An Interpretable and Sample Efficient Deep Kernel for Gaussian Process

Yijue Dai[1], **Tianjian Zhang**[1], **Zhidi Lin**[1], **Feng Yin**[*1], **Sergios Theodoridis**[1,2], **Shuguang Cui**[1]
[1]The Chinese University of Hong Kong (Shenzhen) and SRIBD, 518172, China.
[2]Dept. of Informatics and Telecommunications, National and Kapodistrian University of Athens, 15772, Greece
[*]Correspondence author: yinfeng@cuhk.edu.cn

Abstract

We propose a novel Gaussian process kernel that takes advantage of a deep neural network (DNN) structure but retains good interpretability. The resulting kernel is capable of addressing four major issues of the previous works of similar art, i.e., the optimality, explainability, model complexity, and sample efficiency. Our kernel design procedure comprises three steps: (1) Derivation of an optimal kernel with a non-stationary dot product structure that minimizes the prediction/test mean-squared-error (MSE); (2) Decomposition of this optimal kernel as a linear combination of shallow DNN subnetworks with the aid of multi-way feature interaction detection; (3) Updating the hyper-parameters of the subnetworks via an alternating rationale until convergence. The designed kernel does not sacrifice interpretability for optimality. On the contrary, each subnetwork explicitly demonstrates the interaction of a set of features in a transformation function, leading to a solid path toward explainable kernel learning. We test the proposed kernel with both synthesized and real-world data sets, and the proposed kernel is superior to its competitors in terms of prediction performance in most cases. Moreover, it tends to maintain the prediction performance and be robust to data over-fitting issue, when reducing the number of samples.

1 INTRODUCTION

Over the recent years, Bayesian deep learning techniques are becoming popular due to the ever-increasing interests in learning with uncertainties, learning with small (non-stationary) data, and continual learning, etc (Khan, 2019; Salimbeni et al., 2019). As a representative, Gaussian process (GP) models for machine learning constitute a class of important Bayesian non-parametric models that are tightly linked with the support vector machines (SVM) and deep neural network (DNN) among other salient machine learning models (Williams and Rasmussen, 2006). Given a finite set of data samples and a GP prior, the desired targets/outputs are then represented via Bayes rule in the form of posterior (multivariate) Gaussian distribution. In contrast to a single point estimate given by a deterministic model such as the widely used DNN, GP models also provide an uncertainty bound that is valuable for critical decision-making. GP models are also simple in terms of mathematical formulation and tractable in terms of statistical inference, therefore they have found a plethora of applications in the past decades.

Two major technical issues prohibit the wider use of GP models. The first issue is the scalability due to their $O(n^3)$ computational complexity for the model training, while the second issue lies in the optimal kernel design. Scalable GP models can be obtained, for instance, by exploring: 1) the local structures of the kernel matrix (Ambikasaran et al., 2015); 2) the state-space model reformulation and Kalman filter (Sarkka et al., 2013); 3) the Bayesian committee machine (BCM) using a number of distributed computing units (Deisenroth and Ng, 2015); 4) the variational Bayesian formulation (Titsias, 2009); and 5) the iterative methods (Dong et al., 2017; Ubaru et al., 2017; Gardner et al., 2018). A complete survey of the existing scalable GP models can be found in (Liu et al., 2020).

In this paper, we solely focus on the second technical issue, namely the optimal kernel design. It is well known that a good kernel function is capable of lifting raw features to a high-dimensional space, where regression and classification can be done more effectively. In order to meet the challenges brought by kernel selection, there has been a substantial body of literature exploring automatic or optimal kernel learning for GP models. Generally speaking, the existing optimal kernel learning approaches can be broadly divided into three categories, including: (1) multiple kernel learning (Chen et al., 2012); (2) spectral kernel learning (Quiñonero-Candela et al., 2010; Wilson and Adams, 2013); (3) deep kernel learning (DKL) (Wilson et al., 2016a; Arora et al., 2019).

The idea behind the multiple kernel learning is to select a linear or nonlinear combination of primitive kernels via a specific optimization method with the goal to let data determine the best kernel configuration. For instance, a linear combination of the Matern kernel, the squared-exponential (SE) kernel and the periodic kernel was applied to CO_2 concentration prediction, as shown in the eq. (5.19) of (Williams and Rasmussen, 2006). However, the main drawback lies in that the primitive kernels are often selected subjectively and combined with ad-hoc weights. The spectral kernel learning is built around the idea of approximating the spectral density of a desired stationary kernel by a mixture of basis functions, such as Dirac functions in (Quiñonero-Candela et al., 2010) or Gaussian basis functions in (Wilson and Adams, 2013). The DKL approaches received more attention due to the outstanding prediction performance, and they can be further divided into two classes. The first class of approaches proposed to embed neural network (NN) structures into the state-of-the-art GP kernels, representative works include (Wilson et al., 2016a,b; Al-Shedivat et al., 2017). This class of deep kernels is capable to learn unstructured real data set and verified to be effective in various application sectors, including but not limited to industrial polymerization processes, crop yield prediction, image annotation, and visible light communication. The second class of deep kernels was designed while linking the GP models with deep neural networks (DNNs) for studying the learning dynamics of the latter. Representative deep kernels include the arccosine kernel (Cho and Saul, 2009), neural tangent kernel (NTK) (Jacot et al., 2018), and the convolutional neural tangent kernel (CNTK) (Arora et al., 2019). The major problems with the most recent NTK and CNTK kernels lie in the recursive evaluation of the kernel as well as the ideal assumptions made on the DNNs that all together make these kernels still less competent than the corresponding DNNs with the best setup found so far.

In this paper, we follow the basic idea of the first class of deep kernels to develop a new member. The reason for choosing this class of kernels is primarily due to their powerful kernel expressiveness and the resulting superior prediction performance reported from various different application sectors. However, some drawbacks are prominent in the existing works (Wilson et al., 2016a,b; Al-Shedivat et al., 2017). The first one is the loss of kernel interpretability since the embedded DNN is lack of interpretability. Secondly, the existing works require a large number of data samples to efficiently train a fully-connected over-parameterized DNN embedded in an elementary GP kernel, for instance the SE kernel; otherwise, data-overfitting can be perceived for small data cases. To maintain the good data-fitting performance, while alleviating the aforementioned drawbacks, we propose to design an optimal kernel, in which the NN structure is decomposed into a linear combination of shallow subnetworks with the aid of feature interaction detection, which is deemed as a research frontier towards explainable AI. Our contributions of this work include:

- Derivation of a non-stationary optimal kernel function that minimizes the test mean-squared-error (MSE). With the given theorem on the optimality, we argue that an extra elementary GP kernel may be redundant, and by avoiding it, improved numerical stability can be obtained and the data-overfitting problem alleviated.
- Implementation of the derived optimal kernel through decomposing a fully-connected over-parameterized DNN into a linear combination of shallow subnetworks, forming a generative additive model with significantly reduced (more than 85 percent) total number of model parameters. A small batch of model parameters for each subnetwork can potentially be tuned alternatively, allowing for better usage of the computation resources.
- Enhanced prediction accuracy and robustness can be harvested for small data cases due to the improved kernel interpretability and well reduced model parameters.

The remainder of this paper is organized as follows. In section 2, we firstly introduce some representative related works. In section 3, we briefly go through the background of Gaussian process regression (GPR) and the first class of DKL approaches. In section 4, we first introduce an optimal kernel and further implement it to be a better interpretable and sample efficient deep kernel with the aid of feature interaction detection. Section 5 presents some experimental results, which confirm that our proposed kernels outperform various competing kernels on a variety of data sets. Finally, conclusions are made in Section 6.

2 RELATED WORK

As our optimal yet interpretable kernel is a deep kernel with the aid of feature interaction detection, the most related DKL approaches and interaction detection methods are surveyed in this section.

