Multi-objective training of Generative Adversarial Networks with multiple discriminators

Isabela Albuquerque [*1] **João Monteiro** [*1] **Thang Doan** [2] **Breandan Considine** [3] **Tiago Falk** [1]
Ioannis Mitliagkas [3]

Abstract

Recent literature has demonstrated promising results for training Generative Adversarial Networks by employing a set of discriminators, in contrast to the traditional game involving one generator against a single adversary. Such methods perform single-objective optimization on some simple consolidation of the losses, e.g. an arithmetic average. In this work, we revisit the multiple-discriminator setting by framing the simultaneous minimization of losses provided by different models as a multi-objective optimization problem. Specifically, we evaluate the performance of multiple gradient descent and the hypervolume maximization algorithm on a number of different datasets. Moreover, we argue that the previously proposed methods and hypervolume maximization can all be seen as variations of multiple gradient descent in which the update direction can be computed efficiently. Our results indicate that hypervolume maximization presents a better compromise between sample quality and computational cost than previous methods.

1. Introduction

Generative Adversarial Networks (GANs) (Goodfellow et al., 2014) offer a new approach to generative modeling, using game-theoretic training schemes to implicitly learn a given probability density. Prior to the emergence of GAN architectures, realistic generative modeling remained elusive. While offering unprecedented realism, GAN training still remains fraught with stability issues. Commonly reported shortcomings involve the lack of useful gradient signal provided by the discriminator, and mode collapse, i.e. lack of diversity in the generator's samples.

Considerable research has been devoted in recent literature to overcome training instability[1] within the GAN framework. Some architectures such as BEGAN (Berthelot et al., 2017) have applied auto-encoders as discriminators and proposed a new loss function to help stabilize training. Methods such as TTUR (Heusel et al., 2017), in turn, have attempted to define separate schedules for updating the generator and discriminator. The PacGAN algorithm (Lin et al., 2017) proposes to modify the discriminator's architecture to accept m concatenated samples as input. These samples are jointly classified as either real or generated, and the authors show that such an approach can help to enforce sample diversity. Furthermore, *spectral normalization* was applied to the discriminator's parameters in SNGAN (Miyato et al., 2018) aiming to ensure Lipschitz continuity, which is empirically shown to yield high quality samples across several sets of hyperparameters. Alternatively, recent approaches have proposed to tackle GAN instability issues with multiple discriminators. Neyshabur et al. (2017) propose a setting such that one generator is trained against a set of discriminators, where each one sees a fixed random projection of the inputs. Prior work, including Durugkar et al. (2016) have also explored training with multiple discriminators.

In this paper, we build upon Neyshabur et al. (2017)'s framework and propose reformulating their average loss minimization to further stabilize GAN training. Specifically, we propose treating the loss signal provided by each discriminator as an independent objective function. To achieve this, we simultaneously minimize the losses using multi-objective optimization techniques. Namely, we exploit well known methods in optimization literature such as the multiple gradient descent (MGD) algorithm (Désidéri, 2012). However, due to MGD's prohibitively high cost in the case of large neural networks, we propose to use more efficient

[*]Equal contribution [1]INRS-EMT, Université du Québec, Montreal, Canada [2]Desautels Faculty of Management, McGill University, Montreal, Canada [3]Quebec Artificial Intelligence Institute, Université de Montréal, Montreal, Canada. Correspondence to: Isabela Albuquerque <isabela.albuquerque@emt.inrs.ca>, João Monteiro <joao.monteiro@emt.inrs.ca>.

Proceedings of the 36^{th} International Conference on Machine Learning, Long Beach, California, PMLR 97, 2019.

[1]*Instability* in the sense commonly used in GANs literature, i.e. when the discriminator is able to easily distinguish between real and fake samples during the training phase (Neyshabur et al., 2017; Arjovsky et al., 2017; Berthelot et al., 2017).

alternatives such as maximizing hypervolume in the region defined between a fixed, shared upper bound on the losses, which we refer to as the *nadir point* $\boldsymbol{\eta}^*$, and each of the component losses. In contrast to Neyshabur et al. (2017)'s approach, where the average loss is minimized when training the generator, hypervolume maximization (HVM) optimizes a weighted loss, and the generator's training will assign greater importance to feedback from discriminators against which it performs poorly.

Experiments performed on MNIST show that HVM presents a useful compromise between *computational cost* and *sample quality* when compared to GMAN's average loss minimization (low quality and cost), and MGD (high quality and cost). Our results indicate that increasing the number of discriminators consequently increases the generator's robustness to hyperparameter settings. In addition, experiments performed on CIFAR-10 indicate the method described produces higher quality and more diverse generator samples as measured by several quantitative metrics. Moreover, image quality and sample diversity are once more shown to consistently improve as we increase the number of discriminators.

In summary, our main contributions are as follows: **(i)** A variation of the single-solution HVM is introduced as an alternative to MGD for the case of large neural networks. **(ii)** We offer a new perspective on multiple-discriminator GAN training by framing it in the context of multi-objective optimization, and draw similarities between previous approaches under this setting and MGD, commonly employed as a general solver for multi-objective optimization.

The remainder of this document is organized as follows: Section 2 introduces definitions for multi-objective optimization and MGD. In Section 3, we describe prior relevant literature. The HVM algorithm is detailed in Section 4, with experiments and results presented in Section 5. Conclusions and directions for future work are drawn in Section 6.

2. Preliminaries

In this section we provide some definitions regarding multi-objective optimization from prior literature which will be useful in the following sections. Boldface notation is used to denote vector-valued variables and functions.

Multi-objective optimization. A multi-objective optimization problem is defined as (Deb, 2001):

$$\min \mathbf{F}(\mathbf{x}) = [f_1(\mathbf{x}), f_2(\mathbf{x}), ..., f_K(\mathbf{x})]^T, \mathbf{x} \in \Omega \quad (1)$$

where K is the number of objectives, Ω is the variables space and $\mathbf{x} = [x_1, x_2, ..., x_n]^T \in \Omega$ is a decision vector or possible solution to the problem. $\mathbf{F} : \Omega \to \mathbb{R}^K$ is a set of K-objective functions that maps the n-dimensional variables space to the K-dimensional objective space.

Pareto-dominance. Let $\mathbf{x}_1$ and $\mathbf{x}_2$ be two decision vectors. $\mathbf{x}_1$ is said to dominate $\mathbf{x}_2$ (denoted by $\mathbf{x}_1 \prec \mathbf{x}_2$) if and only if $f_i(\mathbf{x}_1) \leq f_i(\mathbf{x}_2)$ for all $i \in \{1, 2, \ldots, K\}$ and $f_j(\mathbf{x}_1) < f_j(\mathbf{x}_2)$ for some $j \in \{1, 2, \ldots, K\}$. If a decision vector $\mathbf{x}$ is dominated by no other vector in Ω, $\mathbf{x}$ is called a non-dominated solution.

Pareto-optimality. A decision vector $\mathbf{x}^* \in \Omega$ is said to be Pareto-optimal if and only if there is no $\mathbf{x} \in \Omega$ such that $\mathbf{x} \prec \mathbf{x}^*$, i.e. $\mathbf{x}^*$ is a non-dominated solution. The Pareto-optimal Set (PS) is defined as the set of all Pareto-optimal solutions $\mathbf{x} \in \Omega$, i.e., PS $= \{\mathbf{x} \in \Omega \mid \mathbf{x}$ is Pareto optimal$\}$. The set of all objective vectors $\mathbf{F}(\mathbf{x})$ such that $\mathbf{x}$ is Pareto-optimal is called Pareto front (PF), that is $PF = \{\mathbf{F}(\mathbf{x}) \in \mathbb{R}^K \mid \mathbf{x} \in PS\}$.

