Self-Distilling Generative Adversarial Networks

Antoni Nowinowski
Poznan University of Technology
antoni.nowinowski@cs.put.poznan.pl

Krzysztof Krawiec
Poznan University of Technology
krzysztof.krawiec@cs.put.poznan.pl

Abstract

In modern GANs, maintaining an Exponential Moving Average (EMA) of the generator's weights is a standard practice, as such an averaged model consistently outperforms the actively trained generator. However, the EMA generator is used for final deployment only and does not influence the training process. To address this missed opportunity, we introduce Self-Distilled GAN (SD-GAN) that employs the EMA generator as a teacher to guide the active generator (student) via perceptual loss. We prove the local asymptotic stability of SD-GAN in the simplified continuous-time Dirac-GAN setting and show that it dampens the parasitic cycling behavior that plagues conventional GANs. Empirical evaluations across established architectures and datasets demonstrate that SD-GAN improves the final image quality on several metrics (FID and random-FID in particular), stabilizes the optimization trajectory and provides additional learning guidance that is not trivially correlated with the conventional adversarial loss. It also proves effective for fine-tuning pretrained GAN models.

1 Introduction

Obtaining well-performing generators by training generative adversarial networks (GANs) [1] can be notoriously challenging, one reason being that the game played between the generator (G) and the discriminator (D) tends to engage in a cyclic "chase" or entirely fail to converge [2]. The resulting G may then fail to capture the entirety of the distribution represented by the training sample or produce samples of inferior quality. To address this challenge, it is common to obtain final generators from GANs using weight averaging, which is typically performed outside or inside the training loop. In the former case, the averaged model is kept "on the side" and does not influence the training process in any way, being used only for the post-training evaluations, as in stochastic weight averaging [3]. In the latter case, the averaged model is used as a teacher in self-distillation (SD) student-teacher mode, e.g., in Mean Teachers [4], BYOL [5], and the DINO family of models [6, 7, 8]. In most of those scenarios, it is common to aggregate the models using exponential moving average (EMA), which naturally attributes more importance to the more recent models. Specifically, at each training step t, the EMA weights θ_{EMA} are updated using the weights θ of the active model at decay rate β:

$$\theta_{EMA}^{(t)} = \beta\theta_{EMA}^{(t-1)} + (1-\beta)\theta^{(t)}. \tag{1}$$

In this study, we propose Self-Distilled GAN (SD-GAN) that incorporates SD into GAN training (Fig. 1): rather than learning from the adversarial loss alone, the generator G also learns from its self-distilled teacher G_{EMA} obtained from G using EMA. To this end, G and G_{EMA} are queried on the same latent vector and the similarity of the images they produce is assessed using a popular form of perceptual loss (LPIPS, [9]). This changes the training trajectory of G, and subsequently the discriminator D. We show that this extension of the traditional GAN paradigm stabilizes training, makes it faster and ultimately results in generators that produce more realistic images.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

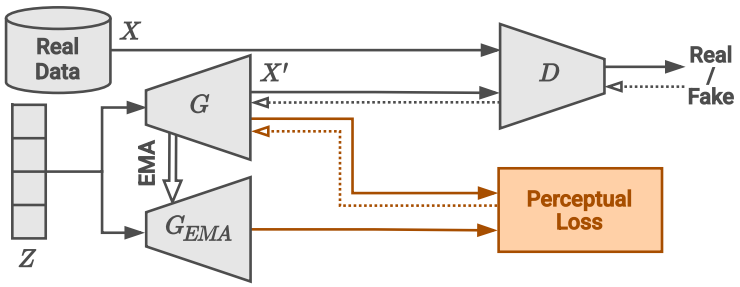


Figure 1: Overview of the SD-GAN framework. The training is guided by both the conventional GAN loss and the self-distillation loss. Dotted lines indicate gradient flow; gradients from the perceptual loss do not propagate through G_{EMA}.

2 Related Work

Weight averaging in GANs. An early attempt to stabilize GAN training by leveraging the historical states of the players was *historical averaging* [10], which penalized both G and D by the distance between the current and the historical parameters, defined as $\|\theta^{(t)} - \frac{1}{T}\sum_{i=1}^{T}\theta^{(t-i)}\|^2$. However, it neither maintains a shadow generator for inference nor uses exponential discounting.

To our knowledge, maintaining an EMA of the weights of a GAN generator was first introduced as part of the Progressive Growing of GANs (ProGAN) training framework [11]. In order to maintain the generator's responsiveness, the authors opted not to use learning rate decay. Consequently, the weights fluctuated significantly during training, leading to visual artifacts. To dampen this variance and produce stable outputs, they maintained a passive EMA of the generator's weights.

The above techniques were introduced as practical engineering heuristics to improve inference quality, without theoretical justification or citations to the prior literature. To bridge this gap, Yazici et al. [12] conducted a formal analysis of weight averaging in GANs. They noted that standard gradient descent in adversarial minimax games frequently fails to converge, as the players continuously orbit the optimal solution in limit cycles. Through a theoretical analysis, they proved that while EMA does not directly achieve the Nash equilibrium, it exponentially shrinks the amplitude of these limit cycles. Consequently, the active G model continues exploring the loss landscape, while the passive G_{EMA} is confined to the neighborhood of the equilibrium point. Extensive empirical evaluations confirmed that EMA models consistently achieve higher image fidelity than their actively trained counterparts. This dynamic has also been studied in [13] across a set of image classification tasks, revealing that EMA models reliably converge to fundamentally different, flatter basins of the loss landscape, not only improving final generalization but yielding remarkable early-stage performance, largely explaining their success when used as teachers.

Following the introduction in ProGAN, EMA of generator weights became a standard component of most state-of-the-art generative models. The technique was adopted in BigGAN [14], the StyleGAN lineage of models [15, 16, 17, 18], ProjectedGAN [19], StyleGAN-XL [20], StyleGAN-T [21], SAN [22], and StyleNAT [23], as well as in diffusion models [24, 25, 26]. However, in contrast to SD-GAN, in all these architectures, G_{EMA} serves as the end-of-run outcome, while not directly impacting the training process.

Stabilization and consistency regularization. Alongside weight averaging, various regularization techniques have been proposed to stabilize the adversarial game. For instance, CR-GAN [27] and bCR [28] apply consistency terms to the discriminator, penalizing prediction changes across differently augmented views of the same image, while methods like DiffAugment [29] apply differentiable augmentations to both real and fake samples to prevent discriminator overfitting. On the generator side, latent consistency regularization (zCR) [28] maximizes the difference between images generated from perturbed latent codes to promote diversity. Furthermore, LeCam regularization [30] utilizes discriminator EMA to anchor its predictions to historical states. While these methods are highly effective, they predominantly prevent discriminator overfitting or enforce latent consistency. In contrast, SD-GAN introduces temporal consistency directly into the generator's trajectory.

Knowledge distillation. In the modern generative landscape, knowledge distillation is widely utilized, though predominantly to improve computational efficiency rather than training stability. This includes distilling lightweight student GANs from larger teacher models [31, 32]. Concurrently, adversarial distillation techniques have been applied to diffusion models to reduce inference times, enabling high-fidelity sampling in a few steps [33, 34] or a single step [35]. Beyond model compression, some elements of self-distillation combined with GANs have also been used as a component for a specific supervised downstream task [36]. In contrast, SD-GAN utilizes it to directly improve the primary model's training trajectory and final image quality. Recently, consistency models [37], a new class of diffusion-inspired generative models, have similarly demonstrated the efficacy of EMA-based self-distillation, utilizing a slowly updated target network as a stable reference to guide the active model. Finally, SD-GAN should not be confused with Self-Distilled StyleGAN [38], as the latter focuses on generative self-filtering to curate noisy, multi-modally distributed web data.

3 Self-Distilled GAN

The motivation for SD-GAN is similar to that postulated in previous methods that engaged weight averaging and self-distillation [4]: (i) to provide the learner (student) with a form of long-term memory (the teacher) that integrates the effects of the learning process, and (ii) to use the teacher as an anchor to stabilize the training, i.e. to avoid catastrophic forgetting of the already acquired knowledge, while allowing for a fair share of exploration. This is particularly relevant in an adversarial game, where G and D can keep responding to one another without settling, yielding cyclic parameter trajectories [2]. Passive EMA smooths the G_{EMA} used for evaluation but does not affect the update of the active G. Here, in contrast, G_{EMA} participates in training by providing a smoothed reference that regularizes G. We implement this idea by extending the traditional GAN adversarial loss function with a self-distillation loss applied to the outputs of the active generator model G and the averaged model G_{EMA}. Specifically, G and G_{EMA} are queried on the same random latent vector z, and the images they produce are compared using an SD loss component $\mathcal{L}_{SD}$. The gradient calculated from $\mathcal{L}_{SD}$ affects only G, i.e. G_{EMA} is updated only via EMA (Fig. 1).

In general, $\mathcal{L}_{SD}$ can be any image-comparing loss. We use a perceptual loss in the main experiments and examine pixel-wise and discriminator-feature alternatives in Sec. 4. Specifically, we utilize LPIPS [9], which evaluates the distance between the feature activations of the two images when passed through a pretrained network. The objective function for the generator is a weighted sum of the standard adversarial loss $\mathcal{L}_{adv}$ and $\mathcal{L}_{SD}$:

$$\mathcal{L}_G = \mathcal{L}_{adv} + \alpha \cdot \mathcal{L}_{SD}(G(z), G_{EMA}(z)) \tag{2}$$

where α controls the weight of the guidance provided by the EMA teacher. In our implementation, we additionally apply the same differentiable random spatial transformation to both generated images before evaluating $\mathcal{L}_{SD}$ (Appendix C). It is not necessary for the self-distillation itself, but the results of the ablation in Fig. 5 suggest that this augmentation provides a small additional benefit.

