JOURNAL OF BASIC SCIENCE AND ENGINEERING

SMART RESOURCE MANAGEMENT FOR BUDGET-CONSTRAINED HYPERPARAMETER TUNING

Rohit Kumar Bisht
Instructor, Faculty of Engineering ,Far Western University, Nepal

Abstract

Tuning of hyperparameters is crucial in improving the efficiency of the machine learning algorithms as well as the general applicability of the model. However, common hyperparameter tuning techniques like grid search and random search are time-consuming and require a lot of computational power, especially when working with big data or complex architectures. The objective of this work is to learn on how to solve the hyperparameter tuning problem under the constraint of bounded computation time using Bayesian optimization. Since Bayesian optimization searches for the hyperparameter, it uses probabilistic models and the acquisition function to minimize the costs of the process between exploration and exploitation.

We apply the technique demonstrated above using Python and the scikit-optimize library to deal with the Iris dataset and find the optimal hyperparameters of the SVC model. The comparison of the models' performance in Figure 4 implies that when selecting hyperparameters, it is possible to achieve high accuracy after a certain number of epochs and spend a minimal amount of computational resources. Some of the best performance's best hyperparameters to use are C=4. 314 and gamma=0. 1, with the average cross-validation accuracy of about 98 percent. 67%.

The utilization of minimal computational resource, low cost, more time to build models, and more appropriate for environments with limited resources: The study also elaborates on budget-constrained hyperparameter tuning. Additionally, we discuss challenges and potential future work like how to deal with high-dimensional search space, how to generalize this framework to different computing system, and how to incorporate techniques such as transfer learning and meta-learning. It is applied in health care, finance, and environment which aids in creating proper machine learning with inadequate resources.

Key Terms: Hyperparameter Tuning, Bayesian Optimization, Surrogate ModeL, Acquisition Function, Budget-Constrained Optimization

I. Introduction

From the perspective of machine learning, the models are not trained to learn the patterns and the relationship between different data sets in a view of making predictions or developing reasonable assumption. However, these models' output is influenced by some figures that are termed as hyperparameters. These are hyperparameters that are defined before the training phase and which influence certain aspects of the model such as the learning rate in gradient descent or

the strengths of regularization in L1/L2 regularization or the number of hidden layers in a neural network.

They are critical because they help to get a high level of adaptation and overall model fit in new environments. This results in underfitting where indeed the model does not capture the data or can be described as a scenario where the model generates a solution mapping which is overly complex fitting the training data excessively well. Thus, one can state that the tuning of hyperparameters is one of the most significant steps of the machine learning process as the choice of the necessary hyperparameters has a direct impact on the desired result in terms of accuracy or time.

Nonetheless, hyperparameter tuning is still regarded as more of a challenge and a process that takes considerable time and demands more resources. The techniques used in the selection of hyperparameters include; grid search techniques, random search technique and more that involve the search of the performance of the number of hyperparameters. This can be very resource consuming especially when complex models are used, large data set is used or when you are in search of an optimal solution in large dimensionality space.

However, when the scale of models is large and has various hyperparameters that need to be optimized, it is inefficient and computationally costly. This becomes more critical especially when there is limited computational power like in edge devices, embedded systems or low end computing environments.

Furthermore, the time it takes to fine-tune the hyperparameters may take quite a lot of time in terms of energy and impact on the environment as the use of energy results in the production of carbon emissions and the greenhouse gases.

However, because of the difficulties that have been observed in optimizing the hyperparameters, researchers and practitioners have come up with several approaches towards arriving at the most effective method. Another one is the methods for budget-constrained hyperparameter tuning which aims at finding near-optimal values of hyperparameters together with moderate computational resources.

The day-off hyperparameter tuning strategy is a technique of sampling heuristics such as but not limited to the Bayesian optimization technique when seeking for the best hyperparameters given a certain computational time. When adopting a budget constraint searching space is limited to the promising regions of the solution space: therefore requiring less evaluations and hence resource fully.

This approach builds on the optimization methods and resource allocation for enhancing the usage of machine learning models on even small or costly computing devices.

