NATIONAL RESEARCH UNIVERSITY
«HIGHER SCHOOL OF ECONOMICS»

Faculty of Computer Science

Использование экспоненциально сглаженных моделей для обучения Генеративных Состязательных Сетей

Exponential Moving Average Teachers for Generative Adversarial Networks

Qualification paper – Master of Science Dissertation

Field of study 01.04.02 «Applied Mathematics and Informatics»

Program: «Data Science»

Student

Sergey Chervontsev

Supervisor

Artem Babenko

Yandex

Moscow, June 2021

Abstract

Adversarial training is currently one of the leading paradigms in large-scale generative modelling. For problems like image generation, it corresponds to the family of Generative Adversarial Networks (GANs) that jointly train two networks — generator and discriminator — to produce artificial images from the random noise and to distinguish between real and artificial samples.

However, the applicability of GANs is significantly limited due to the mode collapse — a phenomenon when the model fits only a subset of the training distribution. Additionally, they suffer from the unstable optimization process that could invalidate outputs of the generator in just a few gradient steps and fail to converge even when the model is ε-close to the equilibrium.

In this thesis, we notice that in most GANs, a significant improvement in the mode coverage is obtained by smoothing the weights of a generator during training and then generating samples at test time with a network having smoothed weights (generator-EMA). Inspired by the self-distillation methods of semi-supervised learning like Mean Teacher, we propose a simple modification, called LeakyEMA, which additionally exploits outputs of generator-EMA during training.

We prove its benefits theoretically on a simplified problem and draw connections with other stabilization techniques. Extensive experiments on a wide range of datasets and models demonstrate the superiority of our method over existing approaches. Namely, on a challenging Imagenet generation task, LeakyEMA improves Frechet Inception Distance (FID) — a metric of visual quality and diversity – by 12.7 percents on a state-of-the-art BigGAN architecture. It also consistently improves Spectral-Normalized GANs on various LSUN subsets by $20 - 40$ percents and StyleGAN-2 model on a Cifar-10 dataset by 6.5 percents.

Аннотация (Summary in Russian)

Состязательное обучение в настоящее время является одной из ведущих парадигм в крупномасштабном генеративном моделировании. Для таких задач, как генерация изображений, оно соответствует семейству генериративных состязательных сетей (GAN-ы), которые совместно обучают две сети — генератор и дискриминатор — для генерации искусственных выборок из случайного шума и различения реальных и искусственных выборок.

Однако применимость GAN-ов значительно ограничена из-за коллапса мод — явления, когда обученная модель генерирует только подмножество обучающего распределения. Кроме того, они страдают от нестабильного процесса оптимизации, который может сделать недействительными выходы генератора всего за несколько градиентных шагов и не сойтись, даже если модель находится ε-близко к состоянию равновесия.

В этой работе мы замечаем, что в большинстве GAN-ов значительное увеличение числа покрытых мод достигается за счет сглаживания весов генератора во время обучения, а затем генерации выборок во время тестирования с помощью сети, имеющей сглаженные веса ("усредненный генератор"). Вдохновленные методами само-дистилляции для обучения с небольшим количеством разметки, таких как Mean Teacher, мы предлагаем простую модификацию под названием LeakyEMA, которая дополнительно использует выходы усреднённого генератора во время обучения.

Мы доказываем её преимущества теоретически на упрощенной задаче и проводим связи с другими методами стабилизации. Обширные эксперименты на большом числе датасетов и моделей демонстрируют превосходство нашего метода над существующими подходами. А именно, в сложной задаче создания искусственных изображений на данных Imagenet, LeakyEMA улучшает Frechet Inception Distance — метрику визуального качества и разнообразия — на 12.7 процентов используя наилучшую на данный момент архитектуру BigGAN. Она также стабильно улучшает качество спектрально-нормализованных GAN-ов для различных подмножеств датасета LSUN на $20 - 40$ процентов и модель StyleGAN-2 на датасете Cifar-10 на 6.5 процентов.

Contents

1 Introduction

1.1 Generative Modeling

	Explicit	Implicit
One-pass	Variational [33], Normalizing Flows [54]	**Adversarial** [21], Latent optimization [7]
Iterative	Autoregressive [51], Denoising [25]	Energy-based [15], Gradient estimation [26]

Table 1: Dichotomies of existing generative families. The focus of this work is in bold.

Generative modelling is a problem where one wishes to construct an explicitly defined process that produces samples of the unknown distribution represented by some (limited) set of examples [20].

More formally, given an (unknown) distribution $\mathtt{p}(x)$ and a dataset $\{x_i : x_i \sim \mathtt{p}(x)\}_{i=1}^N$ we are interested in training a (stochastic) process G_θ parameterized by θ that receives some random input z sampled from a fixed distribution $\mathtt{p}_0(z)$ (e.g. standard normal) and outputs $\hat{x} = G_\theta(z)$ that is "realistic" in terms of $\mathtt{p}(x)$. Moreover, we require that a pushforward distribution $\hat{\mathtt{p}}_\theta(x)$ defined by transferring $\mathtt{p}_0(z)$ through G_θ is "close" to $\mathtt{p}(x)$: $\forall x \in \operatorname{supp}(\mathtt{p})\ \hat{\mathtt{p}}_\theta(x) \triangleq \mathtt{p}_0(\{z : \ G_\theta(z) = x\}) \approx \mathtt{p}(x)$.

Depending on the training method, we may either obtain $\mathtt{p}_\theta$ in the *explicit* form, which would correspond to the *explicit* generative modelling, or only get a (parameterized) sampler from $\mathtt{p}_\theta$, which corresponds to the *implicit* generative modelling. Additionally, one could obtain $\hat{x}$ with a single forward pass on a neural network or through an iterative process that uses one or multiple neural networks to progressively "get closer" to the desired distribution $\mathtt{p}$.

Therefore, we could separate the space of existing approaches by two attributes — whether they are explicit or implicit, and whether they are one-pass or iterative, as shown in table 1.1. In this thesis, we focus on implicit one-pass generative modelling, which is motivated by two reasons.

First, while iterative methods outperform one-pass in quality, they are currently much more computationally demanding [8]. Furthermore, they are less interpretable because one has to reason about a process where for a given layer of a neural network, its representations influence not just the outputs of subsequent layers but also the features on previous layers during the next inference steps.

Second, models with explicit density usually rely on some form of encoder that maps real images in a latent space $\mathcal{Z}$ and are learned to maximize likelihood or its estimates on training data. Currently, it is not clear whether maximum likelihood is the desired objective in the first place [63, 13], as well as whether a single network pass could implement sufficiently powerful encoding for a given G [3]. We could address the second problem by restricting G to be invertible [54], but this reduces its representation capacity and increases computational demands.

Hence in the subsequent sections, we consider only one-pass implicit models.

1.2 Generative Adversarial Networks

For the problem of implicit generative modelling, a leading paradigm nowadays is a Generative Adversarial Network (GAN) [19].

It solves this task in a game-theoretic manner by noticing that if two samplers model different distributions we could train a sufficiently powerful network D (discriminator) to separate them (e.g. via binary classification). As such, a good sampler G (generator) is the one for which the trained discriminator would be bad at solving the separation task. This gives us the following objective:

$$\hat{\theta} = \arg\min_{\theta} \max_{\psi} V(G_\theta, D_\psi; \mathtt{p}_X), \tag{1}$$

where θ and ψ are trainable parameters of the generator and discriminator respectively, $\mathtt{p}_X$ is the empirical distribution of a given dataset X and V is some discriminator objective that forces it to separate training and generated data, e.g. log-likelihood in a binary classification task:

$$V(G_\theta, D_\psi; \mathtt{p}_X) = \mathbb{E}_{z\sim \mathtt{p}_0(z)} \varphi(-D_\psi(G_\theta(z))) + \frac{1}{|X|}\sum_{x_i \in X} \varphi(D_\psi(x_i)),\ \ \varphi(t) = \log\sigma(t) = \log[\frac{1}{1+e^{-t}}].$$

Note that maximization of V wrt D forces it to output low scores $D_\psi(x)$ on artificial inputs and large scores on natural images. By minimizing the obtained solution wrt G we force it to "fool" D into believing that generated images also have large scores, which for a sufficiently good discriminator is only possible when distributions $\mathtt{p}(x)$ and $\hat{\mathtt{p}}_\theta(x)$ match [21].

