Data-Efficient GAN Training Beyond (Just) Augmentations: A Lottery Ticket Perspective

Anonymous Author(s)
Affiliation
Address
email

Abstract

Training generative adversarial networks (GANs) with limited real image data generally results in deteriorated performance and collapsed models. To conquer this challenge, we are inspired by the latest observation of [1, 2], that one can discover independently trainable and highly sparse subnetworks (a.k.a., lottery tickets) from GANs. Treating this as an inductive prior, we suggest a brand-new angle towards data-efficient GAN training: by first identifying the lottery ticket from the original GAN using the small training set of real images; and then focusing on training that sparse subnetwork by re-using the same set. Both steps have lower complexity and are more data-efficient to train. We find our coordinated framework to offer orthogonal gains to existing real image data augmentation methods [3, 4], and we additionally offer a new feature-level augmentation that can be applied together with them. Comprehensive experiments endorse the effectiveness of our proposed framework, across various GAN architectures (SNGAN, BigGAN, and StyleGAN-V2) and diverse datasets (CIFAR-10, CIFAR-100, Tiny-ImageNet, and ImageNet). Our training framework also displays powerful few-shot generalization ability, i.e., generating high-fidelity images by training from scratch with just 100 real images, without any pre-training.

1 Introduction

Generative adversarial networks (GANs) have achieved breakthroughs [5, 6, 7, 8, 9, 10, 11, 12, 13, 14] over the past few years, blessed by the quantity, diversity and high quality of natural images available in the general domain. However, for specific application domains, it could become challenging or even infeasible to collect a sufficiently large-scale dataset, due to various constraints on the imaging expense, subject type, image quality, privacy, copyright status, and more. That prohibits GANs' broader applications in these domains, e.g., for generating synthetic training data [15]. Examples of such domains include medical images, images from scientific experiments, images of rare species, or photos of a specific person or landmark. Eliminating the need of immense datasets for GAN training is highly demanded for those scenarios. Naively training GAN with scarce samples leads to overconfident discriminators that overfit the small training data [16, 17, 3, 4]; it usually ends up with training divergence and drastic performance degradation (evidenced later in Figure 2).

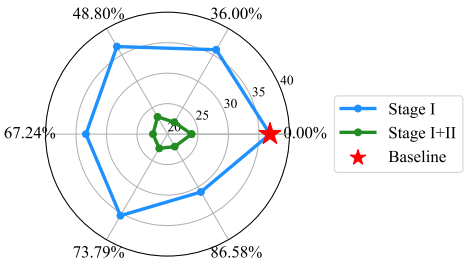


Figure 1: FIDs on training BigGAN on 10% training data from CIFAR-100. Smaller distance to the origin indicates smaller FID/better performance. Compared to the vanilla training baseline (★, i.e., dense model or 0% sparsity), our method's Stage I (●) finds highly sparse lottery tickets from the original BigGAN, with a range of sparsity up to 86.58%. Higher sparsity appears to bring better data-efficiency. Stage II further boosts the training of those found sparse subnetworks, by incorporating existing data-level augmentation [4] and our newly proposed feature-level augmentation (●).

Submitted to 35th Conference on Neural Information Processing Systems (NeurIPS 2021). Do not distribute.

In this paper, we address the above issue from a brand new perspective by decomposing the challenging GAN training in limited data regimes into two sequential sub-problems: (i) finding independent trainable subnetworks (i.e., lottery tickets in GANs) [2, 1]; then (ii) training the located subnetworks, which we show is more data-efficient by itself, and can further benefit from aggressive augmentations (both the input data and feature levels). Either sub-problem becomes much less data-hungry to train, and the two sub-problems re-use the same small training set of real images. Although this paper focuses on tackling the data-efficient training of GANs, such a coordinated framework might potentially be generalized to training other deep models with high data efficiency too.

Our key enabling technique is to leverage the lottery ticket hypothesis (**LTH**) [18]. LTH shows feasibility to locate highly sparse subnetworks (called "*winning tickets*") that are capable of training in isolation to match or even outperform the performance of original unpruned models. Recently, [1, 2] revealed the existence of winning ticket in GANs (called "*GAN tickets*"). However, none of the existing works discuss the influence of training data size on locating and training those tickets. Our work takes one step further, and show that one can identify same high-quality GAN tickets even in the data-scarce regime. The found GAN tickets also serve as a sparse structural prior to solve the second sub-problem with less data, while maintaining an unimpaired *trainability* blessed by the LTH assumption [18]. Figure 1 (the outer circle's blue dots) evidences that we can identify sparse GAN tickets that achieve superior performance than full GANs in the data-scarce scenarios.

The new lottery ticket angle complements the existing augmentation techniques [19, 20, 21], and we further show that they can be organically combined for boosted performance[1]. When we train the identified lottery ticket, we demonstrate its training can benefit as well from the latest *data-level* augmentation strategies, ADA [3] and DiffAug [4]. Furthermore, we introduce a novel ***feature-level* augmentation** that can be applied in parallel to data-level. It injects adversarial perturbations into GANs' intermediate features to implicitly regularize both discriminator and generator. Combining the new feature-level and existing data-level augmentations in training GAN tickets leads to more stabilized training dynamics, and establishes new state-of-the-arts for data-efficient GAN training.

Extensive experiments are conducted on a variety of latest GAN architectures and datasets, which consistently validate the effectiveness of our proposal. For example, our BigGAN tickets at 36.00% and 67.24% sparsity levels reach an (FID, IS) of $(23.14, 52.98)$ and $(70.91, 7.03)$, on Tiny-ImageNet 64×64 and ImageNet 128×128, with 10% and 25% training data, respectively. On CIFAR-10 and CIFAR-100, for SNGAN and BigGAN tickets at $67.24\% \sim 86.58\%$ sparsity, our results with only 10% training data can even surpass their dense counterparts. Impressively, our method can generate high-quality images on par with other GAN transfer learning approaches, by training on as few as 100 real samples and without using any pre-training.

2 Related Work

GANs and Data-Efficient GAN Training. GANs [22] have gained popularity in diverse computer vision scenarios. To stabilize GAN training and improve the visual fidelity and diversity of generated images, extensive studies have been conducted, such as sophisticated network architectures [23, 11, 24, 25, 26], improved training recipes [27, 7, 28, 29], and more stable objectives [30, 31, 32, 33, 34]. [35, 36] utilize semi- and self-supervised learning to pursue label efficiency in GAN training.

Recently, how to train GANs without sufficient real images in the target domain sparkles new interests. There have been efforts on adapting a pre-trained GAN generator, including BSA [37], AdaFM [38], Elastic Weight Consolidation [39], and Few-Shot GAN [40]. However, those methods assume a large, related source domain as *pre-training*, based on which they further alleviate target domain data limitation by only tuning small subsets of weights. They are hence in a **completely different track** from our "stand-alone" data-efficient training goal where no pre-training is leveraged. [41, 42] select core-sets of training data to speed up GAN training. A few recent attempts [4, 3] leverage differentiable or adaptive data augmentations to significantly improve GAN training in limited data regimes. Lately, [43] investigates a regularization approach, on constraining the distance between the current prediction of the real image and a moving average variable that tracks the historical predictions of the generated image, that complements the data augmentation methods.

