Convolutional Normalizing Flows for Deep Gaussian Processes

Haibin Yu
Tencent
Shenzhen, China
haibin@u.nus.edu

Dapeng Liu
Tencent
Shenzhen China
rocliu@tencent.com

Bryan Kian Hsiang Low
National University of Singapore
Republic of Singapore
lowkh@comp.nus.edu.sg

Patrick Jaillet
MIT
Cambridge, MA, USA
jaillet@mit.edu

Abstract—**Deep Gaussian processes (DGPs), a hierarchical composition of GP models, have successfully boosted the expressive power of their single-layer counterpart. However, it is impossible to perform exact inference in DGPs, which has motivated the recent development of variational inference-based methods. Unfortunately, either these methods yield a biased posterior belief or it is difficult to evaluate their convergence. This paper introduces a new approach for specifying flexible, arbitrarily complex, and scalable approximate posterior distributions. The posterior distribution is constructed through a *normalizing flow* (NF) which transforms a simple initial probability into a more complex one through a sequence of invertible transformations. Moreover, a novel *convolutional normalizing flow* (CNF) is developed to improve the time efficiency and capture dependency between layers. Empirical evaluation shows that CNF DGP outperforms the state-of-the-art approximation methods for DGPs.**

Index Terms—**Normalizing flow, Gaussian process, variational inference**

I. Introduction

Gaussian process (GP) models [1]–[13] have been widely applied in the machine learning community as they are capable of providing closed-form predictions in the form of a Gaussian distribution and formal measures of predictive uncertainty. Some examples include safety-critical applications [14], computer vision [15], Bayesian optimization [16]–[27], active learning [28]–[40], among others. However, the expressiveness of GP models are limited by their kernel functions which are challenging to design and often require expert knowledge for various complex learning tasks. To this end, in recent years, we have witnessed a successful hierarchical composition of GP models into a multi-layer *deep* GP (DGP) model [41]–[47], which has boosted the expressive power significantly. Unfortunately, unlike the single-layer counterpart, DGPs do not provide tractable inference, which has motivated a series of approximate inference methods. In particular, most previous works focus on *variational inference* (VI) [41], [42], [45], [48] by imposing Gaussian posterior assumptions. However, it has been pointed out by the work of [46] that the posterior belief demonstrates non-Gaussian patterns, hence potentially compromising the performance of the VI methods due to biased posterior estimation. To address this, the work of

[46] proposed to utilize *Markov chain Monte Carlo* (MCMC) sampling method to draw unbiased samples from the posterior belief. However, it is computational costly for generating samples in both training and prediction due to its sequential sampling procedure, let alone the difficulty in evaluating the convergence.

To remedy the assumptions above, the work of [47] proposed the *implicit posterior variational inference* (IPVI) framework for DGP inference which can recover the unbiased posterior distribution efficiently. The method is inspired by the generative adversarial networks [49] which casts the DGP inference into a two-player game with the aim of searching for a Nash equilibrium. However, it is also mentioned in the paper that the Nash equilibrium is not guaranteed to be obtained.

This paper proposes a novel framework which utilizes the notion of *normalizing flows* [50]–[52] (NFs) which model the complex posterior distribution directly through a sequence of invertible neural networks. The benefits of choosing NFs are their expressiveness in modeling complex real-world data distributions and the use of NFs has demonstrated success in supervised, semi-supervised, and unsupervised learning.

II. Background and Related Work

Gaussian Process (GP). A GP defines a distribution over functions $f : \mathbb{R}^D \to \mathbb{R}$, for which any finite marginals follows a Gaussian distribution [1]. A GP is fully specified by its mean function which is often assumed to be zero and covariance (kernel) function $k : \mathbb{R}^D \times \mathbb{R}^D \to \mathbb{R}$. Suppose that a set of N inputs $\mathbf{X} \triangleq \{\mathbf{x}_n\}_{n=1}^N$ and their corresponding noisy outputs $\mathbf{y} \triangleq \{y_n\}_{n=1}^N$ are available where $y_n \triangleq f(\mathbf{x}_n) + \varepsilon$ (corrupted by an i.i.d. Gaussian noise $\varepsilon \sim \mathcal{N}(0, \sigma_n^2)$). Then, the set of latent outputs $\mathbf{f} \triangleq \{f(\mathbf{x}_n)\}_{n=1}^N$ follow a Gaussian distribution $p(\mathbf{f}) = \mathcal{N}(\mathbf{0}, \mathbf{K_{XX}})$ where $\mathbf{K_{XX}}$ denotes a covariance matrix with components $k(\mathbf{x}_n, \mathbf{x}'_n)$ for $n, n' = 1, \ldots, N$. A widely used covariance function $k(\mathbf{x}_n, \mathbf{x}'_n)$ is the *squared exponential* (SE) kernel with *automatic relevance determination* (ARD) $k_\theta(\mathbf{x}, \mathbf{x}') \triangleq \sigma_f^2 \exp(-0.5 \sum_{d=1}^D (x_d - x'_d)^2 / l_d^2)$ where l_d is the lengthscale for the d-th input dimension, σ_f^2 is the kernel variance, and $\theta \triangleq (\{l_d\}_{d=1}^D, \sigma_f)$ are the kernel hyperparameters.

