Deep Mean-Kernel Learning for Bayesian Optimization in IoT-Enabled Data Center

Jinhui Dou*, Bo Cheng*, Jinan Li†, Yu Sun*, Gaoxiang Jiang*, and Haibo Zhou*
*School of Electronic Science and Engineering, Nanjing University, Nanjing, China, 210023.
Emails: {jinhuidou,bocheng,yusun,gaoxiangjiang}@smail.nju.edu.cn, haibozhou@nju.edu.cn.
† Guangdong Branch, China Unicom Group Co., Ltd., Guangzhou, China, 510627.
Email: jinanli@chinaunicom.cn

2024 16th International Conference on Wireless Communications and Signal Processing (WCSP) | 979-8-3503-9064-3/24/$31.00 ©2024 IEEE | DOI: 10.1109/WCSP62071.2024.10827093

***Abstract*—In efforts to mitigate energy consumption in large data centers, numerous optimization strategies have been explored. However, these methods often encounter challenges including prolonged optimization times and settling on local optima. Among diverse algorithms, Bayesian optimization (BO) demonstrates exceptional performance. Due to the slow convergence and limited scalability of existing Bayesian methods in handling high-dimensional data, we explore the influence of the surrogate model on BO, and introduce an innovative optimization approach deep mean-kernel learning, which can converge rapidly to the global optimal solution in high dimensional space and is specifically designed for the refrigeration systems of data centers. In our approach, a neural network is employed to approximate the conventional mean function, while an infinite-width neural network serves as the kernel function, modifying the Gaussian process. Utilizing two prevalent acquisition functions, we conduct comparative experiments of our algorithm against existing optimization methods in the context of data center scenarios. Our findings demonstrate that our approach adeptly manages high-dimensional data and exhibits rapid convergence towards the global optima.**

***Index Terms*—Bayesian optimization, Gaussian process, surrogate model, mean function, kernel function, data center**

I. Introduction

As governments and data center operators increasingly focus on energy consumption issues, reducing carbon emissions and energy consumption in large data centers has become a critical issue [1]. Cooling systems, which account for a significant portion of energy consumption in data centers, are crucial to be optimized [2]. Researchers have proposed various approaches to tackle these optimization challenges. However, these methods generally encounter some issues, such as high computational complexity and a tendency to converge to suboptimal local solutions. Bayesian optimization (BO) [3] is a highly efficient method utilized for optimizing objective functions, particularly beneficial in scenarios involving parameter tuning, hyperparameter optimization of machine learning models, and situations requiring costly evaluations. It typically initiates with some random evaluation points and employs a Gaussian process (GP) to model the objective function. Subsequently, it selects new evaluation points based on an acquisition function (such as expected improvement (EI) [4] or upper confidence bound (UCB)) [5], balancing exploration (seeking new areas) and exploitation (optimizing known good areas), updating the model after each observation, and eventually converging to the optimal solution. Compared to traditional optimization algorithms, BO shows significant advantages. It is highly sample-efficient, making it ideal for costly evaluation problems. Furthermore, BO is renowned for its robust global optimization capabilities, which facilitated by acquisition functions that effectively manage the trade-off between exploration and exploitation, enhancing the probability of identifying the global optimum. Although the application of standard GPs in Bayesian optimization is well-established, certain domains still face complexities that demand innovations beyond traditional BO algorithms.

Deep Kernel Learning (DKL) is a technique that combines deep learning with GP [6]. It uses a neural network to transform the inputs of the GP, learning the feature representation of the data, and then uses the learned features as inputs to the GP. This method creates a flexible kernel function capable of learning from complex data. However, this approach is computationally expensive and may not handle multiple types of problems effectively. Hamiltonian Monte Carlo (HMC) is an efficient Markov Chain Monte Carlo (MCMC) sampling method for drawing samples from complex posterior distributions [7]. It simulates the Hamiltonian dynamics of a physical system to generate proposal samples, which helps to explore probability distributions effectively in high-dimensional spaces. However, the adjustment of parameters such as step size and number of steps, significantly affects the efficiency and stability of the algorithm. Moreover, HMC can be time-consuming in reaching equilibrium distribution, making it unsuitable for problems requiring rapid iterations [8]. Deep Ensembles have been proven to fully approximate Bayesian inference, inheriting the benefits from multiple retrainings of neural networks [9], but consumes considerably more computational resources and is more complex to implement than standard algorithms. The data characteristics of data center cooling systems are intricate, featuring high-dimensional attributes and encompassing significant observational noise. Furthermore, when using optimization algorithms in practice, we aim to obtain optimization results as quickly as possible to better adjust the operating state of the data center cooling equipment and minimize energy consumption.