Deep kernel learning: The idea of the DKL is to place a DNN as the front end of a basic kernel to extract low dimensional embeddings (Wilson et al., 2016a,b). A modified kernel with more expressive embeddings and a more efficient learning structure has been proposed recently, which uses the finite rank Mercer kernel function with mutually orthogonal embeddings (Dasgupta et al., 2018). However, such kind of kernel learning with embeddings from NN structures requires supervised learning with a large number of labeled data for accurate prediction. As the labeled data are always insufficient in many real cases, a semi-supervised DKL has been proposed, which incorporates information from unlabeled data and learning by simultaneously minimizing the negative log marginal likelihood of labeled data and the posterior variance of unlabeled data. However, by directly incorporating the non-transparent DNN into GP models loses the model's explainability totally. Thus more attention should be paid to designing interpretable DKL.

Interaction Detection: Interaction detection has attracted a lot of attention these years, owing to its ability to enhance the model interpretability. In (Lou et al., 2013), the authors proposed to test all interaction pairs in a greedy manner to build *Generalized Additive Models plus Interactions* (GA^{2}M). Although GA^{2}M is transparent and interpretable, it is time consuming. There are other works that try to extract interactions from a trained model. For instance, for a tree-based model, there are works such as iterative Random Forest (iRF), Disentangled Attribution Curves (DAC), etc; for neural networks (NNs), (Tsang et al., 2017) proposed an algorithm called Neural Interaction Detection (NID) by training a sparse ReLU network with L_1 regularization and extracted the interactions by analyzing the weights in the hidden layers. Since NID can produce comparably better quality outcomes, in this paper, we will use it to obtain feature interactions.

3 PRELIMINARIES

In this section, we briefly review the GPR and the DKL, which incorporates NN structures into the state-of-the-art GP kernels. This section serves as the foundation of section 4 for our proposed kernels.

3.1 GAUSSIAN PROCESSES REGRESSION

A GP is a collection of random variables, any finite number of which have a joint Gaussian distribution (Williams and Rasmussen, 2006). In this paper, we focus on random vectors and real-valued Gaussian processes (GPs), which can be completely specified by a mean function $m(\mathbf{x})$ and a kernel function $\kappa(\mathbf{x}, \mathbf{x}'; \boldsymbol{\gamma})$ as

$$f(\mathbf{x}) \sim \mathcal{GP}(m(\mathbf{x}), \kappa(\mathbf{x}, \mathbf{x}'; \boldsymbol{\gamma})), \tag{1}$$

where $\boldsymbol{\gamma}$ comprises the kernel hyper-parameters that need to be optimized. Given a real data set $\mathcal{D} \triangleq \{X, \boldsymbol{y}\}$, where $X = [\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N]^T$ is a matrix of N input vectors of dimension d, and $\boldsymbol{y} = [y_1, y_2, \cdots, y_N]^T$ is a vector of N outputs of the following GPR model:

$$y_i = f(\mathbf{x}_i) + \epsilon_i, \;\; i = 1, 2, \ldots, N, \tag{2}$$

where $y_i \in \mathbb{R}$ is a continuous-valued scalar output; the additive terms $\epsilon_i, i = 1, 2, \ldots, N$ are assumed to be i.i.d white Gaussian noise with variance σ^2; the underlying unknown function $f(\mathbf{x}_i) : \mathbb{R}^d \mapsto \mathbb{R}$ is desired and modeled as a GP. By definition, the collection $f(X) = [f(\mathbf{x}_1), f(\mathbf{x}_2), \cdots, f(\mathbf{x}_N)]^T$ follows a joint Gaussian distribution, i.e.,

$$f(X) \sim \mathcal{N}(\boldsymbol{\mu}_X, K(X, X)), \tag{3}$$

where the matrix $K(X, X)$ is short for $K(X, X; \boldsymbol{\gamma})$, and $\boldsymbol{\mu}_X$ is the mean vector evaluated for X.

GP inference. Established by popular usage, we can divide the original data set into a training set and a test set, namely $\mathcal{D} = [\mathcal{D}_t, \mathcal{D}_*]$. We denote $\mathcal{D}_t \triangleq \{X_t, \boldsymbol{y}_t\}$ and $\mathcal{D}_* \triangleq \{X_*, \boldsymbol{y}_*\}$ as the training and test data set respectively, where $X_t = [\mathbf{x}_{t1}, \mathbf{x}_{t2}, \cdots, \mathbf{x}_{tn}]^T$ and $X_* = [\mathbf{x}_{*1}, \mathbf{x}_{*2}, \cdots, \mathbf{x}_{*m}]^T$. According to the Bayes rule and the Gaussian assumptions, it is easy to see that the posterior distribution $p(\boldsymbol{y}_*|\mathcal{D}_t, X_*, \boldsymbol{\gamma}, \sigma^2)$ also follows a multivariate Gaussian distribution as follows:

$$p(\boldsymbol{y}_*|\mathcal{D}_t, X_*, \boldsymbol{\gamma}, \sigma^2) \sim \mathcal{N}\left(\bar{\mathbf{f}}_*, cov(\mathbf{f}_*)\right), \tag{4}$$

where

$$\bar{\mathbf{f}}_* = \boldsymbol{\mu}_{X_*} + K(X_*, X_t)C(\boldsymbol{y}_t - \boldsymbol{\mu}_{Xt}), \tag{5a}$$

$$cov(\mathbf{f}_*) = K(X_*, X_*) - K(X_*, X_t)CK(X_t, X_*). \tag{5b}$$

The term $K(X_*, X_t)$ represents the $m \times n$ covariance matrix of kernel functions evaluated for X_* and X_t, $C \triangleq (K(X_t, X_t) + \sigma^2 I_n)^{-1}$ for short, and $K(X_t, X_t)$ is the $n \times n$ covariance matrix evaluated for X_t. Then, by maximizing the posterior probability $p(\boldsymbol{y}_*|\mathcal{D}_t, X_*, \boldsymbol{\gamma}, \sigma^2)$, we can obtain the maximum a posterior (MAP) estimator $\bar{\mathbf{f}}_*$ for the desired targets $\boldsymbol{y}_*$.

Kernel learning. The most widely used kernel hyper-parameter learning method is to maximize the log-marginal likelihood $\log \mathcal{L} = \log p(\boldsymbol{y}_t|X_t)$, with respect to the kernel hyper-parameters $\boldsymbol{\gamma}$. Concretely,

$$\log \mathcal{L} \propto -\boldsymbol{y}_t^T (K_{\boldsymbol{\gamma}} + \sigma^2 I_n)^{-1} \boldsymbol{y}_t - \log |K_{\boldsymbol{\gamma}} + \sigma^2 I_n|, \tag{6}$$

where $K_{\boldsymbol{\gamma}}$ is a shorthand notation for $K(X_t, X_t; \boldsymbol{\gamma})$. As we can see, both the inversion $(K_{\boldsymbol{\gamma}} + \sigma^2 I_n)^{-1}$ and the log determinant $\log |K_{\boldsymbol{\gamma}} + \sigma^2 I_n|$ are computationally expensive. In the process of the kernel learning and the DKL in section 3.2, the following chain rule is required to compute the derivatives of $\mathcal{L}$ with respect to the kernel hyper-parameters

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{\gamma}} = \frac{\partial \mathcal{L}}{\partial K_{\boldsymbol{\gamma}}} \frac{\partial K_{\boldsymbol{\gamma}}}{\partial \boldsymbol{\gamma}}. \tag{7}$$

