STACKELBERG GAN: TOWARDS PROVABLE MINIMAX EQUILIBRIUM VIA MULTI-GENERATOR ARCHITECTURES

Anonymous authors
Paper under double-blind review

ABSTRACT
We study the problem of alleviating the instability issue in the GAN training procedure via new architecture design. The discrepancy between the minimax and maximin objective values could serve as a proxy for the difficulties that the alternating gradient descent encounters in the optimization of GANs. In this work, we give new results on the benefits of multi-generator architecture of GANs. We show that the minimax gap shrinks to $\epsilon$ as the number of generators increases with rate $\hat{O}(1/\epsilon)$. This improves over the best-known result of $\hat{O}(1/\epsilon^2)$. At the core of our techniques is a novel application of Shapley-Folkman lemma to the generic minimax problem, where in the literature the technique was only known to work when the objective function is restricted to the Lagrangian function of a constraint optimization problem. Our proposed Stackelberg GAN performs well experimentally in both synthetic and real-world datasets, improving Fréchet Inception Distance by 14.61% over the previous multi-generator GANs on the benchmark datasets.

1 INTRODUCTION
Generative Adversarial Nets (GANs) are emerging objects of study in machine learning, computer vision, natural language processing, and many other domains. In machine learning, study of such a framework has led to significant advances in adversarial defenses (Xiao et al., 2018; Samangouei et al., 2018) and machine security (Athalye et al., 2018; Samangouei et al., 2018). In computer vision and natural language processing, GANs have resulted in improved performance over standard generative models for images and texts (Goodfellow et al., 2014), such as variational autoencoder (Kingma & Welling, 2013) and deep Boltzmann machine (Salakhutdinov & Larochelle, 2010). A main technique to achieve this goal is to play a minimax two-player game between generator and discriminator under the design that the generator tries to confuse the discriminator with its generated contents and the discriminator tries to distinguish real images/texts from what the generator creates.

Despite a large amount of variants of GANs, many fundamental questions remain unresolved. One of the long-standing challenges is designing universal, easy-to-implement architectures that alleviate the instability issue of GANs training. Ideally, GANs are supposed to solve the minimax optimization problem (Goodfellow et al., 2014), but in practice alternating gradient descent methods do not clearly privilege minimax over maximin or vice versa (page 35, Goodfellow (2016)), which may lead to instability in training if there exists a large discrepancy between the minimax and maximin objective values. The focus of this work is on improving the stability of such minimax game in the training process of GANs.

To alleviate the issues caused by the large minimax gap, our study is motivated by the so-called Stackelberg competition in the domain of game theory. In the Stackelberg leadership model, the players of this game are one leader and multiple followers, where the leader firm moves first and then the follower firms move sequentially. It is known that the Stackelberg model can be solved to find a subgame perfect Nash equilibrium. We apply this idea of Stackelberg leadership model to the architecture design of GANs. That is, we design an improved GAN architecture with multiple generators (followers) which team up to play against the discriminator (leader). We therefore name our model Stackelberg GAN. Our theoretical and experimental results establish that: GANs with multi-generator architecture have smaller minimax gap, and enjoy more stable training performances.

Our Contributions. This paper tackles the problem of instability during the GAN training procedure with both theoretical and experimental results. We study this problem by new architecture design.
We propose the Stackelberg GAN framework of multiple generators in the GAN architecture. Our framework is general since it can be applied to all variants of GANs, e.g., vanilla GAN, Wasserstein GAN, etc. It is built upon the idea of jointly optimizing an ensemble of GAN losses w.r.t. all pairs of discriminator and generator.

Differences from prior work. Although the idea of having multiple generators in the GAN architecture is not totally new, e.g., MIX+GAN (Arora et al., 2017), MGAN (Hoang et al., 2018), MAD-GAN (Ghosh et al., 2017) and GMAN (Durugkar et al., 2016), there are key differences between Stackelberg GAN and prior work. a) In MGAN (Hoang et al., 2018) and MAD-GAN (Ghosh et al., 2017), various generators are combined as a mixture of probabilistic models with assumption that the generators and discriminator have infinite capacity. Also, they require that the generators share common network parameters. In contrast, in the Stackelberg GAN model we allow various sampling schemes beyond the mixture model, e.g., each generator samples a fixed but unequal number of data points independently. Furthermore, each generator has free parameters. We also make no assumption on the model capacity in our analysis. This is an important research question as raised by Arora et al. (2018). b) In MIX+GAN (Arora et al., 2017), the losses are ensembled with learned weights and an extra regularization term, which discourages the weights being too far away from uniform. We find it slightly unnecessary because the expressive power of each generator already allows implicit scaling of each generator. In the Stackelberg GAN, we apply equal weights for all generators and obtain improved guarantees. c) In GMAN (Durugkar et al., 2016), there are multiple discriminators while it is unclear in theory why multi-discriminator architecture works well. In this paper, we provide formal guarantees for our model.

We prove that the minimax duality gap shrinks as the number of generators increases (see Theorem 1 and Corollary 2). Unlike the previous work, our result has no assumption on the expressive power of generators and discriminator, but instead depends on their non-convexity. With extra condition on the expressive power of generators, we show that Stackelberg GAN is able to achieve \( \epsilon \)-approximate equilibrium with \( \tilde{O}(1/\epsilon) \) generators (see Theorem 3). This improves over the best-known result in (Arora et al., 2017) which requires generators as many as \( \tilde{O}(1/\epsilon^2) \). At the core of our techniques is a novel application of the Shapley-Folkman lemma to the generic minimax problem, where in the literature the technique was only known to work when the objective function is restricted to the Lagrangian function of a constrained optimization problem (Zhang et al., 2018). This results in tighter bounds than that of the covering number argument as in (Arora et al., 2017). We also note that MIX+GAN is a heuristic model which does not exactly match the theoretical analysis in (Arora et al., 2017), while this paper provides formal guarantees for the exact model of Stackelberg GAN.

We empirically study the performance of Stackelberg GAN for various synthetic and real datasets. We observe that without any human assignment, surprisingly, each generator automatically learns balanced number of modes without any mode being dropped (see Figure 1). Compared with other multi-generator GANs with the same network capacity, our experiments show that Stackelberg GAN enjoys 26.76 Fréchet Inception Distance on CIFAR-10 dataset while prior results achieve 31.34 (smaller is better), achieving an improvement of 14.61%.

Figure 1: Left Figure, Top Row: Standard GAN training on a toy 2D mixture of 8 Gaussians. Left Figure, Bottom Row: Stackelberg GAN training with 8 generator ensembles, each of which is denoted by one color. Right Figure: Stackelberg GAN training with 10 generator ensembles on fashion-MNIST dataset without cherry pick, where each row corresponds to one generator.

- We propose the Stackelberg GAN framework of multiple generators in the GAN architecture. Our framework is general since it can be applied to all variants of GANs, e.g., vanilla GAN, Wasserstein GAN, etc. It is built upon the idea of jointly optimizing an ensemble of GAN losses w.r.t. all pairs of discriminator and generator.

Differences from prior work. Although the idea of having multiple generators in the GAN architecture is not totally new, e.g., MIX+GAN (Arora et al., 2017), MGAN (Hoang et al., 2018), MAD-GAN (Ghosh et al., 2017) and GMAN (Durugkar et al., 2016), there are key differences between Stackelberg GAN and prior work. a) In MGAN (Hoang et al., 2018) and MAD-GAN (Ghosh et al., 2017), various generators are combined as a mixture of probabilistic models with assumption that the generators and discriminator have infinite capacity. Also, they require that the generators share common network parameters. In contrast, in the Stackelberg GAN model we allow various sampling schemes beyond the mixture model, e.g., each generator samples a fixed but unequal number of data points independently. Furthermore, each generator has free parameters. We also make no assumption on the model capacity in our analysis. This is an important research question as raised by Arora et al. (2018). b) In MIX+GAN (Arora et al., 2017), the losses are ensembled with learned weights and an extra regularization term, which discourages the weights being too far away from uniform. We find it slightly unnecessary because the expressive power of each generator already allows implicit scaling of each generator. In the Stackelberg GAN, we apply equal weights for all generators and obtain improved guarantees. c) In GMAN (Durugkar et al., 2016), there are multiple discriminators while it is unclear in theory why multi-discriminator architecture works well. In this paper, we provide formal guarantees for our model.

