Mean Teachers Can Train Superior GANs

Sergey Chervontsev [1 2] **Andrey Voynov** [1] **Artem Babenko** [1 2]

Abstract

Weight averaging has become a standard component of the state-of-the-art generative adversarial networks, like BigGAN (Brock et al., 2018). The exponential moving average (EMA) of the generator's parameters typically produces samples of higher visual quality and with better mode coverage compared to non-averaged weights. However, this advantage of EMA is currently not fully exploited since averaged parameters are used only for evaluation and do not influence the GAN training process. In this work, we demonstrate that the EMA parameters can also guide the training, leading to significantly higher performance. In particular, we introduce a simple (three lines of code) model-agnostic modification of the GAN training protocol and empirically show that it improves generation quality on established benchmarks, including Imagenet. Furthermore, we explain how the proposed EMA training alleviates the disjoint supports problem and theoretically prove its local convergence on a simplified task.

1. Introduction

Nowadays, the state-of-the-art Generative Adversarial Networks (GANs) (Goodfellow et al., 2014) are able to produce diverse photorealistic images, what makes them an important ingredient in a wide range of applications, including image-to-image translation (Isola et al., 2017; Zhu et al., 2017), super-resolution (Ledig et al., 2017; Menon et al., 2020), video generation (Wang et al., 2018), editing via latent transformations (Zhu et al., 2016) and many others. Compared to alternative generative models (Kingma & Welling, 2014; Rezende & Mohamed, 2015), however, GANs are more challenging to train since their adversarial optimization process is prone to vanishing gradients, mode collapse, oscillating or cyclic behavior (Goodfellow, 2016).

Since the seminal paper on GANs (Goodfellow et al., 2014), a plethora of different methods to simplify GAN training have been proposed, including the development of stable architectures (Radford et al., 2015; Karras et al., 2018; Zhang et al., 2019a; Brock et al., 2018; Karras et al., 2020), alternative loss functions (Arjovsky et al., 2017; Nowozin et al., 2016; Mao et al., 2017), appropriate normalizations, regularizers, and other helpful tricks (Salimans et al., 2016; Arjovsky & Bottou, 2017; Gulrajani et al., 2017; Metz et al., 2017; Miyato et al., 2018; Sinha et al., 2020; Peng et al., 2019; Zhang et al., 2019b; Zhao et al., 2020; Chen et al., 2019; Wang et al., 2020). Among such tricks, using the exponential moving average (EMA) of generator's parameters (Gidel et al., 2019; Karras et al., 2018) is one of the most successful and is currently employed by state-of-the-art GANs, e.g., BigGAN (Brock et al., 2018), and StyleGAN (Karras et al., 2020).

In essence, EMA computes the exponentially weighted running average of the generator's weights over the training iterations. Despite its simplicity, EMA generators produce more realistic samples and provide better mode coverage (Yaz et al., 2019), compared to non-averaged generators. However, this advantage is not fully leveraged by existing training protocols since EMA does not affect them and is used only for evaluation.

Meanwhile, for discriminative models, the EMA parameters have already established themselves as an important source of self-supervision. For instance, in the semi-supervised setup, EMA can produce targets to train models on unlabeled data (Tarvainen & Valpola, 2017). For learning from noisy labels, EMA is used to filter out unreliable samples (Nguyen et al., 2019). Finally, very recently, the supervision from EMA was used to develop BYOL (Grill et al., 2020), a new state-of-the-art approach for self-supervised pretraining. In this paper, we show that EMA can guide GAN optimization as well.

In a nutshell, we propose to supervise the GAN training by both real images and EMA samples. To this end, in each training batch, we replace a portion of real images with images generated by the current EMA. This simple modification, which we refer to as *LeakyEMA*, can be universally applied to any GAN architecture or loss function and provides consistent improvements on the standard datasets, including BigGAN on Imagenet.

LeakyEMA is motivated by the prior observations that fake and real distributions often have non-overlapping supports,

[1]Yandex Research, Moscow, Russia
[2]Higher School of Economics, Moscow, Russia

which is a well-known problem in the GAN community (Sønderby et al., 2017; Arjovsky & Bottou, 2017). In this case, discriminators cannot produce reliable gradients, and the training process becomes slow and unstable. We argue that the problem of disjoint supports can be substantially mitigated by using EMA samples "to bridge" the real and fake distributions, which is illustrated on Figure 1. Under this perspective, LeakyEMA can be considered as a smart version of the instance noise trick (Sønderby et al., 2017; Arjovsky & Bottou, 2017), where EMA images effectively serve as noisy samples from the real distribution. While several previous works have successfully addressed the instability caused by disjoint supports, we show that LeakyEMA is complementary to them and provides significant gains even on top of the state-of-the-art BigGAN model.

From the theoretical standpoint, we analyze the local convergence of LeakyEMA by examining the Jacobian eigenvalues of the associated gradient vector field. On the illustrative Dirac-GAN task (Mescheder et al., 2018), we prove that all Jacobian eigenvalues have negative real-part at the Nash-equilibrium, therefore, LeakyEMA is locally convergent.

To sum up, the contributions of our paper are the following:

1. We propose LeakyEMA — a simple and universal regularizer that significantly improves the performance of the state-of-the-art GANs, which is confirmed by extensive experiments on a wide range of datasets.

2. We explain the connection between LeakyEMA, the disjoint supports problem, and the instance noise trick. In particular, LeakyEMA can be treated as an advanced variant of instance noise with higher sample-efficiency.

3. We theoretically prove the local convergence of training with LeakyEMA on a simplified task.

2. Related work

In this section, we position our work in the context of the relevant literature.

Stable GAN training is a long-standing challenge for the computer vision community. Over the years, it has been addressed from several sides, including both theoretical and empirical efforts. A fruitful research line develops GAN architectures that allow for more stable optimization (Radford et al., 2015; Karras et al., 2018; Zhang et al., 2019a; Brock et al., 2018; Karras et al., 2019; 2020). Other works investigate alternative loss functions for generators and discriminators (Arjovsky et al., 2017; Nowozin et al., 2016; Mao et al., 2017; Li et al., 2017). However, this research direction has been shown to be less promising since the recent large-scale comparison (Lucic et al., 2018) has demonstrated that under fair comparison, all loss functions result in similar performance. Finally, a large amount of normalizations, regularizers, and other heuristics have been proposed (Salimans et al., 2016; Arjovsky & Bottou, 2017; Gulrajani et al., 2017; Miyato et al., 2018; Metz et al., 2017; Tolstikhin et al., 2017; Peng et al., 2019; Tao & Wang, 2020).

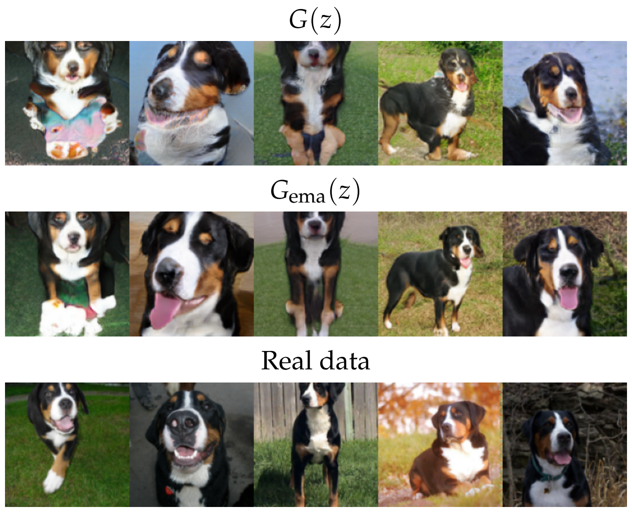

Figure 1. Samples produced by the BigGAN generator on the $100k$-th training step (top), samples produced by the EMA generator on the same step (middle), real samples from Imagenet (bottom). The EMA samples are more realistic and provide a natural "bridge" between the real and generator's distribution.