Although simple and promising, this method has several drawbacks.

The first is that for training it with gradient descent, we need to differentiate the worst-case objective V wrt θ, which is intractable in the general case as it requires optimizing D for many steps until convergence. Even worse, the ideal discriminator D_{ψ^*} can produce vanishing gradients for G without additional regularizations [4]. This is possible whenever the distributions $\mathtt{p}$ and $\hat{\mathtt{p}}_\theta$ have *disjoint supports*, which is often the case in high-dimensional problems [5].

A partial remedy for this is that in practice, G and D are usually trained in an alternating fashion against their previous states, which gives the following trajectory:

$$\begin{cases} \hat{\psi}_t = \hat{\psi}_{t-1} + \gamma \frac{\partial V}{\partial \psi}(G_{\hat{\theta}_{t-1}}, D_{\hat{\psi}_{t-1}}; \mathtt{p}_X); \\ \hat{\theta}_t = \hat{\theta}_{t-1} - \gamma \frac{\partial V}{\partial \theta}(G_{\hat{\theta}_{t-1}}, D_{\hat{\psi}_t}; \mathtt{p}_X); \end{cases} \tag{2}$$

where γ is a learning rate, and the updates are performed until convergence.

This method is called Gradient Descent-Ascent (GDA), and its stable points are different from the stable points of (1), which correspond to the Nash equilibria [27]. Surprisingly, GDA can produce reasonable solutions even when the "ideal" GAN fails [16, 17], which is attributed to its implicit regularization

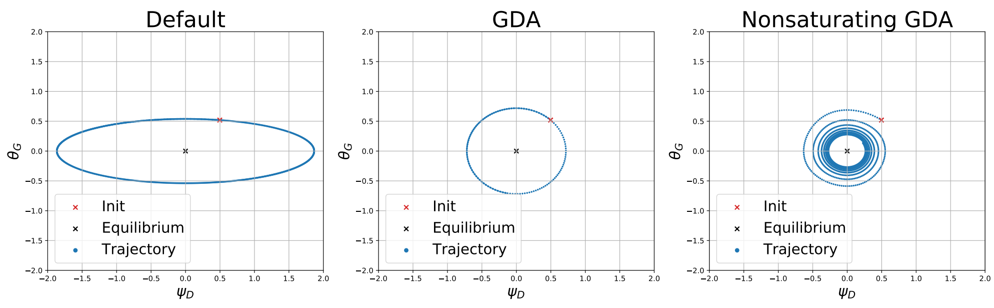

Figure 1: Trajectories of various adversarial methods on a toy one-dimensional Dirac problem [43]. Here, the ground truth distribution is given by δ_0, the generator is formulated as δ_θ, and the discriminator is linear: $D_\psi(x) = \psi x$. We train models with (from left to right): the original adversarial objective (1), alternating gradient descent (2), GDA with a nonsaturating objective (3).

effects [57]. Nevertheless, it is believed that the quality of gradients provided by D at the G update step is far from optimal [39, 70, 58].

The second drawback of generative adversarial training is its instability or "cycling behaviour". This issue arises because neither (1) nor (2) may be formulated in terms of gradient descent on a single objective. As such, even when V is convex wrt θ and ψ, the Jacobian eigenvalues of their gradients may have a large imaginary part, which means that the joint trajectory of (θ, ψ) will rotate around the equilibrium [6, 43]. We demonstrate an example of this in Figure 1.

A minor modification that addresses this is to replace a shared function V with a pair of individual losses $\mathcal{L}_G$, $\mathcal{L}_D$ that yield slightly less rotation. For instance, [21] proposes to use:

$$\begin{cases} \mathcal{L}_D(G_\theta, D_\psi; \mathrm{p}_X) = \mathbb{E}_{z\sim \mathrm{p}_0(z)} \varphi(-D_\psi(G_\theta(z))) + \frac{1}{|X|}\sum_{x_i \in X} \varphi(D_\psi(x_i)); \\ \mathcal{L}_G(G_\theta, D_\psi) = \mathbb{E}_{z\sim \mathrm{p}_0(z)} \varphi(D_\psi(G_\theta(z))), \end{cases} \tag{3}$$

which is called a nonsaturating objective. Additionally, many existing regularization techniques for GANs implicitly reduce this cycling behaviour, as we discuss in section 2.

The third drawback of adversarial training is its tendency towards degenerate solution referred to as mode collapse [32, 44, 60]. A mode collapse occurs when the generator learns to mimic only a subset of the training data, while the discriminator focuses on distinguishing high-level features of this particular subset and thus penalizes the generator if it tries to improve its diversity at the cost of lower quality of individual samples. This high-level intuition serves as a motivation for our approach described in section 3.

Despite those issues, the state-of-the-art GANs [9, 29] are able to produce diverse photorealistic images on challenging domains, which makes them an important ingredient in a wide range of applications, including image editing [72, 1, 64], unsupervised representation learning [14, 67], inverse rendering [52, 71], saliency detection [65, 2, 41], super-resolution [42] and many others. This puts additional weight on the importance of solving aforementioned drawbacks.

2 Related Work

2.1 GAN Regularizations

Stable GAN training is a long-standing challenge in the computer vision community. Over the years, it has been addressed from several sides, including architectural changes [53, 30, 70, 9, 28, 29], different loss functions [5, 49, 37, 38], alternative optimization methods [12, 45, 66, 10, 11] and regularizers for both generator [29, 50] and discriminator [46, 23, 59, 43, 55] objectives.

The recent extensive comparisons [40, 35] demonstrate that under an equal hyperparameter budget with fixed architecture, the most significant benefits come from the use of discriminator regularizations. This is because there are many imperceptible details by which training and generated distributions may differ and usually discriminator overfits and labels even validation data as fake [31]. It is one of the sources of "cycling behaviour" since even when the generator is optimal, but the training distribution is not achievable, the discriminator provides non-zero gradients, and the system deviates from the "equilibrium".

Instance noise (IN) [59, 4] addresses this problem by adding isotropic Gaussian noise to both real and fake images during the discriminator step. We can view it as a simple data augmentation that effectively creates a noticeable overlap of the "spread" distributions, containing challenging samples for the discriminator. As a result, it simultaneously reduces the gap between training and optimal fake distributions, stabilizing the model around equilibrium and fixes the problem of disjoint supports.

One of the difficulties in IN is that convergence is only guaranteed in expectation, i.e. with a fixed batch, we need to average gradients across many noise samples added to inputs. Gradient penalties [23, 43, 55] avoid this by considering first-order analytical approximations of this expectation, which we can minimize heuristically by enforcing the discriminator gradient norms to remain constant across different inputs [55]. They also have roots in the famous Wasserstein objective [4, 5], which prevents the divergence by restricting the Lipschitz norm of the discriminator.

An alternative regularization approach is Spectral Normalization [46], which dynamically rescales weights based on the estimates of their largest singular value, thus forcing all the layers to stay 1-Lipschitz.

We view our work as an extension of the original IN approach that uses a "smarter" version of the noising distribution, as discussed in section 3. We also found the benefits of it to be cumulative with those of gradient penalties and spectral normalization (see section 4).

