Parameter Sensitivity Analysis for the Progressive Sampling-Based Bayesian Optimization Method for Automated Machine Learning Model Selection

Weipeng Zhou[0000-0003-1215-8043] and Gang Luo(✉)[0000-0001-7217-4008]

University of Washington, Seattle, WA 98195, USA
wzhou87@uw.edu, luogang@uw.edu

Abstract. As a key component of automating the entire process of applying machine learning to solve real-world problems, automated machine learning model selection is in great need. Many automated methods have been proposed for machine learning model selection, but their inefficiency poses a major problem for handling large data sets. To expedite automated machine learning model selection and lower its resource requirements, we developed a progressive sampling-based Bayesian optimization (PSBO) method to efficiently automate the selection of machine learning algorithms and hyper-parameter values. Our PSBO method showed good performance in our previous tests and has 20 parameters. Each parameter has its own default value and impacts our PSBO method's performance. It is unclear for each of these parameters, how much room for improvement there is over its default value, how sensitive our PSBO method's performance is to it, and what its safe range is. In this paper, we perform a sensitivity analysis of these 20 parameters to answer these questions. Our results show that these parameters' default values work well. There is not much room for improvement over them. Also, each of these parameters has a reasonably large safe range, within which our PSBO method's performance is insensitive to parameter value changes.

Keywords: Sensitivity analysis, automated machine learning model selection, progressive sampling, Bayesian optimization

1 Introduction

In machine learning, model selection refers to the selection of an effective combination of a machine learning algorithm and hyper-parameter values for a given supervised machine learning task [5]. As a crucial component of automating the entire process of applying machine learning to solve real-world problems, automated model selection is in great need, particularly by citizen data scientists with limited machine learning expertise. In the past few years, many methods, such as Auto-WEKA [8], Auto-sklearn [3], Vizier [1], and AutoGluon-Tabular [2], have been proposed for automating model selection, making it a current hot topic in computer science [5]. Recently, automated methods have outperformed human experts at model selection [1, 2]. Also, multiple major high-tech companies like Google have adopted automated methods as the default model selection approach for building various machine learning models [1]. Despite all of these exciting progresses, a major road blocker still exists in making this wonderful

technology widely accessible. Using existing methods to automate model selection on a large data set brings high financial and environmental costs. It often requires using several dozen to several hundred powerful servers continuously for several weeks or several months, incurs a computational cost of several million dollars, and leads to emission of several hundred tons of carbon dioxide due to the large amount of energy consumed [6]. Consequently, researchers and organizations with limited budgets cannot afford using automated model selection to build high-performance machine learning models on large data sets. To address this issue, we developed a progressive sampling-based Bayesian optimization (PSBO) method [9]. It can reduce automated model selection's execution overhead by two orders of magnitude and improve model accuracy at the same time. This is a major progress in green computing and in making automated model selection on large data sets affordable by researchers and organizations with limited budgets.

Our PSBO method has 20 parameters impacting its performance. Each parameter has its own default value that was set empirically in our prior paper [9]. It is unclear for each of these parameters, how much room for improvement there is over its default value, how sensitive our PSBO method's performance is to it, and what its safe range is. In this paper, we perform a sensitivity analysis of these 20 parameters to answer these questions. Our results show that these parameters' default values work well. There is not much room for improvement over them. Also, each of these parameters has a reasonably large safe range, within which our PSBO method's performance is insensitive to parameter value changes.

The rest of the paper is organized as follows. Section 2 reviews our PSBO method. Section 3 describes how we did the parameter sensitivity analysis for our PSBO method. Section 4 shows our experimental results. Section 5 concludes this paper.

2 Review of Our PSBO Method

In this section, we review our PSBO method that has 20 parameters. Table 1 lists these parameters, their definitions, and their default values.

Table 1. The definitions and default values of the parameters of our PSBO method.