Lottery Ticket Hypothesis and GAN Tickets. [18] claims the existence of independently trainable sparse subnetworks that can match or even surpass the performance of dense networks. [44, 45]

[1] We also tried to add augmentations in the lottery ticket finding stage, but did not observe visible impact.

scale up LTH by rewinding [46, 47]. Follow-up researches evidence LTH across broad fields, including visual recognition [18, 44, 48, 49, 50, 51, 52, 53, 54, 55, 56], natural language processing [45, 57, 47, 58, 59, 60], graph neural network [61], and reinforcement learning [57].

Recently, LTH has been extended to GANs by [2, 1], who validated the existence of winning tickets in the mini-max game beyond minimization. Compared with the aforementioned work, our work is the first to study LTH in the data-scarce regime (for GANs, and in general). Besides finding highly compact yet same capable subnetworks, our work reveals LTH's power in saving training data - an appealing perspective never being examined before.

Adversarial Training and Augmentations. Deep neural networks suffer from severe performance degradation [62, 63] when facing adversarial inputs [63, 64, 65]. To address this notorious vulnerability, various defense mechanisms [66, 67, 68, 69, 70, 71, 72, 73, 74, 75] have been proposed. Among others, adversarial training-based approaches achieve superior adversarial robustness [63, 64, 65], although at the price of sacrificing benign generalization [76, 66, 67, 68, 69, 70, 71].

Several recent works investigate enhancing model (benign) generalization ability with adversarial training [77, 78, 79, 80, 81, 82]. They adopt adversarially perturbed input images, embeddings or intermediate features, into model training to ameliorate performance on the clean test sets. Specifically, the damaging effects of adversarial training could be controlled by extra batch normalization [77] or so. Different from those minimization problems that previous work has focused on, the two-player GAN optimization is more challenging . Generally, the adversarial competition between two players poses impediments to exploit extra adversarial information during training GANs.

3 Methodology

3.1 Revisiting GANs and the Overfitting Challenge

Generative adversarial networks (GANs) are dedicated to modeling the target distribution with the two-player game formulation of a generator $\mathcal{G}$ and a discriminator $\mathcal{D}$. Specifically, the generator $\mathcal{G}$ takes a random sampled latent vector $\boldsymbol{z}$ (e.g., from a Gaussian distribution) as input and outputs the fake sample $\mathcal{G}(\boldsymbol{z})$. The discriminator $\mathcal{D}$ aims to distinguish generated fake samples $\mathcal{G}(\boldsymbol{z})$ from real samples $\boldsymbol{x}$. Alternative optimizations for the discriminator's loss $\mathcal{L}_\mathcal{D}$ and the generator's loss $\mathcal{L}_\mathcal{G}$ are adopted in the standard GAN training, which can be depicted as follows:

$$\begin{aligned} \mathcal{L}_\mathcal{D} &:= \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[f_\mathcal{D}(-\mathcal{D}(\boldsymbol{x}))] + \mathbb{E}_{\boldsymbol{z}\sim p(\boldsymbol{z})}[f_\mathcal{D}(\mathcal{D}(\mathcal{G}(\boldsymbol{z})))] \\ \mathcal{L}_\mathcal{G} &:= \mathbb{E}_{\boldsymbol{z}\sim p(\boldsymbol{z})}[f_\mathcal{G}(-\mathcal{D}(\mathcal{G}(\boldsymbol{z})))], \end{aligned}$$

where loss functions $f_\mathcal{D}(x), f_\mathcal{G}(x)$ have multiple choices, e.g., the non-saturating loss [6] with $f_\mathcal{D}(x) = f_\mathcal{G}(x) = \log(1+e^x)$, and the hinge loss [23] with $f_\mathcal{D}(x) = \max(0, 1+x)$ and $f_\mathcal{G}(x) = x$. $p_{\text{data}}(\boldsymbol{x})$ and $p(\boldsymbol{z})$ represent the data distribution of real samples and latent vectors. $\mathcal{L}_\mathcal{D}$ is maximized to update $\mathcal{D}$'s parameters ϕ (i.e., $\mathcal{D}(\cdot) := \mathcal{D}(\cdot, \phi)$), and $\mathcal{L}_\mathcal{G}$ is minimized to update $\mathcal{G}$'s parameters θ (i.e., $\mathcal{G}(\cdot) = \mathcal{G}(\cdot, \theta)$).

Algorithm 1 Data-Efficient Iterative Magnitude Pruning Procedures

1: **Input:** Initial two masks $m_g = 1^{\|\theta\|_0}$ and $m_d = 1^{\|\phi\|_0}$; Initialization weights θ_0 and ϕ_0
2: **Output:** $\{\mathcal{G}(\cdot, \theta_0 \odot m_g), \mathcal{D}(\cdot, \phi_0 \odot m_d)\}$
3: **repeat**
4: Training $\{\mathcal{G}(\cdot, \theta_0 \odot m_g), \mathcal{D}(\cdot, \phi_0 \odot m_d)\}$ for t epochs *with limited training data*
5: Pruning $\rho = 20\%$ of remaining weights in both $\mathcal{G}$ and $\mathcal{D}$
6: Updating the binary masks m_g and m_d accordingly
7: Rewinding weights of $\mathcal{G}$, $\mathcal{D}$ to θ_0 and ϕ_0
8: **until** masks reach the desired sparsity level

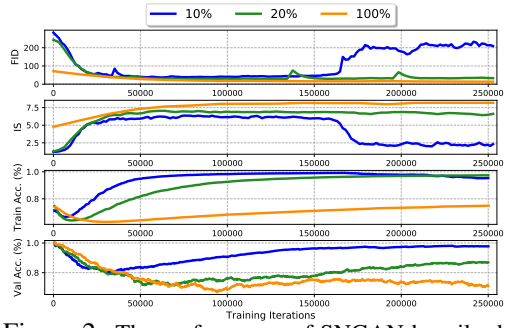


Figure 2: The performance of SNGAN heavily degrades with limited amount training data. Top two figures show that training with 10% of CIFAR-10 data incurs Fréchet Inception Distance (FID) explosion and Inception Score (IS) drop, with the model (**blue curves**) collapsed. Bottom two figures present $\mathcal{D}$'s training and validation accuracies of correctly predicting generated images as fake samples.

Training Failures of GANs under Limited Data. [3, 4] observe that GANs' performance is severely deteriorated when only limited training data available. The discriminator tends to memorize and heavily *overfit* the small training samples, leaving a gap between the real sample's and the generated sample's distribution. As shown in Figure 2, with only 10% CIFAR-10 data available for training SNGAN [23], the training and validation accuracies of the discriminator $\mathcal{D}$ quickly saturate to nearly 100% (ideally close to 50%), which indicates $\mathcal{D}$ to become over-confident in distinguishing real and generated samples. It demonstrates that $\mathcal{D}$ simply memorizes the training data, and such overfitting leads to training collapses and deteriorated quality of generated images.

To address this dilemma, we suggest a new data-efficient GAN training workflow, decomposed into two stages: (i) *finding winning tickets in GANs* via Algorithm 1; then (ii) *training the found GAN tickets*, potentially with both data- and feature-level augmentations, via Algorithm 2. Blessed by LTH, the located GAN ticket shows improved generalization ability, and is further enhanced by augmentations that prevent $\mathcal{D}$ from becoming too confident.

3.2 Data-Efficient Lottery Ticket Finding from GANs

In this section, we provide the preliminaries and setups to identifying data-efficient GAN tickets.

