Hierarchical Implicit Models and Likelihood-Free Variational Inference

Dustin Tran
Columbia University

Rajesh Ranganath
Princeton University

David M. Blei
Columbia University

Abstract

Implicit probabilistic models are a flexible class of models defined by a simulation process for data. They form the basis for theories which encompass our understanding of the physical world. Despite this fundamental nature, the use of implicit models remains limited due to challenges in specifying complex latent structure in them, and in performing inferences in such models with large data sets. In this paper, we first introduce *hierarchical implicit models* (HIMs). HIMs combine the idea of implicit densities with hierarchical Bayesian modeling, thereby defining models via simulators of data with rich hidden structure. Next, we develop *likelihood-free variational inference* (LFVI), a scalable variational inference algorithm for HIMs. Key to LFVI is specifying a variational family that is also implicit. This matches the model's flexibility and allows for accurate approximation of the posterior. We demonstrate diverse applications: a large-scale physical simulator for predator-prey populations in ecology; a Bayesian generative adversarial network for discrete data; and a deep implicit model for text generation.

1 Introduction

Consider a model of coin tosses. With probabilistic models, one typically posits a latent probability, and supposes each toss is a Bernoulli outcome given this probability [38, 16]. After observing a collection of coin tosses, Bayesian analysis lets us describe our inferences about the probability.

However, we know from the laws of physics that the outcome of a coin toss is fully determined by its initial conditions (say, the impulse and angle of flip) [27, 9]. Therefore a coin toss' randomness does not originate from a latent probability but in noisy initial parameters. This alternative model incorporates the physical system, better capturing the generative process. Furthermore the model is *implicit*, also known as a simulator: we can sample data from its generative process, but we may not have access to calculate its density [11, 22].

Coin tosses are simple, but they serve as a building block for complex implicit models. These models, which capture the laws and theories of real-world physical systems, pervade fields such as population genetics [42], statistical physics [1], and ecology [3]; they underlie structural equation models in economics and causality [41]; and they connect deeply to generative adversarial networks (GANs) [19], which use neural networks to specify a flexible implicit density [37].

Unfortunately, implicit models, including GANs, have seen limited success outside specific domains. There are two reasons. First, it is unknown how to design implicit models for more general applications, exposing rich latent structure such as priors, hierarchies, and sequences. Second, existing methods for inferring latent structure in implicit models do not sufficiently scale to high-dimensional or large data sets. In this paper, we design a new class of implicit models and we develop a new algorithm for accurate and scalable inference.

For modeling, § 2 describes *hierarchical implicit models*, a class of Bayesian hierarchical models which only assume a process that generates samples. This class encompasses both simulators in the

31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA.

classical literature and those employed in GANs. For example, we specify a Bayesian GAN, where we place a prior on its parameters. The Bayesian perspective allows GANs to quantify uncertainty and improve data efficiency. We can also apply them to discrete data; this setting is not possible with traditional estimation algorithms for GANs [29].

For inference, § 3 develops *likelihood-free variational inference (*LFVI*)*, which combines variational inference with density ratio estimation [51, 37]. Variational inference posits a family of distributions over latent variables and then optimizes to find the member closest to the posterior [25]. Traditional approaches require a likelihood-based model and use crude approximations, employing a simple approximating family for fast computation. LFVI expands variational inference to implicit models and enables accurate variational approximations with implicit variational families: LFVI does not require the variational density to be tractable. Further, unlike previous Bayesian methods for implicit models, LFVI scales to millions of data points with stochastic optimization.

This work has diverse applications. First, we analyze a classical problem from the approximate Bayesian computation (ABC) literature, where the model simulates an ecological system [3]. We analyze 100,000 time series which is not possible with traditional methods. Second, we analyze a Bayesian GAN, which is a GAN with a prior over its weights. Bayesian GANs outperform corresponding Bayesian neural networks with known likelihoods on several classification tasks. Third, we show how injecting noise into hidden units of recurrent neural networks corresponds to a deep implicit model for flexible sequence generation.

Related Work. This paper connects closely to three lines of work. The first is Bayesian inference for implicit models, known in the statistics literature as approximate Bayesian computation (ABC) [3, 35]. ABC steps around the intractable likelihood by applying summary statistics to measure the closeness of simulated samples to real observations. While successful in many domains, ABC has shortcomings. First, the results generated by ABC depend heavily on the chosen summary statistics and the closeness measure. Second, as the dimensionality grows, closeness becomes harder to achieve. This is the classic curse of dimensionality.

The second is GANs [19]. GANs have seen much interest since their conception, providing an efficient method for estimation in neural network-based simulators. Larsen et al. [30] propose a hybrid of variational methods and GANs for improved reconstruction. Chen et al. [7] apply information penalties to disentangle factors of variation. Donahue et al. [12], Dumoulin et al. [13] propose to match on an augmented space, simultaneously training the model and an inverse mapping from data to noise. Unlike any of the above, we develop models with explicit priors on latent variables, hierarchies, and sequences, and we generalize GANs to perform Bayesian inference.

The final thread is variational inference with expressive approximations [47, 50, 54]. The idea of casting the design of variational families as a modeling problem was proposed in Ranganath et al. [46]. Further advances have analyzed variational programs [44]—a family of approximations which only requires a process returning samples—and which has seen further interest [32]. Implicit-like variational approximations have also appeared in auto-encoder frameworks [34, 36] and message passing [26]. We build on variational programs for inferring implicit models.

2 Hierarchical Implicit Models

Hierarchical models play an important role in sharing statistical strength across examples [17]. For a broad class of hierarchical Bayesian models, the joint distribution of the hidden and observed variables is

$$p(\mathbf{x}, \mathbf{z}, \boldsymbol{\beta}) = p(\boldsymbol{\beta}) \prod_{n=1}^{N} p(\mathbf{x}_n \mid \mathbf{z}_n, \boldsymbol{\beta}) p(\mathbf{z}_n \mid \boldsymbol{\beta}), \tag{1}$$

where $\mathbf{x}_n$ is an observation, $\mathbf{z}_n$ are latent variables associated to that observation (local variables), and $\boldsymbol{\beta}$ are latent variables shared across observations (global variables). See Fig. 1 (left).

With hierarchical models, local variables can be used for clustering in mixture models, mixed memberships in topic models [4], and factors in probabilistic matrix factorization [49]. Global variables can be used to pool information across data points for hierarchical regression [17], topic models [4], and Bayesian nonparametrics [52].

Hierarchical models typically use a tractable likelihood $p(\mathbf{x}_n \mid \mathbf{z}_n, \boldsymbol{\beta})$. But many likelihoods of interest, such as simulator-based models [22] and generative adversarial networks [19], admit high

Figure 1: (**left**) Hierarchical model, with local variables $\mathbf{z}$ and global variables β. (**right**) **Hierarchical implicit model**. It is a hierarchical model where $\mathbf{x}$ is a deterministic function (denoted with a square) of noise $\boldsymbol{\epsilon}$ (denoted with a triangle).

fidelity to the true data generating process and do not admit a tractable likelihood. To overcome this limitation, we develop *hierarchical implicit models* (HIMs).

