Deep Kernels with Probabilistic Embeddings for Small-Data Learning

Ankur Mallick[1]   **Chaitanya Dwivedi**[3]   **Bhavya Kailkhura**[2]   **Gauri Joshi**[1]   **T. Yong-Jin Han**[2]

[1]Carnegie Mellon University, Pittsburgh, Pennsylvania, USA
[2]Lawrence Livermore National Laboratory, Livermore, California, USA,
[3]PathAI, Boston, Massachusetts, USA,

Abstract

Gaussian Processes (GPs) are known to provide accurate predictions and uncertainty estimates even with small amounts of labeled data by capturing similarity between data points through their kernel function. However traditional GP kernels are not very effective at capturing similarity between high dimensional data points. Neural networks can be used to learn good representations that encode intricate structures in high dimensional data, and can be used as inputs to the GP kernel. However the huge data requirement of neural networks makes this approach ineffective in small data settings. To solves the conflicting problems of representation learning and data efficiency, we propose to learn deep kernels on probabilistic embeddings by using a probabilistic neural network. Our approach maps high-dimensional data to a probability distribution in a low dimensional subspace and then computes a kernel between these distributions to capture similarity. To enable end-to-end learning, we derive a functional gradient descent procedure for training the model. Experiments on a variety of datasets show that our approach outperforms the state-of-the-art in GP kernel learning in both supervised and semi-supervised settings. We also extend our approach to other small-data paradigms such as few-shot classification where it outperforms previous approaches on *mini*-Imagenet and CUB datasets.

1 INTRODUCTION

The availability of huge labeled datasets [Deng et al., 2009] has played a key role in the success of deep learning in recent time. Given a sufficient amount of labeled (training) data, deep neural networks achieve superhuman prediction performance on unseen (test) data [Krizhevsky et al., 2012, Graves et al., 2013]. However, machine learning is increasingly being applied to areas where it is challenging to procure large amounts of labeled data, e.g., materials science [Zhang et al., 2020], poverty prediction [Jean et al., 2016], and infrastructure assessment [Oshri et al., 2018]. The accuracy of most machine learning models deteriorates when trained on small datasets. An additional drawback of deep neural networks is the inability to provide accurate uncertainty estimates leading to over-confident but incorrect predictions [Bulusu et al., 2020] especially when trained on small data.

Gaussian Processes (GPs) [Rasmussen, 2003] are non-parametric models that leverage correlation (similarity) between data points to give a probabilistic estimate of the target function values. Exploiting similarity in a non-parametric fashion contributes to sample efficiency, while probabilistic estimation helps to quantify model uncertainty in unlabeled regions of the input space. However, owing to the curse of dimensionality, the number of points required to model the covariance using traditional GP kernels grows exponentially with data dimension [Tripathy et al., 2016] which reduces their efficacy in the *small-data* regime for high-dimensional data. The success of deep neural networks in learning low dimensional representations of high dimensional data has led to works [Wilson et al., 2016] that use a neural network to map data to a low dimensional latent space and then build a GP for prediction on the *latent space*. However, owing to the huge data requirement of neural networks, these approaches are also not as effective in learning latent representations in the *small-data regime*.

To enable powerful, data efficient representation learning, we propose a new approach for learning highly expressive GP kernels with small amounts of labeled data by mapping data points to *probability distributions* in a latent space with a probabilistic neural network [Neal, 2012]. We then leverage the theory of kernel embeddings of distributions [Muandet et al., 2017] to build expressive kernels, and consequently GPs, on the latent probability distributions.

Probabilistic models are known to outperform deterministic

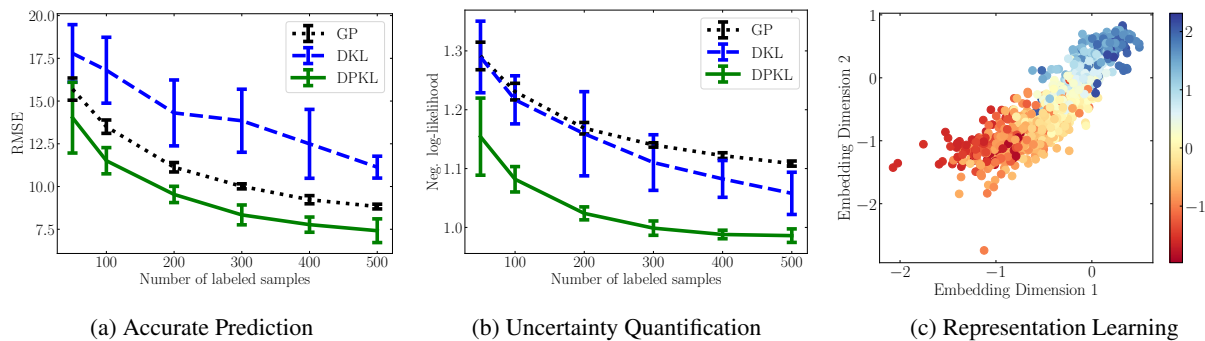| (a) Accurate Prediction | (b) Uncertainty Quantification | (c) Representation Learning |

Figure 1: Performance of our models for regression on the UCI CTSlice Dataset (Dimensionality = 384) [Graf et al., 2011] with $n = \{50, 100, 200, 300, 400, 500\}$ labeled data points. (a) Our approach DPKL, has *significantly lower* average RMSE than conventional GPs and Deep Kernel Learning (DKL). (b) DPKL not only has lower error, but also captures the data distribution much better (lower negative log likelihood) than DKL. (c) Points with similar target values have similar mean embeddings in the 2-dimensional latent space learned by DPKL with $n = 100$ labeled samples.

models when the amount of data is insufficient to capture the complexity of the task [Wilson and Izmailov, 2020]. Intuitively, we expect the uncertainty due to the model's probabilistic nature to improve generalization. This is already seen in the superiority of soft thresholding over hard thresholding in clustering, and in the superior prediction performance of Bayesian Neural Networks (BNNs) over deterministic neural networks in small data settings [Gal, 2016], and we expect to obtain similar benefits on using probabilistic neural networks for GP kernel learning.

Note that our aims differ from works like [Graves, 2011] that train BNNs for prediction. They use the probabilistic nature of the model to capture uncertainty in prediction at the *output* while we use probabilistic neural networks to improve the *learned representation* which can serve as input to *any predictive model*. While we mainly focus on improving *GP regression* through this, we also show improvements in classification using Bayesian logistic regression [Jaakkola and Jordan, 1997] on the learned probabilistic representations to illustrate the flexibility of our approach.

The difference in aims also leads to a difference in training. While BNNs are typically trained by approximating posterior distribution over model parameters, we learn the distribution over model parameters via *Maximum Likelihood Estimation* (MLE) in the space of probability distributions. MLE in our model is performed using functional gradient descent, which is not only a natural approach to perform optimization in the space of functions (probability distributions), but has also recently been shown [Liu and Wang, 2016] to achieve an effective middle ground between the high computational cost of MCMC [Andrieu et al., 2003] and the poor approximation quality of Variational Inference (VI) [Blei et al., 2017] for training probabilistic models.

