A Large-Scale Study on Regularization and Normalization in GANs

Karol Kurach [*1] **Mario Lucic** [*1] **Xiaohua Zhai** [1] **Marcin Michalski** [1] **Sylvain Gelly** [1]

Abstract

Generative adversarial networks (GANs) are a class of deep generative models which aim to learn a target distribution in an unsupervised fashion. While they were successfully applied to many problems, training a GAN is a notoriously challenging task and requires a significant number of hyperparameter tuning, neural architecture engineering, and a non-trivial amount of "tricks". The success in many practical applications coupled with the lack of a measure to quantify the failure modes of GANs resulted in a plethora of proposed losses, regularization and normalization schemes, as well as neural architectures. In this work we take a sober view of the current state of GANs from a practical perspective. We discuss and evaluate common pitfalls and reproducibility issues, open-source our code on Github, and provide pre-trained models on TensorFlow Hub.

1. Introduction

Deep generative models are a powerful class of (mostly) unsupervised machine learning models. These models were recently applied to great effect in a variety of applications, including image generation, learned compression, and domain adaptation (Brock et al., 2019; Menick & Kalchbrenner, 2019; Karras et al., 2019; Lucic et al., 2019; Isola et al., 2017; Tschannen et al., 2018).

Generative adversarial networks (GANs) (Goodfellow et al., 2014) are one of the main approaches to learning such models in a fully unsupervised fashion. The GAN framework can be viewed as a two-player game where the first player, the *generator*, is learning to transform some simple input distribution to a complex high-dimensional distribution (e.g. over natural images), such that the second player, the *discriminator*, cannot tell whether the samples were drawn from the true distribution or were synthesized by the generator. The solution to the classic minimax formulation (Goodfellow et al., 2014) is the Nash equilibrium where neither player can improve unilaterally. As the generator and discriminator are usually parameterized as deep neural networks, this minimax problem is notoriously hard to solve.

In practice, the training is performed using stochastic gradient-based optimization methods. Apart from inheriting the optimization challenges associated with training deep neural networks, GAN training is also sensitive to the choice of the loss function optimized by each player, neural network architectures, and the specifics of regularization and normalization schemes applied. This has resulted in a flurry of research focused on addressing these challenges (Goodfellow et al., 2014; Salimans et al., 2016; Miyato et al., 2018; Gulrajani et al., 2017; Arjovsky et al., 2017; Mao et al., 2017).

Our Contributions In this work we provide a thorough empirical analysis of these competing approaches, and help the researchers and practitioners navigate this space. We first define the GAN landscape – the set of loss functions, normalization and regularization schemes, and the most commonly used architectures. We explore this search space on several modern large-scale datasets by means of hyperparameter optimization, considering both "good" sets of hyperparameters reported in the literature, as well as those obtained by sequential Bayesian optimization.

We first decompose the effect of various normalization and regularization schemes. We show that both gradient penalty (Gulrajani et al., 2017) as well as spectral normalization (Miyato et al., 2018) are useful in the context of high-capacity architectures. Then, by analyzing the impact of the loss function, we conclude that the non-saturating loss (Goodfellow et al., 2014) is sufficiently stable across datasets and hyperparameters. Finally, show that similar conclusions hold for both popular types of neural architectures used in state-of-the-art models. We then discuss some common pitfalls, reproducibility issues, and practical considerations. We provide reference implementations, including training and evaluation code on Github[1], and provide pre-trained models on TensorFlow Hub[2].

[*] Equal contribution [1] Google Research, Brain Team. Correspondence to: Karol Kurach <kkurach@google.com>, Mario Lucic <lucic@google.com>.

Proceedings of the 36th International Conference on Machine Learning, Long Beach, California, PMLR 97, 2019. Copyright 2019 by the author(s).

[1] `www.github.com/google/compare_gan`

[2] `www.tensorflow.org/hub`

2. The GAN Landscape

The main design choices in GANs are the loss function, regularization and/or normalization approaches, and the neural architectures. At this point GANs are extremely sensitive to these design choices. This fact coupled with optimization issues and hyperparameter sensitivity makes GANs hard to apply to new datasets. Here we detail the main design choices which are investigated in this work.

2.1. Loss Functions

Let P denote the target (true) distribution and Q the model distribution. Goodfellow et al. (2014) suggest two loss functions: the minimax GAN and the non-saturating (NS) GAN. In the former the discriminator minimizes the negative log-likelihood for the binary classification task. In the latter the generator maximizes the probability of generated samples being real. In this work we consider the non-saturating loss as it is known to outperform the minimax variant empirically. The corresponding discriminator and generator loss functions are

$$\begin{aligned}\mathcal{L}_{\text{D}} &= -\mathbb{E}_{x\sim P}[\log(D(x))] - \mathbb{E}_{\hat{x}\sim Q}[\log(1-D(\hat{x}))],\\ \mathcal{L}_{\text{G}} &= -\mathbb{E}_{\hat{x}\sim Q}[\log(D(\hat{x}))],\end{aligned}$$

where $D(x)$ denotes the probability of x being sampled from P. In Wasserstein GAN (WGAN) (Arjovsky et al., 2017) the authors propose to consider the Wasserstein distance instead of the Jensen-Shannon (JS) divergence. The corresponding loss functions are

$$\begin{aligned}\mathcal{L}_{\text{D}} &= -\mathbb{E}_{x\sim P}[D(x)] + \mathbb{E}_{\hat{x}\sim Q}[D(\hat{x})],\\ \mathcal{L}_{\text{G}} &= -\mathbb{E}_{\hat{x}\sim Q}[D(\hat{x})],\end{aligned}$$

