CHAIN: Enhancing Generalization in Data-Efficient GANs via lips*CH*itz continuity constr*AI*ned *N*ormalization

Yao Ni[†]   Piotr Koniusz[*,§,†]

[†]The Australian National University   [§]Data61♥CSIRO

[†]firstname.lastname@anu.edu.au

Abstract

Generative Adversarial Networks (GANs) significantly advanced image generation but their performance heavily depends on abundant training data. In scenarios with limited data, GANs often struggle with discriminator overfitting and unstable training. Batch Normalization (BN), despite being known for enhancing generalization and training stability, has rarely been used in the discriminator of Data-Efficient GANs. Our work addresses this gap by identifying a critical flaw in BN: the tendency for gradient explosion during the centering and scaling steps. To tackle this issue, we present CHAIN (lipsCHitz continuity constrAIned Normalization), which replaces the conventional centering step with zero-mean regularization and integrates a Lipschitz continuity constraint in the scaling step. CHAIN further enhances GAN training by adaptively interpolating the normalized and unnormalized features, effectively avoiding discriminator overfitting. Our theoretical analyses firmly establishes CHAIN's effectiveness in reducing gradients in latent features and weights, improving stability and generalization in GAN training. Empirical evidence supports our theory. CHAIN achieves state-of-the-art results in data-limited scenarios on CIFAR-10/100, ImageNet, five low-shot and seven high-resolution few-shot image datasets. Code: https://github.com/MaxwellYaoNi/CHAIN.

1. Introduction

The availability of abundant data, exemplified by ImageNet [19], has driven breakthroughs in deep neural networks [52], particularly in generative models. This data richness has fueled innovations such as Generative Adversarial Networks (GANs) [23], popular in academia and industry. GANs, known for their rapid generation speeds [82] and high-fidelity image synthesis [81], have become go-to tools for applications such as text-to-image generation [35, 82, 94], image-to-image translation [45, 73, 76, 87], video synthesis [86, 103, 110] and 3D generation [92, 114, 124].

Despite the advanced capabilities of modern GANs [8, 40, 41] in creating high-fidelity images, their success largely depends on access to extensive training data. However, in scenarios with limited data, such as medical [43] or art images [42, 46], where data acquisition is expensive and privacy concerns are paramount, GANs face issues such as discriminator overfitting and unstable training [39, 97, 120].

To overcome these obstacles, three main directions stand out. The first leverages massive data augmentation ("MA"), aimed at broadening the available data distribution [29, 33, 39, 55, 108, 113, 120]. The second strategy borrows knowledge from models trained on large datasets [15, 48, 102, 122]. However, these approaches suffer from issues such as the potential leakage of augmentation artifacts [39, 70, 72, 113] and the misuse of pre-training knowledge [22, 50, 106]. The third direction addresses discriminator overfitting and focuses on discriminator regularization to either reduce the capacity of the discriminator [20, 25, 44, 68, 69] or increase the overlap between real and fake data [33, 95, 97], making it harder for the discriminator to learn. While such methods are effective, their mechanism preventing overfitting is not clearly elucidated.

Aligning with the third direction of regularizing the discriminator, we innovate by reconsidering the integration of Batch Normalization (BN) [31] into the discriminator to improve the generalization. BN has been demonstrated, both theoretically and in practice, to improve neural network generalization. This is achieved via its standardization process, effectively aligning training and test distributions in a common space [54, 83, 101]. Additionally, BN reduces the sharpness of the loss landscape [7, 36, 62, 80] and stabilizes the training process by mitigating internal covariate shift.

Given these benefits, Batch Normalization appears as a good solution to preventing discriminator overfitting in GANs. However, large-scale experiments [49, 65, 66, 105, 118] have shown that incorporating BN into the discriminator actually impairs performance. Thus, BN is often omitted in the discriminator of modern GANs, *e.g.*, BigGAN [8], ProGAN [37], StyleGAN 1-3 [38, 40, 41], with few models using BN in the discriminator, *i.e.*, DCGAN [74].

Addressing the challenges of BN in GAN discriminator design, we have identified that the centering and scal-

---

ing steps of BN can lead to gradient explosion, a significant barrier in GAN convergence [44, 63, 96, 115]. To circumvent this issue while leveraging the benefits of BN, we propose replacing the centering step with zero mean regularization and enforcing the Lipschitz continuity constraint on the scaling step. This modification resolves gradient issues and also helps the discriminator effectively balance discrimination and generalization [115] through adaptive interpolation of normalized and unnormalized features.

We call our approach lips<u>CH</u>itz continuity constr<u>AI</u>ned <u>N</u>ormalization, in short, CHAIN, symbolized as ◯. Such a name and symbol represent the role of our model in bridging the gap between seen and unseen data and reducing the divergence between fake and real distributions. Despite CHAIN's simplicity, our theoretical analysis confirms its efficacy in reducing the gradient norm of both latent activations and discriminator weights. Experimental evidence shows that CHAIN stabilizes GAN training and enhances generalization. CHAIN outperforms existing methods that limit discriminator overfitting, achieving state-of-the-art results on data-limited benchmarks such as CIFAR-10/100, ImageNet, 5 low-shot and 7 high-resolution few-shot image generation tasks. Our contributions are as follows:

i. We tackle discriminator overfitting by enhancing GAN generalization, deriving a new error bound that emphasizes reducing the gradient of discriminator weights.

ii. We identify that applying BN in the discriminator, both theoretically and empirically, tends to cause gradient explosion due to the centering and scaling steps of BN.

iii. We provide evidence, both theoretical and practical, that CHAIN stabilizes GAN training by moderating the gradient of latent features, and improves generalization by lowering the gradient of the weights.

2. Background

Improving GANs. Generative Adversarial Networks [23], effective in image generation [8, 40, 57], image-to-image translation [45, 88, 89, 125], video synthesis [86, 103, 110], 3D generation [92, 114, 124] and text-to-image generation [35, 82, 94], suffer from unstable training [44, 96], mode collapse [63, 77], and discriminator overfitting [39, 120]. Improving GANs includes architecture modifications [8, 38, 40, 41, 53, 112], loss function design [3, 71, 119, 123] and regularization design [25, 44, 59, 65, 97]. BigGAN [8] scales up GANs for large-scale datasets with increased batch sizes. StyleGANs [38, 40, 41] revolutionize generator architecture by style integration. OmniGAN [123] modifies the projection loss [64] into a multi-label softmax loss. WGAN-GP [25], SNGAN [65] and SRGAN [59] regularize discriminator using a gradient penalty or spectral norm constraints for stable training. Our novel normalization effectively enhances GANs under limited data scenarios, applicable across various architectures and loss functions.

Image generation under limited data. To address discriminator overfitting in limited data scenarios, where data is scarce or privacy-sensitive, previous methods have employed data augmentation techniques such as DA [120], ADA [39], MaskedGAN [29], FakeCLR [55] and InsGen [108] to expand the data diversity. Approaches [48, 122], KDDLGAN [15], and TransferGAN [102], leverage knowledge from models trained on extensive datasets to enhance performance. However, these approaches may risk leaking augmentation artifacts [39, 72, 113] or misusing pre-trained knowledge [22, 50, 106]. Alternatives such as LeCam loss [97], GenCo [14] and the gradient norm reduction of DigGAN [20] aim to balance real and fake distributions. Our approach uniquely combines generalization benefits from BN with improved stability in GAN training, offering an effective and distinct solution to regularizing discriminator.

GAN Generalization. Deviating from conventional methods that link the generalization of GANs [32, 115] with the Rademacher complexity [6] of neural networks [116], we introduce a new error bound that highlights the need for reducing discrepancies between seen and unseen data for enhanced generalization. This bound is further refined using the so-called non-vacuous PAC-Bayesian theory [10], focusing on discriminator weight gradients for a practical GAN generalization improvement.

Normalization. Batch Normalization (BN) [31] and its variants such as Group Normalization (GN) [104], Layer Normalization (LN) [5], Instance Normalization (IN) [98] have been pivotal in normalizing latent features to improve training. BN, in particular, is renowned for its role in improving generalization across various tasks [7, 36, 62, 80]. However, its application in discriminator design, especially under limited data scenarios where generalization is crucial, remains underexplored. Several BN modifications, such as RMSNorm [111], GraphNorm [9], PowerNorm [85], MBN [107] and EvoNorm [58] have been proposed to address issues such as the gradient explosion in transformers [99] or information loss in graph learning, often by altering or removing the centering step. Our work stands out in GAN discriminator design by linking centering, scaling, and gradient issues in GAN training. Our innovative solution not only mitigates the gradient explosion but also retains the benefits of BN, offering a robust solution for GAN training.

3. Method

We begin by linking GAN generalization with the gradient of discriminator weights, motivating the use of BN for generalization and identifying gradient issues in BN. We then introduce CHAIN, a design that tackles these gradient issues while retaining benefits of BN. Lastly, we present a theoretical justification for CHAIN, underscoring its efficacy in improving generalization and training stability.

3.1. Generalization Error of GAN

The goal of GAN is to train a generator capable of deceiving a discriminator by minimizing the integral probability metric (IPM) [67], typically with the assumption of infinite real and fake distributions (μ, ν). However, in real-world scenarios, we are usually confined to working with a finite real dataset $\hat{\mu}_n$ of size n. This limitation restricts the optimization of GAN to the empirical loss as discussed in [115]:

$$\inf_{\nu \in \mathcal{G}} \{d_{\mathcal{H}}(\hat{\mu}_n, \nu) := \sup_{h \in \mathcal{H}} \{\mathbb{E}_{\boldsymbol{x} \sim \hat{\mu}_n}[h(\boldsymbol{x})] - \mathbb{E}_{\tilde{\boldsymbol{x}} \sim \nu}[h(\tilde{\boldsymbol{x}})]\}\}, \quad (1)$$

where $\boldsymbol{x}$ and $\tilde{\boldsymbol{x}}$ are real and fake samples. Function sets of discriminator and generator, $\mathcal{H}$ and $\mathcal{G}$, are typically parameterized as neural network classes $\mathcal{H}_{nn} := \{h(\cdot; \boldsymbol{\theta}_d) : \boldsymbol{\theta}_d \in \boldsymbol{\Theta}_d\}$ and $\mathcal{G}_{nn} := \{g(\cdot; \boldsymbol{\theta}_g) : \boldsymbol{\theta}_g \in \boldsymbol{\Theta}_g\}$. Given the varied divergence [84, 115] encompassed by the IPM and the variability of discriminator loss function $\phi(\cdot)$ across different tasks and architectures, we integrate it with the discriminator D for simplified analysis [3, 4, 115], yielding $h(\cdot) := \phi(D(\cdot))$. This integration streamlines the alternating optimization process between the discriminator and the generator:

$$\begin{cases} \mathcal{L}_D := \min_{\boldsymbol{\theta}_d} \mathbb{E}_{\tilde{\boldsymbol{x}} \sim \nu_n}[h(\tilde{\boldsymbol{x}}; \boldsymbol{\theta}_d)] - \mathbb{E}_{\boldsymbol{x} \sim \hat{\mu}_n}[h(\boldsymbol{x}; \boldsymbol{\theta}_d)] \\ \mathcal{L}_G := \min_{\boldsymbol{\theta}_g} -\mathbb{E}_{\boldsymbol{z} \sim p_{\boldsymbol{z}}}[h(g(\boldsymbol{z}; \boldsymbol{\theta}_g))], \end{cases} \quad (2)$$

where $\boldsymbol{z} \sim p_z$ represents the noise input to the generator and it is assumed that ν_n minimizes $d_{\mathcal{H}}(\hat{\mu}_n, \nu)$ to a precision $\epsilon \geqslant 0$, implying that $d_{\mathcal{H}}(\hat{\mu}_n, \nu_n) \leqslant \inf_{\nu \in \mathcal{G}} d_{\mathcal{H}}(\hat{\mu}_n, \nu) + \epsilon$.

To evaluate how closely the generator distribution ν_n approximates the unknown infinite distribution μ, we draw on work of Ji *et al.* [32] who extended Theorem 3.1 in [115] by considering the limited access to both real and fake images.

Lemma 3.1 *(Partial results of Theorem 1 in [32].) Assume the discriminator set $\mathcal{H}$ is even, i.e., $h \in \mathcal{H}$ implies $-h \in \mathcal{H}$, and $\|h\|_\infty \leqslant \Delta$. Let $\hat{\mu}_n$ and $\hat{\nu}_n$ be empirical measures of μ and ν_n with size n. Denote $\nu_n^* = \inf_{\nu \in \mathcal{G}} d_{\mathcal{H}}(\hat{\mu}_n, \nu)$. The generalization error of GAN, defined as $\epsilon_{gan} := d_{\mathcal{H}}(\mu, \nu_n) - \inf_{\nu \in \mathcal{G}} d_{\mathcal{H}}(\mu, \nu)$, is bounded as:*

$$\epsilon_{gan} \leqslant 2\Big(\sup_{h \in \mathcal{H}} \big|\mathbb{E}_\mu[h] - \mathbb{E}_{\hat{\mu}_n}[h]\big| + \sup_{h \in \mathcal{H}} \big|\mathbb{E}_{\nu_n^*}[h] - \mathbb{E}_{\hat{\nu}_n}[h]\big|\Big)$$
$$= 2d_{\mathcal{H}}(\mu, \hat{\mu}_n) + 2d_{\mathcal{H}}(\nu_n^*, \hat{\nu}_n). \quad (3)$$

Lemma 3.1 (proof in §B.1) indicates that GAN generalization can be improved by reducing the divergence between real training and unseen data, as well as observed and unobserved fake distributions. Given that the ideal ν_n^* aligns with the observed real data $\hat{\mu}_n$, Lemma 3.1 also implies the importance of narrowing the gap between observed fake and real data. This explains why prior efforts [12, 20, 27, 33, 97] focusing on diminishing the real-fake distribution divergence help limit overfitting. However, caution is needed to avoid an excessive reduction, as this makes the discriminator struggle to differentiate real and fake data [115].

To determine a practical generalization error for GAN parameterized as neural networks, we adopt the PAC Bayesian theory [10] to derive the generalization error bound on GANs. Integrating the analytical process of Theorem 1 from [21], we formulate the following proposition.

Proposition 3.1 *Utilizing notations from Lemma 3.1, we define ϵ_{gan}^{nn} as the generalization error of GAN parameterized as neural network classes. Let $\boldsymbol{\nabla}_{\boldsymbol{\theta}_d}$ and $\boldsymbol{H}_{\boldsymbol{\theta}_d}$ represent the gradient and Hessian matrix of discriminator h evaluated at $\boldsymbol{\theta}_d$ over real training data $\hat{\mu}_n$, and $\widetilde{\boldsymbol{\nabla}}_{\boldsymbol{\theta}_d}$ and $\widetilde{\boldsymbol{H}}_{\boldsymbol{\theta}_d}$ over observed fake data $\hat{\nu}_n$. Denoting $\lambda_{max}^{\boldsymbol{H}}$ and $\lambda_{max}^{\widetilde{\boldsymbol{H}}}$ as the largest eigenvalues of $\boldsymbol{H}_{\boldsymbol{\theta}_d}$ and $\widetilde{\boldsymbol{H}}_{\boldsymbol{\theta}_d}$, respectively, and for any $\omega > 0$, the generalization error is bounded as:*

$$\epsilon_{gan}^{nn} \leqslant 2\omega\big(\|\boldsymbol{\nabla}_{\boldsymbol{\theta}_d}\|_2 + \|\widetilde{\boldsymbol{\nabla}}_{\boldsymbol{\theta}_d}\|_2\big) + 4R\Big(\frac{\|\boldsymbol{\theta}_d\|_2^2}{\omega^2}, \frac{1}{n}\Big)$$
$$+ \omega^2\big(|\lambda_{max}^{\boldsymbol{H}}| + |\lambda_{max}^{\widetilde{\boldsymbol{H}}}|\big), \quad (4)$$

where $R\Big(\frac{\|\boldsymbol{\theta}_d\|_2^2}{\omega^2}, \frac{1}{n}\Big)$, a term related to discriminator weights norm, is inversely related to the data size n.

Prop. 3.1 (proof in §B.2) suggests several strategies to lower the generalization error of GANs. These include increasing data size (n), implementing regularization to decrease weight norm of the discriminator and the largest eigenvalues in Hessian matrices, and crucially, reducing the gradient norm of discriminator weights. Although this proposition is specific to GANs, the concept of regularizing weight gradient norms aligns with findings in other studies [60, 91, 93, 100, 117, 121], which emphasize that reducing weight gradients can smooth the loss landscape, thereby enhancing generalization of various deep learning tasks.

3.2. Motivation and the Batch Normalization Issues

Leveraging the insight of Lemma 3.1 that reducing real-fake divergence boosts generalization, we propose applying Batch Normalization in the discriminator to normalize real and fake data *in separate batches*. As depicted in Figure 1, normalizing real and fake data in separate batches via the centering and scaling steps aligns their statistical moments to lower the real-fake divergence. Moreover, BN's ability to reduce loss sharpness, as indicated by the maximum eigenvalue of Hessian [36, 62, 80], supports the motivation that using BN in the discriminator favors GAN generalization. Yet, incorporating BN poses a risk of gradient explosion.

For a specific layer in a network, consider $\boldsymbol{A} \in \mathbb{R}^{B \times d}$ as the feature input, where B is the batch size and d is the feature size. For brevity, we exclude bias and focus on layer weights $\boldsymbol{W} \in \mathbb{R}^{d \times d}$. In line with studies [9, 61, 80, 85], we also omit the affine transformation step for theoretical clarity, as it does not impact the theoretical validity, and does not change our method. The processing of features
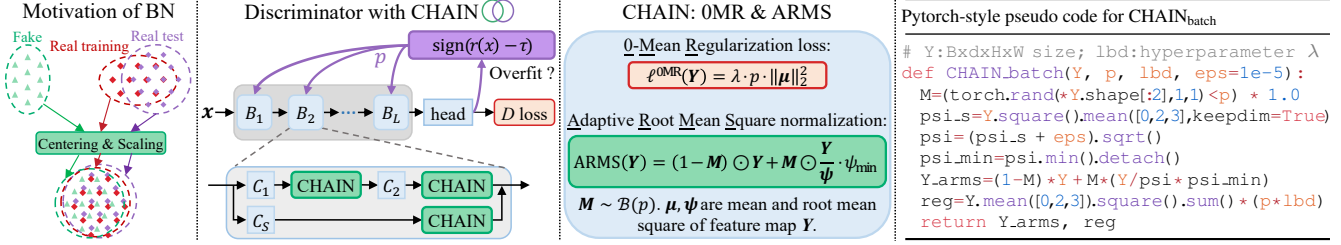
Figure 1. Motivation of using BN, discriminator with CHAIN, modules in CHAIN and the Pytorch-style pseudo-code for CHAIN$_{\text{batch}}$.

through the weights and the Batch Normalization contains:

$$\text{Linear transformation:} \quad \boldsymbol{Y} = \boldsymbol{A}\boldsymbol{W} \qquad (5)$$

$$\text{Centering:} \quad \overset{c}{\boldsymbol{Y}} = \boldsymbol{Y} - \boldsymbol{\mu} \qquad (6)$$

$$\text{Scaling:} \quad \overset{s}{\boldsymbol{Y}} = \overset{c}{\boldsymbol{Y}}/\boldsymbol{\sigma}. \qquad (7)$$

Using these notations, we identify the gradient issues in the centering and scaling steps, as detailed below.