It follows that the marginal likelihood $p(\mathbf{y}) = \mathcal{N}(\mathbf{y} | \mathbf{0}, \mathbf{K_{XX}} + \sigma_n^2 \mathbf{I})$. The GP posterior belief for the

latent outputs $\mathbf{f}^\star \triangleq \{f(\mathbf{x}^\star)\}_{\mathbf{x}^\star \in \mathbf{X}^\star}$ can be computed analytically for any set $\mathbf{X}_\star$ of test inputs:

$$p(\mathbf{f}^\star|\mathbf{y}) = \int p(\mathbf{f}^\star|\mathbf{f})p(\mathbf{f}|\mathbf{y}) \, \mathrm{d}\mathbf{f} \tag{1}$$

which can be written as

$$p(\mathbf{f}^\star|\mathbf{y}) = \mathcal{N}(\boldsymbol{\mu}^\star, \boldsymbol{\Sigma}^\star)$$

where $\boldsymbol{\mu}^\star \triangleq \mathbf{k}_{\mathbf{x}^\star\mathbf{X}}(\mathbf{K}_{\mathbf{XX}} + \sigma_n^2\mathbf{I})^{-1}\mathbf{y}$ and $\boldsymbol{\Sigma}^\star \triangleq k_{\mathbf{x}^\star\mathbf{x}^\star} - \mathbf{k}_{\mathbf{x}^\star\mathbf{X}}(\mathbf{K}_{\mathbf{XX}} + \sigma_n^2\mathbf{I})^{-1}\mathbf{k}_{\mathbf{Xx}^\star}$.

Unfortunately, the inference procedure above incurs $\mathcal{O}(N^3)$ time, hence scaling poorly to massive datasets. To improve its scalability, the *sparse GP* (SGP) models spanned by the unifying view of [53] exploit a set $\mathbf{u} \triangleq \{u_m \triangleq f(\mathbf{z}_m)\}_{m=1}^M$ of inducing output variables for some small set $\mathbf{Z} \triangleq \{\mathbf{z}_m\}_{m=1}^M$ of inducing inputs (i.e., $M \ll N$). Then,

$$p(\mathbf{y}, \mathbf{f}, \mathbf{u}) = p(\mathbf{y}|\mathbf{f}) \, p(\mathbf{f}|\mathbf{u}) \, p(\mathbf{u}) \tag{2}$$

such that $p(\mathbf{f}|\mathbf{u}) = \mathcal{N}(\mathbf{f}|\mathbf{K}_{\mathbf{XZ}}\mathbf{K}_{\mathbf{ZZ}}^{-1}\mathbf{u}, \mathbf{K}_{\mathbf{XX}} - \mathbf{K}_{\mathbf{XZ}}\mathbf{K}_{\mathbf{ZZ}}^{-1}\mathbf{K}_{\mathbf{ZX}})$ where, with a slight abuse of notation, $\mathbf{u}$ is treated as a column vector here, $\mathbf{K}_{\mathbf{XZ}} \triangleq \mathbf{K}_{\mathbf{ZX}}^\top$, and $\mathbf{K}_{\mathbf{ZZ}}$ and $\mathbf{K}_{\mathbf{ZX}}$ denote covariance matrices with components $k(\mathbf{z}_m, \mathbf{z}_{m'})$ for $m, m' = 1, \ldots, M$ and $k(\mathbf{z}_m, \mathbf{x}_n)$ for $m = 1, \ldots, M$ and $n = 1, \ldots, N$, respectively. The SGP predictive belief can also be computed in closed form by marginalizing out $\mathbf{u}$: $p(\mathbf{f}^\star|\mathbf{y}) = \int p(\mathbf{f}^\star|\mathbf{u}) \, p(\mathbf{u}|\mathbf{y}) \, \mathrm{d}\mathbf{u}$.

The work of [54] proposed a principled *variational inference* (VI) framework that approximates the joint posterior belief $p(\mathbf{f}, \mathbf{u}|\mathbf{y})$ with a variational posterior $q(\mathbf{f}, \mathbf{u}) \triangleq p(\mathbf{f}|\mathbf{u}) \, q(\mathbf{u})$ by minimizing the *Kullback-Leibler* (KL) divergence between them, which is equivalent to maximizing a lower bound of the log-marginal likelihood (i.e., also known as the *evidence lower bound* (ELBO)):

$$\mathrm{ELBO} \triangleq \mathbb{E}_{q(\mathbf{f})}[\log p(\mathbf{y}|\mathbf{f})] - \mathrm{KL}[q(\mathbf{u})\|p(\mathbf{u})]$$

where $q(\mathbf{f}) \triangleq \int p(\mathbf{f}|\mathbf{u}) \, q(\mathbf{u}) \, \mathrm{d}\mathbf{u}$.

Deep Gaussian Process (DGP). A DGP model composes multiple layers of GP models. Consider a DGP model with a depth of L such that each DGP layer is associated with a set $\mathbf{F}_{\ell-1}$ of inputs and a set $\mathbf{F}_\ell$ of outputs for $\ell = 1, \ldots, L$ and $\mathbf{F}_0 \triangleq \mathbf{X}$. Let $\boldsymbol{\mathcal{F}} \triangleq \{\mathbf{F}_\ell\}_{\ell=1}^L$, and the inducing inputs and corresponding inducing output variables for DGP layers $\ell = 1, \ldots, L$ be denoted by the respective sets $\boldsymbol{\mathcal{Z}} \triangleq \{\mathbf{Z}_\ell\}_{\ell=1}^L$ and $\boldsymbol{\mathcal{U}} \triangleq \{\mathbf{U}_\ell\}_{\ell=1}^L$. Similar to the joint probability distribution of the SGP model in (2), the joint probability distribution of DGP can be written as

$$p(\mathbf{y}, \boldsymbol{\mathcal{F}}, \boldsymbol{\mathcal{U}}) = \underbrace{p(\mathbf{y}|\mathbf{F}_L)}_{\text{data likelihood}} \underbrace{\left[\prod_{\ell=1}^L p(\mathbf{F}_\ell|\mathbf{U}_\ell)\right] p(\boldsymbol{\mathcal{U}})}_{\text{DGP prior}}.$$