Pareto-stationarity. Pareto-stationarity is a necessary condition for Pareto-optimality. For f_k differentiable everywhere for all k, $\mathbf{F}$ is Pareto-stationary at $\mathbf{x}$ if there exists a set of scalars $\alpha_k, k \in \{1, \ldots, K\}$, such that:

$$\sum_{k=1}^{K} \alpha_k \nabla f_k = \mathbf{0}, \quad \sum_{k=1}^{K} \alpha_k = 1, \quad \alpha_k \geq 0 \quad \forall k. \quad (2)$$

Multiple Gradient Descent. Multiple gradient descent (Schäffler et al., 2002; Désidéri, 2012; Peitz & Dellnitz, 2018; Sener & Koltun, 2018) was proposed for the unconstrained case of multi-objective optimization of $\mathbf{F}(\mathbf{x})$ assuming a convex, continuously differentiable and smooth $f_k(\mathbf{x})$ for all k. MGD finds a common descent direction for all f_k by defining the convex hull of all $\nabla f_k(\mathbf{x})$ and finding the minimum norm element within it. Consider $\mathbf{w}^*$ given by:

$$\begin{aligned} \mathbf{w}^* = \text{argmin}||\mathbf{w}||^2, \quad \mathbf{w} = \sum_{k=1}^{K} \alpha_k \nabla f_k(\mathbf{x}), \\ \text{s.t.} \quad \sum_{k=1}^{K} \alpha_k = 1, \quad \alpha_k \geq 0 \quad \forall k. \end{aligned} \quad (3)$$

$\mathbf{w}^*$ will be either $\mathbf{0}$ in which case $\mathbf{x}$ is a Pareto-stationary point, or $\mathbf{w}^* \neq \mathbf{0}$ and then $\mathbf{w}^*$ is a descent direction for all $f_i(\mathbf{x})$. Similar to gradient descent, MGD consists in finding the *common* steepest descent direction $\mathbf{w}_t^*$ at each iteration t, and then updating parameters with a learning rate λ according to $\mathbf{x}_{t+1} = \mathbf{x}_t - \lambda \frac{\mathbf{w}_t^*}{||\mathbf{w}_t^*||}$.

3. Related work

3.1. Training GANs with multiple discriminators

While we would prefer to always have strong gradients from the discriminator during training, the vanilla GAN makes this difficult to ensure, as the discriminator quickly learns to distinguish real and generated samples (Goodfellow, 2016), thus providing no meaningful error signal to improve the generator thereafter. Durugkar et al. (2016) proposed the

Generative Multi-Adversarial Networks (GMAN) which consists of training the generator against a *softmax* weighted arithmetic average of K different discriminators:

$$\mathcal{L}_G = \sum_{k=1}^{K} \alpha_k(-\mathcal{L}_{D_k}), \tag{4}$$

where $\alpha_k = \frac{e^{\beta(-\mathcal{L}_{D_k})}}{\sum_{j=1}^{K} e^{\beta(-\mathcal{L}_{D_j})}}$, $\beta \geq 0$, and $\mathcal{L}_{D_k}$ is the loss of discriminator k and is defined as:

$$\mathcal{L}_{D_k} = -\mathbb{E}_{\mathbf{x}\sim p_{\text{data}}} \log(D_k(\mathbf{x})) - \mathbb{E}_{\mathbf{z}\sim p_z} \log(1 - D_k(G(\mathbf{z}))). \tag{5}$$

$D_k(\mathbf{x})$ and $G(\mathbf{z})$ correspond to the outputs of the k-th discriminator and the generator, respectively. The goal of using the proposed averaging scheme is to favor discriminators yielding higher losses to the generator (i.e. high $-\mathcal{L}_{D_k}$), thus providing more useful gradients during training. Experiments were performed with $\beta = 0$ (equal weights), $\beta \to \infty$ (only worst discriminator is taken into account), $\beta = 1$, and β learned by the generator. Models with $K = \{2, 5\}$ were tested and evaluated using a proposed metric and the Inception score (Salimans et al., 2016). Results showed that the simple average of discriminator's losses provided the best values for both metrics in most of the considered cases.

Neyshabur et al. (2017) proposed training a GAN with K discriminators using the same architecture. Each discriminator D_k sees a different randomly projected lower-dimensional version of the input image. Random projections are defined by a randomly initialized matrix W_k, which remains fixed during training. Theoretical results provided show the distribution induced by the generator G will converge to the real data distribution p_{data}, as long as there is a sufficient number of discriminators. Moreover, discriminative tasks in the projected space are harder, i.e. real and fake examples are more alike, thus avoiding early convergence of discriminators, which leads to common stability issues in GAN training such as mode-collapse (Goodfellow, 2016). Essentially, the authors trade one hard problem for K easier subproblems. The losses of each discriminator $\mathcal{L}_{D_k}$ are the same as shown in Eq. 5. However, the generator loss $\mathcal{L}_G$ is defined as the sum of the losses provided by each discriminator, as shown in Eq. 6. This choice of $\mathcal{L}_G$ does not exploit available information such as the performance of the generator with respect to each discriminator.

$$\mathcal{L}_G = -\sum_{k=1}^{K} \mathbb{E}_{\mathbf{z}\sim p_z} \log D_k(G(\mathbf{z})). \tag{6}$$

3.2. Hypervolume maximization

Let S be the solutions for a multi-objective optimization problem. The hypervolume $\mathcal{H}$ of S is defined as (Fleischer, 2003): $\mathcal{H}(S) = \mu(\cup_{\mathbf{x}\in S}[\mathbf{F}(\mathbf{x}), \boldsymbol{\eta}^*])$, where μ is the Lebesgue measure and $\boldsymbol{\eta}^*$ is a point dominated by all $\mathbf{x} \in S$ (i.e. $f_i(\mathbf{x})$ is upper-bounded by η), referred to as the *nadir point*. $\mathcal{H}(S)$ can be understood as the size of the space covered by $\{\mathbf{F}(\mathbf{x}) \mid \mathbf{x} \in S\}$ (Bader & Zitzler, 2011).

The hypervolume was originally introduced as a quantitative metric for coverage and convergence of Pareto-optimal fronts obtained through population-based algorithms (Beume et al., 2007). Methods based on direct maximization of $\mathcal{H}$ exhibit favorable convergence even in challenging scenarios, such as simultaneous minimization of 50 objectives (Bader & Zitzler, 2011). In the context of Machine Learning, single-solution HVM has been applied to neural networks as a surrogate loss for mean squared error (Miranda & Von Zuben, 2016), i.e. the loss provided by each example in a training batch is treated as a single cost and the multi-objective approach aims to minimize costs over all examples. Authors show that such method provides an inexpensive boosting-like training.

4. Multi-objective training of GANs with multiple discriminators

We introduce a variation of the GAN game in which the generator solves the following multi-objective problem:

$$\min \boldsymbol{\mathcal{L}}_G(\mathbf{z}) = [l_1(\mathbf{z}), l_2(\mathbf{z}), ..., l_K(\mathbf{z})]^T, \tag{7}$$

where each $l_k = -\mathbb{E}_{z\sim p_z} \log D_k(G(z))$, $k \in \{1, ..., K\}$, is the loss provided by the k-th discriminator. Training proceeds in the usual fashion (Goodfellow et al., 2014), i.e. with alternate updates between the discriminators and the generator. Updates of each discriminator are performed to minimize the loss described in Eq. 5.

A natural choice for our generator's updates is the MGD algorithm, described in Section 2. However, computing the direction of steepest descent $\mathbf{w}^*$ before every parameter update step, as required in MGD, can be prohibitively expensive for large neural networks. Therefore, we propose an alternative scheme for multi-objective optimization and argue that both our proposal and previously published methods can all be viewed as performing a computationally more efficient version of the MGD update rule, without the burden of needing to solve a quadratic program, i.e. computing $\mathbf{w}^*$, every iteration.

4.1. Hypervolume maximization for training GANs

Fleischer (2003) has shown that maximizing $\mathcal{H}$ yields Pareto-optimal solutions. Since MGD converges to a set of Pareto-stationary points, i.e. a superset of the Pareto-optimal solutions, HVM yields a subset of the solutions obtained using MGD. We exploit this property and define the generator

loss as the negative log-hypervolume, as defined in Eq. 8:

$$\mathcal{L}_G = -\mathcal{V} = -\sum_{k=1}^{K} \log(\eta - l_k), \tag{8}$$

where the nadir point coordinate η is an upper bound for all l_k. In Fig. 1 we provide an illustrative example for the case where $K = 2$. The highlighted region corresponds to $e^{\mathcal{V}}$. Since the nadir point $\boldsymbol{\eta}^*$ is fixed, $\mathcal{V}$ will be maximized, and consequently $\mathcal{L}_G$ minimized, if and only if each l_k is minimized. Moreover, by adapting the results shown in (Miranda & Von Zuben, 2016), the gradient of $\mathcal{L}_G$ with respect to any generator's parameter θ is given by:

$$\frac{\partial \mathcal{L}_G}{\partial \theta} = \sum_{k=1}^{K} \frac{1}{\eta - l_k} \frac{\partial l_k}{\partial \theta}. \tag{9}$$

In other words, the gradient can be obtained by computing a weighted sum of the gradients of the losses provided by each discriminator, whose weights are defined as the inverse distance to the nadir point components. This formulation will naturally assign more importance to higher losses in the final gradient, which is another useful property of HVM.

