Random Feature Expansions for Deep Gaussian Processes

Kurt Cutajar [1] **Edwin V. Bonilla** [2] **Pietro Michiardi** [1] **Maurizio Filippone** [1]

Abstract

The composition of multiple Gaussian Processes as a Deep Gaussian Process (DGP) enables a deep probabilistic nonparametric approach to flexibly tackle complex machine learning problems with sound quantification of uncertainty. Existing inference approaches for DGP models have limited scalability and are notoriously cumbersome to construct. In this work we introduce a novel formulation of DGPs based on random feature expansions that we train using stochastic variational inference. This yields a practical learning framework which significantly advances the state-of-the-art in inference for DGPs, and enables accurate quantification of uncertainty. We extensively showcase the scalability and performance of our proposal on several datasets with up to 8 million observations, and various DGP architectures with up to 30 hidden layers.

1. Introduction

Given their impressive performance on machine learning and pattern recognition tasks, Deep Neural Networks (DNNs) have recently attracted a considerable deal of attention in several applied domains such as computer vision and natural language processing; see, e.g., LeCun et al. (2015) and references therein. Deep Gaussian Processes (DGPs; Damianou & Lawrence, 2013) alleviate the outstanding issue characterizing DNNs of having to specify the number of units in hidden layers by implicitly working with infinite representations at each layer. From a generative perspective, DGPs transform the inputs using a cascade of Gaussian Processes (GPs; Rasmussen & Williams,

2006) such that the output of each layer of GPs forms the input to the GPs at the next layer, effectively implementing a deep probabilistic nonparametric model for compositions of functions (Neal, 1996; Duvenaud et al., 2014).

Because of their probabilistic formulation, it is natural to approach the learning of DGPs through Bayesian inference techniques; however, the application of such techniques to learn DGPs leads to various forms of intractability. A number of contributions have been proposed to recover tractability, extending or building upon the literature on approximate methods for GPs. Nevertheless, only few works leverage one of the key features that arguably make DNNs so successful, that is being scalable through the use of mini-batch-based learning (Hensman & Lawrence, 2014; Dai et al., 2016; Bui et al., 2016). Even among these works, there does not seem to be an approach that is truly applicable to large-scale problems, and practical beyond only a few hidden layers.

In this paper, we develop a practical learning framework for DGP models that significantly improves the state-of-the-art on those aspects. In particular, our proposal introduces two sources of approximation to recover tractability, while (i) scaling to large-scale problems, (ii) being able to work with moderately deep architectures, and (iii) being able to accurately quantify uncertainty. The first is a model approximation, whereby the GPs at all layers are approximated using random feature expansions (Rahimi & Recht, 2008); the second approximation relies upon stochastic variational inference to retain a probabilistic and scalable treatment of the approximate DGP model.

We show that random feature expansions for DGP models yield Bayesian DNNs with low-rank weight matrices, and the expansion of different covariance functions results in different DNN activation functions, namely trigonometric for the Radial Basis Function (RBF) covariance, and Rectified Linear Unit (ReLU) functions for the ARC-COSINE covariance. In order to retain a probabilistic treatment of the model we adapt the work on variational inference for DNNs and variational autoencoders (Graves, 2011; Kingma & Welling, 2014) using mini-batch-based stochastic gradient optimization, which can exploit GPU and distributed computing. In this respect, we can view the probabilistic treatment of DGPs approximated through random feature

---
[1]Department of Data Science, EURECOM, France [2]School of Computer Science and Engineering, University of New South Wales, Australia. Correspondence to: Kurt Cutajar <kurt.cutajar@eurecom.fr>, Pietro Michiardi <pietro.michiardi@eurecom.fr>, Edwin V. Bonilla Cutajar <e.bonilla@unsw.edu.au>, Maurizio Filippone <maurizio.filippone@eurecom.fr>.

expansions as a means to specify sensible and interpretable priors for probabilistic DNNs. Furthermore, unlike popular inducing points-based approximations for DGPs, the resulting learning framework does not involve any matrix decompositions in the size of the number of inducing points, but only matrix products. We implement our model in TensorFlow (Abadi et al., 2015), which allows us to rely on automatic differentiation to apply stochastic variational inference.

Although having to select the appropriate number of random features goes against the nonparametric formulation favored in GP models, the level of approximation can be tuned based on constraints on running time or hardware. Most importantly, the random feature approximation enables us to develop a learning framework for DGPs which significantly advances the state-of-the-art. We extensively demonstrate the effectiveness of our proposal on a variety of regression and classification problems by comparing it with DNNs and other state-of-the-art approaches to infer DGPs. The results indicate that for a given DGP architecture, our proposal is consistently faster at achieving better generalization compared to the competitors. Another key observation is that the proposed DGP outperforms DNNs trained with dropout when quantifying uncertainty.

We focus part of the experiments on large-scale problems, such as MNIST8M digit classification and the AIRLINE dataset, which contain over 8 and 5 million observations, respectively. Only very recently there have been attempts to demonstrate performance of GP models on such large data sets (Wilson et al., 2016; Krauth et al., 2016), and our proposal is on par with these latest GP methods. Furthermore, we obtain impressive results when employing our learning framework to DGPs with moderate depth (few tens of layers) on the AIRLINE dataset. We are not aware of any other DGP models having such depth that can achieve comparable performance when applied to datasets with millions of observations. Crucially, we obtain all these results by running our algorithm on a single machine without GPUs, but our proposal is designed to be able to exploit GPU and distributed computing to significantly accelerate our deep probabilistic learning framework (see supplement for experiments in distributed mode).

In summary, the most significant contributions of this work are as follows: (i) we propose a novel approximation of DGPs based on random feature expansions that we study in connection with DNNs; (ii) we demonstrate the ability of our proposal to systematically outperform state-of-the-art methods to carry out inference in DGP models, especially for large-scale problems and moderately deep architectures; (iii) we validate the superior quantification of uncertainty offered by DGPs compared to DNNs.

1.1. Related work

Following the original proposal of DGP models in Damianou & Lawrence (2013), there have been several attempts to extend GP inference techniques to DGPs. Notable examples include the extension of inducing point approximations (Hensman & Lawrence, 2014; Dai et al., 2016) and Expectation Propagation (Bui et al., 2016). Sequential inference for training DGPs has also been investigated in Wang et al. (2016). A recent example of a DGP "natively" formulated as a variational model appears in Tran et al. (2016). Our work is the first to employ random feature expansions to approximate DGPs as DNNs. The expansion of the squared exponential covariance for DGPs leads to trigonometric DNNs, whose properties were studied in Sopena et al. (1999). Meanwhile, the expansion of the arccosine covariance is inspired by Cho & Saul (2009), and it allows us to show that DGPs with such covariance can be approximated with DNNs having ReLU activations.