- We prove that the minimax duality gap shrinks as the number of generators increases (see Theorem 1 and Corollary 2). Unlike the previous work, our result has no assumption on the expressive power of generators and discriminator, but instead depends on their non-convexity. With extra condition on the expressive power of generators, we show that Stackelberg GAN is able to achieve \( \epsilon \)-approximate equilibrium with \( \tilde{O}(1/\epsilon) \) generators (see Theorem 3). This improves over the best-known result in (Arora et al., 2017) which requires generators as many as \( \tilde{O}(1/\epsilon^2) \). At the core of our techniques is a novel application of the Shapley-Folkman lemma to the generic minimax problem, where in the literature the technique was only known to work when the objective function is restricted to the Lagrangian function of a constrained optimization problem (Zhang et al., 2018). This results in tighter bounds than that of the covering number argument as in (Arora et al., 2017). We also note that MIX+GAN is a heuristic model which does not exactly match the theoretical analysis in (Arora et al., 2017), while this paper provides formal guarantees for the exact model of Stackelberg GAN.

- We empirically study the performance of Stackelberg GAN for various synthetic and real datasets. We observe that without any human assignment, surprisingly, each generator automatically learns balanced number of modes without any mode being dropped (see Figure 1). Compared with other multi-generator GANs with the same network capacity, our experiments show that Stackelberg GAN enjoys 26.76 Fréchet Inception Distance on CIFAR-10 dataset while prior results achieve 31.34 (smaller is better), achieving an improvement of 14.61%.
2 STACKELBERG GAN

Before proceeding, we define some notations and formalize our model setup in this section.

Notations. We will use bold lower-case letter to represent vector and lower-case letter to represent scalar. Specifically, we denote by $\theta \in \mathbb{R}^t$ the parameter vector of discriminator and $\gamma \in \mathbb{R}^g$ the parameter vector of generator. Let $D_\theta(x)$ be the output probability of discriminator given input $x$, and let $G_\gamma(z)$ represent the generated vector given random input $z$. For any function $f(u)$, we denote by $f^*(v) := \sup_u \{u^T v - f(u)\}$ the conjugate function of $f$. Let $\bar{f}$ be the convex closure of $f$, which is defined as the function whose epigraph is the convex closed hull of that of function $f$. We define $\bar{f} := -\bar{\bar{f}}(-f)$. We will use $I$ to represent the number of generators.

2.1 Model Setup

Preliminaries. The key ingredient in the standard GAN is to play a zero-sum two-player game between a discriminator and a generator — which are often parametrized by deep neural networks in practice — such that the goal of the generator is to map random noise $z$ to some plausible images/texts $G_\gamma(z)$ and the discriminator $D_\theta(\cdot)$ aims at distinguishing the real images/texts from what the generator creates.

For every parameter implementations $\gamma$ and $\theta$ of generator and discriminator, respectively, denote by the payoff value $\phi(\gamma; \theta) := \mathbb{E}_{x \sim P_d} f(D_\theta(x)) + \mathbb{E}_{z \sim P_z} f(1 - D_\theta(G_\gamma(z)))$, where $f(\cdot)$ is some concave, increasing function. Hereby, $P_d$ is the distribution of true images/texts and $P_z$ is a noise distribution such as Gaussian or uniform distribution. The standard GAN thus solves the following saddle point problems:

$$\inf_{\gamma \in \mathbb{R}^g} \sup_{\theta \in \mathbb{R}^t} \phi(\gamma; \theta), \quad \text{or} \quad \sup_{\theta \in \mathbb{R}^t} \inf_{\gamma \in \mathbb{R}^g} \phi(\gamma; \theta). \tag{1}$$

For different choices of function $f$, problem (1) leads to various variants of GAN. For example, when $f(t) = \log t$, problem (1) is the classic GAN; when $f(t) = t$, it reduces to the Wasserstein GAN. We refer interested readers to the paper of Nowozin et al. (2016) for more variants of GANs.

Stackelberg GAN. Our model of Stackelberg GAN is inspired from the Stackelberg competition in the domain of game theory. Instead of playing a two-player game as in the standard GAN, in Stackelberg GAN there are $I + 1$ players with two firms — one discriminator and $I$ generators. One can make an analogy between the discriminator (generators) in the Stackelberg GAN and the leader (followers) in the Stackelberg competition.

Stackelberg GAN is a general framework which can be built on top of all variants of standard GANs. The objective function is simply an ensemble of losses w.r.t. all possible pairs of generators and discriminator: $\Phi(\gamma_1, ..., \gamma_I; \theta) := \sum_{i=1}^I \phi(\gamma_i; \theta)$. Thus it is very easy to implement. The Stackelberg GAN therefore solves the following saddle point problems:

$$w^* := \inf_{\gamma_1, ..., \gamma_I \in \mathbb{R}^g} \sup_{\theta \in \mathbb{R}^t} \frac{1}{I} \Phi(\gamma_1, ..., \gamma_I; \theta), \quad \text{or} \quad q^* := \sup_{\theta \in \mathbb{R}^t} \inf_{\gamma_1, ..., \gamma_I \in \mathbb{R}^g} \frac{1}{I} \Phi(\gamma_1, ..., \gamma_I; \theta).$$

Figure 2: Architecture of Stackelberg GAN. We ensemble the losses of various generator and discriminator pairs with equal weights.
We term \( w^* - q^* \) the minimax (duality) gap. We note that there are key differences between the naïve ensembling model and ours. In the naïve ensembling model, one trains multiple GAN models independently and averages their outputs. In contrast, our Stackelberg GAN shares a unique discriminator for various generators, thus requires jointly training. Figure 2 shows the architecture of our Stackelberg GAN.

**How to generate samples from Stackelberg GAN?** In the Stackelberg GAN, we expect that each generator learns only a few modes. In order to generate a sample that may come from all modes, we use a mixed model. In particular, we generate a uniformly random value \( i \) from 1 to \( I \) and use the \( i \)-th generator to obtain a new sample. Note that this procedure in independent of the training procedure.

3 Analysis of Stackelberg GAN

In this section, we develop our theoretical contributions and compare our results with the prior work.

3.1 Minimax Duality Gap

We begin with studying the minimax gap of Stackelberg GAN. Our main results show that the minimax gap shrinks as the number of generators increases.

To proceed, denote by \( h_i(u_i) := \inf_{\gamma_i \in \mathbb{R}^q} (-\phi(\gamma_i; \cdot)) (u_i) \), where the conjugate operation is w.r.t. the second argument of \( \phi(\gamma_i; \cdot) \). We clarify here that the subscript \( i \) in \( h_i \) indicates that the function \( h_i \) is derived from the \( i \)-th generator. The argument of \( h_i \) should depend on \( i \), so we denote it by \( u_i \). Intuitively, \( h_i \) serves as an approximate convexification of \( -\phi(\gamma_i; \cdot) \) w.r.t. the second argument due to the conjugate operation. Denote by \( \hat{\partial} h_i \) the convex closure of \( h_i \):

\[
\hat{\partial} h_i(u) := \inf_{\{a^i \}, \{u^i \}} \left\{ \sum_{j=1}^{t+2} a^j h_i(u^j) : \bar{u} = \sum_{j=1}^{t+2} a^j u^j, \sum_{j=1}^{t+2} a^j = 1, a^j \geq 0 \right\}.
\]

\( \hat{\partial} h_i \) represents the convex relaxation of \( h_i \) because the epigraph of \( \hat{\partial} h_i \) is exactly the convex hull of epigraph of \( h_i \) by the definition of \( \hat{\partial} h_i \). Let \( \Delta^\text{minimax}_\theta := \inf_{\gamma_1, ..., \gamma_I \in \mathbb{R}^q} \sup_{\theta \in \mathbb{R}^q} \frac{1}{I} \Phi(\gamma_1, ..., \gamma_I; \cdot) - \inf_{\gamma_1, ..., \gamma_I \in \mathbb{R}^q} \frac{1}{I} \Phi(\gamma_1, ..., \gamma_I; \cdot) \) and \( \Delta^\text{maximin}_\theta := \sup_{\theta \in \mathbb{R}^q} \inf_{\gamma_1, ..., \gamma_I \in \mathbb{R}^q} \frac{1}{I} \Phi(\gamma_1, ..., \gamma_I; \cdot) - \inf_{\gamma_1, ..., \gamma_I \in \mathbb{R}^q} \frac{1}{I} \Phi(\gamma_1, ..., \gamma_I; \cdot) \), where \( \Phi(\gamma_1, ..., \gamma_I; \cdot) := \sum_{i=1}^I \Phi(\gamma_i; \cdot) \) and \( -\hat{\partial} \Phi(\gamma_i; \cdot) \) is the convex closure of \( -\phi(\gamma_i; \cdot) \) w.r.t. argument \( \theta \). Therefore, \( \Delta^\text{maximin}_\theta + \Delta^\text{minimax}_\theta \) measures the non-convexity of objective function w.r.t. argument \( \theta \). For example, it is equal to 0 if and only if \( \phi(\gamma_i; \cdot) \) is concave and closed w.r.t. discriminator parameter \( \theta \).

