approach might be most related to Plug & Play Generative Networks (PPGN) [37] in that both methods sample from the joint distribution $p_G(\boldsymbol{x}, \boldsymbol{y})$ defined by a generator and a pre-trained classifier. While [37] trained an unconditional generator that inverts the features of an ImageNet classifier, our method is generally applicable to any pre-trained class-conditional

generator. Importantly, our goal is novel—to improve the sample diversity of any pre-trained class-conditional generator (here, BigGANs) by changing its class embeddings.

4.1.2 Improving sample quality

Two methods, MH-GAN [38] and DRS [39], have recently been proposed to improve the samples of a pre-trained GAN by harnessing the discriminator to reject low-probability generated samples. However, these methods are only able to improve sample *quality*, not diversity. In addition, they assume that the discriminator is (a) available, which may not always be the case, e.g., in the official BigGAN releases [11]; and (b) optimally trained for their samplers to recover exactly the true distribution. Similar to MH-GAN and PPGN, our method is similar to a Markov chain Monte Carlo (MCMC) sampler that has no rejection steps. A major difference is that we only perform the iterative optimization *once* to update the class embedding. After a desired embedding is found, subsequent sampling of images is fast following standard GANs. In contrast, MH-GAN, DRS, and PPGN samplers often require many rejection or update steps to produce a single image.

4.1.3 Improving sample diversity

Many GAN regularization tricks have been introduced to encourage the samples to be diverse (see [40] for a survey). However, all prior methods require re-training GANs from scratch, which can be computationally expensive e.g., in the BigGAN's case. Fine-tuning GANs may be a more efficient approach [41]. However, finetuning (1) requires both the pre-trained generator and discriminator, which is not always available in practice, and (2) is subject to the known training instability issues (as in Sec. 3.2). Our method is not subject to the above issues and can be viewed as finetuning only the embedding layer but using a maximum likelihood objective instead of a GAN objective.

4.1.4 Generalization

Understanding the image synthesis capability of a trained GAN generator is an active research area. Recent findings have shown that GANs trained on a dataset of scene images contain neurons that can paint common objects such as “trees” or “doors” [42]. [43] found that BigGAN is able to perform some general image transformations such as zoom, rotate, or brightness adjustment, up to a certain limit. However, these methods optimize only the latent variable [43] or both the latent and the generator parameters [42], but not the class embeddings as we do.

4.2 Conclusion

In this work, we explore a popular and outperforming generative model which is the generative adversarial networks known as GANs. By using this framework, the researchers have developed a lot of variant versions and make GANs become the state-of-the-art generative models currently. By leveraging the power of learning data distribution implicitly, GANs can generate realistic and diverse images. However, there are some challenges such as oscillating loss, multiple hyperparameters and mode collapse during the training process of GANs. They limit the real-world applications of GANs. It would be another boom for GANs if we could tackle these problems. We find that the BigGAN class embeddings qualitatively capture class semantics (Sec. 3.1) by the observations of t-SNE for the class embedding. For example, bird classes are nearby in t-SNE visualizations (Fig. 4.18). We show that simply changing class embedding can fix the mode collapse of BigGAN and improve sample diversity for some classes. By using our framework, we make the same ImageNet generator to synthesize images for an entirely different target dataset of Places365 [32], which contains 365 classes of scene images. Here are the contributions of this work:

1. Changing only the embeddings via our method was sufficient to match the diversity (via MS-SSIM and LPIPS metrics) of the real data while keeping the BigGAN generator frozen (Sec. 3.3). A human study found that our method produced more diverse (and similarly realistic) images compared to BigGAN (Sec. 3.5).

2. Our approach improved the sample diversity for two BigGANs released by the authors at both 256×256 and 128×128 resolutions (Sec. 3.7) and some mode-collapse BigGAN snapshots (Sec. 3.8).

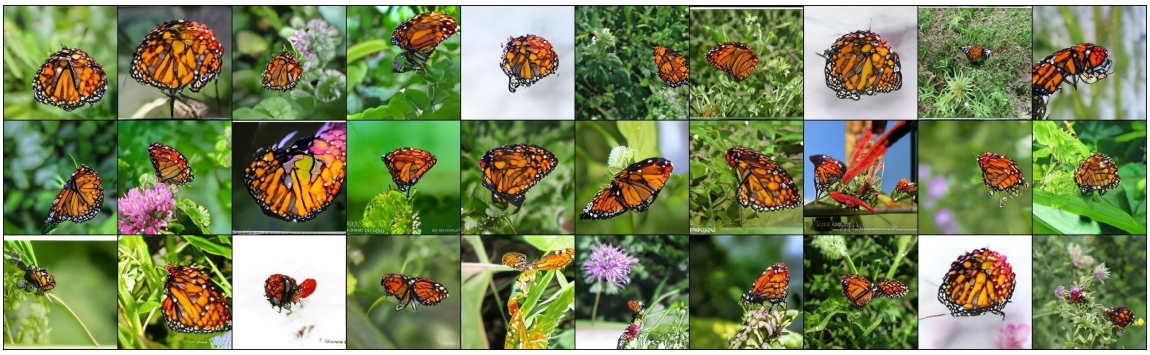
3. By updating only the embedding matrix, we can harness a BigGAN pre-trained on ImageNet to generate images matching the Places365 scene categories (Sec. 3.9).

Method	IS (10 splits) (higher=better)	FID (lower=better)	Inception Accuracy (higher=better)	MS-SSIM (lower=better)	LPIPS (higher=better)
1. ImageNet-30 (Real)	4.18 ± 0.61	n/a	0.92	0.42 ± 0.04	0.70 ± 0.08
2. BigGAN	3.71 ± 0.74	31.36	0.91	0.45 ± 0.05	0.61 ± 0.09
3. AM-L Random					
a. AlexNet	5.06 ± 0.97	46.85	0.71	0.43 ± 0.04	0.66 ± 0.08
b. Inception-v3	4.29 ± 0.56	31.62	0.87	0.44 ± 0.04	0.65 ± 0.08
c. ResNet-50	5.36 ± 0.75	47.23	0.70	0.44 ± 0.04	0.68 ± 0.09
d. Robust ResNet-50	4.59 ± 0.69	43.65	0.76	0.43 ± 0.05	0.63 ± 0.08
4. AM-D-S					
a. AlexNet	5.31 ± 0.60	48.74	0.69	0.43 ± 0.04	0.66 ± 0.08
b. Inception-v3	4.23 ± 0.51	30.24	0.88	0.44 ± 0.04	0.65 ± 0.08
c. ResNet-50	5.78 ± 1.00	52.01	0.66	0.43 ± 0.04	0.68 ± 0.08
d. Robust ResNet-50	4.51 ± 0.79	41.74	0.78	0.44 ± 0.04	0.63 ± 0.09