where the discriminator output $D(x) \in \mathbb{R}$ and D is required to be 1-Lipschitz. Under the optimal discriminator, minimizing the proposed loss function with respect to the generator minimizes the Wasserstein distance between P and Q. A key challenge is ensure the Lipschitzness of D. Finally, we consider the least-squares loss (LS) which corresponds to minimizing the Pearson χ^2 divergence between P and Q (Mao et al., 2017). The corresponding loss functions are

$$\begin{aligned}\mathcal{L}_{\text{D}} &= -\mathbb{E}_{x\sim P}[(D(x)-1)^2] + \mathbb{E}_{\hat{x}\sim Q}[D(\hat{x})^2],\\ \mathcal{L}_{\text{G}} &= -\mathbb{E}_{\hat{x}\sim Q}[(D(\hat{x})-1)^2],\end{aligned}$$

where $D(x) \in \mathbb{R}$ is the output of the discriminator. Intuitively, this loss smooth loss function saturates slower than the cross-entropy loss.

2.2. Regularization and Normalization

Gradient Norm Penalty The idea is to regularize D by constraining the norms of its gradients (e.g. L_2). In the context of Wasserstein GANs and optimal transport this regularizer arises naturally and the gradient norm is evaluated on the points from the *optimal coupling* between samples from P and Q (GP) (Gulrajani et al., 2017). Computing this coupling during GAN training is computationally intensive, and a linear interpolation between these samples is used instead. The gradient norm can also be penalized close to the data manifold which encourages the discriminator to be piece-wise linear in that region (Dragan) (Kodali et al., 2017). A drawback of gradient penalty (GP) regularization scheme is that it can depend on the model distribution Q which changes during training. For Dragan it is unclear to which extent the Gaussian assumption for the manifold holds. In both cases, computing the gradient norms implies a non-trivial running time overhead.

Notwithstanding these natural interpretations for specific losses, one may also consider the gradient norm penalty as a classic regularizer for the complexity of the discriminator (Fedus et al., 2018). To this end we also investigate the impact of a L_2 regularization on D which is ubiquitous in supervised learning.

Discriminator Normalization Normalizing the discriminator can be useful from both the optimization perspective (more efficient gradient flow, more stable optimization), as well as from the representation perspective – the representation richness of the layers in a neural network depends on the spectral structure of the corresponding weight matrices (Miyato et al., 2018).

From the optimization point of view, several normalization techniques commonly applied to deep neural network training have been applied to GANs, namely batch normalization (BN) (Ioffe & Szegedy, 2015) and layer normalization (LN) (Ba et al., 2016). The former was explored in Denton et al. (2015) and further popularized by Radford et al. (2016), while the latter was investigated in Gulrajani et al. (2017). These techniques are used to normalize the activations, either across the batch (BN), or across features (LN), both of which were observed to improve the empirical performance.

From the representation point of view, one may consider the neural network as a composition of (possibly non-linear) mappings and analyze their spectral properties. In particular, for the discriminator to be a bounded operator it suffices to control the operator norm of each mapping. This approach is followed in Miyato et al. (2018) where the authors suggest dividing each weight matrix, including the matrices representing convolutional kernels, by their spectral norm. It is argued that spectral normalization results in discriminators of higher rank with respect to the competing approaches.

2.3. Generator and Discriminator Architecture

We explore two classes of architectures in this study: deep convolutional generative adversarial networks (DCGAN) (Radford et al., 2016) and residual networks

(ResNet) (He et al., 2016), both of which are ubiquitous in GAN research. Recently, Miyato et al. (2018) defined a variation of DCGAN, so called *SNDCGAN*. Apart from minor updates (cf. Section 4) the main difference to DCGAN is the use of an eight-layer discriminator network. The details of both networks are summarized in Table 4. The other architecture, *ResNet19*, is an architecture with five ResNet blocks in the generator and six ResNet blocks in the discriminator, that can operate on 128 × 128 images. We follow the ResNet setup from Miyato et al. (2018), with the small difference that we simplified the design of the discriminator.

The architecture details are summarized in Table 5a and Table 5b. With this setup we were able to reproduce the results in Miyato et al. (2018). An ablation study on various ResNet modifications is available in the Appendix.

2.4. Evaluation Metrics

We focus on several recently proposed metrics well suited to the image domain. For an in-depth overview of quantitative metrics we refer the reader to Borji (2019).

Inception Score (IS) Proposed by Salimans et al. (2016), the IS offers a way to quantitatively evaluate the quality of generated samples. Intuitively, the conditional label distribution of samples containing meaningful objects should have low entropy, and the variability of the samples should be high. which can be expressed as $\texttt{IS} = \exp(\mathbb{E}_{x\sim Q}[d_{KL}(p(y \mid x), p(y))])$. The authors found that this score is well-correlated with scores from human annotators. Drawbacks include insensitivity to the prior distribution over labels and not being a proper *distance*.

