Variational Implicit Processes

Chao Ma [1] **Yingzhen Li** [2] **José Miguel Hernández-Lobato** [1 2]

Abstract

We introduce the implicit processes (IPs), a stochastic process that places implicitly defined multivariate distributions over any finite collections of random variables. IPs are therefore highly flexible implicit priors over *functions*, with examples including data simulators, Bayesian neural networks and non-linear transformations of stochastic processes. A novel and efficient approximate inference algorithm for IPs, namely the variational implicit processes (VIPs), is derived using generalised wake-sleep updates. This method returns simple update equations and allows scalable hyper-parameter learning with stochastic optimization. Experiments show that VIPs return better uncertainty estimates and lower errors over existing inference methods for challenging models such as Bayesian neural networks, and Gaussian processes.

1 Introduction

Probabilistic models with *implicit distributions* as core components have recently attracted enormous interest in both deep learning and the approximate Bayesian inference communities. In contrast to *prescribed probabilistic models* (Diggle & Gratton, 1984) that assign *explicit* densities to possible outcomes of the model, implicit models *implicitly assign* probability measures by the specification of the *data generating process*. One of the most well known implicit distributions is the generator of generative adversarial nets (GANs) (Goodfellow et al., 2014; Arjovsky et al., 2017) that transforms isotropic noise into high dimensional data, using neural networks. In approximate inference context, implicit distributions have also been used as flexible approximate posterior distributions (Rezende & Mohamed, 2015; Liu & Feng, 2016; Tran et al., 2017; Li et al., 2017).

[1]Department of Engineering, University of Cambridge, Cambridge, UK [2]Microsoft Research Cambridge, Cambridge, UK. Correspondence to: Chao Ma <cm905@cam.ac.uk>, José Miguel Hernández-Lobato <jmh233@cam.ac.uk>.

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

This paper explores the extension of implicit models to Bayesian modeling of *random functions*. Similar to the construction of Gaussian processes (GPs), an *implicit process* (IP) assigns implicit distributions over any finite collections of random variables. Therefore IPs can be much more flexible than GPs when complicated models like neural networks are used for the implicit distributions. With an IP as the prior, we can directly perform (variational) posterior inference *over functions* in a non-parametric fashion. This is beneficial for better-calibrated uncertainty estimates like GPs (Bui et al., 2016a). It also avoids typical issues of inference in parameter space, that is, symmetric modes in the posterior distribution of Bayesian neural network *weights*. The function-space inference for IPs is achieved by our proposed *variational implicit process* (VIP) algorithm, which addresses the intractability issues of implicit distributions.

Concretely, our contributions are threefold:

- We formalize implicit stochastic process priors over *functions*, and prove its well-definedness in both finite and infinite dimensional cases. By allowing the usage of IPs with rich structures as priors (e.g., data simulators and Bayesian LSTMs), our approach provides a unified and powerful Bayesian inference framework for these important but challenging deep models.
- We derive a novel and efficient variational inference framework that gives a closed-form approximation to the IP posterior. It does not rely on e.g. density ratio/gradient estimators in implicit variational inference literature which can be inaccurate in high dimensions. Our inference method is computationally cheap, and it allows scalable hyper-parameter learning in IPs.
- We conduct extensive comparisons between IPs trained with the proposed inference method, and GPs/BNNs/Bayesian LSTMs trained with existing variational approaches. Our method consistently outperforms other methods, and achieves state-of-the-art results on a large scale Bayesian LSTM inference task.

2 Implicit Stochastic Processes

In this section, we generalize GPs to implicit stochastic processes. Readers are referred to appendix A for a detailed introduction, but briefly speaking, a GP defines the distribution of a random function f by placing a multivariate Gaussian distribution $\mathcal{N}(\mathbf{f}; \mathbf{m}, \mathbf{K_{ff}})$ over any finite collection of function values $\mathbf{f} = (f(\mathbf{x}_1), ..., f(\mathbf{x}_N))^\top$ evaluated at any given finite collection of input locations $\mathbf{X} = \{\mathbf{x}_n\}_{n=1}^N$. Here $(\mathbf{m})_n = m(\mathbf{x}_n)$ and $(\mathbf{K_{ff}})_{n,n'} = \mathcal{K}(\mathbf{x}_n, \mathbf{x}_{n'})$, and following Kolmogorov consistency theorem (Itô, 1984), the mean and covariance functions $m(\cdot), \mathcal{K}(\cdot, \cdot)$ are shared across all such finite collections. An alternative parameterization of GPs defines the sampling process as $\mathbf{f} \sim \mathcal{N}(\mathbf{f}; \mathbf{m}, \mathbf{K_{ff}}) \Leftrightarrow \mathbf{z} \sim \mathcal{N}(\mathbf{z}; 0, \mathbf{I})$, $\mathbf{f} = \mathbf{Bz} + \mathbf{m}$, with $\mathbf{K_{ff}} = \mathbf{BB}^\top$ the Cholesky decomposition of the covariance matrix. Observing this, we propose a generalization of the generative process by replacing the linear transform of the latent variable $\mathbf{z}$ with a nonlinear one. This gives the following formal definition of implicit stochastic process.

Definition 1 (noiseless implicit stochastic processes)**.** *An implicit stochastic process (IP) is a collection of random variables $f(\cdot)$, such that any finite collection $\mathbf{f} = (f(\mathbf{x}_1), ..., f(\mathbf{x}_N))^\top$ has joint distribution implicitly defined by the following generative process:*

$$\mathbf{z} \sim p(\mathbf{z}), \quad f(\mathbf{x}_n) = g_\theta(\mathbf{x}_n, \mathbf{z}), \quad \forall\, \mathbf{x}_n \in \mathbf{X}. \tag{1}$$

A function distributed according to the above IP is denoted as $f(\cdot) \sim \mathcal{IP}(g_\theta(\cdot, \cdot), p_\mathbf{z})$.

Note that $\mathbf{z} \sim p(\mathbf{z})$ could be infinite dimensional (such as samples from a Gaussian Process). Definition 1 is validated by the following propositions.

Proposition 1 (Finite dimension case)**.** *Let $\mathbf{z}$ be a finite dimensional vector. Then there exists a unique stochastic process, such that any finite collection of random variables has distribution implicitly defined by (1).*

Proposition 2 (Infinite dimension case)**.** *Let $z(\cdot) \sim \mathcal{SP}(0, C)$ be a centered continuous stochastic process on $\mathcal{L}^2(\mathbb{R}^d)$ with covariance function $C(\cdot, \cdot)$. Then the operator $g(\mathbf{x}, z) = O_k(z)(\mathbf{x}) := h(\int_\mathbf{x} \sum_{l=0}^M K_l(\mathbf{x}, \mathbf{x}') z(\mathbf{x}') d\mathbf{x}')$, $0 < M < +\infty$ defines a stochastic process if $K_l \in \mathcal{L}^2(\mathbb{R}^d \times \mathbb{R}^d)$, h is a Borel measurable, bijective function in $\mathbb{R}$ and there exist $0 \leq A < +\infty$ such that $|h(x)| \leq A|x|$ for $\forall x \in \mathbb{R}$.*

Proposition 1 is proved in appendix C.1 using the Kolmogorov extension theorem. Proposition 2 considers *random functions* as the latent input $z(\cdot)$, and introduces a specific form of the transformation/operator g, so that the resulting collection of variables $f(\cdot)$ is still a valid stochastic process (see appendix C.2 for a proof). Note this operator can be recursively applied to build highly non-linear operators over functions (Guss, 2016; Williams, 1997; Stinchcombe, 1999; Le Roux & Bengio, 2007; Globerson & Livni, 2016). These two propositions indicate that IPs form a rich class of priors over functions. Indeed, we visualize some examples of IPs in Figure 1 with discussions as follows:

Example 1 (Data simulators)**.** *Simulators, e.g. physics engines and climate models, are omnipresent in science and engineering. These models encode laws of physics in $g_\theta(\cdot, \cdot)$, use $\mathbf{z} \sim p(\mathbf{z})$ to explain the remaining randomness, and evaluate the function at input locations $\mathbf{x}$: $f(\mathbf{x}) = g_\theta(\mathbf{x}, \mathbf{z})$. We define the **neural sampler** as a specific instance of this class. In this case $g_\theta(\cdot, \cdot)$ is a neural network with weights θ, i.e., $g_\theta(\cdot, \cdot) = \mathrm{NN}_\theta(\cdot, \cdot)$, and $p(\mathbf{z}) =$ Uniform$([-a, a]^d)$.*

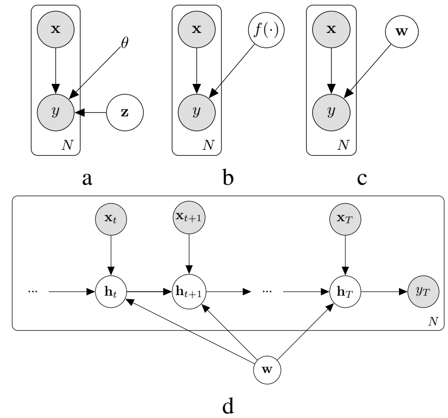

Figure 1: Examples of IPs: (a) Neural samplers; (b) Warped GPs (c) Bayesian neural networks; (d) Bayesian RNNs.