2.2 Parameter smoothing and Mean Teacher

Parameter smoothing is a simple trick that significantly improves model performance in different problems. The idea is to average the parameters along their training trajectory, which may be implemented in two ways — either by taking the mean of last k steps or by accumulating their moving average with a decay coefficient α. More formally, the exponential moving average parameters θ_{EMA} of the model to be used

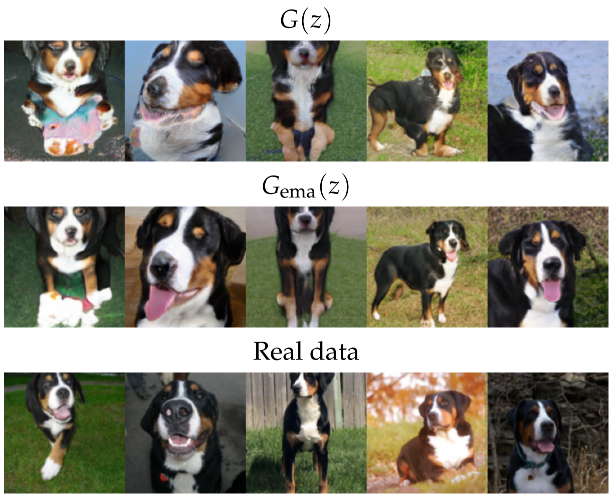

Figure 2: Samples produced by the BigGAN generator on the $100k$-th training step (top), samples produced by the EMA generator on the same step (middle), real samples from Imagenet (bottom). The EMA samples are more realistic and provide a natural "bridge" between the real and generator's distribution.

during evaluation are updated after every training step of original parameters θ in the following way:

$$\theta_{\text{EMA},t} \leftarrow \alpha\theta_{\text{EMA},t-1} + (1-\alpha)\theta_t.$$

Very recently, the EMA parameters were shown to provide useful supervision signal to train discriminative models [62, 48, 22]. For instance, in the context of semi-supervised learning, enforcing the consistency of predictions with the weight-average models significantly improves classification accuracy [62]. BYOL [22], a very recent method, also exploiting the idea of mean teacher prediction, was shown to provide the state-of-the-art performance on self-supervised pretraining.

In the context of GANs, the benefits of weight averaging have been investigated only recently, from both theoretical [18, 68] and empirical standpoints [30]. At the moment, it is employed by state-of-the-art models, e.g., BigGAN and StyleGAN2, which use the exponential moving average (EMA) generator's parameters for evaluation. We also validate their profits empirically in Figure 2 and Table 2.

However, to the best of our knowledge, EMA samples are not currently used to guide the training process by any means, and in this work, we try to address this gap.

Model	SNGAN [46]	SNGAN	BigGAN [9]	StyleGAN2 [29]
Dataset	LSUN Church	LSUN Bedroom	Imagenet	Cifar10
Generator	8.62	18.00	26.08	11.93
Generator-EMA	7.93	13.48	10.14	6.32

Table 2: Best Frechet Inception Distance (FID) [24] scores of various models without and with parameter smoothing. EMA gives significant improvement in all cases.

3 Method

3.1 Formulation

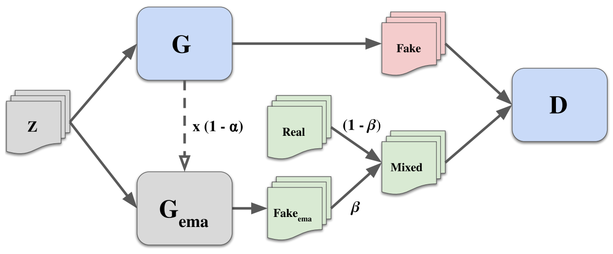

Figure 3: Protocol of training with LeakyEMA. The trainable components are denoted by blue, the images labeled as "real" are colored in green.

The superior quality of θ_{EMA} is widely used during evaluation but is not used for training. Now we propose to exploit it inside the discriminator training step by replacing a subset of real images with the ones generated by G_{EMA}. Formally, our modified objective is the following:

$$\begin{cases} \psi^* = \arg\min_{\psi} \mathcal{L}_D(G_\theta, D_\psi; \widetilde{\mathrm{p}}_d(x)); \\ \theta^* = \arg\min_{\theta} \mathcal{L}_G(G_\theta, D_\psi); \end{cases} \tag{4}$$

where $\tilde{\mathrm{p}}_d(x)$ is a mixture of the distribution defined by G_{EMA} and the empirical distribution:

$$\widetilde{\mathrm{p}}_d(x) = \beta \mathrm{p}_G(x; \theta_{\text{EMA}}) + (1-\beta)\mathrm{p}_X(x);\ \mathrm{p}_X(x) = \mathcal{U}(\{x_i \in X\}) \tag{5}$$

and β determines the portion of the real images replaced by the EMA samples in each training batch. We refer to this modification of the training protocol as **LeakyEMA**. Training with LeakyEMA is schematically presented in Figure 3, and we can effortlessly implement it in a few lines of code. Note that on each training step, we use the same noise samples z for both G and G_{EMA}, a choice we ablate in section 4.1. In experiments, we observed that it is necessary to use LeakyEMA on the later epochs for large-scale tasks since, on the earlier epochs, the EMA samples are poor and cannot reliably guide the training. Therefore, LeakyEMA is used only after the N-th training iteration, where N is a hyperparameter of our method. In the experimental section 4.1, we ablate on both β and N and provide recipes on setting them.

3.2 Computational complexity

Since LeakyEMA requires several EMA samples for each training batch, each optimization step has a computational overhead compared to the conventional GAN training. However, this overhead includes

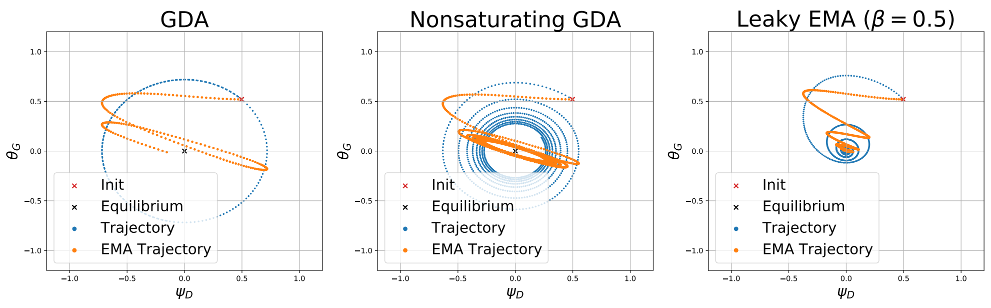

Figure 4: Comparison of the conventional and LeakyEMA protocols trajectories for Dirac-GAN. We depict the initial state, the (desired) equilibrium point and the trajectories of (θ, ψ) and $(\theta_{\text{EMA}}, \psi)$.

only a forward pass in the EMA generator and does not increase the complexity of backward passes in both generator and discriminator. Moreover, the overhead appears only during the last iterations, so the increase in total training time was negligible in all our experiments.

3.3 Connections with Instance Noise

In this section, we explain the relationship between the proposed LeakyEMA training and the well-known instance noise (IN) trick [59, 4]. Both LeakyEMA and IN are motivated by the disjoint supports problem when the model and data distributions do not overlap. In this case, it is easy for the discriminator to achieve perfect classification, which results in vanishing gradients for the generator, therefore gives slow and unstable optimization. As discussed in section 2, IN mitigates this problem by adding isotropic Gaussian noise to both real and fake images. However, in high-dimensional spaces, only a small amount of noisy samples fall into the overlap created by it due to the isotropy of noise and the curse of dimensionality. Therefore, most of the samples are still “easy” for the discriminator. In contrast, LeakyEMA is based on the previous evidence [30, 9, 28] that EMA samples are more appealing compared to the generator samples (see Figure 2). In other words, in terms of realism, EMA samples are “intermediate” between real and fake images. Hence, it is natural to use them to create the overlap between distributions.

3.4 Local Convergence with LeakyEMA

For a theoretical understanding of the regularization effect stemming from LeakyEMA, we follow [43] that considers the behaviour of GANs with various regularizers on a simple 1-dimensional problem and demonstrates that not all of the methods are locally convergent with gradient descent. Using the same analysis, we study the gradient vector field associated with the GAN training dynamics and show that LeakyEMA is locally convergent. In particular, we prove in the supplementary A that all Jacobian eigenvalues of the field have negative real-part in the neighbourhood of the Nash equilibrium. Figure 4 validates this finding empirically.