Parameter	Definition	Default value
C_2	The number of cycles of Bayesian optimization done in the second round.	3
e	The number of random hyper-parameter value combinations, if any, that are tested for each machine learning algorithm in the first round.	20
h_{large}	On a large data set, the number of folds of cross validation done in the final round.	3
h_{small}	On a small data set, the number of folds of cross validation done in the final round.	10
k	The number of folds of progressive sampling used on a small data set. On a large data set, we take $k = 1$ and use 1-fold progressive sampling.	3

L_{f_large}	On a large data set, the maximum number of seconds allowed for doing feature selection when testing a combination of a machine learning algorithm, a feature selection technique, and hyper-parameter values on a fold in the first round of the search process.	20
L_{f_small}	On a small data set, the maximum number of seconds allowed for doing feature selection when testing a combination of a machine learning algorithm, a feature selection technique, and hyper-parameter values on a fold in the first round of the search process.	10
L_{t_large}	On a large data set, the maximum number of seconds allowed for model training when testing a combination of a machine learning algorithm, a feature selection technique, and hyper-parameter values on a fold in the first round of the search process.	20
L_{t_small}	On a small data set, the maximum number of seconds allowed for model training when testing a combination of a machine learning algorithm, a feature selection technique, and hyper-parameter values on a fold in the first round of the search process.	10
m	The minimum number of machine learning algorithms, if any, that we try to keep at the end of each round that is not the final round.	3
n	The number of new hyper-parameter value combinations chosen for testing in each cycle of Bayesian optimization.	10
n_c	For a round that is neither the first nor the last round and a machine learning algorithm, the number of hyper-parameter value combinations that were used in the prior round and are chosen for testing in this round.	10
p	The penalty weight given to a combination of a machine learning algorithm, a feature selection technique, and hyper-parameter values that uses feature selection.	1.1
q	For a combination of a meta or an ensemble machine learning algorithm and hyper-parameter values, the penalty weight given to every base algorithm used in the meta or ensemble algorithm.	0.02
r	The target percentage of machine learning algorithms kept at the end of the first round.	40%
s	In the fifth round, for every machine learning algorithm remaining from the prior round, the number of its top hyper-parameter value combinations chosen for testing.	10
t	The threshold on the product of the number of data instances and the number of features in the data set for deciding whether the data set is a large or a small one.	1,000,000
t_d	For a round that is neither the first nor the last round and a machine learning algorithm, the threshold on the minimum Hamming distance we try to keep among the hyper-	2

	parameter value combinations that were used in the prior round and are chosen for testing in this round.	
u	The upper bound on the total number of data instances in the training and validation samples in the fourth round.	5,000
w	The target percentage of machine learning algorithms kept at the end of a round that is neither the first nor the last round.	70%

2.1 Overview of Our PSBO Method

Given a data set including multiple features and a prediction target, a set of machine learning algorithms and feature selection techniques, and a hyper-parameter space, we perform model selection to find an effective combination of a machine learning algorithm, a feature selection technique, and hyper-parameter values to build a predictive model with a low error rate. Doing this search often requires testing thousands of combinations. Typical automated model selection methods test each such combination on the whole data set. This causes the search process to take a long time and incur a high computational cost, particularly if the data set is large. To reduce the search time and/or the computational cost, researchers have proposed various techniques such as early stopping [4] and distributed computing [7]. These techniques are helpful, but are insufficient for resolving the inefficiency issue. To address this problem, our PSBO method adopts the idea of doing fast tests on small samples of the data set to quickly remove most of the unpromising combinations, and then spending more time on adjusting the promising combinations to come up with the final combination.

More specifically, we do so-called k-fold progressive sampling and proceed in five rounds. In each of the first four rounds, for each fold, we use a training sample to train predictive models and a disjoint validation sample to assess each trained model's error rate. The training sample is initially small and keeps expanding over rounds. At the end of the round, we remove several unpromising machine learning algorithms and shrink the search space. In the fifth round, we identify the best combination of an algorithm, a feature selection technique, and hyper-parameter values to build the final predictive model on the whole data set.

In our PSBO method, we classify the data set as large or small based on whether the number of data instances times the number of features in the data set is over a threshold t. The choice of feature selection technique is treated as a hyper-parameter. We adopt a weight p to penalize the use of feature selection, and a weight q to penalize the use of a meta or an ensemble algorithm. The upper bound on the total number of data instances in the training and validation samples in the fourth round is u. In the following, we describe the five rounds one by one.

2.2 The First Round

In the first round, we start with a small training sample for each fold. For each machine learning algorithm, we test both its default and e random hyper-parameter value combinations, if any, and obtain their error rate estimates. When testing a combination

on a fold, we use the algorithm, the combination, and the training sample to build a predictive model and evaluate the model's error rate on the validation sample. During the test, we place a time limit L_f on feature selection and a time limit L_t on modeling training. On a large data set, we set $L_f = L_{f\_large}$ and $L_t = L_{t\_large}$. On a small data set, we set $L_f = L_{f\_small}$ and $L_t = L_{t\_small}$. At the end of the round, we keep the top r (percent) of algorithms having the smallest error rate estimates and remove the other algorithms. If the top r (percent) of algorithms have $<m$ algorithms, we try to keep the top m algorithms, if any.