Theorem 3.1 *(The issue of the centering step.) Consider* $\boldsymbol{y}_1, \boldsymbol{y}_2$ *as i.i.d. samples from a symmetric distribution centered at* $\boldsymbol{\mu}$*, where the presence of* $\boldsymbol{y}$ *implies* $2\boldsymbol{\mu}-\boldsymbol{y}$ *is also included (important in proof). After the centering step,* $\overset{c}{\boldsymbol{y}}_1, \overset{c}{\boldsymbol{y}}_2$ *are i.i.d. samples from the centered distribution. The expected cosine similarity between these samples is given by:*

$$\mathbb{E}_{\boldsymbol{y}_1,\boldsymbol{y}_2}\big[\cos(\boldsymbol{y}_1,\boldsymbol{y}_2)\big] \geqslant \mathbb{E}_{\overset{c}{\boldsymbol{y}}_1,\overset{c}{\boldsymbol{y}}_2}\big[\cos(\overset{c}{\boldsymbol{y}}_1,\overset{c}{\boldsymbol{y}}_2)\big] = 0. \quad (8)$$

Theorem 3.1 (proof in §B.3) states that after centering by batch normalization, the expected cosine similarity between features drops to zero. This implies that features which are similar in early network layers diverge significantly in the later layers, suggesting that minor perturbations in early layers have the risk to lead to abrupt changes in later layers. Consequently, such an effect implies large gradients.

Theorem 3.2 *(The issue of the scaling step.) The scaling step, defined in Eq. 7, can be expressed as matrix multiplication* $\overset{s}{\boldsymbol{Y}} = \overset{c}{\boldsymbol{Y}} diag(1/\boldsymbol{\sigma})$*. The Lipschitz constant w.r.t. the 2-norm of the scaling step is:*

$$\left\| diag\left(\frac{1}{\boldsymbol{\sigma}}\right) \right\|_{lc} = \frac{1}{\sigma_{min}}, \qquad (9)$$

where $\sigma_{min} = \min_c \sigma_c$ *represents the minimum value in* $\boldsymbol{\sigma}$.

Theorem 3.2 (proof in §B.4) establishes that the Lipschitz constant for the scaling step in batch normalization is inversely proportional to $\sigma_{\min}$. This means if $\sigma_{\min}$ is less than 1, the Lipschitz constant exceeds 1. Given the emphasis placed by previous studies [3, 13, 25, 56, 65] on the importance of lowering the Lipschitz constant in the discriminator, it follows that without a Lipschitz continuity constraint on the scaling step, discriminators employing batch normalization are prone to gradient explosion. See [24] for further insights into the Lipschitz constant of batch normalization concerning the affine transformation step.

3.3. CHAIN ◯◯

To harness the generalization benefits of BN while sidestepping its gradient issue in GAN discriminator, we introduce CHAIN. Our modification involves replacing the centering step (as in Eq. 6) with zero-mean regularization, substituting the scaling step (as in Eq. 7) with Lipschitz continuity constrained root mean square normalization, and removing the affine transformation step for enhanced performance.

We start by calculating the mean $\boldsymbol{\mu}$ and the root mean square $\boldsymbol{\psi}$ across batch and spatial dimensions for features $\boldsymbol{Y} \in \mathbb{R}^{B \times d \times H \times W}$ in a discriminator layer as follows:

$$\mu_c = \frac{1}{B \times H \times W} \sum_b^B \sum_h^H \sum_w^W Y_{b,c,h,w}, \qquad (10)$$

$$\psi_c = \sqrt{\left(\frac{1}{B \times H \times W} \sum_b^B \sum_h^H \sum_w^W Y_{b,c,h,w}^2\right) + \epsilon}, \qquad (11)$$

where ϵ is a small constant to avoid division by 0. The term $Y_{b,c,h,w}$ denotes the (b,c,h,w)-th entry in $\boldsymbol{Y}$ while μ_c and ψ_c represent the c-th element in $\boldsymbol{\mu}$ and $\boldsymbol{\psi}$, respectively.

To achieve a soft zero-mean effect akin to the centering step in Eq. 6 while also avoid its gradient issue, we adopt 0-Mean Regularization (0MR) as follows:

$$\ell^{\text{0MR}}(\boldsymbol{Y}) = \lambda \cdot p \cdot \|\boldsymbol{\mu}\|_2^2, \qquad (12)$$

where λ is a hyperparameter and $p \in [0,1]$ adaptively controls the regularization strength. The term ℓ^{0MR} for layers applying CHAIN is added to the loss of the discriminator.

The root mean square normalization, constrained by Lipschitz condition, is defined as follows:

$$\hat{\boldsymbol{Y}} = \check{\boldsymbol{Y}} \cdot \psi_{\min}, \quad \text{with} \quad \check{\boldsymbol{Y}} = \frac{\boldsymbol{Y}}{\boldsymbol{\psi}}. \qquad (13)$$

where $\psi_{\min} = \min_c \psi_c$ is the minimum in $\boldsymbol{\psi}$, severing to constrain the Lipschitz constant of the normalization to 1.

Normalized features are then adaptively interpolated with unnormalized features to balance discrimination and generalization, as emphasized in [115], leading to the Adaptive Root Mean Square normalization (ARMS):

$$\text{ARMS}(\boldsymbol{Y}) = (1-\boldsymbol{M}) \odot \boldsymbol{Y} + \boldsymbol{M} \odot \frac{\boldsymbol{Y}}{\boldsymbol{\psi}} \cdot \psi_{\min}, \qquad (14)$$

where $\odot$ is the element-wise multiplication after expanding the left-side matrix to $B \times d \times H \times W$ dimension. The matrix $M \in \mathbb{R}^{B \times d}$, with values from a Bernoulli distribution $\mathcal{B}(p)$ with $p \in [0,1]$, controls the interpolation ratio.

To mitigate discriminator overfitting, we allow the factor p, controlling both the regularization strength in Eq. 12 and the interpolation ratio in Eq. 14, to be adaptive based on the discriminator output. Specifically, we calculate the expectation of discriminator output $r(\boldsymbol{x}) = \mathbb{E}[\text{sign}(D(\boldsymbol{x}))]$ w.r.t. real samples $\boldsymbol{x}$ and assess $\varepsilon = \text{sign}(r(\boldsymbol{x}) - \tau) \in \{-1, 0, 1\}$ against a predefined threshold τ. Exceeding τ suggests potential overfitting, as indicated by previous studies [33, 39]. We then adjust p using $p_{t+1} = p_t + \Delta_p \cdot \varepsilon$ with a small Δ_p.

To limit the dependency on the minibatch size in high-resolution GAN training across multiple GPUs, we adopt running cumulative forward/backward statistics, inspired by [30, 85, 107]. We contrast $\text{CHAIN}_{\text{batch}}$, using batch statistics, with CHAIN that applies running cumulative statistics. $\text{CHAIN}_{\text{batch}}$ is elegantly coded as shown in Figure 1, whereas implementation for CHAIN is detailed in §D.1.

As outlined in Figure 1, CHAIN is integrated after convolutional layers $c \in \{C_1, C_2, C_S\}$ within the discriminator blocks B_l for $l \in \{1, ..., L\}$. By applying CHAIN separately on real and fake data, Eq. 12 naturally reduces divergence across seen/unseen and observed real/fake data, consistent with Lemma 3.1. Additionally, Eq. 14 effectively lowers weight gradients of discriminator, aligning with Prop. 3.1.

3.4. Theoretical analysis for CHAIN ◎

Although CHAIN is straightforward and easy to implement, its importance in GAN training is substantial. We provide analyses of how CHAIN modulates gradients, underlining its critical role in enhancing GAN performance.

Theorem 3.3 (*CHAIN reduces the gradient norm of weights/latent features.*) *Denote the loss of discriminator with CHAIN as $\mathcal{L}$, and the resulting batch features as $\dot{\boldsymbol{Y}}$. Let $\check{\boldsymbol{y}}_c \in \mathbb{R}^B$ be c-th column of $\check{\boldsymbol{Y}}$, $\Delta \boldsymbol{y}_c, \Delta \dot{\boldsymbol{y}}_c \in \mathbb{R}^B$ be the c-th column of gradient $\frac{\partial \mathcal{L}}{\partial \boldsymbol{Y}}, \frac{\partial \mathcal{L}}{\partial \dot{\boldsymbol{Y}}}$. Denote $\Delta \boldsymbol{w}_c$ as the c-th column of weight gradient $\frac{\partial \mathcal{L}}{\partial \boldsymbol{W}}$ and λ_{max} as the largest eigenvalue of pre-layer features $\boldsymbol{A}$. Then we have:*

$$\|\Delta \boldsymbol{y}_c\|_2^2 \leqslant \|\Delta \dot{\boldsymbol{y}}_c\|_2^2 \left(\frac{(1-p)\psi_c + p\psi_{min}}{\psi_c} \right)^2$$
$$- \frac{2(1-p)p\psi_{min}}{B\psi_c} (\Delta \dot{\boldsymbol{y}}_c^T \check{\boldsymbol{y}}_c)^2, \qquad (15)$$

$$\|\Delta \boldsymbol{w}_c\|_2^2 \leqslant \lambda_{max}^2 \|\Delta \boldsymbol{y}_c\|_2^2. \qquad (16)$$

Theorem 3.3 (proof in §B.5) reveals that CHAIN significantly modulates gradient norms in GAN training. It states that the squared gradient norm of normalized output is rescaled by $\left(\frac{(1-p)\psi_c + p\psi_{min}}{\psi_c} \right)^2 \leqslant 1$, minus a non-negative term where $(\Delta \dot{\boldsymbol{y}}_c^T \check{\boldsymbol{y}}_c)^2 \geqslant 0$. Considering that $\|\Delta \boldsymbol{y}_c\|_2^2 \geqslant 0$,

CHAIN effectively reduces the gradient norm of latent features. Moreover, given that the eigenvectors of $\text{diag}(1/\boldsymbol{\sigma})$ and pre-layer features $\boldsymbol{A}$ are less likely to align, using CHAIN with a Lipschitz constant of exactly 1 before $\boldsymbol{A}$ further reduces λ_{max}. This dual action not only stabilizes GAN training by reducing latent feature gradients but also improves generalization by lowering the weight gradients.

We additionally present theory and experiments in §C to justify the decorrelation effect of the stochastic $\boldsymbol{M}$ design.

4. Experiments

We conduct experiments on CIFAR-10/100 [47] using Big-GAN [8] and OmniGAN [123], as well as on ImageNet [19] using BigGAN for conditional image generation. We evaluate our method on 5 low-shot datasets [120], which include 100-shot Obama/Panda/Grumpy Cat and AnimalFace Dog/Cat [90], using StyleGAN2 [40]. Additionally, we assess our method on 7 high-resolution few-shot datasets, including Shells, Skulls, AnimeFace [11], Pokemon, Art-Painting, and two medical datasets BreCaHAD [1], MessidorSet1 [18], building upon FastGAN [57]. For comparative purposes, methods involving massive augmentation include DA [120] and ADA [39], termed "MA" in [14], are also included in our evaluation.

Datasets. CIFAR-10 has $50K/10K$ training/testing images in 10 categories at 32×32 resolution, while CIFAR-100 has 100 classes. ImageNet compreises $1.2M/50K$ training/validation images across 1K categories. Following [15, 29], we center-crop and downscale its images to 64×64 resolution. The five low-shot datasets include 100-shot Obama/Panda/Grumpy Cat images, along with AnimalFace (160 cats and 389 dogs) images at 256×256 resolution. The seven few-shot datasets, Shells, Skulls, AnimeFace, Pokemon, Artpainting, BreCaHAD, MessidorSet1, vary from 64 to 1000 images, each at a high 1024×1024 resolution. Following [120], we augment all datasets with x-flips.

Evaluation metrics. We generate 50K images for CIFAR-10/100 and ImageNet to calculate Inception Score (IS) [79] and Fréchet Inception Distance (FID) [26]. For these datasets, tFID is calculated by comparing $50K$ generated images against all training images. Additionally, we compute vFID for CIFAR-10/100 and ImageNet between $10K/50K$ fake and real testing/validation images. For the five low-shot and seven few-shot datasets, FID is measured between $5K$ fake images and the full dataset. Following [20, 55, 120], we run five trails for methods employing CHAIN, reporting average results and omitting standard deviations for clarity, as they fall below 1%. Implementation details and generated images are available in §D.2 and §G.

4.1. Comparison with sate-of-the-art methods

Results on CIFAR-10/100 w/ BigGAN/OmniGAN. Table 1 demonstrates that our method achieves state-of-the-art re-

Table 1. Comparing CIFAR-10/100 results with varying data percentages, using CHAIN *vs.* without it. MA: Massive Augmentation.

Method	MA	CIFAR-10									CIFAR-100								
		10% data			20% data			100% data			10% data			20% data			100% data		
		IS↑	tFID↓	vFID↓	IS↑	tFID↓	vFID↓	IS↑	tFID↓	vFID↓	IS↑	tFID↓	vFID↓	IS↑	tFID↓	vFID↓	IS↑	tFID↓	vFID↓
BigGAN(d=256)	×	8.24	31.45	35.59	8.74	16.20	20.27	9.21	5.48	9.42	7.58	50.79	55.04	9.94	25.83	30.79	11.02	7.86	12.70
+DA	✓	8.65	18.35	22.04	8.95	9.38	13.26	9.39	4.47	8.58	8.86	27.22	31.80	9.73	16.32	20.88	10.91	7.30	11.99
+DigGAN+DA	✓	–	–	17.87	–	–	13.01	–	–	8.49	–	–	24.59	–	–	19.79	–	–	11.63
+LeCam	×	8.44	28.36	33.65	8.95	11.34	15.25	9.45	4.27	8.29	8.14	41.51	46.43	10.05	20.81	25.77	**11.41**	6.82	11.54
+CHAIN	×	8.63	12.02	16.00	8.98	8.15	12.12	9.49	4.18	8.21	10.04	13.13	18.00	10.15	11.58	16.38	11.16	6.04	10.84
LeCam+DA	✓	8.81	12.64	16.42	9.01	8.53	12.47	9.45	4.32	8.40	9.17	22.75	27.14	10.12	15.96	20.42	11.25	6.45	11.26
+KDDLGAN	✓	–	–	13.86	–	–	11.15	–	–	8.19	–	–	22.40	–	–	18.70	–	–	10.12
+CHAIN	✓	8.96	8.54	12.51	9.27	5.92	9.90	9.52	3.51	7.47	10.11	12.69	17.49	10.62	9.02	13.75	11.37	5.26	9.85
OmniGAN(d=1024)	×	6.69	53.02	57.68	8.64	36.75	41.17	10.01	6.92	10.75	6.91	60.46	64.76	10.14	40.59	44.92	12.73	8.36	13.18
+DA	✓	8.99	19.45	23.48	9.49	13.45	17.27	10.13	4.15	8.06	10.01	30.68	34.94	11.35	17.65	22.37	12.94	7.41	12.08
+ADA	✓	7.86	40.05	44.01	9.41	27.04	30.58	10.24	4.95	9.06	8.95	44.65	49.08	12.07	13.54	18.20	13.07	6.12	10.79
+CHAIN	×	9.85	6.81	10.64	9.92	4.78	8.68	10.26	2.63	6.64	12.05	13.12	17.87	12.65	9.61	14.57	13.88	4.09	9.00
+ADA+CHAIN	✓	10.10	6.22	10.09	10.26	3.98	7.93	10.31	2.22	6.28	12.70	9.49	14.23	12.98	7.02	11.87	13.98	4.02	8.93

Table 2. Comparing ImageNet results with varying training data percentages, using our method *vs.* without it.

Method	MA	2.5% data					5% data					10% data				
		50k fake imgs			10k fake imgs		50k fake imgs			10k fake imgs		50k fake imgs			10k fake imgs	
		IS↑	tFID↓	vFID↓	IS↑	tFID↓	IS↑	tFID↓	vFID↓	IS↑	tFID↓	IS↑	tFID↓	vFID↓	IS↑	tFID↓
BigGAN	×	8.61	101.62	100.09	8.43	103.40	6.27	90.32	88.01	6.28	93.26	12.44	50.75	49.84	12.17	52.90
+DA	✓	11.07	86.07	84.48	10.82	87.30	9.15	68.61	66.85	9.01	70.86	16.30	35.16	34.01	15.78	37.76
+ADA	✓	7.93	67.84	66.55	7.86	70.01	11.56	47.56	46.25	11.28	50.15	14.82	31.75	30.68	14.68	34.35
+MaskedGAN	✓	–	–	–	12.68	38.62	–	–	–	12.85	35.70	–	–	–	13.34	26.51
+ADA+KDDLGAN	✓	–	–	–	14.65	28.79	–	–	–	14.06	22.35	–	–	–	14.14	20.32
+CHAIN	×	14.68	30.66	29.32	14.25	32.93	17.34	21.13	19.95	16.64	23.62	20.45	14.70	13.84	19.16	17.34
+ADA+CHAIN	✓	16.57	23.01	21.90	15.70	25.98	19.15	16.14	15.17	18.17	18.77	22.04	12.91	12.17	21.16	15.83

Table 3. FID↓ on seven few-shot datasets, comparing w/ *vs.* w/o CHAIN, based on mean and standard deviation from 5 trails.

Method	sec/$k$img	Shells	Skulls	AnimeFace	BreCaHAD	MessidorSet1	Pokemon	ArtPainting
		64 imgs	97 imgs	120 imgs	162 imgs	400 imgs	833 imgs	1000 imgs
FastGAN [57]	34.40	$138.50_{\pm3.65}$	$97.87_{\pm1.05}$	$54.05_{\pm0.55}$	$63.83_{\pm1.36}$	$38.33_{\pm4.30}$	$45.70_{\pm1.65}$	$43.21_{\pm0.14}$
FreGAN [109]	44.75	$123.75_{\pm4.92}$	$84.58_{\pm0.50}$	$49.09_{\pm0.58}$	$\mathbf{57.87_{\pm0.55}}$	$34.61_{\pm2.48}$	$39.09_{\pm1.35}$	$43.14_{\pm0.69}$
FastGAN$-D_{\mathrm{big}}$	**32.79**	$171.35_{\pm6.91}$	$165.64_{\pm11.47}$	$76.02_{\pm5.37}$	$68.63_{\pm1.18}$	$37.38_{\pm1.73}$	$53.48_{\pm3.55}$	$43.04_{\pm0.24}$
FastGAN$-D_{\mathrm{big}}$+CHAIN	35.94	$\mathbf{78.62_{\pm1.21}}$	$\mathbf{82.47_{\pm2.82}}$	$\mathbf{46.27_{\pm0.36}}$	$58.98_{\pm1.59}$	$\mathbf{28.76_{\pm1.52}}$	$\mathbf{31.94_{\pm2.82}}$	$\mathbf{38.83_{\pm0.49}}$

sults on CIFAR-10/100, surpassing even KDDLGAN [15], which leverages knowledge from CLIP [75].

Results on ImageNet with BigGAN. Maintaining consistency with established benchmarks in [15, 29] (using 10K generated images for IS and tFID), Table 2 demonstrates the superiority of CHAIN, outperforming all leading models and underscoring its exceptional performance.