Similarly, the variational posterior is assumed to be $q(\boldsymbol{\mathcal{F}}, \boldsymbol{\mathcal{U}}) \triangleq \left[\prod_{\ell=1}^L p(\mathbf{F}_\ell|\mathbf{U}_\ell)\right] q(\boldsymbol{\mathcal{U}})$, thus resulting in the following ELBO for the DGP model:

$$\mathrm{ELBO} \triangleq \int q(\mathbf{F}_L) \log p(\mathbf{y}|\mathbf{F}_L) \, \mathrm{d}\mathbf{F}_L - \mathrm{KL}[q(\boldsymbol{\mathcal{U}})\|p(\boldsymbol{\mathcal{U}})] \tag{3}$$

where

$$q(\mathbf{F}_L) \triangleq \int \prod_{\ell=1}^L p(\mathbf{F}_\ell|\mathbf{U}_\ell, \mathbf{F}_{\ell-1}) \, q(\boldsymbol{\mathcal{U}}) \, \mathrm{d}\mathbf{F}_1 \ldots \mathrm{d}\mathbf{F}_{L-1}\mathrm{d}\boldsymbol{\mathcal{U}} \, .$$

Note that $q(\mathbf{F}_L)$ is computed using the reparameterization trick proposed in the work of [55] and Monte Carlo sampling method proposed in the work of [45].

Previous VI frameworks for DGP models [41], [42], [45], [48] have imposed the restrictive Gaussian mean-field assumption on $q(\boldsymbol{\mathcal{U}})$. Unfortunately, it has been pointed by the work of [46] that the true posterior distribution of $q(\boldsymbol{\mathcal{U}})$ usually exhibits a high correlation across the DGP layers and is non-Gaussian, hence potentially compromising the performance of such VI-based DGP models. To further remove these assumptions, [47] proposed the IPVI framework for DGP that can capture the dependency between layers and ideally recover the unbiased posterior distribution. To achieve this, the method casts the DGP inference problem as a two-player game and search for a Nash equilibrium using *best response dynamics* (BRD).[1] However, the works of [56], [57] have pointed out the critical issue of convergence while searching for Nash equilibrium. In fact, [47] also mentioned that there is no guarantee for BRD to converge to a Nash equilibrium, hence giving no assurance of recovering the true posterior distribution.

The goal of DGP inference lies in three aspects: recovery of the true posterior, convergence analysis, and time efficiency. To achieve this, this paper proposes a novel framework based on the idea of *convolutional normalizing flows* (CNF) to model the posterior distribution directly and efficiently, as detailed in Section III.

III. CONVOLUTIONAL NORMALIZING FLOW FOR DGPS

Normalizing Flows. By examining the ELBO in Eq. 3 in detail, we can find out that the maximum of the ELBO is achieved when $\mathrm{KL}[q(\boldsymbol{\mathcal{U}})\|p(\boldsymbol{\mathcal{U}}|\mathbf{y})] = 0$. Our goal is to develop an ideal family of variational distributions $q(\boldsymbol{\mathcal{U}})$ that is flexible enough to represent the true posterior distribution. We introduce the notion of *normalizing flow* [50], [51] (NF) here. An NF describes the transformation of a simple distribution into a complex distribution by repeatedly applying a sequence of invertible mappings. Following the *change of variable* rule, the NF framework can be described as follows: Given a random variable $\mathbf{z} \in \mathbb{R}^D$ with distribution $\pi(\mathbf{z})$, there exists an invertible and smooth mapping $f : \mathbb{R}^D \to \mathbb{R}^D$. Then, the resulting random variable $\mathbf{x} = f(\mathbf{z})$ has the distribution

$$p(\mathbf{x}) = \pi(\mathbf{z}) \left|\det\left(\frac{\mathrm{d}\mathbf{z}}{\mathrm{d}\mathbf{x}}\right)\right| = \pi(\mathbf{z}) \, |\det[f'(\mathbf{z})]|^{-1}$$

where $|\det[f'(\mathbf{z})]|^{-1}$ is the absolute value of the determinant of the Jacobian of f evaluated at $\mathbf{z}$.

Remark 1: It has been proven that certain NFs are universal approximators [58], [59]. In other words, it means that with a

---

[1]This procedure is sometimes called "better-response dynamics" (http://timroughgarden.org/f13/l/l16.pdf).

careful design of the mapping f, we can transform a simple distribution into any complex distribution.[2]

Recall that maximizing the ELBO is equivalent to minimizing the KL divergence, specifically, the KL divergence between our variational posterior $q(\mathcal{U})$ and the true posterior $p(\mathcal{U}|\mathbf{y})$. Following the *Bayes' theorem*, the posterior distribution can be written as

$$p(\mathcal{U}|\mathbf{y}) = \frac{p(\mathbf{y}|\mathbf{F}_L, \mathcal{U})\ p(\mathcal{U})}{p(\mathbf{y}|\mathbf{F}_L)}\ . \tag{4}$$

Similarly, $\mathbf{F}_L$ is a Monte Carlo sample from the predictive distribution of the layer outputs:

$$\mathbf{F}_L \sim \int \prod_{\ell=1}^{L-1} p(\mathbf{F}_\ell|\mathbf{U}_\ell, \mathbf{F}_{\ell-1})\ \mathrm{d}\mathbf{F}_\ell\ .$$