Disjoint supports problem is an important insight on the instability of GANs (Sønderby et al., 2017; Arjovsky & Bottou, 2017). When the supports of the data and model distributions do not overlap, the discriminator can achieve perfect classification, which results in vanishing gradients for the generator. (Sønderby et al., 2017; Arjovsky & Bottou, 2017) alleviate this problem by using the instance noise trick, which "spreads out" the distributions via adding continuous Gaussian noise to both real and fake samples. Such a solution results in a meaningful overlap between the supports, however, since the noise is isotropic, most of noisy samples do not belong to this overlap, and one requires large batches to obtain a sufficient amount of "hard" samples for the discriminator. In our paper, we propose a more sample-efficient way to avoid disjoint supports by using EMA samples "to connect" the data and model distributions. Note that previous works also investigate alternative approaches to prevent the discriminator from perfect classification, e.g., via additional penalty terms or normalizations. In experiments, we confirm their complementarity to our approach by showing LeakyEMA to provide an additional performance boost even on top of the established stabilizing techniques.

Weight averaging in GANs has been investigated only recently, from both theoretical (Gidel et al., 2019; Yaz et al., 2019) and empirical perspectives (Karras et al., 2018). At the moment, it is employed by the state-of-the-art models, e.g., BigGAN and StyleGAN2, which use exponential moving average (EMA) generator's parameters for evaluation. However, to the best of our knowledge, EMA samples are not currently used to guide the training process by any means, and we address this gap in our paper.

EMA training of discriminative models. Very recently, the EMA parameters were shown to provide useful supervision signal to train discriminative models (Tarvainen & Valpola, 2017; Nguyen et al., 2019; Grill et al., 2020). For instance, in the context of semi-supervised learning, enforcing the consistency of predictions with the weight-average models significantly improves classification accuracy (Tarvainen & Valpola, 2017). BYOL (Grill et al., 2020), a very recent method, also exploiting the idea of mean teacher prediction, was shown to provide the state-of-the-art performance of the self-supervised pretraining. Overall, the idea that the models, which parameters are averaged over previous epochs, can reliably guide the training process appears to be successful. In this paper, we show that it shines in the context of GAN training as well.

3. Method

3.1. Preliminaries

Let us consider a GAN consisting of a generator G and a discriminator D. As its input, G receives samples from a prior distribution $z \sim p(z)$ and aims to approximate the target distribution $p_d(x)$. On the other hand, D is trained to discriminate between the samples from $p_d(x)$ and the samples produced by G. G and D are trained jointly in an adversarial fashion. More formally, the GAN training process performs a joint optimization of two objectives:

$$\begin{aligned} \theta_D^* &= \arg\min_{\theta_D} L_D(p_d(x), p_G(x; \theta_G^*); \theta_D); \\ \theta_G^* &= \arg\min_{\theta_G} L_G(p_d(x), p_G(x; \theta_G); \theta_D^*); \end{aligned} \tag{1}$$

where L_D is an objective function designed to improve the ability of D to discriminate between real and fake images, while L_G is designed to achieve the opposite. Since it is difficult to satisfy both conditions simultaneously, the training typically proceeds in an alternating fashion by performing several gradient steps on L_D followed by gradient steps on L_G. Our method does not require any specific design for L_D or L_G and can be used with any existing objectives (Goodfellow et al., 2014; Arjovsky et al., 2017; Mao et al., 2017).

3.2. Exponential Moving Average of Weights

An additional trick that is widely used in practice is to accumulate the weights of generator G during training with

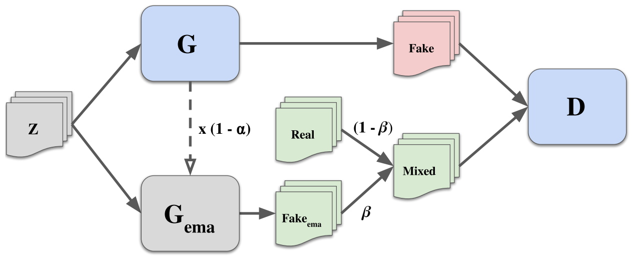

Figure 2. Training with LeakyEMA. The trainable components are denoted by blue, the images labeled as "real" are denoted by green.

an exponential moving average, i.e.

$$\theta_G^{\text{EMA}} \leftarrow \alpha\theta_G^{\text{EMA}} + (1-\alpha)\theta_G. \tag{2}$$

Then on the inference stage, one can use these accumulated weights θ_G^{EMA} to generate fake samples. This simple modification gives a significant boost in performance on top of the state-of-the-art models (Brock et al., 2018; Karras et al., 2020), which is attributed to the widely known instability and oscillating behavior of adversarial training, that is still present in up-to-date models (Berard et al., 2020).

3.3. From the evaluation-only EMA to Mean Teacher

The superior quality of θ_G^{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:

$$\begin{aligned} \theta_D^* &= \arg\min_{\theta_D} L_D(\tilde{p}_d(x), p_G(x; \theta_G^*); \theta_D); \\ \theta_G^* &= \arg\min_{\theta_G} L_G(\tilde{p}_d(x), p_G(x; \theta_G); \theta_D^*); \end{aligned} \tag{3}$$

where $\tilde{p}_d(x)$ is a mixture of two distributions:

$$\tilde{p}_d(x) = \beta p_{G_{\text{EMA}}}(x; \theta_G^{\text{EMA}}) + (1-\beta)p_d(x). \tag{4}$$

where 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 to as **LeakyEMA**. Training with LeakyEMA is schematically presented on Figure 2 and can be easily implemented with a few lines of code. Note that on each training step, the same noise samples z are used for both G and G_{EMA}. In experiments, we observed that for large-scale tasks, it is necessary to use LeakyEMA on the later epochs, 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. We ablate on both β and N and provide recipes on setting them in the experimental section.

3.4. Computational complexity

Since LeakyEMA requires several EMA samples for each training batch, each optimization step has a minor computational overhead compared to the conventional GAN training. Note, however, that this overhead includes 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 on the latter iterations, while the first N iterations have the same complexity. Overall, in all our experiments, the increase in total training time is negligible and was fully justified by the superior generation performance.

3.5. LeakyEMA vs Instance Noise

In this section, we explain the relationship between the proposed LeakyEMA training and the well-known instance noise (IN) trick (Sønderby et al., 2017; Arjovsky & Bottou, 2017). 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, in slow and unstable optimization. IN mitigates this problem by adding isotropic Gaussian noise to both real and fake images, which effectively creates a noticeable overlap of the "spreaded" distributions, containing samples that are difficult for the discriminator. However, in high-dimensional spaces, only a small amount of noisy samples fall into this overlap due to the isotropy of noise, therefore, most of the samples still will be "easy" for the discriminator. In contrast, LeakyEMA is based on the previous evidence (Karras et al., 2018; Brock et al., 2018; Karras et al., 2019) that EMA samples are more appealing compared to the generator samples (see Figure 1). In other words, in terms of realism, EMA samples are "intermediate" between real and fake images, therefore, it is natural to use them to create the overlap between distributions.

To confirm the intuition above, we consider the following synthetic task that represents a disjoint supports setup. We set both real and generated distributions to be multidimensional Gaussians in $\mathbb{R}^{32}$ with uniform covariance matrices. The real data distribution is the Gaussian with $p_{\text{data}} = \mathcal{N}(0, 0.1 \cdot I)$, while the generated data distribution is set to be $p_G = \mathcal{N}(\mu, 0.1 \cdot I)$ with the learnable generator parameter μ. The discriminator model is defined by a linear map $D(x) = \langle x, \xi \rangle + b$ with parameters ξ and b. We use the standard GAN loss (Goodfellow et al., 2014) $L_G = \mathbb{E}_{x\sim p_G} \log(1 - \sigma(D(x)))$, $L_D = -\mathbb{E}_{x\sim p_G} \log(1-\sigma(D(x)))-\mathbb{E}_{x\sim p_{\text{data}}} \log(\sigma(D(x)))$ where σ is the sigmoid function. In three of the four runs below we use a training batch size 2. The optimization is performed with the four following setups:

- *Default training* with no instance noise or LeakyEMA;
- *Instance Noise* with the starting noise variance equal 3. During the training, the variance anneals linearly to 0;
- *Instance Noise with a huge batch* equal to 1024 and all other parametrs the same as above;
- *LeakyEMA* with $\beta = 0.5$ and $N = 0$.

