Proceedings of the 2022 Winter Simulation Conference
B. Feng, G. Pedrielli, Y. Peng, S. Shashaani, E. Song, C.G. Corlu, L.H. Lee, E.P. Chew, T. Roeder, and P. Lendermann, eds.

TREE-STRUCTURED PARZEN ESTIMATORS WITH UNCERTAINTY FOR HYPERPARAMETER OPTIMIZATION OF MACHINE LEARNING ALGORITHMS

Alejandro Morales-Hernández
Inneke Van Nieuwenhuyse
Sebastian Rojas Gonzalez

Data Science Institute, University of Hasselt
Martelarenlaan 42
B-3500, Hasselt, BELGIUM

ABSTRACT

Hyperparameter optimization (HPO) is one of the first tasks to be performed during the application of Machine Learning (ML) algorithms to real problems. Tree-*structured* Parzen estimators (TPE) have demonstrated their ability to find hyperparameter configurations in high dimensions with efficient evaluation budgets. However, as is common in HPO procedures, TPE ignores the fact that the expected performance of the algorithm, for any given HPO configuration, is affected by uncertainty. Building on the TPE algorithm proposed by Bergstra et al. (2011), we propose a strategy to account for this uncertainty and show that its management leads to better algorithm performance.

1 INTRODUCTION

The parameters that need to be specified *before* training an ML algorithm are usually referred to as *hyperparameters* (HP). The selection of the values of the hyperparameters is a challenging task, as traditional optimization methods are often not applicable (Luo 2016). Furthermore, the process is most often stochastic, since we observe different algorithm performances for the same hyperparameter configuration. Instead of focusing on the mean cross-validation error (as is common in the literature on HPO), we show that accounting for this uncertainty during the optimization process can find HP configurations that lead to better algorithm performance.

2 METHOD

The original TPE algorithm starts from observations $\mathbf{X} = \{(\mathbf{x}^{(1)},\bar{y}^{(1)}),\dots,(\mathbf{x}^{(n)},\bar{y}^{(n)})\}$ where $\mathbf{x}^{(i)}$ is a hyperparameter configuration of dimension d and $\bar{y}^{(i)}$ is the mean expected performance of the algorithm, observed after the algorithm has been trained/cross-validated using hyperparameter configuration i. It then splits these observations into a "good" set ($\mathbf{X}_l = \{\mathbf{x}^{(i)}|y^* > \bar{y}^{(i)}\}$) and a "bad" set (the remaining observations), selecting y^* as a quantile γ of the $\bar{y}$ values (so $p(y^* > \bar{y}) = \gamma$, where γ=0.15 in the original paper). The algorithm then uses kernel density estimation on the set $\mathbf{X}_l$ to obtain a density function $l(\mathbf{x})$, and analogously on set $\mathbf{X}_g$ to obtain a density function $g(\mathbf{x})$ (Bergstra et al. 2011). Finally, the ratio $\frac{g(\mathbf{x})}{l(\mathbf{x})}$ guides the search towards the estimated optimal HP configuration. The algorithm has been shown to outperform (deterministic) Gaussian Process-based sequential optimization algorithms (Bergstra et al. 2011) tuning Deep Neural Networks. Yet, the original algorithm neglects the fact that the $\bar{y}^{(i)}$ values are *noisy*. Indeed, they typically result from a k-fold cross-validation protocol, yielding a random sample of k independent $\bar{y}^{(i)}$ per HP configuration i considered. Theoretically, the resulting sample mean $\bar{y}^{(i)}$ is also a random variable: according to the central limit theorem (for k sufficiently large), it will be normally distributed around the true mean (which is estimated by $Y^{(i)}$), with a variance that can be estimated by the sample variance divided by k. We thus propose a modified version of the algorithm, that takes this uncertainty into account by ranking solutions $x^{(i)}$ in decreasing order of their probability of yielding a "good" expected performance ($prob(Y \leq y^*)$, and selecting only the top λ percent of all solutions to enter the set $\mathbf{X}_l$ (where λ is a user-defined parameter).

3 RESULTS AND CONCLUSIONS

We optimized five hyperparameters (neurons, training epochs, initial learning rate, and exponential decay rate for estimates of first and second-moment vector in *adam* solver) of a Multi-Layer Perceptron (MLP) to minimize the cross-validation classification error in the *ILPD* dataset (OpenML ID=41945). We used 20% of the initial dataset as the test set and the remainder for HPO, and apply stratified *k-fold cross-validation* ($k = 10$) to train/validate the ML algorithm. The experiment was repeated 10 times, each time starting from a different random design (consisting of $11d-1 = 54$ HP configurations). We used $\gamma = 0.2$ in both algorithms and $\lambda = 0.2$ for our proposal. Figure 1 shows the *expected* classification error of the two algorithms in the first 100 iterations of the optimization process. The result of a random search (sampling randomly $11d-1$ configurations) is displayed as a reference.

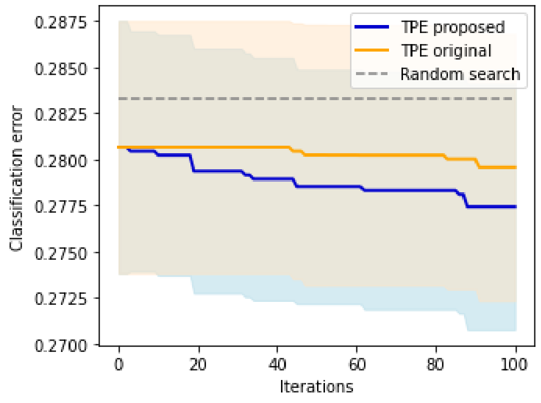

Figure 1: Mean classification error based on 10 macro-replications, for the in *ILPD* dataset. Shadowed area corresponds to the $mean \pm std$ of 10 macro-replications.

Our results show that considering the uncertainty during HPO can lead to higher performance of ML algorithms. Further research will be focused on a sensitivity analysis of the parameters γ and λ, other sources of uncertainty, and their influence on this HPO procedure.

ACKNOWLEDGMENTS

This work was supported by the Flanders Artificial Intelligence Research Program (FLAIR) and the Research Foundation Flanders (FWO Grant 1216021N). The authors want to acknowledge their affiliation to the Core Lab VCCM Flanders Make, Belgium

REFERENCES

Bergstra, J., R. Bardenet, Y. Bengio, and B. Kégl. 2011. "Algorithms For Hyper-Parameter Optimization". In *25th annual conference on neural information processing systems (NIPS 2011)*, Volume 24. Neural Information Processing Systems Foundation.

Luo, G. 2016. "A Review Of Automatic Selection Methods For Machine Learning Algorithms And Hyper-Parameter Values". *Network Modeling Analysis in Health Informatics and Bioinformatics* 5(1):18. https://doi.org/10.1007/s13721-016-0125-6.