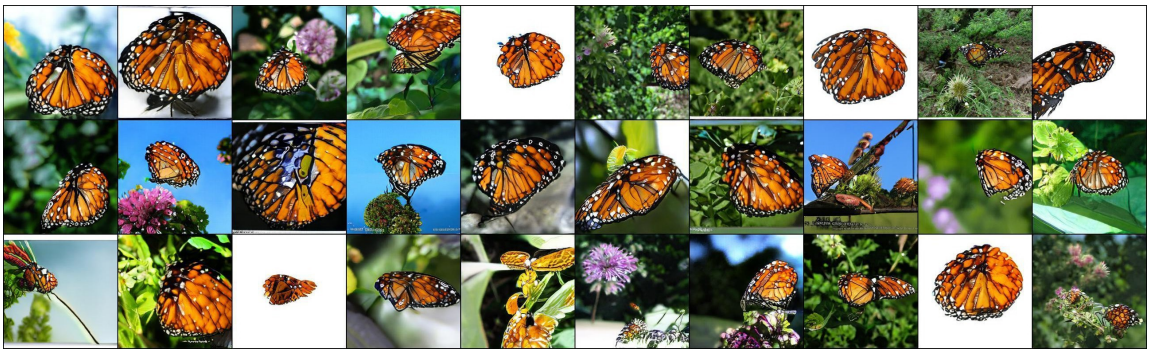
Table 4.1: A comparison of four different classifiers (a–d) across two preliminary AM settings across 30 random classes from the ImageNet-50 low-diversity dataset (see Sec. 2.3). The ImageNet-30 statistics here were computed from 30,000 images = 30 classes $\times$ 1000 images. Similarly, for BigGAN (Row 2) and AM-L and AM-D-S methods (Row 3–4), we generated 1000 256×256 samples per class. We computed the statistics for each initialization method from 5 trials, each with a different random seed. With AM-L (Sec. 3.3), we maximized the log probabilities and used a large learning rate of 0.1. With AM-D-S (Sec. 3.4), we maximized both the log probabilities and a softmax diversity regularization term, and used a small learning rate of 0.01. In sum, across both settings, AM consistently obtained the highest FID and Inception Accuracy (IA) scores with the Inception-v3 classifier (b). That is, it is possible to maximize the FID and IA scores when using Inception-v3 as the classifier in the AM formulation. However, qualitatively, we did not find the AM samples with Inception-v3 to be substantially different from the others.

Method	IS (10 splits) (higher=better)	FID (lower=better)	ResNet-18 Accuracy (higher=better)	MS-SSIM (lower=better)	LPIPS (higher=better)
1. Places-50 (real)	12.17 ± 1.01	N/A	0.57	0.42 ± 0.04	0.70 ± 0.06
2. BigGAN	8.19 ± 0.9	53.15	0.17	0.42 ± 0.05	0.66 ± 0.07
3. AM-L with Mean Initialization					
Trial 1	8.32 ± 0.89	42.38	0.51	0.43 ± 0.05	0.64 ± 0.07
Trial 2	8.39 ± 0.83	44.11	0.48	0.43 ± 0.05	0.64 ± 0.07
Trial 3	8.45 ± 0.84	42.98	0.46	0.43 ± 0.05	0.65 ± 0.07
Trial 4	7.03 ± 0.71	38.39	0.49	0.43 ± 0.05	0.64 ± 0.07
Trial 5	7.03 ± 0.71	38.39	0.49	0.43 ± 0.04	0.65 ± 0.07
Average	7.03 ± 0.51	41.25	0.49	0.43 ± 0.05	0.65 ± 0.07
4. AM-L with Top-5 Initialization					
Trial 1	8.60 ± 0.88	46.92	0.47	0.43 ± 0.05	0.65 ± 0.07
Trial 2	8.45 ± 0.81	41.09	0.52	0.43 ± 0.05	0.65 ± 0.07
Trial 3	8.13 ± 0.71	40.35	0.48	0.43 ± 0.05	0.65 ± 0.07
Trial 4	8.20 ± 0.79	43.56	0.47	0.43 ± 0.05	0.65 ± 0.07
Trial 5	8.37 ± 0.75	39.49	0.50	0.43 ± 0.05	0.65 ± 0.07
Average	8.35 ± 0.79	42.28	0.49	0.43 ± 0.05	0.65 ± 0.07

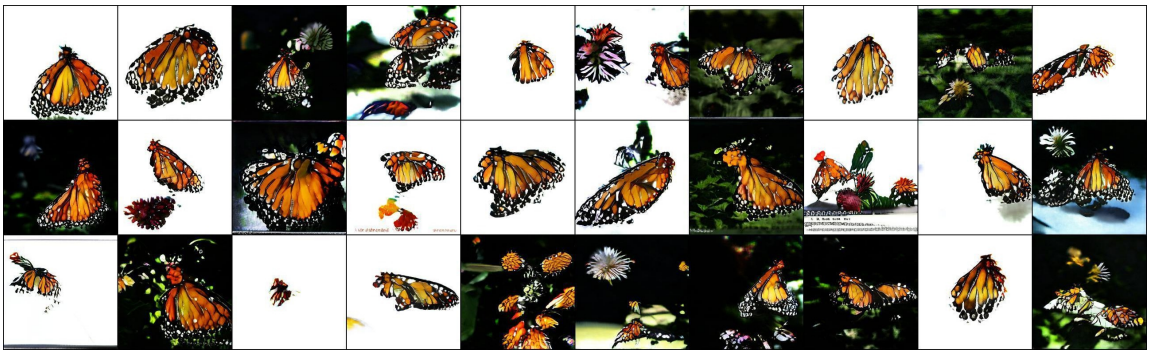
Table 4.2: A comparison of Places-50, BigGAN and AM images. We randomly chose 50 classes in Places365 (i.e., Places-50) to be the evaluation dataset for the experiments in Sec. 3.9. The Places-50 statistics here were computed from 50,000 images = 50 classes $\times$ 1000 images that were randomly selected from the training set of Places365. For BigGAN (Sec. 3.9), we chose the class embedding whose 10 random samples yielded the highest accuracy score for each target Places-50 class and generated 1000 samples per class. With AM-L mean initialization and AM-L top-5 initialization (Sec. 3.9), we maximized the log probabilities and used a large learning rate of 0.1. We found that samples from AM (Row 3-4) are of similar diversity but better quality than BigGAN samples.