In this paper, we propose a method for effectively identifying the optimal solution when handling high-dimensional data. We alter the mean and kernel functions of the GP, sig-

979-8-3503-9064-3/24/$31.00 ©2024 IEEE

nificantly enhancing the applicability of BO. The fundamental contributions of this paper are summarized as follows:

- We build an optimization method for data center scenarios which can improve the efficiency of optimization based on historical data and minimize energy consumption during optimization.
- We propose a novel BO-based algorithm, deep mean-kernel learning (DMKL), which utilizes neural networks to approximate the mean function and I-BNN to approximate the kernel function, effectively addresses significant challenges including prolonged optimization times and difficulties in handling high-dimensional input data.
- We conduct comparative experiments of various optimization schemes in the scenario of data center, which demonstrate that our proposed scheme excels in high-dimensional space and exhibits superior convergence properties.

The remainder of this paper is structured as follows: we begin in Section II by demonstrating the architecture of BO in IoT-enabled data centers. The principles of BO and the DMKL architecture are presented in Section III. The experimental results are elaborated upon in Section IV. We finish with conclusion in Section V.

II. System Model

A. Bayesian Optimization In IoT-Enabled Data Centers

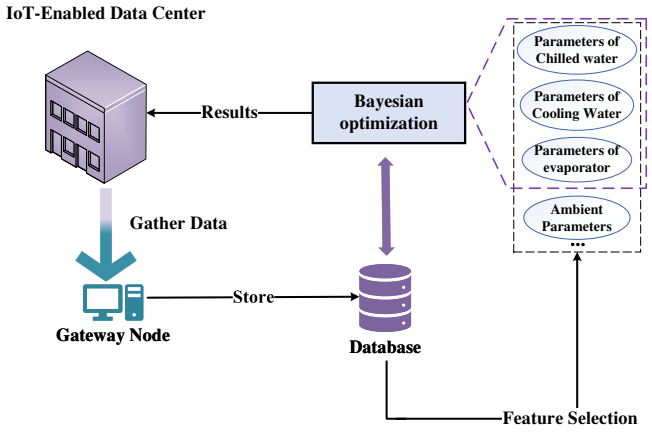


Fig. 1: Bayesian optimization framework for IoT-enabled data centers.

The Bayesian optimization framework for data centers is depicted in Fig. 1. Utilizing advanced sensor technology, we can acquire and monitor the real-time status of data centers, including various types of data such as temperature and water pressure. By deploying such sensor network architecture, we can collect data from each node and then store it in a database, ultimately accumulating a substantial amount of status data. Based on historical data, we optimize the energy consumption of the cooling system by adjusting temperature and pressure to achieve the data center's minimized energy.

III. Optimization Model Structure

A. Bayesian Optimization

BO is a successful application of Bayesian inference. It's particularly suitable for scenarios with high evaluation costs, scarce data, or high model complexity. Our objective is to identify the global maximum of an unknown function using BO. However, in the context of data center optimization, our goal shifts to determining the minimum value of the function. We assume that the energy consumption objective function is $p(\mathbf{x})$, thus the task of identifying the global minimum of an unknown objective function can be defined as follows:

$$\max_{\mathbf{x}\in\mathcal{X}} -p(\mathbf{x}). \tag{1}$$

In BO, a surrogate model is utilized to estimate the posterior predictive distribution of sample points within the parameter space. Subsequently, an acquisition function, informed by this posterior distribution, guides the selection of the next sample point for evaluation in the subsequent iteration.