Results on the seven few-shot datasets with FastGAN. FastGAN [57], known for its memory and time efficiency, yields desirable results on 1024×1024 resolution within one-day training on a single GPU. To integrate our method, we swapped large FastGAN discriminator with BigGAN and removed the small discriminator due to multidimensional output of FastGAN being unsuitable for adjusting our p. This new variant, named FastGAN$-D_{\mathrm{big}}$, is described in Figure 9 of §D.2. Table 3 demonstrates the superior performance of CHAIN on seven 1024×1024 low-shot datasets.

Results on the five low-shot datasets w/ StyleGAN2. Table 4 presents a comparison of CHAIN with other baselines, clearly demonstrating that CHAIN achieves the best results.

4.2. Experimental analysis

Gradient analysis for centering step. Figure 2 illustrates the mean cosine similarity among pre-activation features in the discriminator and the gradient norm of the feature extractor output w.r.t. input for OmniGAN, OmniGAN+0C (using Eq. 6 centering), and OmniGAN+A0C (adaptive interpolation of centered and uncentered features). The near-zero mean cosine similarity in OmniGAN+0C and OmniGAN+A0C corroborates Theorem 3.1, indicating that centering leads to feature difference in later layers and amplifying the gradient effect, as seen in Figure 2b. This observation supports the decision to modify the centering step.

Gradient analysis for scaling step. Figure 3a shows gradient norms of the discriminator output w.r.t. the input and effective rank (eRank) [78] for various models. The CHAIN$_{-LC}$ variant (CHAIN w/o Lipschitz constraint) exhibits gradient explosion, confirming Theorem 3.2. While CHAIN$_{+0C}$ avoids gradient explosion, its centering step causes abrupt feedback changes to the generator, leading to the dimensional collapse [34, 44, 63, 96], evidenced by rank

Table 4. FID↓ of unconditional image generation with StyleGAN2 on five low-shot datasets. † marks a generator pre-trained on full FFHQ [38] dataset, ‡ signifies a pre-trained CLIP [75] model. "MA" means Massive Augmentation, "PT" refers to Pretrained.

Method	MA	PT	100-shot			Animal Face	
			Obama	GrumpyCat	Panda	Cat	Dog
StyleGAN2	×	×	80.20	48.90	34.27	71.71	131.90
+CHAIN	×	×	**28.72**	**27.21**	**9.51**	**38.93**	**53.27**
AdvAug [12]	✓	×	52.86	31.02	14.75	47.40	68.28
ADA	✓	×	45.69	26.62	12.90	40.77	56.83
DA	✓	×	46.87	27.08	12.06	42.44	58.85
ADA+DigGAN	✓	×	41.34	26.75	—	37.61	59.00
LeCam	✓	×	33.16	24.93	10.16	34.18	54.88
GenCo	✓	×	32.21	17.79	9.49	30.89	49.63
InsGen	✓	×	32.42	22.01	9.85	33.01	44.93
MaskedGAN	✓	×	33.78	20.06	8.93	—	—
FakeCLR	✓	×	26.95	19.56	8.42	26.34	42.02
TransferGAN †	✓	✓	39.85	29.77	17.12	49.10	65.57
KDDLGAN ‡	✓	✓	29.38	19.65	8.41	31.89	50.22
AugSelf [27]	✓	×	26.00	19.81	8.36	30.53	48.19
ADA+CHAIN	✓	×	**20.94**	17.61	7.50	19.74	39.10
DA+CHAIN	✓	×	22.87	**17.57**	**6.93**	**19.58**	**30.88**

(a) Mean cosine similarity. (b) Gradient norm.
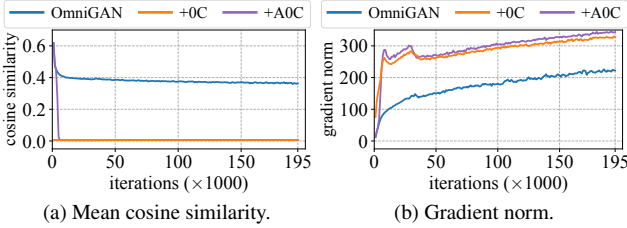
Figure 2. (a) Mean cosine similarity of discriminator pre-activation features, and (b) gradient norm of the feature extractor w.r.t. the input are evaluated for OmniGAN, OmniGAN+0C (using the centering step in Eq. 6), and OmniGAN+A0C (adaptive interpolation between centered and uncentered features). Evaluation conducted on 10% CIFAR-10 data with OmniGAN ($d = 256$).

deficiencies in Figure 3b. In contrast, CHAIN maintains smaller gradient than OmniGAN, aligning with the analysis in Theorem 3.3 w.r.t. reducing gradient in latent features.

Generalization analysis. Figures 4c and 5c show that CHAIN achieves smaller gradient norm of discriminator output w.r.t. weight, supporting the assertion of Theorem 3.3 on reducing weight gradient. This leads to a lower generalization error, as per Prop. 3.1 and Lemma 3.1, evidenced in Figures 4b and 5b. Here, compared to the baseline, CHAIN maintains a smaller discrepancy in discriminator output between real and test images, as well as discrepancy between real and fake images, indicating the effectiveness of CHAIN in improving GAN generalization.

4.3. Ablation studies

Ablation for CHAIN design. Table 5 provides quantitative evidence supporting the design of our method. The inferior results of CHAIN$_{-0MR}$ and CHAIN$_{-ARMS}$ highlight the significance of the 0MR and ARMS modules. Poorer per-

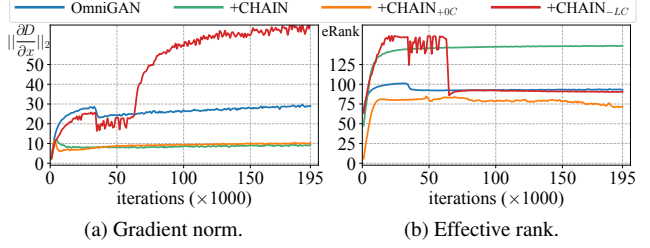(a) Gradient norm. (b) Effective rank.

Figure 3. (a) Gradient norm of discriminator output w.r.t. input during training, and (b) effective rank [78] of the pre-activation features in discriminator, are evaluated on 10% CIFAR-10 data with OmniGAN (d=256). CHAIN$_{+0C}$: CHAIN w/ the centering step. CHAIN$_{-LC}$: CHAIN w/o the Lipschitzness constraint.

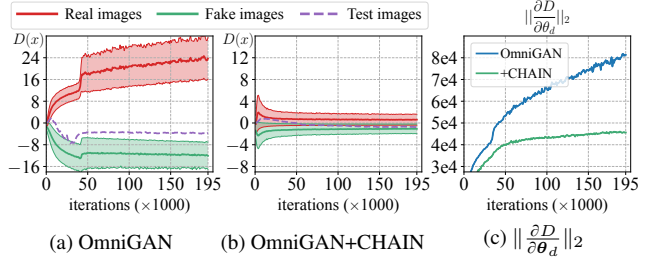(a) OmniGAN (b) OmniGAN+CHAIN (c) $\| \frac{\partial D}{\partial \theta_d} \|_2$

Figure 4. The discriminator output w.r.t. real, fake and test images using (a) OmniGAN, (b) OmniGAN+CHAIN, and (c) the gradient norm of the discriminator output w.r.t. discriminator weights on 10% CIFAR-10 using OmniGAN ($d = 256$). Note the y-axis in (b) is scaled for clearer visualization.

(a) BigGAN (b) BigGAN+CHAIN (c) $\| \frac{\partial D}{\partial \theta_d} \|_2$
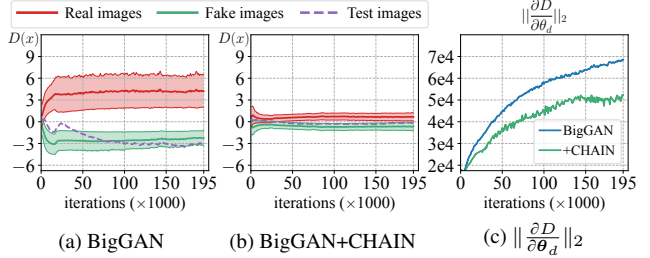
Figure 5. The discriminator output w.r.t. real, fake and test images of (a) BigGAN, (b) BigGAN+CHAIN, along with (c) the gradient norm of the discriminator output w.r.t. discriminator weights on 10% CIFAR-100 with BigGAN ($d = 256$).

formance of CHAIN$_{+0C}$ underscores the need to omit the centering step. The notably worse outcomes of CHAIN$_{-LC}$ emphasize the importance of the Lipschitzness constraint. CHAIN$_{batch}$ underperforming suggests the advantage of using running cumulative statistics. The suboptimal performance of CHAIN$_{Dtm.}$ validate the stochastic M design (Eq. 14), while marginally poorer results of CHAIN$_{+0MR_g}$ indicate limited benefits of applying 0MR in generator training.

Ablation of each factor. Figure 6 explores the impact of applying CHAIN at different points and varying the hyperparameters λ, τ. In Figure 6a, optimal performance is achieved by placing CHAIN after all convolutional layers. Figure 6b demonstrates that employing our approach across

Table 5. Ablation studies. 0C: using centering. A0C: adaptively interpolating centered and uncentered features. CHAIN_{-0MR}: CHAIN w/o 0-mean regularization (0MR, Eq. 12). CHAIN_{-ARMS}: CHAIN w/o the adaptive root mean square normalization (ARMS, Eq. 14). CHAIN_{+0C}: CHAIN w/ centering. CHAIN_{-LC}: CHAIN w/o the Lipschitzness constraint. CHAIN_{batch}: replacing the cumulative with batch statistics. $\text{CHAIN}_{Dtm.}$: replacing the stochastic M in Eq. 14 with deterministic p. CHAIN_{+0MR_g}: Applying ℓ^{0MR} in generator training. $\text{CHAIN}_{+Aff.}$: applying learnable affine transformation. ADrop: adaptive dropout.

| Method	10% CIFAR-10 OmniGAN ($d=256$)			10% CIFAR-100 BigGAN($d=256$)		
	IS↑	tFID↓	vFID↓	IS↑	tFID↓	vFID↓
Baseline	8.49	22.24	26.33	7.58	50.79	55.04
w/ 0C	8.93	31.82	35.57	7.89	37.47	42.27
w/ A0C	8.83	26.45	30.30	8.47	36.86	41.80
CHAIN	9.52	**8.27**	**12.06**	10.04	**13.13**	**18.00**
CHAIN_{-0MR}	9.37	9.20	13.05	9.71	24.26	29.20
CHAIN_{-ARMS}	9.33	12.87	16.87	9.09	24.14	29.59
CHAIN_{+0C}	9.43	8.99	12.71	8.84	22.85	27.91
CHAIN_{-LC}	8.68	22.14	26.37	8.05	30.43	35.15
CHAIN_{batch}	9.42	8.51	12.32	9.85	14.49	19.18
$\text{CHAIN}_{Dtm.}$	**9.59**	9.44	13.21	9.76	15.07	19.85
CHAIN_{+0MR_g}	9.37	8.42	12.25	**10.99**	17.09	22.06
$\text{CHAIN}_{+Aff.}$	9.45	8.49	12.24	10.02	14.19	19.07
ADrop	8.72	14.76	18.48	9.04	29.05	34.01
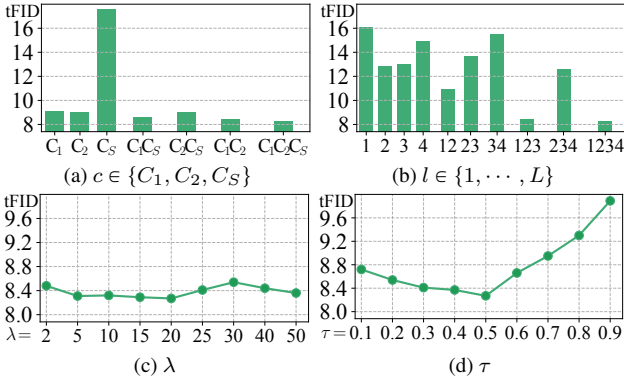
Figure 6. tFID ↓ under different factors. Ablation studies on 10% CIFAR-10 with OmniGAN ($d=256$) w.r.t. different conv. configurations, different blocks, λ and τ for CHAIN.

all blocks yields the best results. Figure 6c shows that varying λ between 2 to 50 does not significantly affect performance, indicating the robustness of CHAIN to λ. Lastly, Figure 6d suggests that setting τ to be 0.5 is preferable.

Comparison with other variants. We compare CHAIN against other normalization techniques such as BN, IN, LN, GN, and BN w/ Lipschitzness constraint (BN_{+LC}), methods preventing discriminator overfitting such as DA, ADA, LeCam, and gradient penalizations for improving generalization. Table 6 details these comparisons. For GN, we optimized group number (n_g) for CIFAR-10 ($n_g=32$) and CIFAR-100 ($n_g=16$). Implementations for AGP_{weight} and AGP_{input} are explained in §D.3. The results in Table 6 show CHAIN outperforms other methods, with AGP_{weight} also

Table 6. Ablation studies. BN_{+LC}: BN w/ Lipschitz constraint. AGP_{weight}: adaptive gradient penalty w.r.t. weights. AGP_{input}: adaptive gradient penalty w.r.t. inputs. LeCam fails to converge on OmniGAN due to its multi-dimensional output design.

Method	10% CIFAR-10 OmniGAN($d=256$)			10% CIFAR-100 BigGAN($d=256$)		
	IS↑	tFID↓	vFID↓	IS↑	tFID↓	vFID↓
Baseline	8.49	22.24	26.33	7.58	50.79	55.04
BN	7.56	37.37	41.52	7.07	55.83	60.46
BN_{+LC}	9.40	14.32	17.75	9.15	25.87	30.83
IN	6.71	53.80	57.76	5.13	83.06	87.40
LN	6.23	101.97	105.58	9.04	26.25	31.22
GN	7.38	49.39	53.46	8.80	31.40	36.53
DA	8.84	12.90	16.67	8.86	27.22	31.80
ADA	**9.67**	13.86	17.70	8.96	20.09	24.90
LeCam	–	–	–	8.30	31.52	36.26
AGP_{input}	8.75	14.78	18.65	8.48	24.95	29.58
AGP_{weight}	9.42	11.86	15.78	9.24	18.52	23.28
CHAIN	9.52	**8.27**	**12.06**	**10.04**	**13.13**	**18.00**

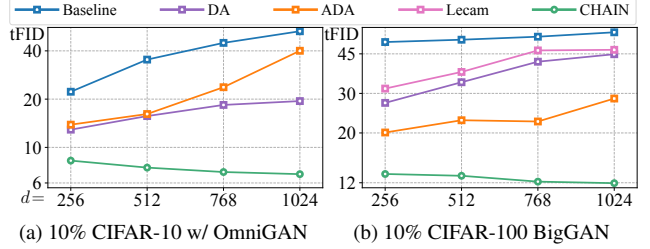(a) 10% CIFAR-10 w/ OmniGAN   (b) 10% CIFAR-100 BigGAN

Figure 7. tFID for different methods with varying feature sizes d.

yielding competitive results, supporting Prop. 3.1 about weight gradient reduction enhancing generalization. Furthermore, Figure 7 indicates that CHAIN benefits from increased network width, unlike other models that deteriorate with wider networks, confirming the superiority of CHAIN.

More analyses. §E compares leading methods, analyzes gradients on CIFAR-100 w/ BigGAN, evaluates eRank against AGP_{weight}, and examines feature norm. CHAIN gains significant improvements with mild extra load (§F).

5. Conclusions

Our method, LipsCHitz contuity constrAIned Normalization (CHAIN), harnesses the generalization benefits of BN to counter discriminator overfitting in GAN training. We refine standard BN by implementing the zero-mean regularization and the Lipschitzness constraint, effectively reducing gradient norms in latent features and discriminator weights. This approach not only stabilizes GAN training but also boosts generalization. Proven in theory and practice, CHAIN excels across diverse backbones and datasets, consistently surpassing existing methods and effectively addressing discriminator overfitting in GANs.

References

[1] Alper Aksac, Douglas J Demetrick, Tansel Ozyer, and Reda Alhajj. Brecahad: a dataset for breast cancer histopathological annotation and diagnosis. *BMC research notes*, 12(1): 1–3, 2019. 5

[2] Pierre Alquier, James Ridgway, and Nicolas Chopin. On the properties of variational approximations of gibbs posteriors. *JMLR*, 17(1):8374–8414, 2016. 13

[3] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *ICML*, pages 214–223. PMLR, 2017. 2, 3, 4

[4] Sanjeev Arora, Rong Ge, Yingyu Liang, Tengyu Ma, and Yi Zhang. Generalization and equilibrium in generative adversarial nets (gans). In *ICML*, pages 224–232, 2017. 3

[5] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer normalization, 2016. cite arxiv:1607.06450. 2

[6] Peter L Bartlett and Shahar Mendelson. Rademacher and gaussian complexities: Risk bounds and structural results. *JMLR*, 3(Nov):463–482, 2002. 2

[7] Nils Bjorck, Carla P Gomes, Bart Selman, and Kilian Q Weinberger. Understanding batch normalization. *NeurIPS*, 31, 2018. 1, 2

[8] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale GAN training for high fidelity natural image synthesis. In *ICLR*, 2019. 1, 2, 5

[9] Tianle Cai, Shengjie Luo, Keyulu Xu, Di He, Tie-yan Liu, and Liwei Wang. Graphnorm: A principled approach to accelerating graph neural network training. In *ICML*, pages 1204–1215. PMLR, 2021. 2, 3

[10] Olivier Catoni. Pac-bayesian supervised classification: the thermodynamics of statistical learning. *arXiv preprint arXiv:0712.0248*, 2007. 2, 3, 13

[11] Brian Chao. Anime face dataset: a collection of high-quality anime faces., 2019. 5

[12] Tianlong Chen, Yu Cheng, Zhe Gan, Jingjing Liu, and Zhangyang Wang. Data-efficient gan training beyond (just) augmentations: A lottery ticket perspective. *NeurIPS*, 34: 20941–20955, 2021. 3, 7, 20

[13] Casey Chu, Kentaro Minami, and Kenji Fukumizu. Smoothness and stability in gans. In *International Conference on Learning Representations*, 2020. 4

[14] Kaiwen Cui, Jiaxing Huang, Zhipeng Luo, Gongjie Zhang, Fangneng Zhan, and Shijian Lu. Genco: generative co-training for generative adversarial networks with limited data. In *AAAI*, pages 499–507, 2022. 2, 5

[15] Kaiwen Cui, Yingchen Yu, Fangneng Zhan, Shengcai Liao, Shijian Lu, and Eric P Xing. Kd-dlgan: Data limited image generation via knowledge distillation. In *CVPR*, pages 3872–3882, 2023. 1, 2, 5, 6

[16] Hadi Daneshmand, Jonas Kohler, Francis Bach, Thomas Hofmann, and Aurelien Lucchi. Batch normalization provably avoids ranks collapse for randomly initialised deep networks. *NeurIPS*, 33:18387–18398, 2020. 20

[17] Hadi Daneshmand, Amir Joudaki, and Francis Bach. Batch normalization orthogonalizes representations in deep random networks. *Advances in Neural Information Processing Systems*, 34:4896–4906, 2021. 20