Subnetworks and winning tickets. A subnetwork of GAN is defined as $\{\mathcal{G}(\cdot, \theta \odot m_g), \mathcal{D}(\cdot, \phi \odot m_d)\}$, where $m_g \in \{0,1\}^{\|\theta\|_0}$ and $m_d \in \{0,1\}^{\|\phi\|_0}$ are binary masks for the generator and discriminator respectively, and $\odot$ is the element-wise product. Let θ_0 and ϕ_0 be the initialization weights of GANs. Following [18, 2], we define *winning tickets* of GAN as subnetworks $\{\mathcal{G}(\cdot, \theta_0 \odot m_g), \mathcal{D}(\cdot, \phi_0 \odot m_d)\}$, that reach a matched or better performance compared to unpruned GANs when trained in isolation with similar training iterations.

Finding data-efficient winning tickets in GANs. To our best knowledge, we are the first to extend LTH to the limited data regimes. In this challenging scenario, *only a small amount of training data are accessible for the finding and training of GAN tickets*. We use unstructured magnitude pruning [83], e.g., Iterative Magnitude Pruning (IMP), to establish the sparse masks m_g and m_d.

As shown in Algorithm 1, we first train the full GAN model for t epochs with limited training samples (e.g., 100-shot), and then perform IMP to globally prune the weights with the lowest magnitude. Zero elements in the obtained masks m_g and m_d index the pruned weights. Before repeating the process again, the weights of the sparse generator $\mathcal{G}(\cdot, \theta \odot m_g)$ and discriminator $\mathcal{D}(\cdot, \phi \odot m_d)$ are rewound to the same initialization θ_0 and ϕ_0, following the convention [18]. The pruning ratio ρ controls the portion of weights removed per round, and we fix $\rho = 20\%$ in all experiments.

Intuitively, identifying a special sparse mask (without requiring to train its weights well) should be an easier and hence more data-efficient task compared to training the full network weights. That was verified by our observations in experiments too: the quality of sparse mask (measured in terms of its achievable re-training performance in the full, or limited data regime) remains to be stable when the training data volume reduces from 100% to 10% of the full training set.

3.3 Data-Level and Feature-level Augmentations for Training GAN Tickets

After locating the GAN ticket at certain sparsity, training it using the vanilla recipe could already attain significantly improved IS and FID compared to the full dense model trained in the same data-limited regime: see Figure 2 outer circle for example.

Next, we discuss how our proposal can be applied together with augmentation-based approaches, for enhanced training of our found GAN tickets. Our natural choices include to plug-in the two recent state-of-the-art data augmentations, i.e., DiffAug [4] and ADA [3]. We further present a new adversarial *feature-level* augmentation (AdvAug), that can be jointly applied together with data-level augmentations to gain additional performance boost.

Specifically, our proposed AdvAug injects adversarial perturbations to intermediate feature embeddings of both $\mathcal{G}$ and $\mathcal{D}$. Similar feature augmentation idea was proven to be useful in NLP [78] and computer vision [82], showing effective to regularize the smoothness of training landscape and enhances the trained model's generalization. The AdvAug scheme is illustrated in Figure 3, and can be mathematically depicted as follows (λ_1 is a controlling hyperparameter):

$$\min_{\theta} \mathcal{L}_{\mathcal{G}} + \lambda_1 \cdot \mathcal{L}_{\mathcal{G}}^{\text{adv}} \; s.t. \; \mathcal{L}_{\mathcal{G}}^{\text{adv}} := \max_{\|\hat{\delta}\|_\infty \leq \epsilon} \mathbb{E}_{\boldsymbol{z} \sim p(\boldsymbol{z})}[f_{\mathcal{G}}(-\mathcal{D}(\mathcal{G}_2(\mathcal{G}_1(\boldsymbol{z}) + \hat{\delta})))]. \tag{1}$$

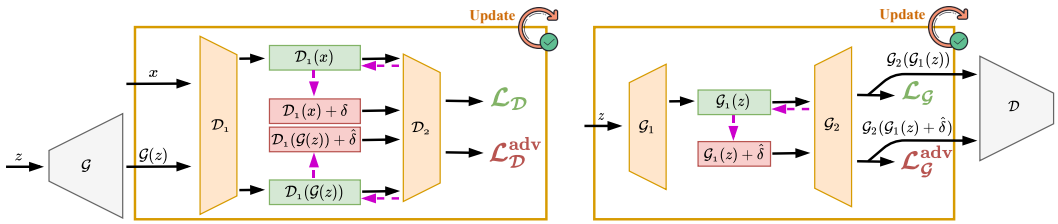


Figure 3: The pipeline of AdvAug for GANs. *Left:* Updating the discriminator $\mathcal{D}$; *Right:* Updating the generator $\mathcal{G}$. Purple arrows donate the path to generate adversarial feature perturbations.

We choose $\lambda_1 = 1$ in all experiments for simplicity. $\mathcal{G} = \mathcal{G}_2 \circ \mathcal{G}_1$ denotes the generator, and between the $\mathcal{G}_1$ and $\mathcal{G}_2$ parts we inject AdvAug. Adversarial perturbations $\hat{\delta}$, generated by multi-step projected gradient descent (PGD) [65], are applied to the intermediate feature space $\mathcal{G}_1(\boldsymbol{z})$. The full procedure of our feature-level augmentation and comprehensive detail description are provided in Appendix A1.

Note that AdvAug only affects the generated images through $\mathcal{G}$ intermediate features, and the classifier learning through $\mathcal{D}$ features. It hence avoids to directly manipulate the real data distribution. One bonus of doing so is that it is potentially better at alleviating the distribution leaking issue [3], i.e., GANs learn to mimic and generate the augmented distribution rather than the real one.

4 Experiments

In this section, we conduct comprehensive experiments on Tiny-ImageNet [84], ImageNet [85], CIFAR-10 [86], and CIFAR-100 based on the unconditional SNGAN [23] and StyleGAN-V2 [9], as well as the class-conditional BigGAN [5]. We adopt the common evaluation metrics, including Fréchet Inception Distance (FID) [87] and Inception Score (IS) [34]. Note that a smaller FID ($\downarrow$) and a larger IS ($\uparrow$) indicate better performing GAN models. Furthermore, we evaluate our proposed method on few-shot generation both with and without pre-training in Section 4.3. Extensive ablation studies analyze effectiveness of each component in Section 4.4.

Implementation and Baseline Details. We follow the popular StudioGAN codebase [88], which contains high-quality re-implementation of BigGAN and SNGAN on ImageNet and CIFAR. For example, our implemented BigGAN baseline performs much better, i.e., FID: 26.44 (ours) v.s. 39.78 (reported) on CIFAR-10, and FID: 36.58 (ours) v.s. 66.71 (reported) on CIFAR-100, than the recent reported baselines in [4], under 10% training data regimes. For detailed configuration, BigGAN takes learning rates of $\{4, 2, 2\} \times 10^{-4}$ for $\mathcal{G}$, of $\{1, 5, 2\} \times 10^{-4}$ for $\mathcal{D}$, batch sizes of $\{256, 256, 64\}$, 1×10^5 training iterations, and $\{1, 2, 5\}$ $\mathcal{D}$ steps per $\mathcal{G}$ step on {Tiny-ImageNet, ImageNet, CIFAR} datasets. SNGAN uses learning rates of 2×10^{-4} for $\mathcal{G}$ and $\mathcal{D}$, batch sizes of 64, 5×10^4 training iterations, and five $\mathcal{D}$ steps per $\mathcal{G}$ step on CIFAR. For StyleGAN-V2 experiments, we use its popular PyTorch implementation[2], and keep the default configuration in [4] including image resolution (256×256), learning rates for $\mathcal{D}/\mathcal{G}$ (2×10^{-4}), batch size (5), and training iterations (1×10^5).