Fréchet Inception Distance (FID) In this approach proposed by Heusel et al. (2017) samples from P and Q are first embedded into a feature space (a specific layer of InceptionNet). Then, assuming that the embedded data follows a multivariate Gaussian distribution, the mean and covariance are estimated. Finally, the Fréchet distance between these two Gaussians is computed, i.e.

$$\texttt{FID} = ||\mu_x - \mu_y||_2^2 + \text{Tr}(\Sigma_x + \Sigma_y - 2(\Sigma_x\Sigma_y)^{\frac{1}{2}}),$$

where (μ_x, Σ_x), and (μ_y, Σ_y) are the mean and covariance of the embedded samples from P and Q, respectively. The authors argue that FID is consistent with human judgment and more robust to noise than IS. Furthermore, the score is sensitive to the visual quality of generated samples – introducing noise or artifacts in the generated samples will reduce the FID. In contrast to IS, FID can detect intra-class mode dropping – a model that generates only one image per class will have a good IS, but a bad FID (Lucic et al., 2018).

Kernel Inception Distance (KID) Bińkowski et al. (2018) argue that FID has no unbiased estimator and suggest KID as an unbiased alternative. In Appendix B we empirically compare KID to FID and observe that both metrics are very strongly correlated (Spearman rank-order correlation coefficient of 0.994 for LSUN-BEDROOM and 0.995 for CELEBA-HQ-128 datasets). As a result we focus on FID as it is likely to result in the same ranking.

2.5. Datasets

We consider three datasets, namely CIFAR10, CELEBA-HQ-128, and LSUN-BEDROOM. The LSUN-BEDROOM dataset contains slightly more than 3 million images (Yu et al., 2015).[3] We randomly partition the images into a train and test set whereby we use 30588 images as the test set. Secondly, we use the CELEBA-HQ dataset of 30K images (Karras et al., 2018). We use the 128×128×3 version obtained by running the code provided by the authors.[4] We use 3K examples as the test set and the remaining examples as the training set. Finally, we also include the CIFAR10 dataset which contains 70K images (32×32×3), partitioned into 60K training instances and 10K testing instances. The baseline FID scores are 12.6 for CELEBA-HQ-128, 3.8 for LSUN-BEDROOM, and 5.19 for CIFAR10. Details on FID computation are presented in Section 4.

2.6. Exploring the GAN Landscape

The search space for GANs is prohibitively large: exploring all combinations of all losses, normalization and regularization schemes, and architectures is outside of the practical realm. Instead, in this study we analyze several slices of this search space for each dataset. In particular, to ensure that we can reproduce existing results, we perform a study over the subset of this search space on CIFAR10. We then proceed to analyze the performance of these models across CELEBA-HQ-128 and LSUN-BEDROOM. In Section 3.1 we fix everything but the regularization and normalization scheme. In Section 3.2 we fix everything but the loss. Finally, in Section 3.3 we fix everything but the architecture. This allows us to decouple some of these design choices and provide some insight on what matters most in practice.

As noted by Lucic et al. (2018), one major issue preventing further progress is the hyperparameter tuning – currently, the community has converged to a small set of parameter values which work on some datasets, and may completely fail on others. In this study we combine the best hyperparameter settings found in the literature (Miyato et al., 2018), and perform sequential Bayesian optimization (Srinivas et al., 2010) to possibly uncover better hyperparameter settings. In a nutshell, in sequential Bayesian optimization one starts by

[3] The images are preprocessed to 128 × 128 × 3 using TensorFlow resize_image_with_crop_or_pad.

