ISSN : 1226-7244 (Print)
ISSN : 2288-243X (Online)
논문번호 20-04-07

j.inst.Korean.electr.electron.eng.Vol.24,No.4,969~974,December 2020
http://dx.doi.org/10.7471/ikeee.2020.24.4.969

Comparison of Hyper-Parameter Optimization Methods for Deep Neural Networks

Ho-Chan Kim*, Min-Jae Kang*★

Abstract

Research into hyper parameter optimization (HPO) has recently revived with interest in models containing many hyper parameters, such as deep neural networks. In this paper, we introduce the most widely used HPO methods, such as grid search, random search, and Bayesian optimization, and investigate their characteristics through experiments. The MNIST data set is used to compare results in experiments to find the best method that can be used to achieve higher accuracy in a relatively short time simulation. The learning rate and weight decay have been chosen for this experiment because these are the commonly used parameters in this kind of experiment.

Key words : hyper parameter optimization, deep neural networks, grid search, random search, Bayesian optimization.

Ⅰ. Introduction

Hyper parameter optimization is an important component in finding the optimal hyper parameters for the training process of neural networks. Hyper parameters represent parameters such as learning rate, weight decay, dropout rate, and batch size. These parameters play an important role in the accuracy and efficiency of neural networks. In the past, we constantly adjusted it through trial and error until satisfactory results are obtained. Therefore, how to optimize hyper-parameters becomes a key issue in a neural network learning algorithm. Traditionally, two search methods have been used: manual search and automatic search. Manual search uses intuition and experience. This method is fine for professionals who are familiar and experienced with the system, but difficult for non-professional users to apply. The grid search is an automatic search that searches an area within a range in a grid format. Also, a random search algorithm has been proposed to improve the problem in grid search. Random search tries any combination of ranges of values. Compare with grid search, random search can be considered more efficient in high-dimensional space. But random search is unreliable for training some complex models. Bayesian method is recently drawing attention to solve this kind of problem. It combines prior information about the unknown function with sample information, to obtain posterior information of the function distribution by using Bayesian formula.

In this paper, we attempt to compare methods

* Dept. of Electrical Eng., Jeju National Univ.
*★ Corresponding author
Email : minjk@jejunu.ac.kr, Tel : +82-64-754-3666
※ Acknowledgment
This work was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF-2018R1D1A1B07045976) in (2018).
Manuscript received Nov. 24, 2020; revised Dec. 13, 2020; accepted Dec. 18, 2020.

of hyper parameter optimization. Section Ⅱ briefly describes model free methods, Section Ⅲ discusses Bayesian optimization method, and in Section Ⅳ, we use MNIST dataset for an experiment to compare the different methods.

Ⅱ. Model Free Methods for HPO

In general, every optimization method can be applied to HPO. We first discuss model-free HPO methods in this section and then describe Bayesian optimization methods next.

1. Manual Search

If you have ever trained a deep learning model, you've probably gone through a lot of trial and error in determining the values of these key hyper-parameters. In initial, candidate hyper-parameter values are selected based on intuition or experience. And then, the hyper-parameter values will change over and over again trial and error until you get a good performance evaluation result. This method of searching for optimal hyper-parameter values in this way is also known as manual search. Manual search is the most intuitive way to optimize hyper-parameter, but it has some problems. First, the process of finding the optimal hyper parameters depends somewhat on luck. Based on your intuition, it is relatively difficult to determine if those parameters which you found are actually optimal parameters The second problem with manual search is that if you want to search multiple types of hyper parameters at the same time, the problem becomes more complicated. As such, there are several types of hyper parameters that show mutual influence relationships, so it becomes very difficult to apply the existing intuition to each of a single hyper parameter when searching for two or more hyper parameters at once.

2. Grid Search and Random Search

Grid search is the primary method for hyper parameter optimization and thoroughly searches hyper parameter sets in a custom range. Users should have prior knowledge of these hyper parameters when selecting all candidates. Grid search can be applied to multiple hyper parameters. Grid search is the simplest search algorithm leading to the most accurate predictions and users can always find the best combination[2]. The advantage of grid search is its mathematical simplicity and the ability to run multiple hyper-parameters in parallel. The downside is that as the number of hyper-parameters to be searched increases, the total search time increases exponentially. For this reason, other search algorithms may have a more favorable function, but grid search is still the most widely used method because of its mathematical simplicity and the ability to search for multiple parameters[3].