3.2 DEEP KERNEL LEARNING

The expressive power of a GP model is mainly determined by the kernel function. A representative deep kernel function is proposed as follows (Wilson et al., 2016a):

$$\kappa_{dkl}(\mathbf{x}, \mathbf{x}'; \boldsymbol{\gamma}) \rightarrow \kappa(h(\mathbf{x}, \mathbf{w}), h(\mathbf{x}', \mathbf{w}); \boldsymbol{\theta}, \mathbf{w}), \quad (8)$$

where $h(\cdot) : \mathbb{R}^d \mapsto \mathbb{R}^l$ represents an embedded DNN that maps an input $\mathbf{x}$ to a low-dimensional embedding, which breaks the curse of dimensionality of the base kernel functions $\kappa(\cdot, \cdot)$. There are some popular base kernel functions, e.g., the following squared-exponential (SE) kernel and the spectral mixture (SM) kernel (Wilson and Adams, 2013). The SE kernel is given by

$$\kappa_{\mathrm{SE}}(\mathbf{x}, \mathbf{x}') = \exp(-\|\boldsymbol{\tau}\|^2/2\ell^2), \quad (9)$$

where $\boldsymbol{\tau} \triangleq \mathbf{x} - \mathbf{x}'$, and $\boldsymbol{\theta} = \ell$ is the length-scale hyper-parameter; and the SM kernel is given by

$$\kappa_{\mathrm{SM}}(\mathbf{x}, \mathbf{x}') = \sum_{q=1}^{Q} w_q \frac{|\Sigma_q|^{\frac{1}{2}}}{(2\pi)^{\frac{d}{2}}} \exp(-\frac{1}{2}\|\Sigma_q^{\frac{1}{2}} \boldsymbol{\tau}\|^2) \cos\langle \boldsymbol{\tau}, 2\pi \boldsymbol{\mu}_q \rangle, \quad (10)$$

where the hyper-parameters $\boldsymbol{\theta}$ include the mixture weights w_q, bandwidth parameters Σ_q, and frequency parameters $\boldsymbol{\mu}_q$. The hyper-parameters $\boldsymbol{\gamma} = (\boldsymbol{\theta}, \mathbf{w})$ in eq. (8) are learnt jointly, where $\boldsymbol{\theta}$ represents the GP kernel parameters and $\mathbf{w}$ represents the NN weights. It is noteworthy that the number of parameters in $\mathbf{w}$ is way larger than that of $\boldsymbol{\theta}$.

On the basis of the traditional kernel learning mentioned in section 3.1, the derivatives with respect to the NN weight variables also need to be computed with the aid of eq. (7) for DKL, namely,

$$\frac{\partial \mathcal{L}}{\partial \mathbf{w}} = \frac{\partial \mathcal{L}}{\partial K_{\gamma}} \frac{\partial K_{\gamma}}{\partial h(\mathbf{x}, \mathbf{w})} \frac{\partial h(\mathbf{x}, \mathbf{w})}{\partial \mathbf{w}}, \quad (11)$$

where $\frac{\partial h(\mathbf{x}, \mathbf{w})}{\partial \mathbf{w}}$ is computed through the standard back-propagation. Overall, a GP with deep kernel is believed to produce a probabilistic mapping with infinite adaptive basis functions and effectively capture data covariances in high dimensions (Mallick et al., 2019). However, the embedded DNN will unfortunately degrade the interpretability of the base GP kernels. Meanwhile, the embedded fully-connected, over-parameterized NN structure needs a large number of labeled data samples to train efficiently. Moreover, training such a large DNN may suffer lots of difficulties, e.g., gradients vanishing, data-overfitting, trapping in a local minima, and so on. One way to alleviate all these problems is to decompose a large DNN into a batch of shallow subnetworks with higher interpretability. Meanwhile, the fitting performance can be retained or even enhanced. We will elaborate on this new idea in section 4.

4 OPTIMAL KERNEL DESIGN

In this section, we first design a novel optimal kernel function based on the minimum test MSE criterion. On the basis of this optimal kernel, we further propose an interpretable deep kernel with the aid of feature interaction detection. The benefits of the proposed kernels will also be discussed.

4.1 OPTIMAL KERNEL FUNCTION

For a given data set $\mathcal{D}$, we want to find the optimal kernel function that gives the minimum test MSE

$$MSE(\bar{\mathbf{f}}_*) = \mathbb{E}[(\bar{\mathbf{f}}_* - f(X_*))(\bar{\mathbf{f}}_* - f(X_*))^T], \quad (12)$$

where $MSE(\bar{\mathbf{f}}_*)$ is the MSE matrix of $\bar{\mathbf{f}}_*$ with respect to the true function value $f(X_*)$ for the test inputs, we commonly consider the corresponding MSE value (i.e., the trace of the MSE matrix), and $\bar{\mathbf{f}}_*$ is the MAP estimator of $p(f(X_*)|\mathcal{D}_*; \gamma)$. In practice, especially when the prior knowledge about the data distribution is not available, it is common to set the mean function of the GP equal to zero, i.e., $m(\mathbf{x}) = 0, \forall \mathbf{x}$. So $\bar{\mathbf{f}}_*$ in eq. (5a) under the zero mean assumption boils down to

$$\bar{\mathbf{f}}_* = K_{X_* X_*} \left[K_{X_* X_*} + \sigma^2 I_m\right]^{-1} \boldsymbol{y}_*, \quad (13)$$

where $K_{X_* X_*}$ denotes $K(X_*, X_*)$ for short. Since $\boldsymbol{y}_* \sim \mathcal{N}(f(X_*), \sigma^2 I_m)$ derived from eq. (2), the expectation of $\bar{\mathbf{f}}_*$ can be derived as

$$\mathbb{E}[\bar{\mathbf{f}}_*] = (I_m + G)^{-1} f(X_*), \quad (14)$$

where $G = \sigma^2 K_{X_* X_*}^{-1}$. We further define

$$\boldsymbol{f}_{bias} \triangleq \mathbb{E}[\bar{\mathbf{f}}_*] - f(X_*) = -(I_m + G)^{-1} G f(X_*), \quad (15)$$

$$\tilde{\boldsymbol{f}} \triangleq \bar{\mathbf{f}}_* - \mathbb{E}[\bar{\mathbf{f}}_*] = (I_m + G)^{-1}(\boldsymbol{y}_* - f(X_*)), \quad (16)$$

then, the test MSE matrix in eq. (12) with a feasible covariance matrix $K_{X_* X_*}$ can be formulated as

$$\begin{aligned} & MSE(\bar{\mathbf{f}}_*)(K_{X_* X_*}) \\ & = \mathbb{E}[\tilde{\boldsymbol{f}} \tilde{\boldsymbol{f}}^T] + \boldsymbol{f}_{bias} \boldsymbol{f}_{bias}^T \\ & = (I_m + G)^{-1}(\sigma^2 I_m + G f(X_*) f(X_*)^T G^T)(I_m + G)^{-1} \\ & = A^{-1}(\sigma^2 I_m + \sigma^4 K_{X_* X_*}^{-1} f(X_*) f(X_*)^T K_{X_* X_*}^{-1}) A^{-1} \\ & = B(\sigma^2 K_{X_* X_*} K_{X_* X_*} + \sigma^4 f(X_*) f(X_*)^T) B, \end{aligned} \quad (17)$$

where $A = (I_m + \sigma^2 K_{X_* X_*}^{-1})$ and $B = (K_{X_* X_*} + \sigma^2 I_m)^{-1}$ are two symmetric matrices. Based on the above derivations, we give the optimal kernel matrix theorem as below.

***Theorem* 4.1.** *Let $\bar{\mathbf{f}}_*$ be the MAP estimator of a GPR with the underlying function $f(X_*)$ evaluated for the test inputs X_*. Let $MSE(\bar{\mathbf{f}}_*)(K_{X_* X_*})$ be the mean-squared-error between the estimator $\bar{\mathbf{f}}_*$ of the GPR with* *any feasible covariance matrix $K_{X_*X_*}$ and the clean test labels. The following matrix inequality holds for any $K_{X_*X_*} \geq \mathbf{0}$:*

$$MSE(\bar{\mathbf{f}}_*)(K_{X_*X_*}) \geq MSE(\bar{\mathbf{f}}_*)(f(X_*)f(X_*)^T). \quad (18)$$

The relevant proof is introduced in Appendix A, and a similar proof mechanism can be found in Theorem 1 of (Chen et al., 2012) for regularized least-squares estimation. As a consequence of the Theorem 4.1, if we know the underlying function $f(\mathbf{x})$, the optimal kernel function based on the minimum test MSE criterion is

$$\kappa_{opt}(\mathbf{x}, \mathbf{x}') = f(\mathbf{x}) \cdot f(\mathbf{x}'). \quad (19)$$

Lemma 4.1. *In general, for a given mean function $m(\mathbf{x}) \neq 0$, we can prove that the optimal kernel function $\kappa_{opt}(\mathbf{x}, \mathbf{x}') = (f(\mathbf{x}) - m(\mathbf{x})) \cdot (f(\mathbf{x}') - m(\mathbf{x}'))$ in a similar way.*