Hierarchical implicit models have the same joint factorization as Eq.1 but only assume that one can sample from the likelihood. Rather than define $p(\mathbf{x}_n \mid \mathbf{z}_n, \boldsymbol{\beta})$ explicitly, HIMs define a function g that takes in random noise $\boldsymbol{\epsilon}_n \sim s(\cdot)$ and outputs $\mathbf{x}_n$ given $\mathbf{z}_n$ and $\boldsymbol{\beta}$,

$$\mathbf{x}_n = g(\boldsymbol{\epsilon}_n \mid \mathbf{z}_n, \boldsymbol{\beta}), \quad \boldsymbol{\epsilon}_n \sim s(\cdot).$$

The induced, implicit likelihood of $\mathbf{x}_n \in A$ given $\mathbf{z}_n$ and $\boldsymbol{\beta}$ is

$$\mathcal{P}(\mathbf{x}_n \in A \mid \mathbf{z}_n, \boldsymbol{\beta}) = \int_{\{g(\boldsymbol{\epsilon}_n \mid \mathbf{z}_n, \boldsymbol{\beta}) = \mathbf{x}_n \in A\}} s(\boldsymbol{\epsilon}_n)\, \mathrm{d}\boldsymbol{\epsilon}_n.$$

This integral is typically intractable. It is difficult to find the set to integrate over, and the integration itself may be expensive for arbitrary noise distributions $s(\cdot)$ and functions g.

Fig. 1 (right) displays the graphical model for HIMs. Noise ($\boldsymbol{\epsilon}_n$) are denoted by triangles; deterministic computation ($\mathbf{x}_n$) are denoted by squares. We illustrate two examples.

Example: Physical Simulators. Given initial conditions, simulators describe a stochastic process that generates data. For example, in population ecology, the Lotka-Volterra model simulates predator-prey populations over time via a stochastic differential equation [57]. For prey and predator populations $x_1, x_2 \in \mathbb{R}^+$ respectively, one process is

$$\begin{aligned}
\frac{\mathrm{d}x_1}{\mathrm{d}t} &= \beta_1 x_1 - \beta_2 x_1 x_2 + \epsilon_1, \qquad \epsilon_1 \sim \text{Normal}(0, 10), \\
\frac{\mathrm{d}x_2}{\mathrm{d}t} &= -\beta_2 x_2 + \beta_3 x_1 x_2 + \epsilon_2, \quad \epsilon_2 \sim \text{Normal}(0, 10),
\end{aligned}$$

where Gaussian noises ϵ_1, ϵ_2 are added at each full time step. The simulator runs for T time steps given initial population sizes for x_1, x_2. Lognormal priors are placed over β. The Lotka-Volterra model is grounded by theory but features an intractable likelihood. We study it in § 4.

Example: Bayesian Generative Adversarial Network. Generative adversarial networks (GANs) define an implicit model and a method for parameter estimation [19]. They are known to perform well on image generation [43]. Formally, the implicit model for a GAN is

$$\mathbf{x}_n = g(\boldsymbol{\epsilon}_n; \boldsymbol{\theta}), \quad \boldsymbol{\epsilon}_n \sim s(\cdot), \tag{2}$$

where g is a neural network with parameters $\boldsymbol{\theta}$, and s is a standard normal or uniform. The neural network g is typically not invertible; this makes the likelihood intractable.

The parameters $\boldsymbol{\theta}$ in GANs are estimated by divergence minimization between the generated and real data. We make GANs amenable to Bayesian analysis by placing a prior on the parameters $\boldsymbol{\theta}$. We call this a Bayesian GAN. Bayesian GANs enable modeling of parameter uncertainty and are inspired by Bayesian neural networks, which have been shown to improve the uncertainty and data efficiency of standard neural networks [33, 39]. We study Bayesian GANs in § 4; Appendix B provides example implementations in the Edward probabilistic programming language [55].

3 Likelihood-Free Variational Inference

We described hierarchical implicit models, a rich class of latent variable models with local and global structure alongside an implicit density. Given data, we aim to calculate the model's posterior $p(\mathbf{z}, \boldsymbol{\beta} \mid \mathbf{x}) = p(\mathbf{x}, \mathbf{z}, \boldsymbol{\beta})/p(\mathbf{x})$. This is difficult as the normalizing constant $p(\mathbf{x})$ is typically

intractable. With implicit models, the lack of a likelihood function introduces an additional source of intractability.

We use variational inference [25]. It posits an approximating family $q \in \mathcal{Q}$ and optimizes to find the member closest to $p(\mathbf{z}, \boldsymbol{\beta} \mid \mathbf{x})$. There are many choices of variational objectives that measure closeness [44, 31, 10]. To choose an objective, we lay out desiderata for a variational inference algorithm for implicit models:

1. *Scalability*. Machine learning hinges on stochastic optimization to scale to massive data [6]. The variational objective should admit unbiased subsampling with the standard technique,

$$\sum_{n=1}^{N} f(\mathbf{x}_n) \approx \frac{N}{M} \sum_{m=1}^{M} f(\mathbf{x}_m),$$

   where some computation $f(\cdot)$ over the full data is approximated with a mini-batch of data $\{\mathbf{x}_m\}$.
2. *Implicit Local Approximations*. Implicit models specify flexible densities; this induces very complex posterior distributions. Thus we would like a rich approximating family for the per-data point approximations $q(\mathbf{z}_n \mid \mathbf{x}_n, \boldsymbol{\beta})$. This means the variational objective should only require that one can sample $\mathbf{z}_n \sim q(\mathbf{z}_n \mid \mathbf{x}_n, \boldsymbol{\beta})$ and not evaluate its density.

One variational objective meeting our desiderata is based on the classical minimization of the Kullback-Leibler (KL) divergence. (Surprisingly, Appendix C details how the KL is the *only* possible objective among a broad class.)

3.1 KL Variational Objective

Classical variational inference minimizes the KL divergence from the variational approximation q to the posterior. This is equivalent to maximizing the *evidence lower bound* (ELBO),

$$\mathcal{L} = \mathbb{E}_{q(\boldsymbol{\beta}, \mathbf{z} \mid \mathbf{x})}[\log p(\mathbf{x}, \mathbf{z}, \boldsymbol{\beta}) - \log q(\boldsymbol{\beta}, \mathbf{z} \mid \mathbf{x})]. \tag{3}$$

Let q factorize in the same way as the posterior,

$$q(\boldsymbol{\beta}, \mathbf{z} \mid \mathbf{x}) = q(\boldsymbol{\beta}) \prod_{n=1}^{N} q(\mathbf{z}_n \mid \mathbf{x}_n, \boldsymbol{\beta}),$$

where $q(\mathbf{z}_n \mid \mathbf{x}_n, \boldsymbol{\beta})$ is an intractable density and since the data $\mathbf{x}$ is constant during inference, we drop conditioning for the global $q(\boldsymbol{\beta})$. Substituting p and q's factorization yields

$$\mathcal{L} = \mathbb{E}_{q(\boldsymbol{\beta})}[\log p(\boldsymbol{\beta}) - \log q(\boldsymbol{\beta})] + \sum_{n=1}^{N} \mathbb{E}_{q(\boldsymbol{\beta}) q(\mathbf{z}_n \mid \mathbf{x}_n, \boldsymbol{\beta})}[\log p(\mathbf{x}_n, \mathbf{z}_n \mid \boldsymbol{\beta}) - \log q(\mathbf{z}_n \mid \mathbf{x}_n, \boldsymbol{\beta})].$$

This objective presents difficulties: the local densities $p(\mathbf{x}_n, \mathbf{z}_n \mid \boldsymbol{\beta})$ and $q(\mathbf{z}_n \mid \mathbf{x}_n, \boldsymbol{\beta})$ are both intractable. To solve this, we consider ratio estimation.