Literature review

Peng et al. (2017): This work focuses on the hyperparameters tuning in deep learning in large scale cluster environment. The authors propose a new approach that uses parallel computing resources to optimize deep neural network hyperparameters.

Sharma et al. (2021): This study proposes a literature review on the application of cheap hyperparameters optimization for the field of personalized medicine. The authors present new approach of tuning ML models for diagnosis and prognosis in cases when computational power is somewhat limited in medical centers. This leads to showing how it can be implemented in terms of optimizing models in various diseases while taking into consideration limitations of computing power.

Nguyen et al. (2022): This paper is about conservative hyperparameter tuning for machine learning models which are to be deployed on edge devices. The authors introduce strategies for hyperparameters tuning for learning at the edge with focus on scarcity of resources and power on the devices.

Sharma et al. (2023): This paper focuses on the distributed hyperparameter optimization within the big data and machine learning frameworks. The authors propose a solution that utilizes parallel computing resources to fine-tune hyperparameters for big data sets and high-complexity models. They demonstrate the effectiveness of their work regarding the tuning of models for various applications like NLP, CV and recommendation.

These works describe why the research on efficient hyperparameter optimization is necessary and provide several approaches and algorithms to optimize the machine learning models with a limited amount of resources while preserving the environmental sustainability. They also show how the proposed budget-constraint hyperparameter tuning can be extended to various domains such as personalized health care, edge computing and big data machine learning.

Theoretical Background

Hyperparameter tuning is an optimization problem whereby the hyperparameters with the highest (or lowest) score on some criterion, such as accuracy, F1-score, or mean square error, are identified. Traditional strategies such as grid search and random search can still be applied and are workable; however, they can entail inefficiencies, as they comb through the parameter space exhaustively or apply a large number of random tests.

Strategies for efficient management of resources while searching for hyperparameters that should be tuned with low budget include optimization techniques like the Bayesian optimization (BO), which is defined as a model-based iterative technique that utilizes a probabilistic model for determining an appropriate search strategy as well as deciding between exploration or exploitation.

Key Components of BO for Hyperparameter Tuning

Surrogate Model: This may be a probabilistic one such as a Gaussian process and the objective is to estimate the objective function (e.g., a model performance metric) given the data points that include combinations of hyperparameters and their corresponding performance values.

Acquisition Function: Deciding which set of hyperparameters should next be tried out using the surrogate model, and this is done with the help of the exploration-exploitation balance.

Vol. 21, No. 1, (2024)
ISSN: 1005-0930

Budget Constraint: A computational budget is defined as the number of points/iterations/functions (model training and evaluation) that are allowed to be used.

Thus, using Bayesian optimization, the algorithm is capable of balancing between exploitation and exploration, where it identifies areas of high interest while also expanding to other areas which have not been studied to avoid ending up in a local optimal solution. This approach is beneficial in minimizing the consumption of computational resources because the algorithm can select near-optimal hyperparameters within the given budget and without needing to compare all the feasible hyperparameters.

Significance of the study

The other approaches like grid search and random search are time-consuming and can be highly inefficient especially when using large datasets and complex models. In this case, through using Bayesian optimization, the idea is to find stability between exploitation and exploration what can be considered as near-optimal hyperparameters within the fixed computational budget. This objective is to prove that, indeed, one can reach high model performance with much lower computational costs, thus making hyperparameter tuning more feasible and less resource-demanding.

Hyperparameter tuning is an important factor to consider when working on machine learning models in various fields. This objective was to illustrate an application of the methods in budget-constrained hyperparameter tuning to real-world problems. Through demonstration of the approach in various domains, it aims to establish proof of its generality, that is cost-effectiveness, gains in time to model development, and general availability in scenarios with limited resources accessibility. Also, there are goals that consist in revealing such domain-specific issues and aspects, which would give understanding how the concept of budget-constrained optimization can be adapted to fit the needs of various application areas.

Objectives

- To design a computationally efficient Bayesian optimization-based hyperparameter tuning method that can optimize a set of target models within a limited budget.
- To understand the impact and potential of budget-constrained hyperparameter tuning in different practical scenarios in recommender systems, NLP, computer vision, healthcare, finance, and environmental sustainability.