Lemma 4.2. *Clearly, the optimal kernel defined by eq. (19) is a valid kernel function with non-stationary property.*

The proof of Lemma 4.2 is also provided in Appendix A. Obviously, the remaining question is that how can we approach the unknown underlying $f(\mathbf{x})$? Inspired by the Universal Approximation Theorem (Hornik, 1991), we choose to approximate the $f(\mathbf{x})$ by a NN with multi-layer feed-forward architecture, like the procedure in (Wilson et al., 2016a). Accordingly, our optimal deep kernel is derived as follows:

$$\kappa_{ok}(\mathbf{x}, \mathbf{x}'; \mathbf{w}) = g(\mathbf{x}, \mathbf{w}) \cdot g(\mathbf{x}', \mathbf{w}), \quad (20)$$

where $g(\cdot) : \mathbb{R}^d \mapsto \mathbb{R}$ represents an universal estimator of $f(\cdot)$, the comprehensive structure of the optimal deep kernel κ_{ok} is shown on the left hand side in Figure 1. As we can see, the optimal deep kernel is only parameterized by the network weights $\mathbf{w}$. In contrast to the DKL in section 3.2, the designed optimal base kernel function in theorem 4.1 gives a simpler and potentially more efficient kernel learning rationale. Besides the optimal kernel design, in section 4.2, we also make efforts on making the DKL more interpretable, with detailed model structure and analyses. Furthermore, in the case of the mean function $m(\mathbf{x}) \neq 0$, we can estimate $m(\mathbf{x})$ by the same pretrained NN that used to detect the feature interactions, for which the applied interaction detection method will be also introduced at the beginning of the section 4.2.

Remark. *Comparing our proposed kernel function $\kappa_{ok}(\mathbf{x}, \mathbf{x}'; \mathbf{w})$ in eq. (20) with the existing deep kernel $k(\mathbf{x}, \mathbf{x}'; \boldsymbol{\gamma})$ in eq. (8), we notice that there is essentially no need to further embed $h(\mathbf{x}, \mathbf{w})$ into a base kernel. Such embedding may cause an overfitted model and numerical instability.*

4.2 INTERPRETABLE OPTIMAL DEEP KERNEL

In practice, we care not only about the performance of the model, but also the interpretability of the model. However, the outputs $g(\mathbf{x}, \mathbf{w})$ learned from NNs are non-transparent and untraceable. There are many works showing that generalized additive models (GAMs) can achieve a good trade-off between functional approximation accuracy and model interpretability, in both machine learning and statistics (Hastie, 2017). We then choose to design an interpretable optimal deep kernel based on a generalized additive model (GAM) taking advantage of feature interaction outcomes obtained by the powerful neural interaction detection (NID) algorithm proposed in (Tsang et al., 2017). The NID directly interpret the learned weights of a feed-forward multi-layer NN. In other words, the NID provides pairwise and multi-way statistical interactions among features inherent in data.

Therefore, the underlying function $f(\mathbf{x})$ can be reshaped as the following GAM based on the detected multi-way interactions:

$$f(\mathbf{x}) = f_1(\mathbf{x}_{[s_1]}) + f_2(\mathbf{x}_{[s_2]}) + \cdots + f_k(\mathbf{x}_{[s_k]}), \quad (21)$$

where the $s_j, j = 1, 2, \ldots, k$ represent the detected k feature interaction sets whose cardinality is kept small preferably for better interpretability, e.g., if $s_1 = \{1, 2\}$, then $\mathbf{x}_{[s_1]} = (x_1, x_2)$, the $f_j(\cdot),\ j = 1, 2, \ldots, k$ are unspecified transformation functions depending on the corresponding input sets. For different input dimensions, the number of the detected interaction sets, k, may be chosen differently. For example, if $\mathbf{x} \in \mathbb{R}^3$, $f(\mathbf{x}) = 2\cos(x_1 + x_2) + 3x_3^2$, and the detected interaction sets are $s_1 = \{1, 2\},\ s_2 = \{3\}$, the corresponding GAM is

$$f(\mathbf{x}) = f_1(x_1, x_2) + f_2(x_3), \quad (22)$$

where $f_1(x_1, x_2) = 2\cos(x_1 + x_2)$ and $f_2(x_3) = 3x_3^2$. As a result, the optimal kernel function becomes

$$\begin{aligned} &\kappa_{opt}(\mathbf{x}, \mathbf{x}') && (23a)\\ &= f(\mathbf{x}) \cdot f(\mathbf{x}') && (23b)\\ &= (f_1(x_1, x_2) + f_2(x_3)) \cdot (f_1(x_1', x_2') + f_2(x_3')). && (23c) \end{aligned}$$

In light of the above GAM, the DNN embedded in the κ_{ok} defined in eq. (20) can now be decomposed as

$$\begin{aligned} \kappa_{iok}(\mathbf{x}, \mathbf{x}') &= (g_1(\mathbf{x}_{[s_1]}, \mathbf{w}_1) + \cdots + g_k(\mathbf{x}_{[s_k]}, \mathbf{w}_k)) \\ &\cdot (g_1(\mathbf{x}'_{[s_1]}, \mathbf{w}_1) + \cdots + g_k(\mathbf{x}'_{[s_k]}, \mathbf{w}_k)), \end{aligned} \quad (24)$$

where $g_j(\mathbf{x}_{[s_j]}, \mathbf{w}_j), j = 1, 2, \ldots, k$ are shallow and expressive subnetworks corresponding to different interaction sets.

In other words, the originally embedded large network $g(\cdot)$ in eq. (20) is decomposed into a GAM con-

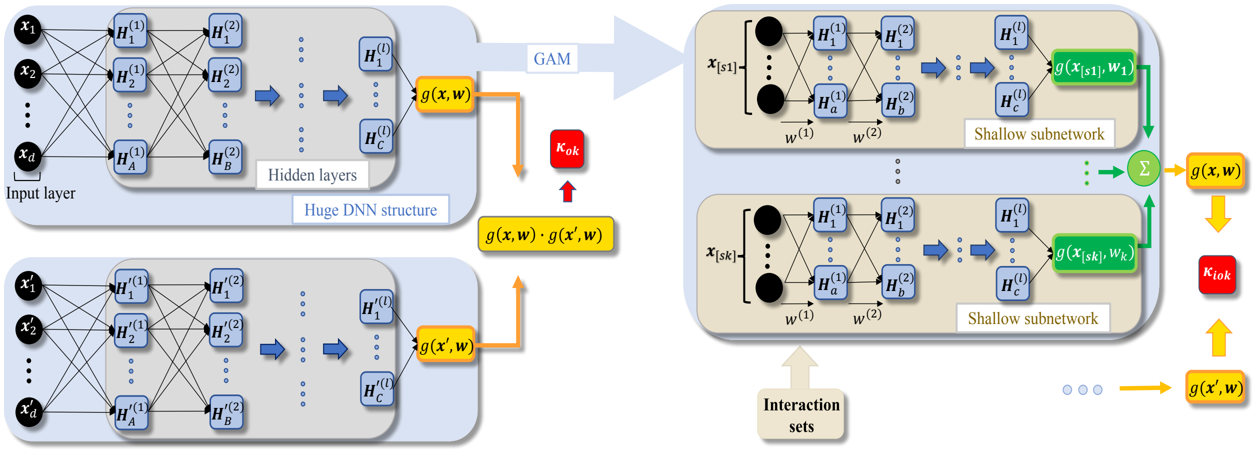


Figure 1: **Left:** The structure of the optimal deep kernel κ_{ok}, A multi-layer fully-connected feed-forward NN is applied to be the universal approximator of the underlying function $f(\mathbf{x})$. **Right:** The structure of the interpretable optimal deep kernel κ_{iok} by reconstruct κ_{ok} to a interpretable GAM model. All deep kernels are learning in a DKL procedure and parametrized by $\mathbf{w}$.

structed by k shallow subnetworks, each transparent subnetwork has neural interaction transparency and high interpretability. The structure of the interpretable optimal deep kernel κ_{iok} is shown on the right hand side of Figure 1. By contrast, the traditional DKL in section 3.2 involves both a base kernel and a DNN, which are parametric mapping functions. Our k_{iok} only involves a batch of shallow subnetworks with much reduced model parameters compared with the traditional deep kernel. With improved model interpretability and reduced parameter space of our GP regression model, overfitting can be alleviated even using a small batch of data. The detailed analyses and experimental results are presented in section 5. Furthermore, we can alternatively train the pivotal subnetworks, using adaptive optimization methods to update the weights, such as the coordinate descent with bandit sampling proposed in (Salehi et al., 2017). In this way, we can dramatically speed up the kernel learning procedure as the subnetworks can be parallel trained on different local GPUs and exploit the SPARK tools.