2.3 The Second to the Fourth Round

In the second round, we increase L_f and L_t by 50%, expand the training sample for each fold, and do four things for each remaining machine learning algorithm. First, we choose n_c hyper-parameter value combinations used in the prior round and test them to obtain their revised error rate estimates. In the selection process, we strike a balance between two goals: a) the selected combinations are away from each other by at least a Hamming distance of t_d, and b) the selected combinations are the ones having the smallest error rate estimates in the prior round. Second, for each combination used in the prior round but not chosen for testing, we multiply its estimated error rate in the prior round by a calculated factor to come up with its revised error rate estimate for the current round. Third, we construct a regression model for estimating a combination's error rate. Fourth, we set $C = C_2$ and do C cycles of Bayesian optimization like that in Thornton *et al*. [8]. In each cycle, we select n new combinations and test them to obtain their error rate estimates. At the end of the second round, we keep the top w (percent) of algorithms having the smallest error rate estimates and remove the other algorithms. If the top w (percent) of algorithms have $<m$ algorithms, we try to keep the top m algorithms, if any.

The third and fourth rounds work in the same way as the second round, except that C is decreased by one per round.

2.4 The Fifth Round

In the fifth round, we increase L_f and L_t by 50%. For each remaining machine learning algorithm, we choose the top s hyper-parameter value combinations having the smallest error rate estimates in the prior round, if any. Then we do h-fold cross validation on at most u data instances to evaluate each (algorithm, top combination) pair and select the best pair to build the final predictive model on the whole data set. h is set to h_{large} or h_{small} depending on whether the data set is a large one or a small one.

3 Experimental Setup and Procedure

Table 2. The data sets used in the parameter sensitivity analysis for our PSBO method.

Name	No. of classes	No. of training instances	No. of test instances	No. of features

Mammographic Mass	2	673	288	6
Car	4	1,209	519	6
Shuttle	7	43,500	14,500	9
Madelon	2	1,820	780	500
Secom	2	1,096	471	591
Arcene	2	100	100	10,000
Waveform	3	3,500	1,500	40
Cardiotocography	3	1,488	638	23
Wine quality	11	3,425	1,469	11
Semeion	10	1,115	478	256
Yeast	10	1,038	446	8
Abalone	28	2,923	1,254	8
KR-vs-KP	2	2,237	959	37
Arrhythmia	16	316	136	279
German credit	2	700	300	20
Diabetic retinopathy debrecen	2	806	345	20
Amazon	49	1,050	450	10,000
Convex	2	8,000	50,000	784
KDD09-appentency	2	35,000	15,000	230
MNIST basic	10	12,000	50,000	784
Dexter	2	420	180	20,000
ROT. MNIST+BI	10	12,000	50,000	784
Parkinson speech	2	728	312	26
Gisette	2	4,900	2,100	5,000
CIFAR-10-small	10	10,000	10,000	3,072
Dorothea	2	805	345	100,000
CIFAR-10	10	50,000	10,000	3,072

In this section, we describe how we did the parameter sensitivity analysis for our PSBO method. Our PSBO method has 20 parameters. Each parameter has its own default value. We used the same 27 data sets adopted in our prior paper [9] (see Table 2) and did our tests on the Hyak computing cluster provided by the University of Washington Information Technology. The cluster runs the CentOS Linux 7.7 operating system and has many computing nodes, each with one 14-core 2.4GHz Intel Xeon E5-2680 central processing unit and 128GB memory. We did 20 experiments, one per parameter. In each experiment, we changed the corresponding parameter's value while keeping the other parameters at their default values. For each value tested for this parameter, we ran our PSBO method five times, each with a distinct random seed, for every data set. Each run was given one central processing unit core and 16GB of memory. At the end of the run, a final predictive model was built on the training data. There are three possible cases for computing the performance measures:

(1) If the parameter is h_{small}, L_{f_small}, L_{t_small}, or k, changing its value impacts our PSBO method's performance on small data sets only. In this case, we computed the average error rate of the final predictive model on the test data and the average search time taken by our PSBO method over all of the runs on every small data set.

(2) If the parameter is h_{large}, L_{f_large}, or L_{t_large}, changing its value impacts our PSBO method's performance on large data sets only. In this case, we computed the

average error rate and the average search time over all of the runs on every large data set.