Theoretical intuition. To better understand the stabilizing mechanism of SD-GAN, we analyze its behavior on the Dirac-GAN [2], a simple, one-dimensional model used to study the convergence of GANs. The Dirac-GAN consists of a true data distribution fixed at zero, a generator modeled as a constant function with a single learnable parameter $G(z) = \theta$, and a linear discriminator $D(x) = \psi \cdot x$. Consequently, the discriminator's score on generated data is $D(G(z)) = \psi\theta$, and the adversarial loss takes the form $\mathcal{L}_{adv} = f(\psi\theta)$ (where f is determined by the GAN variant; here f is identity). The unique Nash equilibrium of this system is $(\theta^*, \psi^*) = (0, 0)$.

Figure 2 illustrates the trajectories of these parameters during training. In the basic GAN (2a, green plot), gradient descent does not converge, with G and D persistently orbiting the equilibrium; incorporating EMA (blue plot) narrows the orbit, but does not improve convergence. The right inset (2b) demonstrates how the training dynamics change with the addition of our self-distillation objective $\mathcal{L}_{SD}$ (simplified to $\alpha(G(z) - G_{EMA}(z))^2$). By continuously penalizing the distance between the active generator and the EMA generator, $\mathcal{L}_{SD}$ introduces a damping force that transforms the circular, non-converging trajectory into a contracting spiral, forcing the parameters to converge in the limit to the exact Nash equilibrium. This happens both for the EMA (orange) and the non-EMA (indigo) configuration. This analytical model corroborates our empirical findings in the simplified Dirac-GAN scenario: by anchoring the active model to its historical average, SD-GAN mitigates oscillatory training dynamics and ensures stable convergence.

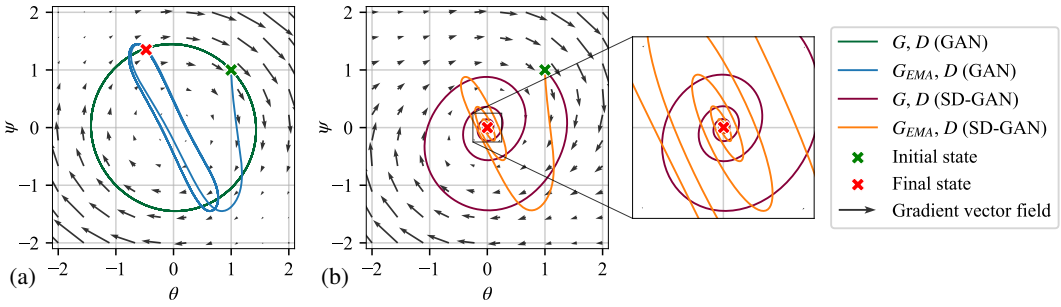


Figure 2: Parameter trajectories of the active generator (θ), EMA generator (θ_{EMA}) and discriminator (ψ) in the Dirac-GAN. **(a)** Standard GAN forms cycles orbiting the Nash equilibrium. **(b)** Self-distillation dampens these cycles into a contracting spiral. The arrows visualize the gradient vector field (b) for the final θ_{EMA}.

Proof of convergence. We follow the above intuition with a more rigorous formalization by examining the gradient vector field and deriving the exact local convergence criterion. As in prior analyses of local GAN stability, we study the continuous-time dynamics through the spectrum of the Jacobian of the vector field at the equilibrium [39, 40, 41, 2]. We consider a continuous-time formulation of the Dirac-GAN minimax game. We denote the active generator parameter as $\theta(t)$, the discriminator parameter as $\psi(t)$, and the EMA generator parameter as $\phi(t)$ (rather than $\theta_{EMA}(t)$, for brevity), and introduce continuous-time learning rates $\eta_G, \eta_D > 0$ for the generator and discriminator respectively. For the adversarial objective $\mathcal{L}_{adv}(\theta, \psi) = f(\psi\theta)$, the local behavior near the Nash equilibrium depends only on $c = f'(0)$, with $c \neq 0$, so we work with the linearized loss function $\mathcal{L}_{adv} = c\psi\theta$. The self-distillation penalty is $\alpha \cdot \mathcal{L}_{SD} = \frac{\alpha}{2}(\theta - \phi)^2$ (the $\frac{1}{2}$ factor is for convenience). We parametrize the EMA decay rate β as an update rate $\eta_\phi = 1 - \beta$. The joint vector field is therefore:

$$\begin{cases} \frac{d\theta}{dt} = -\eta_G(c\psi + \alpha(\theta - \phi)) \\ \frac{d\psi}{dt} = \eta_D c\theta \\ \frac{d\phi}{dt} = \eta_\phi(\theta - \phi) \end{cases} \tag{3}$$

To analyze local stability around the Nash equilibrium $(\theta^*, \psi^*, \phi^*) = (0, 0, 0)$, we construct the Jacobian matrix J of the state vector (θ, ψ, ϕ):

$$J = \begin{bmatrix} -\eta_G\alpha & -\eta_G c & \eta_G\alpha \\ \eta_D c & 0 & 0 \\ \eta_\phi & 0 & -\eta_\phi \end{bmatrix} \tag{4}$$

The system converges asymptotically to the Nash equilibrium if and only if all eigenvalues λ of J have negative real parts [39]. We solve the characteristic equation $\det(\lambda I - J) = 0$ by expanding along the second row:

$$\lambda^3 + (\eta_G\alpha + \eta_\phi)\lambda^2 + (\eta_D\eta_G c^2)\lambda + \eta_D\eta_G\eta_\phi c^2 = 0 \tag{5}$$

To make sure the eigenvalues λ have negative real parts, we apply the Routh–Hurwitz criterion [39], which states that all roots of a 3rd-degree polynomial ($a_3\lambda^3 + a_2\lambda^2 + a_1\lambda + a_0 = 0$) will have strictly negative real parts if and only if (i) $a_3, a_2, a_1, a_0 > 0$ and (ii) $a_2 a_1 > a_3 a_0$:

$$\text{(i)} \quad \begin{array}{ll} a_3\colon 1 > 0 & a_2\colon \eta_G\alpha + \eta_\phi > 0 \\ a_1\colon \eta_D\eta_G c^2 > 0 & a_0\colon \eta_D\eta_G\eta_\phi c^2 > 0 \end{array} \qquad \text{(ii)} \quad \eta_D\eta_G^2 c^2\alpha > 0 \tag{6}$$

Because the learning rates $\eta_G, \eta_D > 0$, the condition simplifies to $\alpha > 0$. Therefore, for any valid learning rates $\eta_G, \eta_D > 0$, and EMA decay rate $\beta \in [0, 1)$, introducing a self-distillation penalty $\alpha > 0$ strictly guarantees that the eigenvalues have negative real parts, proving local asymptotic convergence to the exact Nash equilibrium, rather than the oscillatory behavior of the unregularized Dirac-GAN [2].

4 Results

We validate SD-GAN by applying it to the well-established GAN architectures StyleGAN2-ADA [17], ProjectedGAN [19] and StyleNAT [23], and experimenting with the widely used benchmarks

FFHQ [15] and LSUN Church [42] at 256^2 resolution. We additionally evaluate StyleGAN-XL [20] on ImageNet at 64^2 resolution. We take care to faithfully reproduce the architectures and results of the cited works by relying on the original implementations (including metric calculation), and amending them only as minimally as required to extend them to SD-GAN. See Appendix A for further details. Despite our best efforts, we did not manage to replicate the results of StyleNAT on LSUN Church sufficiently well, so we discard this combination (see Appendix B for details). In addition to the default version of SD-GAN that uses LPIPS as $\mathcal{L}_{SD}$, we also consider a simplified variant that uses pixel-wise L_1 for that purpose. We set α to 1 for StyleGAN2-ADA and ProjectedGAN, 0.25 for StyleNAT, and 100 for StyleGAN-XL (the latter two adjusted for their $\mathcal{L}_{adv}$ scales). The values of β are set to the default values of the baselines to maintain a fair comparison.

Similar to other works, we use the Fréchet Inception Distance (FID, [43]) as the primary metric for measuring the quality of obtained generators. The FID estimates are calculated from a sample of 50k generated images and compared against the full dataset. However, to address the growing criticism regarding its vulnerability to ImageNet-specific biases [20, 44], we complement it with the random-FID (rFID, [20]). By employing an untrained feature extractor, rFID helps verify if improvements on FID result from fundamental gains in image quality rather than metric-specific artifacts. Furthermore, to rigorously test whether the self-distillation objective induces mode collapse or reduces diversity, we use Precision and Recall [45]. Precision estimates how much of the generated distribution lies near real examples, while Recall estimates how much of the real distribution is covered by generated examples. All reported results concern the EMA models.