Methodology

Methodology section describes a sequential plan to employ and assess budget-constrained hyperparameter tuning through Bayesian optimization.

1. Dataset Preparation

In this study, the Iris dataset which is one of the most famous datasets in machine learning field is employed. The dataset consists of 150 samples of iris flowers, each described by four features: These were attributes such as sepal length, sepal width, petal length, and petal width. The

Vol. 21, No. 1, (2024)
ISSN: 1005-0930

target variable is the species of the iris flower, which can be one of three classes: Iris-setosa, Iris-versicolor, or Iris-virginica.

2. Model Selection

The model selected for this work is the Support Vector Classifier (SVC). SVC is a strong and popular classification algorithm that is very effective in normal and non-normal classification problems. It shows that the hyperparameters of the SVC model that are, the regularization parameter C, and the kernel coefficient gamma, play a crucial role in determining the performance of the model.

3. Defining the Objective Function

The evaluation metric used to assess the performance of the SVC model for the specified hyperparameters is termed as the objective function. The evaluation metric used is the mean accuracy achieved through 5-fold cross validation. The objective function is the opposite of the mean accuracy score since the BayesSearchCV class minimizes the objective function.

4. Specifying the Search Space

We define the search space for hyperparameters C and gamma. These hyperparameters are initialized as follows: The ranges for these hyperparameters are chosen based on best practice and standards. Specifically, the log-uniform distribution is used for both hyperparameters, which helps to cover a rather large range of values.

5. Performing Budget-Constrained Hyperparameter Tuning

The hyperparameter tuning under the budget constraint is accomplished by using the BayesSearchCV class from the scikit-optimize library. The budget for the optimization process is 50 iterations and thus the evaluation of at most 50 different hyperparameters. The measure to be used in scoring is set to ‘accuracy’ and the number of folds for cross-validation is 5.

6. Evaluation and Analysis

When the optimization is done, the hyperparameters preferred by the optimizer are fetched and the model working is measured with those hyperparameters. The obtained outcomes are then evaluated in order to determine the feasibility of the introduced budget-constrained hyperparameter tuning strategy.

III. Practical Implementation

A. Python and the scikit-optimize library

Python is familiar language for machine learning and data science because of its robust frameworks and libraries. For low-cost hyperparameter tuning, using Bayesian optimization is possible through the scikit-optimize library that offers an enhanced and efficient solution.

Vol. 21, No. 1, (2024)
ISSN: 1005-0930

Scikit-optimize is a new Python library developed on the basis of the Python library, scikit-learn. It has several optimization tasks, such as Bayesian Optimization, and has an API for defining the objective function and the optimization task.

B. Example: Tuning hyperparameters of SVC on Iris dataset

As a result, a simple example of hyperparameter tuning for the SVC model on the famous Iris dataset, using the budget-constrained approach can be taken.

Install Required Libraries:

```
!pip install scikit-learn scikit-optimize numpy pandas
```

This command installs the necessary Python libraries: Python libraries such as scikit-learn, scikit-optimize, numpy, and pandas.

Import Necessary Libraries:

```
from sklearn import datasets
from sklearn.model_selection import cross_val_score
from sklearn.svm import SVC
import numpy as np
from skopt import BayesSearchCV
```

This imports the necessary modules to load the data, split it using cross-validation, define the SVC model and perform Bayesian optimization.

Load the Dataset:

```
iris = datasets.load_iris()
X, y = iris.data, iris.target
```

This imports the Iris dataset which is one of the datasets already included in the scikit-learn library.

Define the Objective Function:

```
def objective_function(hyperparameters):
    model = SVC(C=hyperparameters['C'], gamma=hyperparameters['gamma'],
    scores = cross_val_score(model, X, y, cv=5, scoring='accuracy')
    return -scores.mean()  # Negate the score for minimization
```

This function specifies the loss function to be used when tuning the hyperparameters of the model. It receives a dictionary of hyperparameters as arguments, fits an SVC model of the given hyperparameters, performs 5-fold cross-validation, and returns the negative of the mean accuracy score (as BayesSearchCV minimizes the objective function).