In all setups, we initialize the generated distribution with $\mu_0 = (1, \dots, 1)^T$. The discriminator parameters are picked from the normal distribution with the same values for all runs. EMA is computed with a coefficient equals 0.99. We perform the adversarial training with 5 discriminator steps per generator step in SGD optimization protocol with a learning rate 0.05. In all the runs, we perform 5000 generator optimization steps. For the Instance Noise, we report the run with the hyperparameters that demonstrated the best score from the parameters grid-search. The training curves are presented on Figure 3 (left), which demonstrates that both default and IN training are slower compared to LeakyEMA. Figure 3 (right) provides an explanation of this behavior by demonstrating the distributions of the discriminator outputs for all setups on a particular training iteration. For both default and Instance Noise training, most of the generator samples are easily "rejected" by D, which results in vanishing gradients for G. In contrast, with our approach, D is less confident, indicating that LeakyEMA is a more efficient way to prevent D from perfect classification.

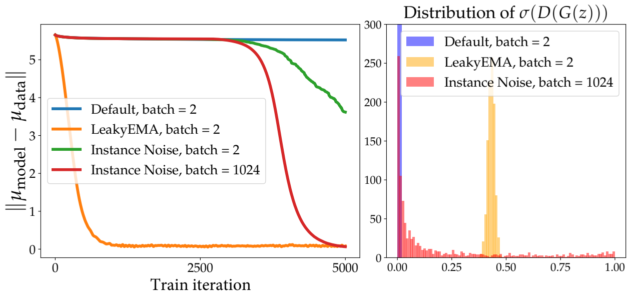

Figure 3. Left: norms of the learnable generated distribution mean μ for different setups. *Right*: probabilities distributions of the generated samples $\sigma(D(G(z)))$ on the step 200.

3.6. Local Convergence with LeakyEMA

For theoretical understanding of the regularization effect stemming from LeakyEMA, we follow (Mescheder et al., 2018) 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 demonstrate theoretically that LeakyEMA is locally convergent. In particular, we prove that all Jacobian eigenvalues of the field have negative real-part in the neighborhood of the Nash-equilibrium. Full proofs are provided in supplementary.

4. Experiments

In this section, we evaluate LeakyEMA on a large number of common benchmarks. First, we investigate its sensitivity to hyperparameters and perform a thorough ablation analysis. Second, we demonstrate that LeakyEMA can be successfully combined with established stabilization techniques. Finally, we show that using LeakyEMA on top of the BigGAN model (Brock et al., 2018) provides a new state-of-the-art generation quality on Imagenet. For quantitative evaluation, we use the Frechet Inception Distance (FID) (Heusel et al., 2017) computed on the image representations extracted from the InceptionV3 model[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 (Miyato et al., 2018) architecture, which generates 128×128 images, originally proposed in (Miyato et al., 2018) (see table $6a$) . We use N_{dis}=5 discriminator updates per single generator step. The optimization is performed by Adam 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 the conventional way for N steps with a Hinge variant of adversarial training objective (Miyato et al., 2018) 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.

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 (Yu et al., 2015) is provided in Table 1. 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 1 are highlighted below:

- 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 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 our experiments, $\beta = 0.5$ appears to be a "sweet spot" in terms of FID; therefore, we use this value in the following experiments.

- In Table 1, 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 figure 4. 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.

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	<u>4.15$\pm$ 0.31</u>	5.01$\pm$ 0.61
$120k$	7.94$\pm$ 2.40	5.25$\pm$ 0.47	<u>4.17$\pm$ 0.24</u>	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 1. Performance of SNGAN trained on LSUN-Church dataset (128x128) with different values of β and N. β=0 corresponds to training without LeakyEMA, therefore, numbers in the second column are the same.

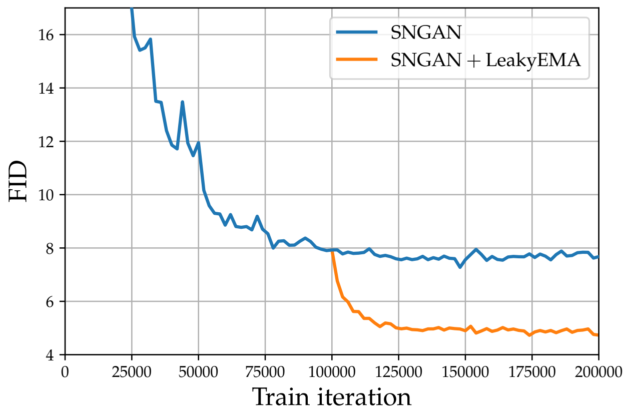

Figure 4. Comparison of FID learning curves for the conventional training and the training with LeakyEMA on the LSUN-Church with $\beta = 0.5$ and $N = 100K$.

The learning curves in terms of FID for both the conventional GAN training and the LeakyEMA training with

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

	Baseline (no EMA)	EMA sync	EMA async-noise	EMA async-model
FID	7.93	4.15	4.16	5.85

Table 2. Performance of different versions of LeakyEMA of SNGAN on the LSUN-Church dataset.

β=0.5, N=100K are presented on Figure 4. Note that the learning curve with LeakyEMA has smaller FID fluctuations, implying more stable training.

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:

1. **EMA-sync**, a fully synchronized version, where training is guided by the samples from the current EMA. 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 pretrained (fixed) EMA is used to guide the training on all iterations.

The performance of three different versions is reported in Table 2, which demonstrates that the asynchronous **EMA-async-model** protocol is inferior in terms of FID. These results highlight the importance of 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 simplicity of implementation.

Are EMA samples necessary? To verify that EMA samples in our approach cannot be replaced by the samples from the generator itself, we perform the following experiment. Instead of using EMA samples to replace a portion of a real images, we tried to use (a) random generator samples (b) generator samples that are "the most realistic" for the current discriminator. We use ResNet-SNGAN on LSUN-Church with N=100K and β=0.5. With both (a) and (b), the performance was equal or inferior to the ResNet-SNGAN trained in a conventional manner. This experiment provides additional evidence that the superiority of EMA over the generator is a key ingredient of the LeakyEMA success.

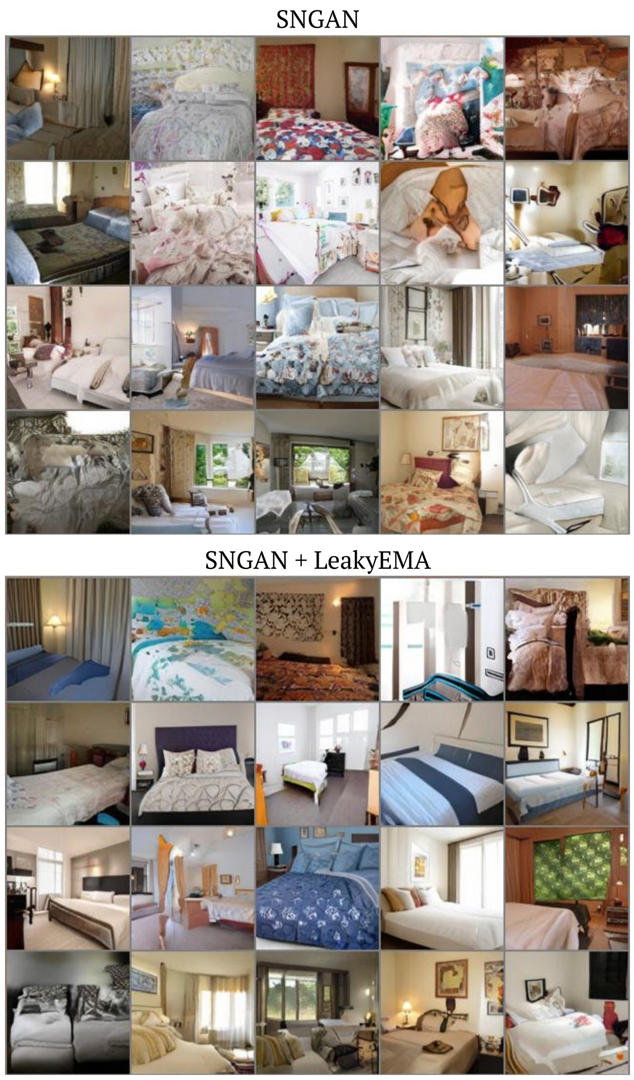

Figure 5. Samples of SNGAN trained on LSUN-Bedroom in conventional (top) and LeakyEMA protocols (bottom). We use $N = 80K$ and β=0.5. Here we present samples from the best checkpoints obtained from both protocols and use the same latent noise samples.