We have the following guarantees on the minimax gap of Stackelberg GAN.

**Theorem 1.** Let \( \Delta_i^\text{minimax} := \sup_{u \in \mathbb{R}^q} \{ h_i(u) - \hat{\partial} h_i(u) \} \geq 0 \) and \( \Delta_i^\text{worst} := \max_{i \in [I]} \Delta_i^\text{minimax} \). Denote by \( t \) the number of parameters of discriminator, i.e., \( \theta \in \mathbb{R}^t \). Suppose that \( h_1(\cdot) \) is continuous and dom\( h_i \) is compact and convex. Then the duality gap can be bounded by

\[
0 \leq w^* - q^* \leq \Delta^\text{minimax}_\theta + \Delta^\text{maximin}_\theta + \epsilon,
\]

provided that the number of generators \( I > \frac{t+1}{\epsilon} \Delta^\text{worst}_\theta \).

**Remark 1.** Theorem 1 makes mild assumption on the continuity of loss and no assumption on the model capacity of discriminator and generators. The analysis instead depends on their non-convexity as being parametrized by deep neural networks. In particular, \( \Delta_i^\text{minimax} \) measures the divergence between the function value of \( h_i \) and its convex relaxation \( \hat{\partial} h_i \); When \( \phi(\gamma_i; \cdot) \) is convex w.r.t. argument \( \gamma_i \), \( \Delta_i^\text{minimax} \) is exactly 0. The constant \( \Delta_i^\text{worst} \) is the maximal divergence among all generators, which does not grow with the increase of \( I \). This is because \( \Delta_i^\text{worst} \) measures the divergence of only one generator and when each generator for example has the same architecture, we have \( \Delta_i^\text{worst} = \Delta_i^\text{minimax} = ... = \Delta_i^\text{maximin} \). Similarly, the terms \( \Delta^\text{minimax}_\theta \) and \( \Delta^\text{maximin}_\theta \) characterize the non-convexity of discriminator. When the discriminator is concave such as logistic regression and support vector machine, \( \Delta^\text{minimax}_\theta = \Delta^\text{maximin}_\theta = 0 \) and we have the following straightforward corollary about the minimax duality gap of Stackelberg GAN.

**Corollary 2.** Under the settings of Theorem 1, when \( \phi(\gamma_i; \cdot) \) is concave and closed w.r.t. discriminator parameter \( \theta \) and the number of generators \( I > \frac{t+1}{\epsilon} \Delta^\text{worst}_\theta \), we have \( 0 \leq w^* - q^* \leq \epsilon \).
Theorem 3. Under the settings of Theorem 1, suppose that for any \( \xi > 0 \), there exists a generator \( G \) such that \( \mathbb{E}_{z \sim P_z} \| G(z) - \mathbf{x} \|_2 \leq \xi \). Let the discriminator and the generators be \( L \)-Lipschitz w.r.t. inputs and parameters, respectively. Then for any \( \epsilon > 0 \), there exist \( I = \frac{4\xi+1}{\epsilon} \Delta_{\text{worst}}^\gamma \) generators \( G_{\gamma_1}^*, ..., G_{\gamma_I}^* \) and a discriminator \( D_0^* \) such that for some value \( V \in \mathbb{R} \),

\[
\forall \gamma_1, ..., \gamma_I \in \mathbb{R}^{|I|}, \quad \Phi(\gamma_1, ..., \gamma_I; \theta^*) \leq V + \epsilon,
\]

\[
\forall \theta \in \mathbb{R}^{|I|}, \quad \Phi(\gamma_1^*, ..., \gamma_I^*; \theta) \geq V - \epsilon.
\] (2)

Related Work. While many efforts have been devoted to empirically investigating the performance of multi-generator GAN, little is known about how many generators are needed so as to achieve certain equilibrium guarantees. Probably the most relevant prior work to Theorem 3 is that of (Arora et al., 2017). In particular, Arora et al. (2017) showed that there exist \( I = \frac{1000\Delta}{\epsilon} \Delta^2 \) generators and one discriminator such that \( \epsilon \)-approximate equilibrium can be achieved, provided that for all \( x \) and any \( \xi > 0 \), there exists a generator \( G \) such that \( \mathbb{E}_{z \sim P_z} \| G(z) - \mathbf{x} \|_2 \leq \xi \). Hereby, \( \Delta \) is a global upper bound of function \( |f| \), i.e., \( f \in [-\Delta, \Delta] \). In comparison, Theorem 3 improves over this result in two aspects: a) the assumption on the expressive power of generators in (Arora et al., 2017) implies our condition \( \mathbb{E}_{z \sim P_z} \| G(z) - \mathbf{x} \|_2 \leq \xi \). Thus our assumption is weaker. b) The required number of generators in Theorem 3 is as small as \( \frac{4\xi+1}{\epsilon} \Delta_{\text{worst}}^\gamma \). We note that \( \Delta_{\text{worst}}^\gamma \ll 2\Delta \) by the definition of \( \Delta_{\text{worst}}^\gamma \). Therefore, Theorem 3 requires much fewer generators than that of (Arora et al., 2017).

4 ARCHITECTURE, CAPACITY AND MODE COLLAPSE/DROPPING

In this section, we empirically investigate the effect of network architecture and capacity on the mode collapse/dropping issues for various multi-generator architecture designs. Hereby, the mode collapsing refers to the phenomenon that generative models simply ignore some hard-to-represent modes of real distributions, and the mode dropping means that some modes of real distributions are “averaged” by generative models. For GAN, it is widely believed that the two issues are caused by the large gap between the minimax and maximin objective function values (see page 35, Goodfellow (2016)).

Our experiments verify that network capacity (change of width and depth) is not very crucial for resolving the mode collapse issue, though it can alleviate the mode dropping in certain senses. Instead, the choice of architecture of generators plays a key role. To visualize this discovery, we test the performance of varying architectures of GANs on a synthetic mixture of Gaussians dataset with 8 modes and 0.01 standard deviation. We observe the following phenomena:

Naively increasing capacity of one-generator architecture does not alleviate mode collapse. It shows that the multi-generator architecture in the Stackelberg GAN effectively alleviates the mode collapse issue. Though naively increasing capacity of one-generator architecture alleviates mode dropping issue, for more challenging mode collapse issue, the effect is not obvious (see Figure 3).

![Figure 3: Comparison of mode collapse/dropping issue of one-generator and multi-generator architectures with varying model capacities. (a) and (b) show that increasing the model capacity can alleviate the mode dropping issue, though it does not alleviate the mode collapse issue. (c) Multi-generator architecture with even small capacity resolves the mode collapse issue.](image)

Stackelberg GAN outperforms multi-branch models. We compare performance of multi-branch GAN and Stackelberg GAN with objective functions:

(Multi-Branch GAN) \( \phi \left( \frac{1}{T} \sum_{i=1}^{T} \gamma_i; \theta \right) \) vs. (Stackelberg GAN) \( \frac{1}{T} \sum_{i=1}^{T} \phi(\gamma_i; \theta) \).
Hereby, the multi-branch GAN has made use of extra information that the real distribution is Gaussian mixture model with probability distribution function \[ \frac{1}{\text{const}} \sum_{i=1}^T p_{\mathcal{X}_i}(x), \] so that each \( \gamma_i \) tries to fit one component. However, even this we observe that with same model capacity, Stackelberg GAN significantly outperforms multi-branch GAN (see Figure 4 (a)(c)) even without access to the extra information. The performance of Stackelberg GAN is also better than multi-branch GAN of much larger capacity (see Figure 4 (b)(c)).

Figure 4: Comparison of mode collapse issue of multi-branch and multi-generator architectures with varying model capacities. (a) and (b) show that increasing the model capacity can alleviate the mode dropping issue, though it does not alleviate the mode collapse issue. (c) Multi-generator architecture with much smaller capacity resolves the mode collapse issue.

Generators tend to learn balanced number of modes when they have same capacity. We observe that for varying number of generators, each generator in the Stackelberg GAN tends to learn equal number of modes when the modes are symmetric and every generator has same capacity (see Figure 5).

Figure 5: Stackelberg GAN with varying number of generators of architecture 2-128-256-512-1024-2.

5 EXPERIMENTS
In this section, we verify our theoretical contributions by the experimental validation.

5.1 MNIST Dataset
We first show that Stackelberg GAN generates more diverse images on the MNIST dataset (LeCun et al., 1998) than classic GAN. We follow the standard preprocessing step that each pixel is normalized via subtracting it by 0.5 and dividing it by 0.5. The detailed network setups of discriminator and generators are in Table 4.