4 Experiments

In this section, we evaluate LeakyEMA on a large number of standard benchmarks.

First, we investigate its sensitivity to hyperparameters and perform ablation analysis. Second, we demonstrate that LeakyEMA can be successfully combined with established stabilization techniques. Finally, we provide aggregated results across a wide range of datasets and architectures, including the BigGAN model [9] on a challenging Imagenet task. For quantitative evaluation, we use the Frechet Inception Distance (FID) [24] computed on the image representations extracted from the InceptionV3 model [61][1]. Unless stated otherwise, we **always report** the quality of samples produced by the **EMA** generator.

4.1 Ablation and sensitivity to hyperparameters

In this part, we use the ResNet-SNGAN [46] architecture, which generates 128×128 images, originally proposed in [46] (see table 6a) . We use N_{dis}=5 discriminator updates per single generator step. The optimization is performed by Adam [34] with a constant learning rate of 2×10^{-4} and β_1=0.5, and we accumulate the generator weights in the EMA version with α=0.999. Training with LeakyEMA proceeds in two stages. During the first stage, we train GAN in a conventional way for N steps with a Hinge variant of adversarial training objective [46] and a batch size of 64. After N steps, the second stage starts, where the EMA samples replace a portion of real images in each training batch.

N	$\beta = 0$	$\beta = 0.25$	$\beta = 0.5$	$\beta = 0.75$
		FID		
$80k$	$7.94_{\pm 2.40}$	$5.33_{\pm 0.45}$	$4.21_{\pm 0.21}$	$5.05_{\pm 0.63}$
$100k$	$7.94_{\pm 2.40}$	$5.26_{\pm 0.47}$	$\underline{4.15}_{\pm 0.31}$	$5.01_{\pm 0.61}$
$120k$	$7.94_{\pm 2.40}$	$5.25_{\pm 0.47}$	$\underline{4.17}_{\pm 0.24}$	$4.83_{\pm 0.35}$
$140k$	$7.94_{\pm 2.40}$	$6.13_{\pm 1.83}$	$5.62_{\pm 2.52}$	$6.06_{\pm 2.04}$

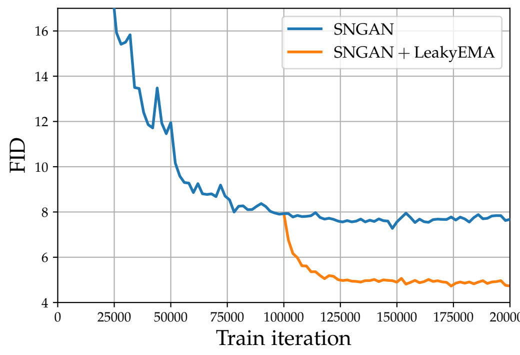

Table 3: **Left:** Best performance of SNGAN trained on LSUN-Church dataset (128×128) with different values of β and N. $\beta = 0$ corresponds to training without LeakyEMA, therefore, numbers in the second column are the same. **Right:** Evolution of FID learning curves in Default and LeakyEMA regime (with $\beta = 0.5,\ N = 100k$).

Sensitivity to β and N. First, we investigate how LeakyEMA performs with different values of β and N hyperparameters. The complete evaluation of SNGAN on the LSUN-Church dataset [69] is provided in Table 3. All the models are trained for 250K generator optimization steps, and we report mean and std values computed for five independent runs. The key observations from Table 3 are the following:

[1] https://github.com/mseitzer/pytorch-fid

- LeakyEMA generally benefits generation performance in a wide range of β values. The lowest FID obtained with LeakyEMA is about four, while the best FID obtained with a conventional training ($\beta = 0$) is about eight.
- By varying β, one can interpolate between the conventional training ($\beta = 0$) and pure self-training ($\beta = 1$), which always collapsed in our experiments. In this ablation, $\beta = 0.5$ appears to be a "sweet spot" in terms of FID; therefore, we use this value in the following experiments.
- In Table 3, we consider only N values that are close to the moment when a learning curve of the conventional training starts to slide down slowly, as shown in the right part of 3. Intuitively, at this moment, EMA samples are already good enough to serve as a supervision signal, but the model has not collapsed yet. Typically, LeakyEMA is not sensitive to N chosen within this region. However, too large N values ($N = 140k$) correspond to partially collapsed models, which results in inferior FID. In general, LeakyEMA increases generation quality for all N chosen after the training curve decline slows down.

Ablation: Should EMA be "synced" with the generator? In principle, one could perform the conventional GAN training until convergence and then use the obtained EMA using its samples to train a new generator with our technique. Despite longer training, such protocol could be reasonable, since in this case, EMA samples have higher quality compared to the EMA samples from intermediate steps, thus, it could provide more reliable supervision. Nevertheless, we show that this "async" strategy is inferior to the setup when the generator and its EMA version are updated synchronously. In particular, we compare three possible versions of training with LeakyEMA:

	Baseline (no leaking)	EMA sync	EMA async-noise	EMA async-model	SAME random	SAME top-k
FID	7.93	4.15	4.16	5.85	8.32	7.91

Table 4: Performance of different methods for leaking in SNGAN on the LSUN-Church dataset.

1. **EMA-sync**, a fully synchronized version, where the samples from the current EMA guide training. The previous experiments were performed with this version.
2. **EMA-async-noise**, the same as above, but on each training iteration, *different noise samples* are used to produce fake images from the generator and EMA. Note that in the **EMA-sync** version, the same noise samples are always used.
3. **EMA-async-model**, a version where the previously pre-trained (fixed) EMA is used to guide the training on all iterations.

We report the performance of three different versions in Table 4, which demonstrates that the asynchronous **EMA-async-model** protocol is inferior in terms of FID. These results highlight the importance of

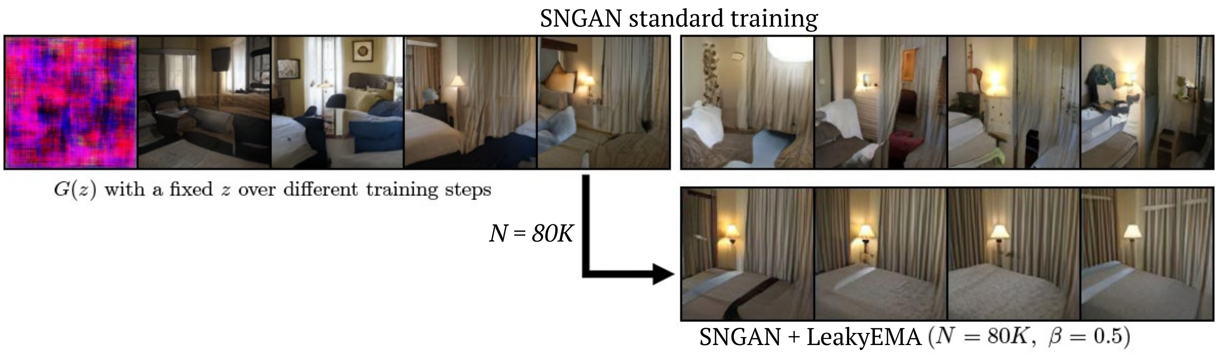
SNGAN standard training
$G(z)$ with a fixed z over different training steps
$N = 80K$
SNGAN + LeakyEMA $(N = 80K,\ \beta = 0.5)$

Figure 5: Evolution of a generated sample $G(z)$ with a fixed z. *Top row*: conventional; *bottom row*: LeakyEMA training, which reduces the sample variation over the training steps.

synchronicity between the current generator and the corresponding EMA, which is crucial for EMA samples to serve as a "connection" of the model and data distributions. While the synchronicity of the noise samples does not affect the LeakyEMA performance, we recommend using the **EMA-sync** version due to the simplicity of implementation.