Figure 3 presents the FID over training effort expressed in thousands of processed real images (kimgs). SD-GAN systematically achieves better FID values (notice log scale), and for the convolutional architectures (StyleGAN2-ADA and ProjectedGAN) does so already from the early iterations of learning. For ProjectedGAN, SD-GAN using LPIPS achieves the same FID as the baseline method at a fraction of the cost ($\sim 32\%$ on FFHQ and $\sim 39\%$ on LSUN Church). Using L_1 as $\mathcal{L}_{SD}$ leads to somewhat worse results than LPIPS, however, usually still better than the baseline. The smoother curves for SD-GAN suggest that rather than slowing exploration or anchoring to the historical state, the distillation objective acts as a stabilizing force, corroborating the theoretical findings from Sec. 3.

Table 1 compares the final metrics achieved by reference methods and their SD-GAN variants. As we observed rFID to vary significantly across independent evaluations due to random feature projection inherent to this metric [20], we compute it five times and report means and standard deviations. The advantage of self-distillation is particularly prominent on this metric, especially for the LPIPS-based SD-GAN; e.g., the ProjectedGAN's rFID drops from 79.3 ± 7.6 to 14.1 ± 2.1 on FFHQ. Thus, the EMA teacher gains hold also for an untrained feature extractor (beyond ImageNet-biased FID). When applied to StyleGAN2-ADA and ProjectedGAN, SD-GAN also tends to improve Recall or Precision, while only moderately deteriorating the other metric, demonstrating that the model maintains diversity and, contrary to potential concerns, is not overly anchored to the averaged state by self-distillation. While SD-GAN yields consistent benefits for convolutional baselines, we observe a slight FID regression for the transformer-based StyleNAT (2.17 to 2.27 on FFHQ), despite initial gains visible in Fig. 3, although the baseline overtook the SD variant only after learning-rate decay began (vertical dashed line). For StyleGAN-XL+SD$_{ramp}$, we delay SD and ramp up α as discriminator blurring [20, 18] fades, to avoid reinforcing high-frequency details hidden by blur.

Table 2 presents the training times and relative computational overheads of training SD-GAN on a single GPU. See Appendix A for the details on our computational infrastructure.

4.1 Sensitivity to hyperparameter settings

We conducted sensitivity analysis for ProjectedGAN on FFHQ at 256^2 across a grid of self-distillation weights ($\alpha \in \{0.5, 1.0, 2.0\}$) and EMA decay rates β, set as to correspond to half-life ($t_{1/2} \in \{10, 20, 40\}$ kimg), so the decay rate is defined as $\beta = 0.5^{b/(1000 t_{1/2})}$ where batch size $b = 64$. Table 3 summarizes the resulting FID scores. The default configuration ($\alpha = 1.0$, $t_{1/2} = 20$ kimg, FID $= 3.15$) used in the experiments reported in Table 1 corresponds to the center cell (FID $= 3.21$, the difference from Table 1 reflects normal run-to-run variance).

Notably, each of these hyperparameter combinations outperforms the baseline ProjectedGAN without SD (FID of 3.90, cf. Table 1), demonstrating that the benefit of EMA-based self-distillation is robust and not overly sensitive to precise tuning. Doubling both the EMA half-life (20 to 40) and the SD

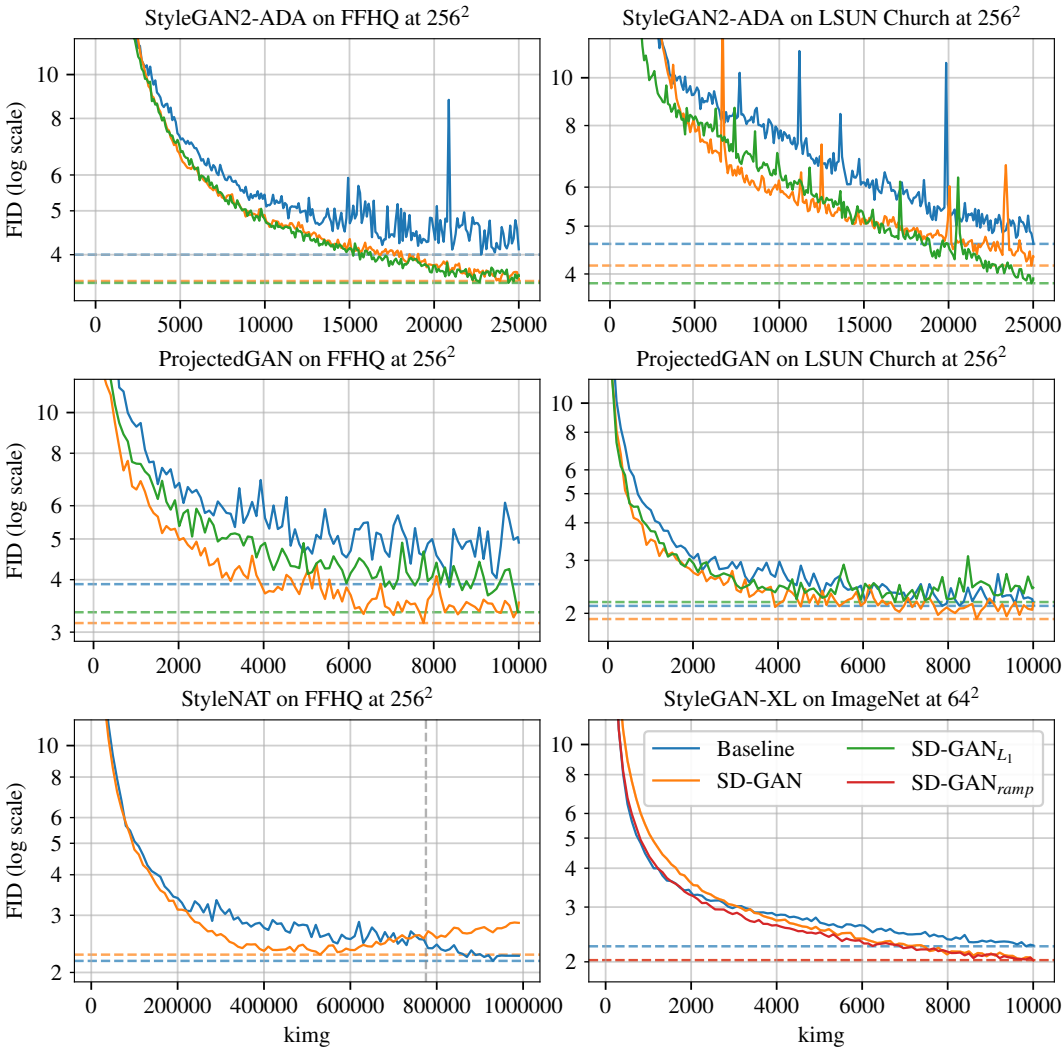


Figure 3: FID scores (log scale) during training on the FFHQ and LSUN Church datasets across three baseline architectures. Following [23], we allowed StyleNAT to learn much longer than the remaining architectures. Horizontal dashed lines indicate the best FID score.

loss weight (1 to 2) yields significant gains, reducing the FID score down to 2.74. While this grid search is limited in scope and density of sampling, the manifesting monotonic improvement of FID w.r.t. both considered hyperparameters signals a stable dependency that is likely to hold for other values of hyperparameters.

4.2 SD-GAN as a fine-tuning technique

Motivated by the significant computational overheads reported in Table 2, we attempt to engage SD-GAN as a fine-tuning technique. Given a pretrained GAN model, we initialize G, G_{EMA} and D with the pretrained weights and continue training using a combination of the standard adversarial loss ($\mathcal{L}_{adv}$) and our self-distillation loss ($\mathcal{L}_{SD}$). Within each comparison, the baseline and SD variants start from the same pretrained checkpoint and retain their native resolution. Figure 4 juxtaposes this with the baseline scenario, which continues training with $\mathcal{L}_{adv}$ only. In Fig. 4a we fine-tune the StyleGAN2-ADA[1] authors' official checkpoint of the model trained on FFHQ at 1024^2 resolution [17], and subject it to training on the same dataset. SD-GAN quickly improves on FID in this megapixel-scale fine-tuning run, while the baseline does not show clear signs of progress. In Fig. 4b

[1]Technically we use the StyleGAN2 checkpoints, as recommended in the official repository.

Table 1: Comparison of evaluation metrics (FID, rFID, Precision, and Recall) on FFHQ and LSUN Church at 256^2 and ImageNet at 64^2. Values in parentheses indicate the original reported scores from the respective papers (see Appendix B for explanations of the discrepancies).

Dataset	Model	FID ↓	rFID ↓	Precision ↑	Recall ↑
FFHQ (256^2)	StyleGAN2-ADA	4.00 (4.30)	1.58 ± 0.11	0.692	0.399
	StyleGAN2-ADA+SD	3.50	1.18 ± 0.27	0.678	0.440
	StyleGAN2-ADA+SD$_{L_1}$	3.46	4.51 ± 0.90	0.679	0.469
	ProjectedGAN	3.90 (3.39)	79.3 ± 7.6	0.662	0.439
	ProjectedGAN+SD	3.15	14.1 ± 2.1	0.666	0.452
	ProjectedGAN+SD$_{L_1}$	3.35	34.4 ± 3.6	0.669	0.446
	StyleNAT	2.17 (2.05)	6.31 ± 0.55	0.698	0.383
	StyleNAT+SD	2.27	4.22 ± 0.70	0.676	0.367
LSUN Church (256^2)	StyleGAN2-ADA	4.60	2.73 ± 0.18	0.626	0.218
	StyleGAN2-ADA+SD	4.16	1.81 ± 0.20	0.619	0.274
	StyleGAN2-ADA+SD$_{L_1}$	3.83	0.977 ± 0.075	0.622	0.321
	ProjectedGAN	2.11 (1.59)	5.7 ± 1.3	0.561	0.474
	ProjectedGAN+SD	1.91	1.92 ± 0.28	0.605	0.446
	ProjectedGAN+SD$_{L_1}$	2.18	2.95 ± 0.46	0.607	0.429
ImageNet (64^2)	StyleGAN-XL	2.24 (1.51)	1.11 ± 0.28	0.690	0.507
	StyleGAN-XL+SD	2.02	1.78 ± 0.95	0.689	0.510
	StyleGAN-XL+SD$_{ramp}$	2.02	1.54 ± 0.62	0.697	0.508

Table 2: Computational requirements of SD-GAN training compared to the baseline methods: the time (in seconds) required to process 1 kimg and the average memory consumption.