[4] github.com/tkarras/progressive_growing_of_gans

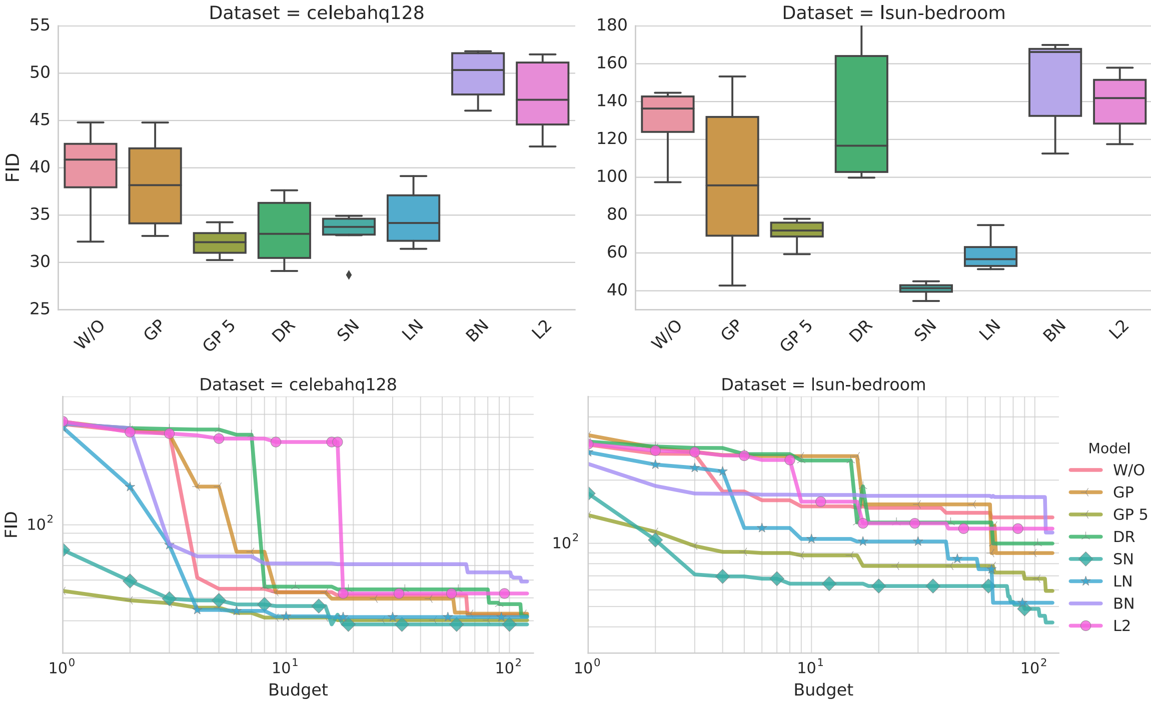

Figure 1. Plots in the first row show the FID distribution for top 5% models (lower is better). We observe that both gradient penalty (GP) and spectral normalization (SN) outperform the non-regularized/normalized baseline (W/O). Unfortunately, none fully address the stability issues. The second row shows the estimate of the minimum FID achievable for a given computational budget. For example, to obtain an FID below 100 using non-saturating loss with gradient penalty, we need to try at least 6 hyperparameter settings. At the same time, we could achieve a better result (lower FID) with spectral normalization and 2 hyperparameter settings. These results suggest that spectral norm is a better practical choice.

Parameter	Discrete Value
Learning rate α	$\{0.0002, 0.0001, 0.001\}$
Reg. strength λ	$\{1, 10\}$
$(\beta_1, \beta_2, n_{dis})$	$\{(0.5, 0.900, 5), (0.5, 0.999, 1),$ $(0.5, 0.999, 5), (0.9, 0.999, 5)\}$

Table 1. Hyperparameter ranges used in this study. The Cartesian product of the fixed values suffices to uncover most of the recent results from the literature.

Parameter	Range	Log
Learning rate α	$[10^{-5}, 10^{-2}]$	Yes
λ for L_2	$[10^{-4}, 10^{1}]$	Yes
λ for non-L_2	$[10^{-1}, 10^{2}]$	Yes
$\beta_1 \times \beta_2$	$[0, 1] \times [0, 1]$	No

Table 2. We use sequential Bayesian optimization (Srinivas et al., 2010) to explore the hyperparameter settings from the specified ranges. We explore 120 hyperparameter settings in 12 rounds of optimization.

evaluating a set of hyperparameter settings (possibly chosen randomly). Then, based on the obtained scores for these hyperparameters the next set of hyperparameter combinations is chosen such to balance the exploration (finding new hyperparameter settings which might perform well) and exploitation (selecting settings close to the best-performing settings). We then consider the top performing models and discuss the impact of the computational budget.

We summarize the fixed hyperparameter settings in Table 1 which contains the "good" parameters reported in recent publications (Fedus et al., 2018; Miyato et al., 2018; Gulrajani et al., 2017). In particular, we consider the Cartesian product of these parameters to obtain 24 hyperparameter settings to reduce the survivorship bias. Finally, to provide a fair comparison, we perform sequential Bayesian optimization (Srinivas et al., 2010) on the parameter ranges provided in Table 2. We run 12 rounds (i.e. we communicate with the oracle 12 times) of sequential optimization, each with a batch of 10 hyperparameter sets selected based on the FID scores from the results of the previous iterations. As we explore the number of discriminator updates per generator update (1 or 5),