The connection between DGPs and DNNs has been pointed out in several papers, such as Neal (1996) and Duvenaud et al. (2014), where pathologies with deep nets are investigated. The approximate DGP model described in our work becomes a DNN with low-rank weight matrices, which have been used in, e.g., Novikov et al. (2015); Sainath et al. (2013); Denil et al. (2013) as a regularization mechanism. Dropout is another technique to speed-up training and improve generalization of DNNs that has recently been linked to variational inference (Gal & Ghahramani, 2016).

Random Fourier features for large scale kernel machines were proposed in Rahimi & Recht (2008), and their application to GPs appears in Lázaro-Gredilla et al. (2010). In the case of squared exponential covariances, variational learning of the posterior over the frequencies was proposed in Gal & Turner (2015) to avoid potential overfitting caused by optimizing these variables. These approaches are special cases of our DGP model when using no hidden layers.

In our work, we learn the proposed approximate DGP model using stochastic variational inference. Variational learning for DNNs was first proposed in Graves (2011), and later extended to include the reparameterization trick to clamp randomness in the computation of the gradient with respect to the posterior over the weights (Kingma & Welling, 2014; Rezende et al., 2014), and to include a Gaussian mixture prior over the weights (Blundell et al., 2015).

2. Preliminaries

Consider a supervised learning scenario where a set of input vectors $X = [\mathbf{x}_1, \ldots, \mathbf{x}_n]^\top$ is associated with a set of (possibly multivariate) labels $Y = [\mathbf{y}_1, \ldots, \mathbf{y}_n]^\top$, where $\mathbf{x}_i \in R^{D_{\mathrm{in}}}$ and $\mathbf{y}_i \in R^{D_{\mathrm{out}}}$. We assume that there is an underlying function $f_o(\mathbf{x}_i)$ characterizing a mapping from the

inputs to a latent representation, and that the labels are a realization of some probabilistic process $p(y_{io}|f_o(\mathbf{x}_i))$ which is based on this latent representation.

In this work, we consider modeling the latent functions using Deep Gaussian Processes (DGPs; Damianou & Lawrence, 2013). Let variables in layer l be denoted by the (l) superscript. In DGP models, the mapping between inputs and labels is expressed as a composition of functions

$$\mathbf{f}(\mathbf{x}) = \left(\mathbf{f}^{(N_{\mathrm{h}}-1)} \circ \ldots \circ \mathbf{f}^{(0)} \right)(\mathbf{x}),$$

where each of the N_{h} layers is composed of a (possibly transformed) multivariate Gaussian process (GP). Formally, a GP is a collection of random variables such that any subset of these are jointly Gaussian distributed (Rasmussen & Williams, 2006). In GPs, the covariance between variables at different inputs is modeled using the so-called *covariance function*.

Given the relationship between GPs and single-layered neural networks with an infinite number of hidden units (Neal, 1996), the DGP model has an obvious connection with DNNs. In contrast to DNNs, where each of the hidden layers implements a parametric function of its inputs, in DGPs these functions are assigned a GP prior, and are therefore nonparametric. Furthermore, because of their probabilistic formulation, it is natural to approach the learning of DGPs through Bayesian inference techniques that lead to principled approaches for both determining the optimal settings of architecture-dependent parameters, such as the number of hidden layers, and quantification of uncertainty.

While DGPs are attractive from a theoretical standpoint, inference is extremely challenging. Denote by $F^{(l)}$ the set of latent variables with entries $f_{io}^{(l)} = f_o^{(l)}(\mathbf{x}_i)$, and let $p(Y|F^{(N_{\mathrm{h}})})$ be the conditional likelihood. Learning and making predictions with DGPs requires solving integrals that are generally intractable. For example, computing the marginal likelihood to optimize covariance parameters $\boldsymbol{\theta}^{(l)}$ at all layers entails solving

$$p(Y|X,\boldsymbol{\theta}) = \int p\left(Y|F^{(N_{\mathrm{h}})}\right) p\left(F^{(N_{\mathrm{h}})}|F^{(N_{\mathrm{h}}-1)}, \boldsymbol{\theta}^{(N_{\mathrm{h}}-1)}\right)$$
$$\times \ldots \times p\left(F^{(1)}|X,\boldsymbol{\theta}^{(0)}\right) dF^{(N_{\mathrm{h}})} \ldots dF^{(1)}.$$

In the following section we use random feature approximations to the covariance function in order to develop a scalable algorithm for inference in DGPs.

2.1. Random Feature Expansions for GPs

We start by describing how random feature expansions can be used to approximate the covariance of a single GP model. Such approximations have been considered previously, for example by Rahimi & Recht (2008) in the context of non-probabilistic kernel machines. Here we focus

on random feature expansions for the radial basis function (RBF) covariance and the ARC-COSINE covariance, which we will use in our experiments.

For the sake of clarity, we will present the covariances without any explicit scaling of the features or the covariance itself. After explaining the random feature expansion associated with each covariance, we will generalize these results in the context of DGPs to include scaling the covariance by a factor σ^2, and scaling the features for Automatic Relevance Determination (ARD) (Mackay, 1994).

2.1.1. RADIAL BASIS FUNCTION COVARIANCE

A popular example of a covariance function, which we consider here, is the Radial Basis Function (RBF) covariance

$$k_{\mathrm{rbf}}(\mathbf{x}, \mathbf{x}') = \exp\left[-\frac{1}{2} \|\mathbf{x} - \mathbf{x}'\|^{\top} \right]. \tag{1}$$

Appealing to Bochner's theorem, any continuous shift-invariant normalized covariance function $k(\mathbf{x}_i, \mathbf{x}_j) = k(\mathbf{x}_i - \mathbf{x}_j)$ is positive definite if and only if it can be rewritten as the Fourier transform of a non-negative measure $p(\boldsymbol{\omega})$ (Rahimi & Recht, 2008). Denoting the spectral frequencies by $\boldsymbol{\omega}$, while assigning $\iota = \sqrt{-1}$ and $\boldsymbol{\delta} = \mathbf{x}_i - \mathbf{x}_j$, in the case of the RBF covariance in equation (1), this yields:

$$k_{\mathrm{rbf}}(\boldsymbol{\delta}) = \int p(\boldsymbol{\omega}) \exp\left(\iota \boldsymbol{\delta}^{\top} \boldsymbol{\omega}\right) d\boldsymbol{\omega}, \tag{2}$$

with a corresponding non-negative measure $p(\boldsymbol{\omega}) = \mathcal{N}(\mathbf{0}, I)$. Because the covariance function and the non-negative measures are real, we can drop the unnecessary complex part of the argument of the expectation, keeping $\cos(\boldsymbol{\delta}^{\top}\boldsymbol{\omega}) = \cos((\mathbf{x}_i - \mathbf{x}_j)^{\top}\boldsymbol{\omega})$ that can be rewritten as $\cos(\mathbf{x}_i^{\top}\boldsymbol{\omega})\cos(\mathbf{x}_j^{\top}\boldsymbol{\omega}) + \sin(\mathbf{x}_i^{\top}\boldsymbol{\omega})\sin(\mathbf{x}_j^{\top}\boldsymbol{\omega})$.