Method	StyleGAN2-ADA		ProjectedGAN		StyleNAT	
	s/kimg	Mem.	s/kimg	Mem.	s/kimg	Mem.
Baseline	12.56 s	3.10 GB	18.78 s	7.45 GB	213 s	29.7 GB
SD-GAN$_{L_1}$	14.09 s	3.36 GB	22.20 s	8.81 GB		
SD-GAN	19.93 s	5.00 GB	28.33 s	9.38 GB	233 s	31.8 GB
SD-GAN$_{L_1}$ overheads	+12.2%	+8.4%	+18.2%	+18.3%		
SD-GAN overheads	+58.7%	+61.3%	+50.9%	+25.9%	+9.4%	+7.1%

Table 3: FID scores for combinations of hyperparameters, ProjectedGAN+SD, FFHQ at 256^2.

α	$t_{1/2}$:	10	20	40
0.5		3.57	3.48	3.44
1.0		3.57	3.21	2.96
2.0		3.17	3.03	2.74

we examine a more realistic fine-tuning scenario, starting from the FFHQ at 256^2 checkpoint[1] and fine-tuning it on the train split of CelebA-HQ dataset at 256^2 [46, 11]. The change of training distribution causes all three models to start from moderate FID values, but they rapidly adapt. Both SD variants not only stabilize the fine-tuning process but also converge to a lower final FID.

4.3 Ablations

Figure 5 compares, for the ProjectedGAN applied to FFHQ at 128^2, the learning dynamics of the complete SD-GAN to its ablations obtained by disabling augmentations (SD-GAN *w/o. aug.*) or replacing the LPIPS-based $\mathcal{L}_{SD}$ with simpler distillation losses, which directly compare the images produced by G and G_{EMA}: L_1, L_2, and Multi-Scale Structural Similarity Index Measure (MS-SSIM,

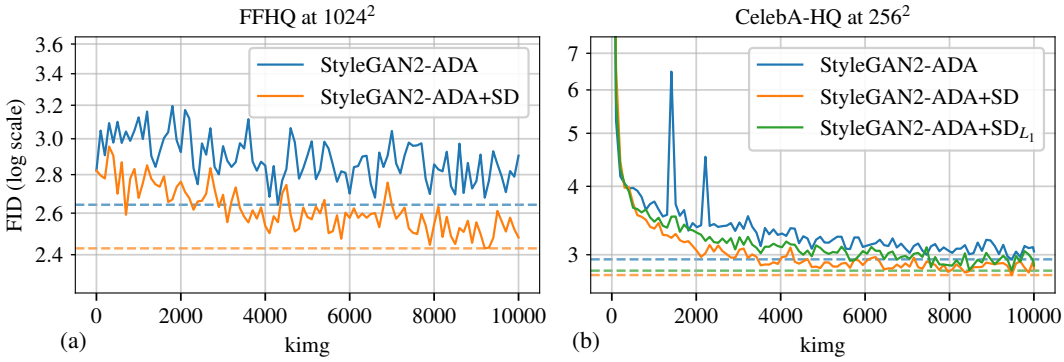


Figure 4: FID scores during fine-tuning on the FFHQ at 1024^2 and CelebA-HQ at 256^2.

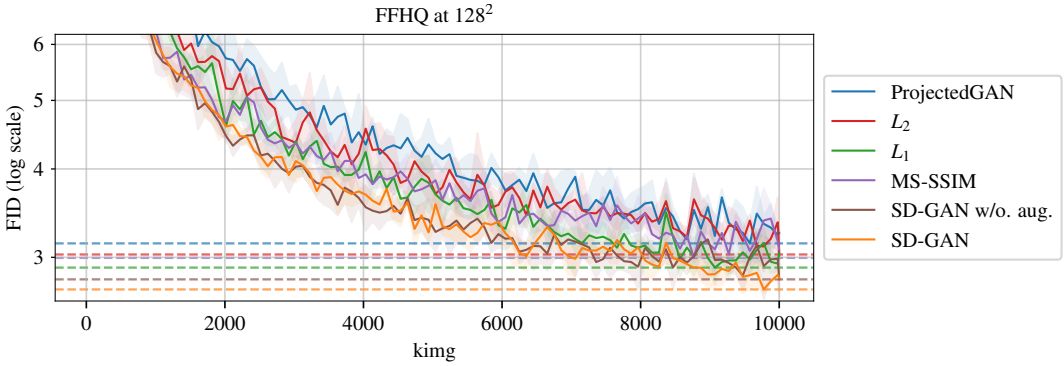


Figure 5: Learning curves for SD-GAN and its ablated variants.

[47]). Combining the latter variants with augmentations would be pointless, so we do not engage them. The figure presents the averages of three runs with shaded standard deviations. While the FID values still fluctuate quite significantly, the LPIPS variants quite clearly converge at the fastest pace, and the augmentations have slight positive impact. A separate comparison with delayed-checkpoint weight-space regularization is reported in Appendix D.

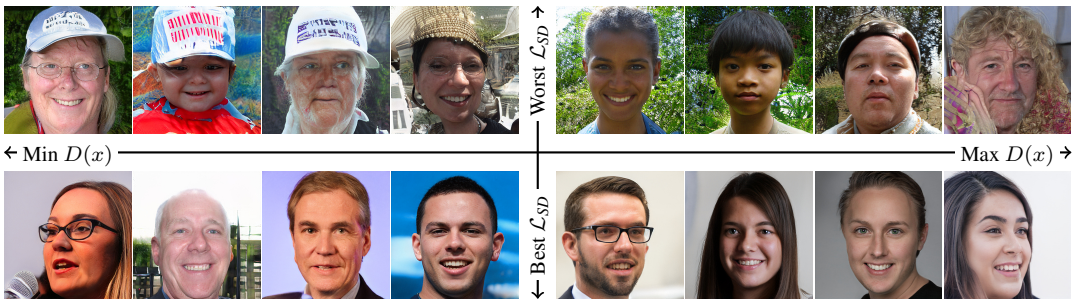
4.4 SD-GAN with discriminator-based self-distillation loss

By being pre-trained on a huge volume of images, LPIPS-based $\mathcal{L}_{SD}$ may suffer from "feature pollution/leakage". To address this, we consider a SD-GAN variant that calculates $\mathcal{L}_{SD}$ using the discriminator. To this aim, we query D on the outputs of G and G_{EMA} (Eq. 1), and extract the intermediate activations from the first three blocks of both queries. Following the LPIPS methodology, we apply channel normalization to these feature maps and compute their L_1 distance. We set $\alpha = 10$ to ensure similar contribution to $\mathcal{L}_G$ as in the LPIPS variant.

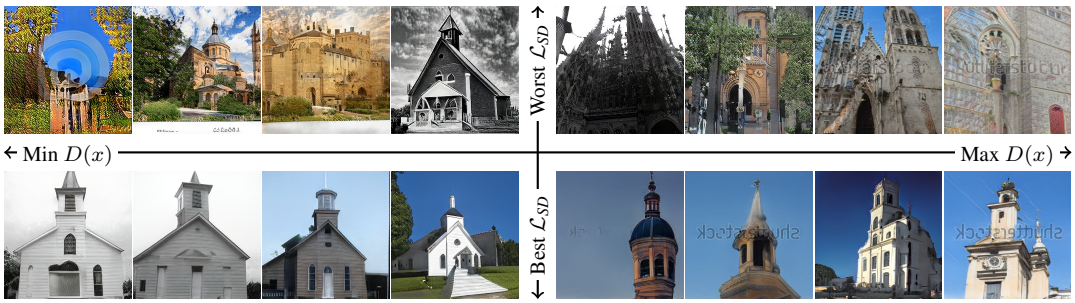
As shown in Table 4, using the discriminator's own feature space for self-distillation works remarkably well. The SD$_{disc}$ configuration achieves FID of 3.52, close to the LPIPS-based variant (3.50, Table 1), without an additional pretrained perceptual network. Its rFID (1.589) is essentially equal to the baseline (1.584) and higher than that of the LPIPS variant (1.181).

Table 4: Comparison of evaluation metrics (FID, rFID, Precision, and Recall) on FFHQ at 256^2.

Model	FID ↓	rFID ↓	Precision ↑	Recall ↑
StyleGAN2-ADA	4.00	1.58 ± 0.11	0.692	0.399
StyleGAN2-ADA+SD	3.50	1.18 ± 0.27	0.678	0.440
StyleGAN2-ADA+SD$_{disc}$	3.52	1.59 ± 0.25	0.682	0.446



(a) StyleGAN2-ADA+SD, FFHQ at 256^2.