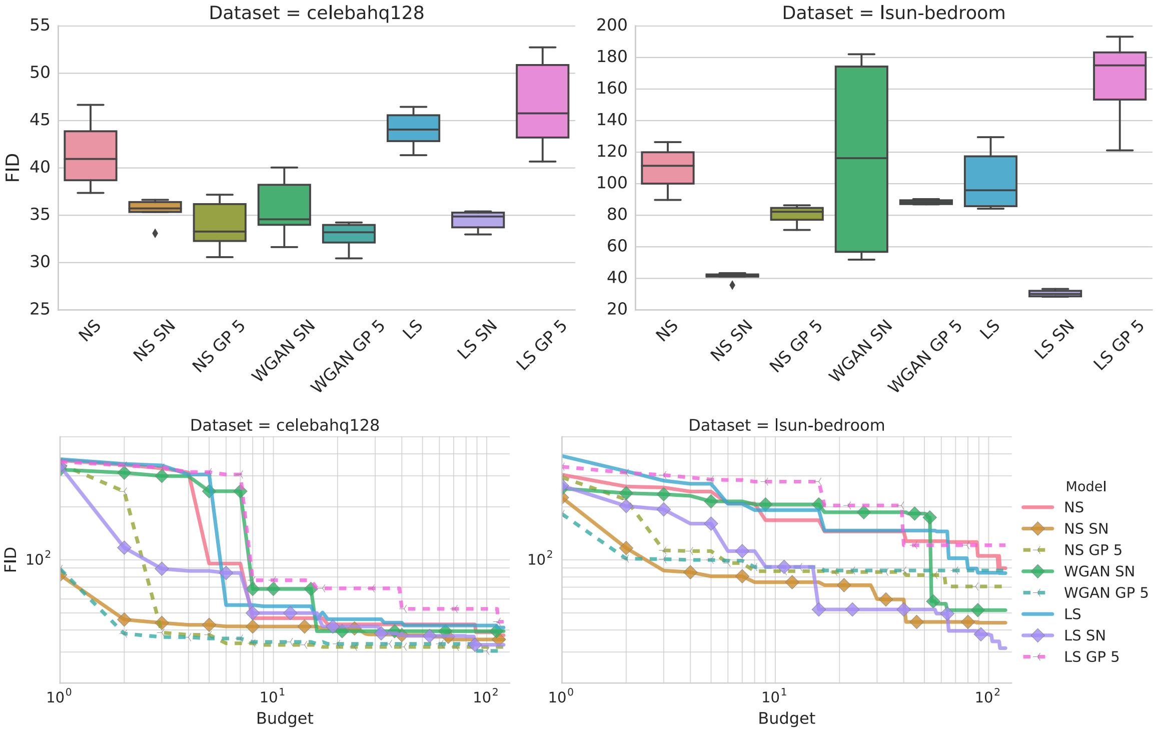

Figure 2. The first row shows the FID distribution for top 5% models. We compare the non-saturating (NS) loss, the Wasserstein loss (WGAN), and the least-squares loss (LS), combined with the most prominent regularization and normalization strategies, namely spectral norm (SN) and gradient penalty (GP). We observe that spectral norm consistently improves the sample quality. In some cases the gradient penalty can help, but there is no clear trend. From the computational budget perspective one can attain lower levels of FID with fewer hyperparameter optimization settings which demonstrates the practical advantages of spectral normalization over competing method.

this leads to an additional 240 hyperparameter settings which in some cases outperform the previously known hyperparameter settings. The batch size is set to 64 for all the experiments. We use a fixed the number of discriminator update steps of 100K for LSUN-BEDROOM dataset and CELEBA-HQ-128 dataset, and 200K for CIFAR10 dataset. We apply the Adam optimizer (Kingma & Ba, 2015).

3. Experimental Results and Discussion

Given that there are 4 major components (loss, architecture, regularization, normalization) to analyze for each dataset, it is infeasible to explore the whole landscape. Hence, we opt for a more pragmatic solution – we keep some dimensions fixed, and vary the others. We highlight two aspects:

1. We train the models using various hyperparameter settings, both predefined and ones obtained by sequential Bayesian optimization. Then we compute the *FID distribution* of the top 5% of the trained models. The lower the median FID, the better the model. The lower the variance, the more stable the model is from the optimization point of view.
2. The tradeoff between the computational budget (for training) and model quality in terms of FID. Intuitively, given a limited computational budget (being able to train only k different models), which model should one choose? Clearly, models which achieve better performance using the same computational budget should be preferred in practice. To compute the minimum attainable FID for a fixed budget k we simulate a practitioner attempting to find a good hyperparameter setting for their model: we spend a part of the budget on the "good" hyperparameter settings reported in recent publications, followed by exploring new settings (i.e. using Bayesian optimization). As this is a random process, we repeat it 1000 times and report the *average* of the minimum attainable FID.

Due to the fact that the training is sensitive to the initial weights, we train the models 5 times, each time with a different random initialization, and report the median FID. The variance in FID for models obtained by sequential Bayesian optimization is handled implicitly by the applied exploration-exploitation strategy.

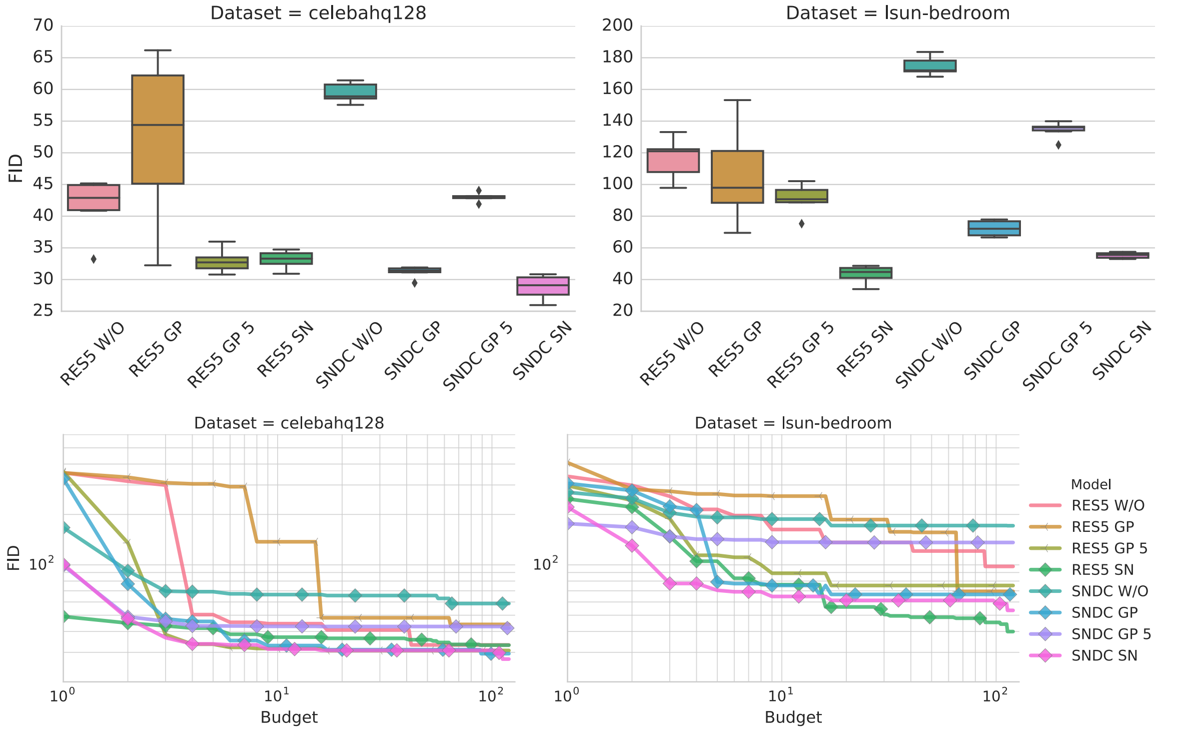

Figure 3. The first row show s the FID distribution for top 5% models. We compare the ResNet-based neural architecture with the SNDCGAN architecture. We use the non-saturating (NS) loss in all experiments, and apply either spectral normalization (SN) or the gradient penalty (GP). We observe that spectral norm consistently improves the sample quality. In some cases the gradient penalty can help, but the need to tune one additional hyperparameter leads to a lower computational efficiency.