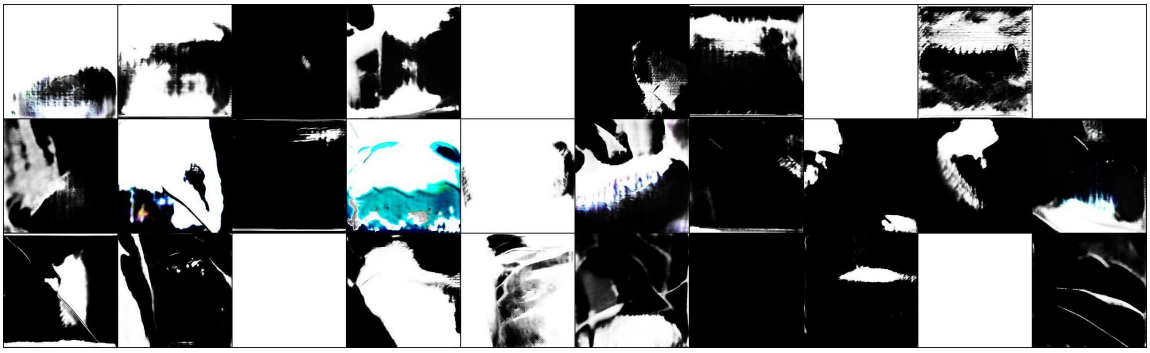
(a) AM alone without the diversity term (i.e., $\lambda = 0$ in Eq. 2.3).



(b) AM with the pixel-wise diversity term (i.e., $\lambda = 0.01$ in Eq. 2.3).

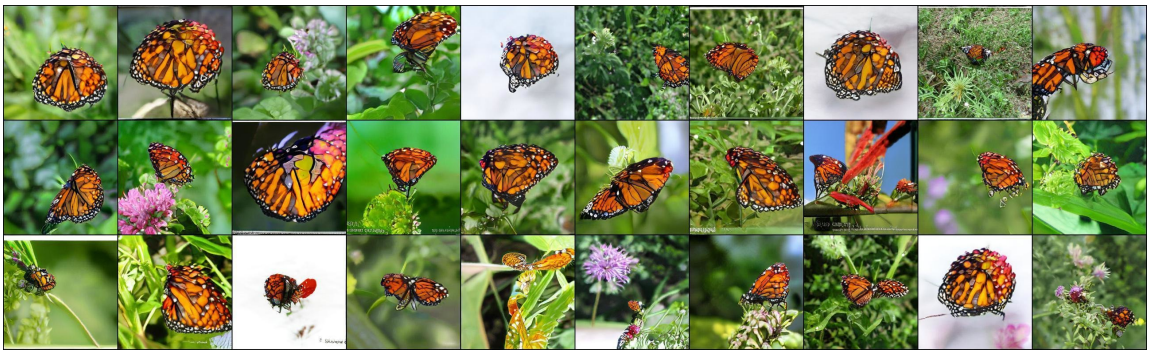


(c) AM with the pixel-wise diversity term (i.e., $\lambda = 0.1$ in Eq. 2.3).



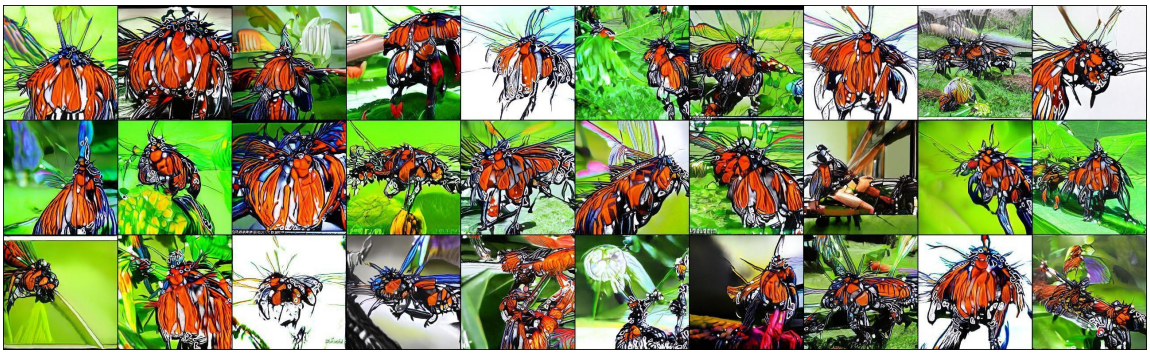
(d) AM with the pixel-wise diversity term (i.e., $\lambda = 1.0$ in Eq. 2.3).

Figure 4.1: The monarch butterfly class (323) samples generated by Activation Maximization (AM) methods when increasing the multiplier λ of a pixel-wise diversity regularization term in Eq. 2.3.

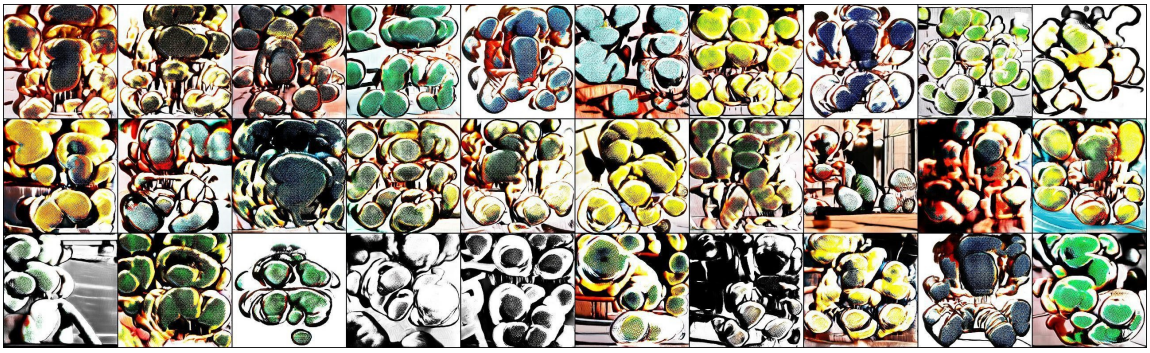


(a) AM alone without the diversity term (i.e., $\lambda = 0$ in Eq. 2.3).

(b) AM with a feature diversity term (i.e., $\lambda = 0.01$ in Eq. 2.3).