Inspired by the idea of NF, we propose to construct the variational posterior $q(\mathcal{U})$ through a sequence of invertible and smooth mappings: $\mathcal{G}(\mathcal{V}) = \mathcal{U}$ where $\mathcal{V}$ is a new random variable with distribution $\pi(\cdot)$. Then, according to the *change of variable* rule,

$$q(\mathcal{U}) = \pi(\mathcal{V}) \times \left| \det\left(\frac{\mathrm{d}\mathcal{G}}{\mathrm{d}\mathcal{V}}\right) \right|^{-1}. \tag{5}$$

Using (5), the KL divergence can be re-written as

$$\mathrm{KL}(q(\mathcal{U}) \| p(\mathcal{U}|\mathbf{y})) = \mathrm{KL}\left(q(\mathcal{U}) \middle\| \frac{p(\mathbf{y}|\mathbf{F}_\ell, \mathcal{U})\ p(\mathcal{U})}{p(\mathbf{y}|\mathbf{F}_\ell)}\right)$$
$$= \int q(\mathcal{U}) \left[\log \frac{q(\mathcal{U})}{p(\mathbf{y}|\mathbf{F}_\ell, \mathcal{U})\ p(\mathcal{U})}\right] \mathrm{d}\mathcal{U} + \log p(\mathbf{y}|\mathbf{F}_\ell)$$
$$= \mathbb{E}_{q(\mathcal{U})}\left[\log q(\mathcal{U}) - \log p(\mathbf{y}|\mathbf{F}_\ell, \mathcal{U}) - p(\mathcal{U})\right] + \log p(\mathbf{y}|\mathbf{F}_\ell)$$
$$= \mathbb{E}_{\pi(\mathcal{V})}\left[\log \pi(\mathcal{V}) - \log\left|\det\left(\frac{\mathrm{d}\mathcal{G}}{\mathrm{d}\mathcal{V}}\right)\right| - \log p(\mathbf{y}|\mathbf{F}_\ell, \mathcal{G}(\mathcal{V}))\right.$$
$$\left. - \log p(\mathcal{G}(\mathcal{V}))\right] + \mathrm{const}\ . \tag{6}$$

Remark 2: Note that the IPVI DGP framework in [47] implicitly represents the variational posterior with samples. Hence, to compute the log-density ratio in the KL divergence, it resorts to the use of a *discriminator* to output a function value to represent it. However, in our NF framework, the log-density for the variational posterior $\log q(\mathcal{U}) = \log \pi(\mathcal{V}) - \log|\det(\mathrm{d}\mathcal{G}/\mathrm{d}\mathcal{V})|$ results in an analytical solution.

Convolutional Normalizing Flows. We will now discuss how the architecture of the normalizing flow is designed for DGP. A naive design is to consider a layer-wise normalizing flow which is illustrated in Fig. 1. However, such a naive design suffers from the following critical issues:

- Fig. 1 shows that to recover the posterior samples of M different inducing variables $\mathbf{U}_\ell \triangleq \{\mathbf{u}_{\ell 1}, \dots, \mathbf{u}_{\ell M}\}$ where $\mathbf{u}_{\ell 1}, \dots, \mathbf{u}_{\ell M} \in \mathbb{R}^{d_\ell}$, it is natural to design the normalizing flow $g_\ell : \mathbb{R}^{M d_\ell} \to \mathbb{R}^{M d_\ell}$. Therefore, a large number of parameters is needed, which will increase the risk of overfitting;

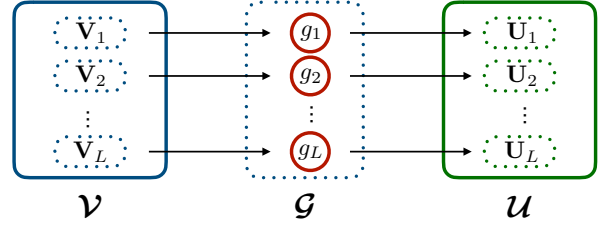[2]We refer the readers to [58]–[62] for a detailed discussion of the proof.

Fig. 1. A naive design of normalizing flow for DGP. The normalizing flow $\mathcal{G}$ is separated into L individual flows.

- Another critical issue is the computational complexity. In general, computing the log-Jacobian determinant incurs $\mathcal{O}(M^3 d_\ell^3)$ time, hence resulting in the difficulty in optimization;
- Such a design fails to capture the dependency of the inducing output variables $\mathbf{U}_\ell$ among different layers. As pointed out by [47], the posterior distribution of $p(\mathcal{U}|\mathbf{y})$ is highly correlated among layers;
- Such a design fails to adequately capture the dependency of the inducing output variables $\mathbf{U}_\ell$ on its corresponding inducing inputs $\mathbf{Z}_\ell$, hence restricting its capability to model the output posterior $\mathcal{U}$ accurately.

To resolve the above issues, we propose a novel normalizing flow architecture with convolution for DGP models, as shown in Fig. 2. Instead of treating $\mathcal{V} \triangleq \{\mathbf{V}_1, \mathbf{V}_2, \dots, \mathbf{V}_L\}$ separately, we decide to stack $\{\mathbf{V}_1, \mathbf{V}_2, \dots, \mathbf{V}_L\}$ to be a three-dimensional tensor denoted as

$$\mathcal{V} \in \mathbb{R}^{M \times 1 \times \sum_{\ell=1}^{L} d_\ell} \tag{7}$$