Ablation: Are EMA samples necessary? To verify that the samples from the generator itself cannot replace EMA samples in our approach, we perform the following experiment. Instead of using EMA samples to replace a portion of real images, we tried to use random generator samples (**SAME-random**) and generator samples that are "the most realistic" for the current discriminator (**SAME-top-k**). We use ResNet-SNGAN on LSUN-Church with N=100K and β=0.5. In both cases, the performance was equal or inferior to the ResNet-SNGAN trained conventionally (see Table 4). Thus, this experiment provides additional evidence that the superiority of EMA over the generator is a vital ingredient of LeakyEMA success.

4.2 Comparison with other regularizers

In these experiments, we show that LeakyEMA can be combined with established stabilization techniques. In particular, we evaluate the training with LeakyEMA using the WGAN-GP [23] and the SNGAN [46] models. For a fair comparison, we use the same model architecture as described in the ablation section. The exact hyperparameter values are provided in supplementary. We compare the models on three datasets from different domains. In all datasets we resize images to 128×128.

1. **LSUN-Church** [69] containing outdoor scene images with multiple details, challenging to generate properly.

2. **LSUN-bedroom** [69], containing indoor images of bedrooms.

3. **FFHQ** [28], containing human face images.

Model	Church	Bedroom	FFHQ
WGAN-GP	9.51	12.66	—
WGAN-GP+LeakyEMA	8.62	11.61	—
SNGAN	7.93	13.48	14.80
SNGAN+LeakyEMA	4.15	4.67	11.87

Table 5: Training with LeakyEMA combined with previous stabilization techniques.

Comparison of WGAN-GP and SNGAN with the conventional and LeakyEMA training on three datasets is presented in 5. We choose N to be the iteration corresponding to the lowest FID obtained in the conventional protocol for all experiments, but not larger than $140k$. "—" in a table cell denotes that WGAN-GP did not converge on the dataset.

In all datasets, LeakyEMA provides a significant performance boost both in terms of FID and subjective visual quality. In Figure 5, we illustrate the evolution of a particular sample during the conventional training and during the LeakyEMA training (with N=80K and β=0.5) for SNGAN trained on Bedroom. To produce the top row, we consider a checkpoint of the generator at the step N=80K, where the LeakyEMA training starts, and train it independently with $\beta = 0$. Notably, our technique stabilizes the sample variability over training steps and also increases its quality.

4.3 Large scale experiments

Having established the benefits of our method over existing regularization techniques, we evaluate it across a wide range of tasks, including the challenging Imagenet benchmark. In the last experiment, we use the publicly available PyTorch implementation of BigGAN[2].

Model	BigGAN	SNGAN	SNGAN	SNGAN	StyleGAN2	StyleGAN2	StyleGAN2
Dataset	Imagenet	Church	Bedroom	FFHQ	Cifar10	Church	Bedroom
Resolution	128×128	128×128	128×128	128×128	32×32	256×256	256×256
Default	10.14	7.93	13.48	14.80	6.32	3.10	7.76
+ LeakyEMA	8.85	4.15	4.67	11.87	5.91	3.48	4.05

Table 6: Comparison of various default models with their LeakyEMA versions.

The aggregated results are provided in Table 6. In general, our method performs favourably across a wide range of setups. Unfortunately, we found that our technique **mostly fails when combined with StyleGAN2 on large-resolution datasets**, a phenomenon to be studied in future work.

We also compare our best configuration with the state-of-the-art checkpoint provided by the BigGAN authors[3] in Table 7. Along with FID, we use Precision and Recall metric [36] computed with embeddings

[2]https://github.com/ajbrock/BigGAN-PyTorch
[3]https://drive.google.com/file/d/1nAle7FCVFZdix2--ks0r5JBkFnKw8ctW/view

	FID	Precision	Recall
Imagenet 128×128			
BigGAN	10.14	0.92	0.21
BigGAN+LeakyEMA	8.85	0.91	0.25

Table 7: Comparison of the model trained with LeakyEMA and the publicly available state-of-the-art BigGAN checkpoint.

from Imagenet pre-trained VGG. The BigGAN trained with LeakyEMA achieves significantly higher Recall, indicating that our LeakyEMA improves the model coverage. We get similar results using more advanced Density and Coverage metrics [47].

In Figure 6, we plot Precision-Recall and FID-IS [56] curves obtained by varying a truncation value in a range $[0.3, 1.0]$ [9], which is a standard way to quantify the quality-diversity tradeoff. The BigGAN+LeakyEMA outperforms the standard BigGAN across all operating points.

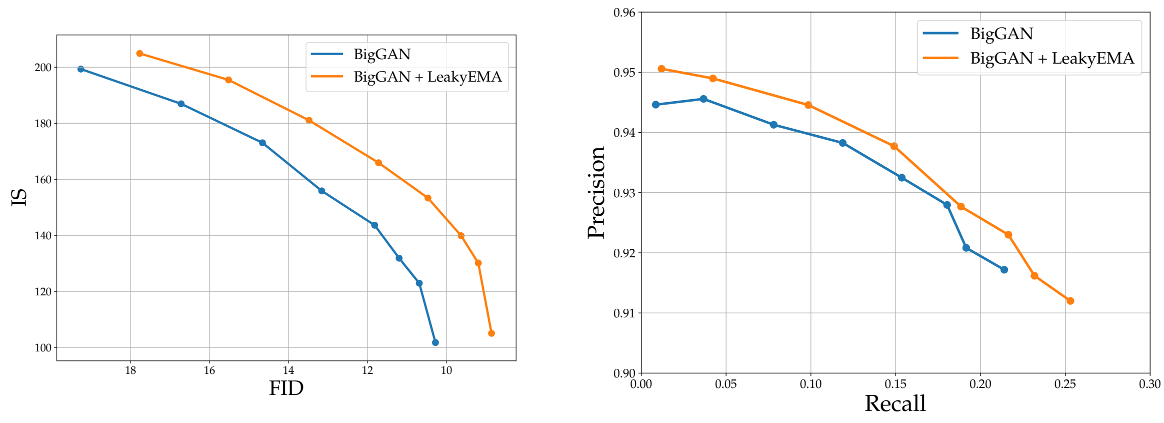

Figure 6: The FID-IS (left) and Precision-Recall (right) curves from varying truncation values linearly from 0.3 to 1.0 for the best model obtained with LeakyEMA and the publicly available BigGAN.

5 Conclusion

Despite the recent progress in stabilizing GANs, the training of large-scale models on complex multi-modal datasets, e.g., Imagenet, is still challenging. This thesis proposed a simple modification to the GAN training protocol that leverages an additional supervision signal from the generator parameters averaged over previous optimization steps. Our technique is simple and efficient, resulting in substantial empirical improvements on the considered benchmarks, including Imagenet. Furthermore, the proposed approach allows for certain theoretical guarantees of local convergence. We expect that given the simplicity and empirical success, an idea of using mean teachers for GANs can be elaborated further from both practical and theoretical standpoints.

References

[1] Rameen Abdal, Yipeng Qin, and Peter Wonka. *Image2StyleGAN: How to Embed Images Into the StyleGAN Latent Space?* 2019. arXiv: 1904.03189 [cs.CV].

[2] Rameen Abdal et al. *Labels4Free: Unsupervised Segmentation using StyleGAN*. 2021. arXiv: 2103.14968 [cs.CV].

[3] Yuval Alaluf, Or Patashnik, and Daniel Cohen-Or. *ReStyle: A Residual-Based StyleGAN Encoder via Iterative Refinement*. 2021. arXiv: 2104.02699 [cs.CV].

[4] Martin Arjovsky and Léon Bottou. *Towards Principled Methods for Training Generative Adversarial Networks*. 2017. arXiv: 1701.04862 [stat.ML].

[5] Martin Arjovsky, Soumith Chintala, and Léon Bottou. *Wasserstein GAN*. 2017. arXiv: 1701.07875 [stat.ML].