Note that, same as the setting in [4], training iterations will be doubled when training GANs with DiffAug. We use implementations in the StudioGAN codebase for DiffAug [4], and the official implementation[3] for ADA [3]. AdvAug with PGD-1 and step size $0.01/0.001$ is applied on CIFAR/(Tiny-)ImageNet datasets, which are tuned by a grid search in Section 4.4. All GANs are trained with 1-8 NVIDA V100 32GB.

4.1 On the Effectiveness of Training with Winning Ticket and AdvAug

We adopt the top-performing model BigGAN [5], and report experiments on both Tiny-ImageNet at 64×64 resolution and ImageNet at 128×128 resolution. We evaluate our proposal on Tiny-ImageNet with $10\%, 20\%, 100\%$ data available, and ImageNet with 25% data available, with results summarized in Tables 1 and 2, respectively. All our results are averaged over three independent evaluation runs (same hereinafter), and the best performance of each column are highlighted.

The following observations can be drawn: *First*, sparse BigGAN tickets can achieve consistently improved performance over the full model (0.00%). Especially, with only 10% training data available, BigGAN tickets at 67.24% sparsity obtain massive gains of 11.30 FID and 1.58 IS on Tiny-ImageNet.

[2] https://github.com/lucidrains/StyleGAN-V2-pytorch. Note that the PyTorch version remains with a small performance gap compared to the TensorFlow implementation in [4].

[3] https://github.com/NVlabs/StyleGAN-V2-ada-pytorch. The official Pytorch implementation provided by [3].

Table 1: **Tiny-ImageNet** 64×64 performance without the truncation trick [5]. FID and IS are measured using 10K samples; the official validation set is utilized as the reference distribution. BigGANs at 0.00% (full unpruned models), $36.00\%, 67.24\%$ sparsity are found and trained with $100\%, 20\%, 10\%$ data, respectively.

Methods	100% training data (full set)		20% training data		10% training data	
	FID (↓)	IS (↑)	FID (↓)	IS (↑)	FID (↓)	IS (↑)
BigGAN (0.00%)	21.54 ± 0.03	18.33 ± 0.15	59.77 ± 0.05	7.81 ± 0.20	84.53 ± 0.08	5.45 ± 0.23
+ AdvAug	21.07 ± 0.03	18.92 ± 0.09	58.55 ± 0.05	8.46 ± 0.19	81.72 ± 0.05	6.32 ± 0.18
BigGAN (36.00%)	20.54 ± 0.05	18.42 ± 0.20	59.56 ± 0.04	7.98 ± 0.20	75.76 ± 0.08	6.49 ± 0.21
+ AdvAug	$\mathbf{20.02 \pm 0.04}$	$\mathbf{19.15 \pm 0.18}$	58.24 ± 0.05	8.55 ± 0.20	71.47 ± 0.07	6.86 ± 0.20
BigGAN (67.24%)	26.37 ± 0.03	16.38 ± 0.15	59.02 ± 0.03	8.17 ± 0.18	73.23 ± 0.05	6.68 ± 0.15
+ AdvAug	25.59 ± 0.03	17.62 ± 0.16	$\mathbf{57.60 \pm 0.04}$	$\mathbf{8.94 \pm 0.18}$	$\mathbf{70.91 \pm 0.05}$	$\mathbf{7.03 \pm 0.16}$

Figure 4: IS (↑) and FID (↓) results of SNGAN with $10\%, 20\%, 30\%, 50\%, 70\%, 90\%$ training data of **CIFAR-10**. Four settings are evaluated:(i) Dense (unpruned SNGAN), (ii) Dense+Aug (we only apply AdvAug here), (iii) Sparse Tickets (pruned SNGAN), (iv) Sparse Tickets+Aug, where the top performing variants are highlighted with **black boxes**. SNGAN tickets with $20\%, 36\%, 36\%, 67\%, 49\%, 36\%$ sparsity levels are adopted accordingly. IS and FID are measured using 10K samples; the validation set is utilized as the reference.

Second, feature-level augmentation (AdvAug) consistently improves all training cases, from dense to sparse, and from full data to limited data. In particular, larger sparsity (e.g., 67.24%) with less training data available (e.g., 10%) tend to benefit more from applying AdvAug, which is aligned with our design principle. *Third*, while the full data regime (100% data) does not necessarily prefer the highest sparsity (moderate sparsity still benefits), the limited data regimes (10% or 20% data) see monotonically increasing gains as the ticket sparsity goes higher. That is understandable since the former may need more model capacity to absorb full training data, while the latter case hinges on sparsity to avoid overfitting their limited training data.

Besides, we report another group of experiments of training SNGAN on CIFAR-10, using from 10% to 90% training data. The results are summarized in Figure 4. The conclusions we can draw are highly consistent with the above BigGAN case: (1) at the same data availability (from 10% to even 90%), training a sparse ticket is always preferred over training the dense model; (2) AdvAug is also consistently helpful in all cases; (3) for both sparsity and AdvAug, they can contribute to larger gains when training data gets smaller.

Table 2: **ImageNet** 128×128 performance without the truncation trick [5]. FID and IS are measured using 50K samples; the validation set is utilized as the reference distribution. BigGANs at 0.00% and 36.00% sparsity levels are adopted, and only 25% training data are available in all training stages.

Methods	25% training data	
	FID (↓)	IS (↑)
BigGAN (0.00%)	25.37 ± 0.07	46.50 ± 0.40
+ AdvAug	23.95 ± 0.06	47.95 ± 0.32
BigGAN (36.00%)	24.03 ± 0.08	50.07 ± 0.51
+ AdvAug	$\mathbf{23.14 \pm 0.07}$	$\mathbf{52.98 \pm 0.47}$

4.2 Incorporating Our Proposal with Latest Data Augmentations and Regularization

Combining DiffAug We first incorporate DiffAug [4], as a representative of latest data-level augmentation, into our proposal and show the complementary gains. We conduct experiments on the

Table 3: **CIFAR-10 and CIFAR-100** results. FID (↓) are measured using 10K samples; the validation set is utilized as the reference distribution. Full dense models and sparse winning tickets of BigGAN are reported with $100\%, 20\%, 10\%$, respectively. Specifically, BigGAN tickets with 67.24% and 86.58% sparsity levels are reported for the 100%, 20% and 10% training data regimes. Performance reported is averaged over three independent evaluation runs; all standard deviation are less than 1%.

Methods	CIFAR-10			CIFAR-100		
	100% data	20% data	10% data	100% data	20% data	10% data
Dense BigGAN	8.57	17.38	26.44	11.83	22.13	36.58
+ DiffAug [4]	8.09	13.04	17.40	10.60	18.32	25.69
+ DiffAug + AdvAug	**7.70**	12.19	14.40	**8.96**	17.94	23.94
Sparse BigGAN Tickets	8.26	16.03	25.41	11.73	21.05	30.96
+ DiffAug [4]	8.19	12.83	16.74	10.73	17.43	23.80
+ DiffAug + AdvAug	8.15	**12.02**	**14.38**	**10.14**	**17.19**	**22.37**

class-conditional BigGAN and unconditional SNGAN models with CIFAR-10 and CIFAR-100. For BigGAN, we utilize $100\%, 20\%, 10\%$ data to locate GAN tickets; and we then training them with data-level DiffAug, or with both DiffAug and feature-level AdvAug, as shown in Table 3. Consistent observations can be drawn: *First*, similarly to our previous observations on AdvAug, DiffAug also shows to contribute more when the training data becomes more limited; *Second*, combining DiffAug and AdvAug improves over either alone, and leads to the best results across all cases.