Figure 1 shows the results of implementing our model, Deep

Probabilistic Kernel Learning (DPKL), and baseline approaches - Deep Kernel Learning (DKL) [Wilson and Nickisch, 2015], and a GP with a Squared Exponential (SE) kernel, on a regression task of predicting location of CT slice images on the axial axis of the human body [Graf et al., 2011] given a $384-$dimensional feature vector extracted from the image. We use Root Mean Squared Error (RMSE) (Figure 1a) to measure prediction accuracy, and negative log-likelihood (Figure 1b) to measure uncertainty quantification (lower is better for both). Our model *outperforms both baselines* on both counts and also learns embeddings that clearly capture similarity between data points with respect to target values (Figure 1c).

We also extend our model to two other paradigms in small data settings – semi-supervised learning [Zhu and Goldberg, 2009] and few-shot classification [Chen et al., 2019]. The former involves adding a regularizer to DPKL that provably incorporates additional structural information from unlabeled data, while the latter uses the learned probabilistic embedding to improve the performance of Bayesian Logistic Regression for classification especially for settings like few-shot learning where we only have a small number of training examples of each class.

2 BACKGROUND AND RELATED WORK

Given n training data points, $\mathbf{X} \in \mathcal{X} \subseteq \mathbb{R}^{n \times D}$, and targets, $\mathbf{y} \in \mathbb{R}^{n \times 1}$, our goal is to accurately predict targets y_* for unseen (test) data points $\mathbf{x}_*$, assuming small training set size (small n) and high dimensionality (large D).

Gaussian Processes. Gaussian Processes (GPs) [Rasmussen, 2003] are non-parametric models that, when appropriately designed, can give high quality predictions and uncertainty estimates in the small data (small n)

regime. A GP defines a *probability distribution* over functions $H : \mathbb{R}^d \to \mathbb{R}$ such that individual function values form a multivariate Gaussian distribution i.e. $H(\mathbf{X}) = [H(\mathbf{x}_1), \ldots, H(\mathbf{x}_n)] \sim \mathcal{N}(\boldsymbol{\mu}, \mathbf{K})$ with entries of the mean vector given by $\mu_i = \mu(\mathbf{x}_i)$, and of the covariance matrix given by $K_{ij} = K(\mathbf{x}_i, \mathbf{x}_j)$. A GP is fully specified by the mean function μ (typically $\mu \equiv 0$) and the covariance kernel K (For eg. SE or Matern) which measures similarity between data points. Data is also assumed to be corrupted by noise $\eta \sim \mathcal{N}(0, \sigma_\eta^2)$ and the overall covariance is

$$\text{Cov}(H(\mathbf{x}_i), H(\mathbf{x}_j)) = K(\mathbf{x}_i, \mathbf{x}_j) + \sigma_\eta^2 \mathbb{1}\{i = j\}$$

Given the training data $\mathbf{X}$ and associated targets $\mathbf{y}$, the predicted target function value $H(\mathbf{x}_*)$ at an unlabeled data point $\mathbf{x}_*$ follows a Gaussian posterior distribution, i.e. $\Pr(H(\mathbf{x}_*)|\mathbf{X}, \mathbf{y}, \mathbf{x}_*) = \mathcal{N}(\mu(\mathbf{x}_*), \sigma^2(\mathbf{x}_*))$ where $\mu(\mathbf{x}_*) = \mathbf{k}_*^T(\mathbf{K} + \sigma_\eta^2)^{-1}\mathbf{y}$ and $\sigma^2(\mathbf{x}_*) = k_{**} - \mathbf{k}_*^T(\mathbf{K} + \sigma_\eta^2)^{-1}\mathbf{k}_*$ with $\mathbf{k}_* = K(\mathbf{X}, \mathbf{x}_*)$ and $k_{**} = K(\mathbf{x}_*, \mathbf{x}_*)$. We refer readers to [Rasmussen, 2003] for further details.

Deep Kernel Learning. Deep Kernel Learning (DKL) [Wilson et al., 2016, Wilson and Nickisch, 2015, Al-Shedivat et al., 2016] uses a neural network to learn a GP kernel as $\mathbf{K}_{ij} = K(g_{\mathbf{w}}(\mathbf{x}_i), g_{\mathbf{w}}(\mathbf{x}_j))$ where K is a standard GP kernel and $g_{\mathbf{w}}(.)$ represents the neural network with parameters $\mathbf{w}$ which can be learned by minimizing the GP negative log likelihood, i.e.,

$$\mathbf{w}^* = \arg\min_{\mathbf{w}}(-\log\Pr(\mathbf{y} \mid \mathbf{X}, \mathbf{w})), \quad \mathbf{y} \sim \mathcal{N}(\boldsymbol{\mu}, \mathbf{K}) \tag{1}$$

Standard kernels (like SE or Matern) make simplifying assumptions on the structure and smoothness of functions which do not hold in high dimensions due to the curse of dimensionality [Garnett et al., 2013, Rana et al., 2017]. DKL avoids these assumptions by mapping high-dimensional data to low-dimensional embeddings using a neural network. But, like all neural network based models, it requires a lot of labeled data for accurate prediction.

Probabilistic Neural Networks. Probabilistic models learn a distribution over model parameters thus incorporating uncertainty in predictive modeling. The implicit regularization due to the probabilistic nature can improve generalization performance particularly when there is insufficient data. Prior works on Probabilistic or Bayesian Neural Networks (BNNs) [Neal, 2012, Graves, 2011, Hernández-Lobato and Adams, 2015] seek to learn the posterior distribution over model parameters that best explains the data. While these works use the BNN to directly predict the data, we use a probabilistic neural network to improve the quality of low dimensional embeddings of high dimensional data. The probabilistic embeddings given by our model serve as inputs to predictive models like GPs. Thus our approach aims to improve GP kernel learning and not replace BNNs. Moreover BNNs do not capture *similarity* between data points or

provide closed form uncertainty estimates on test data like GPs do. Hence GPs and BNNs are generally not used for the same tasks.

Semi-Supervised Learning. Recently, [Jean et al., 2018] proposed Semi-Supervised Deep Kernel Learning (SSDKL) approach which uses posterior regularization, given a large amount of unlabeled data, to reduce the labeled data requirement for GP kernel learning. Various other semi-supervised learning approaches [Kingma et al., 2014, Odena, 2016, Tarvainen and Valpola, 2017, Laine and Aila, 2016] have also been studied which yield accurate predictions with few labeled and many unlabeled examples. For such settings, we propose a semi-supervised model, SSDPKL in section 3 which is constrained to ensure that the unlabeled data is not projected too far away from the labeled data in the latent space, and outperforms SSDKL for regression (section 4).

Other related work. There is a line of work on Deep GPs [Damianou and Lawrence, 2013, Salimbeni and Deisenroth, 2017] seeks to improve the performance of GPs in the *big-data* regime by hierarchically stacking multiple GPs and training the model using Variational Inference. The authors of [Salimbeni and Deisenroth, 2017] observe that DeepGP is not better than a single layer GP for small datasets (with similar or more labeled examples than what we have considered) while we show that DPKL generally has lower RMSE than GP in these settings. Regression on distribution inputs has been theoretically studied in Póczos et al. [2013], Bachoc et al. [2017, 2018]. They assume that the input itself is a distribution while we use neural networks to learn distributional embeddings of deterministic inputs. There has also been a surge of interest in classification in the small-data regime or few-shot classification. Recent results are summarised in [Chen et al., 2019]. In particular, the baseline models considered in their work can be directly augmented with DPKL to improve predictive performance in the few-shot setting as we show in Section 4.