3.2 Ratio Estimation for the KL Objective

Let $q(\mathbf{x}_n)$ be the empirical distribution on the observations $\mathbf{x}$ and consider using it in a "variational joint" $q(\mathbf{x}_n, \mathbf{z}_n \mid \boldsymbol{\beta}) = q(\mathbf{x}_n) q(\mathbf{z}_n \mid \mathbf{x}_n, \boldsymbol{\beta})$. Now subtract the log empirical $\log q(\mathbf{x}_n)$ from the ELBO above. The ELBO reduces to

$$\mathcal{L} \propto \mathbb{E}_{q(\boldsymbol{\beta})}[\log p(\boldsymbol{\beta}) - \log q(\boldsymbol{\beta})] + \sum_{n=1}^{N} \mathbb{E}_{q(\boldsymbol{\beta}) q(\mathbf{z}_n \mid \mathbf{x}_n, \boldsymbol{\beta})} \left[\log \frac{p(\mathbf{x}_n, \mathbf{z}_n \mid \boldsymbol{\beta})}{q(\mathbf{x}_n, \mathbf{z}_n \mid \boldsymbol{\beta})} \right]. \tag{4}$$

(Here the proportionality symbol means equality up to additive constants.) Thus the ELBO is a function of the ratio of two intractable densities. If we can form an estimator of this ratio, we can proceed with optimizing the ELBO.

We apply techniques for ratio estimation [51]. It is a key idea in GANs [37, 56], and similar ideas have rearisen in statistics and physics [21, 8]. In particular, we use class probability estimation: given a sample from $p(\cdot)$ or $q(\cdot)$ we aim to estimate the probability that it belongs to $p(\cdot)$. We model

this using $\sigma(r(\cdot;\boldsymbol{\theta}))$, where r is a parameterized function (e.g., neural network) taking sample inputs and outputting a real value; σ is the logistic function outputting the probability.

We train $r(\cdot;\boldsymbol{\theta})$ by minimizing a loss function known as a proper scoring rule [18]. For example, in experiments we use the log loss,

$$\mathcal{D}_{\text{log}} = \mathbb{E}_{p(\mathbf{x}_n,\mathbf{z}_n \mid \boldsymbol{\beta})}[-\log\sigma(r(\mathbf{x}_n,\mathbf{z}_n,\boldsymbol{\beta};\boldsymbol{\theta}))] + \mathbb{E}_{q(\mathbf{x}_n,\mathbf{z}_n \mid \boldsymbol{\beta})}[-\log(1-\sigma(r(\mathbf{x}_n,\mathbf{z}_n,\boldsymbol{\beta};\boldsymbol{\theta})))]. \quad (5)$$

The loss is zero if $\sigma(r(\cdot;\boldsymbol{\theta}))$ returns 1 when a sample is from $p(\cdot)$ and 0 when a sample is from $q(\cdot)$. (We also experiment with the hinge loss; see § 4.) If $r(\cdot;\boldsymbol{\theta})$ is sufficiently expressive, minimizing the loss returns the optimal function [37],

$$r^*(\mathbf{x}_n,\mathbf{z}_n,\boldsymbol{\beta}) = \log p(\mathbf{x}_n,\mathbf{z}_n \mid \boldsymbol{\beta}) - \log q(\mathbf{x}_n,\mathbf{z}_n \mid \boldsymbol{\beta}).$$

As we minimize Eq.5, we use $r(\cdot;\boldsymbol{\theta})$ as a proxy to the log ratio in Eq.4. Note r estimates the log ratio; it's of direct interest and more numerically stable than the ratio.

The gradient of $\mathcal{D}_{\text{log}}$ with respect to $\boldsymbol{\theta}$ is

$$\mathbb{E}_{p(\mathbf{x}_n,\mathbf{z}_n \mid \boldsymbol{\beta})}[\nabla_{\boldsymbol{\theta}}\log\sigma(r(\mathbf{x}_n,\mathbf{z}_n,\boldsymbol{\beta};\boldsymbol{\theta}))] + \mathbb{E}_{q(\mathbf{x}_n,\mathbf{z}_n \mid \boldsymbol{\beta})}[\nabla_{\boldsymbol{\theta}}\log(1-\sigma(r(\mathbf{x}_n,\mathbf{z}_n,\boldsymbol{\beta};\boldsymbol{\theta})))]. \quad (6)$$

We compute unbiased gradients with Monte Carlo.

3.3 Stochastic Gradients of the KL Objective

To optimize the ELBO, we use the ratio estimator,

$$\mathcal{L} = \mathbb{E}_{q(\boldsymbol{\beta} \mid \mathbf{x})}[\log p(\boldsymbol{\beta}) - \log q(\boldsymbol{\beta})] + \sum_{n=1}^{N}\mathbb{E}_{q(\boldsymbol{\beta} \mid \mathbf{x})q(\mathbf{z}_n \mid \mathbf{x}_n,\boldsymbol{\beta})}[r(\mathbf{x}_n,\mathbf{z}_n,\beta)]. \quad (7)$$

All terms are now tractable. We can calculate gradients to optimize the variational family q. Below we assume the priors $p(\boldsymbol{\beta}), p(\mathbf{z}_n \mid \boldsymbol{\beta})$ are differentiable. (We discuss methods to handle discrete global variables in the next section.)

We focus on reparameterizable variational approximations [28, 48]. They enable sampling via a differentiable transformation T of random noise, $\delta \sim s(\cdot)$. Due to Eq.7, we require the global approximation $q(\boldsymbol{\beta};\boldsymbol{\lambda})$ to admit a tractable density. With reparameterization, its sample is

$$\boldsymbol{\beta} = T_{\text{global}}(\boldsymbol{\delta}_{\text{global}};\boldsymbol{\lambda}), \quad \boldsymbol{\delta}_{\text{global}} \sim s(\cdot),$$

for a choice of transformation $T_{\text{global}}(\cdot;\boldsymbol{\lambda})$ and noise $s(\cdot)$. For example, setting $s(\cdot) = \mathcal{N}(0,1)$ and $T_{\text{global}}(\boldsymbol{\delta}_{\text{global}}) = \mu + \sigma\boldsymbol{\delta}_{\text{global}}$ induces a normal distribution $\mathcal{N}(\mu,\sigma^2)$.

Similarly for the local variables $\mathbf{z}_n$, we specify

$$\mathbf{z}_n = T_{\text{local}}(\boldsymbol{\delta}_n,\mathbf{x}_n,\boldsymbol{\beta};\boldsymbol{\phi}), \quad \boldsymbol{\delta}_n \sim s(\cdot).$$

Unlike the global approximation, the local variational density $q(\mathbf{z}_n \mid \mathbf{x}_n;\boldsymbol{\phi})$ need not be tractable: the ratio estimator relaxes this requirement. It lets us leverage implicit models not only for data but also for approximate posteriors. In practice, we also amortize computation with inference networks, sharing parameters ϕ across the per-data point approximate posteriors.