5 EXPERIMENTS

In this section, we experimentally evaluate the performance of GPs with the proposed optimal deep kernels κ_{ok} and κ_{iok} based on a range of tests in section 5.2, including a regression performance and interpretability test on varied synthetic data sets (Test 1); diverse collection of regression tasks from UCI and Kaggle repository (Test 2); stability and uncertainty test (Test 3); and the test of regression accuracy versus decreasing training data (Test 4). The experimental results demonstrate that the DKL with the proposed optimal kernels substantially outperform GPs with some state-of-the-art kernels, as well as the stand-alone DNNs.

5.1 EXPERIMENTAL SETUP

Table 1 shows the embedded NN structures in the deep kernel κ_{dkl} (Wilson et al., 2016a); the proposed optimal deep kernel κ_{ok}; and the proposed interpretable optimal deep kernel κ_{iok}. Since we focus on the scalar output GPR, for fair comparison, the NN structures of κ_{ok} are almost the same as the structure of κ_{dkl} used for the regression tasks in (Wilson et al., 2016a). As we can see, for $n \leq 6000$, the required parameters (i.e., weights) of the subnetworks embedded in κ_{iok} are reduced $(1-\rho)$, i.e., 85% (99% for $n > 6000$) total number of the compared large DNN structures in κ_{dkl}. The activation functions are chosen to be ReLU, the Xavier normal initialization is used to initialize the weights in each layer and Gaussian initialization with zero mean and unit variance is used for initializing the bias terms, we use GPytorch (Gardner et al., 2018) for regression with Adam optimizer. The commonly used root-mean-squared-error (RMSE) metric are applied for all regression tasks in section 5.2. The interaction sets $s_j, j = 1, 2, \ldots, k$ are obtained from the NID algorithm proposed in (Tsang et al., 2017), more concretely with a standard multi-layer perceptron (hidden layers size:"140-100-60-20") and a additional univariate networks (hidden layers size:"10-10-10") summed at the output. When training the above NNs, L_1 regularization is used and the regularizer is set to 5e-5.

Furthermore, as the lottery ticket hypothesis (LTH) states that there exist subnetworks (winning tickets) can reach the same level or even better test accuracy, compared to the original feed-forward dense DNN (Frankle and Carbin, 2018). Therefore, we apply LTH to the DNN that embedded in the proposed optimal deep kernel κ_{ok}.

Table 1: The comparison of neural network structures embedded in the deep kernels, (where $s_j, j = 1, 2, \ldots, k$ are the detected interaction sets, and ρ is the ratio of required NN parameters between κ_{iok} and κ_{dkl}, the following NN structures may change slightly depending on varied input data sets.)

	$n \leq 6000$	$n \geq 6000$
κ_{dkl}	d-1000-500-50-2	d-1000-1000-500-50-2
κ_{ok}	d-1000-500-50-1	d-1000-1000-500-50-1
κ_{iok}	$[s_j]$-500-300-50-1	$[s_j]$-1000-500-50-1
ρ	$\approx 15\%$	$\approx 1\%$

Concretely, we use the strategy of iterative pruning with resetting given in Appendix B of (Frankle and Carbin, 2018), where the surviving weights are reset after 300 to 500 iterations, depending on the data sets. And we prune 10% of the weights of all intermediate hidden layers, 5% of the last layer for each pruning. Thus, the κ_{ok} with embeddings obtained from the pruned sparse NNs may require a comparable number of parameters as the interpretable optimal deep kernel κ_{iok}.

5.1.1 Bechmark Approaches

The DKL with the proposed interpretable optimal deep kernel κ_{iok}, denoted by D-IOK, is compared with several state-of-the-art approaches including:

- D-RBF (Wilson et al., 2016a): Deep kernel Learning with ARD kernel incorporating DNN embeddings. The base kernel contains both a signal variance and a length-scale as the parameters.
- D-MAT: Deep kernel Learning with $Mat\acute{e}rn$-5/2 kernel incorporating DNN embeddings.
- D-SM (Wilson et al., 2016a): Deep kernel Learning with the Spectral Mixture kernel incorporating DNN embeddings.
- K-ARC (Cho and Saul, 2009): A GP with a kind of positive-definite kernel that can mimic the learning mechanism of multi-layer NNs.
- D-OK: Deep kernel Learning with the proposed optimal kernel function κ_{ok} incorporating sparse DNN embeddings.
- DNNs: The stand-alone deep neural networks with the same structures and initialization as the embedded DNN in the κ_{ok}.

5.1.2 Datasets

For GPR tasks, we firstly conduct our experiments with synthetic data sets generated by functions $F_1(\mathbf{x})$-$F_7(\mathbf{x})$ listed in Table 2. Secondly, several real data sets including three benchmark UCI regression data sets, namely $Skillcraft$, $Elevators$ (Wilson et al., 2016a) and $Parkinsons$ (where the last column of label are used), and the $4th$ index of two Kaggle regression data sets applied in (Tsang et al., 2017), $Bike\ sharing$, $California\ housing$ data sets are tested. The input data are standardized, and all randomly divided into three non-overlapping segments with 80% for training, 10% for validation, and 10% for test. The resulting RMSE values to be given in section 5.2 are averaged over 5 independent experiments for all data sets.

Table 2: Test suite of data generating functions

$F_1(\mathbf{x})$	$2\cos(x_1 + x_2) + 3x_3x_4 + x_5^3$
$F_2(\mathbf{x})$	$\exp\lvert x_1 - x_2\rvert + \lvert x_3x_4\rvert + \log(x_5^2 + x_6^2)$
$F_3(\mathbf{x})$	$\sin(x_1) - x_2^2 + \pi^{x_3x_4}\sqrt{2\lvert x_5\rvert}$
$F_4(\mathbf{x})$	$\sqrt{\exp(x_1 + x_2)} + x_3x_4x_5 + 2^{x_6+x_7}$
$F_5(\mathbf{x})$	$\sin(x_1) + (x_2 + 1)^{2\lvert x_3\rvert} + \pi^{x_4x_5}\sqrt{2\lvert x_6\rvert}$
$F_6(\mathbf{x})$	$\log(2x_1 + x_2 + 3) + \arccos(0.9x_3) + \sin(x_4 + x_5 - x_6) + 3x_7x_8 + \exp(\lvert x_9x_{10}\rvert + 1)$
$F_7(\mathbf{x})$	$\sqrt{\exp(x_1^2 + 1)} + (x_2x_3x_4)^3 + \sin(x_5 + x_6)$

Table 3: The test RMSE for different approaches applied to synthetic datasets, with n training points and dimensions d.

	n	d	DNN	D-RBF	D-OK	D-IOK
$F_1(\mathbf{x})$	0.8k	5	0.183	0.153	0.101	**0.081**
$F_2(\mathbf{x})$	1.6k	6	0.281	0.226	0.228	**0.127**
$F_3(\mathbf{x})$	2.4k	5	0.149	0.114	0.115	**0.101**
$F_4(\mathbf{x})$	3.2k	7	0.098	0.095	0.092	**0.092**
$F_5(\mathbf{x})$	2.4k	6	0.201	0.144	0.113	**0.093**
$F_6(\mathbf{x})$	4.8k	10	0.198	0.165	0.182	**0.097**
$F_7(\mathbf{x})$	5.6k	6	0.099	0.053	0.035	**0.027**

5.2 EXPERIMENTAL RESULTS

Test 1: We now consider a range of synthetic regression tasks. Table 3 reports the test RMSE for 1) the stand-alone DNNs; 2) the D-RBF; and 3) our proposed D-OK and D-IOK.