B. Surrogate Model

GP is a common choice for the surrogate model in BO thanks to its excellent probability prediction properties. GP is a kind of stochastic process, in which subsets of finite dimensions obey a multivariate Gaussian distribution. The probability density function of a one-dimensional normal distribution is given by:

$$\phi(x) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(\frac{-(x-\mu)^2}{2\sigma^2}\right). \tag{2}$$

For multi-dimensional variables, on the premise that each dimension is independent, we can derive the probability density function as follows:

$$gp(\mathbf{x}) = \frac{1}{(2\pi)^{d/2}|K|^{1/2}} \exp\left(-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^\top K^{-1}(\mathbf{x}-\boldsymbol{\mu})\right), \tag{3}$$

where $K \in \mathbb{R}^{n\times n}$ represents the covariance matrix, which is a diagonal matrix within the space, and $\boldsymbol{\mu}$ denotes the mean vector. The posterior distribution of $o : \mathbf{x} \mapsto \mathbf{R}^d$ at a given location $\mathbf{x}$ is a gaussian distribution. The GP prior over objective function is defined as:

$$p(o \mid \mathbf{x}, \mathcal{D}, \boldsymbol{\theta}) = \mathcal{N}\left(o \mid \mu(\mathbf{x}), \sigma^2(\mathbf{x})\right), \tag{4}$$

$$\mu(\mathbf{x} \mid \mathcal{D}, \boldsymbol{\theta}) = m(\mathbf{x}) + \boldsymbol{\kappa}(\mathbf{x}, X)K^{-1}(\mathbf{f}-\mathbf{m}), \tag{5}$$

$$\sigma^2(\mathbf{x} \mid \mathcal{D}, \boldsymbol{\theta}) = \kappa(\mathbf{x},\mathbf{x}) - \boldsymbol{\kappa}(\mathbf{x}, X)^\top K^{-1}\kappa(X,\mathbf{x}), \tag{6}$$

$m(\mathbf{x})$ is the vector comprised of the mean function.

Kernel and mean functions can be considered as defining a Bayesian prior based on available data. The kernel function characterizes the structural form of the fitting function, while the mean function denotes the expected prior value of the function at each point. Specifically, we predict the function value using the posterior mean $\mu(\mathbf{x} \mid \mathcal{D}, \boldsymbol{\theta})$ and estimate the associated uncertainty using the posterior variance $\sigma^2(\mathbf{x} \mid$

$\mathcal{D}, \boldsymbol{\theta}$). The most commonly used kernel function is the Radial Basis Function (RBF), whose expression is:

$$\phi(r) = e^{-\epsilon r^2}, \tag{7}$$

in which r is the Euclidean distance from center point to the current point, and ϵ is a positive tuning parameter, called the length scale parameter, which controls the width of the function.

Essentially, the distribution of a GP is the joint distribution of all these infinitely many random variables, which encompasses the complete set of possible configurations that the random variables can assume across the input space. This framework allows for a comprehensive modeling of correlations between points based on their relative locations in the input space, facilitating predictions and uncertainty estimations in various domains of application.

C. Acquisition Function

The two most commonly used acquisition functions are EI and UCB. EI is designed to directly measure the expected improvement at a point relative to the current best observation [10]. Specifically, it calculates the expected value of obtaining an improvement when sampling at a given point. The expression for EI is as follows:

$$\mathrm{EI}(\mathbf{x}) = (\mu(\mathbf{x}) - f(\mathbf{x}^*))\phi(Z) + \sigma(\mathbf{x})\phi(z). \tag{8}$$

In Eq. (8), μ and $\sigma(x)$ respectively refer to the predicted mean and standard deviation at point $\mathbf{x}$, $f(\mathbf{x}^*)$ is the currently known best value of the objective function, Φ is the cumulative distribution function of the standard normal distribution and ϕ is the probability density function. z is a random variable defined as:

$$z = \frac{\mu(\mathbf{x}) - f(\mathbf{x}^*)}{\sigma(\mathbf{x})}. \tag{9}$$

The UCB function considers both exploration (sampling in areas of high uncertainty) and exploitation (sampling in areas predicted to perform well). Calculating UCB typically involves the mean and variance of GP predictions. This function consists of the predicted mean plus a term proportional to the predicted standard deviation. This coefficient can be adjusted to balance exploration and exploitation. UCB has advantages in optimization problems with high uncertainty [11], as it ensures a degree of exploration to avoid falling into local optima. The expression for UCB is as follows:

$$\mathrm{UCB}(\mathbf{x}) = \mu(\mathbf{x}) + \kappa\sigma(\mathbf{x}). \tag{10}$$