3.1. Regularization and Normalization

The goal of this study is to compare the relative performance of various regularization and normalization methods presented in the literature, namely: batch normalization (BN) (Ioffe & Szegedy, 2015), layer normalization (LN) (Ba et al., 2016), spectral normalization (SN), gradient penalty (GP) (Gulrajani et al., 2017), Dragan penalty (DR) (Kodali et al., 2017), or L_2 regularization. We fix the loss to non-saturating loss (Goodfellow et al., 2014) and the ResNet19 with generator and discriminator architectures described in Table 5a. We analyze the impact of the loss function in Section 3.2 and of the architecture in Section 3.3. We consider both CELEBA-HQ-128 and LSUN-BEDROOM with the hyperparameter settings shown in Tables 1 and 2.

The results are presented in Figure 1. We observe that adding batch norm to the discriminator hurts the performance. Secondly, gradient penalty can help, but it doesn't stabilize the training. In fact, it is non-trivial to strike a balance of the loss and regularization strength. Spectral normalization helps improve the model quality and is more computationally efficient than gradient penalty. This is consistent with recent results in Zhang et al. (2019). Similarly to the loss study, models using GP penalty may benefit from 5:1 ratio of discriminator to generator updates. Furthermore, in a separate ablation study we observed that running the optimization procedure for an additional 100K steps is likely to increase the performance of the models with GP penalty.

3.2. Impact of the Loss Function

Here we investigate whether the above findings also hold when the loss functions are varied. In addition to the non-saturating loss (NS), we also consider the the least-squares loss (LS) (Mao et al., 2017), or the Wasserstein loss (WGAN) (Arjovsky et al., 2017). We use the ResNet19 with generator and discriminator architectures detailed in Table 5a. We consider the most prominent normalization and regularization approaches: gradient penalty (Gulrajani et al., 2017), and spectral normalization (Miyato et al., 2018). Other parameters are detailed in Table 1. We also performed a study on the recently popularized hinge loss (Lim & Ye, 2017; Miyato et al., 2018; Brock et al., 2019) and present it in the Appendix.

The results are presented in Figure 2. Spectral normalization improves the model quality on both datasets. Similarly, the gradient penalty can help, but finding a good regularization tradeoff is non-trivial and requires a large computational budget. Models using the GP penalty benefit from 5:1 ratio of discriminator to generator updates (Gulrajani et al., 2017).

3.3. Impact of the Neural Architectures

An interesting practical question is whether our findings also hold for different neural architectures. To this end, we also perform a study on SNDCGAN from Miyato et al. (2018). We consider the non-saturating GAN loss, gradient penalty and spectral normalization. While for smaller architectures regularization is not essential (Lucic et al., 2018), the regularization and normalization effects might become more relevant due to deeper architectures and optimization considerations.

The results are presented in Figure 3. We observe that both architectures achieve comparable results and benefit from regularization and normalization. Spectral normalization strongly outperforms the baseline for both architectures.

Simultaneous Regularization and Normalization A common observation is that the Lipschitz constant of the discriminator is critical for the performance, one may expect simultaneous regularization and normalization could improve model quality. To quantify this effect, we fix the loss to non-saturating loss (Goodfellow et al., 2014), use the Resnet19 architecture (as above), and combine several normalization and regularization schemes, with hyperparameter settings shown in Table 1 coupled with 24 randomly selected parameters. The results are presented in Figure 4. We observe that one may benefit from additional regularization and normalization. However, a lot of computational effort has to be invested for somewhat marginal gains in FID. Nevertheless, given enough computational budget we advocate simultaneous regularization and normalization – spectral normalization and layer normalization seem to perform well in practice.

4. Challenges of a Large-Scale Study

In this section we focus on several pitfalls we encountered while trying to reproduce existing results and provide a fair and accurate comparison.

Metrics There already seems to be a divergence in how the FID score is computed: (1) Some authors report the score on training data, yielding a FID between 50K training and 50K generated samples (Unterthiner et al., 2018). Some opt to report the FID based on 10K test samples and 5K generated samples and use a custom implementation (Miyato et al., 2018). Finally, Lucic et al. (2018) report the score with respect to the test data, in particular FID between 10K test samples, and 10K generated samples. The subtle differences will result in a mismatch between the reported FIDs, in some cases of more than 10%. We argue that FID should be computed with respect to the test dataset. Furthermore, whenever possible, one should use the same number of instances as previously reported results. Towards this end we use 10K test samples and 10K generated samples on CIFAR10 and LSUN-BEDROOM, and 3K vs 3K on CELEBA-HQ-128 as in in Lucic et al. (2018).

Details of Neural Architectures Even in popular architectures, like ResNet, there is still a number of design decisions one needs to make, that are often omitted from the reported results. Those include the exact design of the ResNet block (order of layers, when is ReLu applied, when to upsample and downsample, how many filters to use). Some of these differences might lead to potentially unfair comparison. As a result, we suggest to use the architectures presented within this work as a solid baseline. An ablation study on various ResNet modifications is available in the Appendix.