Combining Other Data Augmentations and Regularization We then extend our combination study to other recent data augmentation and regularization approaches, e.g., ADA [3] and R_{LC} [43]. Experiments are conducted on CIFAR-100 with StyleGAN-V2 backbone, and results are collected in Table 4. We observe that plugging in either ADA [3] or DiffAug [4] into our framework could improve sparse GAN winning tickets, and the gain is also enlarged when ADA is combined with AdvAug. Regard to R_{LC}[4], it is less effective combined with other augmentations.

Table 4: **CIFAR-100** results with only 10% training data available. FID (↓) of three evaluation runs are measured using 10K samples; the validation set is utilized as the reference distribution. Sparse StyleGAN-V2 tickets at 48.80% sparsity are adopted.

Methods	10% training data
Dense StyleGAN-V2	13.59 ± 0.06
+ DiffAug [4]	12.90 ± 0.04
+ ADA [3]	12.87 ± 0.03
+ ADA + R_{LC} [43]	13.01 ± 0.02
Sparse StyleGAN-V2 Tickets	13.05 ± 0.07
+ DiffAug [4]	12.53 ± 0.03
+ ADA [3]	12.20 ± 0.03
+ ADA + R_{LC} [43]	12.48 ± 0.04
+ ADA [3] + AdvAug	$\mathbf{12.11 \pm 0.05}$

Taking above together, it has been clearly shown that our proposal is orthogonal to those existing efforts and is of independent merit. Moreover, combing them would lead to more powerful pipelines for data-efficient GAN training.

4.3 Few-Shot Generation

It is tedious, laborious and sometime impossible to collect a large-scale dataset for certain images of interest. To tackle the few-shot image generation problem, [89] utilizes pre-training from external large-scale datasets and performs fine-tuning under limited data scenarios; [90], [37] and [91] partially fine-tune the GANs with part of the GAN model being frozen.

We compare these transfer learning approaches[5] with our data-efficient training scheme. **Differently from them all**, ours is training from scratch and is free of any pre-training, while all transfer learning methods start from on a pre-trained StyleGAN-V2 model on the FFHQ face dataset [8].

Our comparison experiments are conducted using StyleGAN-V2 on the AnimalFace [92] dataset (160 cats and 389 dogs), and the 100-shot Obama, Grumpy Cat, and Panda datasets provided by [4]. As shown in Table 5, our method finds data-efficient GAN ticket at 48.80% sparsity levels, that can be trained with only 100 training samples from scratch (*without any pre-training*) and show competitive performance to other transfer learning algorithms. Visualizations of style space interpolation and few-shot generation are provided in Figure 5 and 6.

[4] [43] advocates the best-performing configuration is ADA+R_{LC}, with FID 13.01 on 10% data of CIFAR-100.
[5] Implementations are from the codebase of [90].

Table 5: **Few-shot generation.** Following the setting in [4], we calculate the FID with 5K samples and the training dataset is adopted as the reference distribution. All transfer learning methods have their pre-trainings from FFHQ [8]. StyleGAN-V2 tickets at 48.80% sparsity level are found and used in our method.

Methods	Pre-training?	100-shot by [4]			AnimalFace	
		Obama	Grumpy Cat	Panda	Cat	Dog
Scale/shift [37]	Yes	50.72	34.20	21.38	54.83	83.04
MineGAN [91]	Yes	50.63	35.54	14.84	54.45	93.03
TransferGAN [89]	Yes	48.73	34.06	23.20	52.61	82.38
FreezeD [90]	Yes	41.87	31.22	17.95	47.70	70.46
StyleGAN-V2 (0.00%)	No	89.18	61.97	90.96	95.75	164.54
+ DiffAug + AdvAug	No	54.11	35.46	15.94	54.02	72.47
StyleGAN-V2 Tickets (48.80%)	No	73.92	56.81	82.45	85.92	153.90
+ DiffAug + AdvAug	No	**52.86**	**31.02**	**14.75**	**47.40**	**68.28**

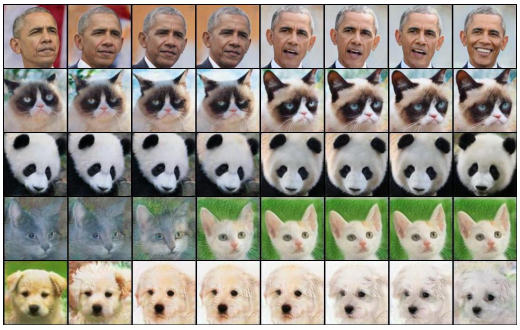


Figure 5: Style interpolation visualizations of StyleGAN-V2 tickets (48.80%) with AdvAug only on 100-shot Obama, Grumpy Cat, Panda, and AnimalFace datasets, respectively.

Figure 6: Few-shot generalization results of StyleGAN-V2 tickets (48.80%) with AdvAug only on 100-shot Obama, Grumpy Cat, Panda, and AnimalFace datasets, respectively.

4.4 Ablation and Analysis

Pruning Ratio ρ in the Ticket Finding. To understand the effect of the pruning ratio in IMP to the quality of data-efficient GAN tickets, we experiment on SNGAN with 10% data of CIFAR-10 and $\rho = 10\%, 20\%, 40\%$ as the pruning ratio. As shown in Figure 7 (*Left*), all three IMP settings find data-efficient winning tickets in GAN; IMP with a lower pruning ratio tends to identify higher-quality GAN tickets in terms of FID, while it usually costs much more to reach the same level of sparsity as higher pruning ratios do.

Augment $\mathcal{G}$ or $\mathcal{D}$: Either or Both. We apply AdvAug on $\mathcal{G}$ or $\mathcal{D}$ only, and AdvAug on both $\mathcal{G}$ and $\mathcal{D}$. Only 10% data of CIFAR-10 are available for finding GAN tickets and training with AdvAug. Results are summarized in Figure 7 (*Middle*). Either employing AdvAug on $\mathcal{D}$ or $\mathcal{G}$ consistently obtains significant performance improvements (i.e., largely reducing the FID) over all sparsity levels, and augmenting both $\mathcal{D}$ and $\mathcal{G}$ further enhance the found data-efficient GAN tickets. Our results also show that sparser GAN tickets can benefit more from AdvAug, such as subnetworks with $20\%, 36\%, 83.22\%$ sparsity.