[6] Hugo Berard et al. "A Closer Look at the Optimization Landscapes of Generative Adversarial Networks". In: *International Conference on Learning Representations*. 2020. URL: https://openreview.net/forum?id=HJeVnCEKwH.

[7] Piotr Bojanowski et al. *Optimizing the Latent Space of Generative Networks*. 2019. arXiv: 1707.05776 [stat.ML].

[8] Sam Bond-Taylor et al. *Deep Generative Modelling: A Comparative Review of VAEs, GANs, Normalizing Flows, Energy-Based and Autoregressive Models*. 2021. arXiv: 2103.04922 [cs.LG].

[9] Andrew Brock, Jeff Donahue, and Karen Simonyan. *Large Scale GAN Training for High Fidelity Natural Image Synthesis*. 2019. arXiv: 1809.11096 [cs.LG].

[10] Tatjana Chavdarova et al. *Reducing Noise in GAN Training with Variance Reduced Extragradient*. 2020. arXiv: 1904.08598 [stat.ML].

[11] Tatjana Chavdarova et al. *Taming GANs with Lookahead-Minmax*. 2020. arXiv: 2006.14567 [stat.ML].

[12] Constantinos Daskalakis and Ioannis Panageas. *The Limit Points of (Optimistic) Gradient Descent in Min-Max Optimization*. 2018. arXiv: 1807.03907 [math.OC].

[13] Sander Dieleman. *Musings on typicality*. 2020. URL: https://benanne.github.io/2020/09/01/typicality.html.

[14] Jeff Donahue and Karen Simonyan. *Large Scale Adversarial Representation Learning*. 2019. arXiv: 1907.02544 [cs.CV].

[15] Yilun Du and Igor Mordatch. "Implicit Generation and Modeling with Energy Based Models". In: *Advances in Neural Information Processing Systems*. Ed. by H. Wallach et al. Vol. 32. Curran Associates, Inc., 2019. URL: https://proceedings.neurips.cc/paper/2019/file/378a063b8fdb1db941e34f4bde584c7d-Paper.pdf.

[16] Farzan Farnia and Asuman Ozdaglar. *GANs May Have No Nash Equilibria.* 2020. arXiv: 2002.09124 [cs.LG].

[17] William Fedus et al. *Many Paths to Equilibrium: GANs Do Not Need to Decrease a Divergence At Every Step.* 2018. arXiv: 1710.08446 [stat.ML].

[18] Gauthier Gidel et al. “A Variational Inequality Perspective on Generative Adversarial Networks”. In: *International Conference on Learning Representations.* 2019.

[19] Ian Goodfellow. “NIPS 2016 tutorial: Generative adversarial networks”. In: *Advances in Neural Information Processing Systems* (2016).

[20] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning.* http://www.deeplearningbook.org. MIT Press, 2016.

[21] Ian J. Goodfellow et al. *Generative Adversarial Networks.* 2014. arXiv: 1406.2661 [stat.ML].

[22] Jean-Bastien Grill et al. “Bootstrap your own latent: A new approach to self-supervised learning”. In: *NeurIPS* (2020).

[23] Ishaan Gulrajani et al. “Improved training of wasserstein gans”. In: *Advances in neural information processing systems.* 2017, pp. 5767–5777.

[24] Martin Heusel et al. “Gans trained by a two time-scale update rule converge to a local nash equilibrium”. In: *Advances in Neural Information Processing Systems.* 2017, pp. 6626–6637.

[25] Jonathan Ho, Ajay Jain, and Pieter Abbeel. “Denoising Diffusion Probabilistic Models”. In: *Advances in Neural Information Processing Systems.* Ed. by H. Larochelle et al. Vol. 33. Curran Associates, Inc., 2020, pp. 6840–6851. URL: https://proceedings.neurips.cc/paper/2020/file/4c5bcfec8584af0d967f1ab10179ca4b-Paper.pdf.

[26] Aapo Hyvärinen. “Estimation of Non-Normalized Statistical Models by Score Matching”. In: *Journal of Machine Learning Research* 6.24 (2005), pp. 695–709. URL: http://jmlr.org/papers/v6/hyvarinen05a.html.

[27] Chi Jin, Praneeth Netrapalli, and Michael Jordan. “What is Local Optimality in Nonconvex-Nonconcave Minimax Optimization?” In: *Proceedings of the 37th International Conference on Machine Learning.* Ed. by Hal Daumé III and Aarti Singh. Vol. 119. Proceedings of Machine Learning Research. PMLR, 13–18 Jul 2020, pp. 4880–4889. URL: http://proceedings.mlr.press/v119/jin20e.html.

[28] Tero Karras, Samuli Laine, and Timo Aila. “A style-based generator architecture for generative adversarial networks”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition.* 2019, pp. 4401–4410.

[29] Tero Karras et al. *Analyzing and Improving the Image Quality of StyleGAN.* 2020. arXiv: 1912.04958 [cs.CV].

[30] Tero Karras et al. “Progressive Growing of GANs for Improved Quality, Stability, and Variation”. In: *International Conference on Learning Representations*. 2018.

[31] Tero Karras et al. “Training Generative Adversarial Networks with Limited Data”. In: *Advances in Neural Information Processing Systems*. Ed. by H. Larochelle et al. Vol. 33. Curran Associates, Inc., 2020, pp. 12104–12114. URL: https://proceedings.neurips.cc/paper/2020/file/8d30aa96e72440759f74bd2306c1fa3d-Paper.pdf.

[32] Mahyar Khayatkhoei, Maneesh K. Singh, and Ahmed Elgammal. “Disconnected Manifold Learning for Generative Adversarial Networks”. In: *Advances in Neural Information Processing Systems 31*. Ed. by S. Bengio et al. Curran Associates, Inc., 2018, pp. 7354–7364. URL: http://papers.nips.cc/paper/7964-disconnected-manifold-learning-for-generative-adversarial-networks.pdf.

[33] Diederik P Kingma and Max Welling. *Auto-Encoding Variational Bayes*. 2014. arXiv: 1312.6114 [stat.ML].

[34] Diederik P. Kingma and Jimmy Ba. *Adam: A Method for Stochastic Optimization*. 2017. arXiv: 1412.6980 [cs.LG].

[35] Karol Kurach et al. *A Large-Scale Study on Regularization and Normalization in GANs*. 2019. arXiv: 1807.04720 [cs.LG].

[36] Tuomas Kynkäänniemi et al. “Improved precision and recall metric for assessing generative models”. In: *Advances in Neural Information Processing Systems*. 2019, pp. 3929–3938.

[37] Chun-Liang Li et al. “Mmd gan: Towards deeper understanding of moment matching network”. In: *Advances in Neural Information Processing Systems*. 2017, pp. 2203–2213.

[38] Jae Hyun Lim and Jong Chul Ye. *Geometric GAN*. 2017. arXiv: 1705.02894 [stat.ML].

[39] Xuanqing Liu and Cho-Jui Hsieh. “Rob-GAN: Generator, Discriminator, and Adversarial Attacker”. In: *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019, pp. 11226–11235. DOI: 10.1109/CVPR.2019.01149.

[40] Mario Lucic et al. “Are gans created equal? a large-scale study”. In: *Advances in neural information processing systems*. 2018, pp. 700–709.

[41] Luke Melas-Kyriazi et al. *Finding an Unsupervised Image Segmenter in Each of Your Deep Generative Models*. 2021. arXiv: 2105.08127 [cs.CV].

[42] Sachit Menon et al. *PULSE: Self-Supervised Photo Upsampling via Latent Space Exploration of Generative Models*. 2020. arXiv: 2003.03808 [cs.CV].

[43] Lars Mescheder, Sebastian Nowozin, and Andreas Geiger. “Which Training Methods for GANs do actually Converge?” In: *International Conference on Machine Learning (ICML)*. 2018.

[44] Luke Metz et al. *Unrolled Generative Adversarial Networks*. 2017. arXiv: 1611.02163 [cs.LG].

[45] Luke Metz et al. “Unrolled generative adversarial networks”. In: *ICLR* (2017).