The gradient with respect to global parameters $\boldsymbol{\lambda}$ under this approximating family is

$$\nabla_{\boldsymbol{\lambda}}\mathcal{L} = \mathbb{E}_{s(\boldsymbol{\delta}_{\text{global}})}[\nabla_{\boldsymbol{\lambda}}(\log p(\boldsymbol{\beta}) - \log q(\boldsymbol{\beta}))] + \sum_{n=1}^{N}\mathbb{E}_{s(\boldsymbol{\delta}_{\text{global}})s_n(\boldsymbol{\delta}_n)}[\nabla_{\boldsymbol{\lambda}}r(\mathbf{x}_n,\mathbf{z}_n,\beta)]. \quad (8)$$

The gradient backpropagates through the local sampling $\mathbf{z}_n = T_{\text{local}}(\boldsymbol{\delta}_n,\mathbf{x}_n,\boldsymbol{\beta};\boldsymbol{\phi})$ and the global reparameterization $\boldsymbol{\beta} = T_{\text{global}}(\boldsymbol{\delta}_{\text{global}};\boldsymbol{\lambda})$. We compute unbiased gradients with Monte Carlo. The gradient with respect to local parameters $\boldsymbol{\phi}$ is

$$\nabla_{\boldsymbol{\phi}}\mathcal{L} = \sum_{n=1}^{N}\mathbb{E}_{q(\boldsymbol{\beta})s(\boldsymbol{\delta}_n)}[\nabla_{\boldsymbol{\phi}}r(\mathbf{x}_n,\mathbf{z}_n,\boldsymbol{\beta})]. \quad (9)$$

where the gradient backpropagates through T_{local}.[1]

Algorithm 1: Likelihood-free variational inference (LFVI)

Input : Model $\mathbf{x}_n, \mathbf{z}_n \sim p(\cdot \mid \boldsymbol{\beta})$, $p(\boldsymbol{\beta})$
Variational approximation $\mathbf{z}_n \sim q(\cdot \mid \mathbf{x}_n, \boldsymbol{\beta}; \boldsymbol{\phi})$, $q(\boldsymbol{\beta} \mid \mathbf{x}; \boldsymbol{\lambda})$,
Ratio estimator $r(\cdot; \boldsymbol{\theta})$
Output: Variational parameters $\boldsymbol{\lambda}$, $\boldsymbol{\phi}$
Initialize $\boldsymbol{\theta}$, $\boldsymbol{\lambda}$, $\boldsymbol{\phi}$ randomly.
while *not converged* **do**
  Compute unbiased estimate of $\nabla_{\boldsymbol{\theta}} \mathcal{D}$ (Eq.6), $\nabla_{\boldsymbol{\lambda}} \mathcal{L}$ (Eq.8), $\nabla_{\boldsymbol{\phi}} \mathcal{L}$ (Eq.9).
  Update $\boldsymbol{\theta}$, $\boldsymbol{\lambda}$, $\boldsymbol{\phi}$ using stochastic gradient descent.
end

3.4 Algorithm

Algorithm 1 outlines the procedure. We call it *likelihood-free variational inference (*LFVI*)*. LFVI is black box: it applies to models in which one can simulate data and local variables, and calculate densities for the global variables. LFVI first updates $\boldsymbol{\theta}$ to improve the ratio estimator r. Then it uses r to update parameters $\{\boldsymbol{\lambda}, \boldsymbol{\phi}\}$ of the variational approximation q. We optimize r and q simultaneously. The algorithm is available in Edward [55].

LFVI is scalable: we can unbiasedly estimate the gradient over the full data set with mini-batches [24]. The algorithm can also handle models of either continuous or discrete data. The requirement for differentiable global variables and reparameterizable global approximations can be relaxed using score function gradients [45].

Point estimates of the global parameters $\boldsymbol{\beta}$ suffice for many applications [19, 48]. Algorithm 1 can find point estimates: place a point mass approximation q on the parameters $\boldsymbol{\beta}$. This simplifies gradients and corresponds to variational EM.

4 Experiments

We developed new models and inference. For experiments, we study three applications: a large-scale physical simulator for predator-prey populations in ecology; a Bayesian GAN for supervised classification; and a deep implicit model for symbol generation. In addition, Appendix F, provides practical advice on how to address the stability of the ratio estimator by analyzing a toy experiment. We initialize parameters from a standard normal and apply gradient descent with ADAM.

Lotka-Volterra Predator-Prey Simulator. We analyze the Lotka-Volterra simulator of § 2 and follow the same setup and hyperparameters of Papamakarios and Murray [40]. Its global variables $\boldsymbol{\beta}$ govern rates of change in a simulation of predator-prey populations. To infer them, we posit a mean-field normal approximation (reparameterized to be on the same support) and run Algorithm 1 with both a log loss and hinge loss for the ratio estimation problem; Appendix D details the hinge loss. We compare to rejection ABC, MCMC-ABC, and SMC-ABC [35]. MCMC-ABC uses a spherical Gaussian proposal; SMC-ABC is manually tuned with a decaying epsilon schedule; all ABC methods are tuned to use the best performing hyperparameters such as the tolerance error.

Fig. 2 displays results on two data sets. In the top figures and bottom left, we analyze data consisting of a simulation for $T = 30$ time steps, with recorded values of the populations every 0.2 time units. The bottom left figure calculates the negative log probability of the true parameters over the tolerance error for ABC methods; smaller tolerances result in more accuracy but slower runtime. The top figures compare the marginal posteriors for two parameters using the smallest tolerance for the ABC methods. Rejection ABC, MCMC-ABC, and SMC-ABC all contain the true parameters in their 95% credible interval but are less confident than our methods. Further, they required $100,000$ simulations from the model, with an acceptance rate of 0.004% and 2.990% for rejection ABC and MCMC-ABC respectively.

[1]The ratio r indirectly depends on $\boldsymbol{\phi}$ but its gradient w.r.t. $\boldsymbol{\phi}$ disappears. This is derived via the score function identity and the product rule (see, e.g., Ranganath et al. [45, Appendix]).

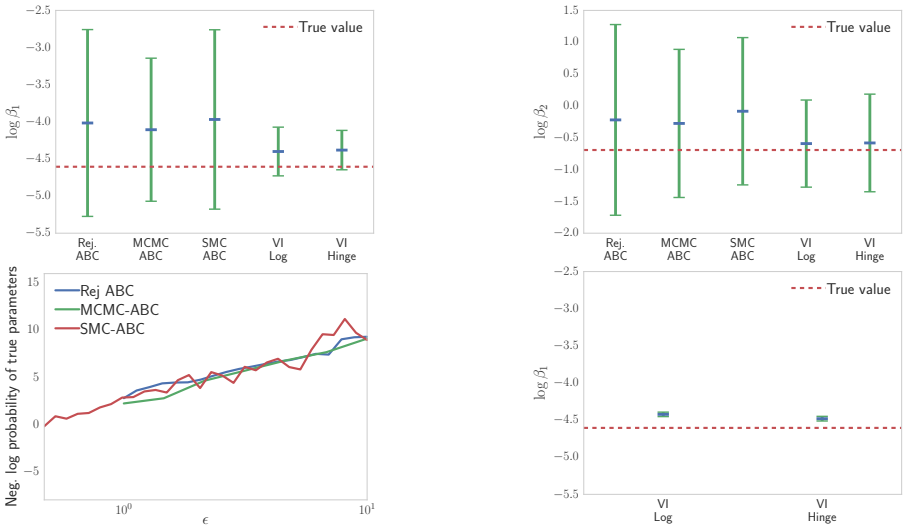


Figure 2: (top) Marginal posterior for first two parameters. **(bot. left)** ABC methods over tolerance error. **(bot. right)** Marginal posterior for first parameter on a large-scale data set. Our inference achieves more accurate results and scales to massive data.

	Test Set Error			
Model + Inference	Crabs	Pima	Covertype	MNIST
Bayesian GAN + VI	0.03	**0.232**	**0.154**	**0.0136**
Bayesian GAN + MAP	0.12	0.240	0.185	0.0283
Bayesian NN + VI	**0.02**	0.242	0.164	0.0311
Bayesian NN + MAP	0.05	0.320	0.188	0.0623

Table 1: Classification accuracy of Bayesian GAN and Bayesian neural networks across small to medium-size data sets. Bayesian GANs achieve comparable or better performance to their Bayesian neural net counterpart.

The bottom right figure analyzes data consisting of $100,000$ time series, each of the same size as the single time series analyzed in the previous figures. This size is not possible with traditional methods. Further, we see that with our methods, the posterior concentrates near the truth. We also experienced little difference in accuracy between using the log loss or the hinge loss for ratio estimation.