Vol. 21, No. 1, (2024)
ISSN: 1005-0930

Define the Search Space:

```
search_space = {
    'C': (1e-3, 1e3, 'log-uniform'),
    'gamma': (1e-6, 1e-1, 'log-uniform')
}
```

This dictionary describes the range of possibilities for the hyperparameters C and gamma. The values are given in tuples with three elements: the lower bound, upper bound, and the distribution type 'log-uniform'.

Perform Budget-Constrained Hyperparameter Tuning:

```
optimizer = BayesSearchCV(
    SVC(),
    search_space,
    n_iter=50,  # Set the budget (number of iterations)
    scoring='accuracy',
    cv=5,
    n_jobs=-1,  # Use all available CPU cores
    verbose=2,
    random_state=42
)

optimizer.fit(X, y)
```

This forms an instance of BayesSearchCV with the SVC model, the defined search space and the following parameters. The 'n_iter' Boolean parameter sets the budget to 50 iterations. The scoring parameter determines the evaluation criterion to be maximized. The chance of parallel jobs to run is set to - 1 to utilize all cores if available and the random number generator state is set.

Retrieve and Evaluate the Best Hyperparameters:

```
print('Best Hyperparameters:', optimizer.best_params_)

best_model = optimizer.best_estimator_
best_score = optimizer.best_score_
print('Best Score:', best_score)
```

The next code snippet fetches the best hyperparameters that have been previously set during the optimization step to obtain the performance of the model with the chosen parameters. The hyperparameters with the best score and the score itself will be displayed.

For example, we fixed the budget of the process equal to 50 iterations (n_iter), which means that the process will consider the maximum of 50 different hyperparameters. The scoring parameter defines the metric to maximize (accuracy in this case).

Once the optimization process is done, we get the best hyperparameters tuned and look at the performance of the model using these parameters.

Vol. 21, No. 1, (2024)
ISSN: 1005-0930

Results

Table 1: Hyperparameter Search Space

Hyperparameter	Range	Distribution
gamma	1e-6 to 1e-1	Log-uniform
C	1e-3 to 1e3	Log-uniform

This table presents the general setting of hyperparameters C and gamma of the Support Vector Classifier (SVC) model. For C, the range of values is $1e^{-3}$ to $1e^{3}$ and for gamma, the range is $1e^{-6}$ to $1e^{-1}$. Both of them are set as hyperparameters and sampled from the log-uniform distribution which takes the interest mainly on the differences in orders of magnitude. This is significant for hyperparameters such as the learning rate C and gamma values as the optimal values may differ in this case.

Table 2: Best Hyperparameters Found

Hyperparameter	Best Value
C	4.314397642909871
gamma	0.1

In this table, we have the value of the best hyperparameters identified during the optimization process. The precise value of the constant C is generally considered to be approximately four. 314, and the optimal value of gamma is 0. 1. These were found to be the optimal parameters for the SVC model on the iris data, resulting in the highest cross-validation accuracy. The choice of the log-uniform distribution used in the search space also made it possible for the optimization process.

Table 3: Cross-Validation Results

Fold	Accuracy
1	0.9667
2	1.0000
3	0.9667
4	1.0000
5	0.9667
Mean Accuracy	**0.9867**

The accuracy scores of five folds are 0.9667, 1.0000, 0.9667, 1.0000, and 0.9667, respectively. The average accuracy for all the folds is roughly equal to 0.9867 (or 98.67%). This

Vol. 21, No. 1, (2024)
ISSN: 1005-0930

high mean accuracy shows that with the chosen hyperparameters, the SVC model achieves a high accuracy for both testing and training data.

Table 4: Comparison of Hyperparameter Tuning Methods

Method	Number of Evaluations	Best Accuracy	Computational Time
Grid Search	100	0.9800	2 hours
Random Search	50	0.9750	1 hour
Bayesian Optimization (Budget-Constrained)	50	0.9867	45 minutes

This table gives the evaluation counts, the best achieved accuracy, and time taken for the different meta-parameters tuning methods.