Table 3 shows that for most data sets, the D-RBF consistently outperforms the stand-alone DNNs, meaning that incorporating DNN into GP can not only provide uncertainty estimation but also improve the regression performance. We see that D-OK performs much better than the D-RBF in most cases, benefiting from the optimal kernel function. Taking advantage of the feature interactions, the proposed D-IOK outperforms all other competitors by far, showing that the GAM with shallow subnetworks can achieve the best performance in all cases.

Table 4: The test RMSE for different approaches with real regression data sets

Datasets	n	d	K-ARC	DNN	D-MAT	D-SM	D-OK	D-IOK
Skillcraft	3338	19	0.211±0.01	0.194±0.01	0.202±0.01	0.206±0.01	**0.188±0.00**	0.189±0.00
Parkinsons	5875	20	11.53±0.18	10.13±3.04	10.17±1.35	17.38±2.21	9.59±1.88	**8.53±1.92**
Elevators	16599	18	0.199±0.00	0.080±0.01	0.071±0.01	0.079±0.01	0.069±0.00	**0.069±0.00**
Bike sharing	17379	15	0.375±0.01	0.342±0.06	0.214±0.02	0.283±0.04	**0.183±0.01**	0.281±0.01
Cal housing	20640	8	0.344±0.01	0.390±0.01	0.342±0.01	0.342±0.01	**0.338±0.01**	0.339±0.01

We further test the interpretability of D-IOK by comparing the outputs $g_1(\mathbf{x}_{s_1}, \mathbf{w}_1)$ and $g_2(\mathbf{x}_{s_2}, \mathbf{w}_2)$ in $F_3(\mathbf{x})$ to the noisy samples generated from the corresponding ground truth transformation functions $f_1(\mathbf{x}_{[s_1]}) = \sin(x_1)$ and $f_2(\mathbf{x}_{[s_2]}) = x_2^2$ respectively. Figure 2 shows that the subnetworks can fit the transformation functions well. Since D-IOK is developed by an interpretable GAM of shallow subnetworks based on the detected feature interaction sets. Each detected interaction set contains 2-3 features, thus it can be conveniently visualized and analyzed by domain experts thereafter. Take the real California housing data set for instance, we can observe strong and meaningful interactions for sets $\mathbf{x}_{[s_1]}$ = (x_1: longitude, x_2: latitude), $\mathbf{x}_{[s_2]}$ = (x_4: total rooms, x_7: households) from Table 5. This makes sense, as $\mathbf{x}_{[s_1]}$ may indicate the location, $\mathbf{x}_{[s_2]}$ may indicate average living area per household. The experimental results explicitly demonstrate the excellent performance and interpretability of the D-IOK on simulated data, the tests with more complicated real data sets are presented in Test 2.

Test 2: We now consider the regression tasks of a large set of real data with varying size and properties, the test RMSE results are compared for 1) the stand-alone DNNs; 2) the D-MAT and the D-SM; 3) the K-ARC; and 4) our proposed D-OK and D-IOK.

Table 4 shows that there is no significant performance gap among all results achieved by the kernel based approaches on the $California\ housing$ data set, as the data set have enough training data and small input dimension. But the stand-alone DNNs are uncompetitive in most cases. The kernel function in K-ARC is defined as $\kappa_{\text{arccos}}(\mathbf{x}, \boldsymbol{y}) = \phi(\mathbf{x}) \cdot \phi(\boldsymbol{y})$, where we use ramp activation function with two successive applications of the nonlinear mapping $\phi(\cdot)$ for comparison. Although the learning mechanism of the NNs with infinite hidden neurons is approximated, the K-ARC can hardly achieve effective performance improvements in GPR tasks. The two compared DKL approaches, D-MAT and D-SM, with elementary base kernel function have relatively poor performance. Comparing to the nonparametric kernel learning methods, such as the latest functional kernel learning (FKL) introduced by (Benton et al., 2019), and others (Tobar et al., 2015; Oliva et al., 2016). The performance of the FKL is comparable to the D-SM, while other methods are slightly inferior due to the unstable performance of the non-stationary and high dimensional real data sets. By contrast, the D-OK and D-IOK applied the optimal kernel functions κ_{ok} and κ_{iok} can bring substantial additional performance gains for all data sets. The computation time for the D-OK is around three-fifths of a second per epoch, while this has been reduced to around a fifth of a second for the D-IOK.

We observed that the D-OK performs similar results to the D-IOK for the real data sets. The reason might be due to the more severe influence of the falsely detected interaction sets brought by the NID algorithm. We can refer to the detected outcomes reported in Table 5. A modified model that can potentially alleviate this problem is: $f(\mathbf{x}) = a_1 f_1(\mathbf{x}_{[s_1]}) + a_2 f_2(\mathbf{x}_{[s_2]}) + \cdots + a_k f_k(\mathbf{x}_{[s_k]})$, and we regularize the weights $\mathbf{a} = [a_1, a_2, \ldots, a_k]^T$ in the cost function for training the kernel hyper-parameters. In this way, those subnetworks with the right interaction sets will be retained, while the subnetworks with falsely detected interaction sets will be deactivated. Another reason might be that the LTH applied to the D-OK can lead to a sparse DNN (constituted by winning tickets), which brings additional performance gain (Frankle and Carbin, 2018). But we need to iteratively prune the large DNN for many times in order to obtain the winning tickets. While the D-IOK with the subnetworks based on feature interactions can also be seen as the outcomes after sparsifying the original large DNN in some sense.

Test 3 : The top two figures in Figure 3 show that D-OK is more robust than D-RBF. We conduct the regression tasks of $F_1(\mathbf{x})$ and $F_2(\mathbf{x})$ for twenty times independent experiments, as we all know, the GP with an appropriate kernel function will have less prediction uncertainty and the extra kernel parameters of D-RBF may incur data-overfitting. A comparison on the prediction uncertainty can be found in Appendix B. Note that the D-SM have more kernel parameters than D-RBF, worse test results can be imaged.

Test 4: The trend of the test RMSE versus decreasing percent of training data generated from $F_1(\mathbf{x})$ and $F_5(\mathbf{x})$ are depicted at the bottom of Figure 3, showing that the sample efficiency of the optimal deep kernels is superior to the basic RBF kernel in the case of insufficient input data. The D-SM is uncompetitive in the case of insufficient input data, since more kernel parameters need to be trained.

All tests have shown that the DKL with our proposed non-stationary optimal deep kernel functions outperform all other competitors considerably. As a conclusion, the well structured κ_{iok} with the aid of feature interaction detection definitely improves the performance of GP regression and the interpretability of the DKL. Furthermore, our work is also applicable to GP classification.

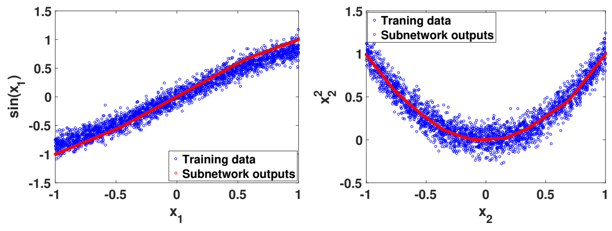


Figure 2: **Left** : The comparison between $g_1(\mathbf{x}_{[s_1]}, \mathbf{w}_1)$ constructed with the training data and the ground truth sub-function $f_1(\mathbf{x}_{[s_1]}) = \sin(x_1)$ in $F_3(\mathbf{x})$; **Right** : The comparison between $g_2(\mathbf{x}_{[s_2]}, \mathbf{w}_2)$ constructed with the training data and the ground truth sub-function $f_2(\mathbf{x}_{[s_2]}) = x_2^2$ in $F_3(\mathbf{x})$.