Bayesian Generative Adversarial Networks. We analyze Bayesian GANs, described in § 2. Mimicking a use case of Bayesian neural networks [5, 23], we apply Bayesian GANs for classification on small to medium-size data. The GAN defines a conditional $p(y_n \mid \mathbf{x}_n)$, taking a feature $\mathbf{x}_n \in \mathbb{R}^D$ as input and generating a label $y_n \in \{1, \dots, K\}$, via the process

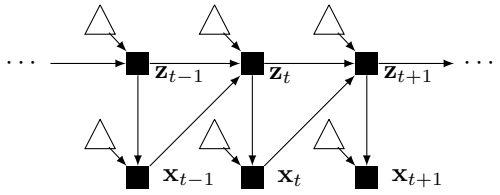
$$y_n = g(\mathbf{x}_n, \boldsymbol{\epsilon}_n \mid \boldsymbol{\theta}), \qquad \boldsymbol{\epsilon}_n \sim \mathcal{N}(0, 1), \tag{10}$$

where $g(\cdot \mid \boldsymbol{\theta})$ is a 2-layer multilayer perception with ReLU activations, batch normalization, and is parameterized by weights and biases $\boldsymbol{\theta}$. We place normal priors, $\boldsymbol{\theta} \sim \mathcal{N}(0, 1)$.

We analyze two choices of the variational model: one with a mean-field normal approximation for $q(\boldsymbol{\theta} \mid \mathbf{x})$, and another with a point mass approximation (equivalent to maximum a posteriori). We compare to a Bayesian neural network, which uses the same generative process as Eq.10 but draws from a Categorical distribution rather than feeding noise into the neural net. We fit it separately using a mean-field normal approximation and maximum a posteriori. Table 1 shows that Bayesian GANs generally outperform their Bayesian neural net counterpart.

Note that Bayesian GANs can analyze discrete data such as in generating a classification label. Traditional GANs for discrete data is an open challenge [29]. In Appendix E, we compare Bayesian GANs with point estimation to typical GANs. Bayesian GANs are also able to leverage parameter uncertainty for analyzing these small to medium-size data sets.

One problem with Bayesian GANs is that they cannot work with very large neural networks: the ratio estimator is a function of global parameters, and thus the input size grows with the size of the



(a) A deep implicit model for sequences. It is a recurrent neural network (RNN) with noise injected into each hidden state. The hidden state is now an implicit latent variable. The same occurs for generating outputs.

```
1  −x+x/x**x*//x*x+
2  x/x*x+x*x/x+x+x+
3  /+x*x+x*x/x/x+x+
4  /x+*x+x*x/x+x−x+
5  x/x*x/x*x+x+x+x−
6  x+x+x/x*x*x+x/x+
```

(b) Generated symbols from the implicit model. Good samples place arithmetic operators between the variable x. The implicit model learned to follow rules from the context free grammar up to some multiple operator repeats.

neural network. One approach is to make the ratio estimator not a function of the global parameters. Instead of optimizing model parameters via variational EM, we can train the model parameters by backpropagating through the ratio objective instead of the variational objective. An alternative is to use the hidden units as input which is much lower dimensional [53, Appendix C].

Injecting Noise into Hidden Units. In this section, we show how to build a hierarchical implicit model by simply injecting randomness into hidden units. We model sequences $\mathbf{x} = (\mathbf{x}_1, \dots, \mathbf{x}_T)$ with a recurrent neural network. For $t = 1, \dots, T$,

$$\begin{aligned} \mathbf{z}_t &= g_z(\mathbf{x}_{t-1}, \mathbf{z}_{t-1}, \boldsymbol{\epsilon}_{t,z}), \quad & \boldsymbol{\epsilon}_{t,z} &\sim \mathcal{N}(0, 1), \\ \mathbf{x}_t &= g_x(\mathbf{z}_t, \boldsymbol{\epsilon}_{t,x}), & \boldsymbol{\epsilon}_{t,x} &\sim \mathcal{N}(0, 1), \end{aligned}$$

where g_z and g_x are both 1-layer multilayer perceptions with ReLU activation and layer normalization. We place standard normal priors over all weights and biases. See Fig. 3a.

If the injected noise $\boldsymbol{\epsilon}_{t,z}$ combines linearly with the output of g_z, the induced distribution $p(\mathbf{z}_t \mid \mathbf{x}_{t-1}, \mathbf{z}_{t-1})$ is Gaussian parameterized by that output. This defines a stochastic RNN [2, 15], which generalizes its deterministic connection. With nonlinear combinations, the implicit density is more flexible (and intractable), making previous methods for inference not applicable. In our method, we perform variational inference and specify q to be implicit; we use the same architecture as the probability model's implicit priors.

We follow the same setup and hyperparameters as Kusner and Hernández-Lobato [29] and generate simple one-variable arithmetic sequences following a context free grammar,

$$S \to x \| S + S \| S - S \| S * S \| S / S,$$

where $\|$ divides possible productions of the grammar. We concatenate the inputs and point estimate the global variables (model parameters) using variational EM. Fig. 3b displays samples from the inferred model, training on sequences with a maximum of 15 symbols. It achieves sequences which roughly follow the context free grammar.

5 Discussion

We developed a class of hierarchical implicit models and likelihood-free variational inference, merging the idea of implicit densities with hierarchical Bayesian modeling and approximate posterior inference. This expands Bayesian analysis with the ability to apply neural samplers, physical simulators, and their combination with rich, interpretable latent structure.

More stable inference with ratio estimation is an open challenge. This is especially important when we analyze large-scale real world applications of implicit models. Recent work for genomics offers a promising solution [53].

Acknowledgements. We thank Balaji Lakshminarayanan for discussions which helped motivate this work. We also thank Christian Naesseth, Jaan Altosaar, and Adji Dieng for their feedback and comments. DT is supported by a Google Ph.D. Fellowship in Machine Learning and an Adobe Research Fellowship. This work is also supported by NSF IIS-0745520, IIS-1247664, IIS-1009542, ONR N00014-11-1-0651, DARPA FA8750-14-2-0009, N66001-15-C-4032, Facebook, Adobe, Amazon, and the John Templeton Foundation.

References

[1] Anelli, G., Antchev, G., Aspell, P., Avati, V., Bagliesi, M., Berardi, V., Berretti, M., Boccone, V., Bottigli, U., Bozzo, M., et al. (2008). The totem experiment at the CERN large Hadron collider. *Journal of Instrumentation*, 3(08):S08007.

[2] Bayer, J. and Osendorfer, C. (2014). Learning stochastic recurrent networks. *arXiv preprint arXiv:1411.7610*.

[3] Beaumont, M. A. (2010). Approximate Bayesian computation in evolution and ecology. *Annual Review of Ecology, Evolution and Systematics*, 41(379-406):1.

[4] Blei, D. M., Ng, A. Y., and Jordan, M. I. (2003). Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3(Jan):993–1022.

[5] Blundell, C., Cornebise, J., Kavukcuoglu, K., and Wierstra, D. (2015). Weight uncertainty in neural network. In *International Conference on Machine Learning*.

[6] Bottou, L. (2010). Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010*, pages 177–186. Springer.

[7] Chen, X., Duan, Y., Houthooft, R., Schulman, J., Sutskever, I., and Abbeel, P. (2016). InfoGAN: Interpretable representation learning by information maximizing generative adversarial nets. In *Neural Information Processing Systems*.

[8] Cranmer, K., Pavez, J., and Louppe, G. (2015). Approximating likelihood ratios with calibrated discriminative classifiers. *arXiv preprint arXiv:1506.02169*.

[9] Diaconis, P., Holmes, S., and Montgomery, R. (2007). Dynamical bias in the coin toss. *SIAM*, 49(2):211–235.

