Bayesian Modelling and Monte Carlo Inference for GAN

Hao He [1] **Hao Wang** [1] **Guang-He Lee** [1] **Yonglong Tian** [1]

Abstract

Bayesian modelling is a principal framework to perform model aggregation, which has been a primary mechanism to combat mode collapsing in the context of Generative Adversarial Networks (GANs). In this paper, we propose a novel Bayesian modelling framework for GANs, which iteratively learns a distribution over generators. We tailor stochastic gradient Hamiltonian Monte Carlo with novel gradient approximation to perform Bayesian inference. Theoretically, we prove any generator distribution which produces the target data distribution is an equilibrium of our algorithm. Empirical evidence on categorical distributed data and synthetic high-dimensional multi-modal data demonstrates the superior performance of our method over the start-of-art multi-generator and other Bayesian treatment for GANs.

1. Introduction

Generative Adversarial Networks (GAN) (Goodfellow et al., 2014) is a popular method to learn a distribution on complex data such as natural images, videos and texts. However, it is notoriously hard to train and suffers from *mode collapse*. There has been a series of works addressing these issues. One noticeable thread focuses on objective design, which improves the original Jensen-Shannon divergence with more stable pseudo-metrics such as f-divergence (Nowozin et al., 2016), χ^2-divergence (Mao et al., 2017), and Wasserstein distance (Arjovsky et al., 2017). However, when a single generator does not include enough model capacity to capture the granularity in data distribution practically, evidently the resulting generator still suffer from inaccurate results no matter which distance metric is employed.

An alternative remedy is to learn multiple generators instead of a single generator. This type of methods (Hoang et al., 2018; Tolstikhin et al., 2017; Wang et al., 2016) is motivated by a straightforward intuition that multiple generators can better model multi-modal distributions since each generator only needs to capture a subset of the modes. To entail model aggregation, Bayesian modelling is a natural and principle framework to articulate the aggregation process in a probabilistic way.

Recently, Saatci and Wilson (Saatci & Wilson, 2017) proposes a probabilistic framework for GAN under Bayesian inference, which is called Bayesian GAN. It shows that modelling the distribution of generator helps alleviate mode collapse and motivates interpretability of the learned generators. The probabilistic model is built upon conditional distribution of generator and discriminator, whose maximum likelihood estimation can be realized as a metaphor of typical GAN objective. Finally, the distribution of generator is obtained by marginalizing the joint distribution defined by the conditionals.

In this paper, we follow the idea of learning a distribution of generators, since it entails a general form of learning GAN with multiple (or infinite) generators. However, we prove that existing Bayesian method (Saatci & Wilson, 2017) may lead to incompatible conditionals, which suggest that the underlying joint distribution actually does not exist. As a remedy, we propose a novel Bayesian modeling framework, which models the distributions of generators and discriminators separately. Our main contributions are:

- We prove the previous Bayesian method (Saatci & Wilson, 2017) for any minimax GAN objective induces incompatibility of its defined conditional distributions.
- We prove that, under our Bayesian framework, any generator distribution faithful to the data distribution is an equilibrium, which does not hold for the previous Bayesian method (Saatci & Wilson, 2017)
- We propose two special Monte Carlo inference algorithms for our Bayesian model which efficiently approximate the gradient of a non-differentiable term.
- Empirical studies on categorical distributed data and synthetic high-dimensional multi-modal data demonstrate the superiority of the proposed framework over the state-of-the-art GAN methods.

[1]Computer Science and Artificial Intelligence Laboratory, Massachusetts Institute of Technology. Correspondence to: Hao He <haohe@mit.edu>.

Presented at the ICML 2018 workshop on Theoretical Foundations and Applications of Deep Generative Models, Stockholm, Sweden, PMLR 80, 2018.

2. Related Work

Generative Adversarial Networks is powerful class of methods to learn a generative model for any complex target data distribution. There is game between a generator and a discriminator. Both of them adapt their strategies to minimize their own loss function involving the other:

$$\min_g \mathcal{L}_g(\theta_g;\theta_d),\ \ \min_d \mathcal{L}_d(\theta_d;\theta_g). \tag{1}$$

Eqn. 1 gives a general mathematical form of the game. The loss functions (termed as *GAN objective* in this paper) are elaborately chosen such that at the equilibrium, the generator can generate a target data distribution. Table 1 summarizes several widely used GAN objectives, including the original min-max version where $\mathcal{L}_g(\theta_g;\theta_d) = -\mathcal{L}_d(\theta_d;\theta_g)$, non-saturating version of original GAN (Goodfellow, 2016), LSGAN (Mao et al., 2017), and WGAN (Arjovsky et al., 2017). For readers unfamiliar with GAN, we refer to Sec. 3.1 for a detailed explanation of the notations.

Training GAN with multiple generators is considered in several recent works to mitigate the mode collapse problem. In the spirit of boosting algorithm, (Wang et al., 2016) propose to incrementally train new generator using a subset of training data that are not well captured by previous generators, while (Tolstikhin et al., 2017) further propose a more robust mechanism to reweight samples in the training set for new generator. From the perspective of game theory, MIX-GAN (Arora et al., 2017) extends the game between a single generator and discriminator to the multiple-player setting. Other works resort to third party classifiers to help multiple generators and discriminators achieve better equilibrium, such as MGAN (Hoang et al., 2018), MAD-GAN (Ghosh et al., 2017).

Bayesian GAN (Saatci & Wilson, 2017) is a different approach which models a joint distribution of generators and discriminators by giving the conditional posteriors in Eqn. 2 with the likelihood terms ($\exp\{-\mathcal{L}_g(\theta_g;\theta_d)\}$ and $\exp\{-\mathcal{L}_d(\theta_d;\theta_g)\}$), and priors ($p(\theta_g|\alpha_g)$ and $p(\theta_d|\alpha_d)$) parameterized by α_d and α_g. As we can see, the likelihood is specifically designed such that maximizing the likelihood of the conditional distribution is equivalently optimizing the corresponding GAN objective. The authors argue that instead of doing point mass maximum likelihood estimation as optimization based GAN does, marginalizing the generator's distribution which is multi-modal itself offers better ability to learn a multi-modal data distribution.

$$\begin{aligned} p(\theta_g|\theta_d) &\propto \exp\{-\mathcal{L}_g(\theta_g;\theta_d)\}p(\theta_g|\alpha_g). \\ p(\theta_d|\theta_g) &\propto \exp\{-\mathcal{L}_d(\theta_d;\theta_g)\}p(\theta_d|\alpha_d). \end{aligned} \tag{2}$$

In order to obtain the generator's marginal distribution, Stochastic Gradient Hamiltonian Monte Carlo (SGHMC) (Chen et al., 2014) is employed. SGHMC is a method to generate samples from a given distribution $p(\mathsf{x})$ as long as it is differentiable.[1] With SGHMC, Bayesian GAN updates the Monte Carlo samples, $\{\theta_{d;m}^{(t)}\}_{m=1}^M$ and $\{\theta_{g;m}^{(t)}\}_{m=1}^M$ as described in Eqn. 3.

$$\begin{aligned} \{\theta_{g;m}^{(t+1)}\}_{m=1}^M \sim p(\theta_g|\{\theta_{d;m}^{(t)}\}_{m=1}^M) &= \Big(\prod_m p(\theta_g|\theta_{d;m}^{(t)})\Big)^{\frac{1}{M}} \\ &= \exp\{-\frac{1}{M}\sum_m \mathcal{L}_g(\theta_g;\theta_{d;m}^{(t)})\}p(\theta_g|\alpha_g). \\ \{\theta_{d;m}^{(t+1)}\}_{m=1}^M \sim p(\theta_d|\{\theta_{g;m}^{(t)}\}_{m=1}^M) &= \Big(\prod_m p(\theta_d|\theta_{g;m}^{(t)})\Big)^{\frac{1}{M}} \\ &= \exp\{-\frac{1}{M}\sum_m \mathcal{L}_d(\theta_d;\theta_{g;m}^{(t)})\}p(\theta_d|\alpha_d). \end{aligned} \tag{3}$$

The underlying distributions $q_g^{(t)}(\theta_g), q_d^{(t)}(\theta_d)$ represented by the Monte Carlo samples are actually updated as Eqn. 4.

$$\begin{aligned} q_g^{(t+1)}(\theta_g) &\propto \exp\{-\mathsf{E}_{\ d\ \ q_d^{(t)}}\mathcal{L}_g(\theta_g;\theta_d)\}p(\theta_g|\alpha_g). \\ &\propto \exp\{\mathsf{E}_{\ d\ \ q_d^{(t)}}\log p(\theta_g|\theta_d)\} \\ q_d^{(t+1)}(\theta_d) &\propto \exp\{-\mathsf{E}_{\ g\ \ q_g^{(t)}}\mathcal{L}_d(\theta_d;\theta_g)\}p(\theta_d|\alpha_d) \\ &\propto \exp\{\mathsf{E}_{\ g\ \ q_g^{(t)}}\log p(\theta_d|\theta_g)\}. \end{aligned} \tag{4}$$

To facilitate discussion, we categorize GAN frameworks into the following taxonomy: *optimization based method* and *probabilistic method*. Optimization based method sets up an explicit mini-max game between the generator and discriminator, where an equilibrium typically characterize a generator faithful to data distribution in an explicit way. In probabilistic method, generators and discriminators evolve as particles of underlying distributions, where an equilibrium is searched from a stochastic exploration in the distribution space (of the generators and discriminators).