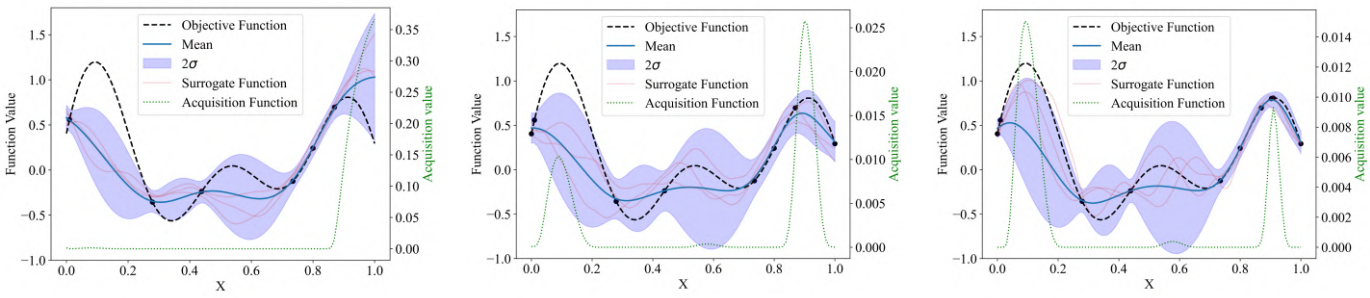
Overall, EI tends to choose areas that are likely to exceed the current best observation, while UCB balances selections between areas of good current performance and high uncertainty.

We now select a standard GP as the surrogate function, with EI as the acquisition function. Fig. 2 shows the process of BO in finding the maximum of a single-objective target function with one-dimensional input over five iterations. We start with six initial points and perform five iterations. Initially, EI fails to identify the maximum position of the function, which successfully attains the maximum value at a locally optimal position by the third iteration. As new points are acquired with each iteration, the acquisition function reaches its maximum near the unknown maximum of the target function, pinpointing the location of the maximum value of the unknown target function upon reaching the fifth iteration.

D. Deep Mean-Kernel Learning

Due to the curse of dimensionality, common covariance functions may not accurately represent high-dimensional input data, and GPs tend to struggle with scalability in high-dimensional input spaces. In contrast, neural networks excel in handling high-dimensional data due to their flexible and deep architectures. Within the constraints of an infinitely wide architecture, a single-layer fully connected neural network can be theoretically equated to a GP. Consequently, it is worth considering the integration of different network structures to replace the mean and kernel functions in GPs when dealing with high-dimensional data [12]. This approach can enhance the model's ability to capture complex patterns in the data. Additionally, considering the iteration speed of the overall optimization process is crucial, as it impacts the efficiency and practicality of the model in real-world applications. By modifying these components, we aim to improve both the performance and scalability of GPs for high-dimensional datasets.

Bayesian neural network (BNN) is grounded in the principles of Bayesian statistics, which integrate measures of uncertainty into the neural network learning process. Central to the BNN methodology is the treatment of network weights as probability distributions rather than as fixed values. This probabilistic framework not only facilitates learning from data but also enables the incorporation of prior knowledge and the quantification of uncertainty in predictions. During the training phase, BNN eschews the optimization of fixed weight values in favor of updating the posterior distributions of these weights. The concept of an Infinite-width neural networks (I-BNNs) pertains to the behavior observed when the number of nodes in each hidden layer tends toward infinity. Leveraging the central limit theorem, it has been demonstrated that a bayesian neural network with a single hidden layer of infinite width converges to a GP characterized by a neural network-based covariance function [13]. This insight has subsequently been extended to encompass deep neural networks [14]. Notably, they are adept at managing non-stationarity and utilize a fixed covariance function, thus providing a comparatively robust prior. I-BNNs does not predict a definite output directly, but rather the distribution of the output, in a way that naturally captures variations and uncertainties in the input data. When new data becomes available, I-BNNs can update their posterior distributions without the need for retraining from scratch, enabling the model to efficiently adapt to changes in the data. We use an I-BNNs structure to replace the kernel function in a standard GP and a neural network framework to replace the mean function.