3 DEEP PROBABILISTIC KERNEL LEARNING

In this section we first describe our model - Deep Probabilistic Kernel Learning (DPKL) illustrated in Figure 2a in the context of GP regression, and then explain extensions to classification and semi-supervised learning. The proofs of theoretical results are contained in Appendix A.

3.1 MODEL OVERVIEW

DPKL predicts output $y \in \mathbb{R}$ given input $\mathbf{x} \in \mathcal{X} \subseteq \mathbb{R}^D$ using a Gaussian Process (GP) learned over *probability distributions* in a low dimensional latent space $\mathcal{Z} \subseteq \mathbb{R}^d$ where distributions in $\mathcal{Z}$ are obtained by passing the input data through a probabilistic neural network. The dimensionality
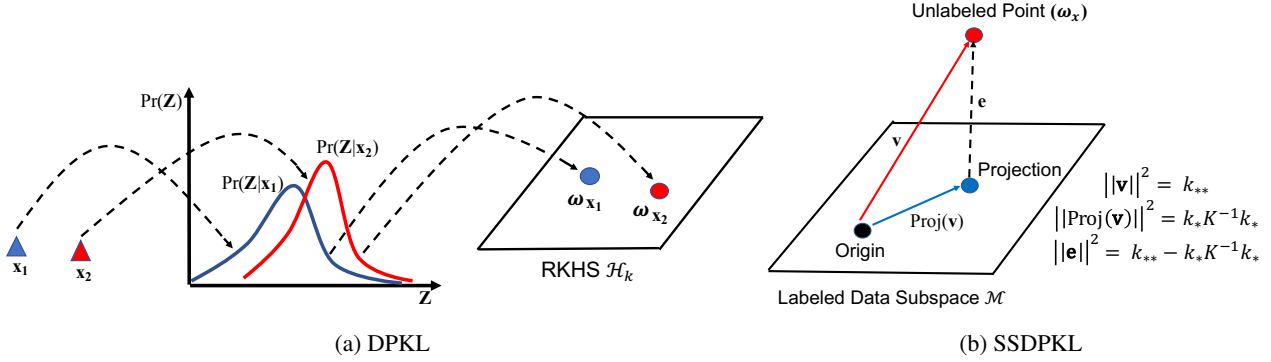
Figure 2: (a) Our model (DPKL) maps training points to probability distributions in the latent space and learns a GP whose kernel is the inner product between the RKHS embeddings of these distributions. (b) If unlabeled data is available we train a semi-supervised model (SSDPKL) which is a regularized version of DPKL. The regularizer is the average squared error between the RKHS norm of unlabeled points and their projections onto the subspace spanned by the labeled points.

Algorithm 1: Deep Probabilistic Kernel Learning

Input : Training points $\mathbf{X}$ and targets $\mathbf{y}$ along with a set of initial model parameters $\{\mathbf{w}_i^{(0)}\}_{i=1}^m$

Output : A set of model parameters $\{\mathbf{w}_i\}_{i=1}^m \sim \hat{p}(\mathbf{w})$ where $\hat{p}(\mathbf{w})$ is obtained by performing functional gradient descent on equation (8)

1 Initialize model parameters $\{\mathbf{w}_i^{(0)}\}_{i=1}^m \sim p_0(\mathbf{w})$ for some known $p_0(\mathbf{w})$

2 **for** *iteration t* **do**

3     $\mathbf{w}_i^{(t+1)} = \mathbf{w}_i^{(t)} - \epsilon_t \phi(\mathbf{w}_i^{(t)}),$

4   where

      $\phi(\mathbf{w}) = \sum_{l=1}^m \kappa(\mathbf{w}, \mathbf{w}_l)\nabla_{\mathbf{w}_l}\hat{L}(\mathbf{w}_1, \ldots, \mathbf{w}_m)$

5 **end**

---

d of $\mathcal{Z}$ is *much smaller* than the dimensionality D of $\mathcal{X}$.

Probabilistic Latent Space Mapping. Each data point $\mathbf{x}$ passes through a probabilistic neural network with parameters $\mathbf{W} \sim p(\mathbf{W})$ to give a random variable $\mathbf{Z} = g_\mathbf{W}(\mathbf{x}) \in \mathcal{Z}$. Thus a point $\mathbf{x}_i$ is represented by the latent distribution $p(\mathbf{Z}|\mathbf{x}_i)$ due to the stochasticity of $\mathbf{W}$.

GP Regression over Latent Distributions. We assume a GP prior $H \sim \mathcal{GP}(0, \mathbf{K})$ over target functions H which take *distributions* $p(\mathbf{Z}|\mathbf{x}_i)$ as input and output $\hat{y}_i$, an estimate of the true label y_i. $\mathbf{K}$ is the kernel matrix which models covariance between GP inputs (probability distributions). Given any two probability distributions $p(\mathbf{Z}|\mathbf{x}_i)$ and $p(\mathbf{Z}|\mathbf{x}_j)$, the corresponding entry of $\mathbf{K}$ is given by

$$K_{ij} = \mathbb{E}_{\mathbf{z} \sim p(\mathbf{Z}|\mathbf{x}_i),\, \mathbf{z}' \sim p(\mathbf{Z}|\mathbf{x}_j)}[k(\mathbf{z}, \mathbf{z}')] \quad (2)$$

Here k can be any standard kernel like the SE kernel for which the above expression would be

$$K_{ij} = \mathbb{E}_{\mathbf{z} \sim \mathrm{Pr}(\mathbf{Z}|\mathbf{x}_i),\, \mathbf{z}' \sim \mathrm{Pr}(\mathbf{Z}|\mathbf{x}_j)}[\exp(-\frac{1}{2}||\mathbf{z} - \mathbf{z}'||^2)] \quad (3)$$

This choice of kernel between probability distributions is inspired by the Set Kernel of Gärtner et al. [2002] for which consistency results and minimax rates were given in Szabó et al. [2016]. Previous works [Muandet et al., 2012, 2017] have used similar kernels to build predictive models, such as SVMs, with distribution inputs. K_{ij} corresponds to the inner product between distributions $p(\mathbf{Z}|\mathbf{x}_i)$ and $p(\mathbf{Z}|\mathbf{x}_j)$ in the Reproducing Kernel Hilbert Space (RKHS) $\mathcal{H}_k$ defined by the kernel k [Muandet et al., 2017] and thus $\mathbf{K}$ is positive definite. We expect that: a) The *lower dimensionality* of $\mathcal{Z}$ will improve the performance of simple kernels like SE in capturing the covariance (similarity) between points. b) *Probability distributions* in the latent space $\mathcal{Z}$ will better capture uncertainty due to scarcity of training data than point embeddings. Hence we perform GP regression over distributions in $\mathcal{Z}$.