Example 2 (Warped Gaussian Processes)**.** *Warped Gaussian Processes (Snelson et al., 2004) is also an interesting example of IPs. Let $z(\cdot) \sim p(z)$ be a sample from a GP prior, and $g_\theta(\mathbf{x}, z)$ is defined as $g_\theta(\mathbf{x}, z) = h(z(\mathbf{x}))$, where $h(\cdot)$ is a one dimensional monotonic function.*

Example 3 (Bayesian neural network)**.** *In a Bayesian neural network, the synaptic weights W with prior $p(W)$ play the role of $\mathbf{z}$ in (1). A function is sampled by $W \sim p(W)$ and then setting $f(\mathbf{x}) = g_\theta(\mathbf{x}, W) = \mathrm{NN_W}(\mathbf{x})$ for all $\mathbf{x} \in \mathbf{X}$. In this case θ could be the hyper-parameters of the prior $p(W)$ to be tuned.*

Example 4 (Bayesian RNN)**.** *Similar to Example 3, a Bayesian recurrent neural network (RNN) can be defined by considering its weights as random variables, and taking as function evaluation an output value generated by the RNN after processing the last symbol of an input sequence.*

3 Variational Implicit Processes

Consider the following regression model with an IP prior over the regression function:

$$f(\cdot) \sim \mathcal{IP}(g_\theta(\cdot, \cdot), p_\mathbf{z}),\ y = f(\mathbf{x}) + \epsilon,\ \epsilon \sim \mathcal{N}(0, \sigma^2). \tag{2}$$

Equation (2) defines an implicit model $p(\mathbf{y}, \mathbf{f}|\mathbf{x})$, which is intractable in most cases. Note that it is common to add Gaussian noise ϵ to an implicit model, e.g. see the noise smoothing trick used in GANs (Sønderby et al., 2016; Salimans et al., 2016). Given an observed dataset $\mathcal{D} = \{\mathbf{X}, \mathbf{y}\}$ and a set of test inputs $\mathbf{X}_*$, Bayesian predictive inference computes the predictive distribution $p(\mathbf{y}_*|\mathbf{X}_*, \mathbf{X}, \mathbf{y}, \theta)$, which itself requires interpolating over posterior $p(f|\mathbf{X}, \mathbf{y}, \theta)$. Besides prediction, we also want to learn the model parameters θ and σ by maximizing the marginal likelihood: $\log p(\mathbf{y}|\mathbf{X}, \theta) = \log \int_{\mathbf{f}} p(\mathbf{y}|\mathbf{f})p(\mathbf{f}|\mathbf{X}, \theta)d\mathbf{f}$, with $\mathbf{f} = f(\mathbf{X})$ being the evaluation of f on the points in $\mathbf{X}$. Unfortunately, both the prior $p(\mathbf{f}|\mathbf{X}, \theta)$ and the posterior $p(f|\mathbf{X}, \mathbf{y}, \theta)$ are intractable as the implicit process does not allow point-wise density evaluation, let alone the marginalization tasks. Therefore, to address these, we must resort to approximate inference.

We propose a generalization of the *wake-sleep* algorithm (Hinton et al., 1995) to handle both intractabilities. This method returns (i) an approximate posterior distribution $q(f|\mathbf{X}, \mathbf{y})$ which is later used for predictive inference, and (ii) an approximation to the marginal likelihood $p(\mathbf{y}|\mathbf{X}, \theta)$ for hyper-parameter optimization. We use the posterior of a GP to approximate the posterior of the IP, i.e. $q(f|\mathbf{X}, \mathbf{y}) = q_{\mathcal{GP}}(f|\mathbf{X}, \mathbf{y})$, since GP is one of the few existing tractable distributions over functions. A high-level summary of our algorithm is the following:

- ***Sleep phase***: sample function values $\mathbf{f}$ and noisy outputs $\mathbf{y}$ as indicated in (2). This *dreamed* data is then used as the *maximum-likelihood (ML)* target to fit a GP. This is equivalent to minimizing $\mathrm{D_{KL}}[p(\mathbf{y}, \mathbf{f}|\mathbf{X}, \theta)||q_{\mathcal{GP}}(\mathbf{y}, \mathbf{f}|\mathbf{X})]$ for any possible $\mathbf{X}$.

- ***Wake phase***: The optimal GP posterior approximation $q_{\mathcal{GP}}(\mathbf{f}|\mathbf{X}, \mathbf{y})$ obtained in the sleep phase is used to construct a variational approximation to $\log p(\mathbf{y}|\mathbf{X}, \theta)$, which is then optimized with respect to θ.

Our approach has two key advantages. First, the algorithm has no explicit sleep phase computation, since the sleep phase optimization has an analytic solution that can be directly plugged into the wake-phase objective. Second, the proposed wake phase update is highly scalable, as it is equivalent to a Bayesian linear regression task with random features sampled from the implicit process. With our wake-sleep algorithm, the evaluation of the implicit prior density is no longer an obstacle for approximate inference. We call this inference framework the *variational implicit process* (VIP). In the following sections we give specific details on both the wake and sleep phases.

3.1 Sleep phase: GP posterior as variational distribution

This section proposes an approximation to the IP posterior $p(\mathbf{f}|\mathbf{X}, \mathbf{y}, \theta)$. The naive variational inference (Jordan et al., 1999) would require computing the joint distribution $p(\mathbf{y}, \mathbf{f}|\mathbf{X}, \theta)$ which is intractable. However, sampling from this joint distribution is straightforward. We leverage this idea in the *sleep phase* of our wake-sleep algorithm to approximate the joint distribution $p(\mathbf{y}, \mathbf{f}|\mathbf{X}, \theta)$ instead.

Precisely, for any finite collection of variables $\mathbf{f}$ with their input locations $\mathbf{X}$, we approximate $p(\mathbf{y}, \mathbf{f}|\mathbf{X}, \theta)$ with a simpler distribution $q(\mathbf{y}, \mathbf{f}|\mathbf{X}) = q(\mathbf{y}|\mathbf{f})q(\mathbf{f}|\mathbf{X})$ instead. We choose $q(\mathbf{f}|\mathbf{X})$ to be a GP with mean and covariance functions $m(\cdot)$ and $\mathcal{K}(\cdot, \cdot)$, respectively, and write the prior as $q(\mathbf{f}|\mathbf{X}) = q_{\mathcal{GP}}(\mathbf{f}|\mathbf{X}, m, \mathcal{K})$. The sleep-phase update minimizes the following KL divergence:

$$q^\star_{\mathcal{GP}} = \underset{m,\mathcal{K}}{\mathrm{argmin}}\,\mathcal{U}(m, \mathcal{K}), \tag{3}$$
$$\text{with} \quad \mathcal{U}(m, \mathcal{K}) = \mathrm{D_{KL}}[p(\mathbf{y}, \mathbf{f}|\mathbf{X}, \theta)||q_{\mathcal{GP}}(\mathbf{y}, \mathbf{f}|\mathbf{X}, m, \mathcal{K})].$$

We further assume $q(\mathbf{y}|\mathbf{f}) = p(\mathbf{y}|\mathbf{f})$, which reduces $\mathcal{U}(m, \mathcal{K})$ to $\mathrm{D}_{KL}[p(\mathbf{f}|\mathbf{X}, \theta)||q_{\mathcal{GP}}(\mathbf{f}|\mathbf{X}, m, \mathcal{K})]$. In this case the optimal $m(\cdot)$ and $\mathcal{K}(\cdot, \cdot)$ are equal to the mean and covariance functions of the IP, respectively:

$$m^\star(\mathbf{x}) = \mathbb{E}[f(\mathbf{x})], \tag{4}$$
$$\mathcal{K}^\star(\mathbf{x}_1, \mathbf{x}_2) = \mathbb{E}[(f(\mathbf{x}_1) - m^\star(\mathbf{x}_1))(f(\mathbf{x}_2) - m^\star(\mathbf{x}_2))].$$

Below we also write the optimal solution as $q^\star_{\mathcal{GP}}(\mathbf{f}|\mathbf{X}, \theta) = q_{\mathcal{GP}}(\mathbf{f}|\mathbf{X}, m^\star, \mathcal{K}^\star)$ to explicitly specify the *dependency on prior parameters* θ [1]. In practice, the mean and covariance functions are estimated by by Monte Carlo, which leads to *maximum likelihood* training (MLE) for the GP with *dreamed* data from the IP. Assume S functions are drawn from the IP: $f^\theta_s(\cdot) \sim \mathcal{IP}(g_\theta(\cdot, \cdot), p_{\mathbf{z}}), s = 1, \dots, S$. The optimum of $\mathcal{U}(m, \mathcal{K})$ is then estimated by the *MLE solution*:

$$m^\star_{\text{MLE}}(\mathbf{x}) = \frac{1}{S}\sum_s f^\theta_s(\mathbf{x}), \tag{5}$$

$$\mathcal{K}^\star_{\text{MLE}}(\mathbf{x}_1, \mathbf{x}_2) = \frac{1}{S}\sum_s \Delta_s(\mathbf{x}_1)\Delta_s(\mathbf{x}_2), \tag{6}$$
$$\Delta_s(\mathbf{x}) = f^\theta_s(\mathbf{x}) - m^\star_{\text{MLE}}(\mathbf{x}).$$

To reduce computational costs, the number of dreamed samples S is often small. Therefore, we perform *maximum a posteriori* instead of MLE, by putting an inverse Wishart process prior (Shah et al., 2014) $\mathcal{IWP}(\nu, \Psi)$ over the GP covariance function $\mathcal{K}$ (Appendix C.3).