[10] Dieng, A. B., Tran, D., Ranganath, R., Paisley, J., and Blei, D. M. (2017). The χ-Divergence for Approximate Inference. In *Neural Information Processing Systems*.

[11] Diggle, P. J. and Gratton, R. J. (1984). Monte Carlo methods of inference for implicit statistical models. *Journal of the Royal Statistical Society: Series B (Methodological)*, pages 193–227.

[12] Donahue, J., Krähenbühl, P., and Darrell, T. (2017). Adversarial feature learning. In *International Conference on Learning Representations*.

[13] Dumoulin, V., Belghazi, I., Poole, B., Lamb, A., Arjovsky, M., Mastropietro, O., and Courville, A. (2017). Adversarially learned inference. In *International Conference on Learning Representations*.

[14] Dziugaite, G. K., Roy, D. M., and Ghahramani, Z. (2015). Training generative neural networks via maximum mean discrepancy optimization. In *Uncertainty in Artificial Intelligence*.

[15] Fraccaro, M., Sønderby, S. K., Paquet, U., and Winther, O. (2016). Sequential neural models with stochastic layers. In *Neural Information Processing Systems*.

[16] Gelman, A., Carlin, J. B., Stern, H. S., Dunson, D. B., Vehtari, A., and Rubin, D. B. (2013). *Bayesian data analysis*. Texts in Statistical Science Series. CRC Press, Boca Raton, FL.

[17] Gelman, A. and Hill, J. (2006). *Data analysis using regression and multilevel/hierarchical models*. Cambridge University Press.

[18] Gneiting, T. and Raftery, A. E. (2007). Strictly proper scoring rules, prediction, and estimation. *Journal of the American Statistical Association*, 102(477):359–378.

[19] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014). Generative adversarial nets. In *Neural Information Processing Systems*.

[20] Goodfellow, I. J. (2014). On distinguishability criteria for estimating generative models. In *ICLR Workshop*.

[21] Gutmann, M. U., Dutta, R., Kaski, S., and Corander, J. (2014). Statistical Inference of Intractable Generative Models via Classification. *arXiv preprint arXiv:1407.4981*.

[22] Hartig, F., Calabrese, J. M., Reineking, B., Wiegand, T., and Huth, A. (2011). Statistical inference for stochastic simulation models–theory and application. *Ecology Letters*, 14(8):816–827.

[23] Hernández-Lobato, J. M., Li, Y., Rowland, M., Hernández-Lobato, D., Bui, T., and Turner, R. E. (2016). Black-box α-divergence minimization. In *International Conference on Machine Learning*.

[24] Hoffman, M. D., Blei, D. M., Wang, C., and Paisley, J. W. (2013). Stochastic variational inference. *Journal of Machine Learning Research*, 14(1):1303–1347.

[25] Jordan, M. I., Ghahramani, Z., Jaakkola, T. S., and Saul, L. K. (1999). An introduction to variational methods for graphical models. *Machine Learning*.

[26] Karaletsos, T. (2016). Adversarial message passing for graphical models. In *NIPS Workshop*.

[27] Keller, J. B. (1986). The probability of heads. *The American Mathematical Monthly*, 93(3):191–197.

[28] Kingma, D. P. and Welling, M. (2014). Auto-Encoding Variational Bayes. In *International Conference on Learning Representations*.

[29] Kusner, M. J. and Hernández-Lobato, J. M. (2016). GANs for sequences of discrete elements with the Gumbel-Softmax distribution. In *NIPS Workshop*.

[30] Larsen, A. B. L., Sønderby, S. K., Larochelle, H., and Winther, O. (2016). Autoencoding beyond pixels using a learned similarity metric. In *International Conference on Machine Learning*.

[31] Li, Y. and Turner, R. E. (2016). Rényi Divergence Variational Inference. In *Neural Information Processing Systems*.

[32] Liu, Q. and Feng, Y. (2016). Two methods for wild variational inference. *arXiv preprint arXiv:1612.00081*.

[33] MacKay, D. J. C. (1992). *Bayesian methods for adaptive models*. PhD thesis, California Institute of Technology.

[34] Makhzani, A., Shlens, J., Jaitly, N., and Goodfellow, I. (2015). Adversarial autoencoders. *arXiv preprint arXiv:1511.05644*.

[35] Marin, J.-M., Pudlo, P., Robert, C. P., and Ryder, R. J. (2012). Approximate Bayesian computational methods. *Statistics and Computing*, 22(6):1167–1180.

[36] Mescheder, L., Nowozin, S., and Geiger, A. (2017). Adversarial variational Bayes: Unifying variational autoencoders and generative adversarial networks. *arXiv preprint arXiv:1701.04722*.

[37] Mohamed, S. and Lakshminarayanan, B. (2016). Learning in implicit generative models. *arXiv preprint arXiv:1610.03483*.

[38] Murphy, K. (2012). *Machine Learning: A Probabilistic Perspective*. MIT Press.

[39] Neal, R. M. (1994). *Bayesian Learning for Neural Networks*. PhD thesis, University of Toronto.

[40] Papamakarios, G. and Murray, I. (2016). Fast ϵ-free inference of simulation models with Bayesian conditional density estimation. In *Neural Information Processing Systems*.

[41] Pearl, J. (2000). *Causality*. Cambridge University Press.

[42] Pritchard, J. K., Seielstad, M. T., Perez-Lezaun, A., and Feldman, M. W. (1999). Population growth of human Y chromosomes: a study of Y chromosome microsatellites. *Molecular Biology and Evolution*, 16(12):1791–1798.

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

[44] Ranganath, R., Altosaar, J., Tran, D., and Blei, D. M. (2016a). Operator variational inference. In *Neural Information Processing Systems*.

[45] Ranganath, R., Gerrish, S., and Blei, D. M. (2014). Black box variational inference. In *Artificial Intelligence and Statistics*.

[46] Ranganath, R., Tran, D., and Blei, D. M. (2016b). Hierarchical variational models. In *International Conference on Machine Learning*.

[47] Rezende, D. J. and Mohamed, S. (2015). Variational inference with normalizing flows. In *International Conference on Machine Learning*.

[48] Rezende, D. J., Mohamed, S., and Wierstra, D. (2014). Stochastic backpropagation and approximate inference in deep generative models. In *International Conference on Machine Learning*.

[49] Salakhutdinov, R. and Mnih, A. (2008). Bayesian probabilistic matrix factorization using Markov chain Monte Carlo. In *International Conference on Machine Learning*, pages 880–887. ACM.

[50] Salimans, T., Kingma, D. P., and Welling, M. (2015). Markov chain Monte Carlo and variational inference: Bridging the gap. In *International Conference on Machine Learning*.

[51] Sugiyama, M., Suzuki, T., and Kanamori, T. (2012). Density-ratio matching under the Bregman divergence: A unified framework of density-ratio estimation. *Annals of the Institute of Statistical Mathematics*.

[52] Teh, Y. W. and Jordan, M. I. (2010). Hierarchical Bayesian nonparametric models with applications. *Bayesian Nonparametrics*, 1.

[53] Tran, D. and Blei, D. M. (2017). Implicit causal models for genome-wide association studies. *arXiv preprint arXiv:1710.10742*.

[54] Tran, D., Blei, D. M., and Airoldi, E. M. (2015). Copula variational inference. In *Neural Information Processing Systems*.

[55] Tran, D., Kucukelbir, A., Dieng, A. B., Rudolph, M., Liang, D., and Blei, D. M. (2016). Edward: A library for probabilistic modeling, inference, and criticism. *arXiv preprint arXiv:1610.09787*.