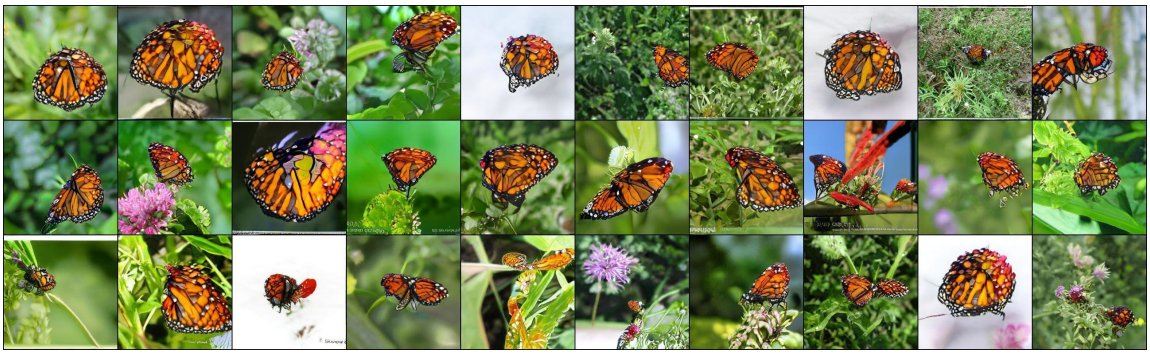


(c) AM with a feature diversity term (i.e., $\lambda = 0.1$ in Eq. 2.3).



(d) AM with a feature diversity term (i.e., $\lambda = 1.0$ in Eq. 2.3).

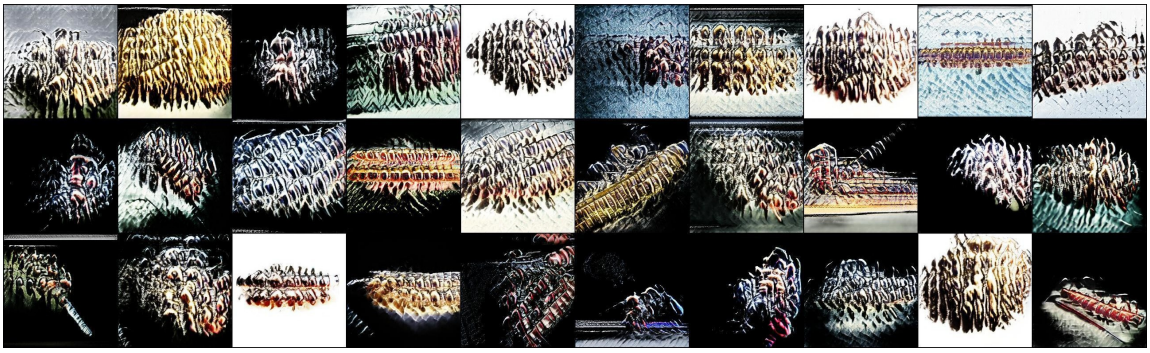
Figure 4.2: The monarch butterfly class (323) samples generated by Activation Maximization (AM) methods when increasing the multiplier λ of a conv5 feature diversity regularization term in Eq. 2.3.



(a) AM alone without the diversity term (i.e., $\lambda = 0$ in Eq. 2.3).

(b) AM with a softmax diversity term (i.e., $\lambda = 2$ in Eq. 2.3).

(c) AM with a softmax diversity term (i.e., $\lambda = 10$ in Eq. 2.3).



(d) AM with a softmax diversity term (i.e., $\lambda = 100$ in Eq. 2.3).

Figure 4.3: The monarch butterfly class (323) samples generated by Activation Maximization (AM) methods when increasing the multiplier λ of a softmax probability diversity regularization term in Eq. 2.3.

(a) BigGAN samples generated with the original daisy class embedding (no noise).

(b) BigGAN samples generated with the daisy class embedding $\boldsymbol{c}' = \boldsymbol{c} + \epsilon$ where noise $\epsilon \sim \mathcal{N}(0, 0.1)$.

(c) BigGAN samples generated with the daisy class embedding $\boldsymbol{c}' = \boldsymbol{c} + \epsilon$ where noise $\epsilon \sim \mathcal{N}(0, 0.3)$.

(d) BigGAN samples generated with the daisy class embedding $\boldsymbol{c}' = \boldsymbol{c} + \epsilon$ where noise $\epsilon \sim \mathcal{N}(0, 0.5)$.

Figure 4.4: BigGAN samples when increasing the amount of noise added to the original daisy class embedding vector. That is, four panels (a–d) are generated using the same set of 30 latent vectors $\{\boldsymbol{z}^i\}_{30}$ but with a different class embedding $\boldsymbol{c}'$.

(a) ImageNet images (b) BigGAN samples [11]

(a) Samples from the window screen class (904).

(b) Samples from the manhole cover class (640).

(c) Samples from the greenhouse class (580).

(d) Samples from the cardoon class (946).

Figure 4.5: Example mode-collapse classes from the ImageNet-50 subset where BigGAN samples (right) exhibit substantially lower diversity compared to the real data (left).

(a) ImageNet samples from the parachute class.

(b) BigGAN samples (left) and AM samples (right), both generated using the BigGAN 138k snapshot.

(c) BigGAN samples (left) and AM samples (right), both generated using the BigGAN 140k snapshot.

(d) BigGAN samples (left) and AM samples (right), both generated using the BigGAN 142k snapshot.

(e) BigGAN samples (left) and AM samples (right), both generated using the BigGAN 144k snapshot.

(f) BigGAN samples (left) and AM samples (right), both generated using the BigGAN 146k snapshot.

Figure 4.6: Applying our AM method to 5 different 128×128 BigGAN training snapshots (b–f) yielded samples (right) that qualitatively are more diverse and recognizable to be from the parachute class compared to the original BigGAN samples (left). While the original BigGAN samples are almost showing only the blue sky (d–f), AM samples show large and colorful parachutes.

(a) ImageNet samples from the pickelhaube class.

(b) BigGAN samples (left) and AM samples (right), both generated using the BigGAN 138k snapshot.

(c) BigGAN samples (left) and AM samples (right), both generated using the BigGAN 140k snapshot.

(d) BigGAN samples (left) and AM samples (right), both generated using the BigGAN 142k snapshot.

(e) BigGAN samples (left) and AM samples (right), both generated using the BigGAN 144k snapshot.

(f) BigGAN samples (left) and AM samples (right), both generated using the BigGAN 146k snapshot.

Figure 4.7: The same figure as Fig. 4.6 but for the pickelhaube class (715).

(a) ImageNet samples from the digital clock class.

(b) BigGAN samples (left) and AM samples (right), both generated using the BigGAN 138k snapshot.

(c) BigGAN samples (left) and AM samples (right), both generated using the BigGAN 140k snapshot.

(d) BigGAN samples (left) and AM samples (right), both generated using the BigGAN 142k snapshot.