Strength and Locations of Injecting AdvAug. To better interpret the influence of the strength and layer locations of injected adversarial feature perturbations, we comprehensively examine SNGAN on 10% training data of CIFAR-10 across different step sizes, the number of PGD steps, and locations (i.e., where to apply AdvAug). When studying one of the factors, we fix the other factors with the best setup. From Figure 8, several observations can be drawn:

- Figure 8 (a) and (b) show that adopting AdvAug with step size 0.01/0.001 and PGD-1/3 assists the data-efficient GAN training, while AdvAug with step size 0.05/0.1 and PGD-5/7 perform worse than the baseline without AdvAug (i.e., the setting with zero step size or zero step PGD in Figure 8). It reveals that overly strong AdvAug can hurt performance.
- As shown in Figure 8 (c) and (d), augmenting the last layer of the discriminator $\mathcal{D}$ and the first layer (i.e., the closest layer to the input latent vector) of the generator $\mathcal{G}$ appears to be

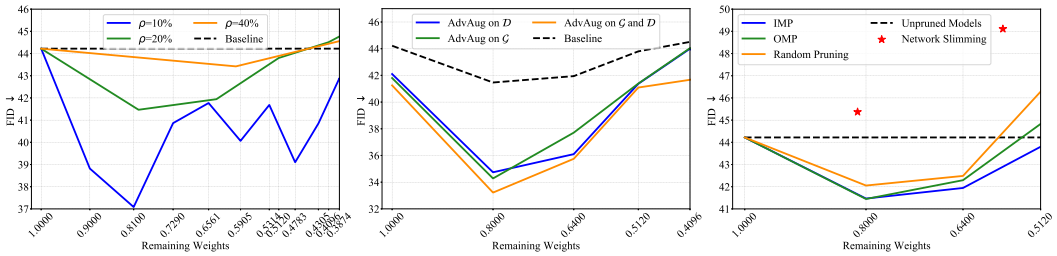
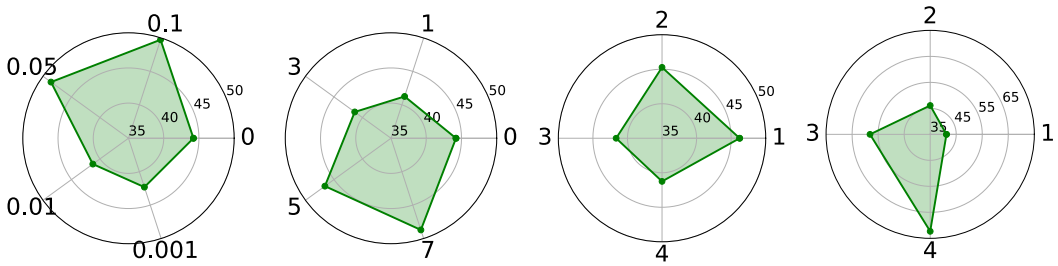


Figure 7: Performance of training GAN with 10% data of CIFAR-10. Remaining weight indicates the sparsity levels of identified GAN tickets. *Left*: FID of the data-efficient GAN tickets found by IMP with pruning ratios $\rho = 10\%, 20\%, 40\%$. *Middle*: FID of the data-efficient GAN tickets trained with different settings of AdvAug, including baseline without AdvAug, AdvAug on $\mathcal{D}$ or $\mathcal{G}$ only, and AdvAug on both $\mathcal{G}$ and $\mathcal{D}$. *Right*: FID of trained SNGAN tickets found by IMP, Random Pruning, OMP, and Network Slimming [93].



(a) Step size (b) The number of steps (c) Layer index in $\mathcal{D}$ (d) Layer index in $\mathcal{G}$

Figure 8: Ablation study on the location and strength of introducing AdvAug to data-efficient GAN training. The step size and the number of steps roughly indicate the strength of generated adversarial perturbations, e.g., a smaller step size or fewer steps for PGD means less aggressive perturbations [65]. FID (↓) is reported.

the best configuration for utilizing AdvAug. It seems that injecting adversarial perturbations into the "high-level" feature embeddings in general benefits more to mitigate the overfitting issue in data limited regimes.

In summary, we observe that applying AdvAug to the last layer of $\mathcal{D}$ and the first layer of $\mathcal{G}$, with PGD-1 and step size 0.01, seems to be a sweet-point configuration for data-efficient GAN training, which is hence adopted as our default setting.

Comparison with Baselines. Naive baselines, i.e., random pruning, one-shot magnitude pruning (OMP) [83, 2], network slimming (NS [93], and random noise augmentation, are evaluated in Figure 7 (*Right*) and Table 6. Compared to random pruning, OMP and NS, IMP produces much better GAN tickets, especially at high sparsity levels (e.g., $\geq 48.8\%$); Compared to augmenting features with random noise sampled from $\mathcal{N}(0, 0.01^2)$, AdvAug also achieves larger performance gains on both 100% and 10% training data regimes.

Table 6: Performance of SNGAN models augmented by Gaussian Noise or AdvAug on 100% and 10% training data.

Methods	100% training data		10% training data	
	IS	FID	IS	FID
Baseline	8.29	15.69	5.24	44.22
+ Gaussian Noise	8.30	14.52	5.53	44.86
+ AdvAug	8.42	13.99	6.10	41.25

5 Conclusion and Discussion of Broad Impact

We introduce a novel perspective for data-efficient GAN training by leveraging lottery tickets, which can be further enhanced by augmentations including our newly introduced feature-level augmentation. Comprehensive experiments consistently demonstrate the effectiveness of our proposal, on diverse GAN architectures, objectives, and datasets. Note that although finding lottery tickets requires costly train-prune-retrain process, the goal and interest of our work is focused on *data efficiency*. It would be an intriguing future work to pursue data and resource efficiency (training and inference) together.

As shown in above, this research is scientific in nature. We do not think that it will pose substantial risk of societal harm. Our proposed frameworks are most likely to lessen the data and financial cost of training generative models in real-world scenarios.

References

[1] N. M. Kalibhat, Y. Balaji, and S. Feizi. Winning lottery tickets in deep generative models. *arXiv preprint arXiv:2010.02350*, 2020.

[2] Xuxi Chen, Zhenyu Zhang, Yongduo Sui, and Tianlong Chen. Gans can play lottery tickets too. In *ICLR*, 2021.

[3] Tero Karras, Miika Aittala, Janne Hellsten, Samuli Laine, Jaakko Lehtinen, and Timo Aila. Training generative adversarial networks with limited data. *arXiv preprint arXiv:2006.06676*, 2020.

[4] Shengyu Zhao, Zhijian Liu, Ji Lin, Jun-Yan Zhu, and Song Han. Differentiable augmentation for data-efficient gan training. In *NeurIPS*, 2020.

[5] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale gan training for high fidelity natural image synthesis. *arXiv preprint arXiv:1809.11096*, 2018.

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

[7] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of gans for improved quality, stability, and variation. *arXiv preprint arXiv:1710.10196*, 2017.

[8] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *CVPR*, 2019.

[9] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of stylegan. In *CVPR*, 2020.

[10] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral Normalization for Generative Adversarial Networks. In *ICLR*, 2018.

[11] Takeru Miyato and Masanori Koyama. cgans with projection discriminator. In *ICLR*, 2018.

[12] Orest Kupyn, Tetiana Martyniuk, Junru Wu, and Zhangyang Wang. Deblurgan-v2: Deblurring (orders-of-magnitude) faster and better. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8878–8887, 2019.

[13] Xinyu Gong, Shiyu Chang, Yifan Jiang, and Zhangyang Wang. Autogan: Neural architecture search for generative adversarial networks. In *Proceedings of the IEEE International Conference on Computer Vision*, 2019.

[14] Yifan Jiang, Xinyu Gong, Ding Liu, Yu Cheng, Chen Fang, Xiaohui Shen, Jianchao Yang, Pan Zhou, and Zhangyang Wang. Enlightengan: Deep light enhancement without paired supervision. *IEEE Transactions on Image Processing*, 30:2340–2349, 2021.

[15] Ashish Shrivastava, Tomas Pfister, Oncel Tuzel, Josh Susskind, Wenda Wang, and Russ Webb. Learning from simulated and unsupervised images through adversarial training. In *CVPR*, 2017.