The importance of the expansion above is that it allows us to interpret the covariance function as an expectation that can be estimated using Monte Carlo. Defining $\mathbf{z}(\mathbf{x}|\boldsymbol{\omega}) = [\cos(\mathbf{x}^{\top}\boldsymbol{\omega}), \sin(\mathbf{x}^{\top}\boldsymbol{\omega})]^{\top}$, the covariance function can be therefore unbiasedly approximated as

$$k_{\mathrm{rbf}}(\mathbf{x}_i, \mathbf{x}_j) \approx \frac{1}{N_{\mathrm{RF}}} \sum_{r=1}^{N_{\mathrm{RF}}} \mathbf{z}(\mathbf{x}_i|\tilde{\boldsymbol{\omega}}_r)^{\top} \mathbf{z}(\mathbf{x}_j|\tilde{\boldsymbol{\omega}}_r), \tag{3}$$

with $\tilde{\boldsymbol{\omega}}_r \sim p(\boldsymbol{\omega})$. This has an important practical implication, as it provides the means to access an approximate explicit representation of the mapping induced by the covariance function that, in the RBF case, is infinite dimensional (Shawe-Taylor & Cristianini, 2004). Various results have been established on the accuracy of the random Fourier feature approximation; see, e.g., Rahimi & Recht (2008).
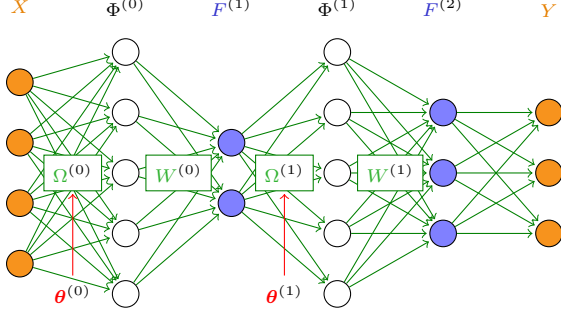
Figure 1. The proposed DGP approximation. At each hidden layer GPs are replaced by their two-layer weight-space approximation. Random-features $\Phi^{(l)}$ are obtained using a weight matrix $\Omega^{(l)}$. This is followed by a linear transformation parameterized by weights $W^{(l)}$. The prior over $\Omega^{(l)}$ is determined by the covariance parameters $\boldsymbol{\theta}^{(l)}$ of the original GPs.

2.1.2. ARC-COSINE COVARIANCE

We also consider the ARC-COSINE covariance of order p

$$
k_{\text{arc}}^{(p)}(\mathbf{x}, \mathbf{x}') = \frac{1}{\pi} \left(\|\mathbf{x}\| \, \|\mathbf{x}'\| \right)^p J_p \left(\cos^{-1} \left(\frac{\mathbf{x}^\top \mathbf{x}'}{\|\mathbf{x}\| \|\mathbf{x}'\|} \right) \right),
$$
(4)

where we have defined

$$
J_p(\alpha) = (-1)^p (\sin \alpha)^{2p+1} \left(\frac{1}{\sin \alpha} \frac{\partial}{\partial \alpha} \right)^p \left(\frac{\pi - \alpha}{\sin \alpha} \right).
$$

Let $H(\cdot)$ be the Heaviside function. Following Cho & Saul (2009), an integral representation of this covariance is:

$$
k_{\text{arc}}^{(p)}(\mathbf{x}, \mathbf{x}') = 2 \int H(\boldsymbol{\omega}^\top \mathbf{x}) \left(\boldsymbol{\omega}^\top \mathbf{x} \right)^p H(\boldsymbol{\omega}^\top \mathbf{x}') \left(\boldsymbol{\omega}^\top \mathbf{x}' \right)^p
$$
$$
\times \, \mathcal{N}(\boldsymbol{\omega}|0, I) d\boldsymbol{\omega}.
$$
(5)

This integral formulation immediately suggests a random feature approximation for the ARC-COSINE covariance in equation (4), noting that it can be seen as an expectation of the product of the same function applied to the inputs to the covariance. As before, this provides an approximate explicit representation of the mapping induced by the covariance function. Interestingly, for the ARC-COSINE covariance of order $p = 1$, this yields an approximation based on popular Rectified Linear Unit (ReLU) functions. We note that when $p = 0$ the resulting Heaviside activations are unsuitable for our inference scheme, given that they yield systematically zero gradients.

3. Random Feature Expansions for DGPs

In this section, we present our approximate formulation of DGPs which, as we illustrate in the experiments, leads to a practical learning algorithm for these deep probabilistic

nonparametric models. We propose to employ the random feature expansion at each layer, and by doing so we obtain an approximation to the original DGP model as a DNN (Figure 1).

Assume that the GPs have zero mean, and define $F^{(0)} := X$. Also, assume that the GP covariances at each layer are parameterized through a set of parameters $\boldsymbol{\theta}^{(l)}$. The parameter set $\boldsymbol{\theta}^{(l)}$ comprises the layer-wise GP marginal variances $(\sigma^2)^{(l)}$ and lengthscale parameters $\Lambda^{(l)} = \text{diag}((\ell_1^2)^{(l)}, \ldots, (\ell_{D_F^{(l)}}^2)^{(l)})$.

Considering a DGP with RBF covariances, taking a "weight-space view" of the GPs at each layer, and extending the results in the previous section, we have that

$$
\Phi_{\text{rbf}}^{(l)} = \sqrt{\frac{(\sigma^2)^{(l)}}{N_{\text{RF}}^{(l)}}} \left[\cos \left(F^{(l)} \Omega^{(l)} \right), \sin \left(F^{(l)} \Omega^{(l)} \right) \right],
$$
(6)

and $F^{(l+1)} = \Phi_{\text{rbf}}^{(l)} W^{(l)}$. At each layer, the priors over the weights are $p\left(\Omega_{\cdot j}^{(l)} \right) = \mathcal{N}\left(\mathbf{0}, \left(\Lambda^{(l)} \right)^{-1} \right)$ and $p\left(W_{\cdot i}^{(l)} \right) = \mathcal{N}(\mathbf{0}, I)$. Each matrix $\Omega^{(l)}$ has dimensions $D_{F^{(l)}} \times N_{\text{RF}}^{(l)}$. On the other hand, the weight matrices $W^{(l)}$ have dimensions $2N_{\text{RF}}^{(l)} \times D_{F^{(l+1)}}$ (weighting of sine and cosine random features), with the constraint that $D_{F^{(N_{\text{h}})}} = D_{\text{out}}$.