Figure 6 shows the diversity of generated digits by Stackelberg GAN with varying number of generators. When there is only one generator, the digits are not very diverse with many repeated “1”s and much fewer “2”s. As the number of generators increases, the generated images tend to be more diverse. In particular, for 10-generator Stackelberg GAN, each generator is associated with one or two digits without any digit being missed.

5.2 Fashion-MNIST Dataset
We also observe better performance by the Stackelberg GAN on the Fashion-MNIST dataset. Fashion-MNIST is a dataset which consists of 60,000 examples. Each example is a 28 × 28 grayscale image associating with a label from 10 classes. We follow the standard preprocessing step that each pixel is normalized via subtracting it by 0.5 and dividing it by 0.5. We specify the detailed network setups of discriminator and generators in Table 4.

Figure 7 shows the diversity of generated fashions by Stackelberg GAN with varying number of generators. When there is only one generator, the generated images are not very diverse without...
Figure 6: Standard GAN vs. Stackelberg GAN on the MNIST dataset without cherry pick. **Left Figure:** Digits generated by the standard GAN. It shows that the standard GAN generates many “1”’s which are not very diverse. **Middle Figure:** Digits generated by the Stackelberg GAN with 5 generators, where every two rows correspond to one generator. **Right Figure:** Digits generated by the Stackelberg GAN with 10 generators, where each row corresponds to one generator.

Figure 7: Generated samples by Stackelberg GAN on CIFAR-10 dataset without cherry pick. **Left Figure:** Examples generated by the standard GAN. It shows that the standard GAN fails to generate bags. **Middle Figure:** Examples generated by the Stackelberg GAN with 5 generators, where every two rows correspond to one generator. **Right Figure:** Examples generated by the Stackelberg GAN with 10 generators, where each row corresponds to one generator.

any bags being found. As the number of generators increases, the generated images tend to be more diverse. In particular, for 10-generator Stackelberg GAN, each generator is associated with one class without any class being missed.

5.3 CIFAR-10 Dataset

We then implement Stackelberg GAN on the CIFAR-10 dataset. CIFAR-10 includes 60,000 32×32 training images, which fall into 10 classes (Krizhevsky & Hinton, 2009). The architecture of generators and discriminator follows the design of DCGAN in (Radford et al., 2015). We train models with 5, 10, and 20 fixed-size generators. The results show that the model with 10 generators performs the best. We also train 10-generator models where each generator has 2, 3 and 4 convolution layers. We find that the generator with 2 convolution layers, which is the most shallow one, performs the best. So we report the results obtained from the model with 10 generators containing 2 convolution layers. Figure 8a shows the samples produced by different generators. The samples are randomly drawn instead of being cherry-picked to demonstrate the quality of images generated by our model.

For quantitative evaluation, we use Inception score and Fréchet Inception Distance (FID) to measure the difference between images generated by models and real images.

**Results of Inception Score.** The Inception score measures the quality of a generated image and is correlated well with human’s judgment (Salimans et al., 2016). We report the Inception score obtained by our Stackelberg GAN and other baseline methods in Table 1. For fair comparison, we only consider the baseline models which are completely unsupervised model and do not need any label information. Instead of directly using the reported Inception scores by original papers, we replicate the experiment of MGAN using the code, architectures and parameters reported by their original papers, and evaluate the scores based on the new experimental results. Table 1 shows that our model achieves a score of 7.62 in CIFAR-10 dataset, which outperforms the state-of-the-art models.
(a) Samples on CIFAR-10. (b) Samples on Tiny ImageNet.

Figure 8: Examples generated by Stackelberg GAN on CIFAR-10 (left) and Tiny ImageNet (right) without cherry pick, where each row corresponds to samples from one generator.

For fairness, we configure our Stackelberg GAN with the same capacity as MGAN, that is, the two models have comparative number of total parameters. When the capacity of our Stackelberg GAN is as small as DCGAN, our model improves over DCGAN significantly.

**Results of Fréchet Inception Distance.** We then evaluate the performance of models on CIFAR-10 dataset using the Fréchet Inception Distance (FID), which better captures the similarity between generated images and real ones (Heusel et al., 2017). As Table 1 shows, under the same capacity as DCGAN, our model reduces the FID by 20.74%. Meanwhile, under the same capacity as MGAN, our model reduces the FID by 14.61%. This improvement further indicates that our Stackelberg GAN with multiple light-weight generators help improve the quality of the generated images.

Table 1: Quantitative evaluation of various GANs on CIFAR-10 dataset. All results are either reported by the authors themselves or run by us with codes provided by the authors. Every model is trained without label. Methods with higher inception score and lower Fréchet Inception Distance are better.

<table>
<thead>
<tr>
<th>Model</th>
<th>Inception Score</th>
<th>Fréchet Inception Distance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Real data</td>
<td>11.24 ± 0.16</td>
<td>-</td>
</tr>
<tr>
<td>WGAN (Arjovsky et al., 2017)</td>
<td>3.82 ± 0.06</td>
<td>-</td>
</tr>
<tr>
<td>MIX+WGAN (Arora et al., 2017)</td>
<td>4.04 ± 0.07</td>
<td>-</td>
</tr>
<tr>
<td>Improved-GAN (Salimans et al., 2016)</td>
<td>4.36 ± 0.04</td>
<td>-</td>
</tr>
<tr>
<td>ALI (Dumoulin et al., 2017)</td>
<td>5.34 ± 0.05</td>
<td>-</td>
</tr>
<tr>
<td>BEGAN (Berthelot et al., 2017)</td>
<td>5.62</td>
<td>-</td>
</tr>
<tr>
<td>MAGAN (Wang et al., 2017)</td>
<td>5.67</td>
<td>-</td>
</tr>
<tr>
<td>GMAN (Durugkar et al., 2016)</td>
<td>6.00 ± 0.19</td>
<td>-</td>
</tr>
<tr>
<td>DCGAN (Radford et al., 2015)</td>
<td>6.40 ± 0.05</td>
<td>37.7</td>
</tr>
<tr>
<td><strong>Ours (capacity as DCGAN)</strong></td>
<td><strong>7.02 ± 0.07</strong></td>
<td><strong>29.88</strong></td>
</tr>
<tr>
<td>D2GAN (Nguyen et al., 2017)</td>
<td>7.15 ± 0.07</td>
<td>-</td>
</tr>
<tr>
<td>MAD-GAN (our run) (Ghosh et al., 2017)</td>
<td>6.67 ± 0.07</td>
<td>34.10</td>
</tr>
<tr>
<td>MGAN (our run) (Hoang et al., 2018)</td>
<td>7.52 ± 0.1</td>
<td>31.34</td>
</tr>
<tr>
<td><strong>Ours (capacity 1×MGAN≈ 1.8×DCGAN)</strong></td>
<td><strong>7.62 ± 0.07</strong></td>
<td><strong>26.76</strong></td>
</tr>
</tbody>
</table>

5.4 **Tiny ImageNet Dataset**

We also evaluate the performance of Stackelberg GAN on the Tiny ImageNet dataset. The Tiny ImageNet is a large image dataset, where each image is labelled to indicate the class of the object inside the image. We resize the figures down to 32 × 32 following the procedure described in (Chrabaszcz et al., 2017). Figure 8b shows the randomly picked samples generated by 10-generator Stackelberg GAN. Each row has samples generated from one generator. Since the types of some images in the Tiny ImageNet are also included in the CIFAR-10, we order the rows in the similar way as Figure 8a.

6 **Conclusions**

In this work, we tackle the problem of instability during GAN training procedure, which is caused by the huge gap between minimax and maximin objective values. The core of our techniques is a multi-generator architecture. We show that the minimax gap shrinks to $\epsilon$ as the number of generators increases with rate $\tilde{O}(1/\epsilon)$, when the maximization problem w.r.t. the discriminator is concave. This improves over the best-known results of $\tilde{O}(1/\epsilon^2)$. Experiments verify the effectiveness of our proposed methods.
REFERENCES


Quan Hoang, Tu Dinh Nguyen, Trung Le, and Dinh Phung. MGAN: Training generative adversarial nets with multiple generators. 2018.


A SUPPLEMENTARY EXPERIMENTS

Figure 9: Effects of generator architecture of Stackelberg GAN on a toy 2D mixture of Gaussians, where the number of generators is set to be 8. **Top Row:** The generators have one hidden layer. **Middle Row:** The generators have two hidden layers. **Bottom Row:** The generators have three hidden layers. It shows that with the number of hidden layers increasing, each generator tends to learn more modes. However, mode collapse never happens for all three architectures.

