Using Bayesian Optimization for yielding better results in machine learning models

Email: qwluo@umich.edu

Chloe Luo
College of Literature, Science, and the Arts
University of Michigan, Ann Arbor, MI, 48104, USA

Abstract—Bayesian Optimization is a method often used to solve black-box optimization problems where the analytical expression or derivatives of the optimization function f(x) is unavailable. Bayesian Optimization uses the surrogate model to approximate the objective function and uses the acquisition function to draw samples based on the current observation to achieve better performance possible. In this work, we look into the general methods and steps of Bayesian optimization and conduct several experiments using Bayesian optimization to tune the hyperparameters of machine learning models on sklearn datasets. Experiment results show that Bayesian optimization generally yields better results than the random search method.

Keywords-component: Bayesian Optimization, hyperparameters , surrogate model, acquisition function

I. INTRODUCTION

In this section, we introduce the background of this research and emphasize the importance and applications of Bayesian optimization.

A. Black Box Optimization Problem

The black box optimization problem is a kind of optimization problem where the optimization function f(x) is not explicitly known, including its analytical expression or its derivatives. Informally, a black-box function can be seen as a mapping from the input space to the output space. We can only guess the structure of a black-box function by continuously feeding data into it and obtaining the corresponding output values. However, guessing and computing may cost a large amount of time because the black-box function is usually computationally expensive.

More specifically, Black-box optimization has the following characteristics: (a) Optimization objective: there is no explicit function expression, and no information about its gradient; its value domain distribution is usually a complex, non-convex, and non-concave function with multiple peaks. (b) Function space: the function space is usually composed of continuous, discrete integer values and is often of high dimension in practical applications. (c) Optimization objective solution: the calculation of a solution is usually very time-consuming; in large-scale deep learning training, it may take several days or even weeks to do one hyperparameter search.

Black-box optimization has wide applications. For example, the tuning of hyperparameters of a machine learning model is a typical black-box optimization problem. Similarly, in deep reinforcement learning, the actions of the agent to interact with the environment can also be modeled as a black-box optimization problem. Considering the wide range of use of the black-box optimization problem, the algorithms to better solve this problem are a hotspot of research.

There are several ways to solve the black box optimization problem including grid search, random search, and Bayesian optimization.

Grid search is the most basic black-box optimization method. The method creates a finite set of values for each parameter to be optimized and then evaluates the performance on a grid composed of Cartesian products of these parameters. Since the number of evaluations required increases exponentially with the number of parameters, this approach is difficult to use in high-dimensional black-box optimization situations where the number of parameters is large. In addition, if we obtain a very dense grid (the base of the set of values corresponding to each parameter is too large), it can also be computationally expensive even for a small number of parameters.

A simple alternative to grid search is the random search method. As the name implies, random search evaluates the performance based on a random selection of possible values of the parameters until we run out of computational resources. When some of the parameters are much more important than others, the random search algorithm is often more effective than the grid search method. When we evaluate the performance for B times for a total of N parameters, grid search can only evaluate the performance of each parameter at $B^{\frac{1}{N}}$ locations, while the random search can evaluate the performance of each parameter for B different values.

Since the random search algorithm does not make any assumptions about the machine learning model and is able to approximate the optimal solution of the model given sufficient computational resources, it is a useful baseline for evaluating a black-box optimization algorithm. Random search algorithms are often combined with more complex optimization algorithms to improve the convergence rate of the algorithm and to increase the exploratory nature of the model. Because random optimization algorithms can search the entire parameter space, they are often used to initialize more complex algorithms. However, when the computational resource is scarce, the random search algorithm may not yield satisfactory results. Therefore, effective algorithms for black-box optimization problems are still needed and the Bayesian optimization method is currently the most used method for the black-box optimization problem.

B. Bayesian Optimization

In recent years, Bayesian optimization has been more and more widely used in solving black-box function problems and has become a mainstream method for hyperparametric optimization. Bayesian optimization is a method that uses Bayes' theorem to guide the search to find the minimum or maximum value of the objective function, that is, in each iteration, using the previously observed historical information (a priori knowledge) to carry out the next optimization. In simpler terms, when carrying out an iteration, the method will first review the results of the previous iteration and then draw samples near the area where better performance is obtained because it is more likely to get good results in locations where the previous results are already good. In this way, the efficiency of the search is greatly improved.

More specifically, the advantages of applying Bayesian optimization are as follows: It is a global optimization approach in which the objective function only needs to satisfy local smoothness assumptions such as consistent continuity or Lipschitz (Lipschitz) continuity. It has the ability to obtain approximate solutions of complex objective functions with a smaller number of evaluations. The acquisition function of Bayesian optimization can do effective exploration and exploitation.

To find the global optimal solution as quickly as possible, Bayesian optimization solves the problem in two steps. Step 1 is to learn an agent model, also known as the surrogate model. Step 2 is to decide the next acquisition point by using the acquisition function.

The surrogate model defines prior over functions which can be used to incorporate prior beliefs about the objective function. A popular choice of surrogate models is the Gaussian processes (GPs) whose posterior is cheap to evaluate.