Approximation with Random Fourier Features. Observe that in equation (2) the complexity of computing K_{ij} is $\mathcal{O}(m^2)$ (if we use m samples of $\mathbf{z}$ and $\mathbf{z}'$ to approximate the expectation). Since the total number of (i, j) pairs is n^2 the overall complexity will be too large ($\mathcal{O}(n^2 m^2)$). However, if k is shift-invariant, we can use the Random Fourier Features approximation [Rahimi and Recht, 2008] to approximate it as

$$k(\mathbf{z}, \mathbf{z}') \approx \mathbf{f}_\mathbf{V}(\mathbf{z})^T \mathbf{f}_\mathbf{V}(\mathbf{z}') \quad (4)$$

where $\mathbf{V}$ is a matrix whose rows $\mathbf{v}_1, \ldots, \mathbf{v}_R$ are i.i.d samples from the Fourier Transform of k and $\mathbf{f}_\mathbf{V}(\mathbf{z})$ is a vector with $2R$ elements whose l^{th} element is given by

$$\mathbf{f}_\mathbf{V}(\mathbf{z})_r = \begin{cases} \frac{1}{\sqrt{R}}\cos(\mathbf{v}_r^T\mathbf{z}), 1 \le l \le R \\ \frac{1}{\sqrt{R}}\sin(\mathbf{v}_{r-R}^T\mathbf{z}), R < r \le 2R \end{cases} \quad (5)$$

The overall kernel is now linear and the computation has complexity $\mathcal{O}(n^2R + nmR)$.

3.2 TRAINING ALGORITHM AND ITS DERIVATION

The forward pass described above, is used to compute the kernel between data points and consequently the predicted GP mean and variance for test data (see section 2). To train the model we need to find the optimal distribution $p(\mathbf{W})$ over its parameters in this setting.

Since the latent embedding $\mathbf{Z}$ for a data point $\mathbf{x}$ is given by $\mathbf{Z} = g_\mathbf{W}(\mathbf{x})$, $\mathbf{W} \sim p(\mathbf{W})$, we can rewrite equation (2) as

$$K_{ij} = \mathbb{E}_{\mathbf{w},\mathbf{w}' \sim p(\mathbf{W})}[k(g_\mathbf{w}(\mathbf{x}_i), g_{\mathbf{w}'}(\mathbf{x}_j))] = K_{ij}[p] \quad (6)$$

If Random Fourier Features are used then

$$K_{ij} \approx \mathbb{E}_{\mathbf{w} \sim p(\mathbf{W})}[\mathbf{f}_\mathbf{V}(g_\mathbf{w}(\mathbf{x}_i))]^T \mathbb{E}_{\mathbf{w}' \sim p(\mathbf{W})}[\mathbf{f}_\mathbf{V}(g_{\mathbf{w}'}(\mathbf{x}_j))]$$

In both cases we can view K_{ij} as a *functional* of $p(\mathbf{W})$. The overall data likelihood is then also a functional of p and the negative log likelihood is given by (see [Rasmussen, 2003])

$$L[p] = -\log \Pr(\mathbf{y}|\mathbf{X}, p(\mathbf{W})) \quad (7)$$
$$= \frac{1}{2}\mathbf{y}^T(\mathbf{K}[p] + \sigma^2\mathbf{I})^{-1}\mathbf{y} + \frac{1}{2}\log(\det(\mathbf{K}[p] + \sigma^2\mathbf{I}))$$

where $\det(\mathbf{A})$ denotes the determinant of a matrix $\mathbf{A}$.

Thus a *maximum likelihood* estimate, $p^*(\mathbf{W})$, of the distribution over model parameters is given by

$$p^*(\mathbf{W}) = \underset{p(\mathbf{W}) \in \mathcal{P}}{\arg\min} L[p] \quad (8)$$

where $\mathcal{P}$ is the class of distributions in which we seek p^*.

Since the negative log likelihood, $L[p]$, is a *functional* of the distribution p, we can use Functional Gradient Descent to search for distributions in $\mathcal{P}$ that minimize its value. However the choice of $\mathcal{P}$ is extremely critical to the success of this approach. Previous works in learning GP kernels using neural networks [Wilson et al., 2016, Jean et al., 2018] which seek a point estimate of the model parameters essentially assume that $\mathcal{P} = \{p(\mathbf{w})|p(\mathbf{w}) = p(\mathbf{w})\delta(\bar{\mathbf{w}})\}$, i.e., $\mathcal{P}$ is the space of Dirac Deltas centered around $\bar{\mathbf{w}}$. As we show in section 4, these approaches are not very effective in small data settings, possibly due to the restrictive nature of the search space. However, choosing an arbitrary search space might make it challenging to compute K_{ij} as in equation (6) due to the difficulty in computing expectations with respect to general high dimensional probability distributions.

Recent works [Liu and Wang, 2016], have shown that an effective middle-ground in such settings is to choose $\mathcal{P}$ as

$$\mathcal{P} = \{p(\mathbf{u})|\mathbf{u} = \mathbf{w} + s(\mathbf{w}), \mathbf{w} \sim p_0(\mathbf{w}), s \in \mathcal{H}_\kappa\} \quad (9)$$

where $\mathcal{H}_\kappa$ is a RKHS given by a kernel κ between model parameters $\mathbf{w}$ (note that this is *not* the RKHS $\mathcal{H}_k$ into which distributions in the latent space $\mathcal{Z}$ are embedded and which

is given by the kernel k). $\mathcal{P}$ includes all smooth transformations from the initial distribution p_0. Distributions in this set $\mathcal{P}$ can closely approximate almost any distribution, particularly those admitting Lipschitz continuous densities [Villani, 2008].

Moreover, observe that for any smooth one-to-one transform $\mathbf{u} = T(\mathbf{w})$, $\mathbf{w} \sim p(\mathbf{W})$, the entries of $\mathbf{K}$ under the transformed distribution $p_{[T]}(\mathbf{u})$ can be written as

$$K_{ij} = \mathbb{E}_{\mathbf{u},\mathbf{u}' \sim p_{[T]}(\mathbf{u})}[k(g_\mathbf{u}(\mathbf{x}_i), g_{\mathbf{u}'}(\mathbf{x}_j))] \quad (10)$$
$$= \mathbb{E}_{\mathbf{w},\mathbf{w}' \sim p(\mathbf{W})}[k(g_{T(\mathbf{w})}(\mathbf{x}_i), g_{T(\mathbf{w}')}(\mathbf{x}_j))]. \quad (11)$$

Since the above holds for infinitesimal shifts $\mathbf{u} = \mathbf{w} + s(\mathbf{w})$, a tractable choice of p_0 (For eg. Gaussian), enables efficient approximation of K_{ij} by sample averages with samples $\mathbf{u}_i$, $\mathbf{u}_i = \mathbf{w}_i + s(\mathbf{w}_i), \mathbf{w}_i \sim p_0(\mathbf{w})$. Thus $\mathcal{P}$ is sufficiently large but permits efficient computation.

Moreover K_{ij} in equation (11) is a *functional* of the transformation T (for fixed $p(\mathbf{W})$) i.e. a functional of the shift s in our case. Therefore, the problem of finding $p^*(\mathbf{W})$ in equation (8) reduces to the problem of finding the optimal shift (given $p_0(\mathbf{W})$) i.e.

$$s^*(\mathbf{W}) = \underset{s(\mathbf{W})}{\arg\min} L[s]. \quad (12)$$

Since $s \in \mathcal{H}_\kappa$, ($\mathcal{H}_\kappa$ is the RKHS for the kernel κ), we can solve equation (12) via functional gradient descent. The following is an empirical estimate of the functional gradient of the negative log-likelihood at $s = 0$.