Figure 9 shows how the architecture of generators affects the distributions of samples by each generators. The enlarged versions of samples generated by Stackelberg GAN with architectures shown in Table 5 and Table 6 are deferred to Figures 10, 11, 12 and 13.

B PROOFS OF MAIN RESULTS

B.1 PROOFS OF THEOREM 1 AND COROLLARY 2: MINIMAX DUALITY GAP

The statement $0 \leq w^* - q^*$ is by the weak duality. Thus it suffices to prove the other side of the inequality. All notations in this section are defined in Section 3.1.

We first show that

$$\inf_{\gamma_1, \ldots, \gamma_I \in \mathbb{R}^g} \sup_{\theta \in \mathbb{R}^t} \frac{1}{I} \tilde{\Phi}(\gamma_1, \ldots, \gamma_I; \theta) - \sup_{\theta \in \mathbb{R}^t} \inf_{\gamma_1, \ldots, \gamma_I \in \mathbb{R}^g} \frac{1}{I} \tilde{\Phi}(\gamma_1, \ldots, \gamma_I; \theta) \leq \epsilon.$$

Denote by

$$p(u) := \inf_{\gamma_1, \ldots, \gamma_I \in \mathbb{R}^g} \sup_{\theta \in \mathbb{R}^t} \left\{ \tilde{\Phi}(\gamma_1, \ldots, \gamma_I; \theta) - u^T \theta \right\}.$$

We have the following lemma.

**Lemma 4.** We have

$$\sup_{\theta \in \mathbb{R}^t} \inf_{\gamma_1, \ldots, \gamma_I \in \mathbb{R}^g} \tilde{\Phi}(\gamma_1, \ldots, \gamma_I; \theta) = (\mathcal{cl}p)(\mathbf{0}) \leq p(\mathbf{0}) = \inf_{\gamma_1, \ldots, \gamma_I \in \mathbb{R}^g} \sup_{\theta \in \mathbb{R}^t} \tilde{\Phi}(\gamma_1, \ldots, \gamma_I; \theta).$$

**Proof.** By the definition of $p(\mathbf{0})$, we have $p(\mathbf{0}) = \inf_{\gamma_1, \ldots, \gamma_I \in \mathbb{R}^g} \sup_{\theta \in \mathbb{R}^t} \tilde{\Phi}(\gamma_1, \ldots, \gamma_I; \theta)$. Since $(\mathcal{cl}p)(\cdot)$ is the convex closure of function $p(\cdot)$ (a.k.a. weak duality theorem), we have $(\mathcal{cl}p)(\mathbf{0}) \leq p(\mathbf{0})$. We now show that

$$\sup_{\theta \in \mathbb{R}^t} \inf_{\gamma_1, \ldots, \gamma_I \in \mathbb{R}^g} \tilde{\Phi}(\gamma_1, \ldots, \gamma_I; \theta) = (\mathcal{cl}p)(\mathbf{0}).$$
Note that \( p(u) = \inf_{\gamma_1, \ldots, \gamma_l \in \mathbb{R}^r} p_{\gamma_1, \ldots, \gamma_l}(u) \), where \( p_{\gamma_1, \ldots, \gamma_l}(u) = \sup_{\theta \in \mathbb{R}^t} \{ \tilde{\Phi}(\gamma_1, \ldots, \gamma_l; \theta) - u^T \theta \} = (-\tilde{\Phi}(\gamma_1, \ldots, \gamma_l; :))^*(-u) \), and that
\[
\inf_{u \in \mathbb{R}^t} \{ p_{\gamma_1, \ldots, \gamma_l}(u) + u^T \mu \} = -\sup_{u \in \mathbb{R}^t} \{ u^T (-\mu) - p_{\gamma_1, \ldots, \gamma_l}(u) \}
\]
\[
= -(-\tilde{\Phi}(\gamma_1, \ldots, \gamma_l; :))^*(\mu) = \tilde{\Phi}(\gamma_1, \ldots, \gamma_l; \mu). \quad \text{(by conjugate theorem)}
\]
So we have
\[
(\tilde{c}p)(0) = \sup_{\mu \in \mathbb{R}^t} \inf_{u \in \mathbb{R}^t} \{ p(u) + u^T \mu \} \quad \text{(by Lemma 8)}
\]
\[
= \sup_{\mu \in \mathbb{R}^t} \inf_{\gamma_1, \ldots, \gamma_l \in \mathbb{R}^r} \inf_{u \in \mathbb{R}^t} \{ p_{\gamma_1, \ldots, \gamma_l}(u) + u^T \mu \} \quad \text{(by the definition of } p(u))
\]
\[
= \sup_{\mu \in \mathbb{R}^t} \inf_{\gamma_1, \ldots, \gamma_l \in \mathbb{R}^r} \inf_{u \in \mathbb{R}^t} \{ p_{\gamma_1, \ldots, \gamma_l}(u) + u^T \mu \}
\]
\[
\geq \sup_{\mu \in \mathbb{R}^t} \inf_{\gamma_1, \ldots, \gamma_l \in \mathbb{R}^r} \tilde{\Phi}(\gamma_1, \ldots, \gamma_l; \mu). \quad \text{(by Eqn. (3))}
\]

By Lemma 4, it suffices to show \( p(0) - (\tilde{c}p)(0) \leq (t + 1)\Delta_{\gamma}^{\text{worst}} \). We have the following lemma.

**Lemma 5.** *Under the assumption in Theorem 1, \( p(0) - (\tilde{c}p)(0) \leq (t + 1)\Delta_{\gamma}^{\text{worst}} \).*

**Proof.** We note that
\[
p(u) := \inf_{\gamma_1, \ldots, \gamma_l \in \mathbb{R}^r} \sup_{\theta \in \mathbb{R}^t} \left\{ \tilde{\Phi}(\gamma_1, \ldots, \gamma_l; \theta) - u^T \theta \right\}
\]
\[
= \inf_{\gamma_1, \ldots, \gamma_l \in \mathbb{R}^r} \sup_{\theta \in \mathbb{R}^t} \left\{ \sum_{i=1}^l \tilde{c}\phi(\gamma_i; \theta) - u^T \theta \right\} \quad \text{(by the definition of } \tilde{\Phi})
\]
\[
= \inf_{\gamma_1, \ldots, \gamma_l \in \mathbb{R}^r} \left\{ \sum_{i=1}^l \tilde{c}\phi(\gamma_i; :)^*(-u) \right\} \quad \text{(by the definition of conjugate function)}
\]
\[
= \inf_{\gamma_1, \ldots, \gamma_l \in \mathbb{R}^r} \inf_{u_1, \ldots, u_l = -u} \left\{ -\tilde{c}\phi(\gamma_1; :)^*(u_1) \right\} \quad \text{(by Lemma 7)}
\]
\[
= \inf_{u_1, \ldots, u_l = -u} \left\{ (\tilde{\phi}(\gamma_1; :))^*(u_1) + \ldots + (\tilde{\phi}(\gamma_l; :))^*(u_l) \right\}
\]
\[
\geq \inf_{u_1, \ldots, u_l = -u} \{ h_1(u_1) + \ldots + h_l(u_l) \}, \quad \text{(by the definition of } h_i(\cdot))
\]
where \( u_1, \ldots, u_l, u \in \mathbb{R}^t \). Therefore,
\[
p(0) = \inf_{u_1, \ldots, u_l \in \mathbb{R}^t} \sum_{i=1}^l h_i(u_i), \quad \text{s.t. } \sum_{i=1}^l u_i = 0.
\]
Consider the subset of \( \mathbb{R}^{t+1} \):
\[
\mathcal{Y}_i := \{ y_i \in \mathbb{R}^{t+1} : y_i = [u_i, h_i(u_i)], u_i \in \text{dom} h_i \}, \quad i \in [I].
\]
Define the vector summation
\[
\mathcal{Y} := \mathcal{Y}_1 + \mathcal{Y}_2 + \ldots + \mathcal{Y}_I.
\]
Since \( h_i(\cdot) \) is continuous and \( \text{dom} h_i \) is compact, the set
\[
\{(u_i, h_i(u_i)) : u_i \in \text{dom} h_i \}
\]
is compact. So \( \mathcal{Y}, \text{conv}(\mathcal{Y}), \mathcal{Y}_i, \text{and conv}(\mathcal{Y}_i), i \in [I] \) are all compact sets. According to the definition of \( \mathcal{Y} \) and the standard duality argument (Bertsekas, 2009), we have

\[
p(0) = \inf \{ w : \text{there exists } (r, w) \in \mathcal{Y} \text{ such that } r = 0 \},
\]

and

\[
\tilde{c}p(0) = \inf \{ w : \text{there exists } (r, w) \in \text{conv}(\mathcal{Y}) \text{ such that } r = 0 \}.
\]

We are going to apply the following Shapley-Folkman lemma.