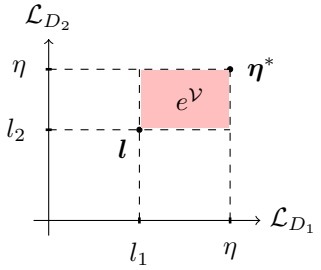


Figure 1: 2D example of the objective space where the generator loss is being optimized.

Nadir point selection. It is evident from Eq. 9 that the selection of η directly affects the importance assignment of gradients provided by different discriminators. Particularly, as the quantity $\min_k\{\eta - l_k\}$ grows, the multi-objective GAN game approaches the one defined by the simple average of l_k. Previous literature has discussed in depth the effects of the selection of η in the case of population-based methods (Auger et al., 2009; 2012). However, those results are not readily applicable for the single-solution case. As will be shown in Section 5, our experiments indicate that the choice of η plays an important role in the final quality of samples. Nevertheless, this effect becomes less relevant as the number of discriminators increases.

Nadir point adaptation. Similarly to (Miranda & Von Zuben, 2016), we propose an adaptive scheme for η such that at iteration t: $\eta^t = \delta \max_k\{l_k^t\}$, where $\delta > 1$ is a user-defined parameter which will be referred to as *slack*. This enforces $\min_k\{\eta^t - l_k^t\}$ to be higher when $\max_k\{l_k^t\}$ is high and low otherwise, which induces a similar behavior as an average loss when training begins and automatically places more importance on the discriminators in which performance is worse as training progresses.

We further illustrate the proposed adaptation scheme in Fig. 2. Consider a two-objective problem with $l_1^t > 0$ and $l_2^t > 0$ corresponding to $\mathcal{L}_{D_1}$ and $\mathcal{L}_{D_2}$ at iteration t, respectively. If no adaptation is performed and η is left unchanged throughout training, as represented by the red dashed lines in Fig. 2, $\eta - l_1^t \approx \eta - l_2^t$ for a large enough t. This will assign similar weights to gradients provided by the different losses, which defeats the purpose of employing HVM rather than average loss minimization. Assuming that losses decrease with time, after T updates, $\eta^T = \delta \max\{l_1^T, l_2^T\} < \eta$, since losses are now closer to 0. The employed adaptation scheme thus keeps the gradient weighting relevant even when losses become low. This effect will become more aggressive as training progresses, assigning more gradient importance to higher losses, as $\eta^T - \max\{l_1^T, l_2^T\} < \eta^0 - \max\{l_1^0, l_2^0\}$.

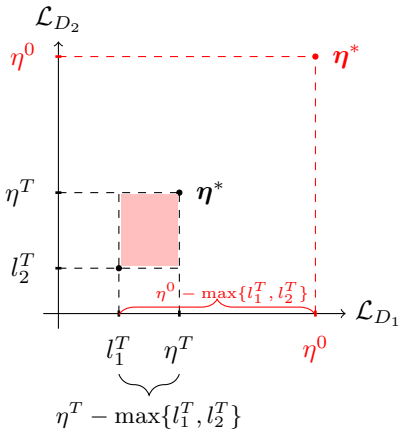


Figure 2: Losses and nadir point at $t = T$, and nadir point at $t = 0$ (in red).

Comparison to average loss minimization. The upper bound proven by Neyshabur et al. (2017) assumes that the marginals of the real and generated distributions are identical along all random projections. However, average loss minimization does not ensure equally good approximation between the marginals in all directions. In the case of competing discriminators, i.e. when decreasing the loss on one projection increases the loss on another, the distribution of losses can be uneven. With HV on the other hand, especially when η is reduced during training, the overall loss will remain high as long as there are discriminators with high loss. This objective tends to prefer central regions, where all discriminators present roughly equally low losses.

4.2. Relationship between multiple discriminator GANs and MGD

All methods described previously for the solution of GANs with multiple discriminators, i.e. average loss minimization (Neyshabur et al., 2017), GMAN's weighted average (Durugkar et al., 2016) and HVM can be defined as MGD-like two-step algorithms consisting of: *Step 1* - consolidate all gradients into a single update direction (compute the set $\alpha_{1,...,K}$); *Step 2* - update parameters in the direction returned in Step 1. The definition of *Step 1* for the considered methods can be summarized as follows:

1. MGD: $\alpha_{1:K} = \text{argmin}_\alpha ||\mathbf{w}||, \quad \text{s.t.} \quad \sum_{k=1}^{K} \alpha_k = 1, \alpha_k \geq 0 \; \forall k \in \{1, ..., K\}$
2. Average loss minimization (Neyshabur et al., 2017): $\alpha_k = \frac{1}{K}$
3. GMAN (Durugkar et al., 2016): $\alpha_k = \text{softmax}(l_{1:K})_k$
4. Hypervolume maximization: $\alpha_k = \frac{1}{T(\eta - l_k)}$, $T = \sum_{k=1}^{K} \frac{1}{\eta - l_k}$

5. Experiments

We performed four sets of experiments aiming to understand the following phenomena: (i) How alternative methods for training GANs with multiple discriminators perform in comparison to MGD; (ii) How alternative methods perform in comparison to each other in terms of sample quality and coverage; (iii) How the varying number of discriminators impacts performance given the studied methods; and (iv) Whether the multiple-discriminator setting is practical given the added cost involved in training a set of discriminators.

Firstly, we exploited the relatively low dimensionality of MNIST and used it as testbed for comparing MGD with the other approaches, i.e. average loss minimization (AVG), GMAN's weighted average loss, and HV, proposed in this work. Moreover, multiple initializations and *slack* combinations were evaluated in order to investigate how varying the number of discriminators affects robustness to those factors. Then, experiments were performed with an upscaled version of CIFAR-10 at the resolution of 64x64 pixels while increasing the number of discriminators. Upscaling was performed with the aim of running experiments utilizing the same architecture described in (Neyshabur et al., 2017). We evaluated HV's performance compared to baseline methods in terms of its resulting sample quality. Additional experiments were carried out with CIFAR-10 at its original resolution in order to provide a clear comparison with well known single-discriminator settings. We further analyzed HV's impact on the diversity of generated samples using the stacked MNIST dataset (Srivastava et al., 2017). Finally, the computational cost and performance are compared for the single- vs. multiple-discriminator cases. Samples of generators trained on stacked MNIST and CIFAR-10 are presented in the Appendix along with samples from CelebA (at two different resolutions) as well as the Cats dataset at a 256×256 resolution.

In all experiments performed, the same architecture, set of hyperparameters and initialization were used for both AVG, GMAN and our proposed method, the only variation being the generator loss. Unless stated otherwise, Adam (Kingma & Ba, 2014) was used to train all the models with learning rate, β_1 and β_2 set to 0.0002, 0.5 and 0.999, respectively. Mini-batch size was set to 64. The Fréchet Inception Distance (FID) (Heusel et al., 2017) was used for comparison. Details on FID computation can be found in Appendix A.

5.1. MGD compared with alternative methods

We employed MGD in our experiments with MNIST and, in order to do so, a quadratic program has to be solved prior to every parameters update. For this, we used Scipy's implementation of the Serial Least Square Quadratic Program solver. Three and four fully connected layers with *LeakyReLU* activations were used for the generator and discriminator, respectively. Dropout was also employed in the discriminator and the random projection layer was implemented as a randomly initialized norm-1 fully connected layer, reducing the vectorized dimensionality of MNIST from 784 to 512. The output layer of a pretrained *LeNet* (LeCun et al., 1998) was used for FID computation. Experiments over 100 epochs with 8 discriminators are reported in Fig. 3 and Fig. 4. In Fig. 3, box-plots refer to 30 independent computations of FID over 10000 images sampled from the generator which achieved the minimum FID at train time. FID results are measured at training time with 1000 images and the best values are reported in Fig. 4 along with the necessary time to achieve it.

MGD outperforms all tested methods. However, its cost per iteration does not allow its use in more relevant datasets outside MNIST. HV, on the other hand, performs closer to MGD than the considered baselines, while introducing no relevant extra cost. In Fig. 5, we analyze convergence in the Pareto-stationarity sense by plotting the norm of the update direction for each method, given by $||\sum_{k=1}^{K} \alpha_k \nabla l_k||$. All methods converged to similar norms, leading to the conclusion that different Pareto-stationary solutions will result in generators with distinct sample quality. Best FID as a function of wall-clock time is shown in Fig. 20 at the Appendix.