Random search is a basic improvement to grid search. The search process continues until the predetermined iteration number or the desired accuracy are reached. Random search is similar to grid search, but has been proven to produce better results due to the following two benefits[3]: First, random search can perform better, especially if some hyper-parameters are not evenly distributed. In this search pattern, random searches are relatively more likely to find the optimal configuration than grid searches. Secondly, it is not a promise that random searches will give you the best value, but you can get good value. And the more time you spend, the more likely you are to find a better set of hyper-parameters. Conversely, for grid search, we cannot guarantee better results even with long search times. In most cases, random search is more effective than grid search, but it is still a computationally intensive method. Random searches generally require more time and computational resources than other search methods.

Ⅲ. Bayesian Optimization Method

Bayesian Optimization originally aims to find

an optimal solution that maximizes or minimizes the function value by assuming an unknown objective function that receives an input. Bayesian optimization is an iterative algorithm with two key ingredients[4]: a probabilistic surrogate model and an acquisition function to decide which point to evaluate next. In each iteration, the surrogate model is fitted to all observations of the target function made so far. Then the acquisition function, which uses the predictive distribution of the probabilistic model, determines the utility of different candidate points, trading off exploration and exploitation.

1. Surrogate model

If you have ever trained a deep learning model, you've probably gone through a lot of trial and error in determining the values of these key hyper-parameters. The Gaussian process (GP) can be the default choice of surrogate models for the objective function. Like Gaussian distribution defined by mean and covariance, Gaussian process is defined by its mean function $m: x \rightarrow R$ and its covariance function $k: x \times x \rightarrow R$. The Gaussian process can be expressed as

$$f(x) \sim GP(m(x), k(x,x')) \tag{1}$$

Assuming that the Gaussian process mean function $m(x) = 0$, the covariance function k can be written as bellows

$$k(x_i, x_j) = \exp\left(-\frac{1}{2} \| x_i - x_j \|^2\right) \tag{2}$$

where x_i, x_j represent the ith and jth samples, respectively. The process of determining the posterior distribution of f(x) is as follows.

1) Assume the function f values are drawn with the observed training set $D = \{x_i, f_i(x)\}_{i=1}^{t}$, and following to a multivariate normal distribution $f \sim N(0, K)$, where

$$K = \begin{bmatrix} k(x_1,x_1) & k(x_1,x_2) & \cdots & k(x_1,x_t) \\ k(x_2,x_1) & k(x_2,x_2) & \cdots & k(x_2,x_t) \\ \vdots & \vdots & \ddots & \vdots \\ k(x_t,x_1) & k(x_t,x_2) & \cdots & k(x_t,x_t) \end{bmatrix} \tag{3}$$

The function k in (3) measures the degree of approximation between two samples where the diagonal element $k(x_i, x_i) = 1$.

2) Compute the function value $f(x_{t+1})$ at the new sample point x_{t+1} based on the function f. The function value f_{t+1} follows the $t+1$ dimensional normal distribution:

$$\begin{bmatrix} f_{1:t} \\ f_{t+1} \end{bmatrix} \sim N\left(0, \begin{bmatrix} K & k^T \\ k^T & k(x_{t+1}, x_{t+1},) \end{bmatrix}\right) \tag{4}$$

Where $f_{1:t} = [f_1, f_2, ..., f_t]^T, k^T = [k(x_{t+1}, x_1), k(x_{t+1}, x_2), ..., k(x_{t+1}, x_t)]$ and f_{t+1} follows one-dimensional normal distribution, i.e. $f_{t+1} \sim N(u_{t+1}, \sigma^2_{t+1})$, where