The original sleep phase algorithm in (Hinton et al., 1995) also finds a posterior approximation by minimizing (4). However, the original approach would define the q distribution as $q(\mathbf{y}, \mathbf{f}|\mathbf{X}) = p(\mathbf{y}|\mathbf{X}, \theta)q_{\mathcal{GP}}(\mathbf{f}|\mathbf{y}, \mathbf{X})$, which builds a *recognition model* that can be directly transfered for later inference. By contrast, we define $q(\mathbf{y}, \mathbf{f}|\mathbf{X}) = p(\mathbf{y}|\mathbf{f})q_{\mathcal{GP}}(\mathbf{f}|\mathbf{X})$, which corresponds to an approximation of the IP prior. In other words, we approximate an intractable

[1] This allows us to compute gradients w.r.t. θ through $m^\star$ and $\mathcal{K}^\star$ using reparameterization trick (by definition of IP, $f(\mathbf{x}) = g_\theta(\mathbf{x}, \mathbf{z})$), during the wake phase in Section 3.2.

generative model using another generative model with a GP prior and later, the resulting GP posterior $q^\star_{\mathcal{GP}}(\mathbf{f}|\mathbf{X},\mathbf{y})$ is employed as the variational distribution. Importantly, we never explicitly perform the sleep phase updates, that is, the optimization of $\mathcal{U}(m,\mathcal{K})$, as there is an analytic solution readily available, which can potentially save a significant amount of computation.

Another interesting observation is that the sleep phase's objective $\mathcal{U}(m,\mathcal{K})$ also provides an upper-bound to the KL divergence between the posterior distributions,

$$\mathcal{J} = \mathrm{D}_{\mathrm{KL}}[p(\mathbf{f}|\mathbf{X},\mathbf{y},\theta)||q_{\mathcal{GP}}(\mathbf{f}|\mathbf{X},\mathbf{y})].$$

One can show that $\mathcal{U}$ is an upper-bound of $\mathcal{J}$ according to the non-negativity and chain rule of the KL divergence:

$$\mathcal{U}(m,\mathcal{K}) = \mathcal{J} + \mathrm{D}_{\mathrm{KL}}[p(\mathbf{y}|\mathbf{X},\theta)||q_{\mathcal{GP}}(\mathbf{y}|\mathbf{X})] \geq \mathcal{J}. \quad (7)$$

Therefore, $\mathcal{J}$ is also decreased when the mean and covariance functions are optimized during the sleep phase. This bounding property justifies $\mathcal{U}(m,\mathcal{K})$ as a appropriate variational objective for posterior approximation.

3.2 Wake phase: a scalable approach to learning the model parameters θ

In the wake phase of the original wake-sleep algorithm, the IP model parameters θ are optimized by maximizing a variational lower-bound on the log marginal likelihood $\log p(\mathbf{y}|\mathbf{X},\theta)$. Unfortunately, this requires evaluating the IP prior $p(\mathbf{f}|\mathbf{X},\theta)$ which is intractable. But recall from (7) that during the sleep phase $\mathrm{D}_{\mathrm{KL}}[p(\mathbf{y}|\mathbf{X},\theta)||q_{\mathcal{GP}}(\mathbf{y}|\mathbf{X})]$ is also minimized. Therefore we directly approximate the log marginal likelihood using the *optimal* GP from the sleep phase, i.e.

$$\log p(\mathbf{y}|\mathbf{X},\theta) \approx \log q^\star_{\mathcal{GP}}(\mathbf{y}|\mathbf{X},\theta). \quad (8)$$

This again demonstrates the key advantage of the proposed sleep phase update via generative model matching. Also it is a sensible objective for predictive inference as the GP returned by wake-sleep will be used for making predictions.

Similar to GP regression, optimizing $\log q^\star_{\mathcal{GP}}(\mathbf{y}|\mathbf{X},\theta)$ can be computationally expensive for large datasets. Therefore sparse GP approximation techniques (Snelson & Ghahramani, 2006; Titsias, 2009; Hensman et al., 2013; Bui et al., 2016b) are applicable, but we leave them to future work and consider an alternative approach that is related to random feature approximations of GPs (Rahimi & Recht, 2008; Gal & Turner, 2015; Gal & Ghahramani, 2016a; Balog et al., 2016; Lázaro-Gredilla et al., 2010).

Note that $\log q^\star_{\mathcal{GP}}(\mathbf{y}|\mathbf{X},\theta)$ can be approximated by the log marginal likelihood of a Bayesian linear regression model with S randomly sampled dreamed functions, and a coefficient vector $\mathbf{a} = (a_1,...,a_S)$:

$$\log q^\star_{\mathcal{GP}}(\mathbf{y}|\mathbf{X},\theta) \approx \log \int \prod_n q^\star(y_n|\mathbf{x}_n,\mathbf{a},\theta)p(\mathbf{a})d\mathbf{a}, \quad (9)$$

$$\begin{aligned} q^\star(y_n|\mathbf{x}_n,\mathbf{a},\theta) &= \mathcal{N}\left(y_n;\mu(\mathbf{x}_n,\mathbf{a},\theta),\sigma^2\right), \\ \mu(\mathbf{x}_n,\mathbf{a},\theta) &= m^\star(\mathbf{x}_n) + \frac{1}{\sqrt{S}}\sum_s \Delta_s(\mathbf{x}_n)a_s, \quad (10) \\ \Delta_s(\mathbf{x}_n) &= f^\theta_s(\mathbf{x}_n) - m^\star(\mathbf{x}_n),\ p(\mathbf{a}) = \mathcal{N}(\mathbf{a};0,\mathbf{I}). \end{aligned}$$

For scalable inference, we follow Li & Gal (2017) to approximate (9) by the α-energy (see Appendix B), with $q_\varphi(\mathbf{a}) = \mathcal{N}(\mathbf{a};\boldsymbol{\mu},\boldsymbol{\Sigma})$ and mini-batch data $\{\mathbf{x}_m,y_m\} \sim \mathcal{D}^M$:

$$\begin{aligned} &\log q^\star_{\mathcal{GP}}(\mathbf{y}|\mathbf{X},\theta) \approx \mathcal{L}^\alpha_{\mathcal{GP}}(\theta,\varphi) \\ &= \frac{N}{\alpha M}\sum_m^M \log \mathbb{E}_{q_\varphi(\mathbf{a})}\left[q^\star(y_m|\mathbf{x}_m,\mathbf{a},\theta)^\alpha\right] \quad (11) \\ &- \mathrm{D}_{\mathrm{KL}}[q_\varphi(\mathbf{a})||p(\mathbf{a})]. \end{aligned}$$

See Algorithm 1 for the full algorithm. When $\alpha \to 0$ the α-energy reduces to the variational lower-bound, and empirically the α-energy returns better approximations when $\alpha > 0$. For Bayesian linear regression (10) the exact posterior of $\mathbf{a}$ is a multivariate Gaussian, which justifies our choice of $q_\varphi(\mathbf{a})$. Stochastic optimization is applied to optimize θ and φ jointly, making our method highly scalable.