Proposition 1. *If we draw m realizations of model parameters $\mathbf{w}_1, \ldots, \mathbf{w}_m \sim p(\mathbf{w})$, $p \in \mathcal{P}$, the functional gradient of the negative log likelihood can be approximated as*

$$\nabla_s L \mid_{s=0} \simeq \sum_{l=1}^{m} \kappa(\mathbf{w}_l, .)\nabla_{\mathbf{w}_l}\hat{L}(\mathbf{w}_1, \ldots, \mathbf{w}_m) \quad (13)$$

where, $\hat{L} = \frac{1}{2}\mathbf{y}^T(\hat{\mathbf{K}} + \sigma^2\mathbf{I})^{-1}\mathbf{y} + \frac{1}{2}\log(det(\hat{\mathbf{K}} + \sigma^2\mathbf{I}))$

is the empirical negative log likelihood, κ is the kernel between model parameters, $\hat{K}_{ij} = \frac{1}{m^2}\sum_{l,l'=1}^{m} k_{ij}(\mathbf{w}_l, \mathbf{w}_{l'})$ are the entries of the empirical kernel matrix $\hat{\mathbf{K}}$.

To estimate the optimal shift, s^* (corresponding to the optimal distribution p^*) we draw an initial set of samples $\mathbf{w}_1, \ldots, \mathbf{w}_m \sim p_0(\mathbf{w})$ and iteratively apply the functional gradient descent transformation $\mathbf{u} = \mathbf{w} - \epsilon\nabla_s L \mid_{s=0}$. Since these transformed weights correspond to samples from the transformed distribution $p_{[T]}(\mathbf{u})$ we only need to evaluate the gradient at $s = 0$, which is a crucial advantage since the expression for the gradient at non-zero shifts is much more complicated. The functional gradient in equation (13) is a weighted average of individual sample gradients with the kernel between model parameters κ controlling the effect that different samples have on each other. For $m = 1$ it

reduces to gradient descent on negative log likelihood as in Deep Kernel Learning [Wilson et al., 2016]. Algorithm 1 summarises the training procedure for our model, DPKL.

Bias in Gradient Estimates. We note that the gradient estimates in equation (13) are biased as the expectation with respect to weights $\mathbf{w}$ appears inside the matrix inversion in equation (7). While DPKL already outperforms the current state-of-the-art over a wide range of experiments, future work on reducing this bias may give further improvements (as in [Belghazi et al., 2018]).

Connection with SVGD. While our choice of search space $\mathcal{P}$ follows that in Stein's Variational Gradient Descent (SVGD) [Liu and Wang, 2016] the goal of our work is quite different from theirs. SVGD is a general purpose *Bayesian Inference* algorithm which estimates posterior distributions by minimizing KL divergence w.r.t the true posterior, and has no particular connection to sample efficiency or representation learning, while we seek to improve the quality of learned latent representations and kernels in small data settings by performing *Maximum Likelihood* estimation over the space of probability distributions $\mathcal{P}$. The different loss functions also lead to different functional gradient estimates.

3.3 EXTENSIONS

Semi-Supervised Learning (SSDPKL). While our base model, DPKL, only uses labeled data, it can be augmented with unlabeled data using the regularizer of [Jean et al., 2018] to give a semi-supervised model which we call SSDPKL. Assuming the dataset $\mathbf{X}$ has n_l labeled points $\mathbf{X}_L$ (with labels $\mathbf{y}_L$) and n_u unlabeled points $\mathbf{X}_U$, the new optimization problem is

$$p^*(\mathbf{W}) = \underset{p(\mathbf{W}) \in \mathcal{P}}{\arg\min} \frac{1}{n_l} L[p] + \frac{\alpha}{n_u} \sum_{\mathbf{x} \in \mathbf{X}_U} \sigma^2(\mathbf{x}). \quad (14)$$

Here $\sigma^2(\mathbf{x})$ is the GP posterior variance (see section 2) and α is the regularization parameter. Since [Jean et al., 2018] also restrict $\mathcal{P}$ to Dirac Delta functions, our choice of $\mathcal{P}$ in equation (9) can generalize this in the same way as it generalizes Deep Kernel Learning. Training involves using functional gradient descent. The following result (illustrated in figure 2b) explains the rationale behind using this regularized loss function in our model.

Proposition 2. *For any unlabeled point, $\mathbf{x} \in \mathbf{X}_U$, if $\omega_{\mathbf{x}}$ is the embedding in the RKHS $\mathcal{H}_k$, of $p(\mathbf{Z}|\mathbf{x})$ and $\omega_{\mathbf{x}}^L$, is its orthogonal projection onto the subspace $\mathcal{H}_k^L \subseteq \mathcal{H}_k$, spanned by RKHS embeddings of $p(\mathbf{Z}|\mathbf{x}_i), i = 1, \ldots, n_l, \mathbf{x}_i \in \mathbf{X}_L$ then, assuming the GP kernel matrix $\mathbf{K}$ is invertible, the posterior variance of $\mathbf{x}$ is given by $\sigma^2(\mathbf{x}) = ||\omega_{\mathbf{x}} - \omega_{\mathbf{x}}^L||_{\mathcal{H}_k}^2$.*

Thus minimizing posterior variance encourages the model to learn a mapping that spans distributions in $\mathcal{Z}$ corresponding to unlabeled data, which can improve generalization.

Classification. The connection between Gaussian Processes and Bayesian Linear Regression (see [Rasmussen, 2003]) motivates us to consider Bayesian Logistic Regression (BLR) as our base model for classification. Assuming there are C classes, the probability of a data point $\mathbf{x}_i$ belonging to class c under this model is given by

$$\rho_{ic} = \frac{\exp(\theta_c^T \mathbf{x}_i)}{\sum_{c'=1}^{C} \exp(\theta_{c'}^T \mathbf{x}_i)}, \quad \theta_c \sim q(\theta) \quad (15)$$

The model can be trained using any Bayesian Inference algorithm to find the optimal posterior distribution over the weights θ_c for each class. When $||\theta_c||_2 = ||\mathbf{x}_i||_2 = 1$, $\exp(\theta_c^T \mathbf{x}_i) \propto \exp(-\frac{1}{2}||\mathbf{x} - \theta_c||^2)$ and equation (15) can be rewritten as

$$\rho_{ic} = \frac{k(\mathbf{x}_i, \theta_c)}{\sum_{c'=1}^{C} k(\mathbf{x}_i, \theta_{c'})}, \quad \theta_c \sim q(\theta) \quad (16)$$

where k is the SE kernel. Previous works [Chen et al., 2019] have shown that such normalization improves accuracy in small-data (few-shot) classification. This motivates us to replace the high-dimensional data points $\mathbf{x}_i$ with low-dimensional *probabilistic embeddings* $\mathbf{Z} = g_{\mathbf{W}}(\mathbf{x})$, $\mathbf{W} \sim p(\mathbf{W})$ analogous to DPKL, due to which ρ_{ic} can be written in terms of the corresponding probabilistic kernel as

$$\rho_{ic} = \frac{\mathbb{E}_{\mathbf{w} \sim p(\mathbf{W})}[k(g_{\mathbf{w}}(\mathbf{x}_i), \theta_c)]}{\sum_{c'=1}^{C} \mathbb{E}_{\mathbf{w} \sim p(\mathbf{W})}[k(g_{\mathbf{W}}(\mathbf{x}_i), \theta_{c'})]}, \quad \theta_c \sim q(\theta)$$

To reduce the computational cost the kernel k can be approximated by the corresponding Random Fourier Features as in Section 3.1. Now in addition to Bayesian Inference for learning the posterior over θ_c we also learn the optimal distribution over model parameters $p^*(\mathbf{W}) \in \mathcal{P}$ via the Maximum Likelihood estimation approach of Algorithm 1. The negative log likelihood in this case is given by

$$L = \sum_{i=1}^{n} \sum_{c=1}^{C} y_{ic} \log \mathbb{E}[\rho_{ic}] \quad (17)$$

The expectation in equation (17) is approximated using samples drawn from the current estimate of the posterior of θ_c. Experiments in Section 4 show that this model outperforms BLR and is competitive with the state-of-the-art in few-shot classification.