(b) StyleGAN2-ADA+SD, LSUN Church at 256^2.

Figure 6: Qualitative interplay between the objectives. **Left vs. right:** smaller vs. larger *realness* according to D. **Top vs. bottom:** larger vs. smaller difference between G and G_{EMA} (smaller $\mathcal{L}_{SD}$). The adversarial objective pulls generator to the right, while the self-distillation objective pulls it to the bottom. See Fig. 9 for per-image values of $\mathcal{L}_{SD}$ and D.

4.5 The interplay between loss components

To better understand the interplay between the adversarial and self-distillation objectives, we analyze the visual characteristics of the images favored by each loss component. In Fig. 6, we juxtapose selected images obtained by querying the final G_{EMA} models on a sample of 10k latent vectors. For each latent, we generate images using both G_{EMA} and the active generator G, compute the LPIPS distance $\mathcal{L}_{SD}$ between them, and select the 200 images with the largest and smallest $\mathcal{L}_{SD}$ (respectively top and bottom row in the figure). Then we query the discriminator on the G_{EMA} outputs and select the 4 highest and lowest scoring images, placing them respectively in the right and left quadrants. The figure presents thus the extremes of the joint distribution of $\mathcal{L}_{SD}$ (vertical) and $\mathcal{L}_{adv}$ (horizontal), rather than a representative sample of the generated distribution, and illustrates the distinct image characteristics associated with the two scores. Specifically, the images with worse $\mathcal{L}_{SD}$ (top rows) exhibit high-frequency variations, complex backgrounds, and structural artifacts. Minimizing the self-distillation distance (bottom rows) yields remarkably cleaner compositions. Meanwhile, the discriminator (right columns) enforces photorealistic textures and high fidelity, even on detailed images, and its score correlates with image quality, as noted in [48, 49].

4.6 Evolution of generated samples

To quantify the stabilizing effect of the self-distillation objective, we measure the perceptual distance of the StyleGAN2-ADA generators across different training checkpoints (from 20% to 100% kimgs) using fixed latent codes. This metric functions similarly to the Perceptual Path Length [15] but is measured across the training trajectory, rather than through latent space interpolation. Table 5 reports the average perceptual distance and its standard deviation of the generated samples across these checkpoints, calculated using a sample size of 480. SD-GAN significantly reduces the perceptual variance between samples compared to the baselines.

To illustrate the visual effect of the above stabilization, in Fig. 7 we present the evolution of generated images for a fixed latent vector z throughout the training process, for the sequences whose average



Figure 7: Evolution of generated images for a fixed latent vector z during training. Top two rows: StyleGAN2-ADA; bottom two rows: StyleGAN2-ADA+SD. See Figure 10 for more.

perceptual distance is closest to the mean value reported in Table 5. The images generated by the baseline model exhibit very high variance throughout the run; the facial features and church architecture change drastically. In contrast, the face generated by SD-GAN for z remains relatively stable, and could likely be identified as the same person in all shown images. Similarly, the overall structure of the church is quite consistent.

4.7 Limitations

We identify the following limitations: (i) The LPIPS-based $\mathcal{L}_{SD}$ may entrench ImageNet-induced biases and lead to "feature leakage". Though the results on the rFID metric in Table 1 partially alleviate this concern, an even better mitigation of this bias is the choice of non-perceptual losses, which do not seem to lag far behind (as shown in ablations in Fig. 5), or a discriminator-based $\mathcal{L}_{SD}$ (Sec. 4.4). (ii) Our empirical evaluation covers four representative GAN architectures and three datasets; broader validation across additional GAN families, datasets, and resolutions remains future work. The proof in Sec. 3 is limited to the continuous-time, linearized Dirac-GAN with quadratic SD loss, and as such does not establish convergence for high-dimensional nonlinear GANs with stochastic optimization, perceptual losses, or differentiable augmentations. (iii) Finally, SD-GAN imposes relatively high computational overhead on training, especially when using perceptual $\mathcal{L}_{SD}$. Again, turning to non-perceptual SD losses may alleviate this problem.

5 Conclusions

While nowadays it is diffusion models, rather than GANs, that are the primary tools for image generation, they are often computationally very demanding during inference. Distilling diffusion models into GANs has thus become a common technique to address this deficiency [33, 34, 35]. This makes SD-GAN a potentially useful tool for aiding this process, e.g. fine-tuning the GAN models distilled from diffusion models.

Table 5: Average variation of generated samples across training checkpoints.

	FFHQ		LSUN Church	
Model	Baseline	SD-GAN	Baseline	SD-GAN
StyleGAN2-ADA	0.375 ± 0.020	0.295 ± 0.028	0.482 ± 0.016	0.358 ± 0.033
ProjectedGAN	0.470 ± 0.028	0.411 ± 0.022	0.513 ± 0.030	0.472 ± 0.029

Acknowledgments and Disclosure of Funding

This research was funded by the Polish Ministry of Science and Higher Education, grant no. 0311/SBAD/0770, and by the National Science Centre, Poland, grant no. 2025/57/B/ST6/03737. For the purpose of Open Access, the authors have applied a CC BY public copyright license to any Author Accepted Manuscript (AAM) version arising from this submission.

References

[1] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In: *Proceedings of the 28th International Conference on Neural Information Processing Systems*. Vol. 2. Cambridge, MA, USA: MIT Press, 2014, pp. 2672–2680.

[2] Lars M. Mescheder, Andreas Geiger, and Sebastian Nowozin. Which Training Methods for GANs do actually Converge? In: *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*. Ed. by Jennifer G. Dy and Andreas Krause. Proceedings of Machine Learning Research. PMLR, 2018, pp. 3478–3487.

[3] Pavel Izmailov, Dmitrii Podoprikhin, Timur Garipov, Dmitry P. Vetrov, and Andrew Gordon Wilson. Averaging Weights Leads to Wider Optima and Better Generalization. In: *Proceedings of the Thirty-Fourth Conference on Uncertainty in Artificial Intelligence, UAI 2018, Monterey, California, USA, August 6-10, 2018*. Ed. by Amir Globerson and Ricardo Silva. AUAI Press, 2018, pp. 876–885.

[4] Antti Tarvainen and Harri Valpola. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In: *Proceedings of the 31st International Conference on Neural Information Processing Systems*. Red Hook, NY, USA: Curran Associates Inc., 2017, pp. 1195–1204. ISBN: 978-1-5108-6096-4.

[5] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre H. Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Daniel Guo, Mohammad Gheshlaghi Azar, Bilal Piot, Koray Kavukcuoglu, Rémi Munos, and Michal Valko. Bootstrap your own latent a new approach to self-supervised learning. In: *Proceedings of the 34th International Conference on Neural Information Processing Systems*. NIPS '20. Red Hook, NY, USA: Curran Associates Inc., 2020, pp. 21271–21284. ISBN: 978-1-71382-954-6.

[6] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging Properties in Self-Supervised Vision Transformers. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). 2021, pp. 9650–9660.

[7] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy V. Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Mido Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Hervé Jégou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. DINOv2: Learning Robust Visual Features without Supervision. In: *Trans. Mach. Learn. Res.* 2024 (2024).

[8] Oriane Siméoni, Huy V. Vo, Maximilian Seitzer, Federico Baldassarre, Maxime Oquab, Cijo Jose, Vasil Khalidov, Marc Szafraniec, Seung Eun Yi, Michaël Ramamonjisoa, Francisco Massa, Daniel Haziza, Luca Wehrstedt, Jianyuan Wang, Timothée Darcet, Théo Moutakanni, Leonel Sentana, Claire Roberts, Andrea Vedaldi, Jamie Tolan, John Brandt, Camille Couprie, Julien Mairal, Hervé Jégou, Patrick Labatut, and Piotr Bojanowski. DINOv3. In: *CoRR* abs/2508.10104 (2025). DOI: 10.48550/ARXIV.2508.10104. arXiv: 2508.10104.

[9] Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The Unreasonable Effectiveness of Deep Features as a Perceptual Metric. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2018, pp. 586–595.

[10] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training GANs. In: *Proceedings of the 30th International Conference on Neural Information Processing Systems*. Red Hook, NY, USA: Curran Associates Inc., 2016, pp. 2234–2242. ISBN: 978-1-5108-3881-9.

[11] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive Growing of GANs for Improved Quality, Stability, and Variation. In: *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018.

[12] Yasin Yazici, Chuan-Sheng Foo, Stefan Winkler, Kim-Hui Yap, Georgios Piliouras, and Vijay Chandrasekhar. The Unusual Effectiveness of Averaging in GAN Training. In: *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019.

[13] Daniel Morales-Brotons, Thijs Vogels, and Hadrien Hendrikx. Exponential Moving Average of Weights in Deep Learning: Dynamics and Benefits. In: *Trans. Mach. Learn. Res.* 2024 (2024).

[14] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large Scale GAN Training for High Fidelity Natural Image Synthesis. In: *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019.

[15] Tero Karras, Samuli Laine, and Timo Aila. A Style-Based Generator Architecture for Generative Adversarial Networks. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019, pp. 4401–4410.

[16] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and Improving the Image Quality of StyleGAN. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020, pp. 8110–8119.