Algorithm 1 Variational Implicit Processes (VIP)

Require: data $\mathcal{D} = (\mathbf{X},\mathbf{y})$; IP $\mathcal{IP}(g_\theta(\cdot,\cdot),p_\mathbf{z})$; variational distribution $q_\varphi(\mathbf{a})$; hyper-parameter α

1: **while** not converged **do**
2: sample mini-batch $\{(\mathbf{x}_m,y_m)\}_{m=1}^M \sim \mathcal{D}^M$
3: sample S function values: $\mathbf{z}_s \sim p(\mathbf{z}), f^\theta_s(\mathbf{x}_m) = g_\theta(\mathbf{x}_m,\mathbf{z}_s)$
4: solutions of **sleep phase**: $m^\star(\mathbf{x}_m) = \frac{1}{S}\sum_{s=1}^S f^\theta_s(\mathbf{x}_m)$, $\Delta_s(\mathbf{x}_m) = f^\theta_s(\mathbf{x}_m) - m^\star(\mathbf{x}_m)$
5: compute the **wake phase** energy $\mathcal{L}^\alpha_{\mathcal{GP}}(\theta,\varphi)$ in (11) using (10)
6: gradient descent on $\mathcal{L}^\alpha_{\mathcal{GP}}(\theta,\varphi)$ w.r.t θ,φ, via reparameterization tricks
7: **end while**

3.3 Computational complexity and scalable predictive inference

Assume the evaluation of a sampled function value $f(\mathbf{x}) = g_\theta(\mathbf{x},\mathbf{z})$ for a given input $\mathbf{x}$ takes $\mathcal{O}(C)$ time. The VIP has time complexity $\mathcal{O}(CMS + MS^2 + S^3)$ in training, where M is the size of a mini-batch, and S is the number of random functions sampled from $\mathcal{IP}(g_\theta(\cdot,\cdot),p_\mathbf{z})$. Note that approximate inference techniques in $\mathbf{z}$ space, e.g. mean-field Gaussian approximations to the posterior of Bayesian neural network weights (Blundell et al., 2015; Hernández-Lobato et al., 2016; Li & Gal, 2017), also take $\mathcal{O}(CMS)$ time. Therefore when C is large (typically the case for neural networks) the additional cost is often negligible, as S is usually significantly smaller than the typical number of inducing points in sparse GP ($S = 20$ in the experiments).

Predictive inference follows the standard GP equations to compute $q^{\star}_{\mathcal{GP}}(\mathbf{f}_*|\mathbf{X}_*, \mathbf{X}, \mathbf{y}, \theta^{\star})$ on the test set $\mathbf{X}_*$ with K datapoints: $\mathbf{f}_*|\mathbf{X}_*, \mathbf{X}, \mathbf{y} \sim \mathcal{N}(\mathbf{f}_*; \mathbf{m}_*, \mathbf{\Sigma}_*)$,

$$\begin{aligned} \mathbf{m}_* &= m^{\star}(\mathbf{X}_*) + \mathbf{K}_{*\mathbf{f}}(\mathbf{K}_{\mathbf{ff}} + \sigma^2\mathbf{I})^{-1}(\mathbf{y} - m^{\star}(\mathbf{X})), \\ \mathbf{\Sigma}_* &= \mathbf{K}_{**} - \mathbf{K}_{*\mathbf{f}}(\mathbf{K}_{\mathbf{ff}} + \sigma^2\mathbf{I})^{-1}\mathbf{K}_{\mathbf{f}*}. \end{aligned} \tag{12}$$

Recall that the optimal variational GP approximation has mean and covariance functions defined as (5) and (6), respectively, which means that $\mathbf{K}_{\mathbf{ff}}$ has rank S. Therefore predictive inference requires both function evaluations and matrix inversion, which costs $\mathcal{O}(C(K+N)S + NS^2 + S^3)$ time. This complexity can be further reduced: note that the computational cost is dominated by $(\mathbf{K}_{\mathbf{ff}} + \sigma^2\mathbf{I})^{-1}$. Denote the Cholesky decomposition of the kernel matrix $\mathbf{K}_{\mathbf{ff}} = \mathbf{B}\mathbf{B}^{\top}$. It is straightforward to show that in the Bayesian linear regression problem (10) the exact posterior of $\mathbf{a}$ is $q(\mathbf{a}|\mathbf{X}, \mathbf{y}) = \mathcal{N}(\mathbf{a}; \boldsymbol{\mu}, \mathbf{\Sigma})$, with $\boldsymbol{\mu} = \frac{1}{\sigma^2}\mathbf{\Sigma}\mathbf{B}^{\top}(\mathbf{y} - \mathbf{m}), \sigma^2\mathbf{\Sigma}^{-1} = \mathbf{B}^{\top}\mathbf{B} + \sigma^2\mathbf{I}$. Therefore the parameters of the GP predictive distribution in (12) are reduced to:

$$\mathbf{m}_* = m^{\star}(\mathbf{X}_*) + \boldsymbol{\phi}_*^{\top}\boldsymbol{\mu},\ \mathbf{\Sigma}_* = \boldsymbol{\phi}_*^{\top}\mathbf{\Sigma}\phi_*, \tag{13}$$

with the elements in $\boldsymbol{\phi}_*$ as $(\boldsymbol{\phi}_*)_s = \Delta_s(\mathbf{x}_*)/\sqrt{S}$. This reduces the prediction cost to $\mathcal{O}(CKS + S^3)$, which is on par with e.g. conventional predictive inference techniques for Bayesian neural networks that also cost $\mathcal{O}(CKS)$. In practice we use the mean and covariance matrix from $q(\mathbf{a})$ to compute the predictive distribution. Alternatively one can directly sample $\mathbf{a} \sim q(\mathbf{a})$ and compute $\mathbf{f}_* = \sum_{s=1}^{S} a_s f_s^{\theta}(\mathbf{X}_*)$, which is also an $\mathcal{O}(CKS + S^3)$ inference approach but would have higher variance.

4 Experiments

In this section, we test the capability of VIPs with various tasks, including time series interpolation, Bayesian NN/LSTM inference, and Approximate Bayesian Computation (ABC) with simulators,etc. When the VIP is applied to Bayesian NN/LSTM (Example 3-4), the prior parameters over each weight are tuned individually. We use $S = 20$ for VIP unless noted otherwise. We focus on comparing VIPs as an *inference method* to other Bayesian approaches, with detailed experimental settings presented in Appendix F.

4.1 Synthetic example

We first assess the behaviours of VIPs, including its quality of uncertainty estimation and the ability to discover structures under uncertainty. The synthetic training set is generated by first sampling 300 inputs x from $\mathcal{N}(0, 1)$. Then, for each x obtained, the corresponding target y is simulated as $y = \frac{\cos 5x}{|x|+1} + \epsilon,\ \epsilon \sim \mathcal{N}(0, 0.1)$. The test set consists of 10^3 evenly spaced points on $[-3, 3]$. We use an IP with a Bayesian neural network (1-10-10-1 architecture) as the prior. We use $\alpha = 0$ for the wake-step training. We also compare VIP with the exact full GP with *optimized* compositional kernel (RBF+Periodic), and another BNN with identical architecture but trained using variational dropout (VDO) with dropout rate $p = 0.99$ and length scale $l = 0.001$. The (hyper-)parameters are optimized using 500 epochs (batch training) with Adam optimizer (learning rate = 0.01).

Figure 2 visualizes the results. Compared with VDO and the full GP, the VIP predictive mean recovers the ground truth function better. Moreover, VIP provides the best predictive uncertainty, especially when compared with VDO: it increases smoothly when $|x| \to 3$, where training data is sparse around there. Although the composition of periodic kernel helps the full GP to return a better predictive mean than VDO (but worse than VIP), it still over-fits to the data and returns a poor uncertainty estimate around $|x| \approx 2.5$. Test Negative Log-likelihood (NLL) and RMSE results reveal similar conclusions (see the left two plots in Figure 3), where VIP significantly outperforms VDO and GP.

4.2 Solar irradiance interpolation under missingness

Time series interpolation is an ideal task to evaluate the quality of uncertainty estimate. We compare the VIP ($\alpha = 0$) with a variationally sparse GP (SVGP, 100 inducing points), an exact GP and VDO on the solar irradiance dataset (Lean et al., 1995). The dataset is constructed following (Gal & Turner, 2015), where 5 segments of length 20 are removed for interpolation. All the inputs are then centered, and the targets are standardized. We use the same settings as in Section 4.1, except that we run Adam with learning rate = 0.001 for 5000 iterations. Note that GP/SVGP predictions are reproduced directly from (Gal & Turner, 2015).

Predictive interpolations are shown in Figure 4. We see that VIP and VDO give similar interpolation behaviors. However, VDO overall under-estimates uncertainty when compared with VIP, especially in the interval $[-100, 200]$. VDO also incorrectly estimates the mean function around $x = -150$ where the ground truth there is a constant. On the contrary, VIP is able to recover the correct mean estimation around this interval with high confidence. GP methods recover the exact mean of the training data with high confidence, but they return poor estimates of predictive means for interpolation. Quantitatively, the right two plots in Figure 3 show that VIP achieves the best NLL/RMSE performance, again indicating that its returns high-quality uncertainties and accurate mean predictions.

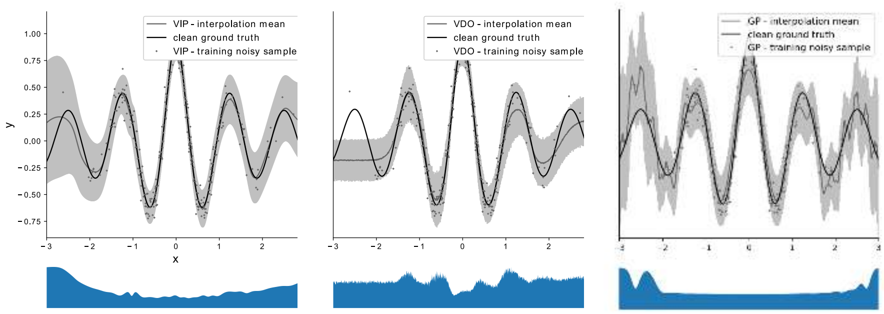

Figure 2: First row: Predictions returned from VIP (**left**), VDO (**middle**) and exact GP with RBF + Periodic kernel (**right**), respectively. **Dark grey dots**: noisy observations; **dark line**: clean ground truth function; **dark gray line**: predictive means; **Gray shaded area**: confidence intervals with 2 standard deviations. **Second row**: Corresponding predictive uncertainties.