4 EXPERIMENTAL RESULTS

We apply our model DPKL, to various classification and regression tasks in the small data regime. All models are implemented in TensorFlow [Abadi et al., 2016]. We use the SE kernel everywhere as the kernel κ between DPKL model parameters $\mathbf{w}$ with bandwidth chosen according to the median heuristic described in [Liu and Wang, 2016].

(a) CTSlice ($D = 384$)   (b) Electric ($D = 6$)   (c) Parkinsons ($D = 20$)

(d) Skillcraft ($D = 18$)   (e) Buzz ($D = 77$)   (f) Elevators ($D = 18$)
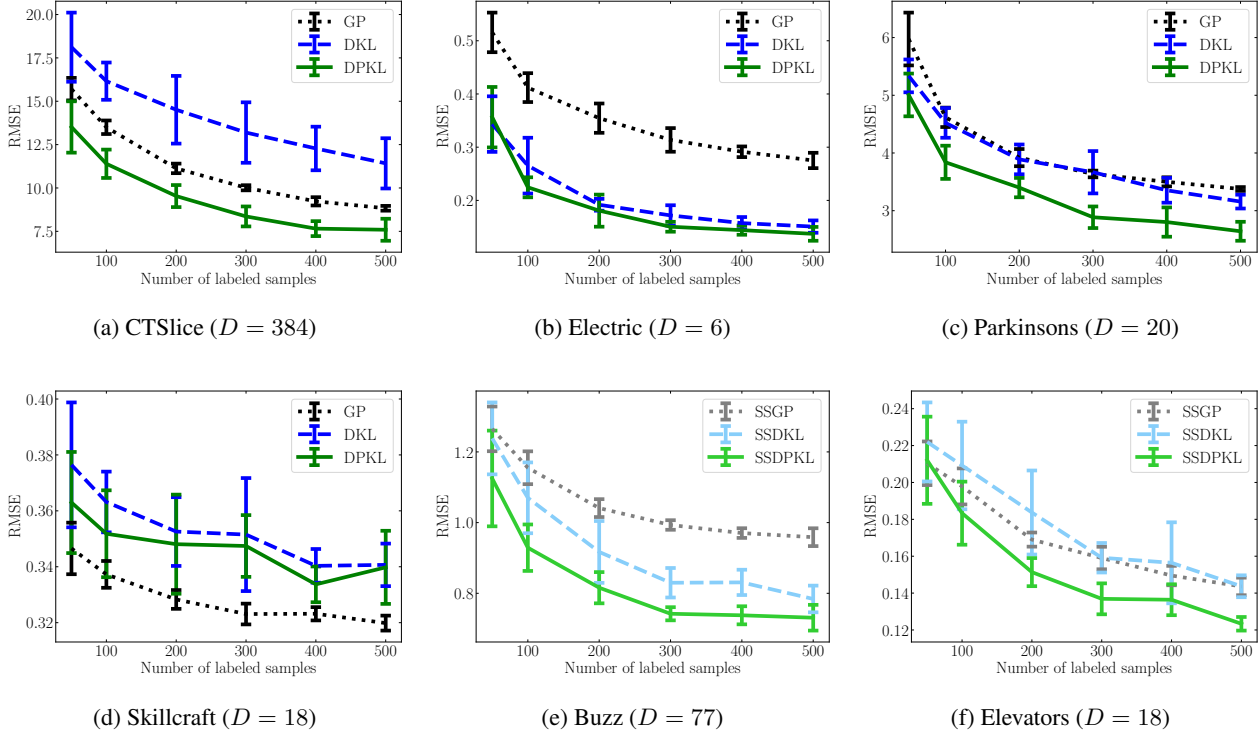
Figure 3: RMSE Results for regression on 6 UCI Datasets with $n = \{50, 100, 200, 300, 400, 500\}$ labeled samples. Here show results for DPKL v/s DKL [Wilson et al., 2016] on 4 datasets and the semi-supervised models SSDPKL v/s SSDKL on the other 2 datasets. The clear superiority of DPKL and SSDPKL over their deterministic counterparts (DKL and SSDKL) confirms the efficacy of our approach. See Appendix B for the remaining results.

UCI Regression. We use DPKL, and its semi-supervised variant, SSDPKL for regression on 6 datasets of varying dimensionality from the UCI repository [Lichman et al., 2013] as shown in Figure 3. We compare our models to DKL [Wilson et al., 2016] and SSDKL [Jean et al., 2018] (both of which use deterministic neural networks to map data into the latent space), and GP and Semi-Superivsed GP (SSGP) (regularized as in Equation (14)). The GP and SSGP models use a Squared Exponential (SE) kernel directly on the data i.e. $k(\mathbf{x}_i, \mathbf{x}_j) = \exp(-\frac{1}{2} \sum_l \frac{1}{h_l^2} (\mathbf{x}_{il} - \mathbf{x}_{jl})^2)$. The DKL and SSDKL models embed the data into a low-dimensional latent space before using a SE kernel i.e. $k(\mathbf{x}_i, \mathbf{x}_j) = \exp(-\frac{1}{2} \|g_{\mathbf{w}}(\mathbf{x}_i) - g_{\mathbf{w}}(\mathbf{x}_j)\|^2)$. The DPKL and SSDPKL models embed the data into a *probability distribution* in the latent space (as described in Section 3) and then use the SE kernel between distributions as given in Equation (3). We used the same neural network architecture as [Jean et al., 2018] ($D - 100 - 50 - 50 - d$) for mapping data points to the latent space in the DKL and DPKL models. Here D is the dimensionality of the datasets and $d = 2$ is the dimensionality of the latent embedding $\mathbf{z} \in \mathcal{Z}$. In DPKL and SSDPKL, we used $m = 10$ samples from this architecture to represent the distribution over model parameters. The lengthscales for the GP SE kernel,

and the neural network weights in DKL are optimized using gradient descent on the negative log-likelihood while the *distribution* over neural network weights in DPKL is optimized using functional gradient descent (Algorithm 1). For each dataset we use $n = \{50, 100, 200, 300, 400, 500\}$ labeled examples. The semi-supervised models – SSDPKL, SSDKL and SSGP, additionally have access to (at most) 10000 unlabeled examples from the corresponding datasets while the supervised models, DPKL, DKL, and GP, do not use any unlabeled data. More details on training procedure are in Appendix C.

Figure 3 shows RMSE for supervised models on 4 datasets and for semi-supervised models on the other 2 datasets. Clearly DPKL (SSDPKL) improves over DKL (SSDKL) across *all* datasets. While GP does a bit better than DPKL on one of the lower dimensional datasets (Skillcraft), we note that DPKL is significantly better than GP on all other datasets, including the high dimensional datasets CTSlice ($D = 384$) and Buzz ($D = 77$), thus clearly illustrating the advantages of our approach in high dimensional settings.