3. Methodology

The section is organized as following. We first summarize the notations used in the paper. Second, we elaborate our Bayesian modelling for GAN and develop constituent prior and likelihood formulations. Then we make a detailed comparison with previous Bayesian method, which highlights theoretical difference between the methods. Finally, we develop algorithms to infer the posterior for our Bayesian modelling.

3.1. Notations

In this paper, we aim to learn a data distribution $p_{data}(\mathsf{x})$, with sample x drawn from $\mathcal{X}$. Our generator and discriminator are parameterized by $\theta_g \in \ _g$ and $\theta_d \in \ _d$. A generator with parameter θ_g defines a mapping from a random noise vector $\mathsf{z} \sim p_z$ to a random vector $G(\mathsf{z};\theta_g)$ over the data space $\mathcal{X}$ which induce a generated data distribution $p_{gen}(\mathsf{x};\theta_g)$. A discriminator is a function of data to

[1] We refer reader to the original paper for more details.

Table 1. Common GAN objectives.

	$L_d(\theta_d;\theta_g)$	$L_g(\theta_g;\theta_d)$
GAN (MIN-MAX)	$\mathsf{E}_{x\ p_{data}}[\log D(x;\theta_d)]\quad \mathsf{E}_{x\ p_{gen}(\ ;\ g)}[\log(1\quad D(x;\theta_d))]$	$L_d(\theta_d;\theta_g)$
GAN (NON-SATURATING)	$\mathsf{E}_{x\ p_{data}}[\log D(x;\theta_d)]\quad \mathsf{E}_{x\ p_{gen}(\ ;\ g)}[\log(1\quad D(x;\theta_d))]$	$\mathsf{E}_{x\ p_{gen}(\ ;\ g)}[\log D(x;\theta_d)]$
WASSERSTEIN GAN	$\mathsf{E}_{x\ p_{data}}[D(x;\theta_d)] + \mathsf{E}_{x\ p_{gen}(\ ;\ g)}[D(x;\theta_d)]$	$L_d(\theta_d;\theta_g)$
LEAST-SQUARES GAN	$\mathsf{E}_{x\ p_{data}}[(D(x;\theta_d)\quad 1)^2]\quad \mathsf{E}_{x\ p_{gen}(\ ;\ g)}[D(x;\theta_d)^2]$	$\mathsf{E}_{x\ p_{gen}(\ ;\ g)}[(D(x;\theta_d)\quad 1)^2]$

a probability score $D(x;\theta_d) : \mathcal{X} \to [0,1]$ [2]. Further, we use $q_g(\theta_g) \in \mathcal{P}^{\Theta_g}$, $q_d(\theta_d) \in \mathcal{P}^{\Theta_d}$ to denote the distribution over generators and discriminators respectively.

The data distribution generated by generators under the distribution $q_g(\theta_g)$ is naturally a mixture of data distribution given by every single generator, $p_{model}(x) = \mathsf{E}_{g\ q_g(\ g)}[p_{gen}(x;\theta_g)]$. Our goal is to learn a generator distribution such that the total mixture of generated data distribution matches our target, i.e. $p_{model}(x) \simeq p_{data}(x)$.

$\mathcal{L}_g(\theta_g;\theta_d)$ and $\mathcal{L}_d(\theta_d;\theta_g)$ denote loss functions of generator and discriminator. The common choices[3] are listed in Table 1. With a slight abuse of the notation, we equalize $\mathcal{L}_g(\theta_g; D(\cdot;\theta_d))$ to $\mathcal{L}_g(\theta_g;\theta_d)$ and extend the notation by replacing $D(\cdot;\theta_d)$ with any function D , i.e., $\mathcal{L}_g(\theta_g; D\ (\cdot))$ represents a loss function for the generator given a virtual discriminator D . Similarity $\mathcal{L}_d(\theta_d; p(\cdot))$ represents the loss function for discriminator given a virtual generator that generates data from distribution $p(\cdot)$.

3.2. Bayesian Modelling for GAN

In this section, we first introduce the likelihood function and desired prior, and then elaborate an iterative posterior refinement algorithm with theoretical analysis.

Likelihood. The likelihood function should encode the information that distributionally reflect the loss of generators $\mathcal{L}_g$ and discriminators $\mathcal{L}_d$. An ideal solution is to form a distribution that is proportional to quantities that evaluates such fitness:

$$\begin{aligned} p(\theta_g) &\propto \exp\{-\mathcal{L}_g(\theta_g; D^{(t)})\}. \\ p(\theta_d) &\propto \exp\{-\mathcal{L}_d(\theta_d; p^{(t)}_{model})\}. \end{aligned} \tag{5}$$

where $p^{(t)}_{model}(x) = \mathsf{E}_{g\ q^{(t)}_g}[p_{gen}(x;\theta_g)]$ is the mixed data distribution under current generator distribution $q^{(t)}_g$ and $D^{(t)}(\cdot) = \mathsf{E}_{d\ q^{(t)}_d}[D(\cdot;\theta_d)]$ is the averaged discriminating score function under current discriminator distribution $q^{(t)}_d$.

We emphasize the difference between that our likelihood model and that of Bayesian GAN (Eqn. 4). In stead of using 'expectation of loss values' (e.g. $\mathsf{E}_{g\ q^{(t)}_g}\mathcal{L}_d(\theta_d;\theta_g)$) as Bayesian GAN did, we propose to use 'loss value of expectation' (e.g. $\mathcal{L}_d(\theta_d; \mathsf{E}_{g\ q^{(t)}_g}[p_{gen}(\cdot;\theta_g)])$). Computing the loss value of mixed data distribution makes more sense since our goal is matching to expectation of generated data distribution to the target data distribution.

Prior. While the likelihood function is rather straightforward, an ideal prior is more involved. When the generated data distribution is increasingly close to the real data distribution, there will be decreasing information for discriminator to distinguish between them; the discriminator will tend to assign equal scores to all data samples, resulting in equal likelihoods for all generators. At that stage, it is better to keep the generator distribution the same as the previous time step, since it already generates the desired data distribution perfectly. Hence, we use the generator distribution in the previous time step as a prior for the next time step. Such dynamically evolving prior for generator turns out to be crucial, In Sec. 3.3 we show the Bayesian model in previous work suffer from bad convergence due to its fixed weakly informative prior.

In contrast, we set a uniform improper prior for the discriminator to pursuit unrestricted adaptability to evolving generators.

Posterior. Integrating the prior mentioned above and the likelihood in Eqn. 5, we formulate our iterative Bayesian model as

$$\begin{aligned} q^{(t+1)}_g(\theta_g) &\propto \exp\{-\mathcal{L}_g(\theta_g; D^{(t)})\} \cdot q^{(t)}_g(\theta_g), \\ q^{(t+1)}_d(\theta_d) &\propto \exp\{-\mathcal{L}_d(\theta_d; p^{(t)}_{model})\}. \end{aligned} \tag{6}$$

This iterative process for updating $q^{(t)}_g$ and $q^{(t)}_d$ can be *universally* applied to any common GAN objectives with a guarantee of *effectiveness* (i.e., any desired generator distribution will be a convergence point of the iteration).

Analysis. Theorem 1 shows that our iterative process is theoretically valid. Note that, here our theorem is stated in the setting of the vanilla GAN objective. While its general version is articulated in the supplementary which applies to any other GAN objectives, such as WGAN (Arjovsky et al., 2017) and LSGAN (Mao et al., 2017).

Theorem 1. *Assume the GAN objective in Eqn. 6 is minmax vanilla GAN objective in Table 1. If there exists an* optimal *generator distribution $q_g(\theta_g)$ satisfying p_{model} , $\mathsf{E}_{g\ q_g}[p_{gen}(\cdot;\theta_g)] = p_{data}$, then there exists a discriminator distribution q_d such that $D\ (\cdot)$, $\mathsf{E}_{d\ q_d} D(\cdot;\theta_d) \equiv 0.5$. Moreover q_g and q_d are the fix points of the iteration defined in Eqn. 6.*

[2] Sometimes $X\ !\ R$ according to different GAN objective.