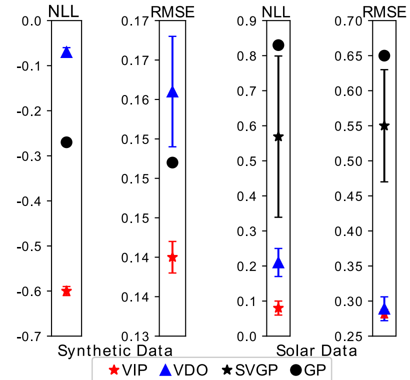

Figure 3: Test performance on synthetic example (**left two**) and solar irradiance interpolation (**right two**)

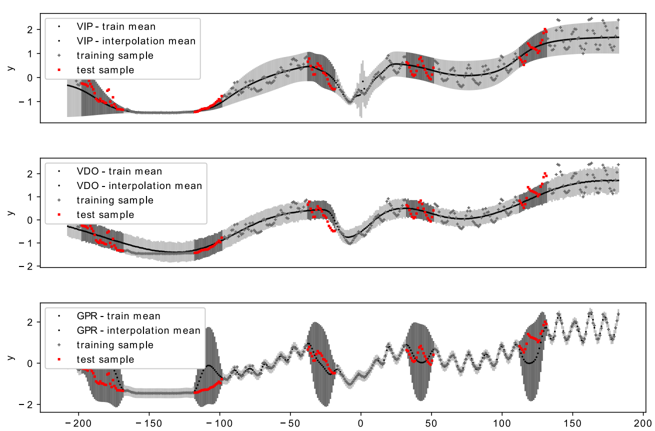

Figure 4: Interpolations returned by VIP (**top**), variational dropout (**middle**), and exact GP (**bottom**), respectively. SVGP visualization is omitted as it looks nearly the same. Here **grey dots**: training data, **red dots**: test data, **dark dots**: predictive means, **light grey and dark grey areas**: Confidence intervals with 2 standard deviations of the training and test set, respectively. Note that our GP/SVGP predictions reproduces (Gal & Turner, 2015).

4.3 Predictive Performance: Multivariate regression

We apply the VIP inference to a Bayesian neural network (VIP-BNN, example 3) and a neural sampler (VIP-NS, example 1) , using real-world multivariate regression datasets from the UCI data repository (Lichman et al., 2013). We mainly compare with the following BNNs baselines: variational Gaussian inference with reparameterization tricks (VI, Blundell et al., 2015), variational dropout (VDO, Gal & Ghahramani, 2016a), and variational alpha dropout (Li & Gal, 2017). We also include the variational GP (SVGP, (Titsias, 2009)), exact GP and the functional BNNs (fBNN)[2], and the results for fBNN is quoted from Sun et al. (2018). All neural networks have two hidden layers of size 10, and are trained for 1,000 (except for fBNNs where the results cited use 2,000 epochs). The observational noise variance for VIP and VDO is tuned over a validation set, as detailed in Appendix F. The α value for both VIP and alpha-variational inference are fixed to 0.5, as suggested in (Hernández-Lobato et al., 2016). The experiments are repeated for 10 times on all datasets except *Protein*, on which we report an averaged results across 5 repetitive runs.

Results are shown in Table 1 and 2 with the best performances boldfaced. Note that our method is not directly comparable to exact (full) GP and fBNN in the last two columns. They are only trained on small datasets since they require the computation of the *exact* GP likelihood, and fBNNs are trained for longer epochs. Therefore they are not included for the overall ranking shown in the last row of the tables. VIP methods consistently outperform other methods, obtaining the best test-NLL in 7 datasets, and the best test RMSE in 8 out of the 9 datasets. In addition, VIP-BNN obtains the best ranking among 6 methods. Note also that VIP marginally outperforms exact GPs and fBNNs (4 of 5 in NLLs), despite the comparison is not even fair. Finally, it is encouraging to see that, despite its general form, the VIP-NS achieves the second best average ranking in RMSE, outperforming many specifically designed BNN algorithms.

4.4 Bayesian LSTM for predicting power conversion efficiency of organic photovoltaics molecules

To demonstrate the scalability and flexibility of VIP, we perform experiments with the Harvard Clean Energy Project Data, the world's largest materials high-throughput virtual screening effort (Hachmann et al., 2014). A large number of molecules of organic photovoltaics are scanned to find those with high power conversion efficiency (PCE) using quantum-chemical techniques. The target value of the dataset is the PCE of each molecule, and the input is the variable-length character sequence of the molecule structures. Previous studies have handcrafted (Pyzer-Knapp et al., 2015; Bui

[2]fBNN is a recent inference method designed for BNNs, where functional priors (GPs) are used to regularize BNN training. See related work for further discussions.

Table 1: Regression experiment: Average test negative log likelihood

Dataset	N	D	VIP-BNN	VIP-NS	VI	VDO	$\alpha = 0.5$	SVGP	exact GP	fBNN
boston	506	13	**2.45±0.04**	**2.45±0.03**	2.76±0.04	2.63±0.10	**2.45±0.02**	2.63±0.04	2.46±0.04	2.30±0.10
concrete	1030	8	**3.02±0.02**	3.13±0.02	3.28±0.01	3.23±0.01	3.06±0.03	3.4±0.01	3.05±0.02	3.09±0.01
energy	768	8	0.60±0.03	**0.59±0.04**	2.17±0.02	1.13±0.02	0.95±0.09	2.31±0.02	0.57±0.02	0.68±0.02
kin8nm	8192	8	**-1.12±0.01**	-1.05±0.00	-0.81±0.01	-0.83±0.01	-0.92±0.02	-0.76±0.00	N/A±0.00	N/A±0.00
power	9568	4	2.92±0.00	2.90±0.00	2.83±0.01	2.88±0.00	**2.81±0.00**	2.82±0.00	N/A±0.00	N/A±0.00
protein	45730	9	**2.87±0.00**	2.96±0.02	3.00±0.00	2.99±0.00	2.90±0.00	3.01±0.00	N/A±0.00	N/A±0.00
red wine	1588	11	**0.97±0.02**	1.20±0.04	1.01±0.02	**0.97±0.02**	1.01±0.02	0.98±0.02	0.26±0.03	1.04±0.01
yacht	308	6	**-0.02±0.07**	0.59±0.13	1.11±0.04	1.22±0.18	0.79±0.11	2.29±0.03	0.10±0.05	1.03±0.03
naval	11934	16	-5.62±0.04	-4.11±0.00	-2.80±0.00	-2.80±0.00	-2.97±0.14	**-7.81±0.00**	N/A±0.00	N/A±0.00
Avg.Rank			**1.77±0.54**	2.77±0.57	4.66±0.28	3.88±0.38	2.55±0.37	4.44±0.66	N/A±0.00	N/A±0.00

Table 2: Regression experiment: Average test RMSE

Dataset	N	D	VIP-BNN	VIP-NS	VI	VDO	$\alpha = 0.5$	SVGP	exact GP	fBNN
boston	506	13	2.88±0.14	**2.78±0.12**	3.85±0.22	3.15±0.11	3.06±0.09	3.30±0.21	2.95±0.12	2.37±0.101
concrete	1030	8	**4.81±0.13**	5.54±0.09	6.51±0.10	6.11±0.10	5.18±0.16	7.25±0.15	5.31±0.15	4.93±0.18
energy	768	8	**0.45±0.01**	**0.45±0.05**	2.07±0.05	0.74±0.04	0.51±0.03	2.39±0.06	0.45±0.01	0.41±0.01
kin8nm	8192	8	**0.07±0.00**	0.08±0.00	0.10±0.00	0.10±0.00	0.09±0.00	0.11±0.01	N/A±0.00	N/A±0.00
power	9568	4	4.11±0.05	4.11±0.04	4.11±0.04	4.38±0.03	4.08±0.00	**4.06±0.04**	N/A±0.00	N/A±0.00
protein	45730	9	**4.25±0.07**	4.54±0.03	4.88±0.04	4.79±0.01	4.46±0.00	4.90±0.01	N/A±0.00	N/A±0.00
red wine	1588	11	**0.64±0.01**	0.66±0.01	0.66±0.01	**0.64±0.01**	0.69±0.01	0.65±0.01	0.62±0.01	0.67±0.01
yacht	308	6	**0.32±0.06**	0.54±0.09	0.79±0.05	1.03±0.06	0.49±0.04	2.25±0.13	0.35±0.04	0.60±0.06
naval	11934	16	**0.00±0.00**	**0.00±0.00**	0.38±0.00	0.01±0.00	0.01±0.00	**0.00±0.00**	N/A±0.00	N/A±0.00
Avg.Rank			**1.33±0.23**	2.22±0.36	4.66±0.33	4.00±0.44	3.11±0.42	4.44±0.72	N/A±0.00	N/A±0.00

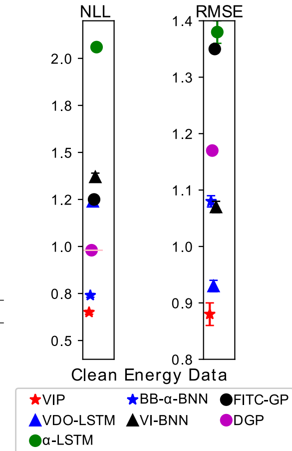

Figure 5: Test performance on clean energy dataset

et al., 2016a; Hernández-Lobato et al., 2016) or learned fingerprint features (Duvenaud et al., 2015) that transforms the raw string data into fixed-size features for prediction.