(a) Iteration 1 (b) Iteration 3 (c) Iteration 5

Fig. 2: The process of surrogate function fitting and corresponding acquisition functions during BO iterations.

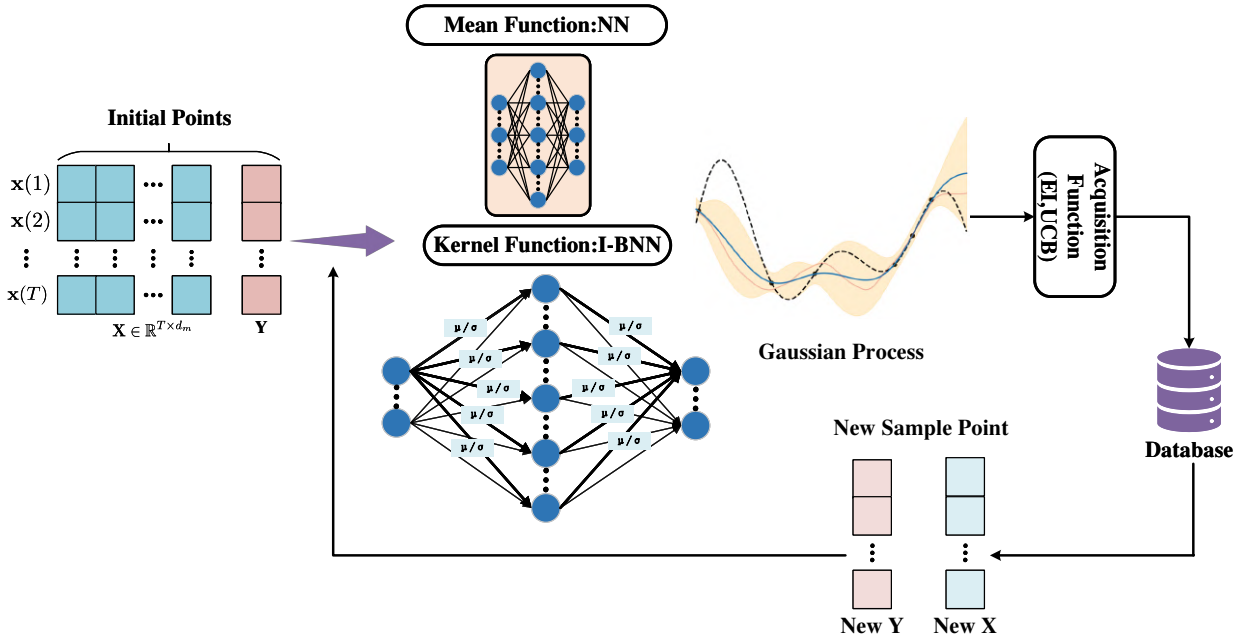


Fig. 3: The overall framework of Deep Mean-Kernel Learning.

IV. Experiments

A. Experiment Setups

Dataset. In this study, we collect substantial and relevant data from IoT-enabled data centers and process this data utilizing methodologies akin to those outlined in [1].In our study, We select ten parameters that are controllable within the data center environment and identify them as input variables. For the purposes of our experiments, we select a dataset comprising 19,627 samples, which are obtained from a real data center to ensure the authenticity and reliability of the analysis.

Baselines. We select the following optimization methods as baselines: standard GPs; HMC [8], deploying two Markov chains with each chain comprising 500 samples, and setting the path length of a complete Hamiltonian trajectory to 0.02; DKL [6], configured with a three-layer neural network where the hidden layers each contain 128 neurons, with 1000 pre-training iterations prior to each model fitting, and 3000 training iterations during each fitting session; and an Ensemble method, utilizing a total of ten neural network models, each featuring a three-layer hidden structure, with each model undergoing 1000 iterations per fitting [9].