[18] Etienne Decencière, Xiwei Zhang, Guy Cazuguel, Bruno Lay, Béatrice Cochener, Caroline Trone, Philippe Gain, Richard Ordonez, Pascale Massin, et al. Feedback on a publicly distributed image database: the messidor database. *Image Analysis & Stereology*, 33(3):231–234, 2014. 5

[19] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, pages 248–255. Ieee, 2009. 1, 5

[20] Tiantian Fang, Ruoyu Sun, and Alex Schwing. Diggan: Discriminator gradient gap regularization for gan training with limited data. *NeurIPS*, 35:31782–31795, 2022. 1, 2, 3, 5

[21] Pierre Foret, Ariel Kleiner, Hossein Mobahi, and Behnam Neyshabur. Sharpness-aware minimization for efficiently improving generalization. In *ICLR*, 2021. 3, 14, 15

[22] Shiming Ge, Bochao Liu, Pengju Wang, Yong Li, and Dan Zeng. Learning privacy-preserving student networks via discriminative-generative distillation. *IEEE Transactions on Image Processing*, 32:116–127, 2022. 1, 2

[23] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014. 1, 2

[24] Henry Gouk, Eibe Frank, Bernhard Pfahringer, and Michael J Cree. Regularisation of neural networks by enforcing lipschitz continuity. *Machine Learning*, 110:393–416, 2021. 4

[25] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. Improved training of wasserstein gans. *Advances in neural information processing systems*, 30, 2017. 1, 2, 4

[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. *NeurIPS*, 30, 2017. 5

[27] Liang Hou, Qi Cao, Yige Yuan, Songtao Zhao, Chongyang Ma, Siyuan Pan, Pengfei Wan, Zhongyuan Wang, Huawei Shen, and Xueqi Cheng. Augmentation-aware self-supervision for data-efficient gan training. *NeurIPS*, 36, 2024. 3, 7, 20

[28] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7132–7141, 2018. 19

[29] Jiaxing Huang, Kaiwen Cui, Dayan Guan, Aoran Xiao, Fangneng Zhan, Shijian Lu, Shengcai Liao, and Eric Xing. Masked generative adversarial networks are data-efficient generation learners. *NeurIPS*, 35:2154–2167, 2022. 1, 2, 5, 6

[30] Sergey Ioffe. Batch renormalization: Towards reducing minibatch dependence in batch-normalized models. *NeurIPS*, 30, 2017. 5, 18

[31] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, pages 448–456. pmlr, 2015. 1, 2

[32] Kaiyi Ji, Yi Zhou, and Yingbin Liang. Understanding estimation and generalization error of generative adversarial

networks. *IEEE Trans. IT.*, 67(5):3114–3129, 2021. 2, 3, 14

[33] Liming Jiang, Bo Dai, Wayne Wu, and Chen Change Loy. Deceive d: Adaptive pseudo augmentation for gan training with limited data. *NeurIPS*, 34:21655–21667, 2021. 1, 3, 5

[34] Minguk Kang, Woohyeon Shim, Minsu Cho, and Jaesik Park. Rebooting acgan: Auxiliary classifier gans with stable training. *NeurIPS*, 34:23505–23518, 2021. 6

[35] Minguk Kang, Jun-Yan Zhu, Richard Zhang, Jaesik Park, Eli Shechtman, Sylvain Paris, and Taesung Park. Scaling up gans for text-to-image synthesis. *CVPR*, 2023. 1, 2

[36] Ryo Karakida, Shotaro Akaho, and Shun-ichi Amari. The normalization method for alleviating pathological sharpness in wide neural networks. *NeurIPS*, 32, 2019. 1, 2, 3

[37] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of gans for improved quality, stability, and variation. In *ICLR*, 2018. 1

[38] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *CVPR*, pages 4401–4410, 2019. 1, 2, 7

[39] Tero Karras, Miika Aittala, Janne Hellsten, Samuli Laine, Jaakko Lehtinen, and Timo Aila. Training generative adversarial networks with limited data. *NeurIPS*, 33:12104–12114, 2020. 1, 2, 5

[40] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of stylegan. In *CVPR*, 2020. 1, 2, 5

[41] Tero Karras, Miika Aittala, Samuli Laine, Erik Härkönen, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Alias-free generative adversarial networks. *NeurIPS*, 34:852–863, 2021. 1, 2

[42] Zaid Khan, Vijay Kumar BG, Samuel Schulter, Xiang Yu, Yun Fu, and Manmohan Chandraker. Q: How to specialize large vision-language models to data-scarce vqa tasks? a: Self-train on unlabeled images! In *CVPR*, pages 15005–15015, 2023. 1

[43] Hee E Kim, Alejandro Cosa-Linan, Nandhini Santhanam, Mahboubeh Jannesari, Mate E Maros, and Thomas Ganslandt. Transfer learning for medical image classification: a literature review. *BMC medical imaging*, 22(1):69, 2022. 1

[44] Naveen Kodali, Jacob Abernethy, James Hays, and Zsolt Kira. On convergence and stability of gans. *arXiv preprint arXiv:1705.07215*, 2017. 1, 2, 6

[45] Lingke Kong, Chenyu Lian, Detian Huang, Yanle Hu, Qichao Zhou, et al. Breaking the dilemma of medical image-to-image translation. *NeurIPS*, 34:1964–1978, 2021. 1, 2

[46] Piotr Koniusz, Yusuf Tas, Hongguang Zhang, Mehrtash Tafazzoli Harandi, Fatih Porikli, and Rui Zhang. Museum exhibit identification challenge for the supervised domain adaptation and beyond. In *ECCV*, pages 815–833. Springer, 2018. 1

[47] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009. 5

[48] Nupur Kumari, Richard Zhang, Eli Shechtman, and Jun-Yan Zhu. Ensembling off-the-shelf models for gan training. In *CVPR*, pages 10651–10662, 2022. 1, 2

[49] Karol Kurach, Mario Lučić, Xiaohua Zhai, Marcin Michalski, and Sylvain Gelly. A large-scale study on regularization and normalization in GANs. In *ICML*, pages 3581–3590. PMLR, 2019. 1

[50] Tuomas Kynkäänniemi, Tero Karras, Miika Aittala, Timo Aila, and Jaakko Lehtinen. The role of imagenet classes in fréchet inception distance. In *ICLR*, 2023. 1, 2

[51] Beatrice Laurent and Pascal Massart. Adaptive estimation of a quadratic functional by model selection. *Annals of statistics*, pages 1302–1338, 2000. 15

[52] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436–444, 2015. 1

[53] Kwonjoon Lee, Huiwen Chang, Lu Jiang, Han Zhang, Zhuowen Tu, and Ce Liu. Vitgan: Training gans with vision transformers. In *International Conference on Learning Representations*, 2021. 2

[54] Yanghao Li, Naiyan Wang, Jianping Shi, Jiaying Liu, and Xiaodi Hou. Revisiting batch normalization for practical domain adaptation. *arXiv preprint arXiv:1603.04779*, 2016. 1

[55] Ziqiang Li, Chaoyue Wang, Heliang Zheng, Jing Zhang, and Bin Li. Fakeclr: Exploring contrastive learning for solving latent discontinuity in data-efficient gans. In *ECCV*, pages 598–615. Springer, 2022. 1, 2, 5

[56] Zinan Lin, Vyas Sekar, and Giulia Fanti. Why spectral normalization stabilizes GANs: Analysis and improvements. In *NeurIPS*, 2021. 4

[57] Bingchen Liu, Yizhe Zhu, Kunpeng Song, and Ahmed Elgammal. Towards faster and stabilized gan training for high-fidelity few-shot image synthesis. In *ICLR*, 2020. 2, 5, 6, 19

[58] Hanxiao Liu, Andy Brock, Karen Simonyan, and Quoc Le. Evolving normalization-activation layers. *NeurIPS*, 33: 13539–13550, 2020. 2

[59] Kanglin Liu, Wenming Tang, Fei Zhou, and Guoping Qiu. Spectral regularization for combating mode collapse in gans. In *ICCV*, 2019. 2

[60] Yong Liu, Siqi Mai, Minhao Cheng, Xiangning Chen, Cho-Jui Hsieh, and Yang You. Random sharpness-aware minimization. *NeurIPS*, 35:24543–24556, 2022. 3

[61] Ping Luo, Xinjiang Wang, Wenqi Shao, and Zhanglin Peng. Towards understanding regularization in batch normalization. In *ICLR*, 2018. 3

[62] Kaifeng Lyu, Zhiyuan Li, and Sanjeev Arora. Understanding the generalization benefit of normalization layers: Sharpness reduction. *NeurIPS*, 35:34689–34708, 2022. 1, 2, 3

[63] Lars Mescheder, Andreas Geiger, and Sebastian Nowozin. Which training methods for gans do actually converge? In *ICML*, pages 3481–3490. PMLR, 2018. 2, 6

[64] Takeru Miyato and Masanori Koyama. cGANs with projection discriminator. In *ICLR*, 2018. 2

[65] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. In *ICLR*, 2018. 1, 2, 4

[66] Youssef Mroueh and Tom Sercu. Fisher gan. *Advances in neural information processing systems*, 30, 2017. 1

[67] Alfred Müller. Integral probability metrics and their generating classes of functions. *Advances in applied probability*, 29(2):429–443, 1997. 3

[68] Yao Ni and Piotr Koniusz. NICE: NoIse-modulated Consistency rEgularization for Data-Efficient GANs. *NeurIPS*, 36, 2024. 1, 20

[69] Yao Ni, Dandan Song, Xi Zhang, Hao Wu, and Lejian Liao. Cagan: Consistent adversarial training enhanced gans. In *IJCAI*, pages 2588–2594, 2018. 1

[70] Yao Ni, Piotr Koniusz, Richard Hartley, and Richard Nock. Manifold learning benefits gans. In *CVPR*, pages 11265–11274, 2022. 1, 20

[71] Sebastian Nowozin, Botond Cseke, and Ryota Tomioka. f-gan: Training generative neural samplers using variational divergence minimization. In *NeurIPS*. Curran Associates, Inc., 2016. 2

[72] Utkarsh Ojha, Yijun Li, Jingwan Lu, Alexei A Efros, Yong Jae Lee, Eli Shechtman, and Richard Zhang. Few-shot image generation via cross-domain correspondence. In *CVPR*, pages 10743–10752, 2021. 1, 2

[73] Gaurav Parmar, Krishna Kumar Singh, Richard Zhang, Yijun Li, Jingwan Lu, and Jun-Yan Zhu. Zero-shot image-to-image translation. In *ACM SIGGRAPH*, pages 1–11, 2023. 1

[74] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015. 1

[75] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *ICML*, pages 8748–8763. PMLR, 2021. 6, 7

[76] Elad Richardson, Yuval Alaluf, Or Patashnik, Yotam Nitzan, Yaniv Azar, Stav Shapiro, and Daniel Cohen-Or. Encoding in style: a stylegan encoder for image-to-image translation. In *CVPR*, pages 2287–2296, 2021. 1

[77] Kevin Roth, Aurelien Lucchi, Sebastian Nowozin, and Thomas Hofmann. Stabilizing training of generative adversarial networks through regularization. *Advances in neural information processing systems*, 30, 2017. 2

[78] Olivier Roy and Martin Vetterli. The effective rank: A measure of effective dimensionality. In *15th European signal processing conference*, pages 606–610. IEEE, 2007. 6, 7, 18, 21

[79] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. *NeurIPS*, 29, 2016. 5

[80] Shibani Santurkar, Dimitris Tsipras, Andrew Ilyas, and Aleksander Madry. How does batch normalization help optimization? *NeurIPS*, 31, 2018. 1, 2, 3, 16

[81] Axel Sauer, Katja Schwarz, and Andreas Geiger. Stylegan-xl: Scaling stylegan to large diverse datasets. In *ACM SIGGRAPH*, pages 1–10, 2022. 1

[82] Axel Sauer, Tero Karras, Samuli Laine, Andreas Geiger, and Timo Aila. StyleGAN-T: Unlocking the power of GANs for fast large-scale text-to-image synthesis. In *ICML*, pages 30105–30118, 2023. 1, 2

[83] Seonguk Seo, Yumin Suh, Dongwan Kim, Geeho Kim, Jongwoo Han, and Bohyung Han. Learning to optimize domain specific normalization for domain generalization. In *ECCV*, pages 68–83. Springer, 2020. 1

[84] Matt Shannon, Ben Poole, Soroosh Mariooryad, Tom Bagby, Eric Battenberg, David Kao, Daisy Stanton, and RJ Skerry-Ryan. Non-saturating gan training as divergence minimization. *arXiv preprint arXiv:2010.08029*, 2020. 3

[85] Sheng Shen, Zhewei Yao, Amir Gholami, Michael Mahoney, and Kurt Keutzer. Powernorm: Rethinking batch normalization in transformers. In *ICML*, pages 8741–8751. PMLR, 2020. 2, 3, 5, 18

[86] Xiaoqian Shen, Xiang Li, and Mohamed Elhoseiny. Mostgan-v: Video generation with temporal motion styles. In *CVPR*, pages 5652–5661, 2023. 1, 2

[87] Fatemeh Shiri, Fatih Porikli, Richard Hartley, and Piotr Koniusz. Identity-preserving face recovery from portraits. In *WACV*, pages 102–111. IEEE, 2018. 1

[88] Fatemeh Shiri, Xin Yu, Fatih Porikli, Richard Hartley, and Piotr Koniusz. Recovering faces from portraits with auxiliary facial attributes. In *WACV*, pages 406–415, 2019. 2

[89] Fatemeh Shiri, Xin Yu, Fatih Porikli, Richard Hartley, and Piotr Koniusz. Identity-preserving face recovery from stylized portraits. *IJCV*, 127:863–883, 2019. 2

[90] Zhangzhang Si and Song-Chun Zhu. Learning hybrid image templates (hit) by information projection. *IEEE Trans. PAMI*, 34(7):1354–1367, 2011. 5

[91] Christian Simon, Piotr Koniusz, Richard Nock, and Mehrtash Harandi. On modulating the gradient for meta-learning. In *ECCV*, pages 556–572. Springer, 2020. 3

[92] Ivan Skorokhodov, Aliaksandr Siarohin, Yinghao Xu, Jian Ren, Hsin-Ying Lee, Peter Wonka, and Sergey Tulyakov. 3D generation on ImageNet. In *ICLR*, 2023. 1, 2

[93] Ke Sun, Piotr Koniusz, and Zhen Wang. Fisher-bures adversary graph convolutional networks. In *Uncertainty in Artificial Intelligence*, pages 465–475. PMLR, 2020. 3

[94] Ming Tao, Bing-Kun Bao, Hao Tang, and Changsheng Xu. Galip: Generative adversarial clips for text-to-image synthesis. *CVPR*, 2023. 1, 2

[95] Song Tao and Jia Wang. Alleviation of gradient exploding in gans: Fake can be real. In *CVPR*, pages 1191–1200, 2020. 1

[96] Hoang Thanh-Tung, Truyen Tran, and Svetha Venkatesh. Improving generalization and stability of generative adversarial networks. In *ICLR*, 2019. 2, 6

[97] Hung-Yu Tseng, Lu Jiang, Ce Liu, Ming-Hsuan Yang, and Weilong Yang. Regularizing generative adversarial networks under limited data. In *CVPR*, pages 7921–7931, 2021. 1, 2, 3

[98] Dmitry Ulyanov, Andrea Vedaldi, and Victor S. Lempitsky. Instance normalization: The missing ingredient for fast stylization. *ArXiv*, abs/1607.08022, 2016. 2

[99] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*. Curran Associates, Inc., 2017. 2

[100] Pengfei Wang, Zhaoxiang Zhang, Zhen Lei, and Lei Zhang. Sharpness-aware gradient matching for domain generalization. In *CVPR*, pages 3769–3778, 2023. 3

[101] Ximei Wang, Ying Jin, Mingsheng Long, Jianmin Wang, and Michael I Jordan. Transferable normalization: Towards improving transferability of deep neural networks. *NeurIPS*, 32, 2019. 1

[102] 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*, pages 218–234, 2018. 1, 2

[103] Yuhan Wang, Liming Jiang, and Chen Change Loy. Styleinv: A temporal style modulated inversion network for unconditional video generation. In *ICCV*, 2023. 1, 2

[104] Yuxin Wu and Kaiming He. Group normalization. In *Proceedings of the European conference on computer vision (ECCV)*, pages 3–19, 2018. 2

[105] Sitao Xiang and Hao Li. On the effects of batch and weight normalization in generative adversarial networks. *arXiv preprint arXiv:1704.03971*, 2017. 1

[106] Chugui Xu, Ju Ren, Deyu Zhang, Yaoxue Zhang, Zhan Qin, and Kui Ren. Ganobfuscator: Mitigating information leakage under gan via differential privacy. *IEEE Transactions on Information Forensics and Security*, 14(9):2358–2371, 2019. 1, 2

[107] Junjie Yan, Ruosi Wan, Xiangyu Zhang, Wei Zhang, Yichen Wei, and Jian Sun. Towards stabilizing batch statistics in backward propagation of batch normalization. In *ICLR*, 2020. 2, 5, 18

[108] Ceyuan Yang, Yujun Shen, Yinghao Xu, and Bolei Zhou. Data-efficient instance generation from instance discrimination. *NeurIPS*, 34:9378–9390, 2021. 1, 2

[109] Mengping Yang, Zhe Wang, Ziqiu Chi, and Yanbing Zhang. FreGAN: Exploiting frequency components for training GANs under limited data. In *NeurIPS*, 2022. 6

[110] Lijun Yu, Yong Cheng, Kihyuk Sohn, José Lezama, Han Zhang, Huiwen Chang, Alexander G. Hauptmann, Ming-Hsuan Yang, Yuan Hao, Irfan Essa, and Lu Jiang. Magvit: Masked generative video transformer. In *CVPR*, pages 10459–10469, 2023. 1, 2

[111] Biao Zhang and Rico Sennrich. Root mean square layer normalization. *NeurIPS*, 32, 2019. 2

[112] Bowen Zhang, Shuyang Gu, Bo Zhang, Jianmin Bao, Dong Chen, Fang Wen, Yong Wang, and Baining Guo. Styleswin: Transformer-based gan for high-resolution image generation. In *CVPR*, pages 11304–11314, 2022. 2

[113] Han Zhang, Zizhao Zhang, Augustus Odena, and Honglak Lee. Consistency regularization for generative adversarial networks. In *ICLR*, 2020. 1, 2

[114] Jianfeng Zhang, Zihang Jiang, Dingdong Yang, Hongyi Xu, Yichun Shi, Guoxian Song, Zhongcong Xu, Xinchao Wang, and Jiashi Feng. Avatargen: a 3d generative model for animatable human avatars. In *ECCV*. Springer, 2023. 1, 2

[115] Pengchuan Zhang, Qiang Liu, Dengyong Zhou, Tao Xu, and Xiaodong He. On the discrimination-generalization tradeoff in GANs. In *ICLR*, 2018. 2, 3, 4

[116] Shan Zhang, Yao Ni, Jinhao Du, Yanxia Liu, and Piotr Koniusz. Semantic transfer from head to tail: Enlarging tail margin for long-tailed visual recognition. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 1350–1360, 2024. 2

[117] Xingxuan Zhang, Renzhe Xu, Han Yu, Hao Zou, and Peng Cui. Gradient norm aware minimization seeks first-order flatness and improves generalization. In *CVPR*, pages 20247–20257, 2023. 3

[118] Zhaoyu Zhang, Mengyan Li, and Jun Yu. On the convergence and mode collapse of gan. In *SIGGRAPH Asia 2018 Technical Briefs*, pages 1–4. 2018. 1

[119] Junbo Zhao, Michael Mathieu, and Yann LeCun. Energy-based generative adversarial network. *arXiv preprint arXiv:1609.03126*, 2016. 2

[120] Shengyu Zhao, Zhijian Liu, Ji Lin, Jun-Yan Zhu, and Song Han. Differentiable augmentation for data-efficient gan training. *NeurIPS*, 33:7559–7570, 2020. 1, 2, 5, 19

[121] Yang Zhao, Hao Zhang, and Xiuyuan Hu. Penalizing gradient norm for efficiently improving generalization in deep learning. In *ICML*, pages 26982–26992. PMLR, 2022. 3

[122] Yong Zhong, Hong Tao Liu, Xiaodong Liu, Fan Bao, Weiran Shen, and Chongxuan Li. Deep generative modeling on limited data with regularization by nontransferable pre-trained models. In *ICLR*, 2023. 1, 2

[123] Peng Zhou, Lingxi Xie, Bingbing Ni, Cong Geng, and Qi Tian. Omni-gan: On the secrets of cgans and beyond. In *ICCV*, pages 14061–14071, 2021. 2, 5

[124] Peng Zhou, Lingxi Xie, Bingbing Ni, and Qi Tian. Cips-3d++: End-to-end real-time high-resolution 3d-aware gans for gan inversion and stylization. *IEEE Trans. PAMI*, 45 (10):11502–11520, 2023. 1, 2

[125] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *ICCV*, pages 2223–2232, 2017. 2

◯◯CHAIN: Enhancing Generalization in Data-Efficient GANs via lips*CH*itz continuity constr*AI*ned Normalization (*Supplementary Material*)

Yao Ni[†]   Piotr Koniusz[*,§,†]
[†]The Australian University   [§]Data61♥CSIRO
[†]firstname.lastname@anu.edu.au

The supplementary material contains notations (§A), theoretical proofs (§B), an explanation for stochastic M design (§C), implementation guidelines (§D), extra experimental results (§E), training overhead (§F), and examples of generated images (§G).