**Lemma 6** (Shapley-Folkman, Starr (1969)). Let \( \mathcal{Y}_i, i \in [I] \) be a collection of subsets of \( \mathbb{R}^m \). Then for every \( y \in \text{conv}(\sum_{i=1}^I \mathcal{Y}_i) \), there is a subset \( \mathcal{I}(y) \subseteq [I] \) of size at most \( m \) such that

\[
y \in \left[ \sum_{i \in \mathcal{I}(y)} \mathcal{Y}_i + \sum_{i \notin \mathcal{I}(y)} \text{conv}(\mathcal{Y}_i) \right].
\]

We apply Lemma 6 to prove Lemma 5 with \( m = t + 1 \). Let \( (\mathbf{r}, \mathbf{w}) \in \text{conv}(\mathcal{Y}) \) be such that

\[
\mathbf{r} = 0, \quad \text{and} \quad \mathbf{w} = \tilde{c}p(0).
\]

Applying the above Shapley-Folkman lemma to the set \( \mathcal{Y} = \sum_{i=1}^I \mathcal{Y}_i \), we have that there are a subset \( \mathcal{I} \subseteq [I] \) of size \( t + 1 \) and vectors

\[
(\mathbf{r}_i, \mathbf{w}_i) \in \text{conv}(\mathcal{Y}_i), \quad i \in \mathcal{I} \quad \text{and} \quad \mathbf{u}_i \in \text{dom}h_i, \quad i \notin \mathcal{I},
\]

such that

\[
\sum_{i \notin \mathcal{I}} \mathbf{r}_i + \sum_{i \in \mathcal{I}} \mathbf{r}_i = \mathbf{r} = 0, \quad \text{(4)}
\]

\[
\sum_{i \notin \mathcal{I}} h_i(\mathbf{u}_i) + \sum_{i \in \mathcal{I}} \mathbf{w}_i = \tilde{c}p(0). \quad \text{(5)}
\]

Representing elements of the convex hull of \( \mathcal{Y}_i \subseteq \mathbb{R}^{t+1} \) by Carathéodory theorem, we have that for each \( i \in \mathcal{I} \), there are vectors \( \{ \mathbf{u}_i \}_{j=1}^{t+2} \) and scalars \( \{ \alpha_i^j \}_{j=1}^{t+2} \in \mathbb{R} \) such that

\[
\sum_{j=1}^{t+2} \alpha_i^j = 1, \quad \alpha_i^j \geq 0, \quad j \in [t + 2],
\]

\[
\mathbf{r}_i = \sum_{j=1}^{t+2} \alpha_i^j \mathbf{u}_i^j =: \mathbf{u}_i \in \text{dom}h_i, \quad \mathbf{w}_i = \sum_{j=1}^{t+2} \alpha_i^j h_i(\mathbf{u}_i^j). \quad \text{(6)}
\]

Recall that we define

\[
\tilde{c}h_i(\mathbf{u}) := \inf_{\{ \alpha^j \}, \{ \mathbf{u}^j \}} \left\{ \sum_{j=1}^{t+2} \alpha^j h_i(\mathbf{u}^j) : \mathbf{u} = \sum_{j=1}^{t+2} \alpha^j \mathbf{u}^j, \sum_{j=1}^{t+2} \alpha^j = 1, \alpha^j \geq 0 \right\},
\]

and \( \Delta_i^j := \sup_{\mathbf{u} \in \mathbb{R}^t} \{ h_i(\mathbf{u}) - \tilde{c}h_i(\mathbf{u}) \} \geq 0 \). We have for \( i \in \mathcal{I} \),

\[
\mathbf{w}_i \geq \tilde{c}h_i \left( \sum_{j=1}^{t+2} \alpha_i^j \mathbf{u}_i^j \right) \quad \text{(by the definition of } \tilde{c}h_i(\cdot)\text{)}
\]

\[
\geq h_i \left( \sum_{j=1}^{t+2} \alpha_i^j \mathbf{u}_i^j \right) - \Delta_i^j \quad \text{(by the definition of } \Delta_i^j\text{)}
\]

\[
= h_i(\mathbf{u}_i) - \Delta_i^j. \quad \text{(by Eqn. (6))}
\]

Thus, by Eqns. (4) and (6), we have

\[
\sum_{i=1}^I \mathbf{u}_i = 0, \quad \mathbf{u}_i \in \text{dom}h_i, \quad i \in [I]. \quad \text{(8)}
\]
Therefore, we have

\[ p(0) = \sum_{i=1}^{I} h_i(\tilde{u}_i) \] (by Eqn. (8))

\[ \leq \dot{\partial} p(0) + \sum_{i \in T} I_i \] (by Eqns. (5) and (7))

\[ \leq \dot{\partial} p(0) + |T| \Delta^\text{worst}_y \]

\[ = \dot{\partial} p(0) + (t + 1) \Delta^\text{worst}_y, \] (by Lemma 6)

as desired. \( \square \)

By Lemmas 4 and 5, we have proved that

\[ \inf_{\gamma_1, \ldots, \gamma_I \in \mathbb{R}^I} \sup_{\theta \in \mathbb{R}^I} \frac{1}{I} \Phi(\gamma_1, \ldots, \gamma_I; \theta) - \sup_{\theta \in \mathbb{R}^I} \inf_{\gamma_1, \ldots, \gamma_I \in \mathbb{R}^I} \frac{1}{I} \Phi(\gamma_1, \ldots, \gamma_I; \theta) \leq \epsilon. \]

To prove Theorem 1, we note that

\[ w^* - q^* := \inf_{\gamma_1, \ldots, \gamma_I \in \mathbb{R}^I} \sup_{\theta \in \mathbb{R}^I} \frac{1}{I} \Phi(\gamma_1, \ldots, \gamma_I; \theta) - \sup_{\theta \in \mathbb{R}^I} \inf_{\gamma_1, \ldots, \gamma_I \in \mathbb{R}^I} \frac{1}{I} \Phi(\gamma_1, \ldots, \gamma_I; \theta) \]

\[ = \inf_{\gamma_1, \ldots, \gamma_I \in \mathbb{R}^I} \sup_{\theta \in \mathbb{R}^I} \frac{1}{I} \Phi(\gamma_1, \ldots, \gamma_I; \theta) - \inf_{\gamma_1, \ldots, \gamma_I \in \mathbb{R}^I} \sup_{\theta \in \mathbb{R}^I} \frac{1}{I} \Phi(\gamma_1, \ldots, \gamma_I; \theta) \]

\[ + \inf_{\gamma_1, \ldots, \gamma_I \in \mathbb{R}^I} \sup_{\theta \in \mathbb{R}^I} \frac{1}{I} \Phi(\gamma_1, \ldots, \gamma_I; \theta) - \sup_{\theta \in \mathbb{R}^I} \inf_{\gamma_1, \ldots, \gamma_I \in \mathbb{R}^I} \frac{1}{I} \Phi(\gamma_1, \ldots, \gamma_I; \theta) \]

\[ + \sup_{\theta \in \mathbb{R}^I} \inf_{\gamma_1, \ldots, \gamma_I \in \mathbb{R}^I} \frac{1}{I} \Phi(\gamma_1, \ldots, \gamma_I; \theta) - \sup_{\theta \in \mathbb{R}^I} \inf_{\gamma_1, \ldots, \gamma_I \in \mathbb{R}^I} \frac{1}{I} \Phi(\gamma_1, \ldots, \gamma_I; \theta) \]

\[ \leq \Delta^\text{minimax}_y + \Delta^\text{maximin}_y + \epsilon, \]

as desired.

When \( \phi(\gamma_I; \theta) \) is concave and closed w.r.t. discriminator parameter \( \theta \), we have \( \dot{\partial} \phi = \phi \). Thus, \( \Delta^\text{minimax}_y = \Delta^\text{maximin}_y = 0 \) and \( 0 \leq w^* - q^* \leq \epsilon \).