Datasets A common issue is related to dataset processing – does LSUN-BEDROOM always correspond to the same dataset? In most cases the precise algorithm for upscaling or cropping is not clear which introduces inconsistencies between results on the “same” dataset.

Implementation Details and Non-Determinism One major issue is the mismatch between the algorithm presented in a paper and the code provided online. We are aware that there is an embarrassingly large gap between a good implementation and a bad implementation of a given model. Hence, when no code is available, one is forced to guess which modifications were done. Another particularly tricky issue is removing randomness from the training process. After one fixes the data ordering and the initial weights, obtaining the same score by training the same model twice is non-trivial due to randomness present in certain GPU operations (Chetlur et al., 2014). Disabling the optimizations causing the non-determinism often results in an order of magnitude running time penalty.

While each of these issues taken in isolation seems minor, they compound to create a mist which introduces friction in practical applications and the research process (Sculley et al., 2018).

5. Related Work

A recent large-scale study on GANs and Variational Autoencoders was presented in Lucic et al. (2018). The authors consider several loss functions and regularizers, and study the effect of the loss function on the FID score, with low-to-medium complexity datasets (MNIST, CIFAR10, CELEBA), and a single neural network architecture. In this limited setting, the authors found that there is no statistically significant difference between recently introduced models and the original non-saturating GAN. A study of the effects of gradient-norm regularization in GANs was recently pre-

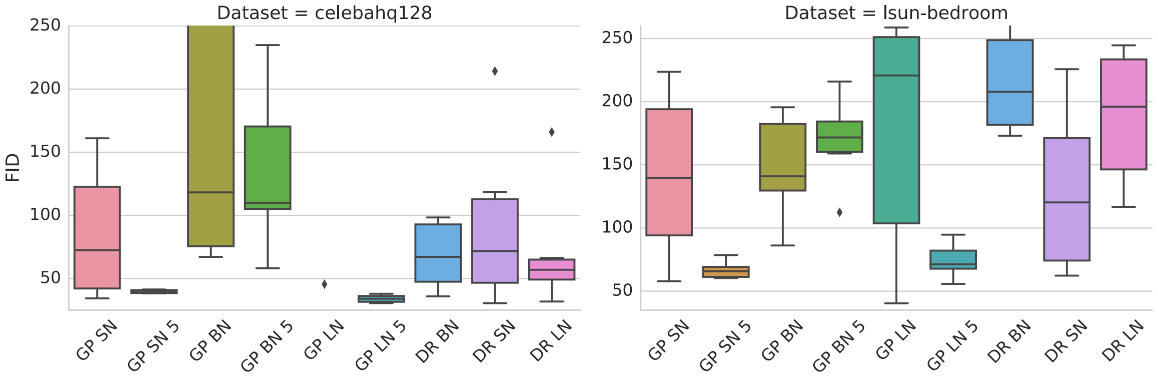

Figure 4. Can one benefit from simultaneous regularization and normalization? The plots show the FID distribution for top 5% models where we compare various combinations of regularization and normalization strategies. Gradient penalty coupled with spectral normalization (SN) or layer normalization (LN) strongly improves the performance over the baseline. This can be partially explained by the fact that SN doesn't ensure that the discriminator is 1-Lipschitz due to the way convolutional layers are normalized.

sented in Fedus et al. (2018). The authors posit that the gradient penalty can also be applied to the non-saturating GAN, and that, to a limited extent, it reduces the sensitivity to hyperparameter selection. In a recent work on spectral normalization, the authors perform a small study of the competing regularization and normalization approaches (Miyato et al., 2018). We are happy to report that we could reproduce these results and we present them in the Appendix.

Inspired by these works and building on the available open-source code from Lucic et al. (2018), we take one additional step in all dimensions considered therein: more complex neural architectures, more complex datasets, and more involved regularization and normalization schemes.

6. Conclusions and Future Work

In this work we study the impact of regularization and normalization schemes on GAN training. We consider the state-of-the-art approaches and vary the loss functions and neural architectures. We study the impact of these design choices on the quality of generated samples which we assess by recently introduced quantitative metrics.

Our fair and thorough empirical evaluation suggests that when the computational budget is limited one should consider non-saturating GAN loss and spectral normalization as default choices when applying GANs to a new dataset. Given additional computational budget, we suggest adding the gradient penalty from Gulrajani et al. (2017) and training the model until convergence. Furthermore, we observe that both classes of popular neural architectures can perform well across the considered datasets. A separate ablation study uncovered that most of the variations applied in the ResNet style architectures lead to marginal improvements in the sample quality.

As a result of this large-scale study we identify the common pitfalls standing in the way of accurate and fair comparison and propose concrete actions to demystify the future results – issues with metrics, dataset preprocessing, non-determinism, and missing implementation details are particularly striking. We hope that this work, together with the open-sourced reference implementations and trained models, will serve as a solid baseline for future GAN research.

Future work should carefully evaluate models which necessitate large-scale training such as BigGAN (Brock et al., 2019), models with custom architectures (Chen et al., 2019; Karras et al., 2019; Zhang et al., 2019), recently proposed regularization techniques (Roth et al., 2017; Mescheder et al., 2018), and other proposals for stabilizing the training (Chen et al., 2018). In addition, given the popularity of conditional GANs, one should explore whether these insights transfer to the conditional settings. Finally, given the drawbacks of FID and IS, additional quantitative evaluation using recently proposed metrics could bring novel insights (Sajjadi et al., 2018; Kynkäänniemi et al., 2019).