A. Notations

Below, we explain the notations used in this work.

Scalars: Represented by lowercase letters (*e.g.*, m, n, p).

Vectors: Bold lowercase letters (*e.g.*, $\boldsymbol{x}$, $\boldsymbol{z}$, $\boldsymbol{\mu}$).

Matrices: Bold uppercase letters (*e.g.*, $\boldsymbol{W}$, $\boldsymbol{M}$, $\boldsymbol{H}$).

Functions: Letters followed by brackets (*e.g.*, $\phi(\cdot)$, $h(\cdot)$, $\mathrm{diag}(\cdot)$).

Function sets: Calligraphic uppercase letters are used (*e.g.*, $\mathcal{H}$, $\mathcal{G}$, $\mathcal{F}$). But note $\mathcal{B}$ specifically denotes the Bernoulli distribution.

Probability measures: Denoted by letters μ, ν, π, $\hat{\rho}$ and p_z.

Expectation: $\mathbb{E}[\cdot]$ represents the average or expected value of a random variable.

B. Proofs

We start with a lemma on the Pac-Bayesian bound, followed by in-depth proofs for the theories outlined in the main paper.

Lemma B.1 *(A variant of the PAC-Bayesian bound adapted from Theorem 4.1 in [2] and from [10].) Let $\mathcal{D}$ be a distribution over $\mathcal{X}$. Denote the prior and posterior probability measure on a hypothesis set $\mathcal{F}$ as $\pi(\cdot), \hat{\rho}(\cdot) \in \mathcal{M}_+^1$, where $\mathcal{M}_+^1$ (positive and normalized to 1) is the set of all probability measures on $\mathcal{F}$. Denote $\phi : \mathcal{F} \times \mathcal{X} \to \mathbb{R}$ and $\mathcal{L}_{\mathcal{D}}^\phi := \mathbb{E}_{\mathcal{D}}[\phi]$ as the loss. For $\alpha > 0$ and $\delta \in (0,1]$, with probability at least $1 - \delta$ over the choice of $\boldsymbol{x} \sim \mathcal{D}^n$ (a subset from $\mathcal{D}$ with size of n), we have:*

$$\mathbb{E}_{f \sim \hat{\rho}} \mathcal{L}_{\mathcal{D}}^\phi(f) \leqslant \mathbb{E}_{f \sim \hat{\rho}} \widehat{\mathcal{L}}_{\mathcal{D}^n}^\phi(f) + \frac{1}{\alpha}\Big[KL(\hat{\rho}\|\pi) + \ln\frac{1}{\delta} + \Omega(\alpha, n) \Big]$$

$$where \qquad \Omega(\alpha, n) = \ln \mathbb{E}_{f \sim \pi} \mathbb{E}_{\mathcal{D}^n} \exp\big\{ \alpha\big(\mathcal{L}_{\mathcal{D}}^\phi(f) - \widehat{\mathcal{L}}_{\mathcal{D}^n}^\phi(f)\big)\big\}. \tag{17}$$

Proof: The Donsker-Varadhan change of measure states that for any measurable function $\varphi : \mathcal{F} \to \mathbb{R}$ and $\forall \hat{\rho}$ on $\mathcal{F}$, we have:

$$\mathbb{E}_{f \sim \hat{\rho}} \varphi(f) \leqslant KL(\hat{\rho}\|\pi) + \ln \mathbb{E}_{f \sim \pi} e^{\varphi(f)}.$$

Denoting $\varphi(f) := \alpha\big(\mathcal{L}_{\mathcal{D}}^\phi(f) - \widehat{\mathcal{L}}_{\mathcal{D}^n}^\phi(f)\big)$, the above inequality yields:

$$\alpha\big(\mathbb{E}_{f \sim \hat{\rho}} \mathcal{L}_{\mathcal{D}}^\phi(f) - \mathbb{E}_{f \sim \hat{\rho}} \widehat{\mathcal{L}}_{\mathcal{D}^n}^\phi(f)\big) = \mathbb{E}_{f \sim \hat{\rho}} \alpha\big(\mathcal{L}_{\mathcal{D}}^\phi(f) - \widehat{\mathcal{L}}_{\mathcal{D}^n}^\phi(f)\big)$$

$$\leqslant KL(\hat{\rho}\|\pi) + \ln \mathbb{E}_{f \sim \pi} e^{\alpha(\mathcal{L}_{\mathcal{D}}^\phi(f) - \widehat{\mathcal{L}}_{\mathcal{D}^n}^\phi(f))}.$$

Applying Markov's inequality to the random variable $\xi_\pi(X) := \mathbb{E}_{f \sim \pi} e^{\alpha(\mathcal{L}_{\mathcal{D}}^\phi(f) - \widehat{\mathcal{L}}_{\mathcal{D}^n}^\phi(f))}$, we obtain:

$$\Pr\big(\xi_\pi \leqslant \frac{1}{\delta}\mathbb{E}[\xi_\pi]\big) \geqslant 1 - \delta.$$

Thus, with probability at least $1 - \delta$ over the choice of $\boldsymbol{x} \sim \mathcal{D}^n$, we obtain:

$$\mathbb{E}_{f \sim \hat{\rho}} \mathcal{L}_{\mathcal{D}}^\phi(f) \leqslant \mathbb{E}_{f \sim \hat{\rho}} \widehat{\mathcal{L}}_{\mathcal{D}^n}^\phi(f) + \frac{1}{\alpha}\Big[KL(\hat{\rho}\|\pi) + \ln\frac{1}{\delta} + \ln \mathbb{E}_{f \sim \pi} \mathbb{E}_{\mathcal{D}^n} e^{\alpha(\mathcal{L}_{\mathcal{D}}^\phi(f) - \widehat{\mathcal{L}}_{\mathcal{D}^n}^\phi(f))} \Big].$$

B.1. Proof of Lemma 3.1

Lemma 3.1 *(Partial results of Theorem 1 in [32].) Assume the discriminator set $\mathcal{H}$ is even,* i.e., $h \in \mathcal{H}$ implies $-h \in \mathcal{H}$ and $\|h\|_\infty \leqslant \Delta$. *Let $\hat{\mu}_n$ and $\hat{\nu}_n$ be empirical measures of μ and ν_n with size n. Denote $\nu_n^* = \inf_{\nu \in \mathcal{G}} d_{\mathcal{H}}(\hat{\mu}_n, \nu)$. The generalization error of GAN, defined as $\epsilon_{gan} := d_{\mathcal{H}}(\mu, \nu_n) - \inf_{\nu \in \mathcal{G}} d_{\mathcal{H}}(\mu, \nu)$, is bounded as:*

$$\epsilon_{gan} \leqslant 2\Big(\sup_{h \in \mathcal{H}} \big|\mathbb{E}_\mu[h] - \mathbb{E}_{\hat{\mu}_n}[h]\big| + \sup_{h \in \mathcal{H}} \big|\mathbb{E}_{\nu_n^*}[h] - \mathbb{E}_{\hat{\nu}_n}[h]\big|\Big) = 2d_{\mathcal{H}}(\mu, \hat{\mu}_n) + 2d_{\mathcal{H}}(\nu_n^*, \hat{\nu}_n).$$

Proof:

$$\epsilon_{\text{gan}} := d_{\mathcal{H}}(\mu, \nu_n) - \inf_{\nu \in \mathcal{G}} d_{\mathcal{H}}(\mu, \nu) = d_{\mathcal{H}}(\mu, \nu_n) - d_{\mathcal{H}}(\hat{\mu}_n, \nu_n) + d_{\mathcal{H}}(\hat{\mu}_n, \nu_n) - \inf_{\nu \in \mathcal{G}} d_{\mathcal{H}}(\mu, \nu)$$

$$= \underbrace{d_{\mathcal{H}}(\mu, \nu_n) - d_{\mathcal{H}}(\hat{\mu}_n, \nu_n)}_{①} + \underbrace{\inf_{v \in \mathcal{G}} d_{\mathcal{H}}(\hat{\mu}_n, \nu) - \inf_{v \in \mathcal{G}} d_{\mathcal{H}}(\mu, \nu)}_{②} + \underbrace{d_{\mathcal{H}}(\hat{\mu}_n, \nu_n) - \inf_{v \in \mathcal{G}} d_{\mathcal{H}}(\hat{\mu}_n, \nu)}_{③}.$$

The three components in the above equation are upper-bounded as follows:

Upper bound ① :

$$d_{\mathcal{H}}(\mu, \nu_n) - d_{\mathcal{H}}(\hat{\mu}_n, \nu_n) = \sup_{h \in \mathcal{H}} |\mathbb{E}_\mu[h] - \mathbb{E}_{\nu_n}[h]| - \sup_{h \in \mathcal{H}} |\mathbb{E}_{\hat{\mu}_n}[h] - \mathbb{E}_{\nu_n}[h]|$$

$$\leqslant \sup_{h \in \mathcal{H}} \big|\mathbb{E}_\mu[h] - \mathbb{E}_{\nu_n}[h] - \mathbb{E}_{\hat{\mu}_n}[h] + \mathbb{E}_{\nu_n}[h]\big| = \sup_{h \in \mathcal{H}} \big|\mathbb{E}_\mu[h] - \mathbb{E}_{\hat{\mu}_n}[h]\big|.$$

Upper bound ② : Denote $\nu^* = \inf_{\nu \in \mathcal{G}} d_{\mathcal{H}}(\mu, \nu)$. Then similar to derivation for ① , we obtain:

$$\inf_{v \in \mathcal{G}} d_{\mathcal{H}}(\hat{\mu}_n, \nu) - \inf_{v \in \mathcal{G}} d_{\mathcal{H}}(\mu, \nu) = \inf_{v \in \mathcal{G}} d_{\mathcal{H}}(\hat{\mu}_n, \nu) - d_{\mathcal{H}}(\mu, \nu^*)$$

$$\leqslant d_{\mathcal{H}}(\hat{\mu}_n, \nu) - d_{\mathcal{H}}(\mu, \nu^*) \leqslant \sup_{h \in \mathcal{H}} \big|\mathbb{E}_\mu[h] - \mathbb{E}_{\hat{\mu}_n}[h]\big|.$$

Upper bound ③ : Here, we consider a practical scenario where the discriminator only has access to finite fake data during optimization. Recall that we denote $\nu_n^* := \inf_{\nu \in \mathcal{G}} d_{\mathcal{H}}(\hat{\mu}_n, \nu)$, thus $d_{\mathcal{H}}(\hat{\mu}_n, \nu_n) \geqslant d_{\mathcal{H}}(\hat{\mu}_n, \nu_n^*)$, leading to the inequality that:

$$d_{\mathcal{H}}(\hat{\mu}_n, \nu_n) - \inf_{v \in \mathcal{G}} d_{\mathcal{H}}(\hat{\mu}_n, \nu) = d_{\mathcal{H}}(\hat{\mu}_n, \nu_n) - d_{\mathcal{H}}(\hat{\mu}_n, \nu_n^*)$$

$$= \big(d_{\mathcal{H}}(\hat{\mu}_n, \nu_n) - d_{\mathcal{H}}(\hat{\mu}_n, \hat{\nu}_n)\big) + \big(d_{\mathcal{H}}(\hat{\mu}_n, \hat{\nu}_n) - d_{\mathcal{H}}(\hat{\mu}_n, \nu_n^*)\big)$$

$$\leqslant \sup_{h \in \mathcal{H}} \big|\mathbb{E}_{\nu_n}[h] - \mathbb{E}_{\hat{\nu}_n}[h]\big| + \sup_{h \in \mathcal{H}} \big|\mathbb{E}_{\nu_n^*}[h] - \mathbb{E}_{\hat{\nu}_n}[h]\big| \leqslant 2\sup_{h \in \mathcal{H}} \big|\mathbb{E}_{\nu_n^*}[h] - \mathbb{E}_{\hat{\nu}_n}[h]\big|.$$

Integrating the three bounds we achieve the final result.

B.2. Proof of Proposition 3.1

Proposition 3.1 *Utilizing notations from Lemma 3.1, we define ϵ_{gan}^{nn} as the generalization error of GAN parameterized as neural network classes. Let $\nabla_{\boldsymbol{\theta}_d}$ and $\boldsymbol{H}_{\boldsymbol{\theta}_d}$ represent the gradient and Hessian matrix of discriminator h evaluated at $\boldsymbol{\theta}_d$ over real training data $\hat{\mu}_n$, and $\widetilde{\nabla}_{\boldsymbol{\theta}_d}$ and $\widetilde{\boldsymbol{H}}_{\boldsymbol{\theta}_d}$ over observed fake data $\hat{\nu}_n$. Denoting $\lambda_{max}^{\boldsymbol{H}}$ and $\lambda_{max}^{\widetilde{\boldsymbol{H}}}$ as the largest eigenvalues of $\boldsymbol{H}_{\boldsymbol{\theta}_d}$ and $\widetilde{\boldsymbol{H}}_{\boldsymbol{\theta}_d}$, respectively, and for any $\omega > 0$, the generalization error is bounded as:*

$$\epsilon_{gan}^{nn} \leqslant 2\omega\big(\|\nabla_{\boldsymbol{\theta}_d}\|_2 + \|\widetilde{\nabla}_{\boldsymbol{\theta}_d}\|_2\big) + 4R\Big(\frac{\|\boldsymbol{\theta}_d\|_2^2}{\omega^2}, \frac{1}{n}\Big) + \omega^2\big(|\lambda_{max}^{\boldsymbol{H}}| + |\lambda_{max}^{\widetilde{\boldsymbol{H}}}|\big),$$

where $R\Big(\frac{\|\boldsymbol{\theta}_d\|_2^2}{\omega^2}, \frac{1}{n}\Big)$, a term related to discriminator weights norm, is inversely related to the data size n.

Proof: We start by deriving a PAC-Bayesian bound for GAN generalization error on real data. This is followed by an approach similar to Theorem 1 in [21], establishing a connection between this error and the discriminator's gradient direction. Finally, a Taylor expansion of the discriminator in the gradient direction is applied, paralleled by a similar formulation for fake data, culminating in our final results.

PAC-Bayesian bound for GAN. Denoting $\mathcal{L}_\mu := \mathbb{E}_\mu[h]$ and the parameter of the discriminator as $\boldsymbol{\theta}_d \in \boldsymbol{\Theta}_d$, and applying Lemma B.1, we obtain:

$$\mathbb{E}_{\boldsymbol{\theta}_d \sim \hat{\rho}} \mathcal{L}_\mu(\boldsymbol{\theta}_d) \leqslant \mathbb{E}_{\boldsymbol{\theta}_d \sim \hat{\rho}} \widehat{\mathcal{L}}_{\hat{\mu}_n}(\boldsymbol{\theta}_d) + \frac{1}{\alpha}\Big[\text{KL}(\hat{\rho}\|\pi) + \ln\frac{1}{\delta} + \Omega(\alpha, n)\Big]$$

$$\text{where} \qquad \Omega(\alpha, n) = \ln \mathbb{E}_{\boldsymbol{\theta}_d \sim \pi} \mathbb{E}_{\hat{\mu}_n} \exp\big\{\alpha\big(\mathcal{L}_\mu(\boldsymbol{\theta}_d) - \widehat{\mathcal{L}}_{\hat{\mu}_n}(\boldsymbol{\theta}_d)\big)\big\}. \tag{18}$$

We then derive the upper bound for $\Omega(\alpha, n)$ on the discriminator. Let ℓ_i represent a realization of the random variable $\mathcal{L}_\mu - h(\boldsymbol{x}_i; \boldsymbol{\theta}_d)$. Given that $h \in [-\Delta, \Delta]$ stated in Lemma 3.1, changing variable $\boldsymbol{x}_i$ to another independent copy $\boldsymbol{x}_i'$, alters

ℓ_i by at most $\frac{2\Delta}{n}$. Utilizing Hoeffding's lemma, we obtain:

$$\mathbb{E}_{\hat{\mu}_n} e^{\alpha(\mathcal{L}_\mu(\boldsymbol{\theta}_d) - \widehat{\mathcal{L}}_{\hat{\mu}_n}(\boldsymbol{\theta}_d))} = \mathbb{E}_{\hat{\mu}_n} \exp\left\{\frac{\alpha}{n}\sum_{i=1}^n \ell_i\right\} = \prod_{i=1}^n \mathbb{E}\exp\left\{\frac{\alpha}{n}\ell_i\right\}$$

$$\leqslant \prod_{i=1}^n \exp\left\{\frac{\alpha^2(2\Delta)^2}{8n^2}\right\} = \exp\left\{\frac{\alpha^2\Delta^2}{2n}\right\}. \tag{19}$$

By inserting Eq. 19 into Eq. 18, and setting $\alpha = n$, we arrive at:

$$\mathbb{E}_{\boldsymbol{\theta}_d \sim \hat{\rho}}\mathcal{L}_\mu(\boldsymbol{\theta}_d) \leqslant \mathbb{E}_{\boldsymbol{\theta}_d \sim \hat{\rho}}\widehat{\mathcal{L}}_{\hat{\mu}_n}(\boldsymbol{\theta}_d) + \frac{1}{n}\Big[\mathrm{KL}(\hat{\rho}\|\pi) + \ln\frac{1}{\delta}\Big] + \frac{\Delta^2}{2}. \tag{20}$$