Similarly, considering a DGP with ARC-COSINE covariances of order $p = 1$, the application of the random feature approximation leads to DNNs with ReLU activations:

$$
\Phi_{\text{arc}}^{(l)} = \sqrt{\frac{2(\sigma^2)^{(l)}}{N_{\text{RF}}^{(l)}}} \max \left(0, F^{(l)} \Omega^{(l)} \right),
$$
(7)

with $\Omega_{\cdot j}^{(l)} \sim \mathcal{N}\left(\mathbf{0}, \left(\Lambda^{(l)} \right)^{-1} \right)$, which are cheaper to evaluate and differentiate than the trigonometric functions required in the RBF case. As in the RBF case, we allowed the covariance and the features to be scaled by $(\sigma^2)^{(l)}$ and $\Lambda^{(l)}$, respectively. The dimensions of the weight matrices $\Omega^{(l)}$ are the same as in the RBF case, but the dimensions of the $W^{(l)}$ matrices are $N_{\text{RF}}^{(l)} \times D_{F^{(l+1)}}$.

3.1. Low-rank weights in the resulting DNN

Our formulation of an approximate DGP using random feature expansions reveals a close connection with DNNs. In our formulation, the design matrices at each layer are $\Phi^{(l+1)} = \gamma \left(\Phi^{(l)} W^{(l)} \Omega^{(l+1)} \right)$, where $\gamma(\cdot)$ denotes the element-wise application of covariance-dependent functions, i.e., sine and cosine for the RBF covariance and ReLU for the ARC-COSINE covariance. Instead, for the DNN case, the design matrices are computed as $\Phi^{(l+1)} = g(\Phi^{(l)} \Omega^{(l)})$, where $g(\cdot)$ is a so-called activation function. In light of this, we can view our approximate DGP model as a DNN. From a probabilistic standpoint, we can interpret our approximate

DGP model as a DNN with specific Gaussian priors over the $\Omega^{(l)}$ weights controlled by the covariance parameters $\boldsymbol{\theta}^{(l)}$, and standard Gaussian priors over the $W^{(l)}$ weights. Covariance parameters act as hyper-priors over the weights $\Omega^{(l)}$, and the objective is to optimize these during training.

Another observation about the resulting DGP approximation is that, for a given layer l, the transformations given by $W^{(l)}$ and $\Omega^{(l+1)}$ are both linear. If we collapsed the two transformations into a single one, by introducing weights $\Xi^{(l)} = W^{(l)}\Omega^{(l+1)}$, we would have to learn $O\left(N_{RF}^{(l)} \times N_{RF}^{(l+1)}\right)$ weights at each layer, which is considerably more than learning the two separate sets of weights. As a result, we can view the proposed approximate DGP model as a way to impose a low-rank structure on the weights of DNNs, which is a form of regularization proposed in the literature of DNNs (Novikov et al., 2015; Sainath et al., 2013; Denil et al., 2013).

3.2. Variational inference

In order to keep the notation uncluttered, let $\boldsymbol{\Theta}$ be the collection of all covariance parameters $\boldsymbol{\theta}^{(l)}$ at all layers. Also, consider the case of a DGP with fixed spectral frequencies $\Omega^{(l)}$ collected into $\boldsymbol{\Omega}$, and let $\mathbf{W}$ be the collection of the weight matrices $W^{(l)}$ at all layers. For $\mathbf{W}$ we have a product of standard normal priors stemming from the approximation of the GPs at each layer $p(\mathbf{W}) = \prod_{l=0}^{N_{\mathrm{h}}-1} p(W^{(l)})$, and we propose to treat $\mathbf{W}$ using variational inference following Kingma & Welling (2014) and Graves (2011), and optimize all covariance parameters $\boldsymbol{\Theta}$. We will consider $\boldsymbol{\Omega}$ to be fixed here, but we will discuss alternative ways to treat $\boldsymbol{\Omega}$ in the next section. In the supplement we also assess the quality of the variational approximation over $\mathbf{W}$, with $\boldsymbol{\Omega}$ and $\boldsymbol{\Theta}$ fixed, by comparing it with MCMC techniques.

The marginal likelihood $p(Y|X, \boldsymbol{\Omega}, \boldsymbol{\Theta})$ involves intractable integrals, but we can obtain a tractable lower bound using variational inference. Defining $\mathcal{L} = \log[p(Y|X, \boldsymbol{\Omega}, \boldsymbol{\Theta})]$ and $\mathcal{E} = \mathrm{E}_{q(\mathbf{W})}(\log[p(Y|X, \mathbf{W}, \boldsymbol{\Omega}, \boldsymbol{\Theta})])$, we obtain

$$\mathcal{L} \geq \mathcal{E} - \mathrm{DKL}\left[q(\mathbf{W})\|p(\mathbf{W})\right], \tag{8}$$

where $q(\mathbf{W})$ acts as an approximation to the posterior over all the weights $p(\mathbf{W}|Y, X, \boldsymbol{\Omega}, \boldsymbol{\Theta})$.

We are interested in optimizing $q(\mathbf{W})$, i.e. finding an optimal approximate distribution over the parameters according to the bound above. The first term can be interpreted as a model fitting term, whereas the second as a regularization term. In the case of a Gaussian distribution $q(\mathbf{W})$ and a Gaussian prior $p(\mathbf{W})$, it is possible to compute the DKL term analytically (see supplementary material), whereas the remaining term needs to be estimated. Assume a Gaussian approximating distribution that factorizes across layers

and weights:

$$q(\mathbf{W}) = \prod_{ijl} q\left(W_{ij}^{(l)}\right) = \prod_{ijl} \mathcal{N}\left(m_{ij}^{(l)}, (s^2)_{ij}^{(l)}\right). \tag{9}$$

The variational parameters are the mean and the variance of each of the approximating factors $m_{ij}^{(l)}, (s^2)_{ij}^{(l)}$, and we aim to optimize the lower bound with respect to these as well as all covariance parameters $\boldsymbol{\Theta}$.