HV sensitivity to initialization and choice of δ. Analysis of the performance sensitivity with the choice of the slack parameter δ and initialization was performed under the following setting: models were trained for 50 epochs on MNIST with HVM using 8, 16, 24 discriminators. Three

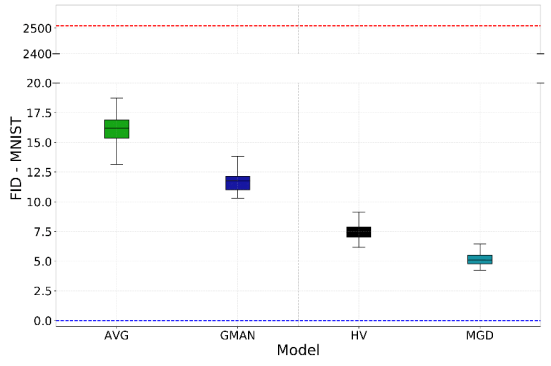


Figure 3: Box-plots corresponding to 30 independent FID computations with 10000 images. MGD performs consistently better than other methods, followed by HVM. Models that achieved minimum FID at training time were used. Red and blue dashed lines represent FID values for a random generator and real data, respectively.

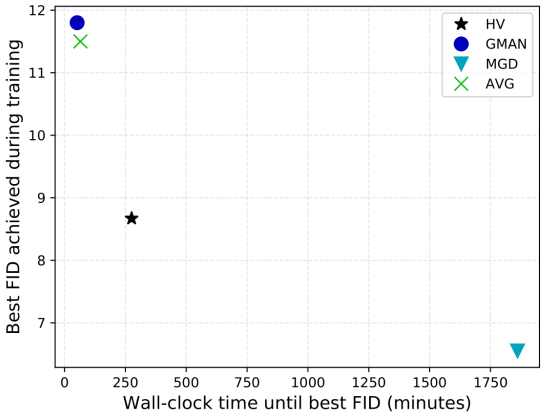


Figure 4: Time vs. best FID achieved during training for each approach. FID values are computed over 1000 generated images after every epoch. MGD performs relevantly better than others in terms of FID, followed by HV. However, MGD is approximately 7 times slower than HV. HV is well-placed in the time-quality trade-off.

independent runs (different initializations) were executed with each $\delta = \{1.05, 1.5, 1.75, 2\}$ and number of discriminators, totaling 36 final models. Fig. 6 reports the box-plots obtained for 5 FID independent computations using 10000 images, for each of the 36 models obtained under the setting described. Results indicate that increasing the number of discriminators yields much smaller variation in the FID obtained by the final model.

5.2. HV as an alternative for MGD

5.2.1. Upscaled CIFAR-10

We evaluate the performance of HV compared to baseline methods using the upscaled CIFAR-10 dataset. FID was computed with a pretrained ResNet (He et al., 2016). ResNet was trained on the 10-class classification task of CIFAR-10 up to approximately 95% test accuracy. DCGAN (Radford et al., 2015) and WGAN-GP (Gulrajani et al., 2017) were included in the experiments for FID reference. Same architectures as in (Neyshabur et al., 2017) were employed for all multi-discriminators settings. An increasing number of discriminators was used. Inception score (IS) (Salimans et al., 2016) as well as FID computed with other models are included in the Appendix-Table 7.

In Fig. 7, we report the box-plots of 15 independent evaluations of FID on 10000 images for the best model obtained with each method across 3 independent runs. Results once more indicate that HV outperforms other methods in terms of quality of the generated samples. Moreover, performance clearly improves as the number of discriminators grows. Fig. 8 shows the FID at train time, i.e. measured with 1000 generated images after each epoch, for the best models across

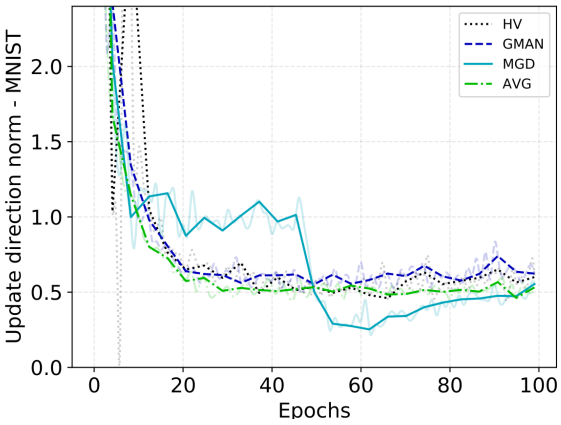


Figure 5: Norm of the update direction over time for each method. While Pareto-stationarity is approximately achieved by all methods, performance varies relevantly in terms of FID.

runs. Models trained against more discriminators converge to smaller values. We report the norm of the update direction $||\sum_{k=1}^{K} \alpha_k \nabla l_k||$ for each method in Fig. 10-(a) in the Appendix.

5.2.2. CIFAR-10

We run experiments with CIFAR-10 in its original resolution, aiming to put our proposed approach in context with previous methods. Similar experiments as described by Miyato et al. (2018) can be found in Table 2, for the model

	FID-ResNet	FID (5k)	IS (5k)	FID (10k)	IS (10k)
SNGAN (Miyato et al., 2018)	-	25.5	7.58 ± 0.12	-	-
WGAN-GP (Miyato et al., 2018)	-	40.2	6.68 ± 0.06	-	-
DCGAN (Miyato et al., 2018)	-	-	6.64 ± 0.14	-	-
SNGAN (our implementation)	1.55	27.93	7.11 ± 0.30	25.29	7.26 ± 0.12
DCGAN + 24 Ds and HV	1.21	27.74	7.32 ± 0.26	24.90	7.45 ± 0.17

Table 1: An evaluation of the effect of adding discriminators on a DCGAN-like model trained on CIFAR-10. Results reach the same level as the best-reported scores for the given architecture in the multiple-discriminator setting.

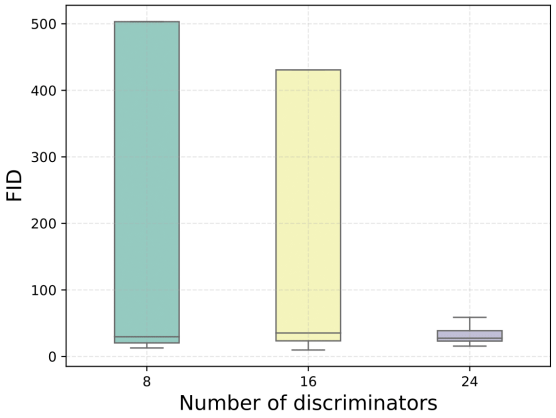


Figure 6: Independent FID evaluations for models obtained with different initializations and slack parameter δ. Sensitivity reduces as the number of discriminators increases.

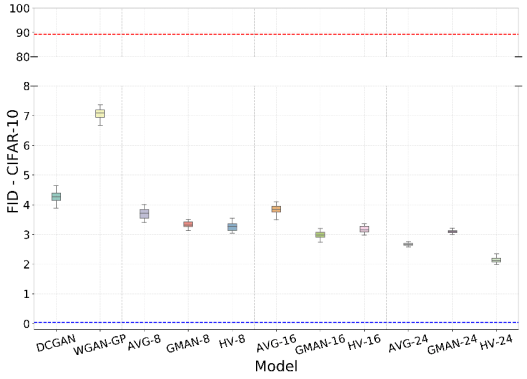


Figure 7: Box-plots of 15 independent FID computations with 10000 images. Dashed lines represent the FID for real data (blue) and a random generator (red). FID was computed with a pretrained ResNet.

Miyato et al. (2018) refer to as *Standard CNN*. The same architecture is employed and spectral normalization is removed from the discriminators, while a random projection input layer is added.

FID and IS are evaluated on 5000 generated images as in (Miyato et al., 2018) as well as 10000 images, as reported in Table 1. These results include our proposed approach and implementation of (Miyato et al., 2018), alongside the FID measured using a ResNet classifier trained in advance on the CIFAR-10 dataset.

As can be seen, the addition of the multiple discriminators setting along with HV yields a relevant shift in performance for the DCGAN-like generator, improving the evaluated metrics while the generator architecture was kept unchanged. Both IS and FID improved relative to WGAN-GP, while outperforming our own implementation of SNGAN. It is worth noting that for this experiment we selected the best performing set of hyperparameters for SNGAN, following the reported setting in prior literature (Miyato et al., 2018).