[16] Martin Arjovsky and Léon Bottou. Towards principled methods for training generative adversarial networks. *arXiv preprint arXiv:1701.04862*, 2017.

[17] Dan Zhang and Anna Khoreva. Pa-gan: Improving gan training by progressive augmentation. 2018.

[18] Jonathan Frankle and Michael Carbin. The lottery ticket hypothesis: Finding sparse, trainable neural networks. In *ICLR*, 2019.

[19] Ngoc-Trung Tran, Viet-Hung Tran, Ngoc-Bao Nguyen, Trung-Kien Nguyen, and Ngai-Man Cheung. Towards good practices for data augmentation in gan training. *arXiv preprint arXiv:2006.05338*, 2020.

[20] Han Zhang, Zizhao Zhang, Augustus Odena, and Honglak Lee. Consistency regularization for generative adversarial networks. *arXiv preprint arXiv:1910.12027*, 2019.

[21] Zhengli Zhao, Sameer Singh, Honglak Lee, Zizhao Zhang, Augustus Odena, and Han Zhang. Improved consistency regularization for gans. *arXiv preprint arXiv:2002.04724*, 2020.

[22] Jie Gui, Zhenan Sun, Yonggang Wen, Dacheng Tao, and Jieping Ye. A review on generative adversarial networks: Algorithms, theory, and applications. *arXiv preprint arXiv:2001.06937*, 2020.

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

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

[25] Han Zhang, Ian Goodfellow, Dimitris Metaxas, and Augustus Odena. Self-attention generative adversarial networks. In *International conference on machine learning*, pages 7354–7363. PMLR, 2019.

[26] Hanting Chen, Yunhe Wang, Han Shu, Changyuan Wen, Chunjing Xu, Boxin Shi, Chao Xu, and Chang Xu. Distilling portable generative adversarial networks for image translation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 3585–3592, 2020.

[27] Emily L Denton, Soumith Chintala, Rob Fergus, et al. Deep generative image models using a laplacian pyramid of adversarial networks. *NeurIPS*, 2015.

[28] Steven Liu, Tongzhou Wang, David Bau, Jun-Yan Zhu, and Antonio Torralba. Diverse image generation via self-conditioned gans. In *CVPR*, 2020.

[29] Han Zhang, Tao Xu, Hongsheng Li, Shaoting Zhang, Xiaogang Wang, Xiaolei Huang, and Dimitris N Metaxas. Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks. In *ICCV*, 2017.

[30] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *ICML*, 2017.

[31] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron Courville. Improved training of wasserstein gans. In *NeurIPS*, 2017.

[32] Xudong Mao, Qing Li, Haoran Xie, Raymond YK Lau, Zhen Wang, and Stephen Paul Smolley. Least squares generative adversarial networks. In *ICCV*, 2017.

[33] Lars Mescheder, Andreas Geiger, and Sebastian Nowozin. Which training methods for gans do actually converge? *arXiv preprint arXiv:1801.04406*, 2018.

[34] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. *arXiv preprint arXiv:1606.03498*, 2016.

[35] Ting Chen, Xiaohua Zhai, Marvin Ritter, Mario Lucic, and Neil Houlsby. Self-supervised gans via auxiliary rotation loss. In *CVPR*, 2019.

[36] 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.

[37] Atsuhiro Noguchi and Tatsuya Harada. Image generation from small datasets via batch statistics adaptation. In *ICCV*, 2019.

[38] Miaoyun Zhao, Yulai Cong, and Lawrence Carin. On leveraging pretrained gans for generation with limited data. In *International Conference on Machine Learning*, pages 11340–11351. PMLR, 2020.

[39] Yijun Li, Richard Zhang, Jingwan Lu, and Eli Shechtman. Few-shot image generation with elastic weight consolidation. *arXiv preprint arXiv:2012.02780*, 2020.

[40] Esther Robb, Wen-Sheng Chu, Abhishek Kumar, and Jia-Bin Huang. Few-shot adaptation of generative adversarial networks. *arXiv preprint arXiv:2010.11943*, 2020.

[41] Samarth Sinha, Han Zhang, Anirudh Goyal, Yoshua Bengio, Hugo Larochelle, and Augustus Odena. Small-gan: Speeding up gan training using core-sets. In *ICML*, 2020.

[42] Terrance DeVries, Michal Drozdzal, and Graham W Taylor. Instance selection for gans. *arXiv preprint arXiv:2007.15255*, 2020.

[43] Hung-Yu Tseng, Lu Jiang, Ce Liu, Ming-Hsuan Yang, and Weilong Yang. Regularizing generative adversarial networks under limited data. *arXiv preprint arXiv:2104.03310*, 2021.

[44] Zhuang Liu, Mingjie Sun, Tinghui Zhou, Gao Huang, and Trevor Darrell. Rethinking the value of network pruning. In *ICLR*, 2019.

[45] Trevor Gale, Erich Elsen, and Sara Hooker. The state of sparsity in deep neural networks. *arXiv preprint arXiv:1902.09574*, 2019.

[46] Jonathan Frankle, Gintare Karolina Dziugaite, Daniel M. Roy, and Michael Carbin. Linear mode connectivity and the lottery ticket hypothesis. In *ICML*, 2020.

[47] Alex Renda, Jonathan Frankle, and Michael Carbin. Comparing rewinding and fine-tuning in neural network pruning. In *ICLR*, 2020.

[48] Chaoqi Wang, Guodong Zhang, and Roger Grosse. Picking winning tickets before training by preserving gradient flow. In *ICLR*, 2020.

[49] Utku Evci, Fabian Pedregosa, Aidan Gomez, and Erich Elsen. The difficulty of training sparse neural networks. *arXiv preprint arXiv:1906.10732*, 2019.

[50] Jonathan Frankle, David J. Schwab, and Ari S. Morcos. The early phase of neural network training. In *ICLR*, 2020.

[51] Pedro Savarese, Hugo Silva, and Michael Maire. Winning the lottery with continuous sparsification. In *NeurIPS*, 2020.

[52] Shihui Yin, Kyu-Hyoun Kim, Jinwook Oh, Naigang Wang, Mauricio Serrano, Jae-Sun Seo, and Jungwook Choi. The sooner the better: Investigating structure of early winning lottery tickets, 2020.

[53] Haoran You, Chaojian Li, Pengfei Xu, Yonggan Fu, Yue Wang, Xiaohan Chen, Richard G. Baraniuk, Zhangyang Wang, and Yingyan Lin. Drawing early-bird tickets: Toward more efficient training of deep networks. In *ICLR*, 2020.

[54] Tianlong Chen, Zhenyu Zhang, Sijia Liu, Shiyu Chang, and Zhangyang Wang. Long live the lottery: The existence of winning tickets in lifelong learning. In *ICLR*, 2021.

[55] Haoyu Ma, Tianlong Chen, Ting-Kuei Hu, Chenyu You, Xiaohui Xie, and Zhangyang Wang. Good students play big lottery better. *arXiv*, abs/2101.03255, 2021.

[56] Tianlong Chen, Jonathan Frankle, Shiyu Chang, Sijia Liu, Yang Zhang, Michael Carbin, and Zhangyang Wang. The lottery tickets hypothesis for supervised and self-supervised pre-training in computer vision models. *arXiv*, abs/2012.06908, 2020.