### B.2 Proofs of Theorem 3: Existence of Approximate Equilibrium

We first show that the equilibrium value \( V \) is \( 2f(1/2) \). For the discriminator \( D_\theta \) which only outputs \( 1/2 \), it has payoff \( 2f(1/2) \) for all possible implementations of generators \( G_{\gamma_1}, \ldots, G_{\gamma_I} \). Therefore, we have \( V \geq 2f(1/2) \). We now show that \( V \leq 2f(1/2) \). We note that by assumption, for any \( \zeta > 0 \), there exists a closed neighbour of implementation of generator \( G_{\xi} \) such that \( E_{x \sim p_d} G_{\xi}(z) - x \|_2 \leq \xi \) for all \( G_{\xi} \) in the neighbour. Such a neighbour exists because the generator is Lipschitz w.r.t. its parameters. Let the parameter implementation of such neighbour of \( G_{\xi} \) be \( \Gamma \). The Wasserstein distance between \( G_{\xi} \) and \( p_d \) is \( \xi \). Since the function \( f \) and the discriminator are \( L_f \)-Lipschitz and \( L \)-Lipschitz, respectively, we have

\[ \left| E_{x \sim G_{\xi}} f(1 - D_\theta(z)) - E_{x \sim p_d} f(1 - D_\theta(x)) \right| \leq O(L_f L \xi). \]

Thus, for any fixed \( \gamma \), we have

\[ \sup_{\theta \in \mathbb{R}^I} E_{x \sim p_d} f(D_\theta(x)) + E_{z \sim G_{\xi}} f(1 - D_\theta(z)) \leq O(L_f L \xi) \]

\[ \leq O(L_f L \xi) \]

which implies that \( \sup_{\theta \in \mathbb{R}^I} \Phi(\gamma_1, \ldots, \gamma_I; \theta) = 2f(1/2) \) for all \( \gamma_1, \ldots, \gamma_I \in \Gamma \). So we have \( V = 2f(1/2) \). This means that the discriminator cannot do much better than a random guess.

The above analysis implies that the equilibrium is achieved when \( D_\theta \) only outputs \( 1/2 \). Denote by \( \Theta \) the small closed neighbourhood such that \( \Phi(\gamma_1, \ldots, \gamma_I; \theta) \) is concave w.r.t. \( \theta \in \Theta \) for any fixed \( \gamma_1, \ldots, \gamma_I \in \Gamma \). We thus focus on the loss on \( \Theta \subset \mathbb{R}^I \) and \( \Gamma \subset \mathbb{R}^9 \):

\[ \Phi(\gamma_1, \ldots, \gamma_I; \theta) := \frac{1}{I} \sum_{i=1}^{I} [E_{x \sim p_d} f(D_\theta(x)) + E_{z \sim p_d} f(1 - D_\theta(G_{\gamma_i}(z))]], \quad \theta \in \Theta, \gamma_1, \ldots, \gamma_I \in \Gamma. \]
Since $\Phi(\gamma_1, \ldots, \gamma_I; \theta)$ is concave w.r.t. $\theta \in \Theta$ for all $\gamma_1, \ldots, \gamma_I \in \Gamma$, by Corollary 2, we have

$$\inf_{\gamma_1, \ldots, \gamma_I \in \Gamma} \sup_{\theta \in \Theta} \frac{1}{I} \Phi(\gamma_1, \ldots, \gamma_I; \theta) - \sup_{\theta \in \Theta} \inf_{\gamma_1, \ldots, \gamma_I \in \Gamma} \frac{1}{I} \Phi(\gamma_1, \ldots, \gamma_I; \theta) \leq \epsilon.$$  

The optimal implementations of $\gamma_1, \ldots, \gamma_I$ is achieved by $\arg\min_{\gamma_1, \ldots, \gamma_I \in \Gamma} \sup_{\theta \in \Theta} \frac{1}{I} \Phi(\gamma_1, \ldots, \gamma_I; \theta)$.

### C Useful Lemmas

**Lemma 7.** Given the function

$$(f_1 + \ldots + f_I)(\theta) := f_1(\theta) + \ldots + f_I(\theta),$$

where $f_i : \mathbb{R}^t \to \mathbb{R}$, $i \in [I]$ are closed proper convex functions. Denote by $f_1^* + \ldots + f_I^*$ the infimal convolution

$$(f_1^* \oplus \ldots \oplus f_I^*)(u) := \inf_{u_1 + \ldots + u_I = u} \{(f_1^*(u_1) + \ldots + f_I^*(u_I)), \quad u \in \mathbb{R}^t.$$  

Provided that $f_1 + \ldots + f_I$ is proper, then we have

$$(f_1 + \ldots + f_I)^*(u) = \text{cl}(f_1^* \oplus \ldots \oplus f_I^*)(u), \quad \forall u \in \mathbb{R}^t.$$  

**Proof.** For all $\theta \in \mathbb{R}^t$, we have

$$f_1(\theta) + \ldots + f_I(\theta) = \sup_{u_1, \ldots, u_I} \left\{ \theta^T u_1 - f_1^*(u_1) + \ldots + \theta^T u_I - f_I^*(u_I) \right\}$$

$$= \sup_{u_1, \ldots, u_I} \left\{ \theta^T (u_1 + \ldots + u_I) - f_1^*(u_1) - \ldots - f_I^*(u_I) \right\}$$

$$= \sup_u \sup_{u_1, \ldots, u_I = u} \left\{ \theta^T u - f_1^*(u_1) - \ldots - f_I^*(u_I) \right\}$$

$$= \sup_u \left\{ \theta^T u - \inf_{u_1, \ldots, u_I = u} f_1^*(u_1) - \ldots - f_I^*(u_I) \right\}$$

$$= \left\{ \theta^T u - (f_1^* + \ldots + f_I^*)(\theta) \right\}$$

Therefore,

$$(f_1^* \oplus \ldots \oplus f_I^*)(u) = \text{cl}(f_1^* \oplus \ldots \oplus f_I^*)(u) = (f_1^* + \ldots + f_I^*)^*(u),$$

where the first equality holds because $(f_1^* \oplus \ldots \oplus f_I^*)$ is convex, the second quality is by standard conjugate theorem, and the last equality holds by conjugating the both sides of Eqn. (9). \qed

**Lemma 8 (Proposition 3.4 (b), Bertsekas (2009)).** For any function $p(u)$, denote by $q(\mu) := \inf_{u \in \mathbb{R}^t} \{p(u) + \mu^T u\}$. We have $\sup_{\mu \in \mathbb{R}^t} \{q(\mu) = \text{cl}(q(0))\}$.

### D Distributional approximation properties of Stackelberg GAN

**Theorem 9.** Suppose that $f$ is strictly concave and the discriminator has infinite capacity. Then, the global optimum of Stackelberg GAN is achieved if and only if

$$\frac{1}{I} \sum_{i=1}^{I} P_{G_{\gamma_i}(x)} = P_d.$$  

**Proof.** We define

$$L(P_d, P_{G_{\gamma_i}(x)}) = \sup_{\theta \in \mathbb{R}^t} \mathbb{E}_{x \sim P_d} f(D_{\theta}(x)) + \mathbb{E}_{z \sim P_z} f(1 - D_{\theta}(G_{\gamma}(z))).$$

Clearly, the vanilla GAN optimization can be understood as projecting under $L$:

$$\inf_{\gamma \in \mathbb{R}^t} L(P_d, P_{G_{\gamma}(x)}).$$
In the Stackelberg GAN setting, we are projecting under a different distance $\tilde{L}$ which is defined as

$$\tilde{L} = \sup_{\theta \in \mathbb{R}^t} \mathbb{E}_{x \sim p_d} f(D_\theta(x)) \frac{1}{I} \sum_{i=1}^I \mathbb{E}_{z \sim p_z} f(1 - D_\theta(G_{\gamma_i}(z))) \quad (10)$$

$$= L \left( P_d, \frac{1}{I} \sum_{i=1}^I P_{G_{\gamma_i}}(z) \right) \quad (11)$$

We note that $f$ is strictly concave and the discriminator has capacity large enough implies the followings: $L(P_1, P_2)$, as a function of $P_2$, achieves the global minimum if and only if $P_2 = P_1$. The theorem then follows from this fact and (11).

E  NETWORK SETUP

Table 2: Architecture and hyper-parameters for the mixture of Gaussians dataset.