(3) For every other parameter, changing its value impacts our PSBO method's performance on all data sets. In this case, we computed the average error rate and the average search time over all of the runs on each of the 27 data sets.

4 Results

In this section, we present our experimental results. Figures 1-20 show the impact of each of our PSBO method's 20 parameters on the average error rate of the final predictive model on the test data and the average search time taken by our PSBO method, one figure per parameter. In each figure, the corresponding parameter's default value is indicated by a vertical bar located near the middle of the figure. The 20 parameters all have reasonably large safe ranges: [2, 3] for C_2, [10, 20] for e, [1, 3] for h_{large}, [6, 12] for h_{small}, [2, 3] for k, [10, 25] for L_{f_large}, [4, 16] for L_{f_small}, [15, 20] for L_{t_large}, [7, 10] for L_{t_small}, [1, 4] for m, [7, 10] for n, [5, 15] for n_c, [1.05, 1.3] for p, [0.015, 0.03] for q, [30%, 40%] for r, [6, 12] for s, [600,000, 1,400,000] for t, [1, 5] for t_d, [3,000, 7,000] for u, and [60%, 70%] for w. For each parameter, its default value is in its safe range, within which our PSBO method's performance is insensitive to parameter value changes. Thus, the 20 parameters' default values all work well. There is not much room for improvement over them.

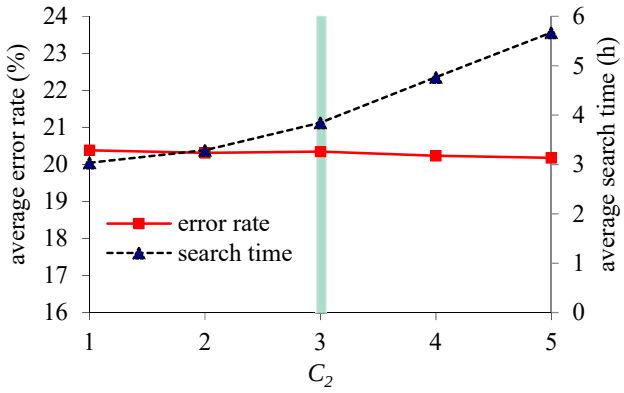

Fig. 1. The average error rate and the average search time vs. C_2.

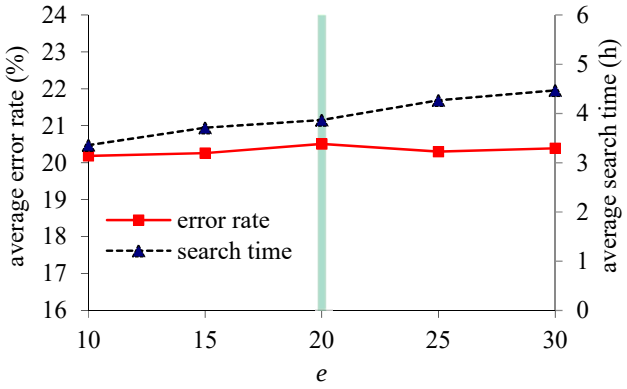

Fig. 2. The average error rate and the average search time vs. e.

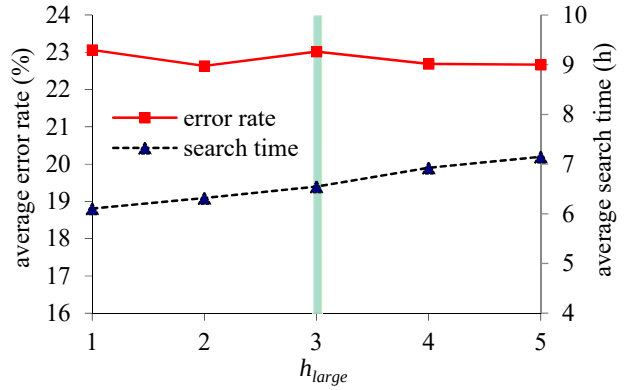

Fig. 3. The average error rate and the average search time vs. h_{large}.