In the case of a likelihood that factorizes across observations, an interesting feature of the expression of the lower bound is that it is amenable to fast stochastic optimization. In particular, we derive a doubly-stochastic approximation of the expectation in the lower bound as follows. First, $\mathcal{E}$ can be rewritten as a sum over the input points, which allows us to estimate it in an unbiased fashion using mini-batches, selecting m points indexed by $\mathcal{I}_m$:

$$\mathcal{E} \approx \frac{n}{m} \sum_{k \in \mathcal{I}_m} \mathrm{E}_{q(\mathbf{W})}(\log[p(\mathbf{y}_k|\mathbf{x}_k, \mathbf{W}, \boldsymbol{\Omega}, \boldsymbol{\Theta})]). \tag{10}$$

Second, each of the elements of the sum can be estimated using N_{MC} Monte Carlo samples, yielding:

$$\mathcal{E} \approx \frac{n}{m} \sum_{k \in \mathcal{I}_m} \frac{1}{N_{\mathrm{MC}}} \sum_{r=1}^{N_{\mathrm{MC}}} \log[p(\mathbf{y}_k|\mathbf{x}_k, \tilde{\mathbf{W}}_r, \boldsymbol{\Omega}, \boldsymbol{\Theta})], \tag{11}$$

with $\tilde{\mathbf{W}}_r \sim q(\mathbf{W})$. In order to facilitate the optimization, we reparameterize the weights as follows:

$$(\tilde{W}_r^{(l)})_{ij} = s_{ij}^{(l)}\epsilon_{rij}^{(l)} + m_{ij}^{(l)}. \tag{12}$$

By differentiating the lower bound with respect to $\boldsymbol{\Theta}$ and the mean and variance of the approximate posterior over $\mathbf{W}$, we obtain an unbiased estimate of the gradient for the lower bound. The reparameterization trick ensures that the randomness in the computation of the expectation is fixed when applying stochastic gradient ascent moves to the parameters of $q(\mathbf{W})$ and $\boldsymbol{\Theta}$ (Kingma & Welling, 2014). Automatic differentiation tools enabled us to compute stochastic gradients automatically, which is why we opted to implement our model in TensorFlow (Abadi et al., 2015).

3.3. Treatment of the spectral frequencies $\boldsymbol{\Omega}$

So far, we have assumed the spectral frequencies $\boldsymbol{\Omega}$ to be sampled from the prior and fixed throughout, whereby we employ the reparameterization trick to obtain $\Omega_{ij}^{(l)} = (\beta^2)_{ij}^{(l)}\varepsilon_{rij}^{(l)} + \mu_{ij}^{(l)}$, with $(\beta^2)_{ij}^{(l)}$ and $\mu_{ij}^{(l)}$ determined by the prior $p\left(\Omega_{\cdot j}^{(l)}\right) = \mathcal{N}\left(\mathbf{0}, \left(\Lambda^{(l)}\right)^{-1}\right)$. We then draw the $\varepsilon_{rij}^{(l)}$'s and fix them from the outset, such that covariance parameters $\boldsymbol{\Theta}$ can be optimized along with $q(\mathbf{W})$. We refer to this variant as PRIOR-FIXED.
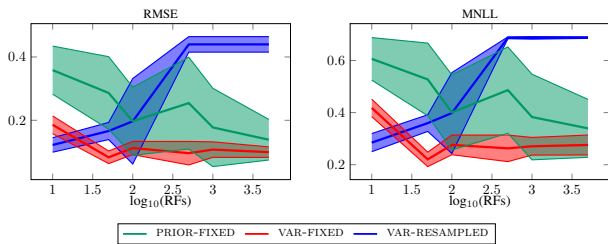
Figure 2. Performance of different strategies for dealing with $\mathbf{\Omega}$ as a function of the number of random features. These can be fixed (PRIOR-FIXED), or treated variationally (with fixed randomness VAR-FIXED and resampled at each iteration VAR-RESAMPLED).

Inspired by previous work on random feature expansions for GPs, we can think of alternative ways to treat these parameters, e.g., Lázaro-Gredilla et al. (2010); Gal & Turner (2015). In particular, we study a variational treatment of $\mathbf{\Omega}$; we refer the reader to the supplementary material for details on the derivation of the lower bound in this case. When being variational about $\mathbf{\Omega}$, we introduce an approximate posterior $q(\mathbf{\Omega})$ which also has a factorized form. We use the reparameterization trick once again, but $\mathbf{\Omega}$ are now sampled from the posterior, which in general has different mean and variances to the prior. We report two variations of this treatment, namely VAR-FIXED and VAR-RESAMPLED. In VAR-FIXED, we fix $\varepsilon_{rij}^{(l)}$ in computing $\mathbf{\Omega}$ throughout the learning of the model, whereas in VAR-RESAMPLED we resample these at each iteration. We note that one can also be variational about $\mathbf{\Theta}$, but we leave this for future work.

In Figure 2, we illustrate the differences between the strategies discussed in this section; we report the accuracy of the proposed one-layer DGP with RBF covariances with respect to the number of random features on one of the datasets that we consider in the experiment section (EEG dataset). For PRIOR-FIXED, more random features result in a better approximation of the GP priors at each layer, and this results in better generalization. When we resample $\mathbf{\Omega}$ from the approximate posterior (VAR-RESAMPLED), we notice that the model quickly struggles with the optimization when increasing the number of random features. We attribute this to the fact that the factorized form of the posterior over $\mathbf{\Omega}$ and $\mathbf{W}$ is unable to capture posterior correlations between the coefficients for the random features and the weights of the corresponding linearized model. Being deterministic about the way spectral frequencies are computed (VAR-FIXED) offers the best performance among the three learning strategies, and this is what we employ throughout the rest of this paper.

3.4. Computational complexity

When estimating the lower bound, there are two main operations performed at each layer, that is $F^{(l)}\Omega^{(l)}$ and $\Phi^{(l)}W^{(l)}$. Recalling that this matrix product is done for

samples from the posterior over $\mathbf{W}$ (and $\mathbf{\Omega}$ when treated variationally) and given the mini-batch formulation, the former costs $\mathcal{O}\left(mD_F^{(l)}N_{\mathrm{RF}}^{(l)}N_{\mathrm{MC}}\right)$, while the latter costs $\mathcal{O}\left(mN_{\mathrm{RF}}^{(l)}D_F^{(l)}N_{\mathrm{MC}}\right)$.

Because of feature expansions and stochastic variational inference, the resulting algorithm does not involve any Cholesky decompositions. This is in sharp contrast with stochastic variational inference using inducing-point approximations (see e.g. Dai et al., 2016; Bui et al., 2016), where such operations could significantly limit the number of inducing points that can be employed.

4. Experiments

We evaluate our model by comparing it against relevant alternatives for both regression and classification, and assess its performance when applied to large-scale datasets. We also investigate the extent to which such deep compositions continue to yield good performance when the number of layers is significantly increased.