$$u_{t+1}(x_{t+1}) = kK^{-1}f_{1:t} \tag{5}$$

$$\sigma^2_{t+1}(x_{t+1}) = -k^T K^{-1} k + k(x_{t+1}, x_{t+1}) \tag{6}$$

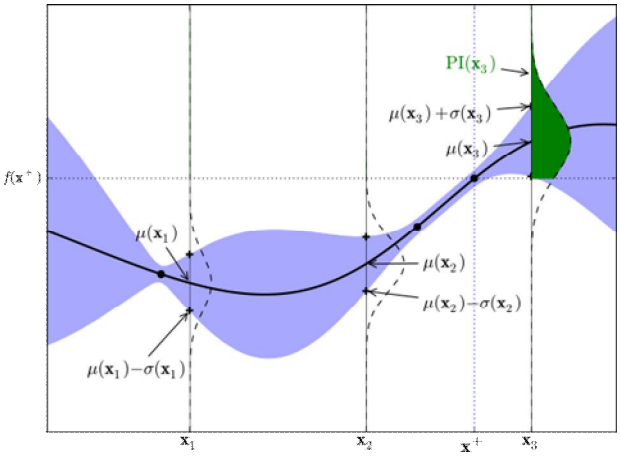


Fig. 1. Visualization for Probability of Improvement(PI)[5].

In the Fig. 1, the solid line is the surrogate model with points observed so far. The maximum function value $f(x^+)$ so far has occurred at the far right point. The purple shaded area shows the degree of standard deviation σ(x) on the estimated surrogate model function. The standard deviation narrows in areas where the gap between the two observed points is narrow. This means that larger function values may occur over the

place that has not yet been investigated. The point of x_3 was chosen as the next best candidate among the different points(x_1, x_2, x_3), because the value of $\mu(x_3)+\sigma(x_3)$ is the biggest among these points. As shown in the above figure, Probability of Improvement, PI(x_3), can be calculated by using the surrogate function and CDF(Cumulative Distributed Function).

The function PI can be expressed as

$$PI(x) = P(f(x) \geq f(x^+)) = \varphi(\frac{u(x)-f(x^+)}{\rho(x)}) \quad (7)$$

As seen in (7), the PI function tries to find a better point than the current optimal value. The larger the positive difference in $u(x)-f(x^+)$, the smaller of $\rho(x)$, the better for the candidate point. Also, this method has been known easily be fallen to the local optimal solution. To solve this problem, a parameter ε is added into PI function as follows

$$PI(x) = P(f(x) \geq f(x^+)+\epsilon) = \varphi(\frac{u(x)-f(x^+)-\epsilon}{\rho(x)}) \quad (8)$$

So the next sampling point can be searched in a larger area.

2. Acquision fuction

Among commonly used acquisition functions, the Expected Improvement(EI) is most often used as an acquisition function and is designed to include both of these exploration strategies and exploitation strategies[6].

The degree of improvement, I, is defined as the difference between the sampling point value and the current optimal value.

$$I(x) = \max(0,\ f_{t+1}(x) - f(x^+)) \quad (9)$$

The $f_{t+1}(x)$ is normally distributed with the mean $\mu(x)$ and the standard deviation $\sigma^2(x)$ satisfying the condition of $f_{t+1}(x)-f(x^+) \geq 0$. Then, the random variable I is also normally distributed with the mean and the standard deviation[7]. The expectation value of I (EI) can be defined with the probability density function of $f(I)$

$$E(I) = \int_0^\infty I f(I)\, dI \quad (10)$$

where

$$f(I) = \frac{1}{\sqrt{2\pi}\,\rho(x)} \exp(-\frac{(u(x)-f(x^+)-I)^2}{2\rho^2(x)}) \quad (11)$$

The strategy of EI function optimization[8] is to find a point of x for maximizing EI with