and design our normalizing flow $\mathcal{G} : \mathbb{R}^{M \times 1 \times \sum_{\ell=1}^{L} d_\ell} \to \mathbb{R}^{M \times 1 \times \sum_{\ell=1}^{L} d_\ell}$ accordingly, as shown in Fig. 2. To this end,
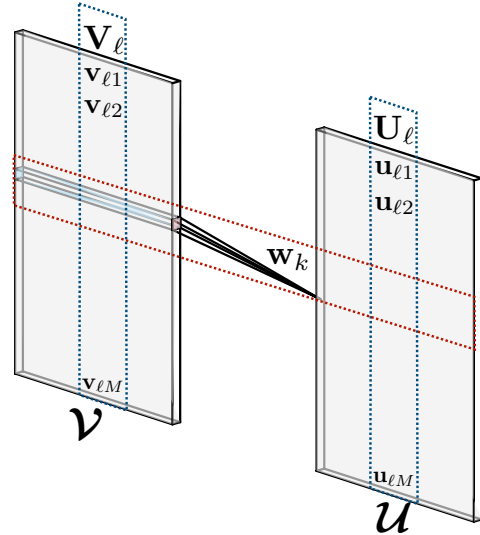

Fig. 2. The normalizing flow with convolution for DGP. The kernel tensor $\mathbf{W}$ can be decomposed into a set of tensors $\{\mathbf{w}_1, \mathbf{w}_2, \dots, \mathbf{w}_K\}$ where $\mathbf{w}_1, \dots, \mathbf{w}_K \in \mathbb{R}^{1 \times 1 \times \sum_{\ell=1}^{L} d_\ell}$ and $K \triangleq \sum_{\ell=1}^{L} d_\ell$. The red box indicates the convolution with $\mathbf{w}_k$.

we propose to convolve $\mathcal{V}$ with a kernel tensor $\mathbf{W}$ where $\mathbf{W} \in \mathbb{R}^{1 \times 1 \times \sum_{\ell=1}^{L} d_\ell \times \sum_{\ell=1}^{L} d_\ell}$, as shown in Fig. 2. In this manner, the normalizing flow $\mathcal{G}$ is fully characterized by the kernel tensor $\mathbf{W}$ with a significantly smaller number of parameters. Hence, the log-Jacobian determinant can be easily written as

$$\log \left| \det \left(\frac{\mathrm{d}\,\mathcal{G}}{\mathrm{d}\,\mathcal{V}} \right) \right| = M \times 1 \times \log |\det(\mathbf{W})| . \qquad (8)$$

Remark 3: Note that compared with the naive design in Fig. 1, computing the log-Jacobian determinant only incurs $\mathcal{O}((\sum_{\ell=1}^{L} d_\ell)^3)$ time, which reduces the time complexity by a factor of $\mathcal{O}(M^3)$. Moreover, compared with the naive design, it is easier to compute the determinant of $\mathbf{W}$. Another advantage is that the kernel tensor $\mathbf{W}$ naturally captures the dependency of inducing variables $\{\mathbf{U}_1, \mathbf{U}_2, \ldots, \mathbf{U}_L\}$ between layers.

Furthermore, to capture the dependency of the inducing output variables $\mathbf{U}_\ell$ on its corresponding inducing inputs $\mathbf{Z}_\ell$, we manually construct the base distribution $\pi(\mathcal{V})$ to depend on the inducing inputs $\mathcal{Z}$. Specifically, for each layer ℓ, the base distribution can be written as

$$\mathbf{V}_\ell \sim \mathcal{N}(\boldsymbol{\mu}_\ell, \boldsymbol{\sigma}_\ell), \quad [\boldsymbol{\mu}_\ell, \boldsymbol{\sigma}_\ell] = \varphi_\ell(\mathbf{Z}_\ell) \qquad (9)$$

where φ_ℓ is a neural network. We represent φ_ℓ using a two-layer neural network with the dimension of the hidden layer being 256 and leaky ReLU activation in the hidden units. Note that it utilizes a separate set of parameters for each different layer ℓ. We observe from our experiments that our normalizing flow for DGP with convolutions improves the performance considerably, which will be shown in Section IV.

IV. EXPERIMENTS AND DISCUSSIONS

A. Regression

UCI Benchmark Regression. Our experiments are first conducted on 7 UCI benchmark regression datasets. We have performed a random $0.9/0.1$ train/test split.
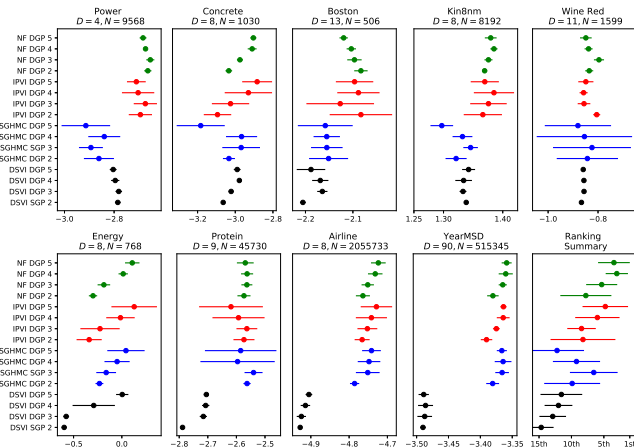

Fig. 3. Mean test log-likelihood and standard deviation achieved by our NF framework (green), IPVI (red), SGHMC (blue), and DSVI (black) for DGPs for UCI benchmark and large-scale regression datasets. Higher test log-likelihood (i.e., to the right) is better.

Large-Scale Regression. We then evaluate the performance of NF on two real-world large-scale regression datasets: (a) YearMSD dataset with a large input dimension $D = 90$ and data size $N \approx 500000$, and (b) Airline dataset with input dimension $D = 8$ and a large data size $N \approx 2$ million. For YearMSD dataset, we use the first 463715 examples as training data and the last 51630 examples as test data. For Airline dataset, we set the last 100000 examples as test data.