[3] The concepts of minimax version and non-saturating version of vanilla GAN are first introduced in (Goodfellow, 2016).

Proof. First, with Eqn. 7 and Eqn. 8 we show $q_d(\theta_d) \propto \mathcal{L}_d(\theta_d; p_{model})$ satisfies $\mathsf{E}_{\theta_d \sim q_d} D(\cdot;\theta_d) \equiv 0.5$. In the following equations, $\mathcal{S}(\theta_d)$ is the symmetric discriminator of θ_d whose definition is delayed to the next paragraph after the proof.

$$\begin{aligned} & q_d(\theta_d) \\ & \propto \mathsf{E}_{\mathbf{x} \sim p_{data}}[\log D(\mathbf{x};\theta_d)] + \mathsf{E}_{\mathbf{x} \sim p_{model}}[\log(1 - D(\mathbf{x};\theta_d))] \\ & = \mathsf{E}_{\mathbf{x} \sim p_{data}}[\log D(\mathbf{x};\theta_d) + \log(1 - D(\mathbf{x};\theta_d))] \\ & \Rightarrow q_d(\theta_d) = q_d(\mathcal{S}(\theta_d)), \end{aligned} \tag{7}$$

$$\begin{aligned} & \mathsf{E}_{\theta_d \sim q_d} D(\mathbf{x};\theta_d) = \int_{\theta_d \in \Theta_d} q_d(\theta_d) D(\mathbf{x};\theta_d) d\theta_d \\ & = \frac{1}{2}\Big(\int_{\theta_d} q_d(\theta_d) D(\mathbf{x};\theta_d) + \int_{\theta_d^0 = \mathcal{S}(\theta_d)} q_d(\theta_d^0) D(\mathbf{x};\theta_d^0) \Big) \\ & = \frac{1}{2} \int_{\theta_d \in \Theta_d} q_d(\theta_d)(D(\mathbf{x};\theta_d) + D(\mathbf{x};\mathcal{S}(\theta_d))) d\theta_d \\ & = \frac{1}{2} \int_{\theta_d \in \Theta_d} q_d(\theta_d) \cdot 1 \cdot d\theta_d = \frac{1}{2}. \end{aligned} \tag{8}$$

Second, according Eqn. 9, we know $\mathcal{L}_g(\theta_g; D)$ is a constant which leads $q_g(\theta_g) \propto \exp\{-\mathcal{L}_g(\theta_g; D)\} \times q_g(\theta_g)$.

$$\begin{aligned} \mathcal{L}_g(\theta_g; D) &= \mathsf{E}_{\mathbf{x} \sim p_{gen}(\cdot;\theta_g)}[\log D(\mathbf{x})] \\ &= \log(0.5), \forall \mathbf{x} \in \mathcal{X}. \end{aligned} \tag{9}$$

□

Note that, in the proof we implicitly make two very weak assumptions of the discriminator parameter's space. First, for any two different parameters $\theta_d \neq \theta_d^0$, the discriminator functions are not equivalent, i.e. $D(\mathbf{x};\theta_d) \not\equiv D(\mathbf{x};\theta_d^0)$. Second, the space of discriminator are symmetric which means for any $\theta_d \in \Theta_d$, there is a $\theta_d^0 \in \Theta_d$ such that $D(\mathbf{x};\theta_d) \equiv 1 - D(\mathbf{x};\theta_d^0)$. [4] According to the first assumption, θ_d^0 is unique. Thus we further define an operator $\mathcal{S}(\theta_d)$ as the mapping from θ_d to its symmetric counterpart θ_d^0.

3.3. Comparison with Bayesian GAN

In this section, we compare our model with Bayesian GAN (Saatci & Wilson, 2017) (refered as BGAN in the rest of the paper) and explain the motivation of our method.

As showed in Eqn. 6 and Eqn. 4, the first difference is the choice of prior distributions. BGAN manually set weakly informative prior for both the generator and the discriminator. Hence during the iterative process, the generator distribution only relies on the previous discriminator distribution, which might be problematic when the discriminator becomes non-informative as we mentioned in Sec. 3.2.

[4] In the case of $D(\mathbf{x};\theta_d) : \mathcal{X} \to [0,1]$, θ_d and θ_d^0 being symmetric means $D(\mathbf{x};\theta_d) \equiv 1 - D(\mathbf{x};\theta_d^0)$ while in the case of $D(\mathbf{x};\theta_d) : \mathcal{X} \to \mathbb{R}$, symmetry means $D(\mathbf{x};\theta_d) \equiv -D(\mathbf{x};\theta_d^0)$.

The second difference happens in the choice of likelihood. BGAN defines its likelihood as distribution condition on a single parameter which satisfies the property that its maximum likelihood estimation reduces to the traditional GAN objective. However, we argue that since our goal is to learn a generator distribution, it would more preferable to directly build a Bayesian model on the distribution of generators.

Below we provide two theoretical analyses to show the inappropriateness of BGAN in certain settings.

Compatibility Issue

In this part, we show BGAN is not suitable for any min-max-style GAN objective due to the incompatibility of its conditional posterior. This issue may limit the usage of BGAN since many common choices of GAN objective are in min-max fashion, such as the original GAN and WGAN.

By iteratively sampling from the conditional posterior in Eqn. 2, BGAN implicitly samples from a joint posterior distribution of θ_d and θ_g. The corresponding marginal distribution is supposed to give a good generator distribution that can produce the target data distribution. However, our theoretical analysis shows that such a presumed joint distribution does not exist. Specifically, according to Lemma 1, the existence of a joint distribution satisfying Eqn. 2 requires the GAN objective $\mathcal{L}(\theta_d, \theta_g) = \mathcal{L}_d(\theta_d; \theta_g) = -\mathcal{L}_g(\theta_g; \theta_d)$ to be decomposable, i.e. $\exists \phi_g, \phi_d$, s.t. $\mathcal{L}(\theta_g, \theta_d) = \phi_g(\theta_g) + \phi_d(\theta_d)$. Apparently, no GAN objective $\mathcal{L}(\theta_d, \theta_g)$ is decomposable. Therefore, conditional posteriors defined in Eqn. 2 are incompatible. Sampling with incompatible conditional distribution is problematic and leads unpredictable behavior (Arnold & Press, 1989; Chen & Ip, 2015).

Lemma 1. *Assume a joint distribution $p(x, y)$ of variable X and Y. Its corresponding conditional distributions have the forms $p(x|y) \propto \exp\{L(x, y)\}q_x(x)$ and $p(y|x) \propto \exp\{-L(x, y)\}q_y(y)$ only if X and Y are independent, i.e., $p(x, y) = p(x)p(y)$ and $L(x, y)$ is decomposable, i.e., $\exists L_x$ and $L_y, L(x, y) = L_x(x) + L_y(y)$.*

Proof.

$$\begin{aligned} & p(x|y) = \alpha(y) \exp\{L(x, y)\}q_x(x), \\ & p(y|x) = \beta(x) \exp\{-L(x, y)\}q_y(y), \\ \Rightarrow\ & p(x, y)^2 = p(x|y)p(y) \times p(y|x)p(x) \\ & \quad = p(x)p(y)\alpha(y)\beta(x)q_x(x)q_y(y) \\ \Rightarrow\ & X, Y \text{ are independent.} \\ \Rightarrow\ & p(x) = p(x|y) \\ \Rightarrow\ & L(x, y) = \log p(x) - \log q_x(x) - \log \alpha(y) \\ \Rightarrow\ & L(x, y) \text{ is decomposable.} \end{aligned} \tag{10}$$

where $\alpha(y) = \big(\int_{\mathbb{R}} \exp\{L(x, y)\}q_x(x)dx\big)^{-1}$ and $\beta(x) = \big(\int_{\mathbb{R}} \exp\{-L(x, y)\}q_y(y)dy\big)^{-1}$. □

Convergence Issue

In this part, we theoretically analyze a simple task of learning a Bernoulli distribution and show that BGAN fails in converging to the desired solution.

Consider the setting where $\mathcal{X} = \{0, 1\}$ is the data space, $\Theta_d = \{\theta_d^0, \theta_d^1\}$ and $\Theta_g = \{\theta_g^0, \theta_g^1\}$ are the space of generator and discriminator parameters. The data distributions captured by the generators are the following Bernoulli distributions $p_{gen}(x; \theta_g^0) = \text{Bern}(x; 0)$ and $p_{gen}(x; \theta_g^1) = \text{Bern}(x; 1)$, while the discriminators are $D(x; \theta_d^0) = \epsilon \mathbb{1}_{[x=1]} + (1-\epsilon)\mathbb{1}_{[x=0]}$ and $D(x; \theta_d^1) = \epsilon \mathbb{1}_{[x=0]} + (1-\epsilon)\mathbb{1}_{[x=1]}$. Further, any distribution of generators can be parameterized as $q_g(\theta_g; \gamma) = \gamma \mathbb{1}_{[\theta_g = \theta_g^1]} + (1-\gamma)\mathbb{1}_{[\theta_g = \theta_g^0]}$. Suppose the target data distribution we aim to learn is $p_{data} = \text{Bern}(\lambda)$. Lemma 2 below shows that the iterative process described by Eqn. 4 in BGAN fails to converge to the desired generator distribution.