4.2. LeakyEMA and other regularizers

In this experiment, we show that LeakyEMA can be combined with established stabilization techniques. In particular, we evaluate the training with LeakyEMA using the WGAN-

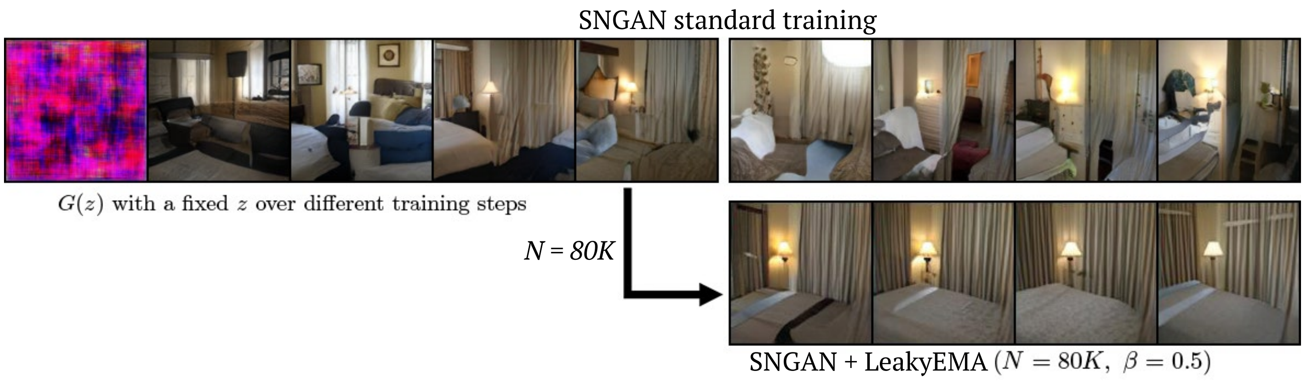
SNGAN standard training
$G(z)$ with a fixed z over different training steps
$N = 80K$
SNGAN + LeakyEMA ($N = 80K,\ \beta = 0.5$)

Figure 6. Evolution of a generated sample $G(z)$ with a fixed z. *Top row*: conventional training; *bottom row*: EMA training, which reduces the sample variation over the training steps, while improving realism.

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 3. Training with LeakyEMA combined with previous stabilization techniques.

GP (Gulrajani et al., 2017) and the SNGAN (Miyato et al., 2018) models. For a fair comparison, we use the same model architecture as described in Section 4.1. The exact hyperparameter values are provided in supplementary. We compare the models on three datasets from different domains. In this experiment, all images were resized to 128×128.

1. **LSUN-Church** (Yu et al., 2015) containing outdoor scene images with multiple details, challenging to generate properly.

2. **LSUN-bedroom** (Yu et al., 2015), containing indoor images of bedrooms.

3. **FFHQ** (Karras et al., 2019), containing human face images.

Comparison of WGAN-GP and SNGAN with the conventional and LeakyEMA training on three datasets is presented in Table 3. For all experiments, we choose N to be the iteration corresponding to the lowest FID obtained in the conventional protocol, 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. Figure 5 compares uncurated SNGAN samples from the models trained with conventional and LeakyEMA training protocols. The same latent codes are used for both models.

On Figure 6, we illustrate the evolution of a particular sample during the conventional training and during the LeakyEMA training (with $N{=}80K$ and $\beta{=}0.5$) 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. To confirm the stabilization effect quantitatively, we perform the following. For both conventional and LeakyEMA training, we compute the LPIPS distance (Zhang et al., 2018) between the fake images corresponding to the same latent code z produced by generators from the consecutive training iterations. LPIPS values averaged over 5000 samples are shown on Figure 7, which illustrates "smoother" image evolution stemming from LeakyEMA.

4.3. BigGAN

This section confirms that LeakyEMA can be used on top of the state-of-the-art BigGAN generator (Brock et al., 2018) and significantly improves its generation quality on the challenging Imagenet benchmark. In this experiment, we use the publicly available PyTorch implementation of BigGAN[2]. Figure 8 illustrates the learning curves of the BigGAN training with and without LeakyEMA. BigGAN+LeakyEMA significantly outperforms the conventional training, achieving

[2]https://github.com/ajbrock/BigGAN-PyTorch

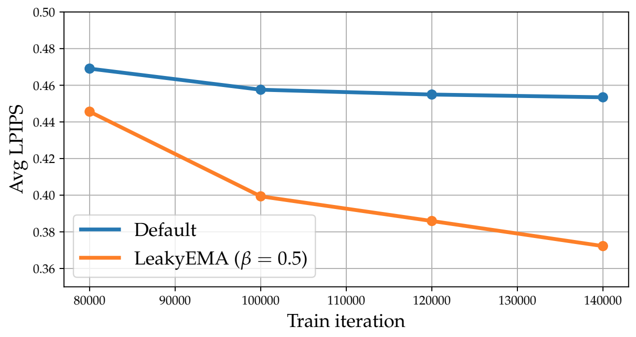

Figure 7. Evolution of generated images with fixed z for SNGAN trained on Bedroom. Average LPIPS distance between 5000 samples generated from the same latent codes on the consecutive training iterations.

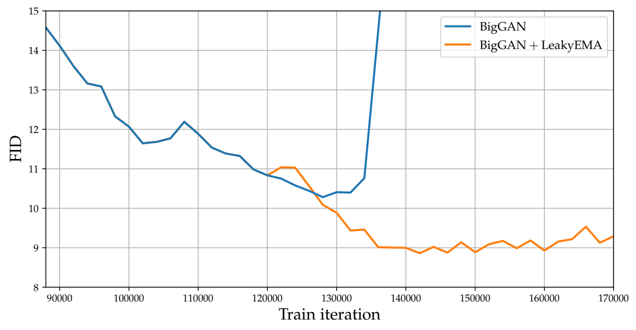

Figure 8. The learning curves of BigGAN training on the Imagenet dataset. Comparison of the conventional training and the LeakyEMA training for with $\beta = 0.75$ and $N = 120K$.

a new state-of-the-art of FID=8.86. Notably, the learning curve of the conventional training demonstrates the collapsing behavior, a well-known issue of BigGAN. In contrast, LeakyEMA results in more stable training progress, achieving much lower FID values and avoiding the collapse.

We compare our best configuration with the state-of-the-art checkpoint provided by the BigGAN authors[3] in Table 6. Along with FID, we use Precision and Recall metrics (Kynkäänniemi et al., 2019) computed with embeddings from Imagenet-pretrained VGG. The BigGAN trained with LeakyEMA achieves significantly higher Recall, indicating that our LeakyEMA improves the model coverage.

For a more detailed comparison of two BigGAN models, we plot the Precision-Recall curves in Figure 9 obtained by varying a truncation value in a range $[0.3, 1.0]$ (Brock et al., 2018), which is a standard way to quantify the quality-diversity tradeoff. The BigGAN+LeakyEMA outperforms the standard BigGAN across all operating points.

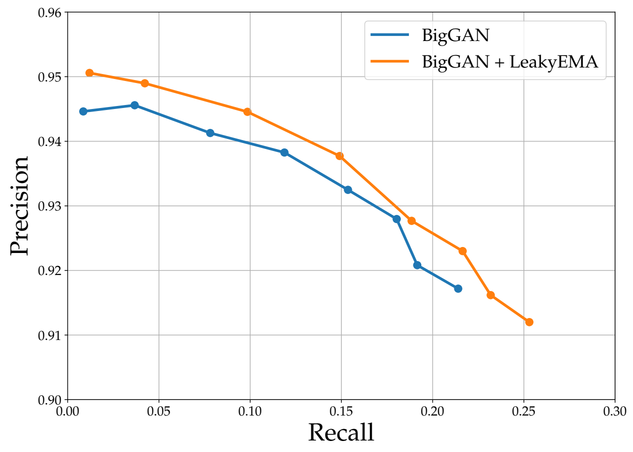

Figure 9. The Precision-Recall curves from varying truncation values linearly from 0.3 to 1.0 for the best model obtained with LeakyEMA and the publicly available BigGAN.

	FID	Precision	Recall
Imagenet 128×128			
BigGAN	10.14	0.92	0.21
BigGAN+LeakyEMA	9.20	0.91	0.24

Table 4. Comparison of the model trained with LeakyEMA and the publicly available state-of-the-art BigGAN checkpoint.