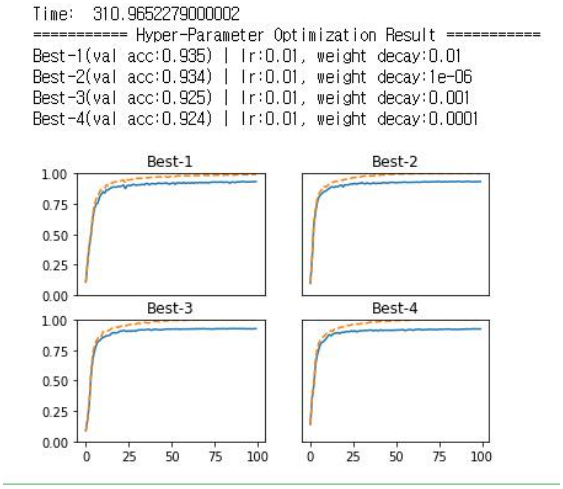
$$x = \arg max\ E[I(x)] \quad (12)$$

IV. EMPIRICAL RESULTS AND OBSERVATION

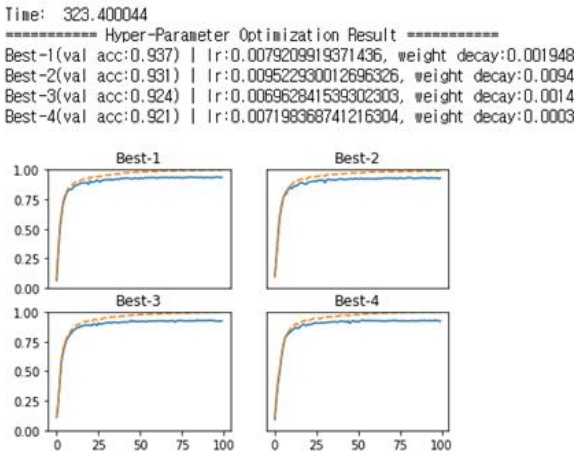
The MNIST digits (LeCun et al., 1998a), dataset has 60,000 training images, 10,000 test images, each showing a 28x28 grey-scale pixel image of one of the 10 digits[9]. We optimized back-propagation networks with five hidden layers, with one hundred hidden units per layer, and with a softmax logistic regression for the output layer. The cost function is a cross entropy error. The neural networks were optimized with stochastic back-propagation on mini-batches of size 100. For experimenting, we tested the different types of HPO method: grid search, random search and Bayesian optimization method. Also, to save training time, we only used 5,000 data from the MNIST dataset, of which 80% were used for training and 20% for validation.

Fig. 2 shows four good simulation results for training and test data accuracy of over 100 epochs, respectively, using the grid search and random search methods. The hyper parameters used in this experiment are the learning rate and weight decay, where learning rate is tested in the range of $e^{-2} \sim e^{-4}$ and weight decay in the range of $e^{-2} \sim e^{-7}$. For grid search, the 18 grid points(3x6) were tested, where 3 grid lines are in

the learning rate and 6lines are in the weight decay. For random search, 18 trials were tested randomly in the same range as the grid search. As shown in Fig. 2, simulation times and results of both methods are very similar. For Bayesian method, Gaussian process and EI are used for the surrogate model and the acquisition function. Fig. 3 shows how the Bayesian method progresses in the first 3 iterations using a one-dimensional graphic where the parameter used is the learning rate. As can be seen from this acquisition function, the next best candidate can occur in the left area due to the wide shading standard deviation. Bayesian method has been tested with the same conditions of the above methods of grid search and random search. In the table 1, the numbers in the learning rate and weight columns are the exponents of the exponential function. The table 1 has shown that the simulation time and test accuracy have been improved slightly compared to the above results. Without the previous experience of experiments, Bayesian method is expected to outperform others.



(a)



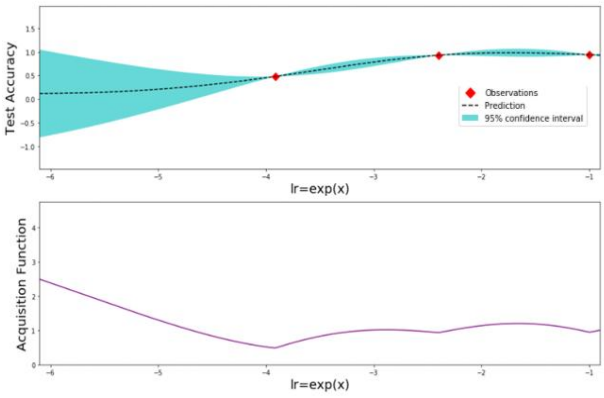
(b)