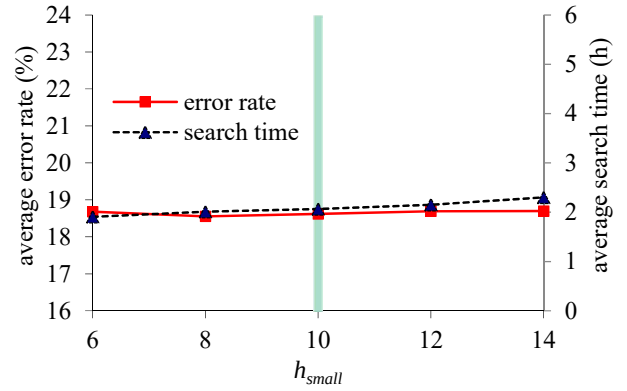

Fig. 4. The average error rate and the average search time vs. h_{small}.

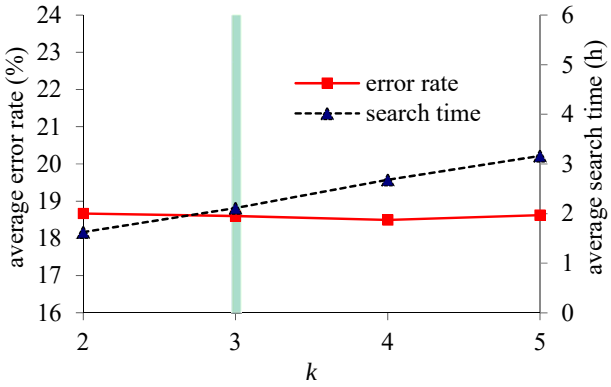

Fig. 5. The average error rate and the average search time vs. k.

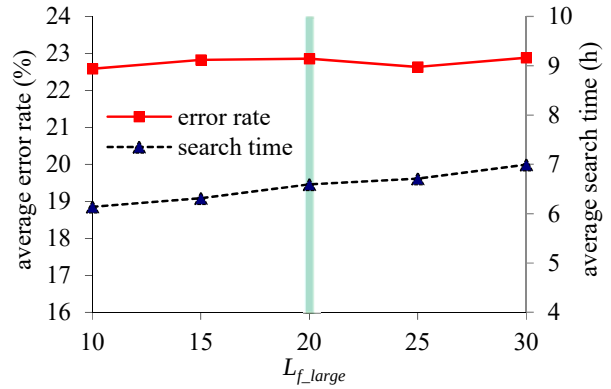

Fig. 6. The average error rate and the average search time vs. L_{f_large}.

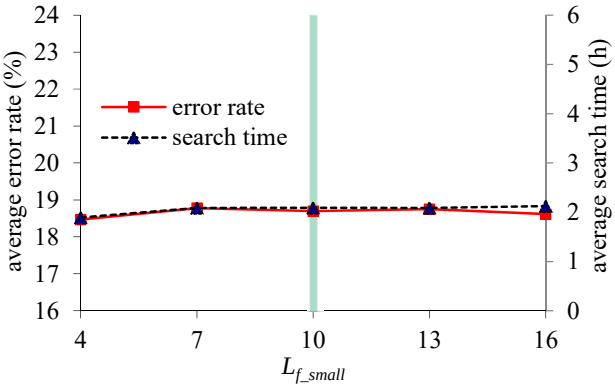

Fig. 7. The average error rate and the average search time vs. L_{f_small}.

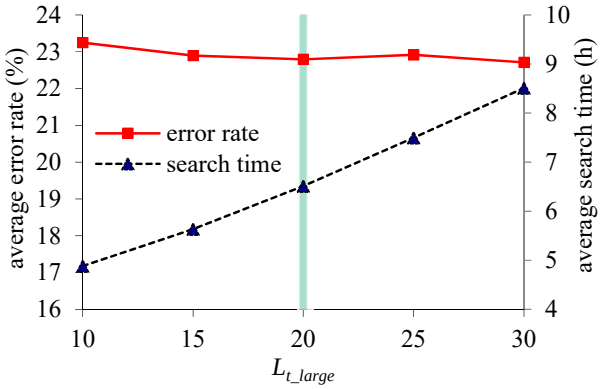

Fig. 8. The average error rate and the average search time vs. L_{t_large}.

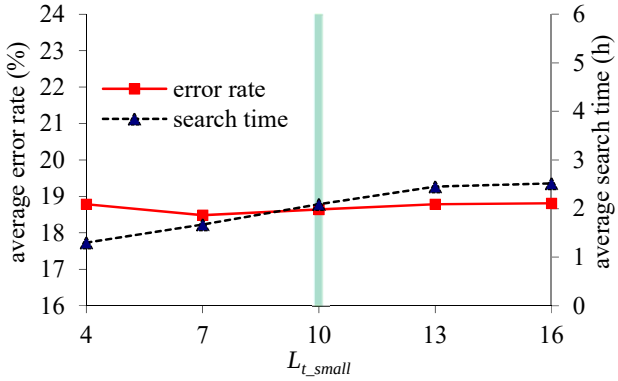

Fig. 9. The average error rate and the average search time vs. L_{t_small}.