We use a VIP with a prior defined by a Bayesian LSTM (200 hidden units) and $\alpha = 0.5$. We replicate the experimental settings in Bui et al. (2016a); Hernández-Lobato et al. (2016), except that our method directly takes raw sequential molecule structure data as input. We compare our approach with a deep GP trained with expectation propagation (DGP, Bui et al., 2016a), variational dropout for LSTM (VDO-LSTM, Gal & Ghahramani, 2016b), alpha-variational inference LSTM (α-LSTM, Li & Gal, 2017), BB-α on BNN (Hernández-Lobato et al., 2016), VI on BNN (Blundell et al., 2015), and FITC GP (Snelson & Ghahramani, 2006). Results for the latter 4 methods are quoted from Hernández-Lobato et al. (2016); Bui et al. (2016a). Results in Figure 5 show that VIP significantly outperforms other baselines and hits a state-of-the-art result in test likelihood and RMSE.

4.5 ABC example: the Lotka–Volterra model

Finally, we apply the VIP on an Approximate Bayesian Computation (ABC) example with the Lotka–Volterra (L-V) model that models the continuous dynamics of stochastic population of a predator-prey system. An L-V model consists of 4 parameters $\theta = \{\theta_1, \theta_2, \theta_3, \theta_4\}$ that controls the rate of four possible random events in the model:

$$\dot{y} = \theta_1 xy - \theta_2 y, \quad \dot{x} = \theta_3 x - \theta_4 xy,$$

where x is the population of the predator, and y is the population of the prey. Therefore the L-V model is an implicit model, which allows the simulation of data but not the evaluation of model density. We follow the setup of (Papamakarios & Murray, 2016) to select the ground truth parameter of the L-V model, so that the model exhibit a oscillatory behavior which makes posterior inference difficult. Then the L-V model is simulated for 25 time units with a step size of 0.05, resulting in 500 training observations. The prediction task is to extrapolate the simulation to the $[25, 30]$ time interval.

Table 3: ABC with the Lotka–Volterra model

Method	VIP-BNN	VDO-BNN	SVGP	MCMC-ABC	SMC-ABC
Test NLL	**0.485**	1.25	1.266	0.717	0.588
Test RMSE	**0.094**	0.80	0.950	0.307	0.357

We consider (approximate) posterior inference using two types of approaches: regression-based methods (VIP-BNN, VDO-BNN and SVGP), and ABC methods (MCMC-ABC (Marjoram et al., 2003) and SMC-ABC (Beaumont et al., 2009; Bonassi et al., 2015)). ABC methods first perform posterior inference in the parameter space, then use the L-V simulator with posterior parameter samples for prediction. By contrast, regression-based methods treat this task as an ordinary regression problem, where VDO-BNN fits an approximate posterior to the NN weights, and VIP-BNN/SVGP perform predictive inference directly in function space. Results are shown in Table 3, where VIP-BNN outperforms others by a large margin in both test NLL and RMSE. More importantly, VIP is the only regression-based method that outperforms ABC methods, demonstrating its flexibility in modeling implicit systems.

5 Related Research

In the world of nonparametric models, Gaussian Processes (GPs, Rasmussen & Williams, 2006) provide accurate uncertainty estimates on unseen data, making them popular choices for Bayesian modelling in the past decades. Unfortunately, the $\mathcal{O}(N^3)$ time and $\mathcal{O}(N^2)$ space complexities make GPs impractical for large-scale datasets, therefore people often resort to approximations (Quiñonero-Candela & Rasmussen, 2005; Snelson & Ghahramani, 2006; Titsias, 2009; Hensman et al., 2013; Bui et al., 2016b; Saatçi, 2012). Another intrinsic issue is the limited representational power of GPs with stationary kernels, limiting the applications of GP methods to high dimensional data (Bengio et al., 2005).

In the world of parametric modeling, deep neural networks are extremely flexible function approximators that enable learning from very high-dimensional and structured data (Bengio, 2009; Hinton et al., 2006; Salakhutdinov & Hinton, 2009; Krizhevsky et al., 2012; Simonyan & Zisserman, 2014). As people starts to apply deep learning techniques to critical applications such as health care, uncertainty quantification of neural networks has become increasingly important. Although decent progress has been made for Bayesian neural networks (BNNs) (Denker & Lecun, 1991; Hinton & Van Camp, 1993; Barber & Bishop, 1998; Neal, 2012; Graves, 2011; Blundell et al., 2015; Hernández-Lobato & Adams, 2015; Li & Gal, 2017), uncertainty in deep learning still remains an open challenge.

Research in the *GP-BNN correspondance* has been extensively explored in order to improve the understandings of both worlds (Neal, 1996; 2012; Williams, 1997; Hazan & Jaakkola, 2015; Gal & Ghahramani, 2016a; Lee et al., 2017; Matthews et al., 2018). Notably, in Neal (1996); Gal & Ghahramani (2016a) a one-layer BNN with non-linearity $\sigma(\cdot)$ and mean-field Gaussian prior is approximately equivalent to a GP with kernel function

$$\mathcal{K}_{\text{VDO}}(\mathbf{x}_1, \mathbf{x}_2) = \mathbb{E}_{p(\mathbf{w})p(b)}[\sigma(\mathbf{w}^\top \mathbf{x}_1 + b)\sigma(\mathbf{w}^\top \mathbf{x}_2 + b)].$$

Later Lee et al. (2017) and Matthews et al. (2018) showed that a deep BNN is approximately equivalent to a GP with a *compositional kernel* (Cho & Saul, 2009; Heinemann et al., 2016; Daniely et al., 2016; Poole et al., 2016) that mimic the deep net. These approaches allow us to construct expressive kernels for GPs (Krauth et al., 2016), or conversely, exploit the *exact* Bayesian inference on GPs to perform exact Bayesian prediction for BNNs (Lee et al., 2017). The above kernel is compared with equation (6) in Appendix E.

Alternative schemes have also been investigated to exploit deep structures for GP model design. These include: (1) *deep GPs* (Damianou & Lawrence, 2013; Bui et al., 2016a), where compositions of GP priors are proposed to represent prior over compositional functions; (2) the search and design of kernels for accurate and efficient learning (van der Wilk et al., 2017; Duvenaud et al., 2013; Tobar et al., 2015; Beck & Cohn, 2017; Samo & Roberts, 2015), and (3) *deep kernel learning* that uses deep neural net features as the inputs to GPs (Hinton & Salakhutdinov, 2008; Wilson et al., 2016; Al-Shedivat et al., 2017; Bradshaw et al., 2017; Iwata & Ghahramani, 2017). Frustratingly, the first two approaches still struggle to model high-dimensional structured data such as texts and images; and the third approach is only Bayesian w.r.t. the last output layer.

The intention of our work is not to understand BNNs as GPs, nor to use deep learning to help GP design. Instead we directly treat a BNN as an instance of implicit processes (IPs), and the GP is used as a *variational distribution* to assist predictive inference.This approximation does not require previous assumptions in the GP-BNN correspondence literature (Lee et al., 2017; Matthews et al., 2018) nor the conditions in compositional kernel literature. Therefore the VIP approach also retains some of the benefits of Bayesian nonparametric approaches, and avoids issues of weight-space inference such as symmetric posterior modes.

To certain extent, the approach in Flam-Shepherd et al. (2017) resembles an inverse of VIP by encoding properties of GP priors into BNN weight priors, which is then used to regularize BNN inference. This idea is further investigated by a concurrent work on functional BNNs (Sun et al., 2018), where GP priors are directly used to regularize BNN training through gradient estimators (Shi et al., 2018).

Concurrent work of neural process (Garnelo et al., 2018) resembles the neural sampler, a special case of IPs. However, it performs inference in $\mathbf{z}$ space using the variational auto-encoder approach (Kingma & Welling, 2013; Rezende et al., 2014), which is not applicable to other IPs such as BNNs. By contrast, the proposed VIP approach applies to any IPs, and performs inference in function space. In the experiments we also show improved accuracies of the VIP approach on neural samplers over many existing Bayesian approaches.