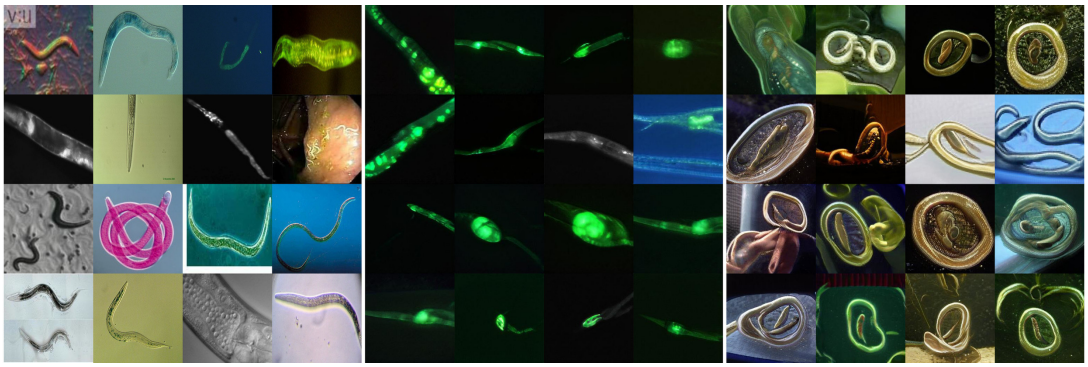
(e) BigGAN samples (left) and AM samples (right), both generated using the BigGAN 144k snapshot.

(f) BigGAN samples (left) and AM samples (right), both generated using the BigGAN 146k snapshot.

Figure 4.8: The same figure as Fig. 4.6 but for the digital clock class (530).

(a) ImageNet (b) BigGAN [11] (c) AM (ours)

(a) Samples from the flatworm class (110).



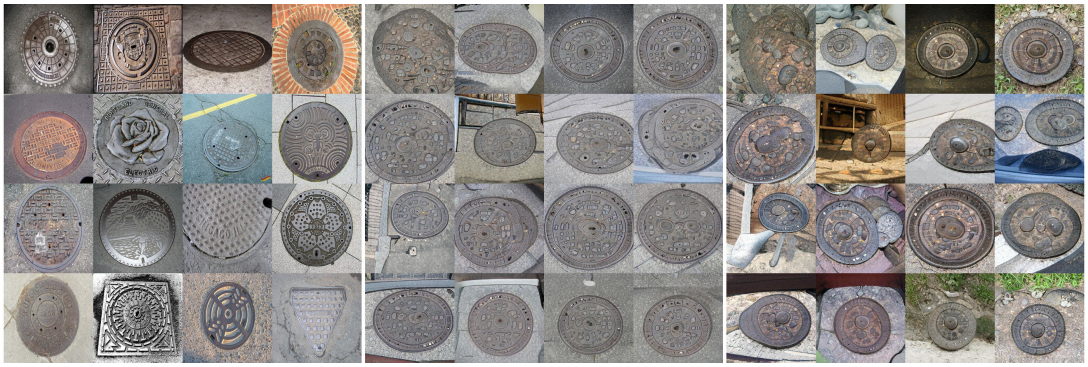
(b) Samples from the nematode class (111).

(c) Samples from the brass class (458).

(d) Samples from the greenhouse class (580).

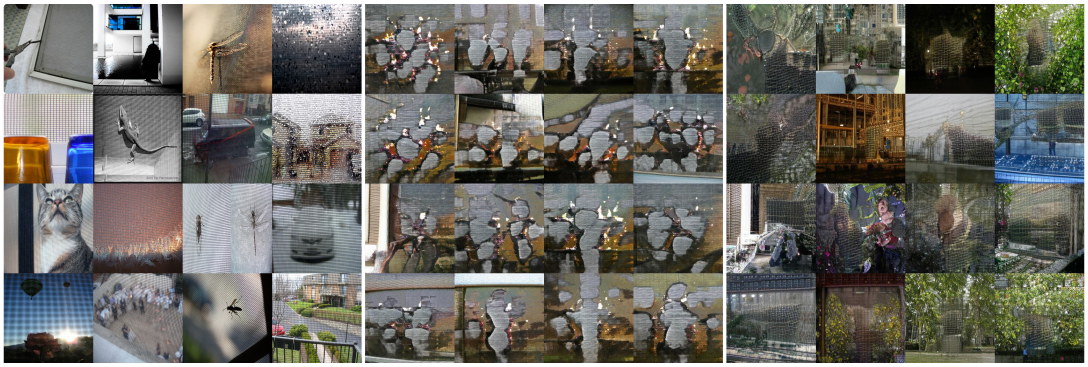
Figure 4.9: A comparison between the 256×256 samples from the ImageNet training set (a), the original BigGAN model (b), and our AM method (c) for four ImageNet-50 low-diversity classes.AM samples (c) are of similar quality but higher diversity than the original BigGAN samples (b).

(a) ImageNet (b) BigGAN [11] (c) AM (ours)



(a) Samples from the manhole cover class (640).

(b) Samples from the spider web class (815).



(c) Samples from the window screen class (904).

(d) Samples from the cardoon class (946).

Figure 4.10: A comparison between the 256×256 samples from the ImageNet training set (a), the original BigGAN model (b), and our AM method (c) for four ImageNet-50 low-diversity classes.AM samples (c) are of similar quality but higher diversity than the original BigGAN samples (b).

(a) ImageNet (b) BigGAN [11] (c) AM (ours)

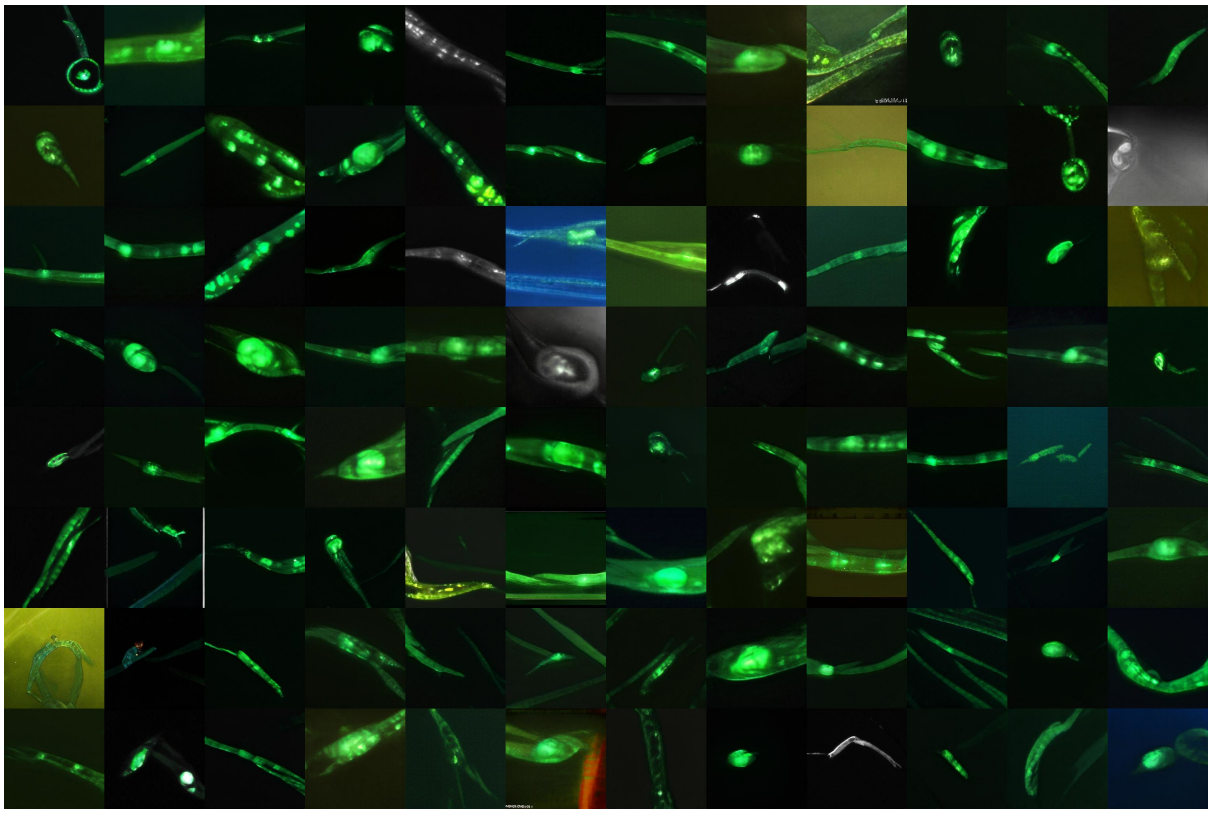
(a) Samples from the pineapple class (953).