<table>
<thead>
<tr>
<th>Operation</th>
<th>Input Dim</th>
<th>Output Dim</th>
<th>BN?</th>
<th>Activation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Generator $G(z) : z \sim \mathcal{N}(0, 1)$</td>
<td>2</td>
<td>16</td>
<td>✓</td>
<td>Tanh</td>
</tr>
<tr>
<td>Linear</td>
<td>16</td>
<td>2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Discriminator</td>
<td>Linear</td>
<td>2</td>
<td>512</td>
<td>Leaky ReLU</td>
</tr>
<tr>
<td>Linear</td>
<td>512</td>
<td>256</td>
<td></td>
<td>Leaky ReLU</td>
</tr>
<tr>
<td>Linear</td>
<td>256</td>
<td>1</td>
<td></td>
<td>Sigmoid</td>
</tr>
<tr>
<td>Number of generators</td>
<td>8</td>
<td>64</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Batch size for real data</td>
<td>64</td>
<td>200</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Number of iterations</td>
<td>200</td>
<td>0.2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Slope of Leaky ReLU</td>
<td>0.2</td>
<td>0.0002</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Learning rate</td>
<td>0.0002</td>
<td>Optimizer</td>
<td>Adam</td>
<td></td>
</tr>
</tbody>
</table>

Table 3: Architecture and hyper-parameters for the MNIST datasets.

<table>
<thead>
<tr>
<th>Operation</th>
<th>Input Dim</th>
<th>Output Dim</th>
<th>BN?</th>
<th>Activation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Generator $G(z) : z \sim \mathcal{N}(0, 1)$</td>
<td>2</td>
<td>512</td>
<td>✓</td>
<td>Tanh</td>
</tr>
<tr>
<td>Linear</td>
<td>512</td>
<td>784</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Discriminator</td>
<td>Linear</td>
<td>2</td>
<td>512</td>
<td>Leaky ReLU</td>
</tr>
<tr>
<td>Linear</td>
<td>512</td>
<td>256</td>
<td></td>
<td>Leaky ReLU</td>
</tr>
<tr>
<td>Linear</td>
<td>256</td>
<td>1</td>
<td></td>
<td>Sigmoid</td>
</tr>
<tr>
<td>Number of generators</td>
<td>10</td>
<td>100</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Batch size for real data</td>
<td>100</td>
<td>0.2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Slope of Leaky ReLU</td>
<td>0.2</td>
<td>0.0002</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Learning rate</td>
<td>0.0002</td>
<td>Optimizer</td>
<td>Adam</td>
<td></td>
</tr>
<tr>
<td>Operation</td>
<td>Input Dim</td>
<td>Output Dim</td>
<td>BN?</td>
<td>Activation</td>
</tr>
<tr>
<td>-----------</td>
<td>-----------</td>
<td>------------</td>
<td>-----</td>
<td>------------</td>
</tr>
<tr>
<td>Generator $G(z)$ : $z \sim \mathcal{N}(0, 1)$</td>
<td>2</td>
<td>128</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>Linear</td>
<td>2</td>
<td>128</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>Linear</td>
<td>128</td>
<td>256</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>Linear</td>
<td>256</td>
<td>512</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>Linear</td>
<td>512</td>
<td>1024</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>Linear</td>
<td>1024</td>
<td>784</td>
<td>✓</td>
<td>Tanh</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Discriminator</th>
<th>Input Dim</th>
<th>Output Dim</th>
<th>Activation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Linear</td>
<td>2</td>
<td>512</td>
<td>Leaky ReLU</td>
</tr>
<tr>
<td>Linear</td>
<td>512</td>
<td>256</td>
<td>Leaky ReLU</td>
</tr>
<tr>
<td>Linear</td>
<td>256</td>
<td>1</td>
<td>Sigmoid</td>
</tr>
</tbody>
</table>

| Number of generators | 10 |
| Batch size for real data | 100 |
| Number of iterations | 500 |
| Slope of Leaky ReLU | 0.2 |
| Learning rate | 0.0002 |
| Optimizer | Adam |

<table>
<thead>
<tr>
<th>Operation</th>
<th>Kernel</th>
<th>Strides</th>
<th>Feature maps</th>
<th>BN?</th>
<th>BN center?</th>
<th>Activation</th>
</tr>
</thead>
<tbody>
<tr>
<td>$G(z)$ : $z \sim \text{Uniform}[-1, 1]$</td>
<td>8x8x128</td>
<td>✓</td>
<td>✓</td>
<td>ReLU</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Fully connected</td>
<td>5x5</td>
<td>2x2</td>
<td>64</td>
<td>✓</td>
<td>✓</td>
<td>ReLU</td>
</tr>
<tr>
<td>Transposed convolution</td>
<td>5x5</td>
<td>2x2</td>
<td>3</td>
<td>✓</td>
<td>✓</td>
<td>Tanh</td>
</tr>
<tr>
<td>Fully connected</td>
<td>8x8x256</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Convolution</td>
<td>5x5</td>
<td>2x2</td>
<td>128</td>
<td>✓</td>
<td>✓</td>
<td>Leaky ReLU</td>
</tr>
<tr>
<td>Convolution</td>
<td>5x5</td>
<td>2x2</td>
<td>256</td>
<td>✓</td>
<td>✓</td>
<td>Leaky ReLU</td>
</tr>
<tr>
<td>Convolution</td>
<td>5x5</td>
<td>2x2</td>
<td>512</td>
<td>✓</td>
<td>✓</td>
<td>Leaky ReLU</td>
</tr>
<tr>
<td>Fully connected</td>
<td>1</td>
<td>✓</td>
<td>✓</td>
<td>Sigmoid</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

| Number of generators | 10 |
| Batch size for real data | 64 |
| Batch size for each generator | 64 |
| Number of iterations | 100 |
| Slope of Leaky ReLU | 0.2 |
| Learning rate | 0.0002 |
| Optimizer | Adam($\beta_1 = 0.5, \beta_2 = 0.999$) |
| Weight, bias initialization | $\mathcal{N}(\mu = 0, \sigma = 0.01), 0$ |
Table 6: Architecture and hyper-parameters for the Tiny ImageNet dataset.

<table>
<thead>
<tr>
<th>Operation</th>
<th>Kernel</th>
<th>Strides</th>
<th>Feature maps</th>
<th>BN?</th>
<th>BN center?</th>
<th>Activation</th>
</tr>
</thead>
<tbody>
<tr>
<td>$G(z) : z \sim \text{Uniform}[-1, 1]$</td>
<td></td>
<td></td>
<td>100</td>
<td>X</td>
<td>X</td>
<td>ReLU</td>
</tr>
<tr>
<td>Fully connected</td>
<td></td>
<td></td>
<td>8x8x256</td>
<td>X</td>
<td>X</td>
<td>ReLU</td>
</tr>
<tr>
<td>Transposed convolution</td>
<td>5x5</td>
<td>2x2</td>
<td>128</td>
<td>X</td>
<td>X</td>
<td>ReLU</td>
</tr>
<tr>
<td>Transposed convolution</td>
<td>5x5</td>
<td>2x2</td>
<td>3</td>
<td>X</td>
<td></td>
<td>Tanh</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Operation</th>
<th>Kernel</th>
<th>Strides</th>
<th>Feature maps</th>
<th>BN?</th>
<th>BN center?</th>
<th>Activation</th>
</tr>
</thead>
<tbody>
<tr>
<td>$D(x)$</td>
<td></td>
<td></td>
<td>8x8x256</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Convolution</td>
<td>5x5</td>
<td>2x2</td>
<td>128</td>
<td>✓</td>
<td>✓</td>
<td>Leaky ReLU</td>
</tr>
<tr>
<td>Convolution</td>
<td>5x5</td>
<td>2x2</td>
<td>256</td>
<td>✓</td>
<td>✓</td>
<td>Leaky ReLU</td>
</tr>
<tr>
<td>Convolution</td>
<td>5x5</td>
<td>2x2</td>
<td>512</td>
<td>✓</td>
<td>✓</td>
<td>Leaky ReLU</td>
</tr>
<tr>
<td>Fully connected</td>
<td></td>
<td></td>
<td>1</td>
<td></td>
<td></td>
<td>Sigmoid</td>
</tr>
</tbody>
</table>

- Number of generators: 10
- Batch size for real data: 64
- Batch size for each generator: 64
- Number of iterations: 300
- Slope of Leaky ReLU: 0.2
- Learning rate: 0.00001
- Optimizer: Adam($\beta_1 = 0.5, \beta_2 = 0.999$)
- Weight, bias initialization: $\mathcal{N}(\mu = 0, \sigma = 0.01), 0$

Figure 10: Examples generated by Stackelberg GAN with 10 generators on CIFAR-10 dataset, where each row corresponds to samples from one generator.
Figure 11: Examples generated by Stackelberg GAN with 10 generators on CIFAR-10 dataset, where each row corresponds to samples from one generator.
Figure 12: Examples generated by Stackelberg GAN with 10 generators on Tiny ImageNet dataset, where each row corresponds to samples from one generator.
Figure 13: Examples generated by Stackelberg GAN with 10 generators on Tiny ImageNet dataset, where each row corresponds to samples from one generator.