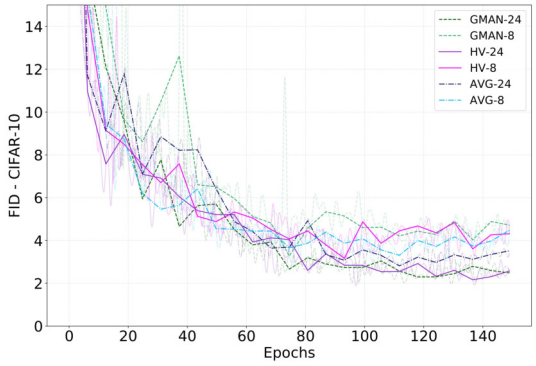


Figure 8: FID estimated over 1000 generated images at train time. Models trained against more discriminators achieve lower FID. FID was computed with a pretrained ResNet.

5.3. Computational cost

In Table 2 we present a comparison of minimum FID (measured with a pretrained ResNet) obtained during training, along with computation cost in terms of time and space for different GANs, with both 1 and 24 discriminators.

By design, the computational cost of training GANs under a multiple-discriminator setting is higher in terms of both FLOPS and memory, if compared with the single-discriminator GAN setting. However, the additional cost results in a corresponding improvement in performance. This effect was consistently observed using 3 different well-known approaches, namely DCGAN (Radford et al., 2015), Least-square GAN (LSGAN) (Mao et al., 2017), and Hinge-GAN (Miyato et al., 2018). The architectures of all single discriminator models follow that of DCGAN, described in (Radford et al., 2015). For the 24-discriminator models, we used the setting described in Section 5.2.1. All models were trained with a minibatch of size 64 over 150 epochs.

We further emphasize that even though training with multiple discriminators may be more computationally expensive when compared to conventional approaches, such a framework supports fully parallel training of the discriminators, a feature which is not trivially possible in other GAN settings. For example in WGAN, the discriminator is serially updated multiple times for each generator update. In Fig. 10-(b) in the Appendix, we provide a comparison in terms of wall-clock time per iteration on all methods evaluated. Serial implementations of discriminator updates with 8 and 16 discriminators were observed to run faster than WGAN-GP. Moreover, all experiments performed within this work were executed in single-GPU hardware, which indicates the multiple discriminator setting is a practical approach.

	# Disc.	FID-ResNet	FLOPS	Memory
DCGAN	1	4.22	8e10	1292
	24	1.89	5e11	5671
LSGAN	1	4.55	8e10	1303
	24	1.91	5e11	5682
HingeGAN	1	6.17	8e10	1303
	24	2.25	5e11	5682

Table 2: Comparison between different GANs with 1 and 24 discriminators in terms of minimum FID-ResNet obtained during training, and FLOPs (MAC) and memory consumption (MB) for a complete training step.

5.4. Effect of the number of discriminators on sample diversity

We repeat the experiments in (Srivastava et al., 2017) aiming to analyze how the number of discriminators affects the sample diversity of the corresponding generator when trained using the HV algorithm. The stacked MNIST dataset is employed and results reported in (Lin et al., 2017) are used for comparison. HV results for 8, 16, and 24 discriminators were obtained with 10k and 26k generator images, averaged over 10 runs. The number of covered modes along with the KL divergence between the generated mode distribution and test data are reported in Table 3.

Model	Modes (Max 1000)	KL
DCGAN (Radford et al., 2015)	99.0	3.400
ALI (Dumoulin et al., 2016)	16.0	5.400
Unrolled GAN (Metz et al., 2016)	48.7	4.320
VEEGAN (Srivastava et al., 2017)	150.0	2.950
PacDCGAN2 (Lin et al., 2017)	1000.0 ± 0.0	0.060 ± 0.003
HV - 8 disc. (10k)	679.2 ± 5.9	1.139 ± 0.011
HV - 16 disc. (10k)	998.0 ± 1.8	0.120 ± 0.004
HV - 24 disc. (10k)	998.3 ± 1.1	0.116 ± 0.003
HV - 8 disc. (26k)	776.8 ± 6.4	1.115 ± 0.007
HV - 16 disc. (26k)	1000.0 ± 0.0	0.088 ± 0.002
HV - 24 disc. (26k)	1000.0 ± 0.0	0.084 ± 0.002

Table 3: Number of covered modes and reverse KL divergence for stacked MNIST. We evaluate HV under a reduced test sample size (10k) with the goal of highlighting the effect provided by the increased number of discriminators on sample diversity.

As in previous experiments, results consistently improved as we increased the number of discriminators. All evaluated models using HV outperformed DCGAN, ALI, Unrolled GAN and VEEGAN. Moreover, HV with 16 and 24 discriminators achieved state-of-the-art coverage values. Thus, increasing each model's capacity by using more discriminators directly resulted in an improvement in the corresponding generator coverage. Training details as well as architecture information are presented in Appendix B.

6. Conclusion

In this work we show that employing multiple discriminators on GAN training is a practical approach for directly trading extra capacity - and thereby extra computational cost - for higher quality and diversity of generated samples. Such an approach is complementary to other advances in GANs training and can be easily used in tandem with other methods. We thus introduce a multi-objective optimization framework for studying multiple discriminator GANs, and showed strong similarities between previous work using such setting and the MGD algorithm. The proposed approach, namely a single-solution variation of the hypervolume maximization, was observed to consistently yield higher quality samples in terms of FID when compared to average loss and GMAN's aggregation rule. We further observed a higher number of discriminators to increase sample diversity and generator robustness.

Deeper analysis of the quantity $||\sum_{k=1}^{K} \alpha_k \nabla l_k||$ is a subject of future investigation. We hypothesize that using it as a penalty term might reduce the necessity of a high number of discriminators. Moreover, we believe the proposed HV method might be of independent interest in other problems in Machine Learning literature that rely on the minimization of a set of loss terms.

Acknowledgements

We thank Sandeep Subramanian, Gauthan Swaminathan, Vikas Verna, and Vikram Voleti for the feedback and helpful comments. We are also grateful to Frédéric Bastien and Simon Lefrançois for the support with setting up the computational resources used in part of this work. We would also like to thank the reviewers for their valuable suggestions.

References

Arjovsky, M., Chintala, S., and Bottou, L. Wasserstein GAN. *arXiv preprint arXiv:1701.07875*, 2017.

Auger, A., Bader, J., Brockhoff, D., and Zitzler, E. Theory of the hypervolume indicator: optimal μ-distributions and the choice of the reference point. In *Proceedings of the tenth ACM SIGEVO workshop on Foundations of genetic algorithms*, pp. 87–102. ACM, 2009.

Auger, A., Bader, J., Brockhoff, D., and Zitzler, E. Hypervolume-based multiobjective optimization: Theoretical foundations and practical implications. *Theoretical Computer Science*, 425:75–103, 2012.

Bader, J. and Zitzler, E. HypE: An algorithm for fast hypervolume-based many-objective optimization. *Evolutionary computation*, 19(1):45–76, 2011.

Berthelot, D., Schumm, T., and Metz, L. BEGAN: Boundary equilibrium generative adversarial networks. *arXiv preprint arXiv:1703.10717*, 2017.

Beume, N., Naujoks, B., and Emmerich, M. SMS-EMOA: Multiobjective selection based on dominated hypervolume. *European Journal of Operational Research*, 181(3): 1653–1669, 2007.

Deb, K. *Multi-objective optimization using evolutionary algorithms*, volume 16. John Wiley & Sons, 2001.

Désidéri, J.-A. Multiple-gradient descent algorithm (MGDA) for multiobjective optimization. *Comptes Rendus Mathematique*, 350(5-6):313–318, 2012.

Dumoulin, V., Belghazi, I., Poole, B., Mastropietro, O., Lamb, A., Arjovsky, M., and Courville, A. Adversarially learned inference. *arXiv preprint arXiv:1606.00704*, 2016.

Durugkar, I., Gemp, I., and Mahadevan, S. Generative multi-adversarial networks. *arXiv preprint arXiv:1611.01673*, 2016.

Fleischer, M. The measure of pareto optima applications to multi-objective metaheuristics. In *International Conference on Evolutionary Multi-Criterion Optimization*, pp. 519–533. Springer, 2003.

Fréchet, M. Sur la distance de deux lois de probabilité. *COMPTES RENDUS HEBDOMADAIRES DES SEANCES DE L ACADEMIE DES SCIENCES*, 244(6): 689–692, 1957.

Goodfellow, I. NIPS 2016 tutorial: Generative adversarial networks. *arXiv preprint arXiv:1701.00160*, 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*, pp. 2672–2680, 2014.

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. 5769–5779, 2017.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

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. 6629–6640, 2017.