(b) Samples from the custard apple class (956).

(c) Samples from the carbonara class (959).

(d) Samples from the pizza class (963).

Figure 4.11: A comparison between the 256×256 samples from the ImageNet training set (a), the original BigGAN model (b), and our AM method (c) for four ImageNet-50 low-diversity classes. AM samples (c) are of similar quality but higher diversity than the original BigGAN samples (b).



(a) Samples from BigGAN.

(b) Samples from AM.

Figure 4.12: A comparison between the 256×256 samples from the original BigGAN model (a), and our AM method (b) for the nematode class (111). AM samples (b) are of similar quality but higher diversity than the original BigGAN samples (a).

(a) Samples from BigGAN.

(b) Samples from AM.

Figure 4.13: A comparison between the 256×256 samples from the original BigGAN model (a), and our AM method (b) for the brass class (458). AM samples (b) are of similar quality but higher diversity than the original BigGAN samples (a).

(a) Samples from BigGAN.

(b) Samples from AM.

Figure 4.14: A comparison between the 256×256 samples from the original BigGAN model (a), and our AM method (b) for the greenhouse class (580). AM samples (b) are of similar quality but higher diversity than the original BigGAN samples (a).

(a) Samples from BigGAN.

(b) Samples from AM.

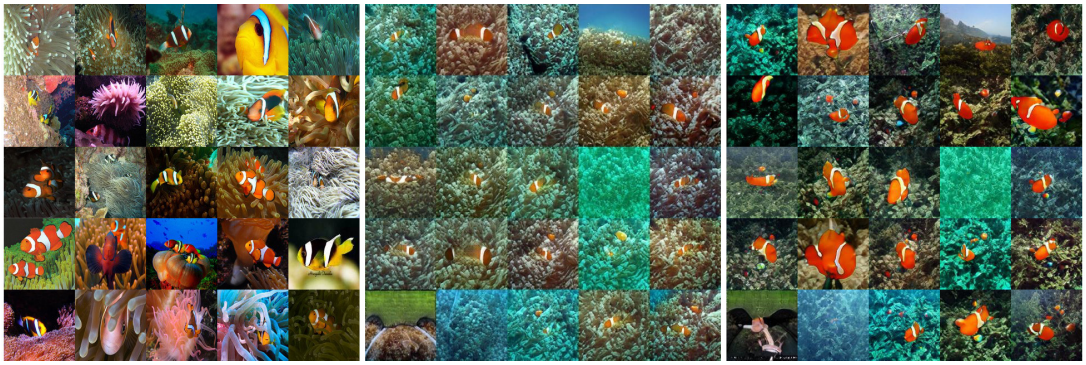
Figure 4.15: A comparison between the 256×256 samples from the original BigGAN model (a), and our AM method (b) for the window screen class (904). AM samples (b) are both of higher quality and higher diversity than the original BigGAN samples (a).

(a) Samples from BigGAN.

(b) Samples from AM.

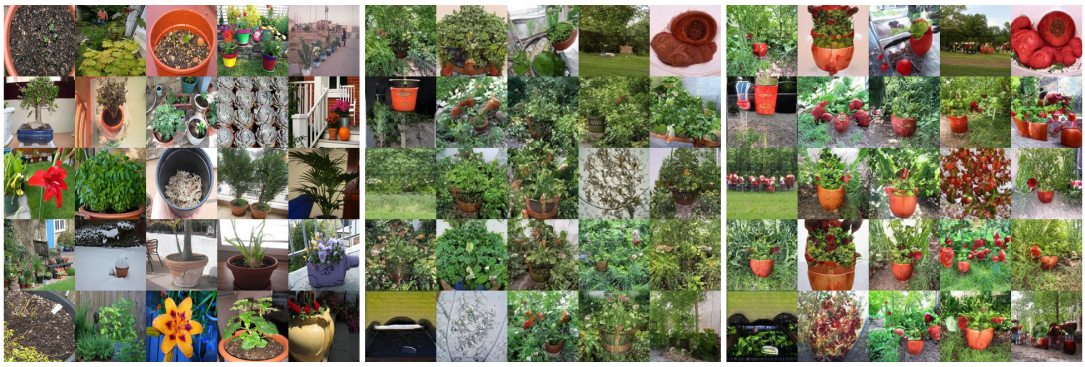
Figure 4.16: A comparison between the 256×256 samples from the original BigGAN model (a), and our AM method (b) for the daisy class (985). AM samples (b) are of similar quality but higher diversity than the original BigGAN samples (a).

(a) ImageNet (b) BigGAN [11] (c) AM (ours)



(a) Samples from the anemone fish class (393).

(b) Samples from the odometer class (685).