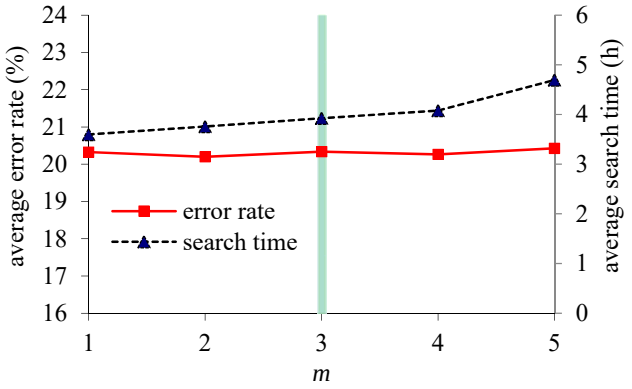

Fig. 10. The average error rate and the average search time vs. m.

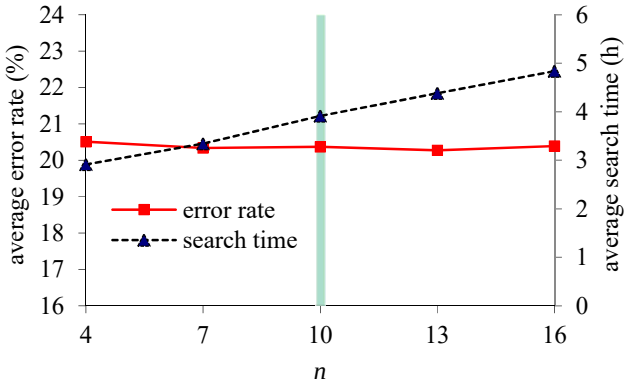

Fig. 11. The average error rate and the average search time vs. n.

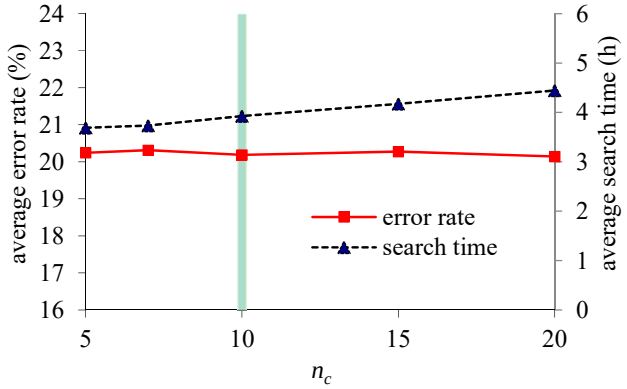

Fig. 12. The average error rate and the average search time vs. n_c.

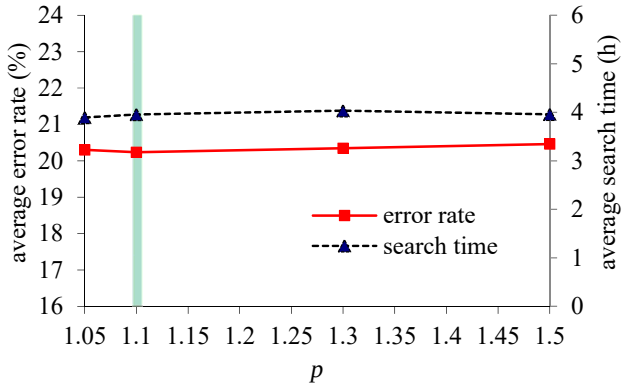

Fig. 13. The average error rate and the average search time vs. p.

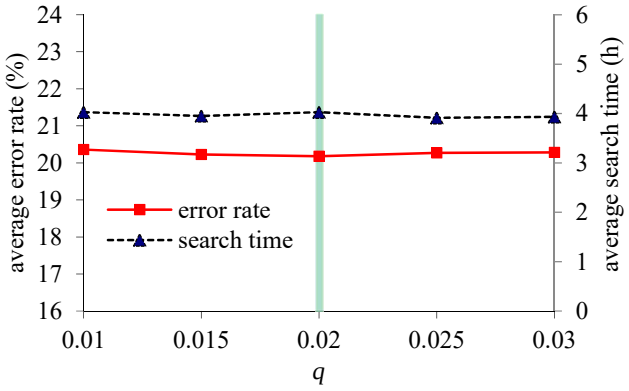

Fig. 14. The average error rate and the average search time vs. *q*.