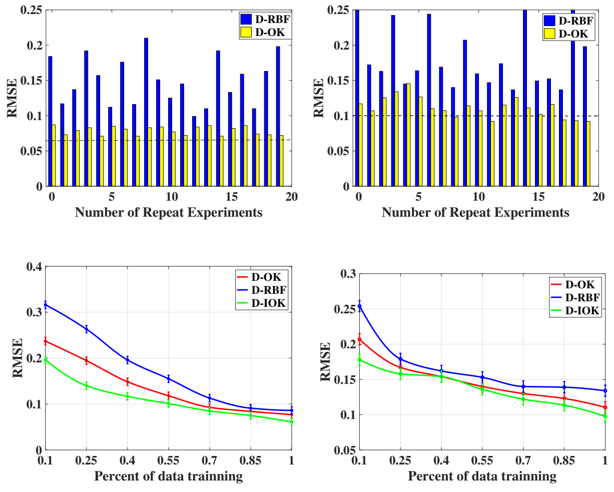


Figure 3: From left to right. **Top**: The stability test between D-OK and D-RBF for synthetic data set generated from $F_1(\mathbf{x})$ and $F_2(\mathbf{x})$ respectively; **Bottom**: The test RMSE versus decreasing percent of training data generated from $F_1(\mathbf{x})$ and $F_5(\mathbf{x})$ respectively.

6 CONCLUSION

In view of the prominent superiority of the deep kernel learning, we make efforts to the derivation of the optimal kernel function and the interpretable deep kernel learning structures. Specifically, we firstly derived an elegant optimal kernel function under certain assumptions. With the proposed interpretable deep kernel learning structure and detected feature interactions, we then proposed an optimal yet explainable and efficient deep kernel. Experimental results verified that this non-stationary valid kernel outperforms other state-of-the-art relevant kernels and offers highly interpretability and stability, making it promising to be applied to deep kernel design.

Table 5: Some detected Interaction sets with decreasing interaction strength for functions $F_1(\mathbf{x})$ to $F_4(\mathbf{x})$. Note Cal refers to the real data $California\ housing$.

Functions	$F_1(\mathbf{x})$	$F_2(\mathbf{x})$	$F_3(\mathbf{x})$	$F_4(\mathbf{x})$	Cal
Sets	$\{3,4\}$	$\{5,6\}$	$\{3,4\}$	$\{3,4,5\}$	$\{1,2\}$
Strengths	13.15	14.34	7.35	4.09	29.21
Sets	$\{1,2\}$	$\{1,2\}$	$\{1,2\}$	$\{6,7\}$	$\{4,7\}$
Strengths	10.43	10.49	1.89	1.43	20.36
Sets	$\{1,4\}$	$\{2,6\}$	$\{3,4,5\}$	$\{1,6,7\}$	$\{2,4,6\}$
Strengths	5.16	2.51	1.73	0.776	10.97
Sets	$\{1,2,3,4\}$	$\{1,2,3\}$	$\{1,3,4\}$	$\{1,2\}$	$\{6,7\}$
Strengths	3.12	1.51	1.73	0.54	6.29

Acknowledgements

This work was mainly supported by the National Key R&D Program of China with grant No. 2018YFB1800800, and partly by Natural Science Foundation of China No. 61701426, Guangdong Zhujiang Project No. 2017ZT07X152, and Shenzhen Peacock Plan No. KQTD2015033114415450.

References

Maruan Al-Shedivat, Andrew Gordon Wilson, Yunus Saatchi, Zhiting Hu, and Eric P Xing. Learning scalable deep kernels with recurrent structure. *Journal of Machine Learning Research*, 18(1):2850–2886, 2017.

Sivaram Ambikasaran, Daniel Foreman-Mackey, Leslie Greengard, David W Hogg, and Michael O'Neil. Fast direct methods for Gaussian processes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(2):252–265, 2015.

Sanjeev Arora, Simon S Du, Wei Hu, Zhiyuan Li, Ruslan Salakhutdinov, and Ruosong Wang. On exact computation with an infinitely wide neural net. *arXiv preprint arXiv:1904.11955*, 2019.

Gregory Benton, Wesley J Maddox, Jayson Salkey, Julio Albinati, and Andrew Gordon Wilson. Function-space distributions over kernels. In *Advances in Neural Information Processing Systems*, pages 14939–14950, 2019.

Tianshi Chen, Henrik Ohlsson, and Lennart Ljung. On the estimation of transfer functions, regularizations and Gaussian processes—revisited. *Automatica*, 48 (8):1525–1535, 2012.

Youngmin Cho and Lawrence K Saul. Kernel methods for deep learning. In *Advances in Neural Information Processing Systems*, pages 342–350, 2009.

Sambarta Dasgupta, Kumar Sricharan, and Ashok Srivastava. Finite rank deep kernel learning. *Bayesian Deep Learning (NeurIPS 2018)*, 2018.

Marc Peter Deisenroth and Jun Wei Ng. Distributed Gaussian processes. In *International Conference on Machine Learning*, pages 1481–1490, July 2015.

Kun Dong, David Eriksson, Hannes Nickisch, David Bindel, and Andrew G Wilson. Scalable log determinants for gaussian process kernel learning. In *Advances in Neural Information Processing Systems*, pages 6327–6337, 2017.

Jonathan Frankle and Michael Carbin. The lottery ticket hypothesis: Training pruned neural networks. *arXiv preprint arXiv:1803.03635*, 2018.

Jacob R Gardner, Geoff Pleiss, David Bindel, Kilian Q Weinberger, and Andrew Gordon Wilson. Gpytorch: Blackbox matrix-matrix gaussian process inference with gpu acceleration. In *Advances in Neural Information Processing Systems*, 2018.

Trevor J Hastie. Generalized additive models. In *Statistical models in S*, pages 249–307. Routledge, 2017.

Kurt Hornik. Approximation capabilities of multilayer feedforward networks. *Neural Networks*, 4(2):251–257, 1991.

Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural tangent kernel: Convergence and generalization in neural networks. In *Advances in Neural Information Processing Systems*, pages 8571–8580, 2018.

Mohammad Emtiyaz Khan. Deep learning with Bayesian principles. *Tutorial on Advances in Neural Information Processing Systems*, 2019.

Haitao Liu, Yew-Soon Ong, Xiaobo Shen, and Jianfei Cai. When Gaussian process meets big data: A review of scalable GPs. *IEEE Transactions on Neural Networks and Learning Systems*, pages 1 – 19, 2020.

Yin Lou, Rich Caruana, Johannes Gehrke, and Giles Hooker. Accurate intelligible models with pairwise interactions. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 623–631, 2013.

Ankur Mallick, Chaitanya Dwivedi, Bhavya Kailkhura, Gauri Joshi, and T Han. Deep probabilistic kernels for sample-efficient learning. *arXiv preprint arXiv:1910.05858*, 2019.

Junier B Oliva, Avinava Dubey, Andrew G Wilson, Barnabás Póczos, Jeff Schneider, and Eric P Xing. Bayesian nonparametric kernel-learning. In *Artificial Intelligence and Statistics*, pages 1078–1086, 2016.

Joaquin Quiñonero-Candela, Carl Edward Rasmussen, AnÃbal R Figueiras-Vidal, et al. Sparse spectrum Gaussian process regression. *Journal of Machine Learning Research*, 11:1865–1881, 2010.

Farnood Salehi, Patrick Thiran, and L. Elisa Celis. Stochastic dual coordinate descent with bandit sampling. *arXiv preprint arXiv:1712.03010*, 2017.

Hugh Salimbeni, Vincent Dutordoir, James Hensman, and Marc Peter Deisenroth. Deep Gaussian processes with importance-weighted variational inference. *arXiv preprint arXiv:1905.05435*, 2019.

Simo Sarkka, Arno Solin, and Jouni Hartikainen. Spatiotemporal learning via infinite-dimensional Bayesian filtering and smoothing: A look at Gaussian process regression through Kalman filtering. *IEEE Signal Processing Magazine*, 30(4):51–61, 2013.

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

Felipe Tobar, Thang D Bui, and Richard E Turner. Learning stationary time series using gaussian processes with nonparametric kernels. In *Advances in Neural Information Processing Systems*, pages 3501–3509, 2015.

Michael Tsang, Dehua Cheng, and Yan Liu. Detecting statistical interactions from neural network weights. *arXiv preprint arXiv:1705.04977*, 2017.