Setting. We commence our experiments by selecting 20 initial points, each consisting of 10 variables across 20 dimensions along with their corresponding objective function values. We employ two well-established acquisition functions, EI and UCB. For EI, we compare the results using 16 randomly sampled points. The optimization of the GP is carried out via the L-BFGS-B algorithm [15], incorporating 10 restarts to maximize the likelihood function. In the implementation of UCB, we set the trade-off parameter between the mean and covariance to 0.2. Our experiments are implemented on the basis of BoTorch [16].

B. Experimental Results

*1) **Convergence**:* Based on historical data, we develop a data center architecture modeled with a neural network. With

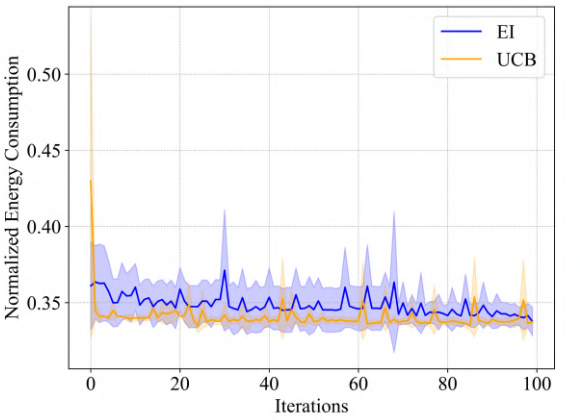


Fig. 4: Convergence curves of DMKL versus different acquisition functions.

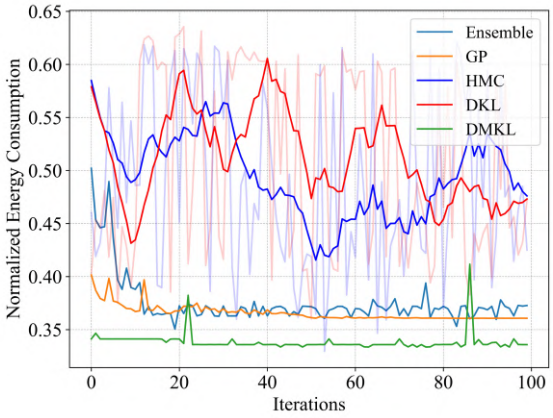


Fig. 6: Convergence curves of different BO algorithms using UCB.

this model we apply our method to the cooling data of a data center with the objective of minimizing normalized energy consumption. By leveraging a reference model during the optimization process, we substantially reduce resource consumption and time costs associated with tuning in actual systems, while also minimizing potential risks [17]. We evaluate the effects of using two common acquisition functions with DMKL. We conduct five trails using each acquisition function and computed the mean and variance of the optimized outcomes. The solid lines indicate the means, while the shaded areas of corresponding colors depict the confidence intervals.

Both acquisition functions lead to the same optimal value, with the curves converging at the same point. However, in terms of convergence speed, EI demonstrated faster convergence than UCB. After convergence, the optimal result is about 0.3358. We can observe that the acquisition function only influences the specific process of the optimization iteration, but it does not impact the final optimization result. In practical applications, considering efficiency in optimization, we can opt for the UCB as the acquisition function.

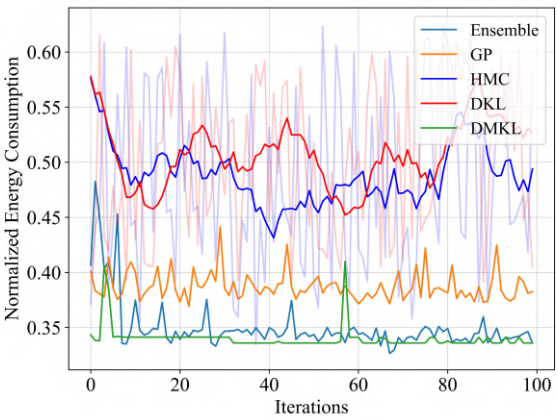


Fig. 5: Convergence curves of different BO algorithms using EI.