6 Conclusions

We presented a variational approach for learning and Bayesian inference over function space based on implicit process priors. It provides a powerful framework that combines the rich flexibilities of implicit models with the well-calibrated uncertainty estimates from (parametric/nonparametric) Bayesian models. As an example, with BNNs as the implicit process prior, our approach outperformed many existing GP/BNN methods and achieved significantly improved results on molecule regression data. Many directions remain to be explored. Better posterior approximation methods beyond GP prior matching in function space will be designed. Classification models with implicit process priors will be developed. Implicit process latent variable models will also be derived in a similar fashion as Gaussian process latent variable models. Future work will investigate novel inference methods for models equipped with other implicit process priors, e.g. data simulators in astrophysics, ecology and climate science.

Acknowledgements

We thank Jiri Hron, Rich Turner, Andrew Foong, David Burt, Wenbo Gong and Cheng Zhang for discussions. Chao Ma thanks Microsoft Research donation, and Natural Science Foundation of Guangdong Province (2017A030313397) for supporting his research.

References

Al-Shedivat, M., Wilson, A. G., Saatchi, Y., Hu, Z., and Xing, E. P. Learning scalable deep kernels with recurrent structure. *The Journal of Machine Learning Research*, 18 (1):2850–2886, 2017.

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

Balog, M., Lakshminarayanan, B., Ghahramani, Z., Roy, D. M., and Teh, Y. W. The mondrian kernel. *arXiv preprint arXiv:1606.05241*, 2016.

Barber, D. and Bishop, C. M. Ensemble learning in Bayesian neural networks. *NATO ASI SERIES F COMPUTER AND SYSTEMS SCIENCES*, 168:215–238, 1998.

Beaumont, M. A., Cornuet, J.-M., Marin, J.-M., and Robert, C. P. Adaptive approximate bayesian computation. *Biometrika*, 96(4):983–990, 2009.

Beck, D. and Cohn, T. Learning kernels over strings using gaussian processes. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, volume 2, pp. 67–73, 2017.

Bengio, Y. Learning deep architectures for ai. *Foundations and trends® in Machine Learning*, 2(1):1–127, 2009.

Bengio, Y., Delalleau, O., and Le Roux, N. The curse of dimensionality for local kernel machines. *Techn. Rep*, 1258, 2005.

Blundell, C., Cornebise, J., Kavukcuoglu, K., and Wierstra, D. Weight uncertainty in neural networks. *arXiv:1505.05424*, 2015.

Bonassi, F. V., West, M., et al. Sequential monte carlo with adaptive weights for approximate bayesian computation. *Bayesian Analysis*, 10(1):171–187, 2015.

Bradshaw, J., Matthews, A. G. d. G., and Ghahramani, Z. Adversarial examples, uncertainty, and transfer testing robustness in Gaussian process hybrid deep networks. *arXiv:1707.02476*, 2017.

Bui, T., Hernández-Lobato, D., Hernandez-Lobato, J., Li, Y., and Turner, R. Deep Gaussian processes for regression using approximate expectation propagation. In *International Conference on Machine Learning*, pp. 1472–1481, 2016a.

Bui, T. D. and Turner, R. E. Tree-structured Gaussian process approximations. In *Advances in Neural Information Processing Systems*, pp. 2213–2221, 2014.

Bui, T. D., Yan, J., and Turner, R. E. A unifying framework for sparse Gaussian process approximation using power expectation propagation. *arXiv:1605.07066*, 2016b.

Cho, Y. and Saul, L. K. Kernel methods for deep learning. In *Advances in Neural Information Processing Systems*, pp. 342–350, 2009.

Cover, T. M. and Thomas, J. A. *Elements of information theory*. John Wiley & Sons, 2012.

Cunningham, J. P., Shenoy, K. V., and Sahani, M. Fast gaussian process methods for point process intensity estimation. In *Proceedings of the 25th International Conference on Machine Learning*, pp. 192–199. ACM, 2008.

Cutajar, K., Bonilla, E. V., Michiardi, P., and Filippone, M. Random feature expansions for deep Gaussian processes. *arXiv:1610.04386*, 2016.

Damianou, A. and Lawrence, N. Deep gaussian processes. In *Artificial Intelligence and Statistics*, pp. 207–215, 2013.

Daniely, A., Frostig, R., and Singer, Y. Toward deeper understanding of neural networks: The power of initialization and a dual view on expressivity. In *Advances in Neural Information Processing Systems*, pp. 2253–2261, 2016.

Denker, J. and Lecun, Y. Transforming neural-net output levels to probability distributions. In *Advances in Neural Information Processing Systems 3*. Citeseer, 1991.

Depeweg, S., Hernández-Lobato, J. M., Doshi-Velez, F., and Udluft, S. Learning and policy search in stochastic dynamical systems with Bayesian neural networks. *arXiv:1605.07127*, 2016.

Diggle, P. J. and Gratton, R. J. Monte carlo methods of inference for implicit statistical models. *Journal of the Royal Statistical Society. Series B (Methodological)*, pp. 193–227, 1984.

Duvenaud, D., Lloyd, J. R., Grosse, R., Tenenbaum, J. B., and Ghahramani, Z. Structure discovery in nonparametric regression through compositional kernel search. *arXiv:1302.4922*, 2013.

Duvenaud, D. K., Maclaurin, D., Iparraguirre, J., Bombarell, R., Hirzel, T., Aspuru-Guzik, A., and Adams, R. P. Convolutional networks on graphs for learning molecular fingerprints. In *Advances in Neural Information Processing Systems*, pp. 2224–2232, 2015.

Flam-Shepherd, D., Requeima, J., and Duvenaud, D. Mapping gaussian process priors to bayesian neural networks. *NIPS Bayesian deep learning workshop*, 2017.

Gal, Y. and Ghahramani, Z. Dropout as a Bayesian approximation: Representing model uncertainty in deep learning. In *International Conference on Machine Learning*, pp. 1050–1059, 2016a.

Gal, Y. and Ghahramani, Z. A theoretically grounded application of dropout in recurrent neural networks. In *Advances in Neural Information Processing Systems*, pp. 1019–1027, 2016b.

Gal, Y. and Turner, R. Improving the Gaussian process sparse spectrum approximation by representing uncertainty in frequency inputs. In *International Conference on Machine Learning*, pp. 655–664, 2015.

Garnelo, M., Schwarz, J., Rosenbaum, D., Viola, F., Rezende, D. J., Eslami, S., and Teh, Y. W. Neural processes. *arXiv preprint arXiv:1807.01622*, 2018.

Globerson, A. and Livni, R. Learning infinite-layer networks: beyond the kernel trick. arxiv preprint. *arXiv preprint arXiv:1606.05316*, 2016.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, pp. 2672–2680, 2014.

Graves, A. Practical variational inference for neural networks. In *Advances in Neural Information Processing Systems*, pp. 2348–2356, 2011.

Guss, W. H. Deep function machines: Generalized neural networks for topological layer expression. *arXiv preprint arXiv:1612.04799*, 2016.

Hachmann, J., Olivares-Amaya, R., Jinich, A., Appleton, A. L., Blood-Forsythe, M. A., Seress, L. R., Roman-Salgado, C., Trepte, K., Atahan-Evrenk, S., Er, S., et al. Lead candidates for high-performance organic photovoltaics from high-throughput quantum chemistry–the harvard clean energy project. *Energy & Environmental Science*, 7(2):698–704, 2014.

Hazan, T. and Jaakkola, T. Steps toward deep kernel methods from infinite neural networks. *arXiv:1508.05133*, 2015.

Heinemann, U., Livni, R., Eban, E., Elidan, G., and Globerson, A. Improper deep kernels. In *Artificial Intelligence and Statistics*, pp. 1159–1167, 2016.

Hensman, J., Fusi, N., and Lawrence, N. D. Gaussian processes for big data. *arXiv:1309.6835*, 2013.

Hernández-Lobato, J. M. and Adams, R. P. Probabilistic backpropagation for scalable learning of Bayesian neural networks. *arXiv:1502.05336*, 2015.

Hernández-Lobato, J. M., Li, Y., Rowland, M., Hernández-Lobato, D., Bui, T., and Turner, R. E. Black-box α-divergence minimization. 2016.

Hinton, G. E. and Salakhutdinov, R. R. Using deep belief nets to learn covariance kernels for Gaussian processes. In *Advances in Neural Information Processing Systems*, pp. 1249–1256, 2008.

Hinton, G. E. and Van Camp, D. Keeping the neural networks simple by minimizing the description length of the weights. In *Proceedings of the Sixth Annual Conference on Computational Learning Theory*, pp. 5–13. ACM, 1993.

Hinton, G. E., Dayan, P., Frey, B. J., and Neal, R. M. The" wake-sleep" algorithm for unsupervised neural networks. *Science*, 268(5214):1158, 1995.

Hinton, G. E., Osindero, S., and Teh, Y.-W. A fast learning algorithm for deep belief nets. *Neural computation*, 18 (7):1527–1554, 2006.