Jolicoeur-Martineau, A. The relativistic discriminator: a key element missing from standard gan. *arXiv preprint arXiv:1807.00734*, 2018.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Lin, Z., Khetan, A., Fanti, G., and Oh, S. PacGAN: The power of two samples in generative adversarial networks. *arXiv preprint arXiv:1712.04086*, 2017.

Mao, X., Li, Q., Xie, H., Lau, R. Y., Wang, Z., and Smolley, S. P. Least squares generative adversarial networks. In *Computer Vision (ICCV), 2017 IEEE International Conference on*, pp. 2813–2821. IEEE, 2017.

Metz, L., Poole, B., Pfau, D., and Sohl-Dickstein, J. Unrolled generative adversarial networks. *arXiv preprint arXiv:1611.02163*, 2016.

Miranda, C. S. and Von Zuben, F. J. Single-solution hypervolume maximization and its use for improving generalization of neural networks. *arXiv preprint arXiv:1602.01164*, 2016.

Miyato, T., Kataoka, T., Koyama, M., and Yoshida, Y. Spectral normalization for generative adversarial networks. *arXiv preprint arXiv:1802.05957*, 2018.

Neyshabur, B., Bhojanapalli, S., and Chakrabarti, A. Stabilizing GAN training with multiple random projections. *arXiv preprint arXiv:1705.07831*, 2017.

Peitz, S. and Dellnitz, M. Gradient-based multiobjective optimization with uncertainties. In *NEO 2016*, pp. 159–182. Springer, 2018.

Radford, A., Metz, L., and Chintala, S. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 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.

Schäffler, S., Schultz, R., and Weinzierl, K. Stochastic method for the solution of unconstrained vector optimization problems. *Journal of Optimization Theory and Applications*, 114(1):209–222, 2002.

Sener, O. and Koltun, V. Multi-task learning as multi-objective optimization. In *Advances in Neural Information Processing Systems*, pp. 527–538, 2018.

Srivastava, A., Valkoz, L., Russell, C., Gutmann, M. U., and Sutton, C. VEEGAN: Reducing mode collapse in GANs using implicit variational learning. In *Advances in Neural Information Processing Systems*, pp. 3310–3320, 2017.

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., and Wojna, Z. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2818–2826, 2016.

Appendix

A - Objective evaluation metric.

In (Heusel et al., 2017), authors proposed to use as a quality metric the squared Fréchet distance (Fréchet, 1957) between Gaussians defined by estimates of the first and second order moments of the outputs obtained through a forward pass in a pretrained classifier of both real and generated data. They proposed the use of Inception V3 (Szegedy et al., 2016) for computation of the data representation and called the metric Fréchet Inception Distance (FID), which is defined as:

$$\text{FID} = ||m_d - m_g||^2 + \text{Tr}(\Sigma_d + \Sigma_g - 2(\Sigma_d\Sigma_g)^{\frac{1}{2}}), \tag{10}$$

where m_d, Σ_d and m_g, Σ_g are estimates of the first and second order moments from the representations of real data distributions and generated data, respectively.

We employ FID throughout our experiments for comparison of different approaches. However, in datasets other than CIFAR-10 at its original resolution, for each dataset in which FID was computed, the output layer of a pretrained classifier on that particular dataset was used instead of Inception. m_d and Σ_d were estimated on the complete test partitions, which are not used during training.

B - Experimental setup for stacked MNIST experiments and generator's samples

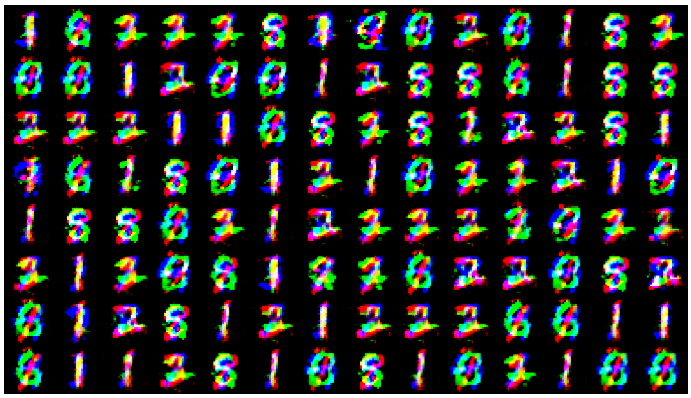
Architectures of the generator and discriminator are detailed in Tables 4 and 5, respectively. Batch normalization was used in all intermediate convolutional and fully connected layers of both models. We employed RMSprop to train all the models with learning rate and α set to 0.0001 and 0.9, respectively. Mini-batch size was set to 64. The setup in (Lin et al., 2017) is employed and we build 128000 and 26000 samples for train and test sets, respectively.

Layer	Outputs	Kernel size	Stride	Activation
Input: $z \sim \mathcal{N}(0, I_{100})$				
Fully connected	2*2*512	4, 4	2, 2	ReLU
Transposed convolution	4*4*256	4, 4	2, 2	ReLU
Transposed convolution	8*8*128	4, 4	2, 2	ReLU
Transposed convolution	14*14*64	4, 4	2, 2	ReLU
Transposed convolution	28*28*3	4, 4	2, 2	Tanh

Table 4: Generator's architecture.

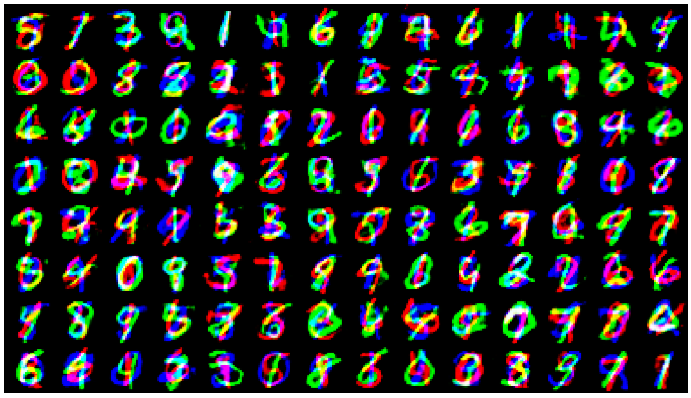
Layer	Outputs	Kernel size	Stride	Activation
Input	28*28*3			
Projection	14*14*3	8, 8	2, 2	
Convolution	7*7*64	4,4	2, 2	LeakyReLU
Convolution	5*5*128	4, 4	2, 2	LeakyReLU
Convolution	2*2*256	4, 4	2, 2	LeakyReLU
Convolution	1	4, 4	2, 2	Sigmoid

Table 5: Discriminator's architecture.



(a) HV - 8 discriminators

(b) HV - 16 discriminators



(c) HV - 24 discriminators

Figure 9: Stacked MNIST samples for HV trained with 8, 16, and 24 discriminators. Samples diversity increases greatly when more discriminators are employed.

C - Extra results on upscaled CIFAR-10

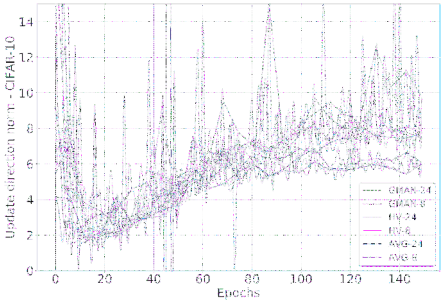
C.1 - Multiple discriminators across different initializations and other scores

Table 6 presents the best FID (computed with a pretrained ResNet) achieved by each approach at train time, along with the epoch in which it was achieved, for each of 3 independent runs. Train time FIDs are computed using 1000 generated images.

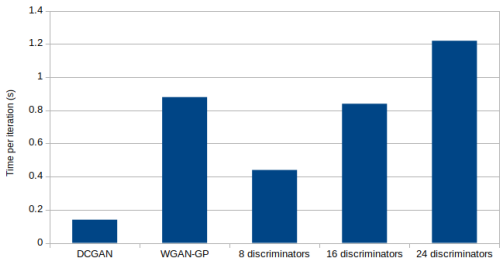
#D	Method	Best FID (epoch)
1	DCGAN	7.09 (68), 9.09 (21), 4.22 (101)
	WGAN-GP	5.09 (117), 5.69 (101) 7.13 (71)
8	AVG	3.35 (105), 4.64 (141), 3.00 (76)
	GMAN	4.28 (123), 4.24 (129), 3.80 (133)
	HV	3.87 (102), 4.54 (82), 3.20 (98)
16	AVG	3.16 (96), 2.50 (91), 2.77 (116)
	GMAN	2.69 (129), 2.36 (144), 2.48 (120)
	HV	2.56 (85), 2.70 (97), 2.68 (133)
24	AVG	2.10 (94), 2.44 (132), 2.43 (129)
	GMAN	2.16 (120), 2.02 (98), 2.13 (130)
	HV	2.05 (83), 1.89 (97), 2.23 (130)

Table 6: Best FID obtained for each approach on 3 independent runs. FID is computed on 1000 generated images after every epoch.