*2) **Performance of BO and Benchmarks**:* We employ various optimization methods to reduce energy consumption, observing the convergence trends over 100 iterations with normalized energy consumption as the target metric. We investigate the correlation between the minimized energy values and the number of iterations for each method. The analysis reveals that the DKL and HMC methods display significant variability and instability, failing to exhibit a consistent trend towards convergence within the span of 100 iterations. To enhance the clarity of the visualization for the non-convergent methods, we apply a smoothing technique to both. This adjustment facilitates a more discernible comparison of their behaviors within the graphical representation. In contrast, three other methods show notable convergence within the specified iteration range. Specifically, Deep Ensemble reaches convergence around the 25th iteration, while the standard GP and DMKL achieve convergence around the 60th iteration. DMKL proves to be the most effective, achieving an optimized value of 0.336.

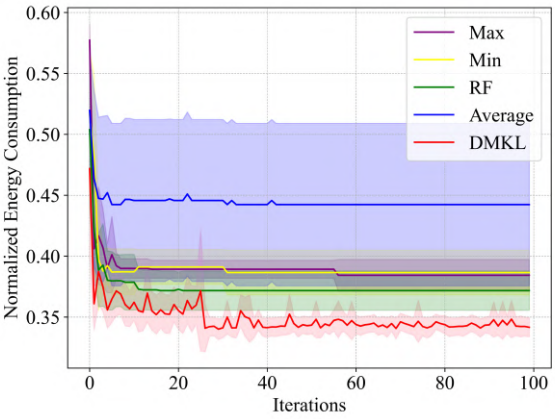


Fig. 7: Convergence curves of DMKL versus different mean functions using EI.

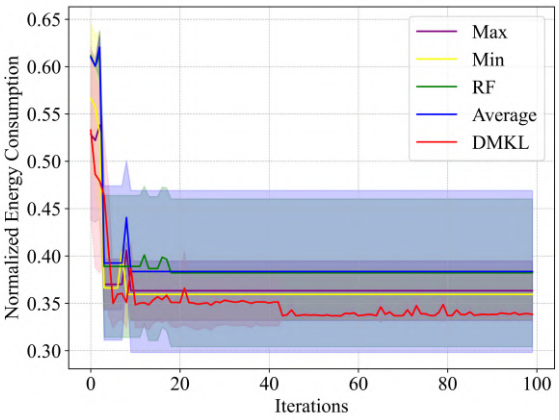


Fig. 8: Convergence curves of DMKL versus different mean functions using UCB.

*3) **Performance Versus Mean Functions**:* We select the maximum, minimum, and random forest as the mean functions for the GP and compare their performance with our proposed neural network structure as the mean function. We observe the convergence and stability of various mean functions over 100 iterations. As illustrated in the figure, we conduct five round of experiments for each mean function, then calculate the mean and confidence intervals from these five experiments. The experimental means are represented by solid lines, while the corresponding confidence intervals are depicted by shaded areas in matching colors. It is evident that the DMKL, significantly outperformes the other mean functions. DMKL still continues to explore in a small range after obtaining a better result, while other mean functions find it difficult to find the next exploration point after convergence, resulting in continuous convergence without achieving better results. while other mean functions struggle to identify the next point for exploration after converging, they often become trapped in local optima, failing to locate the global maximum value. This results in continuous convergence without yielding improved outcomes.

V. Conclusion

In this paper, we have employed BO to tackle the significant challenge of reducing energy consumption in IoT-enabled data center cooling systems. Given the extensive volume and high dimensionality of data, the complexity of the objective function, we have refined the conventional GP structure by integrating a neural network as the mean function and an I-BNN as the kernel function, naming it Deep Mean Kernel Optimization. We have conducted a rigorous evaluation of DMKL, employing two widely recognized acquisition functions and comparing its performance with several state-of-the-art BO algorithms. Furthermore, we have investigated the impact of different mean functions on the optimization results. This examination has yielded valuable insights into how various mean functions influence the optimization process and their relative effectiveness in guiding the optimization strategy. This systematic exploration aids in identifying the most appropriate mean functions for specific optimization scenarios, contributing to the development of more refined and effective optimization methodologies. Overall, the proposed DMKL method has proven highly effective in addressing the excessive energy consumption in data centers. It has notably accelerated the convergence speed of the optimization process, significantly enhanced the quality of optimization. Additionally, it has markedly improved the capacity of the optimization algorithm to handle high-dimensional challenges. For future work, we plan to further assess the robustness and efficiency of DMKL across a wider array of high-dimensional datasets to confirm its generalization capabilities and sustained performance improvements.