[17] Tero Karras, Miika Aittala, Janne Hellsten, Samuli Laine, Jaakko Lehtinen, and Timo Aila. Training generative adversarial networks with limited data. In: *Proceedings of the 34th International Conference on Neural Information Processing Systems*. Red Hook, NY, USA: Curran Associates Inc., 2020, pp. 12104–12114. ISBN: 978-1-71382-954-6.

[18] Tero Karras, Miika Aittala, Samuli Laine, Erik Härkönen, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Alias-free generative adversarial networks. In: *Proceedings of the 35th International Conference on Neural Information Processing Systems*. NIPS '21. Red Hook, NY, USA: Curran Associates Inc., 2021, pp. 852–863. ISBN: 978-1-71384-539-3.

[19] Axel Sauer, Kashyap Chitta, Jens Müller, and Andreas Geiger. Projected GANs converge faster. In: *Proceedings of the 35th International Conference on Neural Information Processing Systems*. Red Hook, NY, USA: Curran Associates Inc., 2021, pp. 17480–17492. ISBN: 978-1-71384-539-3.

[20] Axel Sauer, Katja Schwarz, and Andreas Geiger. StyleGAN-XL: Scaling StyleGAN to Large Diverse Datasets. In: *ACM SIGGRAPH 2022 Conference Proceedings*. SIGGRAPH '22. New York, NY, USA: Association for Computing Machinery, 2022, pp. 1–10. ISBN: 978-1-4503-9337-9. DOI: 10.1145/3528233.3530738.

[21] 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: *Proceedings of the 40th International Conference on Machine Learning*. Vol. 202. ICML'23. Honolulu, Hawaii, USA: JMLR.org, 2023, pp. 30105–30118.

[22] Yuhta Takida, Masaaki Imaizumi, Takashi Shibuya, Chieh-Hsin Lai, Toshimitsu Uesaka, Naoki Murata, and Yuki Mitsufuji. SAN: Inducing Metrizability of GAN with Discriminative Normalized Linear Layer. In: *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024.

[23] Steven Walton, Ali Hassani, Xingqian Xu, Zhangyang Wang, and Humphrey Shi. Efficient Image Generation with Variadic Attention Heads. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*. 2025, pp. 3264–3275.

[24] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In: *Proceedings of the 34th International Conference on Neural Information Processing Systems*. NIPS '20. Red Hook, NY, USA: Curran Associates Inc., 2020, pp. 6840–6851. ISBN: 978-1-71382-954-6.

[25] Yang Song, Jascha Sohl-Dickstein, Diederik P. Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-Based Generative Modeling through Stochastic Differential Equations. In: *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021.

[26] Prafulla Dhariwal and Alex Nichol. Diffusion models beat GANs on image synthesis. In: *Proceedings of the 35th International Conference on Neural Information Processing Systems*. NIPS '21. Red Hook, NY, USA: Curran Associates Inc., 2021, pp. 8780–8794. ISBN: 978-1-71384-539-3.

[27] Han Zhang, Zizhao Zhang, Augustus Odena, and Honglak Lee. Consistency Regularization for Generative Adversarial Networks. In: *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020.

[28] Long Zhao, Zizhao Zhang, Ting Chen, Dimitris N. Metaxas, and Han Zhang. Improved transformer for high-resolution GANs. In: *Proceedings of the 35th International Conference on Neural Information Processing Systems*. NIPS '21. Red Hook, NY, USA: Curran Associates Inc., 2021, pp. 18367–18380. ISBN: 978-1-7138-4539-3.

[29] Shengyu Zhao, Zhijian Liu, Ji Lin, Jun-Yan Zhu, and Song Han. Differentiable augmentation for data-efficient GAN training. In: *Proceedings of the 34th International Conference on Neural Information Processing Systems*. NIPS '20. Red Hook, NY, USA: Curran Associates Inc., 2020, pp. 7559–7570. ISBN: 978-1-7138-2954-6.

[30] Hung-Yu Tseng, Lu Jiang, Ce Liu, Ming-Hsuan Yang, and Weilong Yang. Regularizing Generative Adversarial Networks Under Limited Data. In: *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, virtual, June 19-25, 2021*. Computer Vision Foundation / IEEE, 2021, pp. 7921–7931. DOI: 10.1109/CVPR46437.2021.00783.

[31] Shaojie Li, Jie Wu, Xuefeng Xiao, Fei Chao, Xudong Mao, and Rongrong Ji. Revisiting discriminator in GAN compression: a generator-discriminator cooperative compression scheme. In: *Proceedings of the 35th International Conference on Neural Information Processing Systems*. NIPS '21. Red Hook, NY, USA: Curran Associates Inc., 2021, pp. 28560–28572. ISBN: 978-1-7138-4539-3.

[32] Yuxi Ren, Jie Wu, Xuefeng Xiao, and Jianchao Yang. Online Multi-Granularity Distillation for GAN Compression. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. 2021, pp. 6793–6803.

[33] Axel Sauer, Dominik Lorenz, Andreas Blattmann, and Robin Rombach. Adversarial Diffusion Distillation. In: *Computer Vision – ECCV 2024: 18th European Conference, Milan, Italy, September 29–October 4, 2024, Proceedings, Part LXXXVI*. Berlin, Heidelberg: Springer-Verlag, 2024, pp. 87–103. ISBN: 978-3-031-73015-3. DOI: 10.1007/978-3-031-73016-0_6.

[34] Axel Sauer, Frederic Boesel, Tim Dockhorn, Andreas Blattmann, Patrick Esser, and Robin Rombach. Fast High-Resolution Image Synthesis with Latent Adversarial Diffusion Distillation. In: *SIGGRAPH Asia 2024 Conference Papers, SA 2024, Tokyo, Japan, December 3-6, 2024*. Ed. by Takeo Igarashi, Ariel Shamir, and Hao (Richard) Zhang. ACM, 2024, 106:1–106:11. DOI: 10.1145/3680528.3687625.

[35] Minguk Kang, Richard Zhang, Connelly Barnes, Sylvain Paris, Suha Kwak, Jaesik Park, Eli Shechtman, Jun-Yan Zhu, and Taesung Park. Distilling Diffusion Models Into Conditional GANs. In: *Computer Vision – ECCV 2024: 18th European Conference, Milan, Italy, September 29–October 4, 2024, Proceedings, Part XXVIII*. Berlin, Heidelberg: Springer-Verlag, 2024, pp. 428–447. ISBN: 978-3-031-73389-5. DOI: 10.1007/978-3-031-73390-1_25.

[36] Yuanfeng Zheng, Yuchen Yan, and Hao Jiang. Semi-TSGAN: Semi-Supervised Learning for Highlight Removal Based on Teacher-Student Generative Adversarial Network. In: *Sensors* 24.10 (Jan. 2024), p. 3090. ISSN: 1424-8220. DOI: 10.3390/s24103090.

[37] Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency models. In: *Proceedings of the 40th International Conference on Machine Learning*. Vol. 202. ICML'23. Honolulu, Hawaii, USA: JMLR.org, 2023, pp. 32211–32252.

[38] Ron Mokady, Omer Tov, Michal Yarom, Oran Lang, Inbar Mosseri, Tali Dekel, Daniel Cohen-Or, and Michal Irani. Self-Distilled StyleGAN: Towards Generation from Internet Photos. In: *ACM SIGGRAPH 2022 Conference Proceedings*. SIGGRAPH '22. New York, NY, USA: Association for Computing Machinery, 2022, pp. 1–9. ISBN: 978-1-4503-9337-9. DOI: 10.1145/3528233.3530708.

[39] Hassan K. Khalil. *Nonlinear Systems*. 2nd. Upper Saddle River, NJ: Prentice-Hall, 1996. ISBN: 978-0-13-228024-2.

[40] Lars Mescheder, Sebastian Nowozin, and Andreas Geiger. The numerics of GANs. In: *Proceedings of the 31st International Conference on Neural Information Processing Systems*. NIPS'17. Red Hook, NY, USA: Curran Associates Inc., 2017, pp. 1823–1833. ISBN: 978-1-5108-6096-4.

[41] Vaishnavh Nagarajan and J. Zico Kolter. Gradient descent GAN optimization is locally stable. In: *Proceedings of the 31st International Conference on Neural Information Processing Systems*. NIPS'17. Red Hook, NY, USA: Curran Associates Inc., 2017, pp. 5591–5600. ISBN: 978-1-5108-6096-4.

[42] Fisher Yu, Ari Seff, Yinda Zhang, Shuran Song, Thomas Funkhouser, and Jianxiong Xiao. *LSUN: Construction of a Large-scale Image Dataset using Deep Learning with Humans in the Loop*. June 4, 2016. DOI: 10.48550/arXiv.1506.03365. arXiv: 1506.03365[cs].

[43] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. GANs trained by a two time-scale update rule converge to a local nash equilibrium. In: *Proceedings of the 31st International Conference on Neural Information Processing Systems*. NIPS'17. Red Hook, NY, USA: Curran Associates Inc., 2017, pp. 6629–6640. ISBN: 978-1-5108-6096-4.

[44] George Stein, Jesse C. Cresswell, Rasa Hosseinzadeh, Yi Sui, Brendan Leigh Ross, Valentin Villecroze, Zhaoyan Liu, Anthony L. Caterini, J. Eric T. Taylor, and Gabriel Loaiza-Ganem. Exposing flaws of generative model evaluation metrics and their unfair treatment of diffusion models. In: *Proceedings of the 37th International Conference on Neural Information Processing Systems*. NIPS '23. Red Hook, NY, USA: Curran Associates Inc., 2023, pp. 3732–3784.