In Fig. 10-(a), we report the norm of the update direction $||\sum_{k=1}^{K} \alpha_k \nabla l_k||$ of the best model obtained for each method. Interestingly, different methods present similar behavior in terms of convergence in the Pareto-stationary sense, i.e. the norm upon convergence is lower for models trained against more discriminators, regardless of the employed method.



(a) Norm of the update direction over time for each method. Higher number of discriminators yield lower norm upon convergence.



(b) Time in seconds per iteration of each method for serial updates of discriminators. The different multiple discriminators approaches considered do not present relevant difference in time per iteration.

Figure 10: (a) Norm of update direction. (b) Time per iteration.

We computed extra scores using 10000 images generated by the best model reported in Table 6, i.e. the same models utilized to generate the results shown in Fig. 7. Both IS and FID were computed with original implementations, while FID-VGG and FID-ResNet were computed using a VGG and a ResNet we pretrained. Results are reported with respect to DCGAN's scores to avoid direct comparison with results reported elsewhere for CIFAR-10 on its usual resolution (32×32).

	WGAN-GP	AVG-8	AVG-16	AVG-24	GMAN-8	GMAN-16	GMAN-24	HV-8	HV-16	HV-24
Inception Score	1.08	1.02	1.26	1.36	0.95	1.32	1.42	1.00	1.30	**1.44**
FID	0.80	0.98	0.76	0.73	0.92	0.79	**0.65**	0.89	0.77	0.72
FID-VGG	1.29	0.91	1.03	0.85	0.87	0.78	0.73	0.78	0.75	**0.64**
FID-ResNet	1.64	0.88	0.90	0.62	0.80	0.72	0.73	0.75	0.73	**0.51**

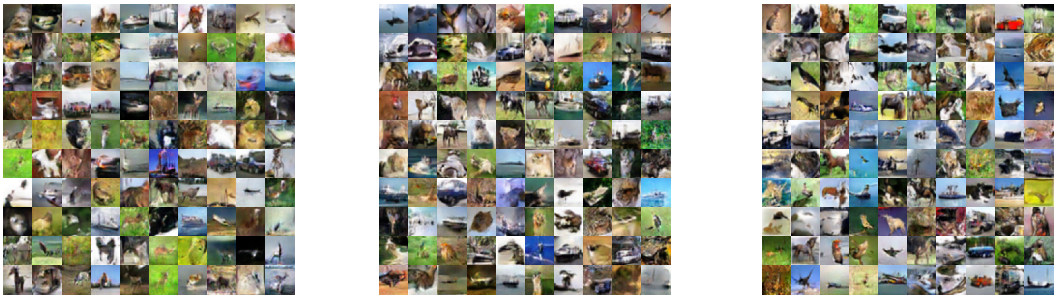
Table 7: Scores of different methods measure on generated samples from the upsacaled CIFAR-10. DCGAN scores are used as reference values, and results report are the ratio between given model and DCGAN scores. IS is better when high, whereas FIDs are better when low.

C.3 - Generated samples

In Figs. 11, 12, and 13 we show random generated samples with 8, 16, and 24 discriminators for AVG, GMAN, and HV, respectively.

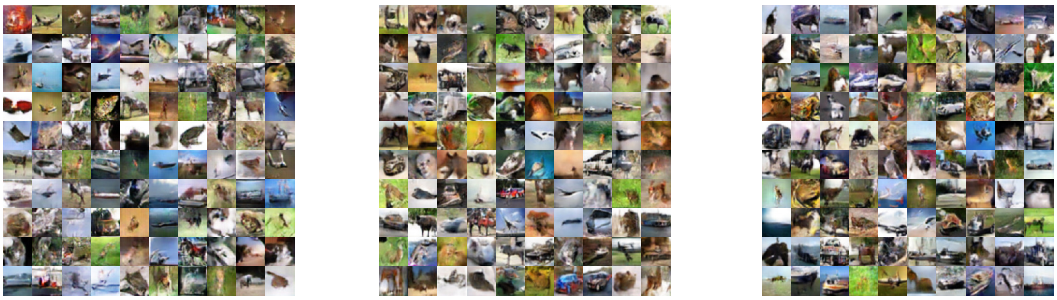
(a) AVG - 8 discriminators (b) AVG - 16 discriminators (c) AVG - 24 discriminators

Figure 11: CIFAR-10 samples for AVG trained with 8, 16, and 24 discriminators.



(a) GMAN - 8 discriminators (b) GMAN - 16 discriminators (c) GMAN - 24 discriminators

Figure 12: CIFAR-10 samples for GMAN trained with 8, 16, and 24 discriminators.



(a) HV - 8 discriminators (b) HV - 16 discriminators (c) HV - 24 discriminators

Figure 13: CIFAR-10 samples for HV trained with 8, 16, and 24 discriminators.

D - CelebA dataset

D.1 - Comparing with other multiple-discriminators approaches

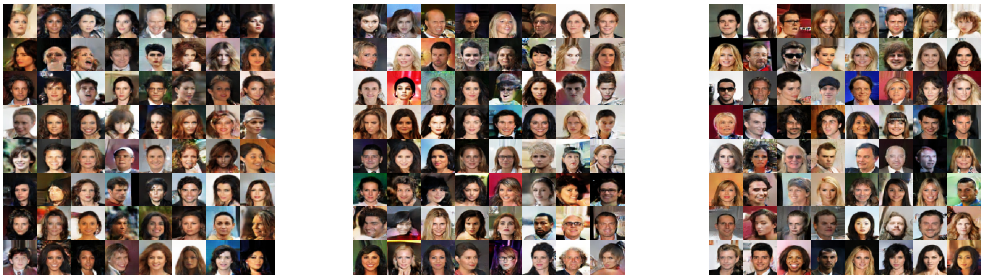
Here, we present samples obtained by generators trained against 8, 16, and 24 discriminators using AVG, GMAN, and HV on the CelebA dataset rescaled to 64x64. Training lasted 100 epochs and samples are shown in Figs. 14, 15, and 16 for AVG, GMAN and HV, respectively. Same setup used for experiments with CIFAR-10 presented in Section 5 were utilized.

(a) AVG - 8 discriminators (b) AVG - 16 discriminators (c) AVG - 24 discriminators

Figure 14: CelebA samples for AVG trained with 8, 16, and 24 discriminators.

(a) GMAN - 8 discriminators (b) GMAN - 16 discriminators (c) GMAN - 24 discriminators

Figure 15: CelebA samples for GMAN trained with 8, 16, and 24 discriminators.



(a) HV - 8 discriminators (b) HV - 16 discriminators (c) HV - 24 discriminators

Figure 16: CelebA samples for HV trained with 8, 16, and 24 discriminators.

D.2 - Generating 128x128 images

In this experiment, we verify whether the proposed multiple discriminators setting is capable of generating higher resolution images. For that, we employed the CelebA at a size of 128x128. We used a similar architecture for both generator and discriminators networks as described in the previous experiments. A convolutional layer with 2048 feature maps was added to both generator and discriminators architectures due to the increase in the image size. Adam optimizer with the same set of hyperparameters as for CIFAR-10 and CelebA 64x64 was employed. We trained models with 6, 8, and 10 discriminators during 24 epochs. Samples from each generator are shown in Figure 17.

(a) HV - 6 discriminators

(b) HV - 8 discriminators

(c) HV - 10 discriminators

Figure 17: 128x128 CelebA samples for HV trained during 24 epochs with 6, 8, and 10 discriminators.

E - Generating 256x256 Cats

We show the proposed multiple-discriminators setting scales to higher resolution even in the small dataset regime, by reproducing the experiments presented in (Jolicoeur-Martineau, 2018). We used the same architecture for the generator. For the discriminator, we removed batch normalization from all layers and used stride equal to 1 at the last convolutional layer, after adding the initial projection step. The Cats dataset [2] was employed, we followed the same pre-processing steps, which, in our case, yielded 1740 training samples with resolution of 256x256. Our model is trained using 24 discriminators and Adam optimizer with the same hyperparameters as for CIFAR-10 and CelebA previously described experiments. In Figure 18 we show generator's samples after 288 training epochs. One epoch corresponds to updating over 27 minibatches of size 64.