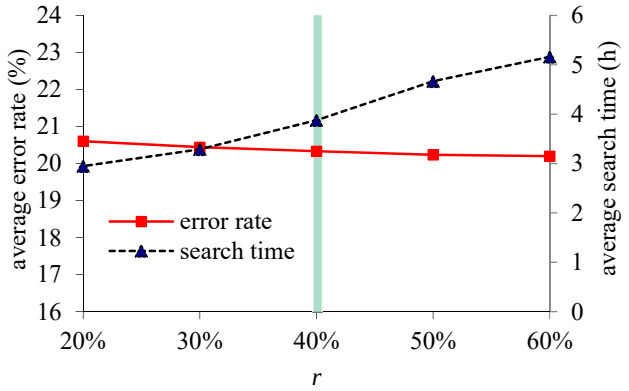

Fig. 15. The average error rate and the average search time vs. *r*.

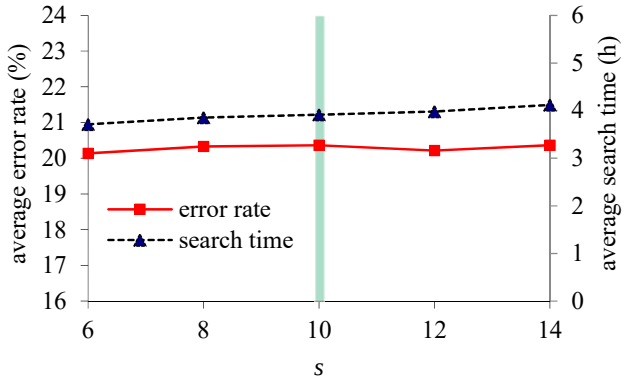

Fig. 16. The average error rate and the average search time vs. *s*.

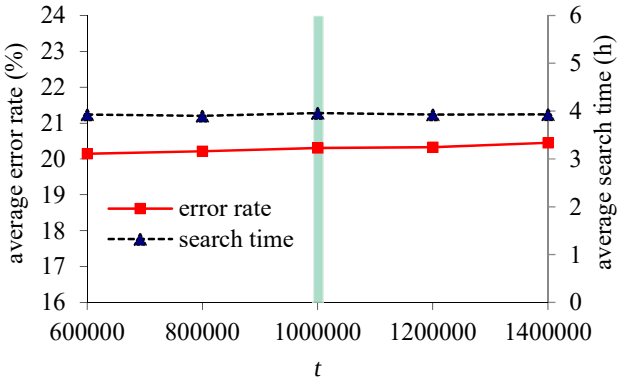

Fig. 17. The average error rate and the average search time vs. t.

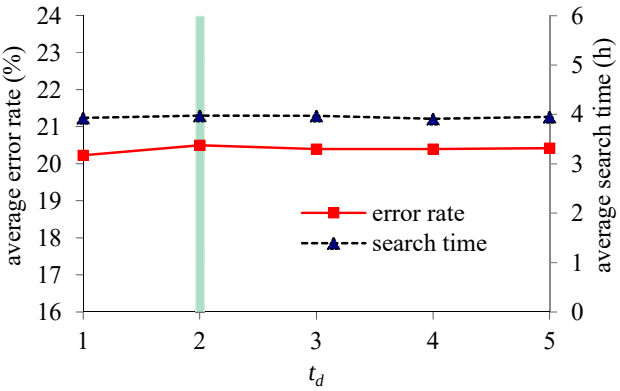

Fig. 18. The average error rate and the average search time vs. t_d.

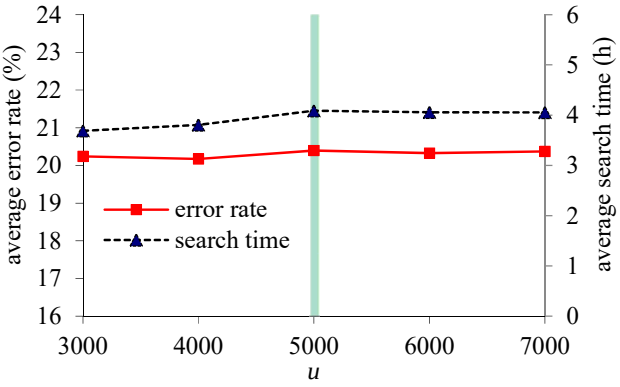

Fig. 19. The average error rate and the average search time vs. u.