[45] Mehdi S. M. Sajjadi, Olivier Bachem, Mario Lucic, Olivier Bousquet, and Sylvain Gelly. Assessing generative models via precision and recall. In: *Proceedings of the 32nd International Conference on Neural Information Processing Systems*. NIPS'18. Red Hook, NY, USA: Curran Associates Inc., 2018, pp. 5234–5243.

[46] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep Learning Face Attributes in the Wild. In: *Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV)*. ICCV '15. USA: IEEE Computer Society, 2015, pp. 3730–3738. ISBN: 978-1-4673-8391-2. DOI: 10.1109/ICCV.2015.425.

[47] Z. Wang, E.P. Simoncelli, and A.C. Bovik. Multiscale structural similarity for image quality assessment. In: *The Thrity-Seventh Asilomar Conference on Signals, Systems & Computers, 2003*. Vol. 2. 2003, 1398–1402 Vol.2. DOI: 10.1109/ACSSC.2003.1292216.

[48] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In: *Proceedings of the 34th International Conference on Machine Learning - Volume 70*. ICML'17. Sydney, NSW, Australia: JMLR.org, 2017, pp. 214–223.

[49] Samaneh Azadi, Catherine Olsson, Trevor Darrell, Ian J. Goodfellow, and Augustus Odena. Discriminator Rejection Sampling. In: *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019.

[50] Elad Hoffer, Itay Hubara, and Daniel Soudry. Train longer, generalize better: closing the generalization gap in large batch training of neural networks. In: *Proceedings of the 31st International Conference on Neural Information Processing Systems*. NIPS'17. Red Hook, NY, USA: Curran Associates Inc., 2017, pp. 1729–1739. ISBN: 978-1-5108-6096-4.

A Software implementation and deployment

The SD-GAN variants used in this study have been implemented by minimally modifying the publicly available 'official' implementations of the reference methods:

- StyleGAN2-ADA: https://github.com/nvlabs/stylegan2-ada-pytorch
- ProjectedGAN: https://github.com/autonomousvision/projected-gan
- StyleNAT: https://github.com/SHI-Labs/StyleNAT
- StyleGAN-XL: https://github.com/autonomousvision/stylegan-xl

The infrastructure consisted of 1 Nvidia H100 GPU, 8 CPU cores, and 32 GB RAM for the single GPU setup and 4 $\times$ Nvidia H100 GPUs, 32 CPU cores and 128 GB RAM for 4 GPU training runs. The total estimated compute we used for all experiments is approximately 10k GPU-hours. The full specification of the nodes we used is:

- CPU: AMD EPYC 9334, 2 $\times$ 32 cores
- RAM: 768 GB
- GPU: 4$\times$ Nvidia H100-94 SXM5, 94GB HBM2e

The implementation of our proposed method, including the training scripts and evaluation routines, is publicly available at https://github.com/antollo/sd-gan. Additionally, a supplementary project page featuring animated visualizations of our generated results is available at https://antollo.github.io/sd-gan.

Typical training runs of SD-GAN using 4 GPUs on the 256^2 datasets took respectively 46 h, 20 h, and 14 days for StyleGAN2-ADA+SD, ProjectedGAN+SD, and StyleNAT+SD. In comparison, their respective non-SD baselines took 41 h, 16 h, and 14 days. The baseline runs exhibited runtimes that are close to the SD-GAN runs; this is because, during the baseline experiments, we still explicitly computed the difference between G and G_{EMA} (i.e., the $\mathcal{L}_{SD}$ value) solely for tracking and monitoring purposes, without using it to update the network.

B Challenges in replication of prior results

Due to the hardware constraints of the Nvidia H100 GPUs (Hopper architecture, `sm_90`) utilized in our experiments, we were required to upgrade PyTorch to ensure compatibility for both StyleGAN2-ADA and ProjectedGAN (compared to the versions used in the original implementations). Specifically, our environment used `torch==2.1.2` compiled with CUDA 12.1.

After several unsuccessful attempts to match the baseline results for StyleNAT on LSUN Church, we excluded this combination from our study. We contacted the authors via a GitHub issue[2], and while they were very helpful and provided feedback, we ultimately failed to achieve outcomes consistent with the reported benchmarks. It is worth noting that the Church configuration file is missing from their repository, leading us to believe that an omitted or forgotten technical detail may be responsible for the discrepancy.

We also encountered replication challenges with ProjectedGAN. The original work utilized an 8-GPU training setup. In the official implementation in a distributed multi-device setting, Batch Normalization layers compute statistics separately on each device's mini-batch fragment. This inadvertently applies the concept of *Ghost Batch Normalization* [50], which is known to aid generalization and yield better results than computing statistics over the entire batch. To simulate this 8-GPU dynamic on our 4-GPU setup, we explicitly implemented Ghost BN. However, even with this correction, the best baseline FID we achieved on LSUN Church was 2.11, falling short of the 1.59 reported in the original paper. Furthermore, evaluating the authors' officially provided model (weights) across various repository commits also failed to reproduce the reported FID. This replication issue appears to be a known problem, mentioned in other independent studies [44] and GitHub issues[3].

[2] https://github.com/SHI-Labs/StyleNAT/issues/18
[3] https://github.com/autonomousvision/projected-gan/issues/103

Regarding StyleGAN-XL, we attempted to replicate the results on ImageNet at 64^2 resolution. The original study did not use a fixed training schedule; rather than that, the authors noted that each progressive growth stage was trained until the FID stopped decreasing [20, Sec. 3.2]. They reported a final FID of 1.51 at this resolution. In our experiments, we initialized the baseline and SD variant from the same pretrained 32^2 stem and trained each of them for $10k$ kimgs at 64^2. The baseline run required approximately 1.3 days on H100 GPU and achieved FID of 2.24. While the original work did not provide an exact training duration for the 64^2 stage, the authors noted that the subsequent 128^2 resolution required 15 V100-days, which may suggest that their training budget was significantly larger than ours.

Beyond the specific cases of StyleNAT and ProjectedGAN, we encountered similar replication failures with several other open-source repositories. While we optimistically assume that these discrepancies stem from our own experimental misconfigurations, these repeated challenges serve as a reminder that reproducibility remains a significant issue.

C Pseudocode

Listing 1 presents the pseudocode of SD-GAN, as implemented in PyTorch.

```
loss_type = 'LPIPS' # or 'L1'
use_aug = True # or False
aug2 = AugmentationSequential(
  RandomHorizontalFlip(p=0.5),
  RandomRotate(degrees=5, p=0.5),
  RandomTranslate(translate=(0.01, 0.01), p=0.5),
  random_apply=2
)

# Inside the training loop:
z = torch.randn([batch_size, z_dim])
fake_images = G(z)
fake_score = D(fake_images)
# Normal adversarial loss, e.g.
adv_loss = -torch.mean(fake_score)
G_loss = adv_loss.mean()
# Self-distillation loss
with torch.no_grad():
    fake_images_ema = G_ema(z)
if use_aug:
    fake_images, fake_images_ema = aug2(fake_images, fake_images_ema)
if loss_type == 'L1':
    SD_loss = l1_loss(fake_images, fake_images_ema)
elif loss_type == 'LPIPS':
    SD_loss = lpips(fake_images, fake_images_ema, net='vgg')
G_loss = G_loss + alpha * SD_loss.mean()
G_loss.backward()
# Continue the training loop
```

Listing 1: PyTorch pseudocode for the SD-GAN generator update step. The highlighted section demonstrates the calculation of the self-distillation loss.

D Comparison with alternative temporal regularization baselines

To isolate the performance gains of SD-GAN that originate specifically from our combination of output-space consistency and EMA, we compared it against a simpler temporal regularization baseline: a delayed generator checkpoint with weight-space regularization. Specifically, we maintained a queue of $N = 10$ historical model weights, updated every T/N kimgs (setting $T = 20$ to match the EMA half-life from our main experiments; see Table 1). We then applied an L_2 penalty between the weights

Table 6: Comparison of the final FID scores on FFHQ at 256^2 achieved by the baseline ProjectedGAN, the variant with the weight-space temporal regularization for different penalty weights λ, and SD-GAN.