In the above regression tasks, the performance metric is the *mean test log-likelihood* (MLL). Fig. 3 shows the results of the mean test log-likelihood and standard deviation over 10 runs. It can be observed that NF generally outperforms other frameworks and the ranking summary shows that our NF framework for a 4-layer DGP model (NF DGP 4) performs the best on average across all regression tasks. For large-scale regression tasks, the performance of NF consistently increases with a greater depth.

Evaluation of ELBO. To further demonstrate the expressiveness of the NF framework, we have computed the estimate of training ELBO for NF DGP and IPVI DGP models on Boston dataset. Table I shows the mean ELBOs of NF and IPVI over 10 runs for the Boston dataset. NF generally achieves higher ELBOs, which agrees with results of the test MLL in Fig. 3.

TABLE I
MEAN ELBOS FOR BOSTON DATASET.

Model	NF	IPVI
DGP 2	-836.48	-846.65
DGP 3	-814.13	-846.45
DGP 4	-762.54	-776.93
DGP 5	-734.23	-758.42

Remark 4: As can be observed from Fig. 3, NF DGP is very robust across different runs (as can be seen from the smaller standard deviations) compared with SGHMC and IPVI which represent the posterior distribution with samples. Moreover, Table I clearly demonstrates that NF works better in recovering the true posterior distribution, as compared with IPVI. The inferior performance for IPVI is due to the convergence issue mentioned previously.

Time efficiency. Table II shows the time efficiency of NF DGP, as compared with IPVI DGP and SGHMC DGP.

TABLE II
TIME INCURRED BY A 5-LAYER DGP MODEL FOR AIRLINE DATASET.

	NF DGP	IPVI DGP	SGHMC DGP
Average training time	0.56 sec	0.42 sec	3.67 sec
Average generation time	0.32 sec	0.31 sec	156.2 sec

Remark 5: As can be observed from Table II, the average training time and generation time for NF DGP are slightly longer than IPVI DGP. Since NF DGP can achieve more superior predictive performance (Fig 3) as it does not suffer from convergence issues, this is an acceptable trade-off.

B. Classification

We evaluate the performance of NF in three classification tasks using the real-world MNIST, fashion-MNIST, and

CIFAR-10 datasets. Both MNIST and fashion-MNIST datasets are gray-scale images of 28×28 pixels. The CIFAR-10 dataset consists of colored images of 32×32 pixels. We utilize a 4-layer DGP model with 100 inducing inputs per layer and a robust-max multiclass likelihood [63].

Convolutional Skip-layer Connection (CSC). For the image datasets, the data distribution has local correlation between pixels. It would be better if the skip-layer connection can incorporate such information as a base for invariant mapping. We change the skip-layer connection from a fully connected one to a convolutional one; $\mathbf{W}$ is a convolution kernel with height and width of 3×3. Note that in CSC, $\mathbf{W}$ is trainable. The results in Table III show that the CSC boosts the DGP performance in real-world image datasets and performs best when integrated with the NF framework.

TABLE III
MEAN TEST ACCURACY (%) ACHIEVED BY NF, IPVI, SGHMC, AND DSVI FOR 3 CLASSIFICATION DATASETS WITH CONVOLUTIONS.

Dataset	MNIST		Fashion-MNIST		CIFAR-10	
	SGP	DGP 4	SGP	DGP 4	SGP	DGP 4
DSVI	97.32	99.16	86.98	91.57	47.15	75.05
SGHMC	96.41	98.15	85.84	88.14	47.32	70.78
IPVI	97.02	99.32	**87.29**	91.78	48.07	76.11
NF	**97.37**	**99.43**	87.14	**92.03**	**48.13**	**76.81**

V. CONCLUSION

This paper proposes a novel NF framework for DGP inference which is targeted at resolving issues of the biased mean-field Gaussian approximation as well as convergence in IPVI. We also propose a novel *convolutional NF* (CNF) architecture for DGPs to better capture dependency between inducing variables and speed up training. Empirical evaluation shows that CNF performs better than other state-of-the-art approximation methods for DGPs in regression and classification tasks. For future work, we plan to extend the CNF framework for DGP to perform semi-supervised learning by incorporating the generative expressiveness of *normalizing flows*.

REFERENCES

[1] C. E. Rasmussen and C. K. Williams, *Gaussian processes for machine learning*. MIT Press, 2006.

[2] T. N. Hoang, Q. M. Hoang, K. H. Low, and J. P. How, "Collective online learning of Gaussian processes in massive multi-agent systems," in *Proc. AAAI*, 2019.

[3] R. Ouyang and K. H. Low, "Gaussian process decentralized data fusion meets transfer learning in large-scale distributed cooperative perception," in *Proc. AAAI*, 2018, pp. 3876–3883.

[4] J. Chen, N. Cao, K. H. Low, R. Ouyang, C. K.-Y. Tan, and P. Jaillet, "Parallel gaussian process regression with low-rank covariance matrix approximations," in *Proc. UAI*, 2013, pp. 152–161.

[5] K. H. Low, J. Yu, J. Chen, and P. Jaillet, "Parallel gaussian process regression for big data: Low-rank representation meets markov approximation," in *AAAI*, 2015, pp. 2821–2827.

[6] Q. M. Hoang, T. N. Hoang, and K. H. Low, "A generalized stochastic variational Bayesian hyperparameter learning framework for sparse spectrum Gaussian process regression," in *Proc. AAAI*, 2017, pp. 2007–2014.