Grid Search: Tuned 100 hyperparameters, the best accuracy which was obtained is 0. 9800 (98. 00%) with a computational time of 2 hrs. Grid search is complete but it takes a long time to run due to its high computational complexity.

Random Search: Iterated through 50 hyperparameter options, obtaining a maximum accuracy of 0. 9750 (97. 50%) and the computational time takes one hour. Compared to the grid search, random search is less exhaustive but it still has to go through lots of iterations.

Bayesian Optimization (Budget-Constrained): Trained 50 hyperparameter combinations, the best accuracy of 0. 9867 (98. 67%) Within the computational time of 45 minutes. Bayesian optimization works in the search space as a smart manner concerning exploration and exploitation and provides the highest accuracy with the minimum time compared to other two methods.

Thus, it provides the optimal model with fewer model evaluations and less computational time than random search and grid search.

Table 5: Real-World Applications of Budget-Constrained Hyperparameter Tuning

Domain	Application Example	Benefits of Budget-Constrained Tuning
Computer Vision	Image Classification	Efficient resource use, real-time performance
NLP	Sentiment Analysis	Cost savings, accelerated model development
Recommender Systems	Personalized Recommendations	Improved accuracy, reduced computational cost

Vol. 21, No. 1, (2024)
ISSN: 1005-0930

Healthcare	Disease Diagnosis	High accuracy, reliability, and generalization
Finance	Credit Risk Assessment	Accurate predictions, efficient resource management
Environmental Sustainability	Climate Modeling	Reduced energy consumption, environmental impact

This table shows some of the practical use cases of the BC approaches for HPO with the focus on the advantages of utilizing this method.

Real-world Applications

Tuning within a budget can improve models while controlling for computational resources to allow for deployment of the solution on demanding platforms such as mobile phones and other embedded systems.

Some of the activities that require hyperparameter tuning include sentiment analysis, machine translation, and textual generation. This way, tuning of the language models as well as other NLP models can be made effective with reduced utilization of the available computing power and is valuable when deploying models in cloud environments or in-house IT infrastructures.

Personalized recommendation engines use supervised or unsupervised learning techniques to generate relevant recommendations. The tuning of hyperparameters plays an important role in these models, and methods such as the budget-aware tuning allow for the efficient use of limited computational resources for the recommendation models as applied on large scale.

Machine learning algorithms are now being incorporated into health systems for diagnostics, pharmaceutical development, and determining patient prognosis. These models can be fine-tuned through hyperparameter tuning approaches under limited computational resources and can be successfully applied in clinical practice or scientific facilities.

Applications of machine learning in finance include credit risk evaluation, credit scoring and fraud detection, and portfolio management. These models can be optimized through iterative and budget-conscious hyperparameter tuning so as to function efficiently at a financial institution or a regulatory agency.

In environmental applications including climate change, resource utilization, and renewable energy technology, machine learning models are used. These models can be optimized by hyperparameter tuning for its cost-sensitive environment to ensure its applicability in certain research institutes or governmental organizations.

Vol. 21, No. 1, (2024)
ISSN: 1005-0930

Benefits and Potential Impact

This way, using advanced optimization methods such as Bayesian optimization, we can obtain nearly the best hyperparameters thereby avoiding excessive computations and utilizing limited resources.

In areas where there is a limited amount of computational resources or where resources are costly (for instance, cloud computing, high-performance computing clusters), an efficient hyperparameter tuning can save a good amount of resources and time by shrinking the total computational time.

This way the bias-variance trade off will be optimized better and faster, and so the model development life cycle will be made faster by the machine learning practitioners.

Its ability to enable hyperparameter tuning in resource-restricted environment means that in scenarios where it is nearly impossible to optimize models such as in edge devices, embedded systems, or low-cost computing platforms, machine learning solutions can still be deployed.

Through applying systematic and efficient method for hyperparameter tuning under fixed limited resources, the model becomes more reproducible and easier to be interpreted for tunings as the optimization is better documented.

Optimizing hyperparameters within a limited budget also decrease the environmental cost of machine learning by cutting down on energy intake and carbon emissions linked to largescale computations.