Model		FID ↓
ProjectedGAN		3.90
ProjectedGAN (w/ t. reg.)	$\lambda = 10^5$	4.70
	$\lambda = 10^4$	4.24
	$\lambda = 10^3$	3.85
	$\lambda = 10^2$	3.96
	$\lambda = 10$	3.96
	$\lambda = 1$	4.20
ProjectedGAN+SD		3.15

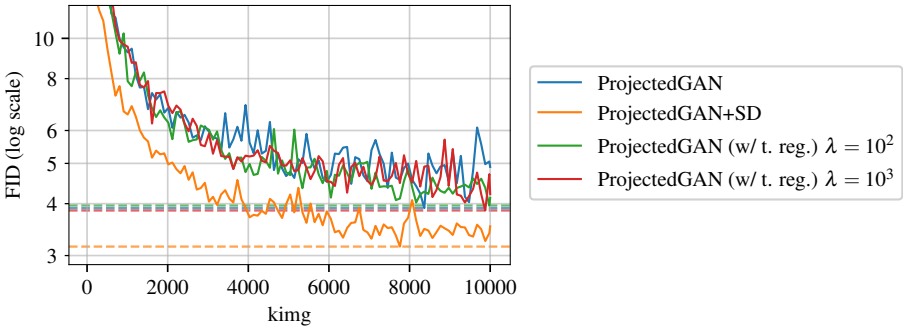


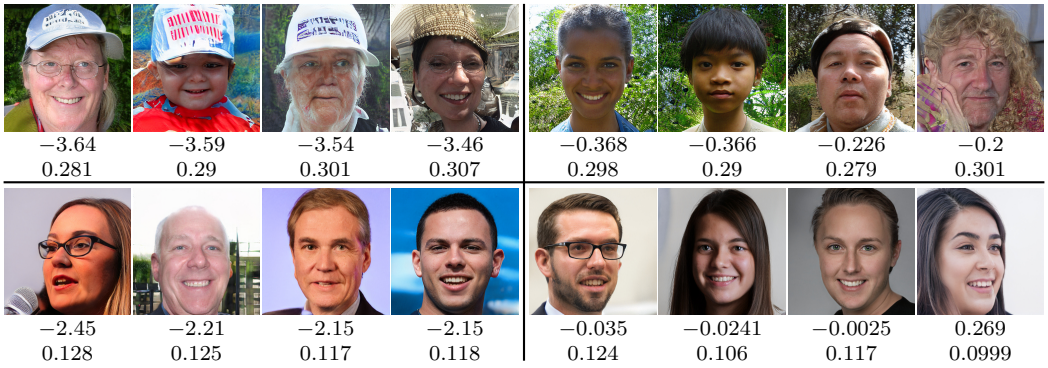
Figure 8: FID scores (log scale) during training on the FFHQ dataset at 256^2 comparing the baseline ProjectedGAN, SD-GAN, and the best weight-space temporal regularization variants. Horizontal dashed lines indicate the best FID score achieved by the respective models.

of the active generator and the oldest historical weights from the queue. Table 6 reports the FID scores for ProjectedGAN on FFHQ at 256^2 across various penalty weights λ.

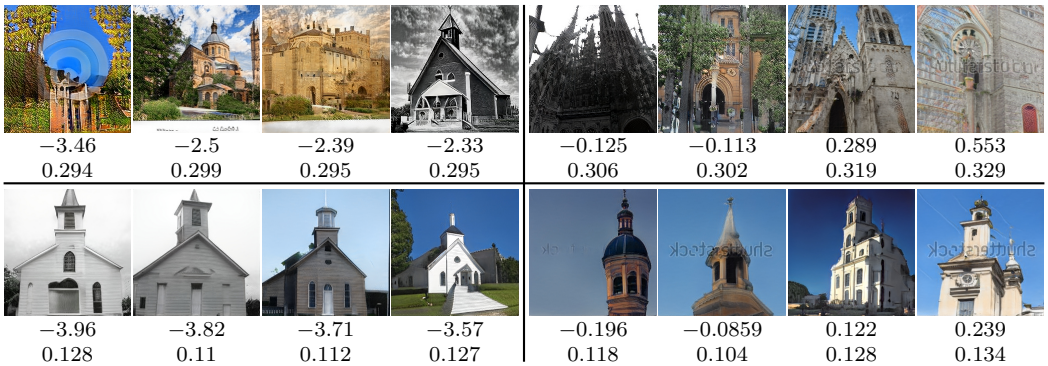
While this temporal smoothing approach yields a negligible improvement over the unregularized baseline at one of the tested penalty weights ($\lambda = 10^3$), yielding an FID of 3.85, it falls significantly short of SD-GAN (FID 3.15). Furthermore, the learning curves in Figure 8 demonstrate that while temporal regularization dampens variance compared to the baseline, it fails to lead to deeper convergence achieved by self-distillation. This confirms that simple weight-space temporal anchoring does not replicate the gains of our method, underscoring the importance of the output-space consistency combined with EMA.

E Auxiliary results

Figure 9 presents the detailed version of Figure 6, with additional information on both objectives. The evolution of images generated during training for a fixed latent vector z is shown in Figure 10. Figure 11 provides a qualitative comparison of the baseline and SD-GAN after fine-tuning on the FFHQ dataset at 1024^2 resolution.



(a) StyleGAN2-ADA+SD, FFHQ at 256^2.



(b) StyleGAN2-ADA+SD, LSUN Church at 256^2.

Figure 9: Qualitative interplay between the objectives (detailed version of Fig. 6). **Left vs. right:** smaller vs. larger *realness* according to D (D logits, top value). **Top vs. bottom:** larger vs. smaller difference between G and G_{EMA} ($\mathcal{L}_{SD}$, bottom value under image). The adversarial objective pulls generator to the right, while the self-distillation objective pulls it to the bottom. Best viewed when zoomed in.



(a) StyleGAN2-ADA (FFHQ)



(b) StyleGAN2-ADA+SD (FFHQ)



(c) StyleGAN2-ADA (LSUN Church)

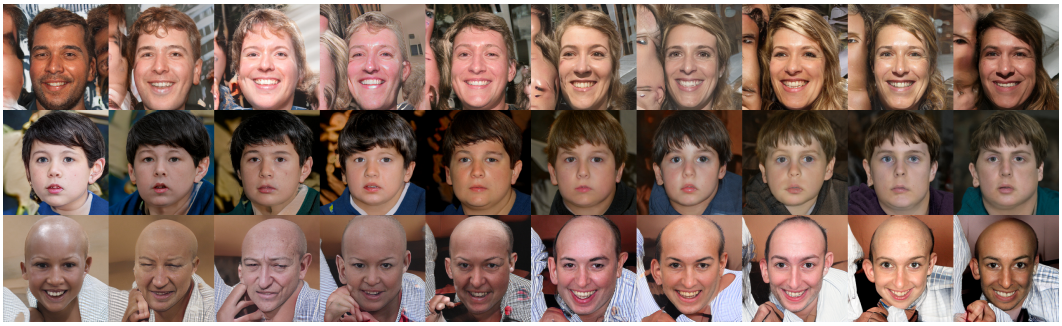


(d) StyleGAN2-ADA+SD (LSUN Church)

Figure 10: Evolution of images generated during training for a fixed latent vector z. *(Continued on next page)*



(e) ProjectedGAN (FFHQ)



(f) ProjectedGAN+SD (FFHQ)



(g) ProjectedGAN (LSUN Church)



(h) ProjectedGAN+SD (LSUN Church)

Figure 10: Evolution of images generated during training for a fixed latent vector z. *(Continued on next page)*



(i) StyleGAN-XL (ImageNet at 64^2, classes: 5 electric ray, 8 hen, 23 vulture, and 29 axolotl)



(j) StyleGAN-XL+SD (ImageNet at 64^2, classes: 5 electric ray, 8 hen, 23 vulture, and 29 axolotl)

Figure 10: Evolution of images generated during training for a fixed latent vector z. The progressions are grouped by model and dataset (detailed version of Figure 7).



(a) StyleGAN2-ADA (FFHQ 1024^2 fine-tuning)



(b) StyleGAN2-ADA+SD (FFHQ 1024^2 fine-tuning)

Figure 11: Qualitative results after fine-tuning on FFHQ at 1024^2. Samples generated using the same latent vectors for both methods are shown for comparison.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The claims made in the abstract and introduction are supported by theoretical and experimental results presented in the paper.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The limitations are discussed in a separate section. This study does not raise significant risks concerning privacy or fairness.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [Yes]

Justification: The theoretical analysis and the proof of convergence in Sec. 3 are to our best knowledge correct (double-checked) and accompanied with relevant assumptions.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: The method is simple and easily reproducible even without using our codebase. All the relevant parameters are provided in the paper, the appendices, or in the referenced prior works.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: The code, model weights, and instructions for reproducibility are publicly available at https://github.com/antollo/sd-gan.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (https://neurips.cc/public/guides/CodeSubmissionPolicy) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (https://neurips.cc/public/guides/CodeSubmissionPolicy) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: All the training and test details are provided in the main paper, the appendices, or in the cited works and software libraries (which we also point the readers to). Any discrepancies w.r.t. cited studies are explained in the main text and appendices.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: The high computational cost of training of the proposed architectures prevented us from conducting multiple runs. Nevertheless, the advantage of our method seems significant on several metrics (FID, rFID, precision, recall).

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: These details are provided at the Appendix A.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The paper conforms with NeurIPS Code of Ethics. We engage only well-known benchmarks (FFHQ, LSUN Churches) that do not raise ethical concerns. No new data has been collected. We do not anticipate this study to have negative societal impact nor harmful consequences.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [N/A]

Justification: We do not anticipate this study to have negative societal impact nor harmful consequences. This study is ethically neutral – at least to the extent in which previous studies on GANs were neutral.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

Answer: [N/A]

Justification: The method proposed in the paper does not pose significant risks that would require guardrails. No LLMs are engaged in this work.

Guidelines:

- The answer [N/A] means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: All significant software packages and repositories have been properly credited in the main text or in the appendices.

Guidelines:

- The answer [N/A] means that the paper does not use existing assets.

- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: The paper does not introduce or release any new datasets or assets. Our proposed method is evaluated on existing, publicly available benchmarks.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: The core method development in this research does not involve LLMs as any important, original, or non-standard components.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.