[46] Takeru Miyato et al. “Spectral Normalization for Generative Adversarial Networks”. In: *International Conference on Learning Representations*. 2018. URL: https://openreview.net/forum?id=B1QRgziT-.

[47] Muhammad Ferjad Naeem et al. “Reliable Fidelity and Diversity Metrics for Generative Models”. In: (2020).

[48] Duc Tam Nguyen et al. “SELF: Learning to Filter Noisy Labels with Self-Ensembling”. In: *International Conference on Learning Representations*. 2019.

[49] Sebastian Nowozin, Botond Cseke, and Ryota Tomioka. “f-gan: Training generative neural samplers using variational divergence minimization”. In: *Advances in neural information processing systems*. 2016, pp. 271–279.

[50] Augustus Odena et al. *Is Generator Conditioning Causally Related to GAN Performance?* 2018. arXiv: 1802.08768 [stat.ML].

[51] Aaron van den Oord et al. *Conditional Image Generation with PixelCNN Decoders*. 2016. arXiv: 1606.05328 [cs.CV].

[52] Xingang Pan et al. “Do 2D {GAN}s Know 3D Shape? Unsupervised 3D Shape Reconstruction from 2D Image {GAN}s”. In: *International Conference on Learning Representations*. 2021. URL: https://openreview.net/forum?id=FGqiDsBUKL0.

[53] Alec Radford, Luke Metz, and Soumith Chintala. “Unsupervised representation learning with deep convolutional generative adversarial networks”. In: *ICLR* (2015).

[54] Danilo Rezende and Shakir Mohamed. “Variational Inference with Normalizing Flows”. In: *Proceedings of the 32nd International Conference on Machine Learning*. Ed. by Francis Bach and David Blei. Vol. 37. Proceedings of Machine Learning Research. Lille, France: PMLR, July 2015, pp. 1530–1538. URL: http://proceedings.mlr.press/v37/rezende15.html.

[55] Kevin Roth et al. “Stabilizing Training of Generative Adversarial Networks through Regularization”. In: *Advances in Neural Information Processing Systems*. Ed. by I. Guyon et al. Vol. 30. Curran Associates, Inc., 2017. URL: https://proceedings.neurips.cc/paper/2017/file/7bccfde7714a1ebadf06c5f4cea752c1-Paper.pdf.

[56] Tim Salimans et al. “Improved techniques for training gans”. In: *Advances in neural information processing systems*. 2016, pp. 2234–2242.

[57] Florian Schäfer, Hongkai Zheng, and Anima Anandkumar. *Implicit competitive regularization in GANs*. 2020. arXiv: 1910.05852 [cs.LG].

[58] Edgar Schonfeld, Bernt Schiele, and Anna Khoreva. “A U-Net Based Discriminator for Generative Adversarial Networks”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2020.

[59] Casper Kaae Sønderby et al. "Amortised MAP Inference for Image Super-resolution". In: *ICLR*. 2017.

[60] Akash Srivastava et al. "VEEGAN: Reducing Mode Collapse in GANs using Implicit Variational Learning". In: *Advances in Neural Information Processing Systems*. Ed. by I. Guyon et al. Vol. 30. Curran Associates, Inc., 2017. URL: https://proceedings.neurips.cc/paper/2017/file/44a2e0804995faf8d2e3b084a1e2db1d-Paper.pdf.

[61] Christian Szegedy et al. "Rethinking the Inception Architecture for Computer Vision". In: (2015). arXiv: 1512.00567 [cs.CV].

[62] Antti Tarvainen and Harri Valpola. "Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results". In: *Advances in neural information processing systems*. 2017, pp. 1195–1204.

[63] Lucas Theis, Aäron van den Oord, and Matthias Bethge. *A note on the evaluation of generative models*. 2016. arXiv: 1511.01844 [stat.ML].

[64] Yuri Viazovetskyi, Vladimir Ivashkin, and Evgeny Kashin. *StyleGAN2 Distillation for Feed-forward Image Manipulation*. 2020. arXiv: 2003.03581 [cs.CV].

[65] Andrey Voynov, Stanislav Morozov, and Artem Babenko. *Big GANs Are Watching You: Towards Unsupervised Object Segmentation with Off-the-Shelf Generative Models*. 2020. arXiv: 2006.04988 [cs.LG].

[66] Yuanhao Wang, Guodong Zhang, and Jimmy Ba. *On Solving Minimax Optimization Locally: A Follow-the-Ridge Approach*. 2019. arXiv: 1910.07512 [cs.LG].

[67] Yinghao Xu et al. "Generative Hierarchical Features from Synthesizing Images". In: *CVPR*. 2021.

[68] Yasin Yaz et al. "The Unusual Effectiveness of Averaging in GAN Training". In: *International Conference on Learning Representations*. 2019.

[69] Fisher Yu et al. "Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop". In: *arXiv preprint arXiv:1506.03365* (2015).

[70] Han Zhang et al. "Self-attention generative adversarial networks". In: *International Conference on Machine Learning*. PMLR. 2019, pp. 7354–7363.

[71] Yuxuan Zhang et al. "Image {GAN}s meet Differentiable Rendering for Inverse Graphics and Interpretable 3D Neural Rendering". In: *International Conference on Learning Representations*. 2021. URL: https://openreview.net/forum?id=yWkP7JuHX1.

[72] Peiye Zhuang, Oluwasanmi O Koyejo, and Alex Schwing. "Enjoy Your Editing: Controllable {GAN}s for Image Editing via Latent Space Navigation". In: *International Conference on Learning Representations*. 2021. URL: https://openreview.net/forum?id=HOFxeCutxZR.

A Theoretical convergence of LeakyEMA

This section seeks to get a theoretical understanding of why applying the LeakyEMA technique improves GAN performance. Since the analysis of a general large-scale setup is intractable, we resort to a substantially simpler problem where:

- Distributions of generated p_G and real p_d data have disjoint supports;
- The discriminator D is optimized jointly with generator G, rather than trained until convergence.

Recently, [43] proposed a one-dimensional toy setup that satisfies both of those requirements and demonstrated that a variety of regularization techniques fail to achieve the desired solution on it. We use it as a motivating example to prove that models trained with the LeakyEMA trick improve convergence.

Definition 1. *The Dirac task is a problem where:*

1. *The input space $\mathcal{X}$ is $\mathbb{R}$;*
2. *Ground-truth distribution $\mathrm{p}_d(x)$ is $\delta_0(x)$;*
3. *Generator outputs a distribution $\mathrm{p}_G(x;\theta) = \delta_\theta(x)$;*
4. *Discriminator is defined by $D(x;\psi) = \psi x$.*

We also consider a common nonsaturating objective for training:

$$\begin{cases} \mathcal{L}_G(\theta;\psi) = \mathbb{E}_{\mathrm{p}_G(x;\theta)}\varphi(D(x;\psi)); \\ \mathcal{L}_D(\theta;\psi) = \mathbb{E}_{\mathrm{p}_G(x;\theta)}\varphi(-D(x;\psi)) + \mathbb{E}_{\mathrm{p}_d(x)}\varphi(D(x;\psi)); \end{cases} \tag{6}$$

where $\varphi(t) = -\log(\sigma(t)) = \log(1+\exp(-t))$ combines NLL objective with an output sigmoid nonlinearity[4]. Using Definition 1, those objectives convert to:

$$\begin{cases} \mathcal{L}_G(\theta;\psi) = \varphi(\psi\theta); \\ \mathcal{L}_D(\theta;\psi) = \varphi(-\psi\theta) + \varphi(0). \end{cases} \tag{7}$$

Additionally, let us define $\omega = (\theta, \phi)^T$ as a joint parameter space which dynamics we analyse. Note that, since $\forall t \ \varphi'(t) = -e^{-t}\sigma(t) < 0$, the Dirac task has a single possible Nash equilibrium $(\theta^*, \psi^*) = (0, 0)$.