Generalization error and the gradient direction of the weight. Continuing, we adopt an analysis parallel to the proof of Theorem 1 in [21]. According to Eq. 12 in their work, if π is a measure on $\mathcal{N}(\boldsymbol{0}, \sigma_\pi^2\boldsymbol{I})$ and $\hat{\rho}$ is a measure on $\mathcal{N}(\boldsymbol{\theta}_d, \sigma^2\boldsymbol{I})$ with the dimension of $\boldsymbol{\theta}$ being k, it follows that:

$$\mathrm{KL}(\hat{\rho}\|\pi) = \frac{1}{2}\Big[1 + k\ln\big(1 + \frac{\|\boldsymbol{\theta}_d\|_2^2}{k\sigma^2}\big)\Big].$$

Subsequently, Eq. 20 transforms into:

$$\mathbb{E}_{\boldsymbol{\varepsilon} \sim \mathcal{N}(\boldsymbol{0}, \sigma^2\boldsymbol{I})}\mathcal{L}_\mu(\boldsymbol{\theta}_d + \boldsymbol{\varepsilon}) \leqslant \mathbb{E}_{\boldsymbol{\varepsilon} \sim \mathcal{N}(\boldsymbol{0}, \sigma^2\boldsymbol{I})}\widehat{L}_{\hat{\mu}_n}(\boldsymbol{\theta}_d + \boldsymbol{\varepsilon}) + \frac{1}{n}\Big[\frac{1}{2} + \frac{k}{2}\ln\big(1 + \frac{\|\boldsymbol{\theta}_d\|_2^2}{k\sigma^2}\big) + \ln\frac{1}{\delta}\Big] + \frac{\Delta^2}{2}. \tag{21}$$

By Lemma 1 of [51], for any positive t, we have:

$$\Pr(\|\boldsymbol{\varepsilon}\|_2^2 - k\sigma^2 \geqslant 2\sigma^2\sqrt{kt} + 2t\sigma^2) \leqslant e^{-t}.$$

Thus, with probability $1 - 1/n$ (where $t = \ln n$), it follows that:

$$\|\boldsymbol{\varepsilon}\|_2^2 \leqslant \sigma^2\big(2\ln n + k + 2\sqrt{k\ln n}\big) \leqslant \sigma^2 k\left(1 + \sqrt{\frac{2\ln n}{k}}\right)^2 \leqslant \omega^2.$$

Assuming, as in [21], that perturbations in discriminator weights have negligible impact on performance over an infinite dataset, and integrating σ back into Eq. 21, we deduce:

$$\mathcal{L}_\mu(\boldsymbol{\theta}_d) \leqslant \mathbb{E}_{\boldsymbol{\varepsilon} \sim \mathcal{N}(\boldsymbol{0}, \sigma^2\boldsymbol{I})}\widehat{\mathcal{L}}_{\hat{\mu}_n}(\boldsymbol{\theta}_d + \boldsymbol{\varepsilon}) + \frac{1}{n}\Big[\frac{1}{2} + \frac{k}{2}\ln\big(1 + \frac{\|\boldsymbol{\theta}_d\|_2^2}{k\sigma^2}\big) + \ln\frac{1}{\delta}\Big] + \frac{\Delta^2}{2}$$

$$\leqslant \max_{\|\boldsymbol{\varepsilon}\|_2^2 \leqslant \omega^2}\widehat{\mathcal{L}}_{\hat{\mu}_n}(\boldsymbol{\theta}_d + \boldsymbol{\varepsilon}) + \frac{1}{n}\Big[\frac{1}{2} + \frac{k}{2}\ln\big(1 + \frac{\|\boldsymbol{\theta}_d\|_2^2}{\omega^2}\big(1 + \sqrt{(2\ln n)/k}\big)^2\big) + \ln\frac{1}{\delta}\Big] + \frac{\Delta^2}{2}.$$

Taylor expansion in the weight gradient direction. Observe that the maximum of $\widehat{\mathcal{L}}_{\hat{\mu}_n}$ occurs when $\boldsymbol{\varepsilon}$ is chosen as $\boldsymbol{\varepsilon} = \frac{\omega\boldsymbol{\nabla}_{\hat{\mu}_n, \boldsymbol{\theta}_d}}{\|\boldsymbol{\nabla}_{\hat{\mu}_n, \boldsymbol{\theta}_d}\|_2}$, which is aligned with the gradient of $\widehat{\mathcal{L}}_{\hat{\mu}_n}$ at $\boldsymbol{\theta}_d$ over $\hat{\mu}_n$. We perform a second-order Taylor expansion of $\widehat{\mathcal{L}}_{\hat{\mu}_n}$ around $\boldsymbol{\theta}_d$. Incorporating the remainder and the higher-order terms from the Taylor expansion into $R\Big(\frac{\|\boldsymbol{\theta}_d\|_2^2}{\omega^2}, \frac{1}{n}\Big)$, we derive:

$$L_\mu(\boldsymbol{\theta}_d) \leqslant \widehat{L}_{\hat{\mu}_n}\Big(\boldsymbol{\theta}_d + \frac{\omega\boldsymbol{\nabla}_{\hat{\mu}_n, \boldsymbol{\theta}_d}}{\|\boldsymbol{\nabla}_{\hat{\mu}_n, \boldsymbol{\theta}_d}\|_2}\Big) + R\Big(\frac{\|\boldsymbol{\theta}_d\|_2^2}{\omega^2}, \frac{1}{n}\Big)$$

$$\approx \widehat{L}_{\hat{\mu}_n}(\boldsymbol{\theta}_d) + \omega\|\boldsymbol{\nabla}_{\hat{\mu}_n, \boldsymbol{\theta}_d}\|_2 + \frac{\omega^2}{2\|\boldsymbol{\nabla}_{\hat{\mu}_n, \boldsymbol{\theta}_d}\|_2^2}\boldsymbol{\nabla}_{\hat{\mu}_n, \boldsymbol{\theta}_d}^T\boldsymbol{H}_{\hat{\mu}_n, \boldsymbol{\theta}_d}\boldsymbol{\nabla}_{\hat{\mu}_n, \boldsymbol{\theta}_d} + R\Big(\frac{\|\boldsymbol{\theta}_d\|_2^2}{\omega^2}, \frac{1}{n}\Big).$$

Simplifying notations, we use $\boldsymbol{\nabla}_{\boldsymbol{\theta}_d}$ and $\boldsymbol{H}_{\boldsymbol{\theta}_d}$ for the gradient and Hessian matrix evaluated at $\boldsymbol{\theta}_d$ over real seen data $\hat{\mu}_n$, and similar $\widetilde{\boldsymbol{\nabla}}_{\boldsymbol{\theta}_d}$ and $\widetilde{\boldsymbol{H}}_{\boldsymbol{\theta}_d}$ for observed fake data $\hat{\nu}_n$. Considering the largest eigenvalue of $\boldsymbol{H}_{\boldsymbol{\theta}_d}$ as $\lambda_{\max}^H$, implying $\boldsymbol{v}^T\boldsymbol{H}_{\boldsymbol{\theta}_d}\boldsymbol{v} \leqslant \lambda_{\max}^H\|\boldsymbol{v}\|_2^2$, we bound the real data part of the generalization error of a GAN (Lemma 3.1) parameterized as network as follows:

$$\sup_{h \in \mathcal{H}_{\mathrm{nn}}} \big|\mathbb{E}_\mu[h] - \mathbb{E}_{\hat{\mu}_n}[h]\big| \leqslant \omega\|\boldsymbol{\nabla}_{\boldsymbol{\theta}_d}\|_2 + \frac{\omega^2}{2\|\boldsymbol{\nabla}_{\boldsymbol{\theta}_d}\|_2^2}\big|\boldsymbol{\nabla}_{\boldsymbol{\theta}_d}^T\boldsymbol{H}_{\boldsymbol{\theta}_d}\boldsymbol{\nabla}_{\boldsymbol{\theta}_d}\big| + R\big(\frac{\|\boldsymbol{\theta}_d\|_2^2}{\omega^2}, \frac{1}{n}\big)$$

$$\leqslant \omega\|\boldsymbol{\nabla}_{\boldsymbol{\theta}_d}\|_2 + \frac{\omega^2}{2}|\lambda_{\max}^H| + R\big(\frac{\|\boldsymbol{\theta}_d\|_2^2}{\omega^2}, \frac{1}{n}\big).$$

Similarly, the fake data part in the generalization error of GAN is:

$$\sup_{h \in \mathcal{H}_{\mathrm{nn}}} \big|\mathbb{E}_{\nu_n^*}[h] - \mathbb{E}_{\hat{\nu}_n}[h]\big| \leqslant \omega\|\widetilde{\boldsymbol{\nabla}}_{\boldsymbol{\theta}_d}\|_2 + \frac{\omega^2}{2}|\lambda_{\max}^{\widetilde{H}}| + R\big(\frac{\|\boldsymbol{\theta}_d\|_2^2}{\omega^2}, \frac{1}{n}\big).$$

By integrating the aforementioned two inequalities into the generalization error as detailed in Lemma 3.1, we arrive at:

$$\epsilon_{\mathrm{gan}}^{\mathrm{nn}} \leqslant 2\omega\big(\|\boldsymbol{\nabla}_{\boldsymbol{\theta}_d}\|_2 + \|\widetilde{\boldsymbol{\nabla}}_{\boldsymbol{\theta}_d}\|_2\big) + \omega^2\big(|\lambda_{\max}^H| + |\lambda_{\max}^{\widetilde{H}}|\big) + 4R\big(\frac{\|\boldsymbol{\theta}_d\|_2^2}{\omega^2}, \frac{1}{n}\big).$$

B.3. Proof of Theorem 3.1

Theorem 3.1 (*The issue of the centering step.*) *Consider $\boldsymbol{y}_1, \boldsymbol{y}_2$ as i.i.d. samples from a symmetric distribution centered at $\boldsymbol{\mu}$, where the presence of $\boldsymbol{y}$ implies $2\boldsymbol{\mu} - \boldsymbol{y}$ is also included. After the centering step, $\overset{c}{\boldsymbol{y}}_1, \overset{c}{\boldsymbol{y}}_2$ are i.i.d. samples from the centered distribution. The expected cosine similarity between these samples is given by:*

$$\mathbb{E}_{\boldsymbol{y}_1, \boldsymbol{y}_2}\big[\cos(\boldsymbol{y}_1, \boldsymbol{y}_2)\big] \geqslant \mathbb{E}_{\overset{c}{\boldsymbol{y}}_1, \overset{c}{\boldsymbol{y}}_2}\big[\cos(\overset{c}{\boldsymbol{y}}_1, \overset{c}{\boldsymbol{y}}_2)\big] = 0.$$

Proof: Given that the distribution is symmetric and even, and $\boldsymbol{\mu}_Y \neq \boldsymbol{0}$, the mean of the L_2 normalized distribution $\mathbb{E}\big[\frac{\boldsymbol{y}}{\|\boldsymbol{y}\|_2}\big] \neq \boldsymbol{0}$. Denoting the mean of the L_2 normalized sample as $\boldsymbol{\mu}_Z \neq \boldsymbol{0}$, we can derive the expectation of the cosine similarity as follows:

$$\mathbb{E}_{\boldsymbol{y}_1, \boldsymbol{y}_2}\big[\cos(\boldsymbol{y}_1, \boldsymbol{y}_2)\big] = \mathbb{E}_{\boldsymbol{y}_1, \boldsymbol{y}_2}\Big[\big\langle \frac{\boldsymbol{y}_1}{\|\boldsymbol{y}_1\|_2}, \frac{\boldsymbol{y}_2}{\|\boldsymbol{y}_2\|_2}\big\rangle\Big] = \mathbb{E}_{\boldsymbol{z}_1, \boldsymbol{z}_2}[\langle \boldsymbol{z}_1, \boldsymbol{z}_2\rangle] = \langle \boldsymbol{\mu}_Z, \boldsymbol{\mu}_Z\rangle = \|\boldsymbol{\mu}_Z\|_2^2 \geqslant 0.$$

In the centered distribution with $\overset{c}{\boldsymbol{\mu}}_Y = \boldsymbol{0}$ and the symmetric probability, the presence of $\overset{c}{\boldsymbol{y}}_2$ implies the inclusion of $-\overset{c}{\boldsymbol{y}}_2$. This leads us to the following derivation:

$$\mathbb{E}_{\overset{c}{\boldsymbol{y}}_1, \overset{c}{\boldsymbol{y}}_2}\big[\cos(\overset{c}{\boldsymbol{y}}_1, \overset{c}{\boldsymbol{y}}_2)\big] = \mathbb{E}_{\overset{c}{\boldsymbol{y}}_1, \overset{c}{\boldsymbol{y}}_2}\Big[\big\langle \frac{\overset{c}{\boldsymbol{y}}_1}{\|\overset{c}{\boldsymbol{y}}_1\|_2}, \frac{\overset{c}{\boldsymbol{y}}_2}{\|\overset{c}{\boldsymbol{y}}_2\|_2}\big\rangle\Big] = \frac{1}{2}\mathbb{E}_{\overset{c}{\boldsymbol{y}}_1, \overset{c}{\boldsymbol{y}}_2}\Big[\big\langle \frac{\overset{c}{\boldsymbol{y}}_1}{\|\overset{c}{\boldsymbol{y}}_1\|_2}, \frac{\overset{c}{\boldsymbol{y}}_2}{\|\overset{c}{\boldsymbol{y}}_2\|_2}\big\rangle + \big\langle \frac{\overset{c}{\boldsymbol{y}}_1}{\|\overset{c}{\boldsymbol{y}}_1\|_2}, \frac{-\overset{c}{\boldsymbol{y}}_2}{\|\overset{c}{\boldsymbol{y}}_2\|_2}\big\rangle\Big] = 0$$

Comparing the above two Equations we obtain the final inequality.

B.4. Proof of Theorem 3.2

Theorem 3.2 (*The issue of the scaling step.*) *The scaling step, defined in Eq. 7, can be expressed as matrix multiplication $\overset{s}{\boldsymbol{Y}} = \overset{c}{\boldsymbol{Y}} diag(1/\boldsymbol{\sigma})$. The Lipschitz constant w.r.t. the 2-norm of the scaling step is:*

$$\Big\|diag\Big(\frac{1}{\boldsymbol{\sigma}}\Big)\Big\|_{lc} = \frac{1}{\sigma_{min}},$$

where $\sigma_{min} = \min_c \sigma_c$ represents the minimum value in $\boldsymbol{\sigma}$.

Proof: Consider $\boldsymbol{\Lambda} = diag(\lambda_1, \cdots, \lambda_d)$, a diagonal matrix. We establish that:

$$\|\boldsymbol{\Lambda}\|_{lc} = \max_{\|\boldsymbol{x}\|_2 = 1}\|\boldsymbol{\Lambda}\boldsymbol{x}\|_2 = \max_{\|\boldsymbol{x}\|_2 = 1}\Big(\sum_{i=1}^d \lambda_i x_i^2\Big)^{1/2} \leqslant \max_{\|\boldsymbol{x}\|_2 = 1}\max_i |\lambda_i|\Big(\sum_{i=1}^d x_i^2\Big)^{1/2} = \max_i |\lambda_i| \cdot \max_{\|\boldsymbol{x}\|_2 = 1}\|\boldsymbol{x}\|_2 = \max_i |\lambda_i|.$$

From this, it follows that:

$$\Big\|diag(\frac{1}{\boldsymbol{\sigma}})\Big\|_{lc} = \max_c \Big|\frac{1}{\sigma_c}\Big| = \frac{1}{\min_c \sigma_c} = \frac{1}{\sigma_{min}}.$$

B.5. Proof of Theorem 3.3

Theorem 3.3 (*CHAIN reduces the gradient norm of weights/latent features.*) *Denote the loss of discriminator with CHAIN as $\mathcal{L}$, and the resulting batch features as $\overset{\smile}{\boldsymbol{Y}}$. Let $\overset{\smile}{\boldsymbol{y}}_c \in \mathbb{R}^B$ be c-th column of $\overset{\smile}{\boldsymbol{Y}}$, $\Delta\boldsymbol{y}_c, \Delta\dot{\boldsymbol{y}}_c \in \mathbb{R}^B$ be the c-th column of gradient $\frac{\partial \mathcal{L}}{\partial \overset{\smile}{\boldsymbol{Y}}}, \frac{\partial \mathcal{L}}{\partial \dot{\boldsymbol{Y}}}$. Denote $\Delta\boldsymbol{w}_c$ as the c-th column of weight gradient $\frac{\partial \mathcal{L}}{\partial \boldsymbol{W}}$ and λ_{max} as the largest eigenvalue of pre-layer features $\boldsymbol{A}$. Then we have:*

$$\|\Delta\boldsymbol{y}_c\|_2^2 \leqslant \|\Delta\dot{\boldsymbol{y}}_c\|_2^2\Big(\frac{(1-p)\psi_c + p\psi_{min}}{\psi_c}\Big)^2 - \frac{2(1-p)p\psi_{min}}{B\psi_c}(\Delta\dot{\boldsymbol{y}}_c^T \overset{\smile}{\boldsymbol{y}}_c)^2.$$

$$\|\Delta\boldsymbol{w}_c\|_2^2 \leqslant \lambda_{max}^2\|\Delta\boldsymbol{y}_c\|_2^2.$$

Proof: Aligning with Theorem 4.1 from [80] we derive the gradients of the latent feature and the weight. For convenience, we define $\dot{\boldsymbol{Y}}$ as the resulted interpolated batch features from Eq. 14. By applying the expectation over the $\boldsymbol{M}$, replacing it with p, and using the chain rule of the backward propagation, we determine the expected gradient for each $\Delta y_c^{(b)}$ within $\Delta\boldsymbol{y}_c \in \mathbb{R}^B$ as follows:

$$\Delta y_c^{(b)} = \Delta\dot{y}_c^{(b)}(1-p) + p\frac{\psi_{min}}{\psi_c}\Big(\Delta\dot{y}_c^{(b)} - \overset{\smile}{y}_c^{(b)} \cdot \frac{1}{B}\sum_i^B(\Delta\dot{y}_c^{(i)} \cdot \overset{\smile}{y}_c^{(i)})\Big)$$

$$= \Delta\dot{y}_c^{(b)}\Big(\frac{(1-p)\psi_c + p\psi_{min}}{\psi_c}\Big) - p\frac{\psi_{min}}{\psi_c}\overset{\smile}{y}_c^{(b)}\frac{1}{B}\sum_{i=1}^B \Delta\dot{y}_c^{(i)} \cdot \overset{\smile}{y}_c^{(i)}.$$

16

The squared gradient norm for $\Delta \boldsymbol{y}_c$ is calculated as follows:

$$\|\Delta \boldsymbol{y}_c\|_2^2 = \|\Delta \dot{\boldsymbol{y}}_c\|_2^2 \Big(\frac{(1-p)\psi_c + p\psi_{\min}}{\psi_c} \Big)^2 - \Big(\frac{2(1-p)p\psi_{\min}}{B\psi_c} + \frac{p^2\psi_{\min}^2}{B\psi_c^2} \Big)(\Delta \dot{\boldsymbol{y}}_c^T \breve{\boldsymbol{y}}_c)^2$$

$$\leqslant \|\Delta \dot{\boldsymbol{y}}_c\|_2^2 \Big(\frac{(1-p)\psi_c + p\psi_{\min}}{\psi_c} \Big)^2 - \frac{2(1-p)p\psi_{\min}}{B\psi_c}(\Delta \dot{\boldsymbol{y}}_c^T \breve{\boldsymbol{y}}_c)^2.$$

Using the chain rule, we derive the gradient w.r.t. the weight as follows:

$$\frac{\partial \mathcal{L}}{\partial W_{ic}} = \sum_{b=1}^{B} \frac{\partial L}{\partial y_c^{(b)}} \frac{\partial y_c^{(b)}}{\partial W_{ic}} = \Delta \boldsymbol{y}_c^T \boldsymbol{a}_c.$$

This leads to:

$$\Delta \boldsymbol{w}_c = \boldsymbol{A}^T \Delta \boldsymbol{y}_c.$$

Considering $\lambda_{\max}$ as the largest eigenvalues of $\boldsymbol{A}$, which suggests $\boldsymbol{v}^T \boldsymbol{A} \boldsymbol{v} \leqslant \lambda_{\max} \|\boldsymbol{v}\|_2^2$, we obtain the following result:

$$\|\Delta \boldsymbol{w}_c\|_2^2 = \Delta \boldsymbol{y}_c^T \boldsymbol{A} \boldsymbol{A}^T \Delta \boldsymbol{y}_c \leqslant \lambda_{\max}^2 \|\Delta \boldsymbol{y}_c\|_2^2.$$

C. The decorrelation effect of the stochastic design M

To analyze why the stochastic design M outperforms the deterministic p, we examine the correlation coefficient between two random variables Y_i, Y_j from two different channels.