[57] Haonan Yu, Sergey Edunov, Yuandong Tian, and Ari S. Morcos. Playing the lottery with rewards and multiple languages: lottery tickets in rl and nlp. In *ICLR*, 2020.

[58] Tianlong Chen, Jonathan Frankle, Shiyu Chang, Sijia Liu, Yang Zhang, Zhangyang Wang, and Michael Carbin. The lottery ticket hypothesis for pre-trained bert networks. *arXiv preprint arXiv:2007.12223*, 2020.

[59] Shrey Desai, Hongyuan Zhan, and Ahmed Aly. Evaluating lottery tickets under distributional shifts. In *Proceedings of the 2nd Workshop on Deep Learning Approaches for Low-Resource NLP*, 2019.

[60] Xiaohan Chen, Yu Cheng, Shuohang Wang, Zhe Gan, Zhangyang Wang, and Jingjing Liu. Earlybert: Efficient bert training via early-bird lottery tickets. *arXiv preprint arXiv:2101.00063*, 2020.

[61] Tianlong Chen, Yongduo Sui, Xuxi Chen, Aston Zhang, and Zhangyang Wang. A unified lottery ticket hypothesis for graph neural networks, 2021.

[62] 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.

[63] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

[64] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial machine learning at scale. *arXiv preprint arXiv:1611.01236*, 2016.

[65] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

[66] Hongyang Zhang, Yaodong Yu, Jiantao Jiao, Eric P Xing, Laurent El Ghaoui, and Michael I Jordan. Theoretically principled trade-off between robustness and accuracy. *arXiv preprint arXiv:1901.08573*, 2019.

[67] Ludwig Schmidt, Shibani Santurkar, Dimitris Tsipras, Kunal Talwar, and Aleksander Madry. Adversarially robust generalization requires more data. In *NeurIPS*, 2018.

[68] Ke Sun, Zhanxing Zhu, and Zhouchen Lin. Towards understanding adversarial examples systematically: Exploring data size, task and model factors. *arXiv preprint arXiv:1902.11019*, 2019.

[69] Preetum Nakkiran. Adversarial robustness may be at odds with simplicity. *arXiv preprint arXiv:1901.00532*, 2019.

[70] David Stutz, Matthias Hein, and Bernt Schiele. Disentangling adversarial robustness and generalization. In *CVPR*, 2019.

[71] Aditi Raghunathan, Sang Michael Xie, Fanny Yang, John C Duchi, and Percy Liang. Adversarial training can hurt generalization. In *ICMLW*, 2019.

[72] Ziyu Jiang, Tianlong Chen, Ting Chen, and Zhangyang Wang. Robust pre-training by adversarial contrastive learning. *arXiv preprint arXiv:2010.13337*, 2020.

[73] Haotao Wang, Tianlong Chen, Shupeng Gui, Ting-Kuei Hu, Ji Liu, and Zhangyang Wang. Once-for-all adversarial training: In-situ tradeoff between robustness and accuracy for free. *arXiv preprint arXiv:2010.11828*, 2020.

[74] Ting-Kuei Hu, Tianlong Chen, Haotao Wang, and Zhangyang Wang. Triple wins: Boosting accuracy, robustness and efficiency together by enabling input-adaptive inference. *arXiv preprint arXiv:2002.10025*, 2020.

[75] Tianlong Chen, Sijia Liu, Shiyu Chang, Yu Cheng, Lisa Amini, and Zhangyang Wang. Adversarial robustness: From self-supervised pre-training to fine-tuning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 699–708, 2020.

[76] Dimitris Tsipras, Shibani Santurkar, Logan Engstrom, Alexander Turner, and Aleksander Madry. Robustness may be at odds with accuracy. *arXiv preprint arXiv:1805.12152*, 2018.

[77] Cihang Xie, Mingxing Tan, Boqing Gong, Jiang Wang, Alan L Yuille, and Quoc V Le. Adversarial examples improve image recognition. In *CVPR*, 2020.

[78] Chen Zhu, Yu Cheng, Zhe Gan, Siqi Sun, Tom Goldstein, and Jingjing Liu. Freelb: Enhanced adversarial training for natural language understanding. In *ICLR*, 2020.

[79] Dilin Wang, Chengyue Gong, and Qiang Liu. Improving neural language modeling via adversarial training. *arXiv preprint arXiv:1906.03805*, 2019.

[80] Zhe Gan, Yen-Chun Chen, Linjie Li, Chen Zhu, Yu Cheng, and Jingjing Liu. Large-scale adversarial training for vision-and-language representation learning. In *NeurIPS*, 2020.

[81] Colin Wei and Tengyu Ma. Improved sample complexities for deep networks and robust classification via an all-layer margin. *arXiv preprint arXiv:1910.04284*, 2019.

[82] Tianlong Chen, Yu Cheng, Zhe Gan, Yu Hu, Zhangyang Wang, and Jingjing Liu. Alfa: Adversarial feature augmentation for enhanced image recognition, 2021.

[83] Song Han, Huizi Mao, and William J. Dally. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. In *ICLR*, 2016.

[84] Jiayu Wu, Qixiang Zhang, and Guoxi Xu. Tiny imagenet challenge. Technical report, Stanford University, 2017.

[85] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009.

[86] Alex Krizhevsky et al. Learning multiple layers of features from tiny images. 2009.

[87] 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 *NeurIPS*, 2017.

[88] Minguk Kang and Jaesik Park. ContraGAN: Contrastive Learning for Conditional Image Generation. *NeurIPS*, 2020.

[89] Yaxing Wang, Chenshen Wu, Luis Herranz, Joost van de Weijer, Abel Gonzalez-Garcia, and Bogdan Raducanu. Transferring gans: generating images from limited data. In *ECCV*, 2018.

[90] Sangwoo Mo, Minsu Cho, and Jinwoo Shin. Freeze discriminator: A simple baseline for fine-tuning gans. *arXiv preprint arXiv:2002.10964*, 2020.

[91] Yaxing Wang, Abel Gonzalez-Garcia, David Berga, Luis Herranz, Fahad Shahbaz Khan, and Joost van de Weijer. Minegan: effective knowledge transfer from gans to target domains with few images. In *CVPR*, 2020.

[92] Zhangzhang Si and Song-Chun Zhu. Learning hybrid image templates (hit) by information projection. *TPAMI*, 2011.

[93] Zhuang Liu, Jianguo Li, Zhiqiang Shen, Gao Huang, Shoumeng Yan, and Changshui Zhang. Learning efficient convolutional networks through network slimming. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2736–2744, 2017.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] Please see Section 5.
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] Please see Section 5.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A] Our work does not have theoretical results.
   (b) Did you include complete proofs of all theoretical results? [N/A] Our work does not have theoretical results.
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] All used datasets are publicly available, and we follows the standard instructions in the cited papers, as shown in Section 4 and A2. All of our codes are included in the supplement.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] All training details are provides in Section 4.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] There independent evaluations are conducted. Meanwhile, the average performance with their standard deviations are reported in our paper.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] The description of adopted computing resources are collected in Section 4.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes] We use the public datasets and also cite their creators, as shown in Section 4 and A2.
   (b) Did you mention the license of the assets? [No] The licenses of the datasets are provided in the cited papers.
   (c) Did you include any new assets either in the supplemental material or as a URL? [Yes] All used datasets are public available. All of our codes are included in the supplement.
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A] We did not use/curate new data.
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A] We only use public and widely adopted datasets in this paper. We do not think there are any issues of personally identifiable information or offensive content.
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]