Itô, K. *An Introduction to Probability Theory*. Cambridge University Press, 1984.

Iwata, T. and Ghahramani, Z. Improving output uncertainty estimation and generalization in deep learning via neural network Gaussian processes. *arXiv:1707.05922*, 2017.

Jordan, M. I., Ghahramani, Z., Jaakkola, T. S., and Saul, L. K. An introduction to variational methods for graphical models. *Machine learning*, 37(2):183–233, 1999.

Kingma, D. P. and Welling, M. Auto-encoding variational bayes. *arXiv:1312.6114*, 2013.

Krauth, K., Bonilla, E. V., Cutajar, K., and Filippone, M. Autogp: Exploring the capabilities and limitations of Gaussian process models. *arXiv:1610.05392*, 2016.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, pp. 1097–1105, 2012.

Lázaro-Gredilla, M., Quiñonero-Candela, J., Rasmussen, C. E., and Figueiras-Vidal, A. R. Sparse spectrum gaussian process regression. *Journal of Machine Learning Research*, 11(Jun):1865–1881, 2010.

Le Roux, N. and Bengio, Y. Continuous neural networks. In *Artificial Intelligence and Statistics*, pp. 404–411, 2007.

Lean, J., Beer, J., and Bradley, R. Reconstruction of solar irradiance since 1610: Implications for climate change. *Geophysical Research Letters*, 22(23):3195–3198, 1995.

Lee, J., Bahri, Y., Novak, R., Schoenholz, S. S., Pennington, J., and Sohl-Dickstein, J. Deep neural networks as Gaussian processes. *arXiv:1711.00165*, 2017.

Li, Y. and Gal, Y. Dropout inference in Bayesian neural networks with alpha-divergences. *arXiv:1703.02914*, 2017.

Li, Y. and Turner, R. E. Rényi divergence variational inference. In *Advances in Neural Information Processing Systems*, pp. 1073–1081, 2016.

Li, Y., Hernández-Lobato, J. M., and Turner, R. E. Stochastic expectation propagation. In *Advances in Neural Information Processing Systems*, pp. 2323–2331, 2015.

Li, Y., Turner, R. E., and Liu, Q. Approximate inference with amortised MCMC. *arXiv:1702.08343*, 2017.

Lichman, M. et al. Uci machine learning repository, 2013.

Liu, Q. and Feng, Y. Two methods for wild variational inference. *arXiv:1612.00081*, 2016.

Loeve, M. In *Probability Theory I-II*. Springer, 1977.

Marjoram, P., Molitor, J., Plagnol, V., and Tavaré, S. Markov chain monte carlo without likelihoods. *Proceedings of the National Academy of Sciences*, 100(26):15324–15328, 2003.

Matthews, A. G. d. G., Van Der Wilk, M., Nickson, T., Fujii, K., Boukouvalas, A., León-Villagrá, P., Ghahramani, Z., and Hensman, J. GPflow: A Gaussian process library using tensorflow. *The Journal of Machine Learning Research*, 18(1):1299–1304, 2017.

Matthews, A. G. d. G., Rowland, M., Hron, J., Turner, R. E., and Ghahramani, Z. Gaussian process behaviour in wide deep neural networks. *arXiv:1804.11271*, 2018.

Minka, T. Power EP. Technical report, Technical report, Microsoft Research, Cambridge, 2004.

Minka, T. P. Expectation propagation for approximate Bayesian inference. In *Proceedings of the Seventeenth Conference on Uncertainty in Artificial Intelligence*, pp. 362–369. Morgan Kaufmann Publishers Inc., 2001.

Neal, R. M. Priors for infinite networks. In *Bayesian Learning for Neural Networks*, pp. 29–53. Springer, 1996.

Neal, R. M. *Bayesian learning for neural networks*, volume 118. Springer Science & Business Media, 2012.

Papamakarios, G. and Murray, I. Fast ε-free inference of simulation models with bayesian conditional density estimation. In *Advances in Neural Information Processing Systems*, pp. 1028–1036, 2016.

Poole, B., Lahiri, S., Raghu, M., Sohl-Dickstein, J., and Ganguli, S. Exponential expressivity in deep neural networks through transient chaos. In *Advances in Neural Information Processing Systems*, pp. 3360–3368, 2016.

Pyzer-Knapp, E. O., Li, K., and Aspuru-Guzik, A. Learning from the harvard clean energy project: The use of neural networks to accelerate materials discovery. *Advanced Functional Materials*, 25(41):6495–6502, 2015.

Quiñonero-Candela, J. and Rasmussen, C. E. A unifying view of sparse approximate Gaussian process regression. *Journal of Machine Learning Research*, 6(Dec):1939–1959, 2005.

Rahimi, A. and Recht, B. Random features for large-scale kernel machines. In *Advances in neural information processing systems*, pp. 1177–1184, 2008.

Rasmussen, C. E. and Williams, C. K. *Gaussian processes for machine learning*, volume 1. MIT press Cambridge, 2006.

Rezende, D. J. and Mohamed, S. Variational inference with normalizing flows. *arXiv:1505.05770*, 2015.

Rezende, D. J., Mohamed, S., and Wierstra, D. Stochastic backpropagation and approximate inference in deep generative models. *arXiv preprint arXiv:1401.4082*, 2014.

Saatçi, Y. *Scalable inference for structured Gaussian process models*. PhD thesis, University of Cambridge, 2012.

Salakhutdinov, R. and Hinton, G. E. Deep boltzmann machines. In *AISTATS*, volume 1, pp. 3, 2009.

Salimans, T., Goodfellow, I., Zaremba, W., Cheung, V., Radford, A., and Chen, X. Improved techniques for training gans. In *Advances in Neural Information Processing Systems*, pp. 2234–2242, 2016.

Samo, Y.-L. K. and Roberts, S. String gaussian process kernels. *arXiv preprint arXiv:1506.02239*, 2015.

Seeger, M., Williams, C., and Lawrence, N. Fast forward selection to speed up sparse Gaussian process regression. In *Artificial Intelligence and Statistics 9*, number EPFL-CONF-161318, 2003.

Shah, A., Wilson, A., and Ghahramani, Z. Student-t processes as alternatives to Gaussian processes. In *Artificial Intelligence and Statistics*, pp. 877–885, 2014.

Shi, J., Chen, J., Zhu, J., Sun, S., Luo, Y., Gu, Y., and Zhou, Y. ZhuSuan: A library for Bayesian deep learning. *arXiv:1709.05870*, 2017.

Shi, J., Sun, S., and Zhu, J. A spectral approach to gradient estimation for implicit distributions. *arXiv preprint arXiv:1806.02925*, 2018.

Simonyan, K. and Zisserman, A. Very deep convolutional networks for large-scale image recognition. *arXiv:1409.1556*, 2014.

Snelson, E. and Ghahramani, Z. Sparse Gaussian processes using pseudo-inputs. In *Advances in Neural Information Processing Systems*, pp. 1257–1264, 2006.

Snelson, E., Ghahramani, Z., and Rasmussen, C. E. Warped gaussian processes. In *Advances in neural information processing systems*, pp. 337–344, 2004.

Sønderby, C. K., Caballero, J., Theis, L., Shi, W., and Huszár, F. Amortised map inference for image super-resolution. *arXiv preprint arXiv:1610.04490*, 2016.

Stinchcombe, M. B. Neural network approximation of continuous functionals and continuous functions on compactifications. *Neural Networks*, 12(3):467–477, 1999.

Sun, S., Zhang, G., Shi, J., and Grosse, R. Functional variational bayesian neural networks. 2018.

Titsias, M. K. Variational learning of inducing variables in sparse Gaussian processes. In *International Conference on Artificial Intelligence and Statistics*, pp. 567–574, 2009.

Tobar, F., Bui, T. D., and Turner, R. E. Learning stationary time series using Gaussian processes with nonparametric kernels. In *Advances in Neural Information Processing Systems*, pp. 3501–3509, 2015.

Tran, D., Ranganath, R., and Blei, D. M. Deep and hierarchical implicit models. *arXiv:1702.08896*, 2017.

Turner, R. E. and Sahani, M. Statistical inference for single- and multi-band probabilistic amplitude demodulation. In *Acoustics Speech and Signal Processing (ICASSP), 2010 IEEE International Conference on*, pp. 5466–5469. IEEE, 2010.

van der Wilk, M., Rasmussen, C. E., and Hensman, J. Convolutional Gaussian processes. In *Advances in Neural Information Processing Systems*, pp. 2845–2854, 2017.

Williams, C. K. Computing with infinite networks. In *Advances in Neural Information Processing Systems*, pp. 295–301, 1997.

Wilson, A. G., Hu, Z., Salakhutdinov, R. R., and Xing, E. P. Stochastic variational deep kernel learning. In *Advances in Neural Information Processing Systems*, pp. 2586–2594, 2016.

Zhu, H. and Rohwer, R. Information geometric measurements of generalisation. 1995.