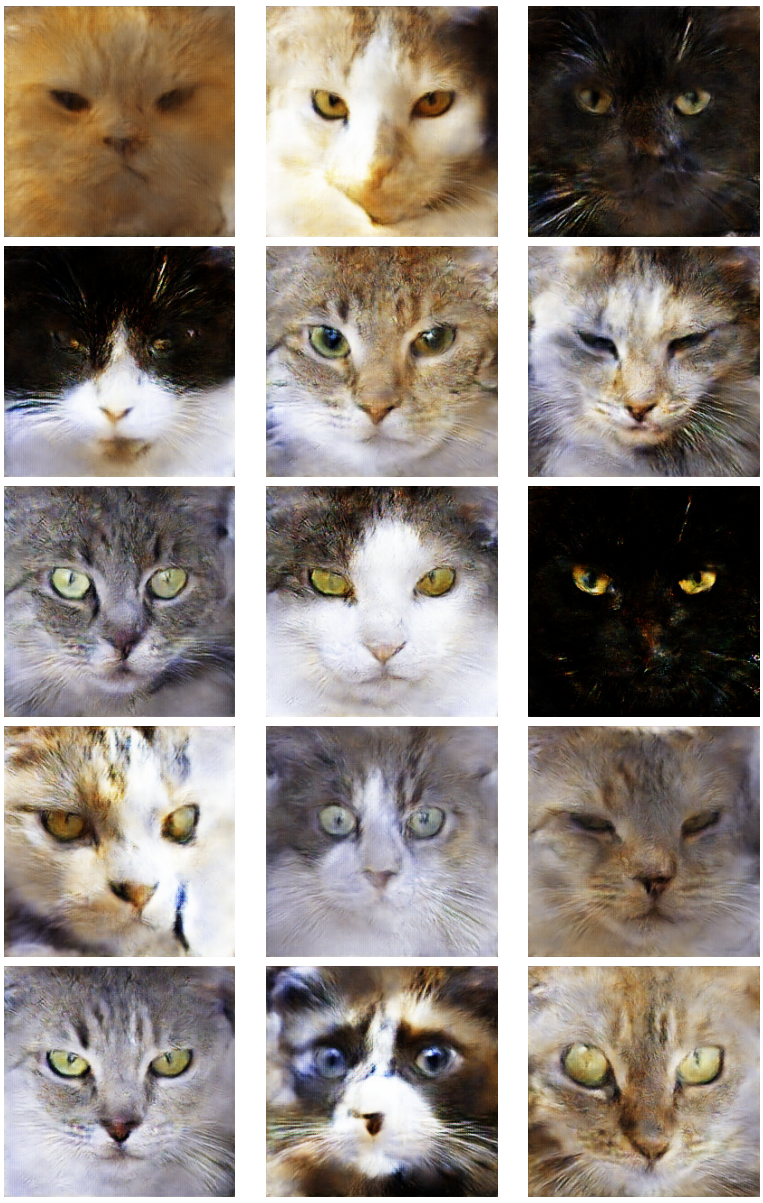
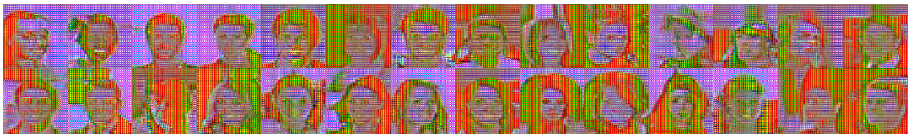


Figure 18: Cats generated using 24 discriminators after 288 training epochs.

[2] https://www.kaggle.com/crawford/cat-dataset

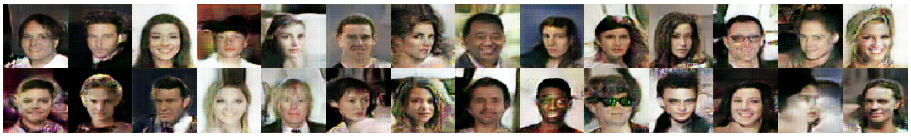
F - Increasing number of random projections

In this experiment we illustrate and confirm the results introduced in (Neyshabur et al., 2017), showing the effect of using an increasing number of random projections to train a GAN. We trained models using average loss minimization with 1 to 6 discriminators on the CelebA dataset for 15 epochs. Samples from the generator obtained in the last epoch are shown in Fig. 19. Generated samples are closer to real data as the number of random projections (and discriminators, consequently) increases.

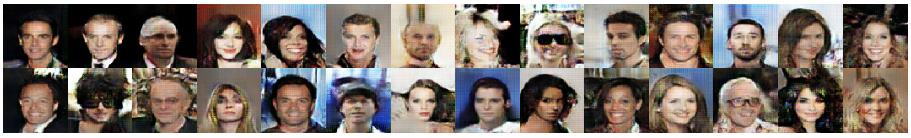


(a) AVG - 1 discriminator

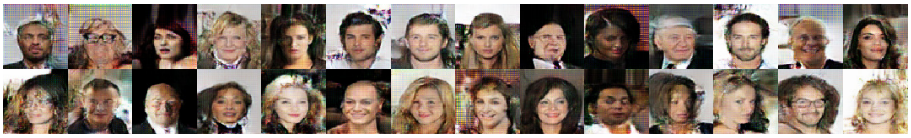
(b) AVG - 2 discriminators



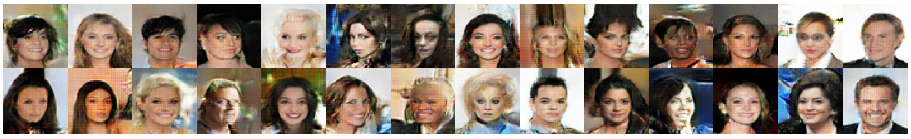
(c) AVG - 3 discriminators



(d) AVG - 4 discriminators



(e) AVG - 5 discriminators



(f) AVG - 6 discriminators

Figure 19: Models trained with AVG during 15 epochs using an increasing number of random projections and discriminators.

H - Wall-clock time for reaching best FID during training on MNIST

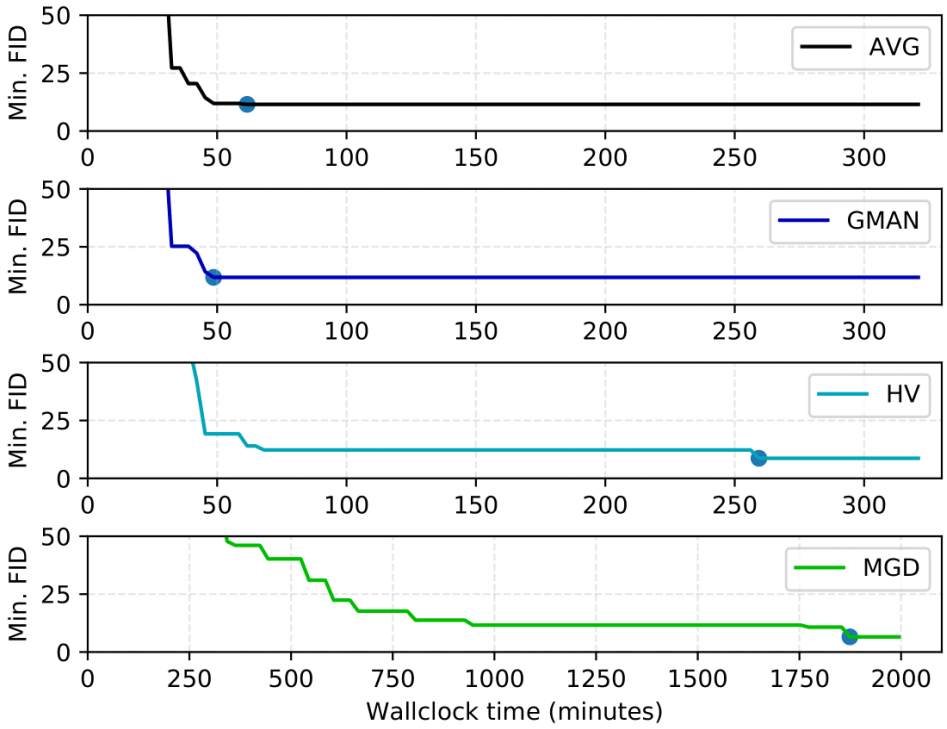


Figure 20: Minimum FID during training. X-axis is in minutes. The blue dot is intended to highlight the moment during training when the minimum FID was reached.