Fig. 2. Grid Search and Random Search
(a) Grid Search (b) Random Search

Table 1. 18 iteration results using Bayesian optimization using two parameter.

iter	target	lr	weight...
1	0.748	-3.256	-2.424
2	0.8609	-2.986	-3.276
3	0.879	-2.871	-3.639
4	0.922	-1.976	-3.734
5	0.9283	-1.258	-3.595
6	0.9324	-1.0	-4.199
7	0.9311	-1.0	-6.0
8	0.1137	-6.0	-6.0
9	0.93	-1.0	-5.25
10	0.9203	-1.596	-5.97
11	0.9417	-1.0	-2.353
12	0.6672	-1.007	-1.017
13	0.9393	-1.0	-2.711
14	0.9273	-1.4	-2.528
15	0.9241	-1.623	-4.678
16	0.8915	-2.693	-6.0
17	0.915	-2.356	-5.2
18	0.0956	-6.0	-1.0

Time: 276.3502429



iter	target	x
1	0.478	-3.915
2	0.928	-2.398
3	0.939	-1.0

Fig. 3. The first 3 iteration steps of Surrogate and Acquisition Model in one dimensional Bayesian progress.

V. Conclusion

In this paper, we use a study of three HPO methods: grid search, random search, and Bayesian optimization method to predict optimal parameters in neural networks. Experimental results show that the Bayesian optimization method performs better than other available methods. The Bayesian method is mathematically complex to apply, but the results did not improve significantly in this experiment. This result may be due to previous experience with this experiment. This allowed us to guess the range of parameters to get good results. Therefore, grid search and random search can yield similar results to Bayesian method. However, the Bayesian optimization method is believed to be the best choice for the black box problem that we have not experienced before.

References

[1] M. Feurer and F. Hutter, "Hyperparameter optimization," in F. Hutter, L. Kotthoff, and J. Vanschoren (Eds.), *Automated Machine Learning*, pp.3-33, Springer, 2019.

[2] J. Bergstra, D. Yamins, and D. D. Cox, "Making a science of model search: Hyperparameter optimization in hundreds of dimensions for vision architectures," in *Proc. of the 30th International Conference on Machine Learning*, vol.28, pp.115-123, 2013. DOI: 10.5555/3042817.3042832

[3] J. Bergstra and Y. Bengio, "Random search for hyper-parameter optimization," *Journal of machine learning research*, vol.13, pp.281-305, 2012. DOI: 10.5555/2188385.2188395

[4] J. Snoek, H. Larochelle, and R. P. Adams, "Practical bayesian optimization of machine learning algorithms," *Advances in Neural Information Processing Systems*, vol.25, pp.2951-2959, 2012.

[5] Brochu, E., Cora, M., and de Freitas, N. "A tutorial on Bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning," In TR-2009-23, UBC, 2009.

[6] J. Wang, J. Xu, and X. Wang, "Combination of hyperband and bayesian optimization for hyperparameter optimization in deep learning," arXiv preprint arXiv:1801.01596, 2018. https://arxiv.org/abs/1801.01596

[7] S. Falkner, A. Klein, and F. Hutter, "Bohb: Robust and efficient hyperparameter optimization at scale," *Proceedings of Machine Learning Research*, vol.80, pp.1437-1446, 2018.

[8] C. Harrington, "Practical guide to hyperparameters optimization for deep learning models," Deep Learning, 2018. https://blog.floydhub.com/guide-to-hyperparameters-search-for-deep-learning-models/

[9] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient based learning applied to document recognition," *Proceedings of the IEEE*, vol.86, no.11, pp.2278-2324, 1998.

BIOGRAPHY

Ho-Chan Kim (Member)

1987 : BS degree in Control & Instrument Engineering, Seoul National University.
1989 : MS degree in Control & Instrument Engineering, Seoul National University.
1994 : PhD degree in Control & Instrument Engineering, Seoul National.
1995~Current : Dept. of Electrical Eng. Jeju National University.

Min-Jae Kang (Member)

1982 : BS degree in Electrical Engineering, Seoul National University.
1991 : PhD degree in Electrical Engineering, Univ. of Louisvill, U.S.A.
1992~Current : Dept. of Electronic Eng. Jeju National University.