Note that although we take the Bernoulli distribution as an example here, this issue is applicable to any distribution over a finite data set $\mathcal{X}$ with BGAN. The categorical distribution example in Sec. 4 empirically verifies this and shows that our Bayesian model does not suffer from this problem.

Lemma 2. *There exists $\lambda \in (0, 1)$ such that the desired generator distribution $q_g(\theta_g) = q_g(\theta_g; \gamma = \lambda)$ is not a fixed point of the iterative process in Eqn. 4. Here we assume the GAN objective is original min-max version while the following proof can be easily adapted to other GAN objectives.*

Proof. Assume at stage t, $q_g^{(t)}$ reaches q_g. In next iteration, $q_d^{(t+1)}(\theta_d) \propto \exp\{\mathbb{E}_{x \sim p_{data}}[\log D(x; \theta_d) + \log(1 - D(x; \theta_d))]\} p(\theta_d|\alpha_d) \propto (\epsilon(1-\epsilon))^{-1} p(\theta_d|\alpha_d) \propto p(\theta_d|\alpha_d)$. With $q_d^{(t+1)}(\theta_d) = p(\theta_d|\alpha_d)$, $q_g^{(t+2)}$ will be proportional to $\exp\{\mathbb{E}_{\theta_d \sim p(\theta_d|\alpha_d); x \sim p_{gen}(x;\theta_g)} \log(D(x; \theta_d))\} p(\theta_g|\alpha_g)$, which is not relevant to λ. Thus we can find a λ such that $q_g^{(t+2)} \neq q_g^{(t)} = q_g$. □

3.4. Inference Algorithm

So far we have introduced our Bayesian modelling for GAN. In this section we develop novel inference algorithms to compute the posterior. Similar to most complex Bayesian methods, exact calculation of the posterior is *intractable*. Following the strategy in (Saatci & Wilson, 2017), we adopt Stochastic Gradient Hamiltonian Monte Carlo to generate samples from the posterior. In each iteration, M Monte Carlo samples $\{\theta_{g;m}^{(t)}\}_{m=1}^{M}$ are generated to approximate the generator distribution $q_g^{(t)}$.

$$\begin{aligned}\nabla_{\theta_d} \log q_d^{(t+1)}(\theta_d) &= -\nabla_{\theta_d} \mathcal{L}_d(\theta_d; p_{model}^{(t)}) \\ &= -\frac{1}{M_g} \sum_{m=1}^{M_g} \nabla_{\theta_d} \mathcal{L}_d(\theta_d; p_{gen}(\cdot; \theta_{g;m}^{(t)})),\end{aligned} \tag{11}$$

Algorithm 1 Our Meta Inference Algorithm

Input: Initial Monte Carlo samples of $\{\theta_{d;m}^{(0)}\}_{m=1}^{M_d}$ and $\{\theta_{g;m}^{(0)}\}_{m=1}^{M_g}$, learning rate η, SGHMC noise factor α, number of updates in SGHMC procedure L.

for $t = 1, \cdots$ **do**
  for $m = 1$ **to** M_d **do**
    $\theta_{d;m} \leftarrow \theta_{d;m}^{(t)}$
    for $l = 1$ **to** L **do**
      $\mathsf{n} \sim \mathcal{N}(0, 2\alpha\eta I)$
      $\mathsf{v} \leftarrow (1-\alpha)\mathsf{v} + \eta \nabla_{\theta_d} \log q_d^{(t+1)}(\theta_{d;m}) + \mathsf{n}$
      $\theta_{d;m} \leftarrow \theta_{d;m} + \mathsf{v}$
    end for
    $\theta_{d;m}^{(t+1)} \leftarrow \theta_{d;m}$
  end for
  for $m = 1$ **to** M_g **do**
    $\theta_{g;m} \leftarrow \theta_{g;m}^{(t)}$
    for $l = 1$ **to** L **do**
      $\mathsf{n} \sim \mathcal{N}(0, 2\alpha\eta I)$
      $\mathsf{v} \leftarrow (1-\alpha)\mathsf{v} + \eta \nabla_{\theta_g} \log q_g^{(t+1)}(\theta_{g;m}) + \mathsf{n}$
      $\theta_{g;m} \leftarrow \theta_{g;m} + \mathsf{v}$
    end for
    $\theta_{g;m}^{(t+1)} \leftarrow \theta_{g;m}$
  end for
end for

$$\begin{aligned}\nabla_{\theta_g} \log q_g^{(t+1)}(\theta_g) &= -\nabla_{\theta_g} \mathcal{L}_g(\theta_g; D^{(t)}) + \nabla_{\theta_g} \log q_g^{(t)}(\theta_g) \\ &= -\nabla_{\theta_g} \mathcal{L}_g(\theta_g; \frac{1}{M_d} \sum_m D(\cdot; \theta_{d;m}^{(t)})) + \nabla_{\theta_g} \log q_g^{(t)}(\theta_g).\end{aligned} \tag{12}$$

Algorithm 1 is our meta algorithm based on SGHMC. As shown in Eqn. 11 and Eqn. 12, the gradients come from two parts: the GAN objective $\mathcal{L}_g, \mathcal{L}_d$ and the prior $q_g^{(t)}$. Obtain GAN objective's gradient is easy while computing the prior gradient, $\nabla_{\theta_g} \log q_g^{(t)}(\theta_g)$, is actually non-trivial since we have no exact analytic form of $\nabla_{\theta_g} \log q_g^{(t)}(\theta_g)$.

To address this challenge, we propose the following two methods to approximate $\nabla_{\theta_g} \log q_g^{(t)}(\theta_g)$, leading to two practical inference algorithms.

Gaussian Mixture Approximation (GMA): Although the analytic form of the distribution $q_g^{(t)}(\theta_g)$ is unknown, we have M_g Monte Carlo samples $\{\theta_{g;m}^{(t)}\}_{m=1}^{M_g}$ from it which enables us to directly approximate the distribution as a Mixture of Gaussian in Eqn. 13, where σ is a hyperparameter and C is the normalization constant.

$$q_g^{(t+1)}(\theta_g) \approx C \exp\{\sum_{m=1}^{M_g} \frac{\|\theta_g - \theta_{g;m}^{(t)}\|_2^2}{2\sigma^2}\} \tag{13}$$

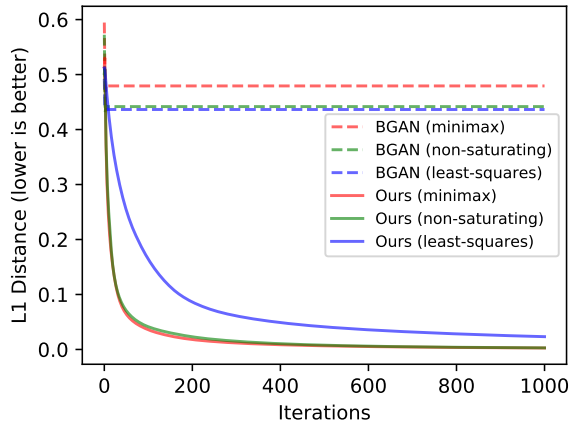

Figure 1. The l_1 distance between generated data distribution and the target versus the number of iterations for our model (in solid line) and BGAN (in dash line). The results with different GAN objectives are indicated by different colors.

Thus the prior gradient can be approximated as following.

$$\nabla_{\theta_g} \log q_g^{(t+1)}(\theta_g) \approx \sum_{m=1}^{M_g} \frac{1}{\sigma^2}(\theta_g - \theta_{g;m}^{(t)}). \tag{14}$$

Partial Summation Approximation (PSA): Based on Eqn. 12, we observe that the gradient of the prior can be recursive unfolded as a summation over all historical GAN objective gradients, shown as:

$$\begin{aligned}\nabla_{\theta_g} \log q_g^{(t+1)}(\theta_g) &= -\nabla_{\theta_g} \mathcal{L}_g(\theta_g; D^{(t)}) + \nabla_{\theta_g} \log q_g^{(t)}(\theta_g) \\ &= -\sum_{i=0} \nabla_{\theta_g} \mathcal{L}_g(\theta_g; D^{(i)}).\end{aligned} \tag{15}$$

It indicates that if we can store all historical discriminator samples $\{\theta_{d;m}^{(i)}\}_{i=1;m=1}^{t;M_d}$, the prior gradient can be computed exactly via the summation. However, computing gradients by all discriminator samples costs huge amount of storage and computational time, which is unaffordable. Hence we propose to maintain a subset of discriminators by subsampling the whole sequence of discriminators.