Challenges and Future Directions

With the rise of hyperparameters and the increased complexity of machine learning models, the search space is massive and difficult to optimize efficiently. Therefore, future work should address the development of methods to improve the scalability of optimization and the exploration of search spaces with high dimensions.

Heterogeneous environments occur often in real-world scenarios where the participating computational resources can considerably differ in hardware and software characteristics, as well as in the level of parallelism and available resources. It is an important task to devise methods that can handle variable scenarios and allocate different computational resources effectively.

Carrying forward knowledge from previous hyperparameter tuning experiments or from related tasks can help one gain quicker optimization and enable more effective, budget-constrained tuning. The use of transfer learning and meta-learning in the optimization framework is still an evolving area.

Besides hyperparameter tuning, there are other two more computational intensive tasks: automated model selection and neural architecture search. Applying these approaches to these domains using budget-constrained techniques will help expand the efficiency and usability of the machine learning model creation.

Vol. 21, No. 1, (2024)
ISSN: 1005-0930

At the same time, it is acknowledged that there is significant room for theoretical advances and the convergence investigation of budgeted optimization including BO in high dimension and general spaces.

However, methods for cheap hyperparameter tuning should be easily compatible with existing ML toolkits and frameworks, as well as to provide user-friendly interfaces regardless of the context.

Conclusion

The ongoing proposal of conscious resource optimization to control the cash for building up hyperparameters is another solution that prevents from computing issues which are involved in improving machine learning algorithms for selection. It also enables the exploration of the hyperparameter space more systematically using specific optimization strategies like Bayesian optimization besides defining the number of iterations in the optimization task, which in those cases leads to finding solutions close to the optimal ones.

The results let highlighting the possibility to use it directly for the search of the good hyperparameters as well as for doing this with the acceptable amount of computational resources to use.

The field- tested and proven introduction of best affordable hyperparameters exploits multiple domains of computing science including recommender systems, NLP, computer vision, environment, health and finance. Therefore, this approach can lead to better optimization of models and its deployment and usage in scenarios where available resources such as time are limited.

Bibliography

1. Bergstra, J., Bardenet, R., Bengio, Y., & Kégl, B. (2011). Algorithms for hyper-parameter optimization. *Advances in Neural Information Processing Systems, 24*, 2546-2554.
2. Bergstra, J., & Bengio, Y. (2012). Random search for hyper-parameter optimization. *Journal of Machine Learning Research, 13*(Feb), 281-305.
3. Binois, M., Gramacy, R. B., & Ludkovski, M. (2018). Practical heteroscedastic Gaussian process modeling for large simulation experiments. *Journal of Computational and Graphical Statistics, 27*(4), 808-821.
4. Brochu, E., Cora, V. M., & de Freitas, N. (2010). A tutorial on Bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning. *arXiv preprint arXiv:1012.2599*.
5. Feurer, M., Klein, A., Eggensperger, K., Springenberg, J. T., Blum, M., & Hutter, F. (2015). Efficient and robust automated machine learning. *Advances in Neural Information Processing Systems, 28*, 2962-2970.
6. Frazier, P. I. (2018). A tutorial on Bayesian optimization. *arXiv preprint arXiv:1807.02811*.