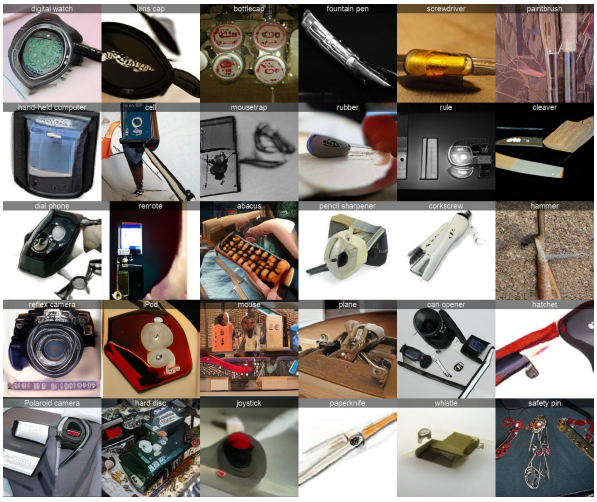
(c) Samples from the flowerpot class (738).

(d) Samples from the consomme class (925).

Figure 4.17: A comparison between the 128×128 samples from the ImageNet training set (a), the original BigGAN model (b), and our AM method (c) for four ImageNet-50 low-diversity classes. AM samples (c) are of similar quality but higher diversity than the original BigGAN samples (b).

(a) Most dogs are located nearby.

(b) Most of birds are located nearby.



(c) Man-made tools are located nearby.

Figure 4.18: Three zoom-in panels cropped out from the t-SNE visualization of 1000 original BigGAN class embeddings in Fig. 4.19. The BigGAN class embeddings are arranged semantically meaningful in the 2-D t-SNE visualization.

Figure 4.19: A 25 × 40 t-SNE 2-D visualization for 1000 original BigGAN class embeddings. At each t-SNE grid, we show a random BigGAN sample for the corresponding class. See https://drive.google.com/open?id=1Jm1sUs1k45xmP71y2o4yYSNzhID8qadB for the high-resolution version of this figure.

Figure 4.20: The same figure as Fig. 4.19 except that here we replace the 50 original BigGAN embeddings for the ImageNet-50 classes with the 50 embeddings found by AM (the highlighted cells). See https://drive.google.com/open?id=1i77bItzL_tM9S8nZ7E58EAUtTb1DFSLL for a high-resolution version of this figure.

(a) Original BigGAN embeddings. (b) Embeddings found by AM.

Figure 4.21: After modifying the 50 embeddings via AM, wre-plotted the t-SNE visualization for the entire 1000 classes. W color-code each class here with a unique border color. The arrangement of the original embeddings (left) are similar to that of the AM embeddings (right). For example, the daisy and spider web were nearby before (left) and also after AM modifications (right). In total, there are 21 classes that appear in both panels here. For each class, here, we show a random image i.e., the original BigGAN samples for the left panel and the samples generated by the AM embeddings. The left (a) and right panels (b) are crops from the Figs. 4.19 and 4.20, respectively.

(a) Interpolation in the embedding space between seaurchin (leftmost) and German shepherd (rightmost).

(b) Interpolation in the embedding space between honeycomb (leftmost) and junco bird (rightmost).

(c) Interpolation in the embedding space between hot pot (leftmost) and cheeseburger (rightmost).

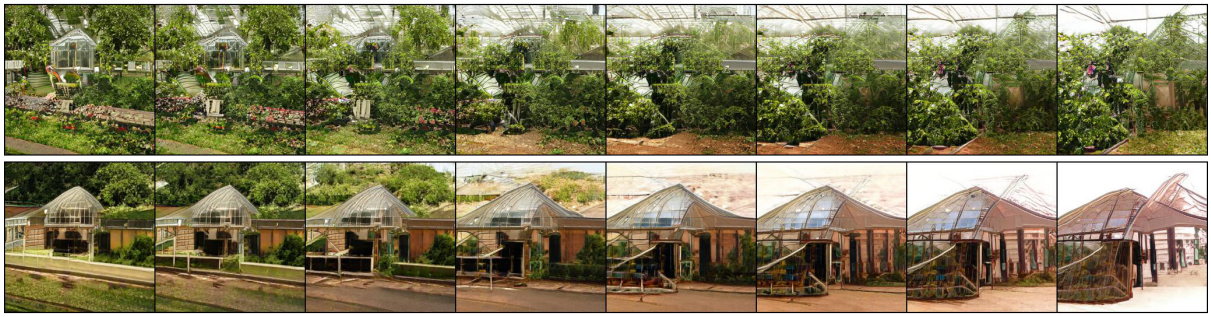
Figure 4.22: The interpolation samples between $\boldsymbol{c}$ class-embedding pairs with latent vectors $\boldsymbol{z}$ held constant. In each panel, the top row shows the interpolation between two original 256×256 BigGAN embeddings while the bottom row shows the interpolation between an embedding found by AM (leftmost) and the original BigGAN embedding (right). In sum, the interpolation samples with the AM embeddings (bottom panels) appear to be similarly plausible as the original BigGAN interpolation samples (top panels).

(a) Interpolation in the embedding space between window screen (leftmost) and water tower (rightmost).

(b) Interpolation in the embedding space between espresso (leftmost) and pop bottle (rightmost).

(c) Interpolation in the embedding space between agaric (leftmost) and bolete (rightmost).

Figure 4.23: The interpolation samples between $\boldsymbol{c}$ class-embedding pairs (from related ImageNet classes e.g., agaric and bolete are both mushrooms) with latent vectors $\boldsymbol{z}$ held constant. In each panel, the top row shows the interpolation between two original 256×256 BigGAN embeddings while the bottom row shows the interpolation between an embedding found by AM (leftmost) and the original BigGAN embedding (right). In sum, the interpolation samples with the AM embeddings (bottom panels) appear to be similarly plausible as the original BigGAN interpolation samples (top panels).



(a) Interpolation in the latent space between two z vectors with the same greenhouse class embedding.

(b) Interpolation in the latent space between two z vectors with the same window screen class embedding.

(c) Interpolation in the latent space between two z vectors with the same espresso class embedding.

(d) Interpolation in the latent space between two z vectors with the same daisy flower class embedding.

Figure 4.24: The interpolation samples between z latent-vector pairs with the same class embeddings. The z-interpolation samples with the AM embeddings (bottom panels) appear to be similarly plausible as the original BigGAN interpolation samples (top panels). For the window screen class (b), AM recovered the human-unrecognizable BigGAN samples into a plausible interpolation between two scenes of windows.

Figure 4.25: A comparison between the 256×256 samples from the Places365 training set (a), the BigGAN samples generated for the ImageNet class whose 10 random samples were given the highest accuracy for the target class in Places365 (b), and our AM samples (c). AM samples (c) are of similar diversity but better quality than the original BigGAN samples (b).

Figure 4.26: The same figure as Fig. 4.25 but for four different classes. While the ImageNet axolotl class samples were given the highest accuracy (bottom panel), they are qualitatively more different from the real jacuzzi images compared to the AM samples which shows the bathtubs.