[3] https://drive.google.com/file/d/1nAle7FCVFZdix2--ks0r5JBkFnKw8ctW/view

5. Conclusion

Despite the recent progress on stabilizing GANs, the training of large-scale models on complex multi-modal datasets, e.g., Imagenet, is still challenging. In this work, we have proposed a simple modification of the GAN training protocol that leverages an additional supervision signal from the generator parameters averaged over previous optimization steps. Being both simple and efficient, our technique results 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

Arjovsky, M. and Bottou, L. Towards principled methods for training generative adversarial networks. *ICLR*, 2017.

Arjovsky, M., Chintala, S., and Bottou, L. Wasserstein generative adversarial networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 214–223, 2017.

Berard, H., Gidel, G., Almahairi, A., Vincent, P., and Lacoste-Julien, S. 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.

Brock, A., Donahue, J., and Simonyan, K. Large scale gan training for high fidelity natural image synthesis. In *International Conference on Learning Representations*, 2018.

Chen, T., Zhai, X., Ritter, M., Lucic, M., and Houlsby, N. Self-supervised gans via auxiliary rotation loss. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 12154–12163, 2019.

Gidel, G., Berard, H., Vignoud, G., Vincent, P., and Lacoste-Julien, S. A variational inequality perspective on generative adversarial networks. In *International Conference on Learning Representations*, 2019.

Goodfellow, I. Nips 2016 tutorial: Generative adversarial networks. *Advances in Neural Information Processing Systems*, 2016.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. Generative adversarial nets. In *Advances in neural information processing systems*, 2014.

Grill, J.-B., Strub, F., Altché, F., Tallec, C., Richemond, P. H., Buchatskaya, E., Doersch, C., Pires, B. A., Guo, Z. D., Azar, M. G., et al. Bootstrap your own latent: A new approach to self-supervised learning. *NeurIPS*, 2020.

Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., and Courville, A. C. Improved training of wasserstein gans. In *Advances in neural information processing systems*, pp. 5767–5777, 2017.

Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., and Hochreiter, S. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *Advances in Neural Information Processing Systems*, pp. 6626–6637, 2017.

Isola, P., Zhu, J.-Y., Zhou, T., and Efros, A. A. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017.

Karras, T., Aila, T., Laine, S., and Lehtinen, J. Progressive growing of gans for improved quality, stability, and variation. In *International Conference on Learning Representations*, 2018.

Karras, T., Laine, S., and Aila, T. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4401–4410, 2019.

Karras, T., Laine, S., Aittala, M., Hellsten, J., Lehtinen, J., and Aila, T. Analyzing and improving the image quality of stylegan. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8110–8119, 2020.

Kingma, D. P. and Welling, M. Auto-encoding variational bayes. In Bengio, Y. and LeCun, Y. (eds.), *2nd International Conference on Learning Representations, ICLR 2014*, 2014.

Kynkäänniemi, T., Karras, T., Laine, S., Lehtinen, J., and Aila, T. Improved precision and recall metric for assessing generative models. In *Advances in Neural Information Processing Systems*, pp. 3929–3938, 2019.

Ledig, C., Theis, L., Huszár, F., Caballero, J., Cunningham, A., Acosta, A., Aitken, A., Tejani, A., Totz, J., Wang, Z., et al. Photo-realistic single image super-resolution using a generative adversarial network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017.

Li, C.-L., Chang, W.-C., Cheng, Y., Yang, Y., and Póczos, B. Mmd gan: Towards deeper understanding of moment matching network. In *Advances in Neural Information Processing Systems*, pp. 2203–2213, 2017.

Lucic, M., Kurach, K., Michalski, M., Gelly, S., and Bousquet, O. Are gans created equal? a large-scale study. In *Advances in neural information processing systems*, pp. 700–709, 2018.

Mao, X., Li, Q., Xie, H., Lau, R. Y., Wang, Z., and Paul Smolley, S. Least squares generative adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pp. 2794–2802, 2017.

Menon, S., Damian, A., Hu, S., Ravi, N., and Rudin, C. Pulse: Self-supervised photo upsampling via latent space exploration of generative models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2437–2445, 2020.

Mescheder, L., Nowozin, S., and Geiger, A. Which training methods for gans do actually converge? In *International Conference on Machine Learning (ICML)*, 2018.

Metz, L., Poole, B., Pfau, D., and Sohl-Dickstein, J. Unrolled generative adversarial networks. *ICLR*, 2017.

Miyato, T., Kataoka, T., Koyama, M., and Yoshida, Y. Spectral normalization for generative adversarial networks. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=B1QRgziT-.

Nguyen, D. T., Mummadi, C. K., Ngo, T. P. N., Nguyen, T. H. P., Beggel, L., and Brox, T. Self: Learning to filter noisy labels with self-ensembling. In *International Conference on Learning Representations*, 2019.

Nowozin, S., Cseke, B., and Tomioka, R. f-gan: Training generative neural samplers using variational divergence minimization. In *Advances in neural information processing systems*, pp. 271–279, 2016.

Peng, X. B., Kanazawa, A., Toyer, S., Abbeel, P., and Levine, S. Variational discriminator bottleneck: Improving imitation learning, inverse rl, and gans by constraining information flow. In *ICLR*, 2019.

Radford, A., Metz, L., and Chintala, S. Unsupervised representation learning with deep convolutional generative adversarial networks. *ICLR*, 2015.

Rezende, D. and Mohamed, S. Variational inference with normalizing flows. In *International Conference on Machine Learning*, pp. 1530–1538, 2015.

Salimans, T., Goodfellow, I., Zaremba, W., Cheung, V., Radford, A., and Chen, X. Improved techniques for training gans. In *Advances in neural information processing systems*, pp. 2234–2242, 2016.

Sinha, S., Zhang, H., Goyal, A., Bengio, Y., Larochelle, H., and Odena, A. Small-gan: Speeding up gan training using core-sets. *ICML*, 2020.

Sønderby, C. K., Caballero, J., Theis, L., Shi, W., and Huszár, F. Amortised MAP inference for image super-resolution. In *ICLR*, 2017.

Tao, S. and Wang, J. Alleviation of gradient exploding in gans: Fake can be real. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

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

Tolstikhin, I. O., Gelly, S., Bousquet, O., Simon-Gabriel, C.-J., and Schölkopf, B. Adagan: Boosting generative models. In *Advances in Neural Information Processing Systems*, pp. 5424–5433, 2017.

Wang, J., Zhou, W., Qi, G.-J., Fu, Z., Tian, Q., and Li, H. Transformation gan for unsupervised image synthesis and representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 472–481, 2020.

Wang, T.-C., Liu, M.-Y., Zhu, J.-Y., Liu, G., Tao, A., Kautz, J., and Catanzaro, B. Video-to-video synthesis. In *Advances in Neural Information Processing Systems*, 2018.

Yaz, Y., Foo, C.-S., Winkler, S., Yap, K.-H., Piliouras, G., Chandrasekhar, V., et al. The unusual effectiveness of averaging in gan training. In *International Conference on Learning Representations*, 2019.

Yu, F., Seff, A., Zhang, Y., Song, S., Funkhouser, T., and Xiao, J. Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop. *arXiv preprint arXiv:1506.03365*, 2015.

Zhang, H., Goodfellow, I., Metaxas, D., and Odena, A. Self-attention generative adversarial networks. In *International Conference on Machine Learning*, pp. 7354–7363. PMLR, 2019a.

Zhang, H., Zhang, Z., Odena, A., and Lee, H. Consistency regularization for generative adversarial networks. In *International Conference on Learning Representations*, 2019b.

Zhang, R., Isola, P., Efros, A. A., Shechtman, E., and Wang, O. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 586–595, 2018.

Zhao, Z., Singh, S., Lee, H., Zhang, Z., Odena, A., and Zhang, H. Improved consistency regularization for gans. *arXiv preprint arXiv:2002.04724*, 2020.

Zhu, J.-Y., Krähenbühl, P., Shechtman, E., and Efros, A. A. Generative visual manipulation on the natural image manifold. In *European conference on computer vision*, pp. 597–613. Springer, 2016.

Zhu, J.-Y., Park, T., Isola, P., and Efros, A. A. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, 2017.