References

[1] Y. Sun, Y. Wang, G. Jiang, B. Cheng, and H. Zhou, "Deep learning-based power usage effectiveness optimization for IoT-enabled data center," *Peer-to-Peer Networking and Applications*, Mar. 2024.

[2] J. Ni and X. Bai, "A review of air conditioning energy performance in data centers," *Renewable and Sustainable Energy Reviews*, vol. 67, pp. 625–640, 2017. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S136403211630541X

[3] H. Wang and K. Yang, *Bayesian Optimization*. Cham: Springer International Publishing, 2023, pp. 271–297. [Online]. Available: https://doi.org/10.1007/978-3-031-25263-1_10

[4] S. Ament, S. Daulton, D. Eriksson, M. Balandat, and E. Bakshy, "Unexpected Improvements to Expected Improvement for Bayesian Optimization," *arXiv e-prints*, p. arXiv:2310.20708, Oct. 2023.

[5] Z. Wang and S. Jegelka, "Max-value Entropy Search for Efficient Bayesian Optimization," *arXiv e-prints*, p. arXiv:1703.01968, Mar. 2017.

[6] S. W. Ober, C. E. Rasmussen, and M. van der Wilk, "The Promises and Pitfalls of Deep Kernel Learning," *arXiv e-prints*, p. arXiv:2102.12108, Feb. 2021.

[7] R. Chandra, R. Chen, and J. Simmons, "Bayesian neural networks via MCMC: a Python-based tutorial," *arXiv e-prints*, p. arXiv:2304.02595, Apr. 2023.

[8] N. K. Vishnoi, "An Introduction to Hamiltonian Monte Carlo Method for Sampling," *arXiv e-prints*, p. arXiv:2108.12107, Aug. 2021.

[9] B. Lakshminarayanan, A. Pritzel, and C. Blundell, "Simple and Scalable Predictive Uncertainty Estimation using Deep Ensembles," *arXiv e-prints*, p. arXiv:1612.01474, Dec. 2016.

[10] J. Mockus, V. Tiesis, and A. Zilinskas, *The application of Bayesian methods for seeking the extremum*, 09 2014, vol. 2, pp. 117–129.

[11] Z. Fan, W. Wang, S. H. Ng, and Q. Hu, "Minimizing UCB: a Better Local Search Strategy in Local Bayesian Optimization," *arXiv e-prints*, p. arXiv:2405.15285, May 2024.

[12] M. Binois and N. Wycoff, "A survey on high-dimensional Gaussian process modeling with application to Bayesian optimization," *arXiv e-prints*, p. arXiv:2111.05040, Nov. 2021.

[13] M. Magris and A. Iosifidis, "Bayesian Learning for Neural Networks: an algorithmic survey," *arXiv e-prints*, p. arXiv:2211.11865, Nov. 2022.

[14] J. Lee, Y. Bahri, R. Novak, S. S. Schoenholz, J. Pennington, and J. Sohl-Dickstein, "Deep Neural Networks as Gaussian Processes," *arXiv e-prints*, p. arXiv:1711.00165, Oct. 2017.

[15] R. H. Byrd, P. Lu, J. Nocedal, and C. Zhu, "A limited memory algorithm for bound constrained optimization," *SIAM Journal on Scientific Computing*, vol. 16, no. 5, pp. 1190–1208, 1995. [Online]. Available: https://doi.org/10.1137/0916069

[16] M. Balandat, B. Karrer, D. R. Jiang, S. Daulton, B. Letham, A. G. Wilson, and E. Bakshy, "BoTorch: A Framework for Efficient Monte-Carlo Bayesian Optimization," *arXiv e-prints*, p. arXiv:1910.06403, Oct. 2019.

[17] Q. Lu, L. D. González, R. Kumar, and V. M. Zavala, "Bayesian optimization with reference models: A case study in mpc for hvac central plants," *Computers I& Chemical Engineering*, vol. 154, p. 107491, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0098135421002696