4.1. Model Comparison

We primarily compare our model to the state-of-the-art DGP inference method presented in the literature, namely DGPs using expectation propagation (DGP-EP; Bui et al., 2016). We originally intended to include results for the variational auto-encoded DGP (Damianou & Lawrence, 2013); however, the results obtained using the available code were not competitive with DGP-EP and we thus decided to exclude them from the figures. We also omitted DGP training using sequential inference (Wang et al., 2016) given that we could not find an implementation of the method and, in any case, the performance reported in the paper is inferior to more recent approaches. We also compare against DNNs in order to present the results in a wider context, and demonstrate that DGPs lead to better quantification of uncertainty. Finally, to substantiate the benefits of using a deep model, we compare against the shallow sparse variational GP (Hensman et al., 2015b) implemented in GPflow (Matthews et al., 2016).

We use the same experimental set-up for both regression and classification tasks using datasets from the UCI repository (Asuncion & Newman, 2007), for models having one hidden layer. The results for architectures with two hidden layers are included in the supplementary material. The specific configurations for each model are detailed below:

DGP-RBF, DGP-ARC : In the proposed DGP with an RBF kernel, we use 100 random features at every hidden layer to construct a multivariate GP with $D_F^{(l)} = 3$, and set the batch size to $m = 200$. We initially only use a single
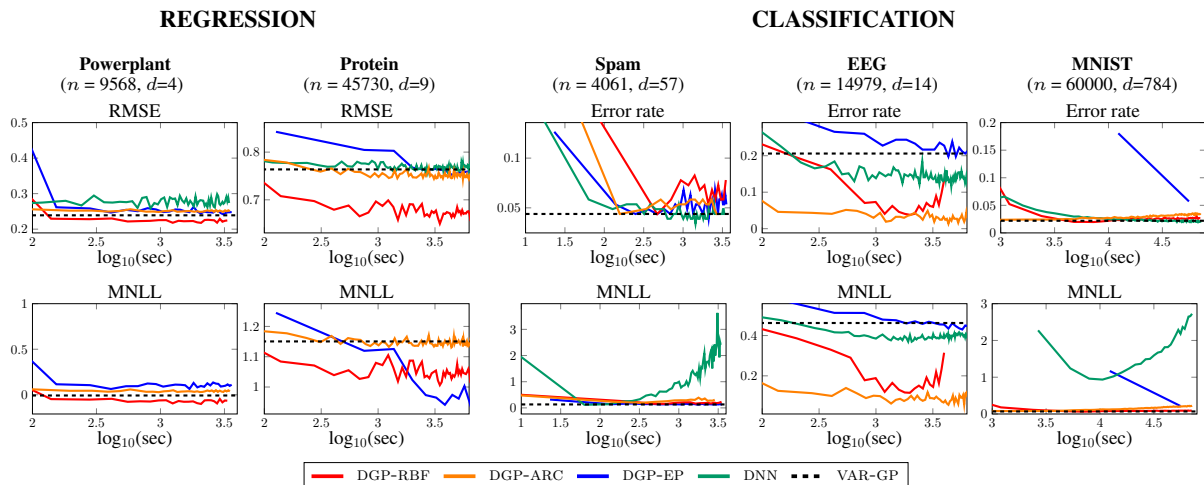
Figure 3. Progression of error rate (RMSE in the regression case) and MNLL over time for competing models. Results are shown for configurations having 1 hidden layer, while the results for models having 2 such layers may be found in the supplementary material.

Monte Carlo sample, and halfway through the allocated optimization time, this is then increased to 100 samples. We employ the Adam optimizer (Kingma & Ba, 2015) with a learning rate of 0.01, and in order to stabilize the optimization procedure, we fix the parameters Θ for $12,000$ iterations, before jointly optimizing all parameters. As discussed in Section 3.3, Ω are optimized variationally with fixed randomness. The same set-up is used for DGP-ARC, the variation of our model using the ARC-COSINE kernel;

DGP-EP [1]: For this technique, we use the same architecture and optimizer as for DGP-RBF and DGP-ARC, a batch size of 200 and 100 inducing points at each hidden layer. For the classification case, we use 100 samples for approximating the Softmax likelihood;

DNN : We construct a DNN configured with a dropout rate of 0.5 at each hidden layer in order to provide regularization during training. In order to preserve a degree of fairness, we set the number of hidden units in such a way as to ensure that the number of weights to be optimized match those in the DGP-RBF and DGP-ARC models when the random features are taken to be fixed.

We assess the performance of each model using the error rate (RMSE in the regression case) and mean negative log-likelihood (MNLL) on withheld test data. The results are averaged over 3 folds for every dataset. The experiments were launched on single nodes of a cluster of Intel Xeon E5-2630 CPUs having 32 cores and 128GB RAM.

Figure 3 shows that DGP-RBF and DGP-ARC consistently outperform competing techniques both in terms of convergence speed and predictive accuracy. This is particu-

---

[1]Code obtained from:
github.com/thangbui/deepGP_approxEP

larly significant for larger datasets where other techniques take considerably longer to converge to a reasonable error rate, although DGP-EP converges to superior MNLL for the PROTEIN dataset. The results are also competitive (and sometimes superior) to those obtained by the variational GP (VAR-GP) in Hensman et al. (2015b). It is striking to see how inferior uncertainty quantification provided by the DNN (which is inherently limited to the classification case, so no MNLL reported on regression datasets) is compared to DGPs, despite the error rate being comparable.

By virtue of its higher dimensionality, larger configurations were used for MNIST. For DGP-RBF and DGP-ARC, we use 500 random features, 50 GPs in the hidden layers, batch size of 1000, and Adam with a 0.001 learning rate. Similarly for DGP-EP, we use 500 inducing points, with the only difference being a slightly smaller batch size to cater for issues with memory requirements. Following Simard et al. (2003), we employ 800 hidden units at each layer of the DNN. The DGP-RBF peaks at 98.04% and 97.93% for 1 and 2 hidden layers respectively. It was observed that the model performance degrades noticeably when more than 2 hidden layers are used (without feeding forward the inputs). This is in line with what is reported in the literature on DNNs (Neal, 1996; Duvenaud et al., 2014). By simply re-introducing the original inputs in the hidden layer, the accuracy improves to 98.2% for the one hidden layer case.

Recent experiments on MNIST using a variational GP with MCMC report overall accuracy of 98.04% (Hensman et al., 2015a), while the AutoGP architecture has been shown to give 98.45% accuracy (Krauth et al., 2016). Using a finer-tuned configuration, DNNs were also shown to obtain 98.5% accuracy (Simard et al., 2003), whereas 98.6% has been reported for SVMs (Schölkopf, 1997). In view of this wider scope of inference techniques, it can be confirmed

Table 1. Performance of our proposal on large-scale datasets.