A. Theoretical convergence of LeakyEMA

To analyze the training with LeakyEMA theoretically, we consider the Dirac-GAN problem, a one-dimensional generative task proposed in (Mescheder et al., 2018). Namely, in the Dirac-GAN, the model distribution $p_G = \delta_\theta$ is concentrated at a single point, and the generator is parameterized by a scalar θ. The discriminator D is defined by a linear map $\theta \to \psi \cdot \theta$ with a learnable multiplier ψ. The data distribution is defined by the Dirac δ_0 function. The generator and discriminator are updated with alternating gradient steps by θ with respect to the generator loss L_G and by ψ with respect to the discriminator loss L_D. (Mescheder et al., 2018) proposed to analyze existing GAN regularization techniques in terms of the dynamics of such gradient updates when a gradient step tends to zero. In this case, the dynamics correspond to the first-order differential equation:

$$\begin{pmatrix} \dot{\theta} \\ \dot{\psi} \end{pmatrix} = \begin{pmatrix} -\nabla_\theta L_G \\ -\nabla_\psi L_D \end{pmatrix}. \quad (5)$$

Here we always write the dot as a derivative with respect to the parameter t. First, we define the exponential moving average (EMA) of a continuous function $\theta(t)$ with a decay factor $\alpha \in (0,1)$. Note that for the discrete case, EMA for a sequence $\theta_1, \dots, \theta_k$ is computed as

$$\begin{aligned} \theta_k^{\mathrm{EMA}} &= (1-\alpha)\cdot\theta_k + \alpha\cdot\theta_{k-1}^{\mathrm{EMA}} = \\ &= (1-\alpha)\cdot(\theta_k + \alpha\theta_{k-1}) + \alpha^2\cdot\theta_{k-2}^{\mathrm{EMA}} = \\ &\cdots = (1-\alpha)\cdot\sum_{i=1}^{k}\alpha^{k-i}\theta_k; \quad (6) \end{aligned}$$

where we set $\theta_1^{\mathrm{EMA}} = (1-\alpha)\cdot\theta_1$[4]. Following (6), we naturally define the EMA of the continuous trajectory as

$$\theta^{\mathrm{EMA}}(t) = (1-\alpha)\int_0^t \alpha^{t-s}\theta(s)ds. \quad (7)$$

[4]Since its impact on θ_k^{EMA} tends to 0 as k grows, this initialization is not important in practice.

For simplicity, below we focus on the nonsaturating objective functions L_G and L_D (Goodfellow et al., 2014)[5] :

$$\begin{aligned} L_G &= \varphi(\psi\theta); \\ L_D &= \varphi(-\psi\theta) + \varphi(\psi\cdot 0); \end{aligned} \quad (8)$$

with $\varphi(x) = -\log(\sigma(x)) = \log(1+e^{-x})$. The original work (Mescheder et al., 2018) showed that the Dirac-GAN (5) with this loss has a single equilibrium point $(\theta^*, \psi^*) = (0,0)$. Depending on the loss functions, solutions of (5) in the neighborhood of this equilibrium either do not converge or converge with a sub-linear rate. In this setup, the objectives appearing from LeakyEMA are of the form:

$$\begin{aligned} L_G^{\mathrm{EMA}} &= \varphi(\psi\theta); \\ L_D^{\mathrm{EMA}} &= \varphi(-\psi\theta) + \beta\cdot\varphi(\psi\theta^{\mathrm{EMA}}); \end{aligned} \quad (9)$$

where $\beta \in (0,1)$ is the fraction of real samples replaced by EMA samples. Here we omit the term $(1-\beta)\cdot\varphi(\psi\cdot 0)$ in the discriminator loss as it is constant and does not affect the gradients. The main result of this section is the following

Lemma 1. *All the integral curves in the neighborhood of the equilibrium point $(0,0)$ of the Dirac-GAN (5) with the LeakyEMA objectives (9) converge to this equilibrium point with a linear rate.*

Proof. First, we consider a new variable $u = \alpha^t \int_0^t \alpha^{-s}\theta(s)ds$ corresponding to the scaled EMA generator. Its derivative can be computed as

$$\begin{aligned} \dot{u}(t) = \alpha^t\cdot(\alpha^{-t}\theta(t)) + \log(\alpha)\cdot\alpha^t\cdot\int_0^t \alpha^{-s}\theta(s)ds = \\ = \theta(t) + \log\alpha\cdot u(t). \quad (10) \end{aligned}$$

Now let us consider an extended first-order system of differential equations for the Dirac-GAN:

$$\begin{pmatrix} \dot{\theta} \\ \dot{\psi} \\ \dot{u} \end{pmatrix} = \begin{pmatrix} -\nabla_\theta L_G^{\mathrm{EMA}} \\ -\nabla_\psi L_D^{\mathrm{EMA}} \\ \theta(t) + \log\alpha\cdot u(t) \end{pmatrix}. \quad (11)$$

Note that the point $(\theta^*, \psi^*, u^*) = (0,0,0)$ is the equilibrium point of this system. Furthermore, each solution of the original system (5) defines a solution of the extended system. Thus, it is sufficient to show that all the integral curves of the extended system converge to the equilibrium. We will show it by verifying that all the eigenvalues of the Jacobian of the vector field (11) have negative real parts.

[5]However, all the calculations below can be also performed for other existing objectives with minimal changes.

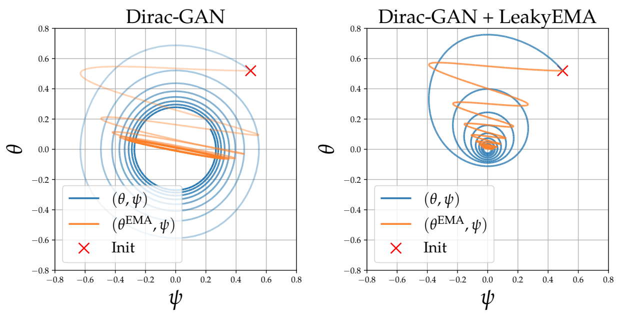

Figure 10. Comparison of the conventional and LeakyEMA protocols trajectories of Dirac-GAN. We depict the initial state, and the trajectories of (θ, ψ) and $(\theta^{\text{EMA}}, \psi)$.

We start with the first two components of (11). First, let us note that $\varphi'(x) = \frac{1}{1+e^x} = \frac{1}{2} - \frac{x}{4} + \overline{o}(x^2)$. Then we have

$$-\nabla_\theta L_G^{\text{EMA}} = \psi \cdot \varphi'(\psi\theta) = \psi \cdot (\frac{1}{2} - \mathcal{O}(\psi\theta)), \quad (12)$$

and similarly

$$\begin{aligned}&-\nabla_\psi L_D^{\text{EMA}} = \\ &= -\theta \cdot \varphi'(-\psi\theta) + \beta \cdot \theta^{\text{EMA}} \cdot \varphi'(\psi\theta^{\text{EMA}}) = \\ &= -\theta \cdot (\frac{1}{2} + \mathcal{O}(\psi\theta)) + \beta \cdot (1-\alpha) \cdot u \cdot \frac{1}{1+e^{\psi(1-\alpha)u}} = \\ &= -\frac{\theta}{2} + \beta \cdot (1-\alpha) \cdot u \cdot (\frac{1}{2} + \mathcal{O}(\psi u)). \quad (13)\end{aligned}$$

Combining (11), (12) and (13) we can write the Jacobian J of the extended system at the equilibrium point $(0, 0, 0)$:

$$J = \begin{pmatrix} 0 & \frac{1}{2} & 0 \\ -\frac{1}{2} & 0 & \frac{\beta\cdot(1-\alpha)}{2} \\ 1 & 0 & \log\alpha \end{pmatrix}. \quad (14)$$

This is a 3×3 matrix and we can explicitly find its eigenvalues in a closed-form solution. We omit the exact formula as it was computed within a symbolic solver and requires an enormous number of lines. For all the $\beta \in (0, 1)$ and $\alpha \in (0, 1)$ all real parts of its eigenvalues appear to be negative. □

Figure 16 demonstrates the dynamics of the conventional trajectories of the Dirac-GAN and the trajectories for LeakyEMA protocol, illustrating the statement of the Lemma 1.

On Figure 11 we visualize the maximal real part of the computed Jacobian eigenvalues for different parameters α and β. As Lemma 1 remains true for the Wasserstein loss, we also provide a similar plot for this objective.