Acknowledgments

We are grateful to Michael Tschannen for detailed comments on this manuscript.

References

Arjovsky, M., Chintala, S., and Bottou, L. Wasserstein Generative Adversarial Networks. In *International Conference on Machine Learning*, 2017.

Ba, J. L., Kiros, J. R., and Hinton, G. E. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

Bińkowski, M., Sutherland, D. J., Arbel, M., and Gretton, A.

Demystifying MMD GANs. In *International Conference on Learning Representations*, 2018.

Borji, A. Pros and cons of GAN evaluation measures. *Computer Vision and Image Understanding*, 2019.

Brock, A., Donahue, J., and Simonyan, K. Large scale GAN training for high fidelity natural image synthesis. In *International Conference on Learning Representations*, 2019.

Chen, T., Zhai, X., Ritter, M., Lucic, M., and Houlsby, N. Self-Supervised Generative Adversarial Networks. In *Computer Vision and Pattern Recognition*, 2018.

Chen, T., Lucic, M., Houlsby, N., and Gelly, S. On Self Modulation for Generative Adversarial Networks. In *International Conference on Learning Representations*, 2019.

Chetlur, S., Woolley, C., Vandermersch, P., Cohen, J., Tran, J., Catanzaro, B., and Shelhamer, E. cuDNN: Efficient primitives for deep learning. *arXiv preprint arXiv:1410.0759*, 2014.

Denton, E. L., Chintala, S., Szlam, A., and Fergus, R. Deep Generative Image Models using a Laplacian Pyramid of Adversarial Networks. In *Advances in Neural Information Processing Systems*, 2015.

Fedus, W., Rosca, M., Lakshminarayanan, B., Dai, A. M., Mohamed, S., and Goodfellow, I. Many paths to equilibrium: GANs do not need to decrease a divergence at every step. In *International Conference on Learning Representations*, 2018.

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.

Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., and Courville, A. Improved training of Wasserstein GANs. In *Advances in Neural Information Processing Systems*, 2017.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Computer Vision and Pattern Recognition*, 2016.

Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., Klambauer, G., and Hochreiter, S. GANs trained by a two time-scale update rule converge to a Nash equilibrium. In *Advances in Neural Information Processing Systems*, 2017.

Ioffe, S. and Szegedy, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

Isola, P., Zhu, J.-Y., Zhou, T., and Efros, A. A. Image-to-image translation with conditional adversarial networks. In *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 *Computer Vision and Pattern Recognition*, 2019.

Kingma, D. and Ba, J. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015.

Kodali, N., Abernethy, J., Hays, J., and Kira, Z. On convergence and stability of GANs. *arXiv preprint arXiv:1705.07215*, 2017.

Kynkäänniemi, T., Karras, T., Laine, S., Lehtinen, J., and Aila, T. Improved precision and recall metric for assessing generative models. *arXiv preprint arXiv:1904.06991*, 2019.

Lim, J. H. and Ye, J. C. Geometric GAN. *arXiv preprint arXiv:1705.02894*, 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*, 2018.

Lucic, M., Tschannen, M., Ritter, M., Zhai, X., Bachem, O., and Gelly, S. High-Fidelity Image Generation With Fewer Labels. In *International Conference on Machine Learning*, 2019.

Mao, X., Li, Q., Xie, H., Lau, R. Y., Wang, Z., and Smolley, S. P. Least squares generative adversarial networks. In *International Conference on Computer Vision*, 2017.

Menick, J. and Kalchbrenner, N. Generating high fidelity images with subscale pixel networks and multidimensional upscaling. In *International Conference on Learning Representations*, 2019.

Mescheder, L., Geiger, A., and Nowozin, S. Which training methods for GANs do actually Converge? *arXiv preprint arXiv:1801.04406*, 2018.

Miyato, T., Kataoka, T., Koyama, M., and Yoshida, Y. Spectral normalization for generative adversarial networks. *International Conference on Learning Representations*, 2018.

Radford, A., Metz, L., and Chintala, S. Unsupervised representation learning with deep convolutional generative adversarial networks. *International Conference on Learning Representations*, 2016.

Roth, K., Lucchi, A., Nowozin, S., and Hofmann, T. Stabilizing training of generative adversarial networks through regularization. In *Advances in Neural Information Processing Systems*, 2017.

Sajjadi, M. S., Bachem, O., Lucic, M., Bousquet, O., and Gelly, S. Assessing generative models via precision and recall. In *Advances in Neural Information Processing Systems*, 2018.

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*, 2016.

Sculley, D., Snoek, J., Wiltschko, A., and Rahimi, A. Winner's Curse? On Pace, Progress, and Empirical Rigor, 2018.

Srinivas, N., Krause, A., Kakade, S., and Seeger, M. W. Gaussian process optimization in the bandit setting: No regret and experimental design. In *International Conference on Machine Learning*, 2010.

Tschannen, M., Agustsson, E., and Lucic, M. Deep generative models for distribution-preserving lossy compression. *Advances in Neural Information Processing Systems*, 2018.

Unterthiner, T., Nessler, B., Seward, C., Klambauer, G., Heusel, M., Ramsauer, H., and Hochreiter, S. Coulomb GANs: Provably Optimal Nash Equilibria via Potential Fields. In *International Conference on Learning Representations*, 2018.

Yu, F., Zhang, Y., Song, S., Seff, A., 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*, 2019.