Theorem C.1 *Let Y_i, Y_j be random variables from the i-th and j-th channels, respectively, where $i \neq j$. Define $\widehat{Y}_i = \frac{Y_i}{\psi_i} \psi_{min}$ as the normalized random variable from channel i after root mean square normalization. Considering an adaptive p under our control, we distinguish between the deterministic version of CHAIN, i.e. CHAIN$_{Dtm.}$ and our stochastic CHAIN as:*

$$\text{Deterministic (CHAIN}_{Dtm.}\text{):} \quad Y_i' = (1-p)Y_i + p\widehat{Y}_i, \tag{22}$$

$$\text{Stochastic (CHAIN):} \quad \dot{Y}_i = (1-m)Y_i + m\widehat{Y}_i, \text{ where } m \sim \mathcal{B}(p). \tag{23}$$

Assuming $\mathbb{E}[Y_i] = \mathbb{E}[Y_j] = 0$, achievable through our zero mean regularization in Eq. 12, and letting $\sigma_i, \sigma_i', \dot{\sigma}_i$ represent the standard deviations of $Y_i, Y_i', \dot{Y}_i$, respectively, we define and relate the correlation coefficients of the two versions as follows:

$$\varrho_{ij}' = \frac{Cov(Y_i', Y_j')}{\sigma_i' \sigma_j'} \quad \geqslant \quad \dot{\varrho}_{ij} = \frac{Cov(\dot{Y}_i, \dot{Y}_j)}{\dot{\sigma}_i \dot{\sigma}_j}. \tag{24}$$

Theorem C.1 reveals that the stochastic CHAIN has a lower correlation coefficient among features from different channels than the deterministic CHAIN$_{Dtm.}$, indicating that the stochastic design M exhibits a decorrelation effect.

Proof: Given $\mathbb{E}[Y_i] = 0$, it follows that $\mathbb{E}[Y_i'] = \mathbb{E}[\dot{Y}_i] = 0$. Using the covariance definition $Cov(Z_1, Z_2) = \mathbb{E}[(Z_1 - \mu_{Z_1})(Z_2 - \mu_{Z_2})]$ for any two random variables Z_1, Z_2, we get:

$$Cov(Y_i', Y_j') = \mathbb{E}[Y_i' Y_j'], \quad Cov(\dot{Y}_i, \dot{Y}_j) = \mathbb{E}[\dot{Y}_i \dot{Y}_j].$$

Since m is stochastic noise independent of $Y_i, \widehat{Y}_i$, and $m \sim \mathcal{B}(p)$ implying in $\mathbb{E}[m] = p$, we conclude:

$$\mathbb{E}[Y_i' Y_j'] = \mathbb{E}[\dot{Y}_i \dot{Y}_j] \rightarrow Cov(Y_i', Y_j') = Cov(\dot{Y}_i, \dot{Y}_j). \tag{25}$$

Next, we explore the relationship between the variances $\sigma_i'^2$ and $\dot{\sigma}_i^2$:

$$\sigma_i'^2 = \mathbb{E}[Y_i'^2] - \mathbb{E}[Y_i']^2 = \mathbb{E}\Big[\Big((1-p+\frac{p\psi_{\min}}{\psi_i})Y_i \Big)^2 \Big] - 0 = \Big(1-p+\frac{p\psi_{\min}}{\psi_i} \Big)^2 \mathbb{E}[Y_i^2], \tag{26}$$

$$\dot{\sigma}_i^2 = \mathbb{E}[\dot{Y}_i^2] - \mathbb{E}[\dot{Y}_i]^2 = (1-p)\mathbb{E}[Y_i^2] + p\mathbb{E}[\widehat{Y}_i^2] - 0 = (1-p+p\frac{\psi_{\min}^2}{\psi_i^2})\mathbb{E}[Y_i^2]. \tag{27}$$

Comparing Eq. 26 and 27, and considering $p \in [0, 1]$, we establish the following relationship:

$$\Big(1-p+p\frac{\psi_{\min}^2}{\psi_i^2} \Big) - \Big(1-p+\frac{p\psi_{\min}}{\psi_i} \Big)^2 = p(1-p) + p(1-p)\frac{\psi_{\min}^2}{\psi_i^2} - 2p(1-p)\frac{\psi_{\min}}{\psi_i}$$

$$= p(1-p)\Big(1 - \frac{\psi_{\min}}{\psi_i} \Big)^2 \geqslant 0.$$

Therefore, $\sigma_i' \leqslant \dot{\sigma}_i$, and similarly $\sigma_j' \leqslant \dot{\sigma}_j$. Coupled with Eq. 25, we derive the following conclusion:

$$\begin{cases} Cov(Y_i', Y_j') = Cov(\dot{Y}_i, \dot{Y}_j) \\ \sigma_i' \sigma_j' \leqslant \dot{\sigma}_i \dot{\sigma}_j \end{cases} \rightarrow \varrho_{ij}' = \frac{Cov(Y_i', Y_j')}{\sigma_i' \sigma_j'} \geqslant \dot{\varrho}_{ij} = \frac{Cov(\dot{Y}_i, \dot{Y}_j)}{\dot{\sigma}_i \dot{\sigma}_j}.$$

Experimental validation. Decorrelation diversifies feature patterns, promoting a higher feature rank. This is demonstrated in Figure 8, where CHAIN, employing the stochastic M over the deterministic value p used by CHAIN$_{Dtm.}$, achieves a higher effective rank (eRank) [78]. This supports Theorem C.1, underscoring the beneficial effect of stochastic design in M for decorrelation, and validates the design choice of CHAIN.
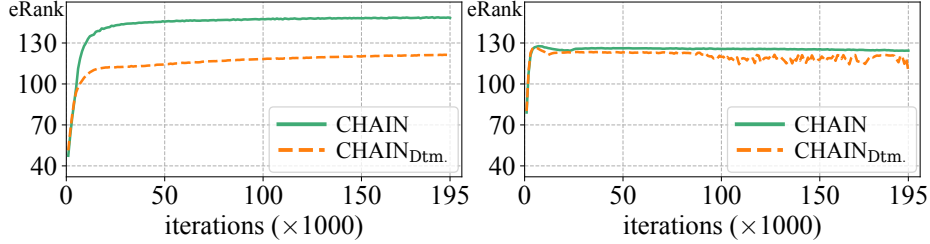
(a) 10% CIFAR-10 with OmniGAN ($d = 256$).   (b) 10% CIFAR-100 with BigGAN ($d = 256$).

Figure 8. Effective rank [78] of all pre-activation features in the discriminator for CHAIN and CHAIN$_{Dtm.}$ on (a) 10% CIFAR-10 using OmniGAN ($d = 256$) and (b) 10% CIFAR-100 with BigGAN ($d = 256$).

D. Implementation Details

In this section, we overcome the mini-batch size limitation of CHAIN$_{batch}$, which relies solely on current batch data statistics, by developing it to CHAIN, which ultilizes cumulative running forward/backward statistics across training. We also provide detailed implementation for Network and hyper-parameter choices, and methods applied in our ablation studies.

D.1. Implementation of CHAIN (running cumulative forward/backward statistics across training)

Inspired by [30, 85, 107], we enhance CHAIN to use running cumulative forward/backward statistics. We simplify our analysis by focusing on the Root Mean Square Normalization (RMSNorm), considering features of a single channel and omitting the channel index. Additionally, we exclude the constant ϵ, used to avoid division by zero, as it is unnecessary for this analysis. This refinement enables the representation of the forward process for the root mean square normalization as follows:

$$\psi^2 = \frac{1}{B} \sum_{b=1}^{B} (y^{(b)})^2, \tag{28}$$

$$\psi = \sqrt{\psi^2}, \tag{29}$$

$$\breve{y}^{(b)} = \frac{y^{(b)}}{\psi}, \tag{30}$$

$$\widehat{y}^{(b)} = \breve{y}^{(b)} \cdot \psi_{\min}. \tag{31}$$

Leveraging the chain rule, the gradient calculation can be expressed as follows:

$$\frac{\partial \mathcal{L}}{\partial \breve{y}^{(b)}} = \frac{\partial \mathcal{L}}{\partial \widehat{y}^{(b)}} \cdot \psi_{\min}, \tag{32}$$

$$\frac{\partial \mathcal{L}}{\partial y^{(b)}} = \frac{1}{\psi} \left[\frac{\partial \mathcal{L}}{\partial \breve{y}^{(b)}} - \breve{y}^{(b)} \cdot \Psi \right], \tag{33}$$

$$\text{where} \quad \Psi = \frac{1}{B} \sum_{i=1}^{B} \frac{\partial \mathcal{L}}{\partial \breve{y}^{(i)}} \cdot \breve{y}^{(i)}. \tag{34}$$

Examining the forward and backward processes reveals that Eq. 28 and 34 are dependent on the batch size. To eliminate this dependency, we propose updating the cumulative statistics for these terms as follows:

$$\overline{\psi^2}_{t+1} = \overline{\psi^2}_t \cdot \alpha_d + \psi^2 \cdot (1 - \alpha_d), \tag{35}$$

$$\overline{\Psi}_{t+1} = \overline{\Psi}_t \cdot \alpha_d + \Psi \cdot (1 - \alpha_d), \tag{36}$$

where α_d, a decay hyperpamameter, is typically set as 0.9. We replace ψ^2, Ψ with their cumulative versions $\overline{\psi^2}, \overline{\Psi}$. This forms an effective algorithm for the normalization part of CHAIN, using cumulative forward/backward statistics, as shown in Alg. 1

Algorithm 1: PyTorch-style pseudo code for Root Mean Square Normalization (RMSNorm) in CHAIN.

```
# Y:BxdxHxW feature, running_psi_sqr:ψ̄², decay:α_d, eps:a small constant
def RMSNorm_forward(Y, running_psi_sqr, decay=0.9, eps=1e-5):
    psi_sqr=Y.square().mean(axis=[0,2,3], keepdim=True) # Eq.28
    running_psi_sqr.data.mul_(decay).add_(psi_sqr, alpha=1-decay) # Eq.35
    running_psi=(running_psi_sqr + eps).sqrt() # Eq.29
    psi_min = running_psi.min().detach()
    Ycheck = Y / running_psi # Eq.30
    return Ycheck * psi_min # Eq.31

# grad_Yhat:BxdxHxW ∂L/∂Y, running_psi:ψ̄, running_Psi_grad:Ψ̄, psi_min:ψ_min decay:α_d
def RMSNorm_backward(grad_Yhat, Ycheck, running_psi, running_Psi_grad, psi_min, decay=0.9):
    grad_Ycheck = grad_Yhat * psi_min # Eq.32
    Psi_grad = (Ycheck * grad_Ycheck).mean(axis=[0,2,3], keepdim=True) # Eq.34
    running_Psi_grad.data.mul_(decay).add_(Psi_grad, alpha=1-decay) # Eq.36
    return (grad_Ycheck - Ycheck * running_Psi_grad) / running_psi # Eq.33
```

D.2. Network and hyper-parameters

CIFAR-10/100. We utilize OmniGAN ($d = 256$ and 1024) and BigGAN ($d = 256$) with a batch size of 32. Following [120], OmniGAN and BigGAN are trained for $1K$ epochs on full data and $5K$ epochs on 10%/20% data setting. CHAIN is integrated into the discriminator, after convolutional layers $c \in \{C_1, C_2, C_S\}$ at all blocks $l \in \{1, 2, 3, 4\}$, with hyperparameters set as $\Delta_p = 0.001, \tau = 0.5, \lambda = 20$.

ImageNet. We build CHAIN upon BigGAN with 512 batch size. We adopt learning rate of 1e-4 for generator and 2e-4 for discriminator. CHAIN is applied after convolutional layers $c \in \{C_1, C_2, C_S\}$ at all blocks $l \in \{1, 2, 3, 4, 5\}$, with hyperparameters $\Delta_p = 0.001, \tau = 0.5, \lambda = 20$.

5 Low-shot images (256×256). We build CHAIN upon StyleGAN2 with a batch size of 64, training until the discriminator has seen 25M real images. CHAIN is applied after convolutions $c \in \{C_1, C_2\}$ at blocks $l \in \{3, 4, 5, 6\}$. We set $\Delta_p = 0.0001, \tau = 0.9, \lambda = 0.05$.

7 Few-shot images (1024×1024) We replace the large discriminator in FastGAN with the one from BigGAN while removing the smaller discriminator. This modification yields FastGAN$-D_{\text{big}}$, with the discriminator network architecture illustrated in Figure 9. We employ a batch size of 8 and run for $100K$ iterations. We equip the discriminator with CHAIN after convolutional layers $c \in \{C_1, C_2, C_S\}$ at blocks $l \in \{1, 2, 3, 4, 5\}$. We set $\Delta_p = 0.001, \tau = 0.5, \lambda = 20$.
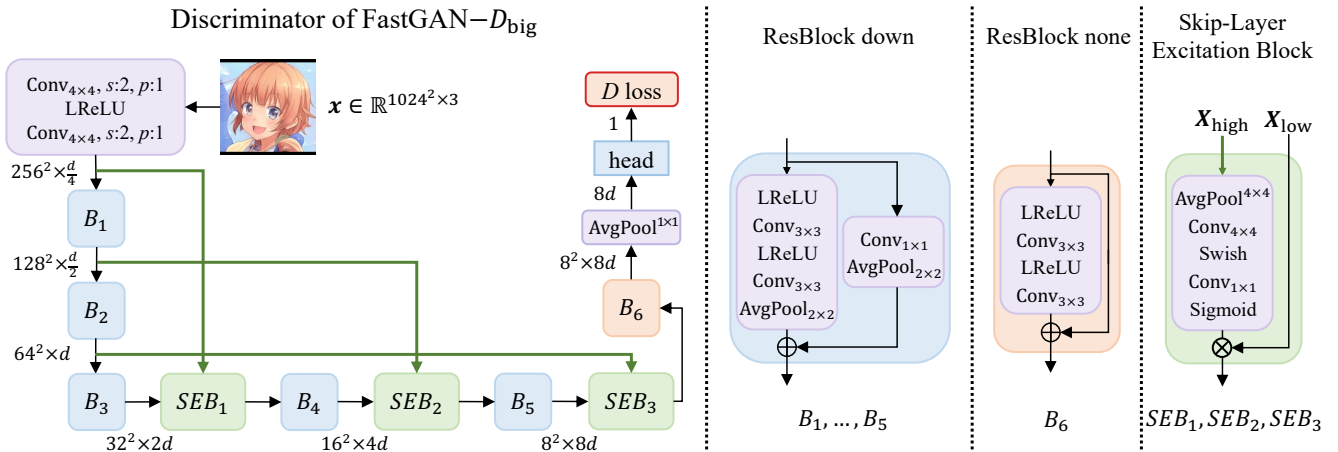

Figure 9. The discriminator of FastGAN$-D_{\text{big}}$. d: The base feature dimension. Conv$_{4\times4}$: A convolutional layer with a 4×4 kernel size. LReLU: Leaky ReLU activation with a slope 0.2. AvgPool$_{2\times2}$: Average pooling downscales by a factor of 2. AvgPool$^{4\times4}$: Adaptive average pooling with a 4×4 output spatial size. X_{high}: The higher resolution feature map. X_{low}: The lower resolution feature map. For more details on skip-layer excitation block, please refer to [57] and [28].

D.3. Implementation for AGP$_{\text{input}}$ and AGP$_{\text{weight}}$

In Table 6, we provide a comparison of CHAIN with two gradient penalization methods: AGP$_{\text{input}}$ and AGP$_{\text{weight}}$. For AGP$_{\text{input}}$, we implement $\|\frac{\partial D}{\partial \boldsymbol{x}}\|_2^2$ and $\|\frac{\partial f}{\partial \boldsymbol{x}}\|_2^2$ where f represents the feature extractor of discriminator D. Regarding AGP$_{\text{weight}}$, we also implement $\|\frac{\partial D}{\partial \boldsymbol{\theta}_d}\|_2^2$ and $\|\frac{\partial f}{\partial \boldsymbol{\theta}_d}\|_2^2$. We search the penalization strength λ_{GP} within the range [1e-10, 20] for each dataset and variant. For 10% CIFAR-10 w/ OmniGAN ($d=256$), the optimal settings are: AGP$_{\text{input}}$ with $\|\frac{\partial f}{\partial \boldsymbol{x}}\|_2^2$ and $\lambda_{\text{GP}}=5$, and AGP$_{\text{weight}}$ with $\|\frac{\partial f}{\partial \boldsymbol{\theta}_d}\|_2^2$ and λ_{GP} set to 1e-6. For 10% CIFAR-100 w/ BigGAN ($d=256$), the best configurations are: AGP$_{\text{input}}$ with $\|\frac{\partial f}{\partial \boldsymbol{x}}\|_2^2$ and $\lambda_{\text{GP}}=5$, and AGP$_{\text{weight}}$ with $\|\frac{\partial D}{\partial \boldsymbol{\theta}_d}\|_2^2$ and λ_{GP} set to 2e-6.

E. Additional Experiments

E.1. Comparison with leading methods

Table 7 compares CHAIN with Lottery-GAN [12], LCSA [70], AugSelf-GAN [27], and NICE [68], showing the superiority of CHAIN. Unlike AugSelf-GAN, LotteryGAN, and NICE, which need extra forward or backward passes for augmentation, and LCSA, which demands more computation and weights for dictionary learning, CHAIN is more efficient, needing negligible computation for normalization.