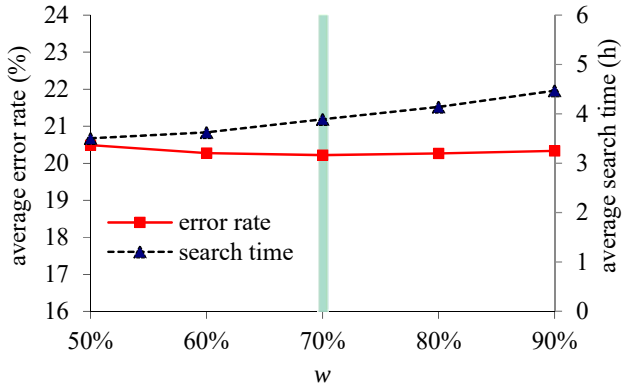

Fig. 20. The average error rate and the average search time vs. *w*.

For each of several parameters such as *e*, its default value works slightly worse than some other values in its safe range, if we only consider the data sets used in our experiments. Yet, we would not recommend changing these several parameters' default values just because of this, as the opposite case could occur on some other data sets unused in our experiments.

5 Conclusion

Our sensitivity analysis shows that the default values of our PSBO method's parameters work well. There is not much room for improvement over them. Also, each of these parameters has a reasonably large safe range, within which our PSBO method's performance is insensitive to parameter value changes. These results fill a gap left by our prior study [9].

Acknowledgment

GL was partially supported by the National Heart, Lung, and Blood Institute of the National Institutes of Health under Award Number R01HL142503. The funders had no role in study design, data collection, and analysis, decision to publish, or preparation of the manuscript. This work was facilitated through the use of advanced computational, storage, and networking infrastructure provided by the Hyak supercomputer system at the University of Washington.

Authors' Contributions

WZ did the computer coding work, conducted the experiments, participated in doing literature review and designing the study, and wrote the paper's first draft. GL conceptualized and designed the study, participated in doing literature review, and rewrote the whole paper. Both authors read and approved the final manuscript.

References

1. Golovin, D., Solnik, B., Moitra, S., Kochanski, G., Karro, J., Sculley, D.: Google Vizier: a service for black-box optimization. In: Proc. 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 1487-1495. ACM Press, New York (2017)
2. Erickson, N., Mueller, J., Shirkov, A., Zhang, H., Larroy, P., Li, M. *et al*.: AutoGluon-Tabular: robust and accurate AutoML for structured data. arXiv preprint arXiv:2003.06505 (2020)
3. Feurer, M., Klein, A., Eggensperger, K., Springenberg, J., Blum, M., Hutter, F.: Efficient and robust automated machine learning. In: Proc. Annual Conference on Neural Information Processing Systems 2015, pp. 2944-2952. (2015)
4. Li, L., Jamieson, K.G., DeSalvo, G., Rostamizadeh, A., Talwalkar, A.: Hyperband: a novel bandit-based approach to hyperparameter optimization. J. Mach. Learn. Res. **18**(185), 1-52 (2017)
5. Luo, G.: A review of automatic selection methods for machine learning algorithms and hyper-parameter values. Netw. Model. Anal. Health Inform. Bioinform. **5**, 18 (2016)
6. Strubell, E., Ganesh, A., McCallum, A.: Energy and policy considerations for deep learning in NLP. In: Proc. 57th Conference of the Association for Computational Linguistics, pp. 3645-3650. Association for Computational Linguistics (2019)
7. Swearingen, T., Drevo, W., Cyphers, B., Cuesta-Infante, A., Ross, A., Veeramachaneni, K.: ATM: a distributed, collaborative, scalable system for automated machine learning. In: Proc. 2017 IEEE International Conference on Big Data, pp. 151-162. IEEE Press, New York (2017)
8. Thornton, C., Hutter, F., Hoos, H.H., Leyton-Brown, K.: Auto-WEKA: combined selection and hyperparameter optimization of classification algorithms. In: Proc. 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 847-855. ACM Press, New York (2013)
9. Zeng, X., Luo, G.: Progressive sampling-based Bayesian optimization for efficient and automatic machine learning model selection. Health Inf. Sci. Syst. **5**(1), 2 (2017)