An added benefit of our approach is better uncertainty quantification. Since all Gaussian Process models output a variance in addition to the mean prediction we use the average negative log likelihood of test data as in [Lakshminarayanan

(a) CTSlice ($D = 384$) (b) Electric ($D = 6$) (c) Parkinsons ($D = 20$)

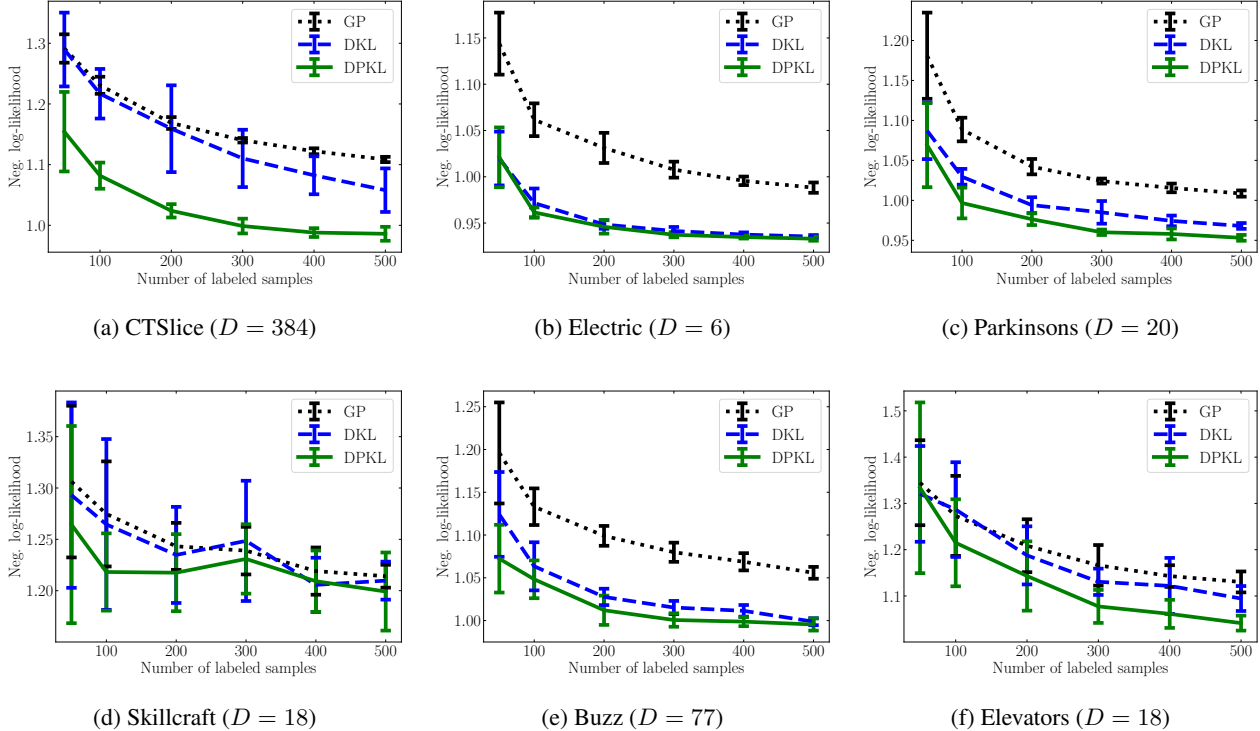(d) Skillcraft ($D = 18$) (e) Buzz ($D = 77$) (f) Elevators ($D = 18$)

Figure 4: Uncertainty Quantification (average negative log-likelihood of test data) results for regression on 6 UCI Datasets with $n = \{50, 100, 200, 300, 400, 500\}$ labeled samples. for DPKL, DKL and GP. Lower negative log-likelihood indicates better uncertainty quantification. DPKL generally has lower negative log-likelihood than GP and DKL.

et al., 2017] to quantify the uncertainty of model predictions. A lower negative log likelihood implies that the model fits the test data better. The results in Figure 4 shows that by projecting high dimensional data to low dimensional probability distributions before making predictions, DPKL consistently outperforms DKL and GP in this metric across all datasets.

Few-shot Classification. We apply DPKL to the 5-shot classification task described in [Chen et al., 2019]. Here a pre-trained base model with a deterministic feature extractor (4-layer convolution backbone) attached to a classifier (any suitable classifier), is fine-tuned on a small number of examples (5 per class) of classes that were *unseen* during training. Accuracy is measured on a test dataset of examples from the *unseen classes*. We keep the backbone (feature extractor) fixed and change the classifier in our experiments. We compare the DPKL Classifier (described in Section 3.3) to the Baseline++ with 1-NN classifier described in Appendix A4 of [Chen et al., 2019] since it is the deterministic version (deterministic embedding, deterministic logistic regression) of the DPKL classification approach, and also to a classifier consisting of a deterministic embedding layer followed by Bayesian Logistic Regression (similar to [Snoek et al., 2015]) which we call DBLR. Results in Table 1 show that DPKL outperforms Baseline++ and DBLR for 5-shot classification on both the CUB and mini-Imagenet datasets which

Method	CUB	mini-Imagenet
Baseline++	68.94 ± 0.74	61.93 ± 0.65
DBLR	68.48 ± 0.71	60.12 ± 0.63
DPKL (Ours)	$\mathbf{69.11 \pm 0.72}$	$\mathbf{64.96 \pm 0.62}$

Table 1: DPKL has higher accuracy than the Baseline++ method of [Chen et al., 2019] as well as DBLR for 5-shot classification on both CUB and mini-Imagenet datasets.

were considered in [Chen et al., 2019]. DPKL outperforming DBLR indicates that improvements are not obtained just by replacing deterministic logistic regression with Bayesian logistic regression but by *probabilistic embeddings* learned in DPKL. More details on training procedure are in Appendix C and additional classification experiments are presented in Appendix B.

5 CONCLUSION

We propose a new approach for small data learning that maps high dimensional data to low dimensional probability distributions and then performs regression/classification on these distributions. The distribution over model parameters

is learned via functional gradient descent. Our model outperforms several baselines in GP regression and few-shot classification while learning a meaningful representation of the data and accurately quantifying uncertainties on test data. In future we plan to theoretically analyze the convergence of our approach, seek potential improvements through optimal kernel selection and unbiased gradient estimation, and apply our model to areas like Bayesian Optimization [Brochu et al., 2010] where GPs have been successful in the past.

Acknowledgements

References

Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. Tensorflow: A system for large-scale machine learning. In *12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16)*, pages 265–283, 2016.

Maruan Al-Shedivat, Andrew Gordon Wilson, Yunus Saatchi, Zhiting Hu, and Eric P Xing. Learning scalable deep kernels with recurrent structure. *arXiv preprint arXiv:1610.08936*, 2016.

Christophe Andrieu, Nando De Freitas, Arnaud Doucet, and Michael I Jordan. An introduction to mcmc for machine learning. *Machine learning*, 50(1-2):5–43, 2003.

François Bachoc, Fabrice Gamboa, Jean-Michel Loubes, and Nil Venet. A gaussian process regression model for distribution inputs. *IEEE Transactions on Information Theory*, 2017.

Francois Bachoc, Alexandra Suvorikova, David Ginsbourger, Jean-Michel Loubes, and Vladimir Spokoiny. Gaussian processes with multidimensional distribution inputs via optimal transport and hilbertian embedding. *arXiv preprint arXiv:1805.00753*, 2018.