Dataset	Accuracy		MNLL	
	RBF	ARC	RBF	ARC
MNIST8M	99.14%	99.04%	0.0454	0.0465
AIRLINE	78.55%	72.76%	0.4583	0.5335

that the results obtained using the proposed architecture are comparable to the state-of-the-art, even if further extensions may be required for obtaining a proper edge. Note that this comparison focuses on approaches without preprocessing, and excludes convolutional neural nets.

4.2. Large-scale Datasets

One of the defining characteristics of our model is the ability to scale up to large datasets without compromising on performance and accuracy in quantifying uncertainty. As a demonstrative example, we evaluate our model on two large-scale problems which go beyond the scale of datasets to which GPs and especially DGPs are typically applied.

We first consider MNIST8M, which artificially extends the original MNIST dataset to 8+ million observations. We trained this model using the same configuration described for standard MNIST, and we obtained 99.14% accuracy on the test set using one hidden layer. Given the size of this dataset, there are only few reported results for other GP models. Most notably, Krauth et al. (2016) recently obtained 99.11% accuracy with the AutoGP framework, which is comparable to the result obtained by our model.

Meanwhile, the AIRLINE dataset contains flight information for 5+ million US flights. Following the procedure described in Hensman et al. (2013) and Wilson et al. (2016), we use this 8-dimensional dataset for classification, where the task is to determine whether a flight has been delayed or not. We construct the test set using the scripts provided in Wilson et al. (2016), where 100,000 data points are held-out for testing. We construct our DGP models using 100 random features at each layer, and set the dimensionality to $D_{F^{(l)}} = 3$. As shown in Table 1, our model works significantly better when using the RBF kernel. In addition, the results are also directly comparable to those obtained by Wilson et al. (2016), which reports accuracy and MNLL of 78.1% and 0.457, respectively.

4.3. Model Depth

Finally, we assess the scalability of our model with respect to additional hidden layers in the constructed model. In particular, we re-consider the AIRLINE dataset and evaluate the performance of DGP-RBF models constructed using up to 30 layers. In order to cater for the increased depth in the
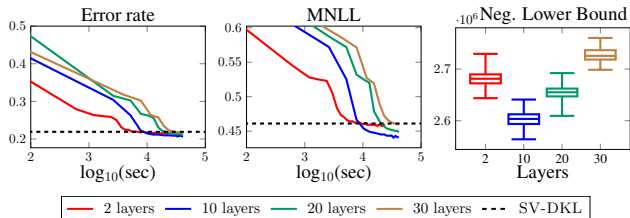

Figure 4. Left and center - Performance of our model on the AIRLINE dataset as function of time for different depths. The baseline (SV-DKL) is taken from Wilson et al. (2016). Right - The box plot of the negative lower bound, estimated over 100 mini-batches of size 50,000, confirms this is a suitable objective for model selection.

model, we feed-forward the original input to each hidden layer, as suggested in Duvenaud et al. (2014).

Figure 4 reports the progression of error rate and MNLL over time for different number of hidden layers, using the results obtained in Wilson et al. (2016) as a baseline (reportedly obtained in about 3 hours). As expected, the model takes longer to train as the number of layers increases. However, the model converges to an optimal state in every case in less than a couple of hours, with an improvement being noted in the case of 10 and 20 layers over the shallower 2-layer model. The box plot within the same figure indicates that the negative lower bound is a suitable objective function for carrying out model selection.

5. Conclusions

In this work, we have proposed a novel formulation of DGPs which exploits the approximation of covariance functions using random features, as well as stochastic variational inference for preserving the probabilistic representation of a regular GP. We demonstrated how inference using this model is not only faster, but also frequently superior to other state-of-the-art methods, with particular emphasis on competing DGP models. The results obtained for both the AIRLINE dataset and the MNIST8M digit recognition problem are particularly impressive since such large datasets have been generally considered to be beyond the computational scope of DGPs. We perceive this to be a considerable step forward in the direction of scaling and accelerating DGPs.

The results obtained on higher-dimensional datasets strongly suggest that approximations such as Fastfood (Le et al., 2013) could be instrumental in the interest of using more random features. We are also currently investigating ways to mitigate the decline in performance observed when optimizing Ω variationally with resampling. The obtained results also encourage the extension of our model to include convolutional layers suitable for computer vision applications.

References

Abadi, Martín, Agarwal, Ashish, Barham, Paul, et al. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015.

Asuncion, Arthur and Newman, David J. UCI machine learning repository, 2007.

Blundell, Charles, Cornebise, Julien, Kavukcuoglu, Koray, and Wierstra, Daan. Weight Uncertainty in Neural Network. In *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, volume 37 of *JMLR Workshop and Conference Proceedings*, pp. 1613–1622. JMLR.org, 2015.

Bui, Thang D., Hernández-Lobato, Daniel, Hernández-Lobato, José M., Li, Yingzhen, and Turner, Richard E. Deep Gaussian Processes for Regression using Approximate Expectation Propagation. In *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*, volume 48 of *JMLR Workshop and Conference Proceedings*, pp. 1472–1481. JMLR.org, 2016.

Cho, Youngmin and Saul, Lawrence K. Kernel methods for deep learning. In *Advances in Neural Information Processing Systems 22: 23rd Annual Conference on Neural Information Processing Systems 2009. Proceedings of a meeting held 7-10 December 2009, Vancouver, British Columbia, Canada.*, pp. 342–350, 2009.

Dai, Zhenwen, Damianou, Andreas, González, Javier, and Lawrence, Neil. Variational auto-encoded deep Gaussian processes. In *Proceedings of the Fourth International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, 2-4 May, 2016*, 2016.

Damianou, Andreas C. and Lawrence, Neil D. Deep Gaussian Processes. In *Proceedings of the Sixteenth International Conference on Artificial Intelligence and Statistics, AISTATS 2013, Scottsdale, AZ, USA, April 29 - May 1, 2013*, volume 31 of *JMLR Proceedings*, pp. 207–215. JMLR.org, 2013.

Denil, Misha, Shakibi, Babak, Dinh, Laurent, Ranzato, Marc'Aurelio, and de Freitas, Nando. Predicting Parameters in Deep Learning. In *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States.*, pp. 2148–2156, 2013.

Duvenaud, David K., Rippel, Oren, Adams, Ryan P., and Ghahramani, Zoubin. Avoiding pathologies in very deep networks. In *Proceedings of the Seventeenth International Conference on Artificial Intelligence and Statistics, AISTATS 2014, Reykjavik, Iceland, April 22-25, 2014*, volume 33 of *JMLR Workshop and Conference Proceedings*, pp. 202–210. JMLR.org, 2014.

Gal, Yarin and Ghahramani, Zoubin. Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning. In *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*, volume 48 of *JMLR Workshop and Conference Proceedings*, pp. 1050–1059. JMLR.org, 2016.