4. Experiments

4.1. Categorical Distribution

Setup: In this toy example, we consider the case where $\mathcal{X}$, Θ_g, and Θ_d are all finite sets, specifically $\mathcal{X} = \{\mathbf{x}_1, \cdots, \mathbf{x}_N\}$, $\Theta_g = \{\theta_g^1, \cdots, \theta_g^{N_g}\}$, $\Theta_d = \{\theta_d^1, \cdots, \theta_d^{N_d}\}$. The target data distribution is a categorical distribution $\text{Cat}(\lambda_{1:N})$ with $\lambda_i = p_{data}(\mathbf{x}_i)$ as the probability of generating data $\mathbf{x}_i$. The data distribution generated by generator i is another categorical distribution $p_{data}(\mathbf{x}; \theta_g^i) = \text{Cat}(\alpha_{1:N}^i)$. Similarly, the distributions of generator and discriminator can also be parameterized as categorical distributions, i.e. $q_g(\theta_g) = \text{Cat}(\beta_{1:N_g})$ and $q_d(\theta_d) = \text{Cat}(\gamma_{1:N_d})$.

In practice, we set $N = 10, N_g = 20, N_d = 100$. The parameters $\lambda_{1:N}$ of the categorical distribution are obtained by firstly sampling i.i.d $\{\tilde{\lambda}_j\}_{j=1}^N$ from the uniform distribution $\mathcal{U}[0,1]$ and then normalizing $\lambda_j = \frac{\tilde{\lambda}_j}{\sum_{j=1}^N \tilde{\lambda}_j}$. Other categorical distribution parameters $\alpha_{1:N}^i$, $\beta_{1:N_g}$ and $\gamma_{1:N_d}$ are also initialized in a similar way. For the discriminators, their function values are randomly generated from $\{D(\mathbf{x}_i; \theta_d^j)\}_{i=1;j=1}^{N;N_d} \sim \mathcal{U}[0,1]$.

Metric: We employ l_1 distance for evaluation which can be directly computed on categorical distributions as follows.

$$\mathcal{D}_{l_1}(p_{data}, p_{model}) = \sum_{\mathbf{x} \in \mathcal{X}} |p_{data}(\mathbf{x}) - p_{model}(\mathbf{x})|. \tag{16}$$

Evaluation: Our Bayesian model (Eqn. 6) and the BGAN (Eqn. 4) are evaluated with three different GAN objectives: min-max, non-satuaring and LSGAN. which are listed in Table 1. In each iteration step, the distributions of generator and discriminator are updated in a closed form since the normalization constant of the categorical posterior can be easily calculated via summation.

Result: Fig. 1 shows the curves of l_1 distance as a function of number of iterations. For each curve, 20 random trials are averaged. Our model converges to the optimum no matter which objective is employed, showing the robustness of our method. However, the BGAN model is quickly trapped in a bad equilibrium. This might be attributed to the fixed prior of BGAN model which restricts the model space that is explored, while our model maintains a dynamic prior for each iteration which encourages the model to explore towards the optimum.

4.2. High-dimensional Multi-modal Synthetic Dataset

In this experiment, we evaluate our model with two different inference algorithms proposed in Sec. 3.4 (denoted as *ours-GMA* and *ours-PSA*). We also compare with three baselines: 1) *GAN*: naively trained multiple generators in the vanilla GAN framework; 2) *MGAN*: Mixture GAN (Hoang et al., 2018) which is the start-of-art method to train GAN with multiple generators; 3) *BGAN*: Bayesian GAN (Saatci & Wilson, 2017).

Setup: We consider a learning task in a high dimensional space $\mathcal{X} = \mathbb{R}^D$. The target distribution is a uniform mixture of n modes, each lying in a d-dimensional sub-space of $\mathcal{X}$. We call this d-dimensional sub-space as intrinsic sub-space of the i-th mode. Specifically, the data of the i-th mode is generated by the following process,

$$\mathbf{z} \sim \mathcal{U}[-1,1]^d, \quad \mathbf{x} = \mathbf{A}_i(\mathbf{z} + \mathbf{b}_i), \tag{17}$$

where entries of the affine transformation matrix $\mathbf{A}_{i;jk} \sim \mathcal{N}(0, \sigma_A^2)$ and the bias vector $\mathbf{b}_i \sim \mathcal{N}(\mathbf{0}, \sigma_b^2 I_d)$ are drawn from the corresponding Gaussian distributions.

In our experiment, n, D, and d are set to 10, 100, and 2.

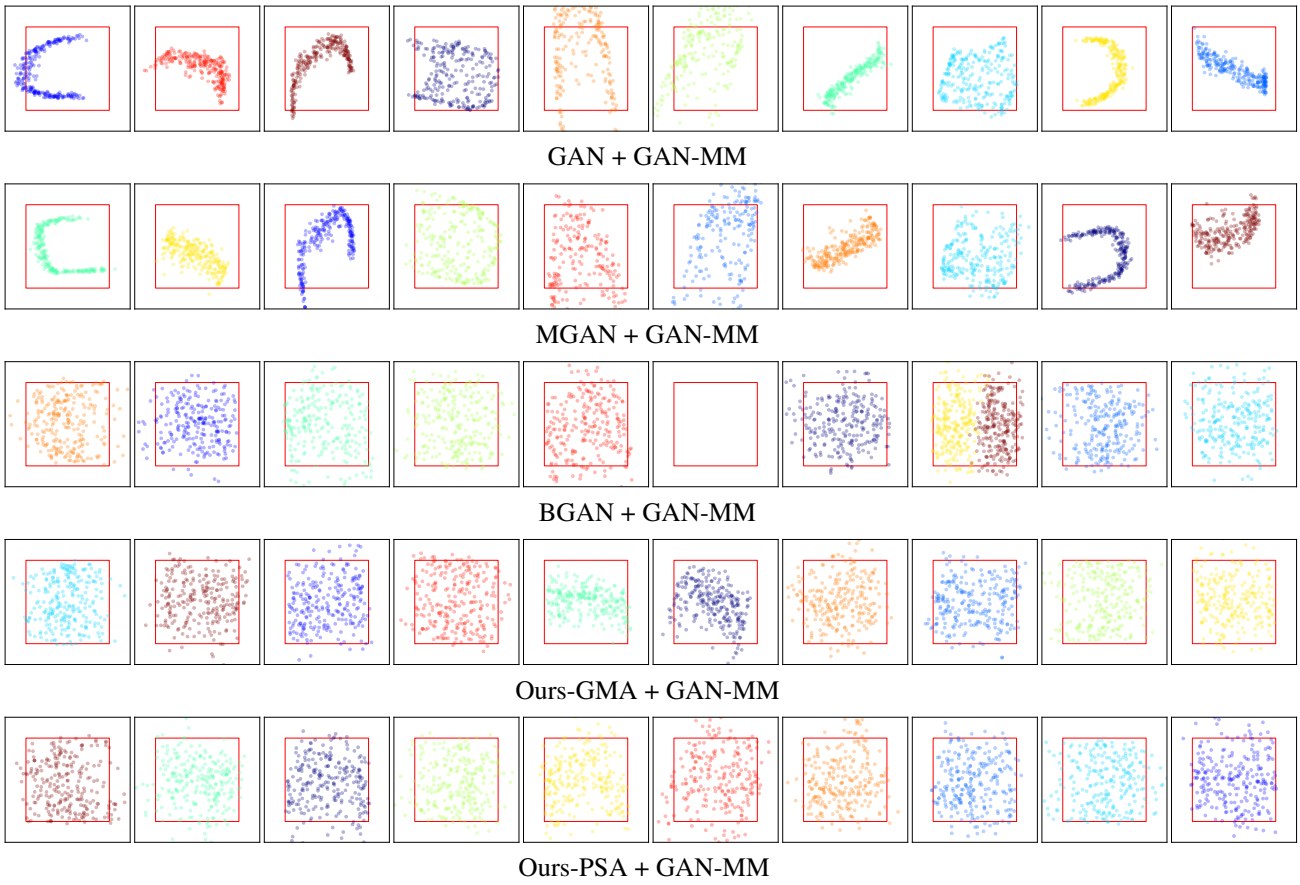

Figure 2. Visualization of the projected hit sets of all models trained with GAN-MM objective. The top two rows show the results of optimization based methods.The bottom three rows present probabilistic method results. In each row, projected hit sets for each mode are plotted in different panels. The red boxes in each panel indicate the region $U[-1,1]^2$ where the target data uniformly distributed. The data points produced by different generators of a model is painted with different colors.