[7] Q. M. Hoang, T. N. Hoang, B. K. H. Low, and C. Kingsford, "Collective model fusion for multiple black-box experts," in *Proc. ICML*, 2019, pp. 2742–2750.

[8] T. N. Hoang, Q. M. Hoang, and B. K. H. Low, "A unifying framework of anytime sparse Gaussian process regression models with stochastic variational inference for big data," in *Proc. ICML*, 2015, pp. 569–578.

[9] ——, "A distributed variational inference framework for unifying parallel sparse Gaussian process regression models," in *Proc. ICML*, 2016, pp. 382–391.

[10] B. K. H. Low, N. Xu, J. Chen, K. K. Lim, and E. B. Özgül, "Generalized online sparse Gaussian processes with application to persistent mobile robot localization," in *Proc. ECML/PKDD Nectar Track*, 2014, pp. 499–503.

[11] T. Teng, J. Chen, Y. Zhang, and B. K. H. Low, "Scalable variational Bayesian kernel selection for sparse Gaussian process regression," in *Proc. AAAI*, 2020, pp. 5997–6004.

[12] N. Xu, K. H. Low, J. Chen, K. K. Lim, and E. B. Ozgul, "GP-Localize: Persistent mobile robot localization using online sparse Gaussian process observation model," in *Proc. AAAI*, 2014, pp. 2585–2592.

[13] H. Yu, T. N. Hoang, B. K. H. Low, and P. Jaillet, "Stochastic variational inference for Bayesian sparse Gaussian process regression," in *Proc. IJCNN*, 2019.

[14] D. Reeb, A. Doerr, S. Gerwinn, and B. Rakitsch, "Learning Gaussian processes by minimizing Pac-Bayesian generalization bounds," in *Proc. NeurIPS*, 2018, pp. 3337–3347.

[15] M. van der Wilk, C. E. Rasmussen, and J. Hensman, "Convolutional Gaussian processes," in *Proc. NeurIPS*, 2017, pp. 2849–2858.

[16] N. Srinivas, A. Krause, S. M. Kakade, and M. Seeger, "Gaussian process optimization in the bandit setting: No regret and experimental design," *Proc. ICML*, pp. 1015–1022, 2010.

[17] E. A. Daxberger and K. H. Low, "Distributed batch Gaussian process optimization," in *Proc. ICML*, 2017, pp. 951–960.

[18] T. N. Hoang, Q. M. Hoang, and K. H. Low, "Decentralized high-dimensional Bayesian optimization with factor graphs," in *Proc. AAAI*, 2018, pp. 3231–3238.

[19] C. K. Ling, K. H. Low, and P. Jaillet, "Gaussian process planning with Lipschitz continuous reward functions: Towards unifying Bayesian optimization, active learning, and beyond," in *Proc. AAAI*, 2016, pp. 1860–1866.

[20] D. Kharkovskii, Z. Dai, and B. K. H. Low, "Private outsourced Bayesian optimization," in *Proc. ICML*, 2020.

[21] D. Kharkovskii, C. K. Ling, and B. K. H. Low, "Nonmyopic Gaussian process optimization with macro-actions," in *Proc. AISTATS*, 2020, pp. 4593–4604.

[22] Z. Dai, H. Yu, B. K. H. Low, and P. Jaillet, "Bayesian optimization meets Bayesian optimal stopping," in *Proc. ICML*, 2019, pp. 1496–1506.

[23] Y. Zhang, T. N. Hoang, B. K. H. Low, and M. Kankanhalli, "Information-based multi-fidelity Bayesian optimization," in *Proc. NeurIPS Workshop on Bayesian Optimization*, 2017.

[24] Y. Zhang, Z. Dai, and B. K. H. Low, "Bayesian optimization with binary auxiliary information," in *Proc. UAI*, 2020, pp. 1222–1232.

[25] Q. P. Nguyen, S. Tay, B. K. H. Low, and P. Jaillet, "Top-k ranking Bayesian optimization," in *Proc. AAAI*, 2021.

[26] Z. Dai, B. K. H. Low, and P. Jaillet, "Federated Bayesian optimization via Thompson sampling," in *Proc. NeurIPS*, 2020, pp. 9687–9699.

[27] Z. Dai, Y. Chen, B. K. H. Low, P. Jaillet, and T.-H. Ho, "R2-B2: Recursive reasoning-based Bayesian optimization for no-regret learning in games," in *Proc. ICML*, 2020, pp. 2291–2301.

[28] C. Zimmer, M. Meister, and D. Nguyen-Tuong, "Safe active learning for time-series modeling with Gaussian processes," in *Proc. NeurIPS*, 2018, pp. 2730–2739.

[29] N. Cao, K. H. Low, and J. M. Dolan, "Multi-robot informative path planning for active sensing of environmental phenomena: A tale of two algorithms," in *Proc. AAMAS*, 2013, pp. 7–14.

[30] T. N. Hoang, K. H. Low, P. Jaillet, and M. Kankanhalli, "Nonmyopic ϵ-Bayes-optimal active learning of Gaussian processes," in *Proc. ICML*, 2014, pp. 739–747.

[31] K. H. Low, J. M. Dolan, and P. Khosla, "Adaptive multi-robot wide-area exploration and mapping," in *Proc. AAMAS*, 2008, pp. 23–30.

[32] ——, "Information-theoretic approach to efficient adaptive path planning for mobile robotic environmental sensing," in *Proc. ICAPS*, 2009, pp. 233–240.

[33] ——, "Active Markov information-theoretic path planning for robotic environmental sensing," in *Proc. AAMAS*, 2011, pp. 753–760.