The acquisition functions are used to propose sampling points in the search space. They usually trade off exploitation and exploration. Exploitation means sampling at locations where the objective is predicted high by the surrogate model and exploration means sampling at locations where the prediction uncertainty is high. In Bayesian optimization, the goal is to maximize the acquisition function to determine the next sampling point.

Bayesian optimization is relatively cheap to conduct and can guarantee a not bad result under the limited computational resources condition. It is the most popular method used to solve the black-box optimization problem. The details of Bayesian optimization are presented in the next section.

II. Method

In this section, we give a more formal and detailed description of the Bayesian optimization method.

In the mathematical process of Bayesian optimization, we perform the following steps.

1: for t = 1,2,... do

2: Find x_t by optimizing the acquisition function over the GP:

$$x_t = \underset{x}{argmax}\, u(x|D_{1:x-1})$$

3: Sample the objective function:

$$y_t = f(x_t) + \varepsilon_t$$

4: Augment the data $D_{1:t} = \{D_{1:t-1}, (x_t: y_t)\}$ and update the GP.

5: end for

III. Experiments

In this work, we mainly focus on the task of hyperparameter tuning for machine learning models. We conduct three experiments on three datasets from the scikit-learn library with three different kinds of machine learning tasks and models. We optimize the hyperparameters of the machine learning models using Bayesian optimization and random search respectively and compare their performances.

A. Experiment setting

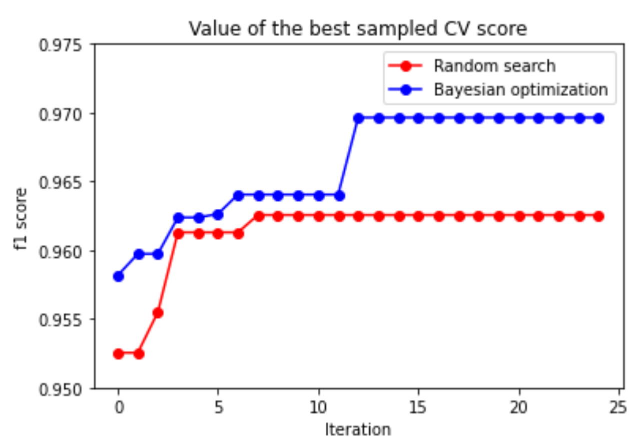
In the experiments, we try two Bayesian optimization libraries, one is the GPyOpt and the other is BayesianOptimization. We use python 3.7, sklearn 0.21.2.

To better evaluate the performance of Bayesian optimization on tuning machine learning model parameters, we conduct a series of experiments on different datasets with different tasks using different machine learning models with different numbers of hyperparameters to tune. We use the random search method as the baseline and compare their performance.

We use the breast cancer dataset from sklearn and use a random forest classifier to solve the binary classification task. Its evaluation criterion is the f1 score. We use the diabetes dataset from sklearn and use xgboost to solve the regression task. Its evaluation criterion is the negative mean squared error. We use the iris cancer dataset from sklearn and use a svm classifier to solve the multi-class classification task. Its evaluation criterion is the micro f1 score.

B. Experiment result

The results of the three experiments are shown below:



Identify applicable sponsor/s here.

Figure 1: results of breast cancer dataset

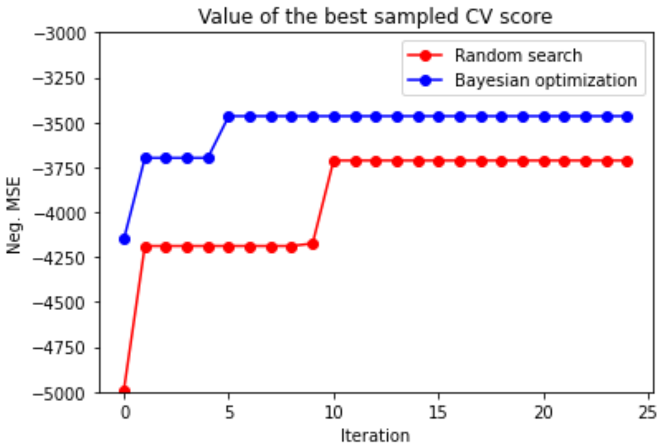


Figure 2: Results of diabetes dataset

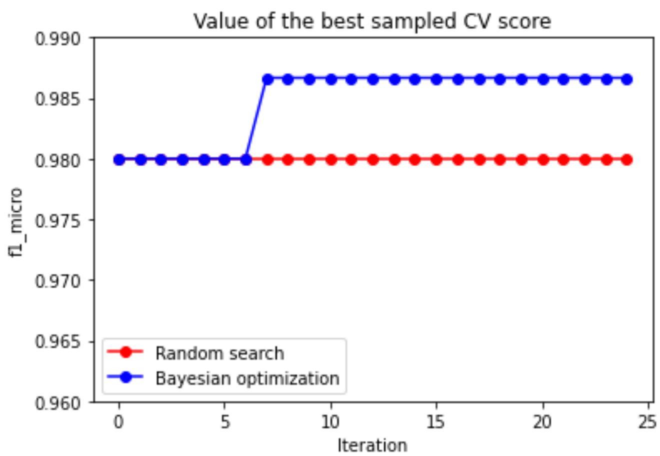


Figure 3: Results of the iris dataset

From the experiment results, we can see that despite the different task settings, models, and the number of parameters, the performance of Bayesian optimization is better than random search.