Hyper-parameters for $\mathbf{A}$ and $\mathbf{b}$ are set to be $\sigma_A = \sigma_b = 5$. Each model train ten generators (ten Monte Carlo generator samples for probabilistic models).

Metric: We define *projection error* ϵ_p for generated data sample $\mathbf{x}$ as the minimum of Euclidean distance between $\mathbf{x}$ and the intrinsic sub-spaces of the modes i.e. $\epsilon_p(\mathbf{x}) = \min_i \epsilon_i(\mathbf{x})$, $\|\mathbf{x} - \mathbf{A}_i(\mathbf{A}_i^T\mathbf{A}_i)^{-1}\mathbf{A}_i^T\mathbf{x}\|_2$. Based on the fact that the average distance between the data from two different modes is 800, we set a threshold of $\eta = 40$. Therefore only the data whose Euclidean distance to the subspace of the mode lowers than η is considered as belonging to that mode.

The trained models are evaluated based on $\{\mathbf{x}_k\}_{k=1}^K \sim p_{model}$, the data samples it generates. We define *hit set* $\mathcal{H}_i$, $\{\mathbf{x}_k | \epsilon_i(\mathbf{x}_k) < \eta\}$ which indicates the data samples belong to each mode. We further define *projected hit set*, $\mathcal{PH}_i$, $\{(\mathbf{A}_i^T\mathbf{A}_i)^{-1}\mathbf{A}_i^T\mathbf{x} - \mathbf{b}_i | \mathbf{x} \in \mathcal{H}_i\}$ to project data in each hit set back to their intrinsic sub-space.

Here we introduce three evaluation metrics: *hit ratio*, *hit error*, and *cover error*. *Hit ratio* $\mathcal{H}_r$, $\frac{\sum_{i=1}^n jH_ij}{K}$ is the percentage of generated data actually belonging to the ground truth mode. *Hit error* $\mathcal{H}_e$, $\frac{\sum_{i=1}^n \sum_{\mathbf{x}\in H_i} \epsilon_i(\mathbf{x})}{\sum_{i=1}^n jH_ij}$ is the averaged Euclidean distance between the data and the intrinsic sub-space of the mode. The last metric *cover error* $\mathcal{C}_e$ is to evaluate how well the generated data covers each mode. Essentially it computes the KL-divergence between the estimated distribution of samples in $\mathcal{PH}_i$ and the uniform distribution over $[-1,1]^d$. Formally, it is defined as the averaged KL-divergence on n modes i.e. $\mathcal{C}_e$, $\frac{1}{n}\sum_{i=1}^n \mathrm{KL}(\hat{p}(\cdot;\mathcal{PH}_i)\|\mathcal{U}[-1,1]^d)$. The intuition is that if the generated is close to the ground truth distribution, they should be uniformly distributed in the square area of each mode.

Model Architecture: For fair comparison, we use the same network architecture for each model. Each generator or discriminator is a three layer perceptron. For the generator, the dimensions of input, hidden layer and output are 10, 1000, and 100 respectively. For the discriminator, the dimensions of input, hidden, output layers are 100, 1000, and 1. All activation functions are leaky ReLU (Maas et al., 2013).

Training Details: All models are optimized by Adam (Kingma & Ba, 2014) with a learning rate of 10^{-4}.

Figure 3. Visualization of the projected hit sets of three probabilistic models trained with the WGAN objective.

For probabilistic methods, the SGHMC noise factor η (Algorithm 1) is set as 10^{-1}.

Result: We evaluate all algorithms under the four different GAN objectives introduced in Table 1 referred to as GAN-MM, GAN-NS, WGAN and LSGAN here.

Optimization-based v.s. probabilistic
Table 2 summarizes the results in terms of hit ratio and hit error. Probabilistic methods including our algorithms and BGAN always achieve a hit ratio of 1, which means every data point generated from these models is very close to one mode of the target distribution. On the other hand, optimization based methods, both GAN and MGAN, consistently have a significantly larger hit error, and sometimes may even generate data samples that do not belong to any mode. Moreover, the data distribution generated by the optimization-based methods fits the target uniform distribution much worse than its probabilistic counterparts, which is quantitatively reflected in the cover error showed in Table 3 and visually demonstrated by the projected hit sets in Fig. 2. According to Fig. 2, data generated by GAN or MGAN tend to be under dispersed and hardly cover the whole square region of the true mode, while data generated by probabilistic methods aligns much better with the ground truth distribution. We attributes this superiority to stronger exploration power in the generator space coming from the randomness in probabilistic methods.

Bayesian GAN v.s. our methods
The incompatibility issue of BGAN with minimax-style GAN objectives theoretically derived in Sec. 3.3 is empirically verified in our experiments. As visualized in Fig. 2, with the GAN-MM objective, BGAN is trapped in a local equilibrium and fails in capturing one mode of the true data. Besides, as shown in Table 3, BGAN with the WGAN objective achieves much poorer coverage than with other GAN objectives, while our model is much more robust to the choice of GAN objectives (consistently lower cover errors). A qualitatively comparison is made in Fig. 6 which shows the data distribution generated by BGAN tends to shrink.

More visual illustrations under different GAN objectives are shown in the supplementary Sec. 6.2.

Table 2. Hit ratios (H_r) and hit errors (H_e) of different methods with different GAN objectives. Each cell contains H_r, H_e.

	GAN-MM	GAN-NS	WGAN	LSGAN
GAN	0.86, 22.6	0.85, 23.1	0.78, 26.7	0.74, 23.1
MGAN	0.82, 24.2	0.84, 25.5	0.67, 31.7	0.81, 23.6
BGAN	1.0, 5.5	1.0, 6.4	1.0, 12.1	1.0, 6.3
OURS-GMA	1.0, 7.4	1.0, 7.7	1.0, 15.5	1.0, 5.3
OURS-PSA	1.0, 5.8	1.0, 6.4	1.0, 12.5	1.0, 6.4

Table 3. Cover errors (C_e) of different methods with different GAN objectives. Note, when the model failed to capture all the modes of data distribution, by definition cover error will be ∞, in this case, the averaged KL-divergence on modes captured by the model is reported in brackets.

	GAN-MM	GAN-NS	WGAN	LSGAN
GAN	12.11	8.86	7.20	∞ (12.07)
MGAN	5.46	6.31	5.00	∞ (4.25)
BGAN	∞ (1.73)	1.76	4.32	1.80
OURS-GMA	1.84	1.73	3.01	1.79
OURS-PSA	1.75	1.75	2.28	1.74

5. Conclusion

In this paper, we propose a novel Bayesian modelling framework for GAN, with a likelihood function establishing a connection to existing GAN models and a novel prior stabilizing the inference process. We propose scalable Bayesian inference algorithms which are asymptotically correct. As future work, we plan to extend the proposed framework to non-parametric Bayesian modelling and investigate more theoretical properties of GANs in the Bayesian context.

References

Arjovsky, Martin, Chintala, Soumith, and Bottou, Léon. Wasserstein gan. arXiv preprint arXiv:1701.07875, 2017.

Arnold, Barry C and Press, S James. Compatible conditional distributions. Journal of the American Statistical Association, 84(405):152–156, 1989.

Arora, Sanjeev, Ge, Rong, Liang, Yingyu, Ma, Tengyu, and Zhang, Yi. Generalization and equilibrium in generative adversarial nets (gans). arXiv preprint arXiv:1703.00573, 2017.

Chen, Shyh-Huei and Ip, Edward H. Behaviour of the gibbs sampler when conditional distributions are potentially incompatible. Journal of statistical computation and simulation, 85(16):3266–3275, 2015.

Chen, Tianqi, Fox, Emily, and Guestrin, Carlos. Stochastic gradient hamiltonian monte carlo. In International Conference on Machine Learning, pp. 1683–1691, 2014.

Ghosh, Arnab, Kulharia, Viveka, Namboodiri, Vinay, Torr, Philip HS, and Dokania, Puneet K. Multi-agent diverse generative adversarial networks. arXiv preprint arXiv:1704.02906, 2017.

Goodfellow, Ian. Nips 2016 tutorial: Generative adversarial networks. arXiv preprint arXiv:1701.00160, 2016.

Goodfellow, Ian, Pouget-Abadie, Jean, Mirza, Mehdi, Xu, Bing, Warde-Farley, David, Ozair, Sherjil, Courville, Aaron, and Bengio, Yoshua. Generative adversarial nets. In Advances in neural information processing systems, pp. 2672–2680, 2014.

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

Kingma, Diederik P and Ba, Jimmy. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980, 2014.

Maas, Andrew L, Hannun, Awni Y, and Ng, Andrew Y. Recti er nonlinearities improve neural network acoustic models. In Proc. icml, volume 30, pp. 3, 2013.