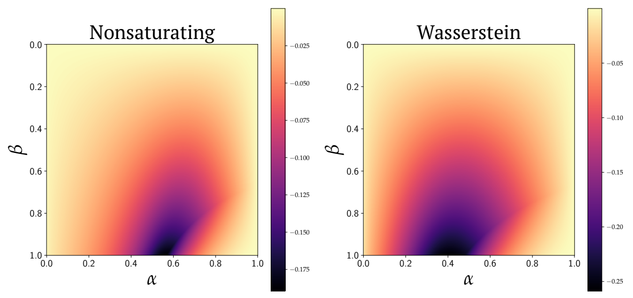

Figure 11. Maximum real parts of the Jacobian eigenvalues for the nonsaturating loss (left) and the Wasserstein loss (right).

We refer to Section 2.1 in (Mescheder et al., 2018) for further details on the relationship between continuous and discrete GAN dynamics and the Dirac-GAN model.

B. Models hyperparameters

Here we report the hyperparameters for the SNGAN and WGAN-GP models from Section 4.2. For both SNGAN and WGAN-GP, we use five discriminator steps per one generator step, and the generator and discriminator are optimized by Adam with batch size 64. SNGAN is optimized with a constant learning rate 2×10^{-4} and $\beta_1, \beta_2 = 0.5, 0.999$. WGAN-GP is optimized with a constant learning rate 10^{-4} and $\beta_1, \beta_2 = 0.5, 0.9$ and with a gradient penalty coefficient equal to ten. For WGAN-GP, we observed that the best LeakyEMA performance corresponds to β=0.25. The generator and discriminator architectures used for WGAN-GP are the networks described in tables $6a, b$ in (Miyato et al., 2018) without spectral normalization.

C. More qualitative results

Figure 12 and Figure 13 illustrate images generated from the same noise samples by SNGAN trained with the conventional and LeakyEMA training protocols.

Figure 14 illustrates the uncurated samples from the "Broccoli", "Volcano" and "Slot" classes produced by two BigGAN models. For these classes, the LeakyEMA training protocol results in the largest improvement of generation performance, as confirmed by the corresponding FID values. Notably, for these classes, the BigGAN with LeakyEMA provides much higher sample diversity. We also report the conditional precision and recall calculated for the selected classes only.

D. Other results

We train WGAN-GP with learning rate $1e-4$ and betas $(0.0, 0.9)$. See Table 5.

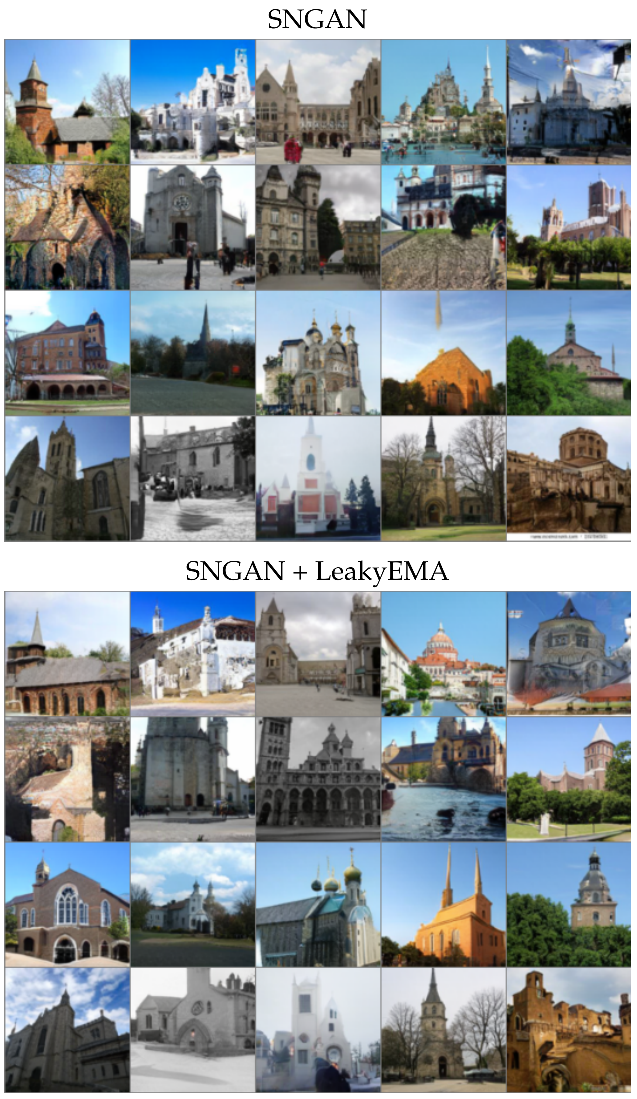

Figure 12. Uncurated samples of SNGAN trained on LSUN-Church with conventional (top) and LeakyEMA protocols (bottom). We use N=100K and β=0.5. Here we present samples from the best checkpoints obtained from both protocols and use the same latent noise samples.

Model	Church	Bedroom	FFHQ
WGAN-GP	12.16	7.72	—
WGAN-GP+LeakyEMA	**10.19**	**7.12**	—
SNGAN	7.93	13.48	14.80
SNGAN+Lookahead	6.06	—	—
SNGAN+LeakyEMA	**5.00**	**4.67**	11.87

Table 5. Comparison of SNGAN with alternative techniques (LeakyEMA on Church is still decreasing).

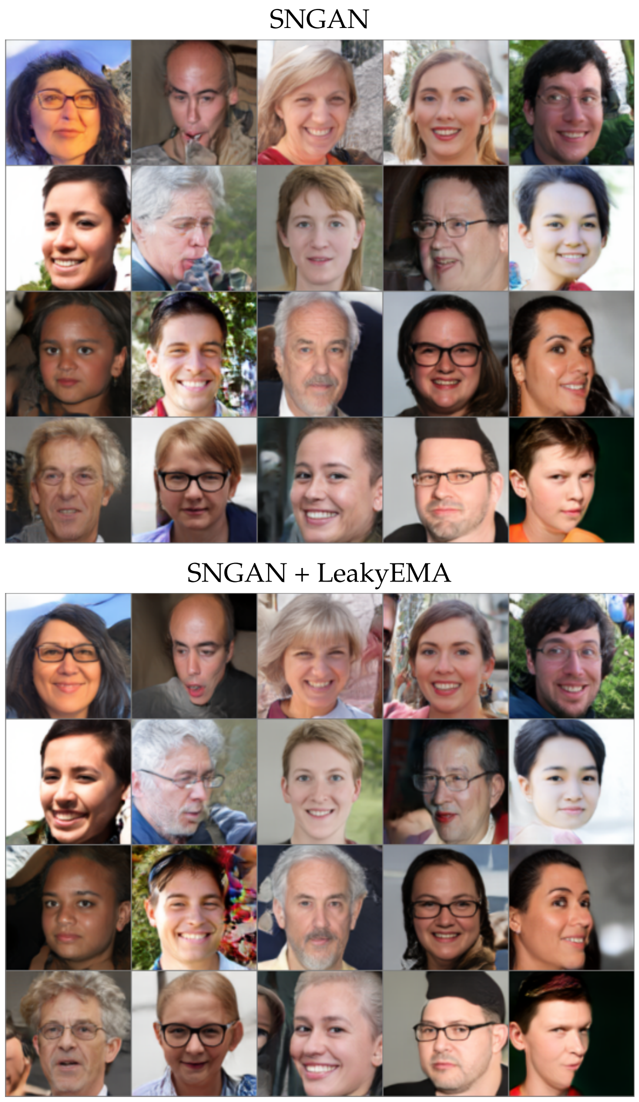

Figure 13. Uncurated samples of SNGAN trained on FFHQ with conventional (top) and LeakyEMA protocols (bottom). We use N=60K and β=0.5. Here we present samples from the best checkpoints obtained from both protocols and use the same latent noise samples.