Shashanka Ubaru, Jie Chen, and Yousef Saad. Fast estimation of tr(f(a)) via stochastic lanczos quadrature. *SIAM Journal on Matrix Analysis and Applications*, 38(4):1075–1099, 2017.

Christopher KI Williams and Carl Edward Rasmussen. *Gaussian processes for machine learning*. MIT press Cambridge, MA, 2006.

Andrew Gordon Wilson and Ryan Adams. Gaussian process kernels for pattern discovery and extrapolation. In *International Conference on Machine Learning*, pages 1067–1075, 2013.

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

Andrew Gordon Wilson, Zhiting Hu, Ruslan Salakhutdinov, and Eric P Xing. Stochastic variational deep kernel learning. In *Advances in Neural Information Processing Systems*, pages 2586–2594, 2016b.

Appendix

A Proofs

Proof of Theorem 4.1. Define $U \triangleq -(K_{X_*X_*}\Sigma + I_m)^{-1}$ and $V \triangleq -(K_o\Sigma + I_m)^{-1}$, let $\Sigma = \sigma^{-2}I_m$, $K_o = f(X_*)f(X_*)^T$, then our target in eq. (18) can be rewritten as

$$U(K_{X_*X_*}\Sigma K_{X_*X_*} + K_o)U^T \geq V(K_o\Sigma K_o + K_o)V^T.$$

using the fact that

$$I + U = -UK_{X_*X_*}\Sigma,\ I + V = -VK_o\Sigma, \tag{25}$$

we can rewrite eq. (18) further as

$$\begin{aligned}&(I+U)\Sigma^{-1}(I+U)^T + UK_oU^T \geq \\ &(I+V)\Sigma^{-1}(I+V)^T + VK_oV^T.\end{aligned} \tag{26}$$

Now we focus on verifying the following equation generated by making the difference of the two sides in eq. (26), thus:

$$\begin{aligned}&(I+U)\Sigma^{-1}(I+U)^T + UK_oU^T \\ &-(I+V)\Sigma^{-1}(I+V)^T - VK_oV^T \\ &= M_0 + (U-V)(\Sigma^{-1})(U-V) + UK_oU^T - VK_oV^T,\end{aligned} \tag{27}$$

where

$$\begin{aligned}M_0 = &(I+V)\Sigma^{-1}U^T + U\Sigma^{-1}(I+V^T) \\ &-(I+V)\Sigma^{-1}V^T - V\Sigma^{-1}(I+V^T),\end{aligned} \tag{28}$$

from the second equation in eq. (25), we can obtain

$$(I+V)\Sigma^{-1} = -VK_o. \tag{29}$$

Substituting eq. (29) into eq. (28), we have

$$M_0 = 2VK_oV^T - VK_oU^T - UK_oV^T. \tag{30}$$

Using the result given in eq. (30), eq. (27) becomes

$$\begin{aligned}&(I+U)\Sigma^{-1}(I+U)^T + UK_oU^T \\ &-(I+V)\Sigma^{-1}(I+V)^T - VK_oV^T \\ &= (U-V)(\Sigma^{-1} + K_o)(U-V)^T.\end{aligned} \tag{31}$$

One can easily see that the right-hand side of eq. (31) is positive semi-definite, so the inequality in eq. (26) holds, as well as eq. (18), which completes the proof. □

Proof of Lemma 4.2. By definition of a kernel matrix, for all $i, j \in 1, 2, \ldots, N$

$$K_{i,j} = \kappa_{opt}(\mathbf{x}_i, \mathbf{x}_j) = f(\mathbf{x}_i) \cdot f(\mathbf{x}_j), \tag{32}$$

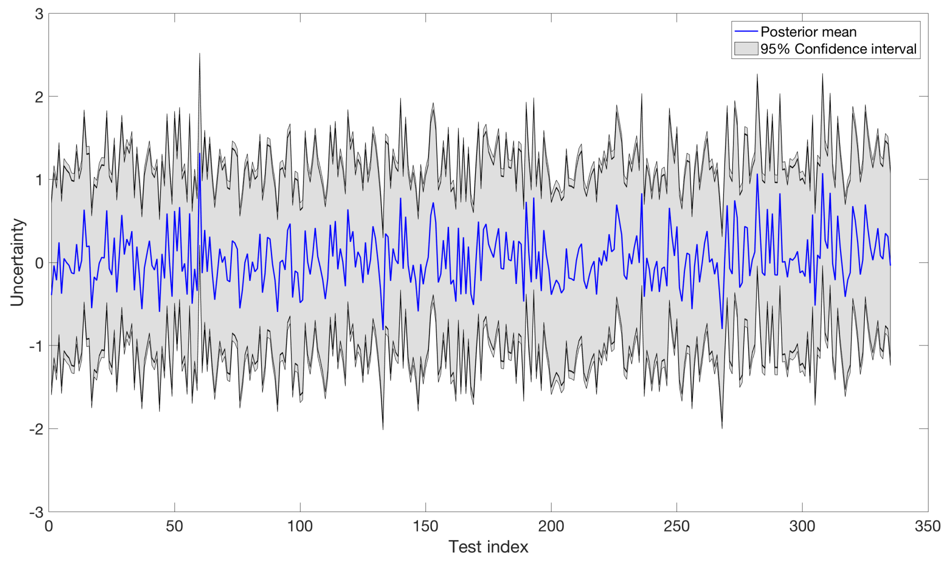
where $f(\mathbf{x}) : \mathbb{R}^d \mapsto \mathbb{R}$. Thus for any $\mathbf{v} \in \mathbb{R}^N$,

$$\begin{aligned}\mathbf{v}^T K\mathbf{v} &= \sum_{i,j} \mathrm{v}_i K_{i,j} \mathrm{v}_j = \sum_{i,j} \mathrm{v}_i f(\mathbf{x}_i) \cdot f(\mathbf{x}_j)\mathrm{v}_j \\ &= \left\| \sum_{i=1}^{N} \mathrm{v}_i f(\mathbf{x}_i) \right\|^2 \geq 0.\end{aligned} \tag{33}$$

Apparently, this optimal kernel is not a function of $\boldsymbol{\tau}$ defined in eq. (9), thus violating the definition of a stationary kernel (Williams and Rasmussen, 2006). □

B Prediction uncertainty

The following Figure 4 shows the prediction uncertainty of the real data $Skillcraft$. In our experiments, the number of the test data is 335. The top one in Figure 4 applied the D-RBF while the bottom one shows the result of our proposed D-OK. We can easily find that the prediction uncertainty of the D-OK is less than that of the D-RBF.



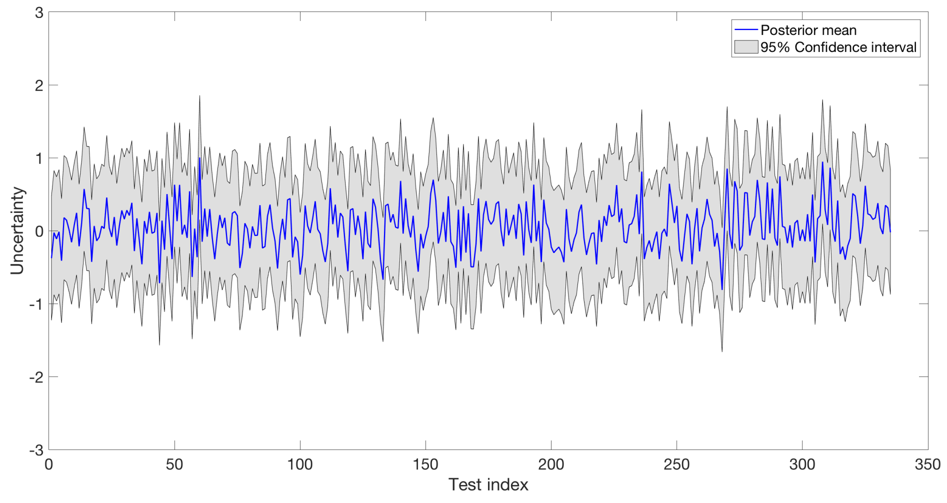


Figure 4: The prediction uncertainty of real data $Skillcraft$ with the traditional D-RBF (top) and our proposed D-OK (bottom). The X-axis shows the test data index from 1 to 335.