Following [43], we study the linear (or exponential) stability of different methods around this point by switching from discrete to infinitesimal dynamics and linearizing the obtained differential equation on w around zero.

We start by revisiting the result of [43]:

[4]Note that $\varphi(t) = -t$ with Lipschitz constraint on D gives WGAN objective

Lemma 1. *A GAN with default nonsaturating objective on Dirac task is not linearly stable at the equilibrium* $(\theta^*, \psi^*) = (0, 0)$.

Proof. The standard simultaneous gradient descent on the objectives (7) gives updates of the form:

$$\omega_{t+1} = \omega_t + \eta v(\theta; \psi); \tag{8}$$

where $v(\theta; \psi) = (-\frac{\partial L_G(\theta;\psi)}{\partial \theta}, -\frac{\partial L_D(\theta;\psi)}{\partial \psi})^T$ is a gradient vector field, and η is a learning rate. In the limit of infinitesimally small t this gives the following differential equation:

$$\dot{\omega} = v(\theta; \psi) = \begin{pmatrix} -\varphi'(\psi\theta)\psi \\ \varphi'(-\psi\theta)\theta \end{pmatrix}. \tag{9}$$

By linearizing this system around equilibrium point $(0, 0)$ we get:

$$\dot{\omega} = \begin{pmatrix} -\varphi'(0)\psi \\ \varphi'(0)\theta \end{pmatrix} = \begin{pmatrix} 0 & -\varphi'(0) \\ \varphi'(0) & 0 \end{pmatrix} \omega. \tag{10}$$

Thus, the eigenvalues of its Jacobian are $\lambda_{1,2} = \pm\varphi'(0)i$ with $\mathrm{Re}(\lambda_{1,2}) = 0$, which implies the system is not linearly stable. □

As demonstrated by [43], similar results hold for both Default GAN, WGAN, and WGAN with Gradient Penalty.

Now we consider adding an exponential moving average of generator parameters θ_{ema} as a stabilization trick. Note that by definition $\theta^\gamma_{\text{ema}}(t) \equiv \gamma\theta^\gamma_{\text{ema}}(t-1) + (1-\gamma)\theta(t)$, which in continuous case $(1 \longrightarrow ds)$ naturally converts to:

$$\theta^\gamma_{\text{ema}}(t) \equiv (1-\gamma)\int_0^t \gamma^{t-s}\theta(s)ds. \tag{11}$$

Let $\xi \equiv \theta^\gamma_{\text{ema}}(t)/(1-\gamma) = \int_0^t \gamma^{t-s}\theta(s)ds$. Then it's time derivative is:

$$\dot{\xi} = \dot{\gamma^t}\int_0^t \gamma^{-s}\theta(s)ds + \gamma^t\gamma^{-t}\theta(t) = \ln(\gamma)\gamma^t\int_0^t \gamma^{-s}\theta(s)ds + \theta(t) = \ln(\gamma)\xi(t) + \theta(t). \tag{12}$$

Recall that training with a LeakyEMA approach means switching from a ground truth distribution $p_d(x)$ to a "regularized" distribution:

$$\widetilde{\mathrm{p}}_d(x) = \beta\mathrm{p}_G(x; \theta^\gamma_{\text{ema}}) + (1-\beta)\mathrm{p}_d(x). \tag{13}$$

In the case of Dirac GAN, this means that the original objectives (7) convert into:

$$\begin{cases} \mathcal{L}_G^{\text{Leaky}}(\theta; \psi) = \varphi(\psi\theta); \\ \mathcal{L}_D^{\text{Leaky}}(\theta; \psi) = \varphi(-\psi\theta) + (1-\beta)\varphi(0) + \beta\varphi(\psi\theta^\gamma_{\text{ema}}). \end{cases} \tag{14}$$

We are ready to state our main result.

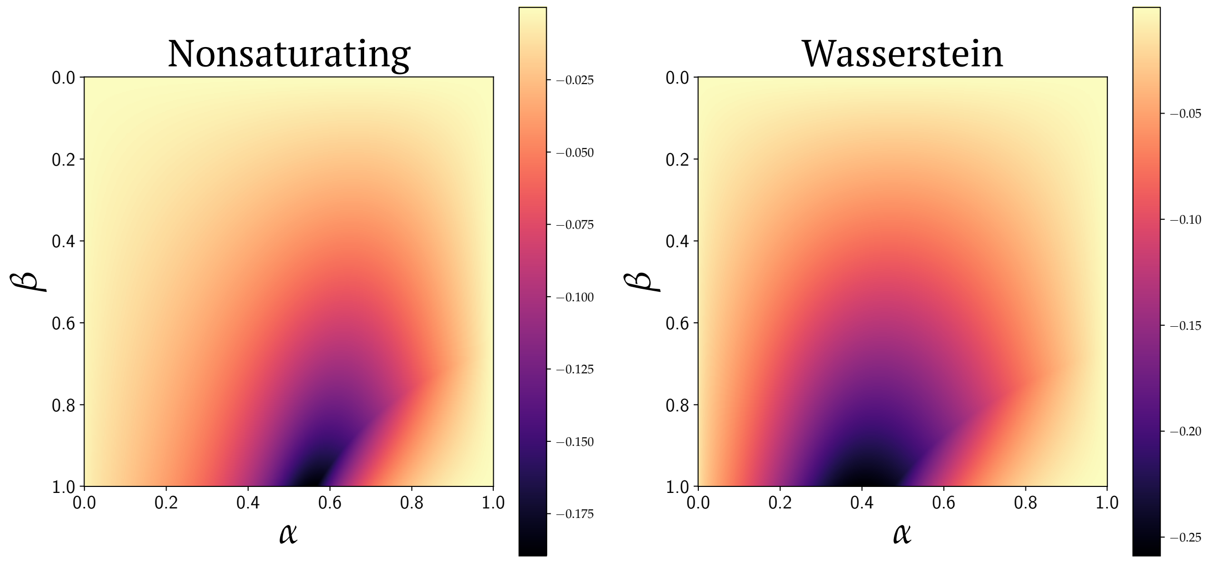

Figure 7: Maximum real components for roots of characteristic polynomial (17) with nonsaturating (left) and Wasserstein (right) objectives. We ablate 10^6 combinations of β and γ.

Lemma 2. *A GAN trained with a regularized nonsaturating objectives (14) is linearly stable at* $(0,0)$.

Proof. Consider an expanded parameter space $\tilde{\omega} = (\theta, \psi, \theta^{\gamma}_{\text{ema}})^T$. Then, based on Lemma 1 and equation (12) the gradient vector field of the system (14) is:

$$v(\tilde{w}) = \begin{pmatrix} -\varphi'(\psi\theta)\psi \\ \varphi'(-\psi\theta)\theta \\ 0 \end{pmatrix} + \rho(\tilde{w}), \text{ where } \rho(\tilde{w}) \equiv \begin{pmatrix} 0 \\ -\beta\varphi'(\psi\theta^{\gamma}_{\text{ema}})\theta^{\gamma}_{\text{ema}} \\ (1-\gamma)\theta + \ln(\gamma)\theta^{\gamma}_{\text{ema}} \end{pmatrix}. \tag{15}$$

Thus, around the equilibrium point $(0,0)$ linearized dynamics is:

$$\dot{\tilde{w}} = \begin{pmatrix} 0 & -\varphi'(0) & 0 \\ \varphi'(0) & 0 & -\beta\varphi'(0) \\ (1-\gamma) & 0 & \ln(\gamma) \end{pmatrix} \tilde{w}; \tag{16}$$

with the characteristic polynomial:

$$\det(A - \lambda I) = -\lambda^3 + \ln(\gamma)\lambda^2 - \varphi'(0)^2\lambda + \varphi'(0)^2(\ln(\gamma) + (1-\gamma)\beta). \tag{17}$$

We notice that it does not have any simple roots, so we solve it numerically with different values of β and γ. In Figure 7 we observe maximum real components for the obtained eigenvalues and see that with $\beta > 0$, they are negative, which concludes the proof.

□