Mao, Xudong, Li, Qing, Xie, Haoran, Lau, Raymond YK, Wang, Zhen, and Smolley, Stephen Paul. Least squares generative adversarial networks. In 2017 IEEE International Conference on Computer Vision (ICCV), pp. 2813–2821. IEEE, 2017.

Nowozin, Sebastian, Cseke, Botond, and Tomioka, Ryota. f-gan: Training generative neural samplers using variational divergence minimization. In Advances in Neural Information Processing Systems, pp. 271–279, 2016.

Saatci, Yunus and Wilson, Andrew G. Bayesian gan. In Advances in neural information processing systems, pp. 3622–3631, 2017.

Tolstikhin, Ilya O, Gelly, Sylvain, Bousquet, Olivier, Simon-Gabriel, Carl-Johann, and Schölkopf, Bernhard. Adagan: Boosting generative models. In Advances in Neural Information Processing Systems, pp. 5430–5439, 2017.

Wang, Yaxing, Zhang, Lichao, and van de Weijer, Joost. Ensembles of generative adversarial networks. arXiv preprint arXiv:1612.00991, 2016.

6. Supplementary Materials

6.1. General Version of Theorem 1

In this section, we articulate a general version of Theorem 1 in the main paper. We consider a general GAN objective with the form of Eqn. 19 and Eqn. 18.

$$L_d(\theta_d, \theta_g) = \mathbb{E}_{x \sim p_{data}}[f_1(D(x; \theta_d))] + \mathbb{E}_{x \sim p_{gen}(\cdot; \theta_g)}[f_2(D(x; \theta_d))] \tag{18}$$

$$L_g(\theta_g, \theta_d) = \mathbb{E}_{x \sim p_{gen}(\cdot; \theta_g)}[f_3(D(x; \theta_d)] \tag{19}$$

We can extend our conclusion in Theorem 1 to any GAN objective with a symmetry property over functions f_1 and f_2 as shown in Eqn. 20. Note that all the common choices of GAN objectives including those listed in Table 1 satisfy this property.

$$\exists c \in \mathbb{R}, \forall x \in \mathbb{R}, \quad f_1(x) = f_2(c - x) \tag{20}$$

Theorem 2. Assume the GAN objective used in Eqn. 6 holds the symmetry property (Eqn. 20). If there exists an generator distribution $q_g(\theta_g)$ satisfying $p_{model} = \mathbb{E}_{\theta_g \sim q_g}[p_{gen}(\cdot; \theta_g)] = p_{data}$, then there exists a discriminator distribution q_d such that $D^*(\cdot) = \mathbb{E}_{\theta_d \sim q_d} D(\cdot; \theta_d) \equiv \frac{c}{2}$. Moreover, q_g and q_d are the fixed points of the iterative process defined in Eqn. 6.

Proof. With Eqn. 21 and Eqn. 22 below, we prove that $q_d(\theta_d) / L_d(\theta_d; p_{model})$ satisfies $\mathbb{E}_{\theta_d \sim q_d} D(\cdot; \theta_d) \equiv \frac{c}{2}$. Note that, in the following equations, $S(\theta_d)$ is the symmetric discriminator of θ_d defined as $D(x; S(\theta_d)) \equiv c - D(x; \theta_d)$.

$$\begin{aligned} & q_d(\theta_d) / \mathbb{E}_{x \sim p_{data}}[f_1(D(x; \theta_d))] + \mathbb{E}_{x \sim p_{model}}[f_2(D(x; \theta_d))] \\ &= \mathbb{E}_{x \sim p_{data}}[f_1(D(x; \theta_d)) + f_2(D(x; \theta_d))] \\ &= \mathbb{E}_{x \sim p_{data}}[f_2(c - D(x; \theta_d)) + f_1(c - D(x; \theta_d))] \\ &= \mathbb{E}_{x \sim p_{data}}[f_2(D(x; S(\theta_d))) + f_1(D(x; S(\theta_d)))] \\ &\Rightarrow q_d(\theta_d) = q_d(S(\theta_d)); \end{aligned} \tag{21}$$

$$\begin{aligned} D^*(x) = \mathbb{E}_{\theta_d \sim q_d} D(x; \theta_d) &= \int_{\theta_d \in \Theta_d} q_d(\theta_d) D(x; \theta_d) d\theta_d \\ &= \frac{1}{2}\left(\int_{\theta_d \in \Theta_d} q_d(\theta_d) D(x; \theta_d) d\theta_d + \int_{\theta'_d = S(\theta_d) \in \Theta_d} q_d(\theta'_d) D(x; \theta'_d) d\theta'_d\right) \\ &= \frac{1}{2} \int_{\theta_d \in \Theta_d} q_d(\theta_d)(D(x; \theta_d) + D(x; S(\theta_d))) d\theta_d \\ &= \frac{1}{2} \int_{\theta_d \in \Theta_d} q_d(\theta_d) \; c \; d\theta_d = \frac{c}{2}. \end{aligned} \tag{22}$$

Hence according Eqn. 23, we know $L_g(\theta_g; D^*)$ is a constant which leads to $q_g(\theta_g) / \exp\{L_g(\theta_g; D^*)\} q_g(\theta_g)$.

$$L_g(\theta_g; D^*) = \mathbb{E}_{x \sim p_{gen}(\cdot; \theta_g)}[f_3(D^*(x))] = f_3(\frac{c}{2}), \quad \forall x \in \mathcal{X}. \tag{23}$$

□

6.2. Visualization of Projected Hit Sets

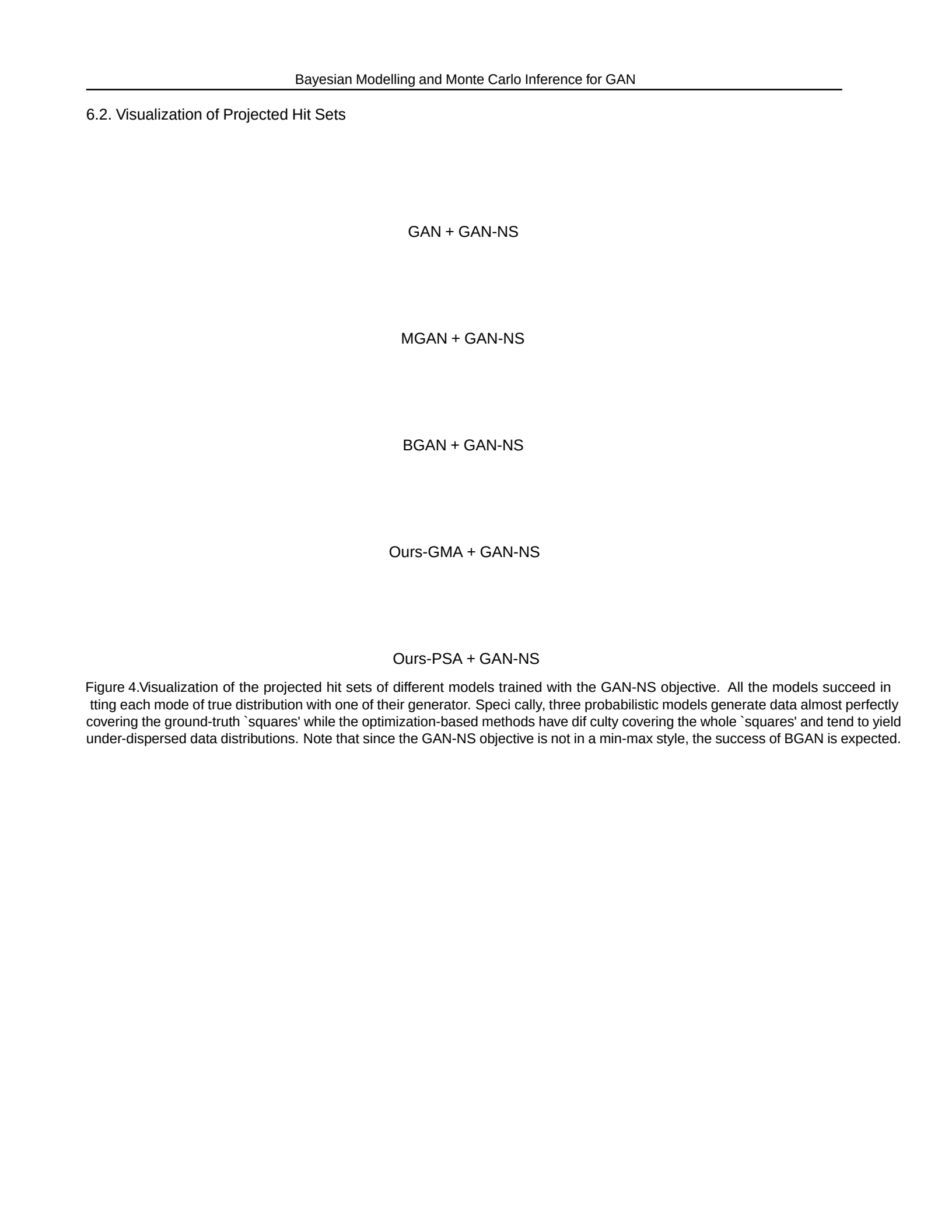

Figure 4.Visualization of the projected hit sets of different models trained with the GAN-NS objective. All the models succeed in tting each mode of true distribution with one of their generator. Speci cally, three probabilistic models generate data almost perfectly covering the ground-truth `squares' while the optimization-based methods have dif culty covering the whole `squares' and tend to yield under-dispersed data distributions. Note that since the GAN-NS objective is not in a min-max style, the success of BGAN is expected.