Mohamed Ishmael Belghazi, Aristide Baratin, Sai Rajeshwar, Sherjil Ozair, Yoshua Bengio, Devon Hjelm, and Aaron Courville. Mutual information neural estimation. In *International Conference on Machine Learning*, pages 530–539, 2018.

David M Blei, Alp Kucukelbir, and Jon D McAuliffe. Variational inference: A review for statisticians. *Journal of the American Statistical Association*, 112(518):859–877, 2017.

Eric Brochu, Vlad M Cora, and Nando De Freitas. A tutorial on bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning. *arXiv preprint arXiv:1012.2599*, 2010.

Saikiran Bulusu, Bhavya Kailkhura, Bo Li, Pramod K Varshney, and Dawn Song. Anomalous example detection in deep learning: A survey. *IEEE Access*, 8:132330–132347, 2020.

Wei-Yu Chen, Yen-Cheng Liu, Zsolt Kira, Yu-Chiang Frank Wang, and Jia-Bin Huang. A closer look at few-shot classification, 2019.

Andreas Damianou and Neil Lawrence. Deep gaussian processes. In *Artificial Intelligence and Statistics*, pages 207–215, 2013.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

Yarin Gal. Uncertainty in deep learning. *University of Cambridge*, 2016.

Roman Garnett, Michael A Osborne, and Philipp Hennig. Active learning of linear embeddings for gaussian processes. *arXiv preprint arXiv:1310.6740*, 2013.

Thomas Gärtner, Peter A Flach, Adam Kowalczyk, and Alexander J Smola. Multi-instance kernels. In *ICML*, volume 2, page 7, 2002.

Franz Graf, Hans-Peter Kriegel, Matthias Schubert, Sebastian Pölsterl, and Alexander Cavallaro. 2d image registration in ct images using radial image descriptors. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 607–614. Springer, 2011.

Alex Graves. Practical variational inference for neural networks. In *Advances in neural information processing systems*, pages 2348–2356, 2011.

Alex Graves, Abdel-rahman Mohamed, and Geoffrey Hinton. Speech recognition with deep recurrent neural networks. In *2013 IEEE international conference on acoustics, speech and signal processing*, pages 6645–6649. IEEE, 2013.

José Miguel Hernández-Lobato and Ryan Adams. Probabilistic backpropagation for scalable learning of bayesian neural networks. In *International Conference on Machine Learning*, pages 1861–1869, 2015.

TS Jaakkola and MI Jordan. Bayesian logistic regression: a variational approach. In *Proceedings of the 1997 Conference on Artificial Intelligence and Statistics, Ft. Lauderdale, FL*, 1997.

Neal Jean, Marshall Burke, Michael Xie, W Matthew Davis, David B Lobell, and Stefano Ermon. Combining satellite imagery and machine learning to predict poverty. *Science*, 353(6301):790–794, 2016.

Neal Jean, Sang Michael Xie, and Stefano Ermon. Semi-supervised deep kernel learning: Regression with unlabeled data by minimizing predictive variance. In *Advances in Neural Information Processing Systems*, pages 5322–5333, 2018.

Diederik P Kingma, Shakir Mohamed, Danilo Jimenez Rezende, and Max Welling. Semi-supervised learning with deep generative models. In *Advances in Neural Information Processing Systems*, pages 3581–3589, 2014.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

Samuli Laine and Timo Aila. Temporal ensembling for semi-supervised learning. *arXiv preprint arXiv:1610.02242*, 2016.

Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In *Advances in Neural Information Processing Systems*, pages 6402–6413, 2017.

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

Qiang Liu and Dilin Wang. Stein variational gradient descent: A general purpose bayesian inference algorithm. In *Advances In Neural Information Processing Systems*, pages 2378–2386, 2016.

Krikamol Muandet, Kenji Fukumizu, Francesco Dinuzzo, and Bernhard Schölkopf. Learning from distributions via support measure machines. In *Advances in neural information processing systems*, pages 10–18, 2012.

Krikamol Muandet, Kenji Fukumizu, Bharath Sriperumbudur, Bernhard Schölkopf, et al. Kernel mean embedding of distributions: A review and beyond. *Foundations and Trends® in Machine Learning*, 10(1-2):1–141, 2017.

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

Augustus Odena. Semi-supervised learning with generative adversarial networks. *arXiv preprint arXiv:1606.01583*, 2016.

Barak Oshri, Annie Hu, Peter Adelson, Xiao Chen, Pascaline Dupas, Jeremy Weinstein, Marshall Burke, David Lobell, and Stefano Ermon. Infrastructure quality assessment in africa using satellite imagery and deep learning. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 616–625. ACM, 2018.

Barnabás Póczos, Aarti Singh, Alessandro Rinaldo, and Larry Wasserman. Distribution-free distribution regression. In *Artificial Intelligence and Statistics*, pages 507–515. PMLR, 2013.

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

Santu Rana, Cheng Li, Sunil Gupta, Vu Nguyen, and Svetha Venkatesh. High dimensional bayesian optimization with elastic gaussian process. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 2883–2891. JMLR. org, 2017.

Carl Edward Rasmussen. Gaussian processes in machine learning. In *Summer School on Machine Learning*, pages 63–71. Springer, 2003.

Hugh Salimbeni and Marc Deisenroth. Doubly stochastic variational inference for deep gaussian processes. In *Advances in Neural Information Processing Systems*, pages 4588–4599, 2017.

Jasper Snoek, Oren Rippel, Kevin Swersky, Ryan Kiros, Nadathur Satish, Narayanan Sundaram, Mostofa Patwary, Mr Prabhat, and Ryan Adams. Scalable bayesian optimization using deep neural networks. In *International Conference on Machine Learning*, pages 2171–2180, 2015.

Zoltán Szabó, Bharath K Sriperumbudur, Barnabás Póczos, and Arthur Gretton. Learning theory for distribution regression. *The Journal of Machine Learning Research*, 17(1):5272–5311, 2016.

Antti Tarvainen and Harri Valpola. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In *Advances in neural information processing systems*, pages 1195–1204, 2017.

Rohit Tripathy, Ilias Bilionis, and Marcial Gonzalez. Gaussian processes with built-in dimensionality reduction: Applications to high-dimensional uncertainty propagation. *Journal of Computational Physics*, 321:191–223, 2016.

Cédric Villani. *Optimal transport: old and new*, volume 338. Springer Science & Business Media, 2008.

Andrew Wilson and Hannes Nickisch. Kernel interpolation for scalable structured gaussian processes (kiss-gp). In *International Conference on Machine Learning*, pages 1775–1784, 2015.

Andrew Gordon Wilson and Pavel Izmailov. Bayesian deep learning and a probabilistic perspective of generalization. *arXiv preprint arXiv:2002.08791*, 2020.

Andrew Gordon Wilson, Zhiting Hu, Ruslan Salakhutdinov, and Eric P Xing. Deep kernel learning. In *Artificial Intelligence and Statistics*, pages 370–378, 2016.

Jize Zhang, Bhavya Kailkhura, and T Han. Leveraging uncertainty from deep learning for trustworthy materials discovery workflows. *arXiv preprint arXiv:2012.01478*, 2020.

Xiaojin Zhu and Andrew B Goldberg. Introduction to semi-supervised learning. *Synthesis lectures on artificial intelligence and machine learning*, 3(1):1–130, 2009.