(a) Places365 (b) BigGAN on ImageNet (c) AM (ours)

Figure 4.27: The same figure as Fig. 4.25 but for four different classes. In the bottom panel, while the BigGAN samples are dock images that contain mostly ships whereas AM samples show more bridges that resemble the real pier samples in Places365.

(a) Places365 (b) BigGAN on ImageNet (c) AM (ours)

Figure 4.28: The same figure as Fig. 4.25 but for four different classes. For the baseball stadium, the top-1 ImageNet class is scoreboard (b), an object commonly found in stadiums. However, the AM samples are more similar to the images from Places365, which often do not contain scoreboards (a vs. c).

References

[1] David Foster. *Generative deep learning: teaching machines to paint, write, compose, and play*. OReilly, 2019.

[2] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *NIPS*, 2014.

[3] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

[4] Michael A. Alcorn, Qi Li, Zhitao Gong, Chengfei Wang, Long Mai, Wei-Shinn Ku, and Anh Nguyen. Strike (with) a pose: Neural networks are easily fooled by strange poses of familiar objects. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[5] Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014.

[6] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

[7] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of GANs for improved quality, stability, and variation. In *International Conference on Learning Representations*, 2018.

[8] Andrei A Rusu, Neil C Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. Progressive neural networks. *arXiv preprint arXiv:1606.04671*, 2016.

[9] Han Zhang, Ian Goodfellow, Dimitris Metaxas, and Augustus Odena. Self-attention generative adversarial networks. *arXiv preprint arXiv:1805.08318*, 2018.

[10] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. *arXiv preprint arXiv:1802.05957*, 2018.

[11] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale GAN training for high fidelity natural image synthesis. In *International Conference on Learning Representations*, 2019.

[12] Mario Lucic, Michael Tschannen, Marvin Ritter, Xiaohua Zhai, Olivier Bachem, and Sylvain Gelly. High-fidelity image generation with fewer labels. *arXiv preprint arXiv:1903.02271*, 2019.

[13] Jeff Donahue and Karen Simonyan. Large scale adversarial representation learning. In *Advances in Neural Information Processing Systems*, pages 10541–10551, 2019.

[14] Qi Li, Long Mai, and Anh Nguyen. Improving sample diversity of a pre-trained, class-conditional gan by changing its class embeddings. *arXiv preprint arXiv:1910.04760*, 2019.

[15] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1125–1134, 2017.

[16] Martín Arjovsky and Léon Bottou. Towards principled methods for training generative adversarial networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*, 2017.

[17] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 248–255. IEEE, 2009.

[18] Andrew Brock. ajbrock/biggan-pytorch: The author's officially unofficial pytorch biggan implementation. `https://github.com/ajbrock/BigGAN-PyTorch`, 2019. (Accessed on 07/25/2019).

[19] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[20] Dingdong Yang, Seunghoon Hong, Yunseok Jang, Tiangchen Zhao, and Honglak Lee. Diversity-sensitive conditional generative adversarial networks. In *International Conference on Learning Representations*, 2019.

[21] Anh Nguyen, Alexey Dosovitskiy, Jason Yosinski, Thomas Brox, and Jeff Clune. Synthesizing the preferred inputs for neurons in neural networks via deep generator networks. In *Advances in Neural Information Processing Systems*, pages 3387–3395, 2016.

[22] Ali Borji. Pros and cons of gan evaluation measures. *Computer Vision and Image Understanding*, 179:41–65, 2019.

[23] Augustus Odena, Christopher Olah, and Jonathon Shlens. Conditional image synthesis with auxiliary classifier gans. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 2642–2651. JMLR. org, 2017.

[24] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 586–595, 2018.

[25] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. In *Advances in neural information processing systems*, pages 2234–2242, 2016.

[26] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *Advances in Neural Information Processing Systems*, pages 6626–6637, 2017.

[27] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605, 2008.

[28] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016.

[29] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[30] Logan Engstrom, Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Brandon Tran, and Aleksander Madry. Learning perceptually-aligned representations via adversarial robustness. *arXiv preprint arXiv:1906.00945*, 2019.

[31] Shane Barratt and Rishi Sharma. A note on the inception score. *arXiv preprint arXiv:1801.01973*, 2018.

[32] Bolei Zhou, Agata Lapedriza, Aditya Khosla, Aude Oliva, and Antonio Torralba. Places: A 10 million image database for scene recognition. *IEEE transactions on pattern analysis and machine intelligence*, 40(6):1452–1464, 2017.

[33] Raymond A Yeh, Chen Chen, Teck Yian Lim, Alexander G Schwing, Mark Hasegawa-Johnson, and Minh N Do. Semantic image inpainting with deep generative models. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5485–5493, 2017.

[34] Jun-Yan Zhu, Philipp Krähenbühl, Eli Shechtman, and Alexei A Efros. Generative visual manipulation on the natural image manifold. In *European Conference on Computer Vision*, pages 597–613. Springer, 2016.

[35] Zhengli Zhao, Dheeru Dua, and Sameer Singh. Generating natural adversarial examples. In *International Conference on Learning Representations*, 2018.

[36] Anh Nguyen, Jason Yosinski, and Jeff Clune. Understanding neural networks via feature visualization: A survey. *arXiv preprint arXiv:1904.08939*, 2019.

[37] Anh Nguyen, Jeff Clune, Yoshua Bengio, Alexey Dosovitskiy, and Jason Yosinski. Plug & play generative networks: Conditional iterative generation of images in latent space. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4467–4477, 2017.

[38] Ryan Turner, Jane Hung, Eric Frank, Yunus Saatchi, and Jason Yosinski. Metropolis-Hastings generative adversarial networks. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 6345–6353, Long Beach, California, USA, 09–15 Jun 2019. PMLR.

[39] Samaneh Azadi, Catherine Olsson, Trevor Darrell, Ian Goodfellow, and Augustus Odena. Discriminator rejection sampling. In *International Conference on Learning Representations*, 2019.

[40] Zhengwei Wang, Qi She, and Tomas E Ward. Generative adversarial networks: A survey and taxonomy. *arXiv preprint arXiv:1906.01529*, 2019.

[41] Yaxing Wang, Chenshen Wu, Luis Herranz, Joost van de Weijer, Abel Gonzalez-Garcia, and Bogdan Raducanu. Transferring gans: generating images from limited data. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 218–234, 2018.

[42] David Bau, Jun-Yan Zhu, Hendrik Strobelt, Bolei Zhou, Joshua B. Tenenbaum, William T. Freeman, and Antonio Torralba. Visualizing and understanding generative adversarial networks. In *International Conference on Learning Representations*, 2019.

[43] Ali Jahanian, Lucy Chai, and Phillip Isola. On the"steerability" of generative adversarial networks. *arXiv preprint arXiv:1907.07171*, 2019.