Gal, Yarin and Turner, Richard. Improving the Gaussian Process Sparse Spectrum Approximation by Representing Uncertainty in Frequency Inputs. In *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, volume 37 of *JMLR Workshop and Conference Proceedings*, pp. 655–664. JMLR.org, 2015.

Graves, Alex. Practical Variational Inference for Neural Networks. In Shawe-Taylor, J., Zemel, R. S., Bartlett, P. L., Pereira, F., and Weinberger, K. Q. (eds.), *Advances in Neural Information Processing Systems 24*, pp. 2348–2356. Curran Associates, Inc., 2011.

Hensman, James and Lawrence, Neil D. Nested Variational Compression in Deep Gaussian Processes, December 2014.

Hensman, James, Fusi, Nicoló, and Lawrence, Neil D. Gaussian processes for big data. In *Proceedings of the Twenty-Ninth Conference on Uncertainty in Artificial Intelligence, UAI 2013, Bellevue, WA, USA, August 11-15, 2013*, 2013.

Hensman, James, de G. Matthews, Alexander G., Filippone, Maurizio, and Ghahramani, Zoubin. MCMC for variationally sparse Gaussian processes. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pp. 1648–1656, 2015a.

Hensman, James, de G. Matthews, Alexander G., and Ghahramani, Zoubin. Scalable variational Gaussian process classification. In *Proceedings of the Eighteenth International Conference on Artificial Intelligence and Statistics, AISTATS 2015, San Diego, California, USA, May 9-12, 2015*, pp. 351–360, 2015b.

Kingma, Diederik P. and Ba, Jimmy. Adam: A method for stochastic optimization. In *Proceedings of the Third*

International Conference on Learning Representations, ICLR 2015, San Diego, California, 7-9 May, 2015, 2015.

Kingma, Diederik P. and Welling, Max. Auto-Encoding Variational Bayes. In *Proceedings of the Second International Conference on Learning Representations, ICLR 2014, Banff, Canada, April 14-16, 2014*, 2014.

Krauth, Karl, Bonilla, Edwin V., Cutajar, Kurt, and Filippone, Maurizio. AutoGP: Exploring the Capabilities and Limitations of Gaussian Process Models. *arXiv preprint 1610.05392*, October 2016.

Lázaro-Gredilla, M., Quinonero-Candela, J., Rasmussen, C. E., and Figueiras-Vidal, A. R. Sparse Spectrum Gaussian Process Regression. *Journal of Machine Learning Research*, 11:1865–1881, 2010.

Le, Quoc V., Sarls, Tams, and Smola, Alexander J. Fastfood - computing Hilbert space expansions in loglinear time. In *ICML (3)*, volume 28 of *JMLR Workshop and Conference Proceedings*, pp. 244–252. JMLR.org, 2013.

LeCun, Yann, Bengio, Yoshua, and Hinton, Geoffrey. Deep learning. *Nature*, 521(7553):436–444, 2015.

Mackay, D. J. C. Bayesian methods for backpropagation networks. In Domany, E., van Hemmen, J. L., and Schulten, K. (eds.), *Models of Neural Networks III*, chapter 6, pp. 211–254. Springer, 1994.

Matthews, Alexander G. de G., van der Wilk, Mark, Nickson, Tom, Fujii, Keisuke., Boukouvalas, Alexis, León-Villagrá, Pablo, Ghahramani, Zoubin, and Hensman, James. GPflow: A Gaussian process library using TensorFlow. *arXiv preprint 1610.08733*, October 2016.

Neal, Radford M. *Bayesian Learning for Neural Networks (Lecture Notes in Statistics)*. Springer, 1 edition, August 1996. ISBN 0387947248.

Novikov, Alexander, Podoprikhin, Dmitry, Osokin, Anton, and Vetrov, Dmitry P. Tensorizing Neural Networks. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pp. 442–450, 2015.

Rahimi, Ali and Recht, Benjamin. Random Features for Large-Scale Kernel Machines. In Platt, J. C., Koller, D., Singer, Y., and Roweis, S. T. (eds.), *Advances in Neural Information Processing Systems 20*, pp. 1177–1184. Curran Associates, Inc., 2008.

Rasmussen, Carl E. and Williams, Christopher. *Gaussian Processes for Machine Learning*. MIT Press, 2006.

Rezende, Danilo Jimenez, Mohamed, Shakir, and Wierstra, Daan. Stochastic backpropagation and approximate inference in deep generative models. In *Proceedings of the 31th International Conference on Machine Learning, ICML 2014, Beijing, China, 21-26 June 2014*, volume 32 of *JMLR Workshop and Conference Proceedings*, pp. 1278–1286. JMLR.org, 2014.

Sainath, Tara N., Kingsbury, Brian, Sindhwani, Vikas, Arisoy, Ebru, and Ramabhadran, Bhuvana. Low-rank matrix factorization for Deep Neural Network training with high-dimensional output targets. In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2013, Vancouver, BC, Canada, May 26-31, 2013*, pp. 6655–6659. IEEE, 2013. doi: 10.1109/ICASSP.2013.6638949.

Schölkopf, Bernhard. *Support vector learning*. PhD thesis, Berlin Institute of Technology, 1997.

Shawe-Taylor, John and Cristianini, Nello. *Kernel Methods for Pattern Analysis*. Cambridge University Press, New York, NY, USA, 2004.

Simard, Patrice Y., Steinkraus, Dave, and Platt, John C. Best Practices for Convolutional Neural Networks Applied to Visual Document Analysis. In *Proceedings of the Seventh International Conference on Document Analysis and Recognition - Volume 2*, ICDAR '03, Washington, DC, USA, 2003. IEEE Computer Society.

Sopena, J. M., Romero, E., and Alquezar, R. Neural networks with periodic and monotonic activation functions: a comparative study in classification problems. In *Artificial Neural Networks, 1999. ICANN 99. Ninth International Conference on (Conf. Publ. No. 470)*, volume 1, 1999. doi: 10.1049/cp:19991129.

Tran, Dustin, Ranganath, Rajesh, and Blei, David M. The Variational Gaussian Process. In *Proceedings of the Fourth International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, 2-4 May, 2016*, 2016.

Wang, Yali, Brubaker, Marcus A., Chaib-draa, Brahim, and Urtasun, Raquel. Sequential inference for deep Gaussian process. In *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics, AISTATS 2016, Cadiz, Spain, May 9-11, 2016*, pp. 694–703, 2016.

Wilson, Andrew Gordon, Hu, Zhiting, Salakhutdinov, Ruslan, and Xing, Eric P. Stochastic variational deep kernel learning. In *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pp. 2586–2594, 2016.