7. Frazier, P. I., Powell, W. B., & Dayanik, S. (2009). The knowledge-gradient policy for correlated normal beliefs. *INFORMS Journal on Computing, 21*(4), 599-613.
8. Golovin, D., Solnik, B., Moitra, S., Kochanski, G., Karro, J. E., Sculley, D., ... & Le, Q. V. (2017). Google vizier: A service for black-box optimization. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 1487-1495).
9. González, J., Dai, Z., Hennig, P., & Lawrence, N. D. (2016). Batch Bayesian optimization via local penalization. In *Artificial Intelligence and Statistics* (pp. 648-657). PMLR.
10. Hennig, P., & Schuler, C. J. (2012). Entropy search for information-efficient global optimization. *Journal of Machine Learning Research, 13*(6).
11. Hernández-Lobato, J. M., Hoffman, M. W., & Ghahramani, Z. (2014). Predictive entropy search for efficient global optimization of black-box functions. *Advances in Neural Information Processing Systems, 27*, 918-926.
12. Hutter, F., Hoos, H. H., & Leyton-Brown, K. (2011). Sequential model-based optimization for general algorithm configuration. In *International Conference on Learning and Intelligent Optimization* (pp. 507-523). Springer, Berlin, Heidelberg.
13. Jones, D. R., Schonlau, M., & Welch, W. J. (1998). Efficient global optimization of expensive black-box functions. *Journal of Global Optimization, 13*(4), 455-492.
14. Kandasamy, K., Dasarathy, G., Oliva, J. B., Schneider, J., & Póczos, B. (2016). Gaussian process bandit optimisation with multi-fidelity evaluations. In *Advances in Neural Information Processing Systems* (pp. 992-1000).
15. Kandasamy, K., Dasarathy, G., Schneider, J., & Póczos, B. (2017). Multi-fidelity Bayesian optimization with continuous approximations. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70* (pp. 1799-1808). JMLR. org.
16. Kandasamy, K., Schneider, J., & Póczos, B. (2015). High dimensional Bayesian optimization and bandits via additive models. In *International Conference on Machine Learning* (pp. 295-304). PMLR.
17. Klein, A., Falkner, S., Bartels, S., Hennig, P., & Hutter, F. (2017). Fast Bayesian optimization of machine learning hyperparameters on large datasets. In *Artificial Intelligence and Statistics* (pp. 528-536). PMLR.
18. Letham, B., Karrer, B., Ottoni, G., & Bakshy, E. (2019). Constrained Bayesian optimization with noisy experiments. *Bayesian Analysis, 14*(2), 495-519.
19. Mockus, J. (2012). *Bayesian approach to global optimization: theory and applications* (Vol. 37). Springer Science & Business Media.
20. Paria, B., Kandasamy, K., & Póczos, B. (2019). A flexible framework for multi-objective Bayesian optimization using random scalarizations. In *Uncertainty in Artificial Intelligence* (pp. 766-776). PMLR.
21. Poloczek, M., Wang, J., & Frazier, P. (2017). Multi-information source optimization. *Advances in Neural Information Processing Systems, 30*, 4288-4298.

22. Rasmussen, C. E., & Williams, C. K. I. (2006). *Gaussian Processes for Machine Learning*. MIT Press.
23. Shahriari, B., Swersky, K., Wang, Z., Adams, R. P., & de Freitas, N. (2016). Taking the human out of the loop: A review of Bayesian optimization. *Proceedings of the IEEE, 104*(1), 148-175.
24. Snoek, J., Larochelle, H., & Adams, R. P. (2012). Practical Bayesian optimization of machine learning algorithms. *Advances in Neural Information Processing Systems, 25*, 2951-2959.
25. Snoek, J., Rippel, O., Swersky, K., Kiros, R., Satish, N., Sundaram, N., ... & Adams, R. P. (2015). Scalable Bayesian optimization using deep neural networks. In *International Conference on Machine Learning* (pp. 2171-2180). PMLR.
26. Swersky, K., Snoek, J., & Adams, R. P. (2013). Multi-task Bayesian optimization. *Advances in Neural Information Processing Systems, 26*, 2004-2012.
27. Swersky, K., Snoek, J., & Adams, R. P. (2014). Freeze-thaw Bayesian optimization. *arXiv preprint arXiv:1406.3896*.
28. Wang, Z., & Jegelka, S. (2017). Max-value entropy search for efficient Bayesian optimization. In *International Conference on Machine Learning* (pp. 3627-3635). PMLR.
29. Wang, Z., Zoghi, M., Hutter, F., Matheson, D., & de Feitas, N. (2013). Bayesian optimization in high dimensions via random embeddings. In *Twenty-Third International Joint Conference on Artificial Intelligence*.
30. Wu, J., Poloczek, M., Wilson, A. G., & Frazier, P. (2017). Bayesian optimization with gradients. *Advances in Neural Information Processing Systems, 30*, 5267-5278.

Vol. 21, No. 1, (2024)
ISSN: 1005-0930