Besides, we also conduct a series of experiments where we use Bayesian Optimization to tune multiple machine learning models including SVM, random forest, logistic regression, and k nearest neighbors on the binary classification task on the cancer dataset.

We then present the best results after the tuning iterations.

Table 1: Results of models on the cancer dataset

Model	F1 score
SVM	0.771
Random Forest	0.971
Logistic Regression	0.963
K Nearest Neighbors	0.953

IV. CHALLENGES

Bayesian optimization is by far the most used method in the black-box optimization problem. However, there are still some difficulties and challenges in using Bayesian optimization.

The first challenge is to choose the surrogate model. In practice, GPs is mostly chosen. However, the complexity of GPs becomes higher when the dimension of the problem becomes higher. How to solve a high-dimensional Bayesian optimization problem is worth exploring and studying.

The second challenge is the acquisition function. When choosing the acquisition function, one of the problems is that we cannot know for sure if this acquisition function is optimal with respect to our surrogate model and the problem we want to optimize.

The third challenge is that the use of Bayesian optimization should be related to the specific optimization problem we want to solve. When applying Bayesian optimization to solve practical problems, we have to choose the appropriate surrogate model and the acquisition function based on the problem setting and the relevant domain knowledge. There is no "one trick fits all".

V. CONCLUSION

In this work, we examine the Bayesian optimization method often used to solve the black-box optimization problem. We give its method and steps and conduct several experiments on hyperparameter tuning for three different machine learning models on datasets from the scikit-learn library. Results show that, compared to the random search method, Bayesian optimization generally achieves better results and is a more effective method for black-box optimization.

ACKNOWLEDGMENT

REFERENCES

1. Turner, R., Eriksson, D., McCourt, M., Kiili, J., Laaksonen, E., Xu, Z., & Guyon, I. (2021, August). Bayesian optimization is superior to random search for machine learning hyperparameter tuning: Analysis of the black-box optimization challenge 2020. In NeurIPS 2020 Competition and Demonstration Track (pp. 3-26). PMLR.
2. Snoek, J., Larochelle, H., & Adams, R. P. (2012). Practical bayesian optimization of machine learning algorithms. Advances in neural information processing systems, 25.
3. Elgeldawi, E., Sayed, A., Galal, A. R., & Zaki, A. M. (2021, November). Hyperparameter tuning for machine learning algorithms used for Arabic sentiment analysis. In Informatics (Vol. 8, No. 4, p. 79). MDPI.
4. [1]"Bayesian optimization - Martin Krasser's Blog," *krasserm.github.io*. http://krasserm.github.io/2018/03/21/bayesian-optimization/
5. [1]J. Snoek, H. Larochelle, and R. P. Adams, "Practical Bayesian Optimization of Machine Learning Algorithms," *arXiv:1206.2944 [cs, stat]*, Aug. 2012, [Online]. Available: https://arxiv.org/abs/1206.2944

6. [1]J. Wu, X.-Y. Chen, H. Zhang, L.-D. Xiong, H. Lei, and S.-H. Deng, “Hyperparameter Optimization for Machine Learning Models Based on Bayesian Optimizationb,” *Journal of Electronic Science and Technology*, vol. 17, no. 1, pp. 26–40, Mar. 2019, doi: 10.11989/JEST.1674-862X.80904120.
7. [1]K. Swersky, J. Snoek, and R. P. Adams, “Multi-Task Bayesian Optimization,” *Neural Information Processing Systems*, 2013. https://proceedings.neurips.cc/paper/2013/hash/f33ba15effa5c10e873bf3842afb46a6-Abstract.html (accessed Sep. 25, 2022).
8. [1]B. Shahriari, K. Swersky, Z. Wang, R. P. Adams, and N. de Freitas, “Taking the Human Out of the Loop: A Review of Bayesian Optimization,” *Proceedings of the IEEE*, vol. 104, no. 1, pp. 148–175, Jan. 2016, doi: 10.1109/jproc.2015.2494218.
9. [1]A. Klein, S. Falkner, S. Bartels, P. Hennig, and F. Hutter, “Fast Bayesian Optimization of Machine Learning Hyperparameters on Large Datasets,” *proceedings.mlr.press*, Apr. 10, 2017. http://proceedings.mlr.press/v54/klein17a.html (accessed Sep. 25, 2022).
10. [1]A. Agnihotri and N. Batra, “Exploring Bayesian Optimization,” *Distill*, vol. 5, no. 5, May 2020, doi: 10.23915/distill.00026.