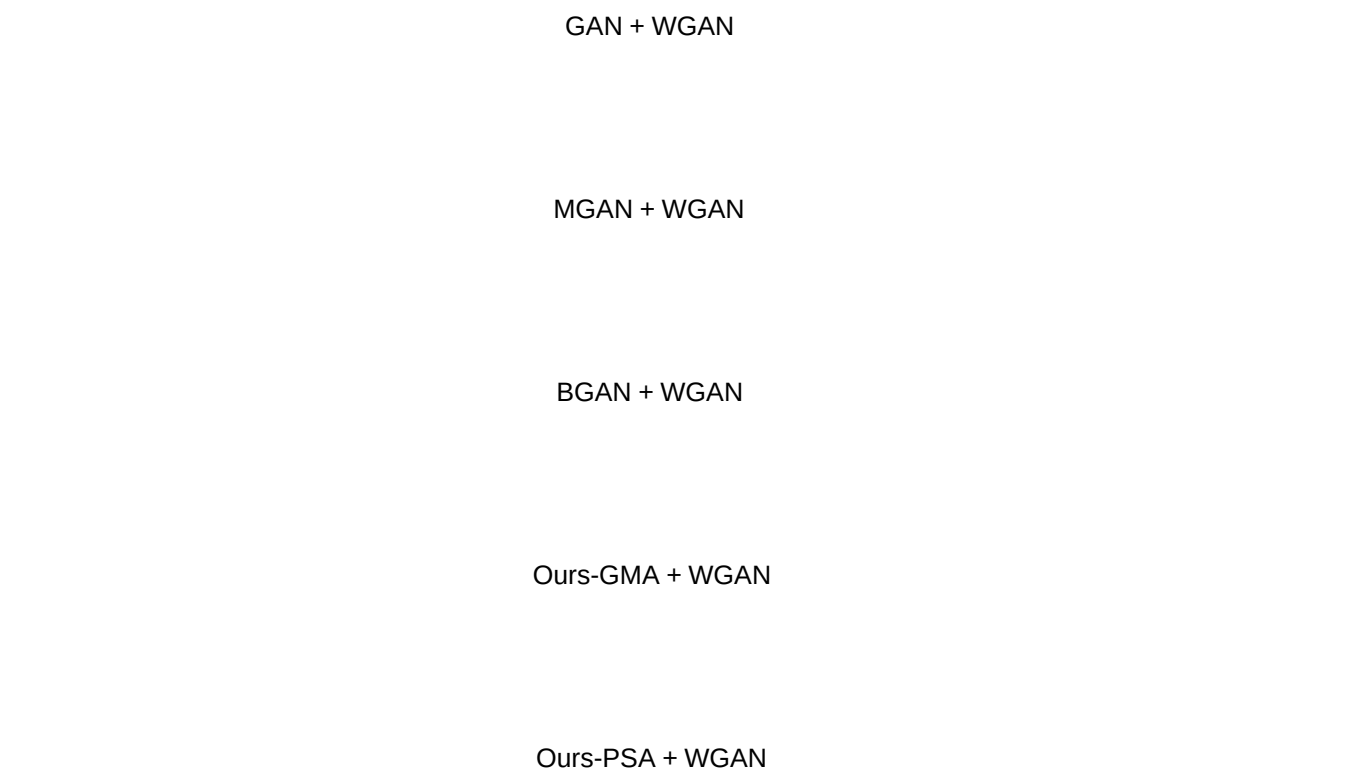

Figure 5.Visualization of the projected hit sets of different models trained with the WGAN objective. As we can see, training the WGAN objective leads to much worse performance for both optimization-based methods and BGAN. On the other hand, our methods are robust to the choice of different GAN objectives and do not suffer from signi cant performance drop when using the WGAN objective.

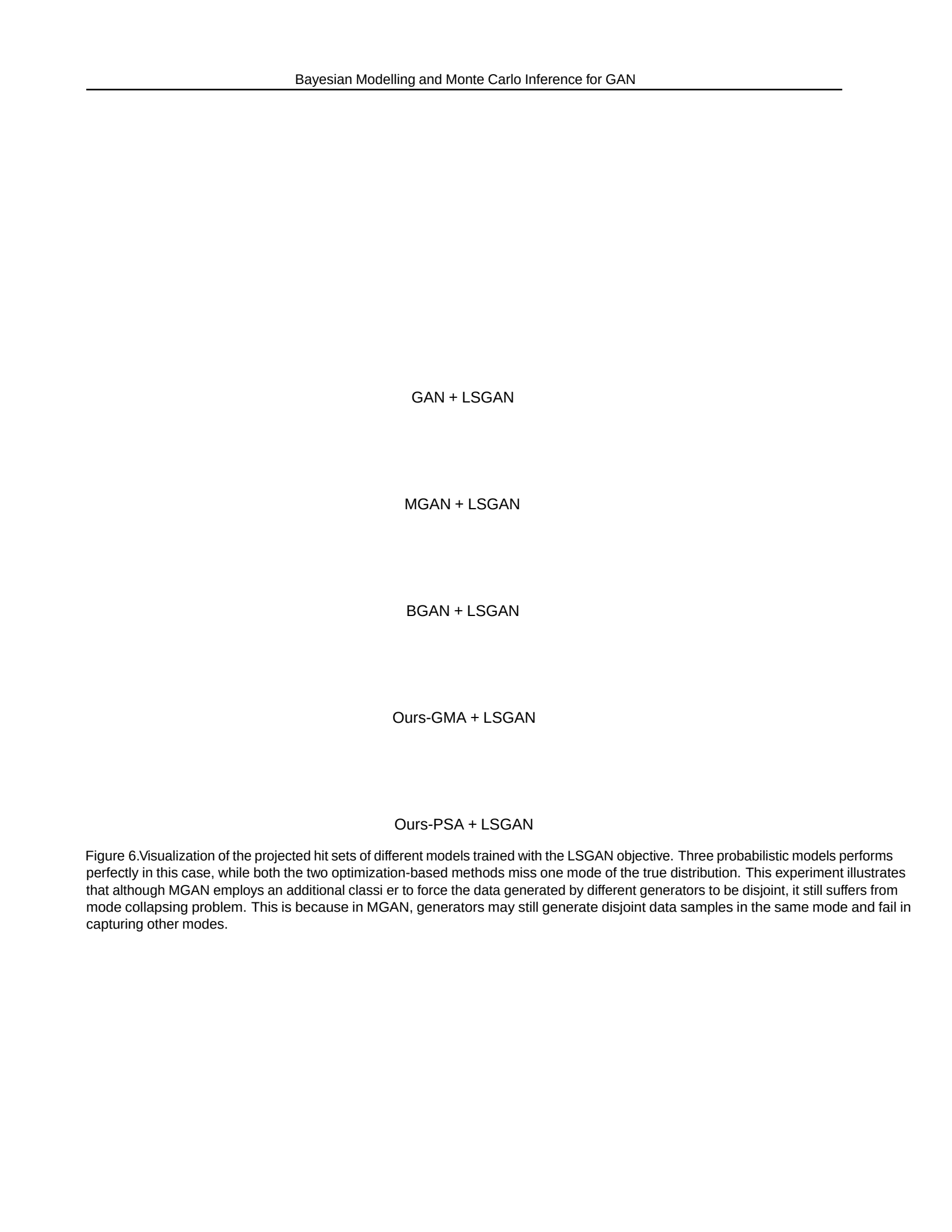

Figure 6.Visualization of the projected hit sets of different models trained with the LSGAN objective. Three probabilistic models performs perfectly in this case, while both the two optimization-based methods miss one mode of the true distribution. This experiment illustrates that although MGAN employs an additional classi er to force the data generated by different generators to be disjoint, it still suffers from mode collapsing problem. This is because in MGAN, generators may still generate disjoint data samples in the same mode and fail in capturing other modes.