[56] Uehara, M., Sato, I., Suzuki, M., Nakayama, K., and Matsuo, Y. (2016). Generative adversarial nets from a density ratio estimation perspective. *arXiv preprint arXiv:1610.02920*.

[57] Wilkinson, D. J. (2011). *Stochastic modelling for systems biology*. CRC press.

A Noise versus Latent Variables

HIMs have two sources of randomness for each data point: the latent variable $\mathbf{z}_n$ and the noise $\boldsymbol{\epsilon}_n$; these sources of randomness get transformed to produce $\mathbf{x}_n$. Bayesian analysis infers posteriors on latent variables. A natural question is whether one should also infer the posterior of the noise.

The posterior's shape—and ultimately if it is meaningful—is determined by the dimensionality of noise and the transformation. For example, consider the GAN model, which has no local latent variable, $\mathbf{x}_n = g(\boldsymbol{\epsilon}_n; \boldsymbol{\theta})$. The conditional $p(\mathbf{x}_n \mid \boldsymbol{\epsilon}_n)$ is a point mass, fully determined by $\boldsymbol{\epsilon}_n$. When $g(\cdot; \boldsymbol{\theta})$ is injective, the posterior $p(\boldsymbol{\epsilon}_n \mid \mathbf{x}_n)$ is also a point mass,

$$p(\boldsymbol{\epsilon}_n \mid \mathbf{x}_n) = \mathbb{I}[\boldsymbol{\epsilon}_n = g^{-1}(\mathbf{x}_n)],$$

where g^{-1} is the left inverse of g.This means for injective functions of the randomness (both noise and latent variables), the "posterior" may be worth analysis as a deterministic hidden representation [12], but it is not random.

The point mass posterior can be found via nonlinear least squares. Nonlinear least squares yields the iterative algorithm

$$\hat{\boldsymbol{\epsilon}}_n = \hat{\boldsymbol{\epsilon}}_n - \rho_t \nabla_{\hat{\boldsymbol{\epsilon}}_n} f(\hat{\boldsymbol{\epsilon}}_n)^\top (f(\hat{\boldsymbol{\epsilon}}_n) - \mathbf{x}_n),$$

for some step size sequence ρ_t. Note the updates will get stuck when the gradient of f is zero. However, the injective property of f allows the iteration to be checked for correctness (simply check if $f(\hat{\boldsymbol{\epsilon}}_n) = \mathbf{x}_n$).

B Implicit Model Examples in Edward

We demonstrate implicit models via example implementations in Edward [55].

Fig. 4 implements a 2-layer deep implicit model. It uses `tf.layers` to define neural networks: `tf.layers.dense(x, 256)` applies a fully connected layer with 256 hidden units and input x; weight and bias parameters are abstracted from the user. The program generates N data points $\mathbf{x}_n \in \mathbb{R}^{10}$ using two layers of implicit latent variables $\mathbf{z}_{n,1}, \mathbf{z}_{n,2} \in \mathbb{R}^d$ and with an implicit likelihood.

Fig. 5 implements a Bayesian GAN for classification. It manually defines a 2-layer neural network, where for each data index, it takes features $\mathbf{x}_n \in \mathbb{R}^{500}$ concatenated with noise $\boldsymbol{\epsilon}_n \in \mathbb{R}$ as input. The output is a label $\mathbf{y}_n \in \{-1, 1\}$, given by the sign of the last layer. We place a standard normal prior over all weights and biases. Running this program while feeding the placeholder $\mathbf{X} \in \mathbb{R}^{N \times 500}$ generates a vector of labels $\mathbf{y} \in \{-1, 1\}^N$.

```
import tensorflow as tf
from edward.models import Normal

# random noise is Normal(0, 1)
eps2 = Normal(tf.zeros([N, d]), tf.ones([N, d]))
eps1 = Normal(tf.zeros([N, d]), tf.ones([N, d]))
eps0 = Normal(tf.zeros([N, d]), tf.ones([N, d]))

# alternate latent layers z with hidden layers h
z2 = tf.layers.dense(eps2, 128, activation=tf.nn.relu)
h2 = tf.layers.dense(z2, 128, activation=tf.nn.relu)
z1 = tf.layers.dense(tf.concat([eps1, h2], 1), 128, activation=tf.nn.relu)
h1 = tf.layers.dense(z1, 128, activation=tf.nn.relu)
x  = tf.layers.dense(tf.concat([eps0, h1], 1), 10, activation=None)
```

Figure 4: Two-layer deep implicit model for data points $\mathbf{x}_n \in \mathbb{R}^{10}$. The architecture alternates with stochastic and deterministic layers. To define a stochastic layer, we simply inject noise by concatenating it into the input of a neural net layer.

```
import tensorflow as tf
from edward.models import Normal

# weights and biases have Normal(0, 1) prior
W1 = Normal(tf.zeros([501, 256]), tf.ones([501, 256]))
W2 = Normal(tf.zeros([256, 1]), tf.ones([256, 1]))
b1 = Normal(tf.zeros(256), tf.ones(256))
b2 = Normal(tf.zeros(1), tf.ones(1))

# set up inputs to neural network
X = tf.placeholder(tf.float32, [N, 500])
eps = Normal(tf.zeros([N, 1]), tf.ones([N, 1]))

# y = neural_network([x, eps])
input = tf.concat([X, eps], 1)
h1 = tf.nn.relu(tf.matmul(input, W1) + b1)
h2 = tf.matmul(h1, W2) + b2
y = tf.reshape(tf.sign(h2), [-1])  # take sign, then flatten
```

Figure 5: Bayesian GAN for classification, taking $\mathbf{X} \in \mathbb{R}^{N \times 500}$ as input and generating a vector of labels $\mathbf{y} \in \{-1, 1\}^N$. The neural network directly generates the data rather than parameterizing a probability distribution.

C KL Uniqueness

An integral probability metric measures distance between two distributions p and q,

$$d(p, q) = \sup_{f \in \mathcal{F}} |\mathbb{E}_p f - \mathbb{E}_q f|.$$

Integral probability metrics have been used for parameter estimation in generative models [14] and for variational inference in models with tractable density [46]. In contrast to models with only local latent variables, to infer the posterior, we need an integral probability metric between it and the variational approximation. The direct approach fails because sampling from the posterior is intractable.

An indirect approach requires constructing a sufficiently broad class of functions with posterior expectation zero based on Stein's method [46]. These constructions require a likelihood function and its gradient. Working around the likelihood would require a form of nonparametric density estimation; unlike ratio estimation, we are unaware of a solution that sufficiently scales to high dimensions.

As another class of divergences, the f divergence is

$$d(p, q) = \mathbb{E}_q\Big[f\left(\frac{p}{q}\right)\Big].$$

Unlike integral probability metrics, f divergences are naturally conducive to ratio estimation, enabling implicit p and implicit q. However, the challenge lies in scalable computation. To subsample data in hierarchical models, we need f to satisfy up to constants $f(ab) = f(a) + f(b)$, so that the expectation becomes a sum over individual data points. For continuous functions, this is a defining property of the $\log$ function. This implies the KL-divergence from q to p is the only f divergence where the subsampling technique in our desiderata is possible.