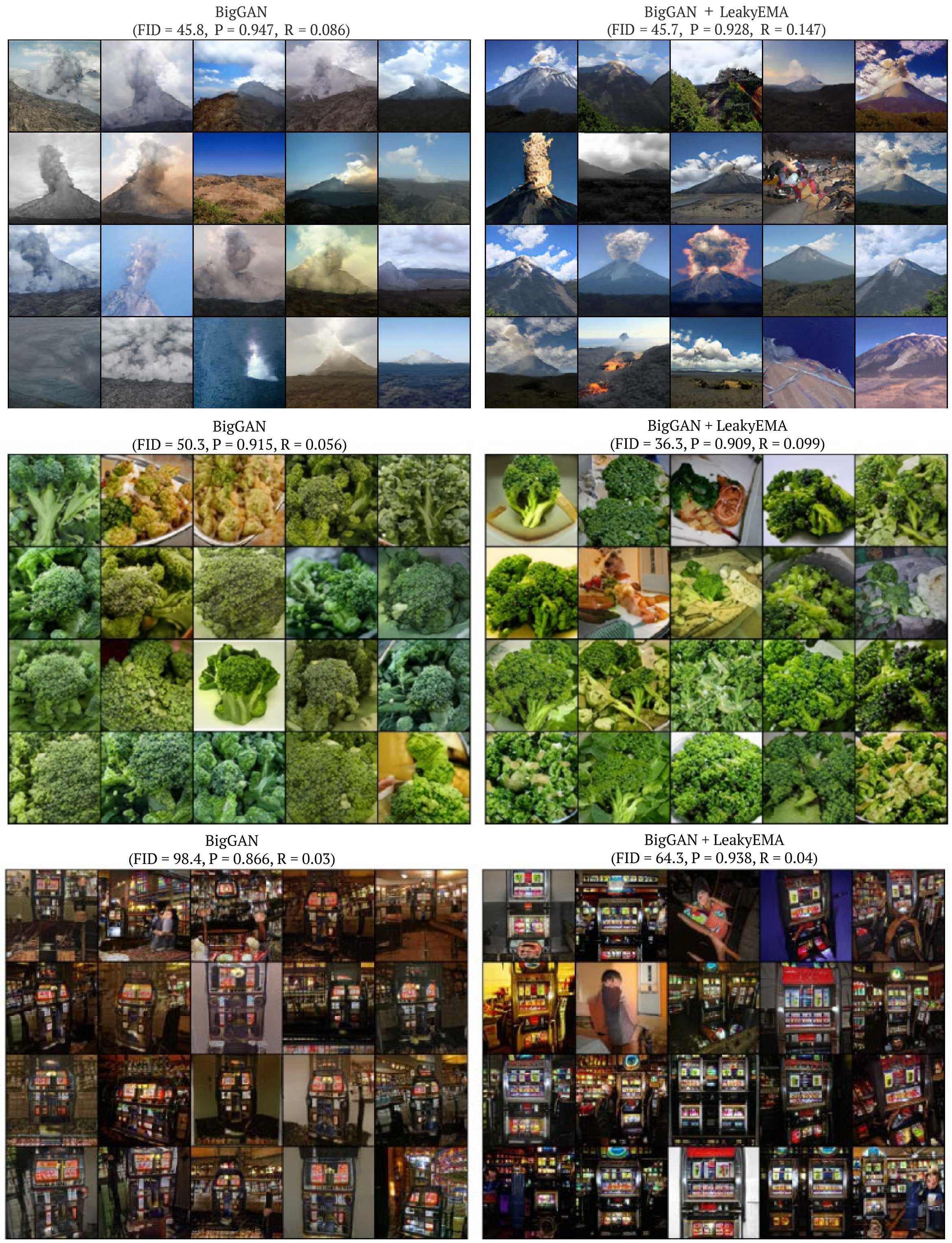
BigGAN
(FID = 45.8, P = 0.947, R = 0.086)
BigGAN + LeakyEMA
(FID = 45.7, P = 0.928, R = 0.147)
BigGAN
(FID = 50.3, P = 0.915, R = 0.056)
BigGAN + LeakyEMA
(FID = 36.3, P = 0.909, R = 0.099)
BigGAN
(FID = 98.4, P = 0.866, R = 0.03)
BigGAN + LeakyEMA
(FID = 64.3, P = 0.938, R = 0.04)

Figure 14. Conditional samples for the BigGAN trained in the conventional protocol (*left*) and LeakyEMA protocol (*right*). Here we present uncurated samples for the “Volcano”, “Broccoli” and “Slot” classes. We also report conditional FID, Precision (P) and Recall (R).

	IS	FID	Precision	Recall	Density	Coverage
Imagenet 128 × 128						
BigGAN	101.75	10.28	0.918	0.214	2.059	0.901
BigGAN+LeakyEMA	105.04	8.86	0.907	0.253	2.093	0.920
Cifar 32 × 32						
BigGAN	9.30	12.47	0.843	0.491	1.379	0.923
BigGAN+LeakyEMA	9.87	11.15	0.694	0.678	0.680	0.837

Table 6. Additional metrics for comparison of the model trained with LeakyEMA and the default.

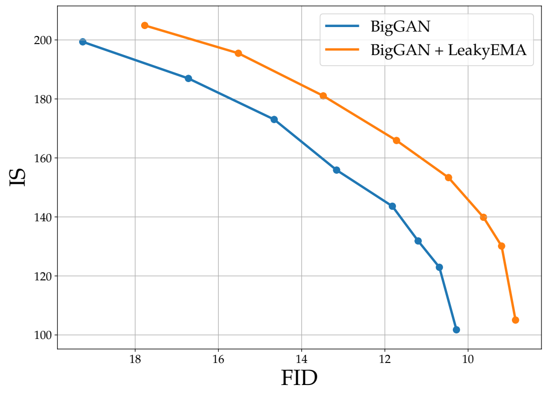

Figure 15. The FID-IS curves from varying truncation values linearly from 0.3 to 1.0 for the best model obtained with LeakyEMA and the publicly available BigGAN.

E. Rebuttal

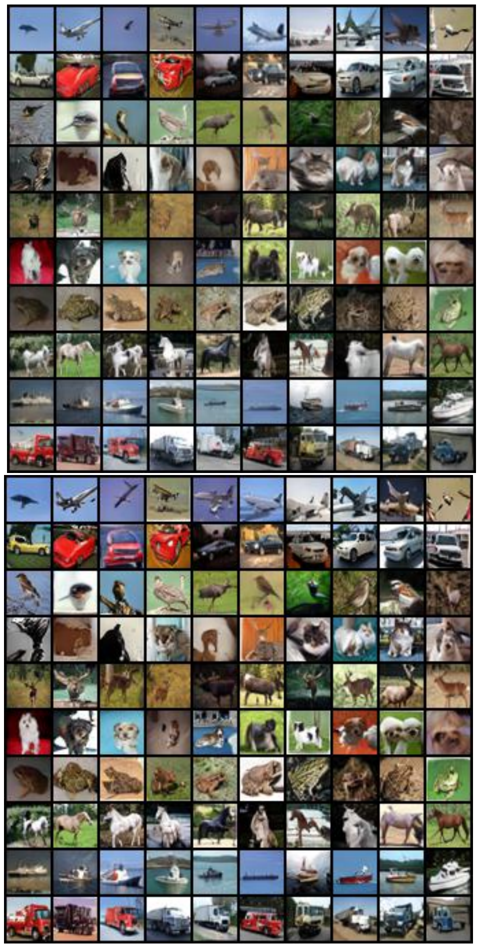

Figure 16. Comparison of the default (upper, FID= 12.47) and LeakyEMA (lower, FID= 11.15) random BigGAN samples on Cifar dataset.

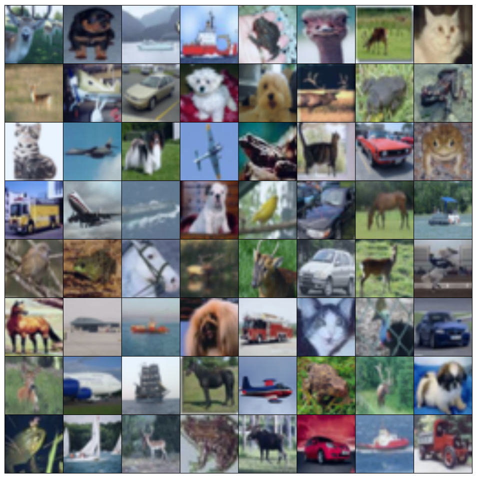

Figure 17. Comparison of the default (upper, FID= 12.47) and LeakyEMA (lower, FID= 11.15) random BigGAN samples on Cifar dataset.