[34] K. H. Low, J. Chen, J. M. Dolan, S. Chien, and D. R. Thompson, "Decentralized active robotic exploration and mapping for probabilistic

field classification in environmental sensing," in *Proc. AAMAS*, 2012, pp. 105–112.

[35] R. Ouyang, K. H. Low, J. Chen, and P. Jaillet, "Multi-robot active sensing of non-stationary Gaussian process-based environmental phenomena," in *Proc. AAMAS*, 2014, pp. 573–580.

[36] Y. Zhang, T. N. Hoang, K. H. Low, and M. Kankanhalli, "Near-optimal active learning of multi-output Gaussian processes," in *Proc. AAAI*, 2016, pp. 2351–2357.

[37] J. Chen, K. H. Low, P. Jaillet, and Y. Yao, "Gaussian process decentralized data fusion and active sensing for spatiotemporal traffic modeling and prediction in mobility-on-demand systems," *IEEE Transactions on Automation Science and Engineering*, vol. 12, no. 3, pp. 901–921, 2015.

[38] J. Chen, K. H. Low, C. K.-Y. Tan, A. Oran, P. Jaillet, J. Dolan, and G. Sukhatme, "Decentralized data fusion and active sensing with mobile sensors for modeling and predicting spatiotemporal traffic phenomena," in *Proc. UAI*, 2012, pp. 163–173.

[39] J. Chen, K. H. Low, and C. K. Y. Tan, "Gaussian process-based decentralized data fusion and active sensing for mobility-on-demand system," in *Proc. RSS*, 2013.

[40] Q. P. Nguyen, B. K. H. Low, and P. Jaillet, "An information-theoretic framework for unifying active learning problems," in *Proc. AAAI*, 2021.

[41] A. Damianou and N. Lawrence, "Deep Gaussian processes," in *Proc. AISTATS*, 2013, pp. 207–215.

[42] J. Hensman and N. D. Lawrence, "Nested variational compression in deep Gaussian processes," arXiv:1412.1370, 2014.

[43] T. Bui, D. Hernández-Lobato, J. Hernandez-Lobato, Y. Li, and R. Turner, "Deep Gaussian processes for regression using approximate expectation propagation," in *Proc. ICML*, 2016, pp. 1472–1481.

[44] K. Cutajar, E. V. Bonilla, P. Michiardi, and M. Filippone, "Random feature expansions for deep Gaussian processes," in *Proc. ICML*, 2017, pp. 884–893.

[45] H. Salimbeni and M. Deisenroth, "Doubly stochastic variational inference for deep Gaussian processes," in *Proc. NeurIPS*, 2017, pp. 4588–4599.

[46] M. Havasi, J. M. Hernández-Lobato, and J. J. Murillo-Fuentes, "Inference in deep Gaussian processes using stochastic gradient Hamiltonian Monte Carlo," in *Proc. NeurIPS*, 2018, pp. 7517–7527.

[47] H. Yu, Y. Chen, Z. Dai, K. H. Low, and P. Jaillet, "Implicit posterior variational inference for deep Gaussian processes," in *Proc. NeurIPS*, 2019, pp. 14 475–14 486.

[48] Z. Dai, A. Damianou, J. González, and N. Lawrence, "Variational auto-encoded deep Gaussian processes," in *Proc. ICLR*, 2016.

[49] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Proc. NeurIPS*, 2014, pp. 2672–2680.

[50] E. G. Tabak and E. Vanden-Eijnden, "Density estimation by dual ascent of the log-likelihood," *Communications in Mathematical Sciences*, vol. 8, no. 1, pp. 217–233, 2010.

[51] E. G. Tabak and C. V. Turner, "A family of nonparametric density estimation algorithms," *Communications on Pure and Applied Mathematics*, vol. 66, no. 2, pp. 145–164, 2013.

[52] D. Rezende and S. Mohamed, "Variational inference with normalizing flows," in *Proc. ICML*, 2015, pp. 1530–1538.

[53] J. Quiñonero-Candela and C. E. Rasmussen, "A unifying view of sparse approximate Gaussian process regression," *The Journal of Machine Learning Research*, vol. 6, pp. 1939–1959, 2005.

[54] M. Titsias, "Variational learning of inducing variables in sparse Gaussian processes," in *Proc. AISTATS*, 2009, pp. 567–574.

[55] D. P. Kingma and M. Welling, "Auto-encoding variational Bayes," in *Proc. ICLR*, 2013.

[56] I. Goodfellow, "NIPS 2016 tutorial: Generative adversarial networks," arXiv:1701.00160, 2016.

[57] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, "Improved techniques for training gans," in *Proc. NeurIPS*, 2016, pp. 2234–2242.

[58] C.-W. Huang, D. Krueger, A. Lacoste, and A. Courville, "Neural autoregressive flows," in *Proc. ICML*, 2018, pp. 2078–2087.

[59] P. Jaini, I. Kobyzev, M. Brubaker, and Y. Yu, "Tails of triangular flows," arXiv:1907.04481, 2019.

[60] C. Villani, *Topics in optimal transportation*. American Mathematical Society, 2003, no. 58.

[61] V. I. Bogachev, A. V. Kolesnikov, and K. V. Medvedev, "Triangular transformations of measures," *Matematicheskii Sbornik*, vol. 196, no. 3, pp. 3–30, 2005.

[62] K. V. Medvedev, "Certain properties of triangular transformations of measures," *Theory of Stochastic Processes*, vol. 14, no. 1, pp. 95–99, 2008.

[63] D. Hernández-Lobato, J. M. Hernández-Lobato, and P. Dupont, "Robust multi-class Gaussian process classification," in *Proc. NeurIPS*, 2011, pp. 280–288.