D Hinge Loss

Let $r(\mathbf{x}_i, \mathbf{z}_i, \boldsymbol{\beta}; \theta)$ output a real value, as with the log loss in Section 4. The hinge loss is

$$\begin{aligned}\mathcal{D}_{\text{hinge}} = &\mathbb{E}_{p(\mathbf{x}_n, \mathbf{z}_n \mid \boldsymbol{\beta})}[\max(0, 1 - r(\mathbf{x}_n, \mathbf{z}_n, \boldsymbol{\beta}; \boldsymbol{\theta}))] + \\ &\mathbb{E}_{q(\mathbf{x}_n, \mathbf{z}_n \mid \boldsymbol{\beta})}[\max(0, 1 + r(\mathbf{x}_n, \mathbf{z}_n, \boldsymbol{\beta}; \boldsymbol{\theta}))].\end{aligned}$$

We minimize this loss function by following unbiased gradients. The gradients are calculated analogously as for the log loss. The optimal r^* is the log ratio.

E Comparing Bayesian GANs with MAP to GANs with MLE

In Section 4, we argued that MAP estimation with a Bayesian GAN enables analysis over discrete data, but GANs—even with a maximum likelihood objective [20]—cannot. This is a surprising result: assuming a flat prior for MAP, the two are ultimately optimizing the same objective. We compare the two below.

For GANs, assume the discriminator outputs a logit probability, so that it's unconstrained instead of on $[0, 1]$. GANs with MLE use the discriminative problem

$$\max_{\boldsymbol{\theta}} \mathbb{E}_{q(\mathbf{x})}[\log \sigma(D(\mathbf{x}; \boldsymbol{\theta}))] + \mathbb{E}_{p(\mathbf{x}; \mathbf{w})}[\log(1 - \sigma(D(\mathbf{x}; \boldsymbol{\theta})))].$$

They use the generative problem

$$\min_{\mathbf{w}} \mathbb{E}_{p(\mathbf{x}; \mathbf{w})}[-\exp(D(\mathbf{x}))].$$

Solving the generative problem with reparameterization gradients requires backpropagating through data generated from the model, $\mathbf{x} \sim p(\mathbf{x}; \mathbf{w})$. This is not possible for discrete $\mathbf{x}$. Further, the exponentiation also makes this objective numerically unstable and thus unusable in practice.

Contrast this with Bayesian GANs with MLE (MAP and a flat prior). This applies a point mass variational approximation $q(\mathbf{w}') = \mathbb{I}[\mathbf{w}' = \mathbf{w}]$. It maximizes the ELBO,

$$\max_{\mathbf{w}} \mathbb{E}_{q(\mathbf{w})}[\log p(\mathbf{w}) - \log q(\mathbf{w})] + \sum_{n=1}^{N} r(\mathbf{x}_n, \mathbf{w}).$$

Figure 6: (left) Difference of ratios over steps of q. Low variance on y-axis means more stable. Interestingly, the ratio estimator is more accurate and stable as q converges to the posterior. **(middle)** Difference of ratios over steps of r; q is fixed at random initialization. The ratio estimator doesn't improve even after many steps. **(right)** Difference of ratios over steps of r; q is fixed at the posterior. The ratio estimator only requires few steps from random initialization to be highly accurate.

The first term is zero for a flat prior $p(\mathbf{w}) \propto 1$ and point mass approximation; the problem reduces to

$$\max_{\mathbf{w}} \sum_{n=1}^{N} r(\mathbf{x}_n, \mathbf{w}).$$

Solving this is possible for discrete $\mathbf{x}$: it only requires backpropagating gradients through $r(\mathbf{x}, \mathbf{w})$ with respect to $\mathbf{w}$, all of which is differentiable. Further, the objective does not require a numerically unstable exponentiation.

Ultimately, the difference lies in the role of the ratio estimators. Recall for Bayesian GANs, we use the ratio estimation problem

$$\mathcal{D}_{\log} = \mathbb{E}_{p(\mathbf{x};\mathbf{w})}[-\log \sigma(r(\mathbf{x}, \mathbf{w}; \boldsymbol{\theta}))] + \mathbb{E}_{q(\mathbf{x})}[-\log(1 - \sigma(r(\mathbf{x}, \mathbf{w}; \boldsymbol{\theta})))].$$

The optimal ratio estimator is the log-ratio $r^*(\mathbf{x}, \mathbf{w}) = \log p(\mathbf{x} \mid \mathbf{w}) - \log q(\mathbf{x})$. Optimizing it with respect to $\mathbf{w}$ reduces to optimizing the log-likelihood $\log p(\mathbf{x} \mid \mathbf{w})$. The optimal discriminator for GANs with MLE has the same ratio, $D^*(\mathbf{x}) = \log p(\mathbf{x}; \mathbf{w}) - \log q(\mathbf{x})$; however, it is a constant function with respect to $\mathbf{w}$. Hence one cannot immediately substitute $D^*(\mathbf{x})$ as a proxy to optimizing the likelihood. An alternative is to use importance sampling; the result is the former objective [20].

F Stability of Ratio Estimator

With implicit models, the difference from standard KL variational inference lies in the ratio estimation problem. Thus we would like to assess the accuracy of the ratio estimator. We can check this by comparing to the true ratio under a model with tractable likelihood.

We apply Bayesian linear regression. It features a tractable posterior which we leverage in our analysis. We use 50 simulated data points $\{\mathbf{y}_n \in \mathbb{R}^2, \mathbf{x}_n \in \mathbb{R}\}$. The optimal (log) ratio is

$$r^*(\mathbf{x}, \boldsymbol{\beta}) = \log p(\mathbf{x} \mid \boldsymbol{\beta}) - \log q(\mathbf{x}).$$

Note the log-likelihood $\log p(\mathbf{x} \mid \boldsymbol{\beta})$ minus $r^*(\mathbf{x}, \boldsymbol{\beta})$ is equal to the empirical distribution $\sum_n \log q(\mathbf{x}_n)$, a constant. Therefore if a ratio estimator r is accurate, its difference with $\log p(\mathbf{x} \mid \boldsymbol{\beta})$ should be a constant with low variance across values of $\boldsymbol{\beta}$.

See Fig. 6. The top graph displays the estimate of $\log q(\mathbf{x})$ over updates of the variational approximation $q(\boldsymbol{\beta})$; each estimate uses a sample from the current $q(\boldsymbol{\beta})$. The ratio estimator r is more accurate as q exactly converges to the posterior. This matches our intuition: if data generated from the model is close to the true data, then the ratio is more stable to estimate.

An alternative hypothesis for Fig. 6 is that the ratio estimator has simply accumulated information during training. This turns out to be untrue; see the bottom graphs. On the left, q is fixed at a random initialization; the estimate of $\log q(\mathbf{x})$ is displayed over updates of r. After many updates, r still produces unstable estimates. In contrast, the right shows the same procedure with q fixed at the posterior. r is accurate after few updates.

Several practical insights appear for training. First, it is not helpful to update r multiple times before updating q (at least in initial iterations). Additionally, if the specified model poorly matches the data, training will be difficult across all iterations.

The property that ratio estimation is more accurate as the variational approximation improves is because $q(\mathbf{x}_n)$ is set to be the empirical distribution. (Note we could subtract any density $q(\mathbf{x}_n)$ from the ELBO in Equation 4.) Likelihood-free variational inference finds $q(\boldsymbol{\beta})$ that makes the observed data likely under $p(\mathbf{x}_n \mid \boldsymbol{\beta})$, i.e., $p(\mathbf{x}_n \mid \boldsymbol{\beta})$ gets closer to the empirical distribution at values sampled from $q(\boldsymbol{\beta})$. Letting $q(\mathbf{x}_n)$ be the empirical distribution means the ratio estimation problem will be less trivially solvable (thus more accurate) as $q(\boldsymbol{\beta})$ improves.

Note also this motivates why we do not subsume inference of $p(\boldsymbol{\beta} \mid \mathbf{x})$ in the ratio in order to enable implicit global variables and implicit global variational approximations. Namely, estimation requires comparing samples between the prior and the posterior; they rarely overlap for global variables.