Table 7. Comparing CIFAR-10/100 results with varying data percentages, using CHAIN *vs.* other leading methods, on BigGAN ($d=256$).

| Method | CIFAR-10 | | | | | | | | | CIFAR-100 | | | | | | | | |
| | 10% data | | | 20% data | | | 100% data | | | 10% data | | | 20% data | | | 100% data | | |
	IS↑	tFID↓	vFID↓	IS↑	tFID↓	vFID↓	IS↑	tFID↓	vFID↓	IS↑	tFID↓	vFID↓	IS↑	tFID↓	vFID↓	IS↑	tFID↓	vFID↓
LeCam+DA	8.81	12.64	16.42	9.01	8.53	12.47	9.45	4.32	8.40	9.17	22.75	27.14	10.12	15.96	20.42	11.25	6.45	11.26
+Lottery-GAN	8.77	11.47	15.48	8.99	7.91	11.83	9.39	4.21	8.25	9.05	20.63	25.31	9.55	15.18	20.01	11.28	6.32	11.10
+LCSA	8.96	10.05	13.88	9.04	6.95	10.95	9.47	3.75	7.83	**10.28**	18.24	23.12	**10.67**	10.16	15.00	11.17	5.85	10.64
+NICE	8.99	9.86	13.81	9.12	6.92	10.89	9.52	3.72	7.81	9.35	14.95	19.60	10.54	10.02	14.93	11.28	5.72	10.40
+AugSelf-GAN	**9.04**	8.98	12.94	9.13	6.42	10.54	9.48	3.68	7.73	9.89	14.02	18.84	10.43	11.32	16.02	11.25	5.43	10.14
+CHAIN	8.96	**8.54**	**12.51**	**9.27**	**5.92**	**9.90**	9.52	**3.51**	**7.47**	10.11	**12.69**	**17.49**	10.62	**9.02**	**13.75**	**11.37**	**5.26**	**9.85**

E.2. Gradient analysis on 10% CIFAR-100 using BigGAN ($d=256$)

In this section, we present experiments conducted on 10% CIFAR-100 using BigGAN ($d=256$). Figure 10 provides additional validation of Theorem 3.1, illustrating how the centering step leads to feature differences and an associated increase in gradients. Meanwhile, Figure 11 confirms Theorem 3.2, highlighting that the scaling step causes gradient explosions during GAN training and results in rank deficiency.

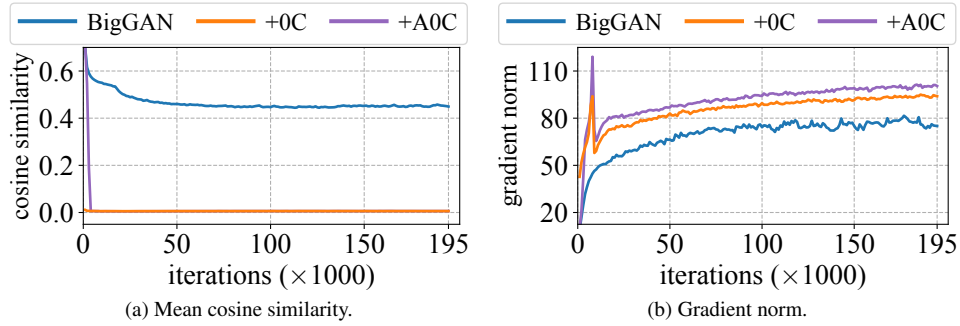(a) Mean cosine similarity.   (b) Gradient norm.

Figure 10. (a) Mean cosine similarity of discriminator pre-activation features, and (b) gradient norm of the feature extractor w.r.t. the input are evaluated for BigGAN, BigGAN+0C (using the centering step in Eq. 6), and BigGAN+A0C (adaptive interpolation between centered and uncentered features). Evaluation conducted on 10% CIFAR-100 data with BigGAN ($d=256$).

E.3. The rank efficiency of CHAIN over AGP$_{\text{weight}}$

Both CHAIN and AGP$_{\text{weight}}$ can reduce the discriminator weight gradient to improve generalization, but CHAIN gains a crucial advantage from normalization. The normalization step in CHAIN balances features among channels and orthogonalizes features [16, 17]. Figure 12 clearly illustrates that CHAIN achieves a higher effective rank compared to AGP$_{\text{weight}}$. Discriminators with higher rank efficiency can fully utilize their width (balanced channels) and depth, resulting in enhanced expressivity and superior representation capability.
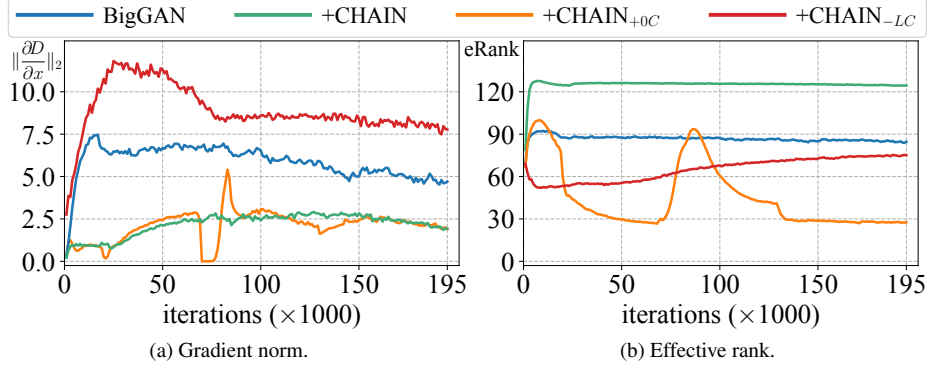
(a) Gradient norm.

(b) Effective rank.

Figure 11. (a) Gradient norm of discriminator output w.r.t. input during training, and (b) effective rank [78] of the pre-activation features in discriminator, are evaluated on 10% CIFAR-100 data with BigGAN ($d = 256$). CHAIN$_{+0C}$ indicates CHAIN with the centering step included, while CHAIN$_{-LC}$ represents CHAIN without the Lipschitzness constraint.

(a) 10% CIFAR-10 with OmniGAN ($d = 256$).
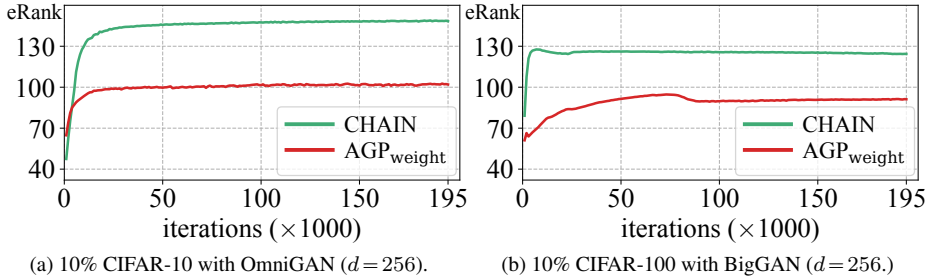
(b) 10% CIFAR-100 with BigGAN ($d = 256$.)

Figure 12. Effective Rank [78] for CHAIN and AGP$_{weight}$ on (a) 10% CIFAR-10 using OmniGAN ($d = 256$) and (b) 10% CIFAR-100 with BigGAN ($d = 256$).

E.4. The stability of feature norm of CHAIN during training

Our work examines modern discriminators with residual blocks, where the main and skip branch features are added at the end of each block (see Figure 1). Despite the scaling factor $\leqslant 1$ induced by the Lipschitz constraint (as in Eq. 13), feature norms remain stable across layers thanks to the skip connections. Figure 13 presents feature norms at the end of each block, averaged over early ($0 - 5k$ iteration) and later training stages ($> 5k$ iteration). Initially, both methods exhibit similar feature norms, but as training processes, baseline norms increase while CHAIN maintains stable norms across layers due to the adaptive interpolation between normalized and unnormalized features (as in Eq. 14).
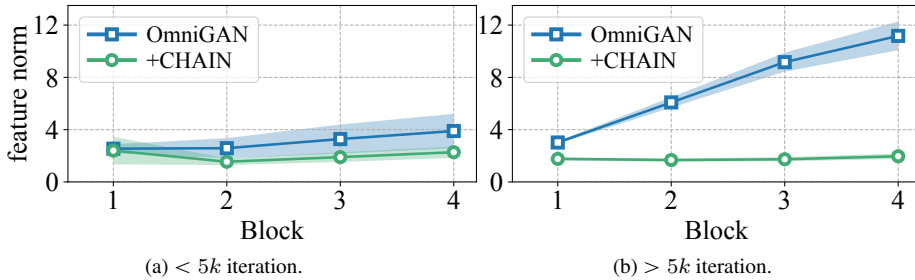

(a) $< 5k$ iteration.

(b) $> 5k$ iteration.

Figure 13. Feature norms during training w/ *vs*. w/o CHAIN, are evaluated on 10% CIFAR-10 using OmniGAN ($d = 256$).

F. Training overhead

Table 8 presents the number of parameters, multiply-accumulate (MACs) operations (for both generator and discriminator), the number of GPUs, and the cost in time (seconds per 1000 images, secs/$k$img). Notably, CHAIN introduces only a small fraction of the time cost, ranging from 6.3% to 9.6% across these datasets.

Table 8. Number of parameters, MACs and secs/$k$img for models with *vs.* without the CHAIN. Experiments were performed on NVIDA A100 GPUs.

Dataset	Resolution	Backbone	d	GPUs	Baseline			+CHAIN		
					#Par.	MACs	sec/$k$img	#Par.	MACs	sec/$k$img
CIFAR-10	32×32	BigGAN	256	1	8.512M	2.788G	0.79	8.512M	2.791G	0.84
		OmniGAN			8.512M	2.788G	0.80	8.512M	2.790G	0.85
CIFAR-100	32×32	BigGAN	256	1	8.811M	2.788G	0.80	8.811M	2.791G	0.85
		OmniGAN			8.811M	2.788G	0.81	8.811M	2.791G	0.85
ImageNet	64×64	BigGAN	384	2	115.69M	18.84G	1.79	115.69M	19.12G	1.91
5 Low-shot datasets	256×256	StyleGAN2	512	2	48.77M	44.146G	5.66	48.77M	44.151G	6.06
7 Few-shot datasets	1024×1024	FastGAN$-D_{\text{big}}$	64	1	42.11M	23.98G	32.79	42.11M	24.00G	35.94

G. Generated Images

Figures 14, 15, 16, 17 and 18 provide images generated on CIFAR-10, CIFAR-100, ImageNet, the 5 low-shot image and the 7 few-shot image datasets, with or without CHAIN. The comparison highlights the enhancement in image quality and diversity achieved with the application of CHAIN.

(a) ADA

(b) ADA+CHAIN

Figure 14. Generated images using (a) ADA and (b) ADA+CHAIN on 10% CIFAR-10 with OmniGAN ($d = 1024$). Note that ADA leaks the rotation augmentation artifacts (row 1, 2 and 10).

(a) DA

(b) DA+CHAIN

Figure 15. Generated images using (a) DA and (b) DA+CHAIN on 10% CIFAR-100 with BigGAN ($d = 256$). We present the last 20 of 100 classes. CHAIN clearly enhances the diversity and quality of the generated images. Notably, DA leaks the cutout augmentation artifacts (row 1, 2, 4 and 18).
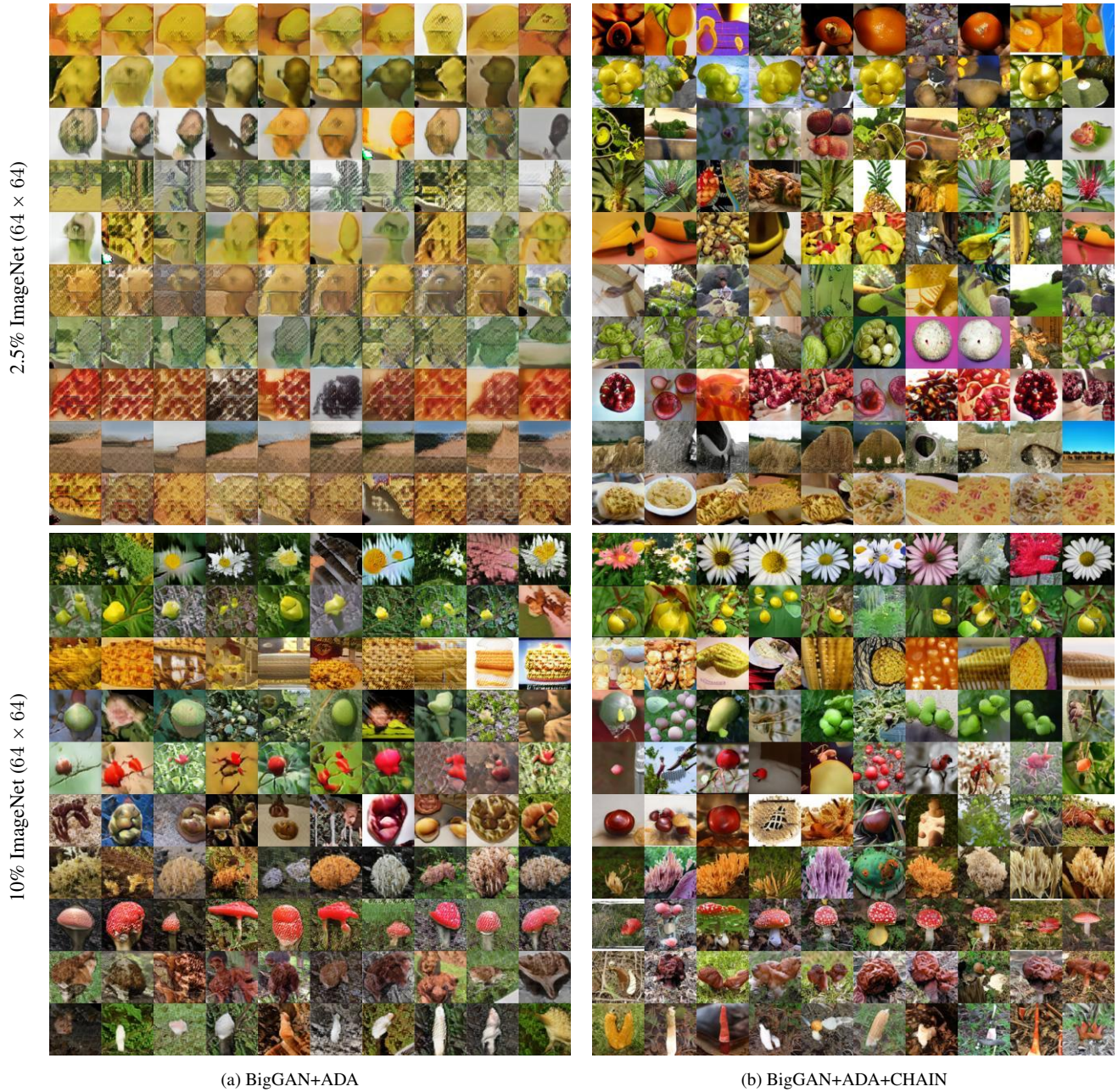
2.5% ImageNet (64 × 64)

10% ImageNet (64 × 64)

(a) BigGAN+ADA

(b) BigGAN+ADA+CHAIN

Figure 16. Visual comparison between ADA *vs*. ADA+CHAIN on 2.5% and 10% ImageNet(64 × 64) data. ADA struggles to capture the structure and diversity of the data, while CHAIN clear improves the diversity and visual quality of generated images.

Figure 17. Visual comparison between ADA and ADA+CHAIN on 100-shot and AnimalFace datasets (256×256). The integration of CHAIN clearly improves the image quality.
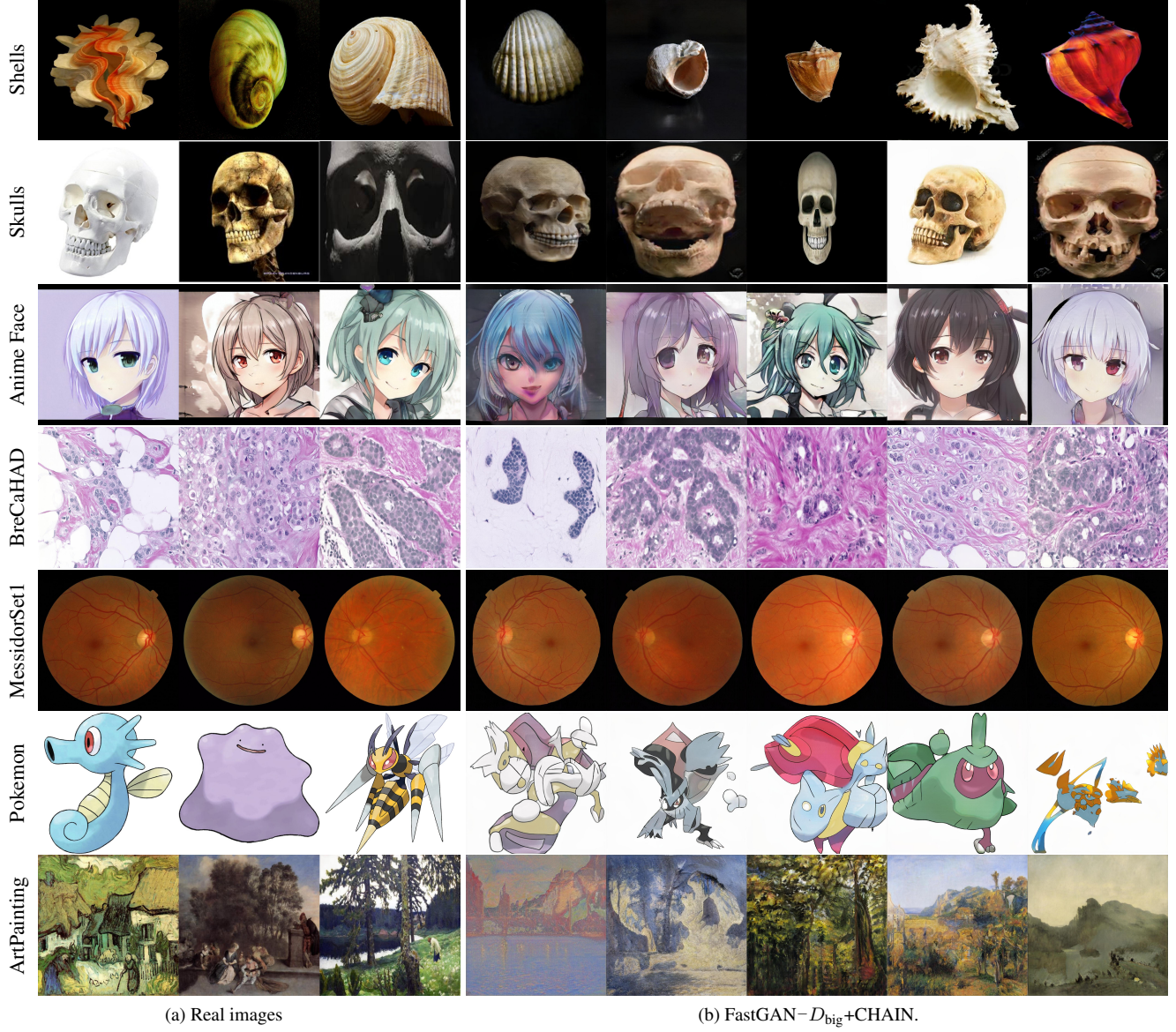
Figure 18. Qualitative results of FastGAN$-D_{\mathrm{big}}$+CHAIN on 7 few-shot image dataset (1024×1024). (a) shows real training images and the (b) presents images generated by FastGAN$-D_{\mathrm{big}}$+CHAIN. CHAIN is capable of generating photo-realistic images with fine details even from a limited number of training samples.