Search Algorithms for Automated Hyper-Parameter Tuning

Leila Zahedi[1,2], Farid Ghareh Mohammadi[3],
Shabnam Rezapour[4], Matthew W. Ohland[5], M. Hadi Amini[1,2]

[1] Knight Foundation School of Computing and Information Sciences
Florida International University (FIU), Miami, FL 33199
[2] Sustainability, Optimization, and Learning for InterDependent networks laboratory
(solid lab), FIU, Miami, FL 33199
[3] Department of Computer Science, University of Georgia, Athens, Georgia, 30602,
[4] Enterprise and Logistics Engineering, FIU, FL 33174
[5] School of Engineering Education, Purdue University, West Lafayette, IN, 47907
lzahe001@cs.fiu.edu, farid.ghm@uga.edu,
srezapou@fiu.edu, ohland@purdue.edu, amini@cs.fiu.edu

Abstract. Machine learning is a powerful method for modeling in different fields such as education. Its capability to accurately predict students' success makes it an ideal tool for decision-making tasks related to higher education. The accuracy of machine learning models depends on selecting the proper hyper-parameters. However, it is not an easy task because it requires time and expertise to tune the hyper-parameters to fit the machine learning model. In this paper, we examine the effectiveness of automated hyper-parameter tuning techniques to the realm of students' success. Therefore, we develop two automated Hyper-Parameter Optimization methods, namely grid search and random search, to assess and improve a previous study's performance. The experiment results show that applying random search and grid search on machine learning algorithms improves accuracy. We empirically show automated methods' superiority on a real-world educational data (MIDFIELD) for tuning HPs of conventional machine learning classifiers. This work emphasizes the effectiveness of automated hyper-parameter optimization while applying machine learning in the education field to aid faculties, directors', or non-expert users' decisions to improve students' success.

Keywords: Hyper-Parameter Tuning, Machine Learning Optimization, AutoML

1 Introduction

⋄ **Motivation** The usage of information technologies in various fields has led to increasingly large-scale data derived from multiple settings. One of these settings is education, concerning better understanding students' pathways and the environments they learn in. Educational Data Mining (EDM) is an emerging field focusing on mining datasets to answer educational research questions [1][2]. One of the crucial EDM applications is predicting students' success [2]. There are a variety of approaches to measure students' success, such as graduation rates, on-time completion, or GPA [3][4][5]. However, these metrics may overestimate or underestimate a sub-population persistence, such as non-traditional students, part-time students, or transferred students. Therefore, these populations are usually neglected in many studies, which leads to a lack of understanding of the educational pathways. This matter behooves the researchers to look for fair metrics to assess students' success. Stickiness is one of these metrics [6].
Tuning HPs is an essential task to make a predictive model that performs at its best [7][8]. Building such models is the primary goal of ML models [9]. However, despite HPs' critical role in the resulting predictive models' quality, they have no clear agreeable defaults in different applications. Manually tuning the HPs not only needs a deep understanding of the models but is also impractical, time and cost-inefficient. Hence, it has become vital to automate the process of optimizing the HPs. In Hyper-Parameter Optimization (HPO), we usually aim to use the value of parameters that significantly contribute to improving a model's accuracy. Therefore the search algorithm looks through different combinations of HP configurations (search space), enabling the model to generate the best model among the candidates through an iterative process [10,11]. ML models' HPs need to be selected automatically and carefully to be effective in the application. For a new dataset, the optimal parameter values depend on the application and, more specifically, the dataset itself. This work helps educational researchers and institutions better understand and develop ML models by identifying the appropriate set of HPs in an effective way.

⋄ **Contribution** This study applies two automated HPO methods to predict students' graduation accurately to address the issues mentioned above regarding manual tuning. Grid Search (GS) and Random Search (RS) are among automated parameter optimization methods. The advantage of using GS and RS is higher learning accuracy and its capability for parallelization; that is not an option for

all the HPO methods. We apply GS and RS to find the most appropriate HPs for different conventional ML algorithms. The reason for selecting various ML algorithms is that the performance of a given ML model not only depends on the fundamental quality of the algorithm but also on the details of its tuning. Therefore, it is hard to decide if a given ML model is genuinely better or better tuned. In this study, leveraging different numbers of ML models provides us with the option to choose the model with the best performance among other ML models rather than exploring a single model.

The main contributions are expanding upon existing research by incorporating automated HPO techniques leveraging conventional ML models into a single pipeline to improve prediction performance and enhance educational decision-making. Also, this is the first time HPO is being applied on this dataset (MIDFIELD[12]). Unlike other studies that study short time-spans and single institution datasets, this study dataset considers a 30-year longitudinal dataset for undergraduate students from 16 universities. In this work, we use a modified definition of graduation, stickiness (the fraction that "stick" to the program or persist), for students who came into contact with their programs [13], to include the populations that are overestimated or underestimated in previous studies. This study aims to improve ML models' classification for the MIDFIELD dataset using GS and RS then compare it to the previous work.

In the following sections, we first develop a clear and formal definition of HPO, and we provide a basic understanding of the concepts and methodologies applied. From there, we discuss the implementation and evaluation of these approaches. Next, we cover experimental results, conclusions, and future work ideas.

2 Related Work

⋄ **Hyper-parameter Optimization** Every ML model has some HPs, and tuning them is essential for making a model work at its best. The notion of HPs is different from parameters. HPs are the parameters that need initialization before training the model since they represent the model architecture. In contrast, parameters can get initialized and updated during the model training [14]. Automatically tuning the HPs is one of the main tasks in automated ML (AutoML) to release the burden of manual tasks and improve the performance, reproducibility, and fairness of various studies [15]. There are several optimization techniques for HPO problems. The most common and conventional HPO methods are manual search or grad student descent (GSD), GS, and RS. Each of them is defined as below:

Manual Search: GSD is the most basic HP tuning method and a typical approach among researchers and students[16]. In this method, users try different HP values based on guessing or domain knowledge and repeat this process until they obtain a satisfactory improved result or when they are out of time. GSD needs a deep understanding of the ML models or sufficient time to get good results. However, the complex nature of ML models and large search spaces make it an impractical approach [17]. Mohammadi *et al.* explored the effect of manual parameter tuning on performance by combining different embedded parameters and improved the accuracy of semantic auto-encoder in an image classification problem [7].

Grid Search: GS is the brute-force way of searching HPs [18], with defined lower and higher bound along with specific steps [19]. GS work based on the Cartesian product of the different set of values, evaluate every configuration and return the combination with the best performance [20]. GS has a simple implementation; however, it can be very inefficient for large search spaces due to its exhaustive nature. This problem exacerbates as data dimensionality increases.

Random Search: RS is another common and standard method of searching HPs [21]. RS chooses the HPs configurations on a random basis (instead of evaluating every configuration) and repeats this process until the defined resources are over. RS is much faster than GS but does not follow a path to find the optimal configuration.

3 Experimental Setup

The baseline ML algorithms (model with default parameters), GS, and RS are implemented in this work. Figure 1 shows the research methodology's flowchart. The first phase to conduct is collecting and preparing the data for ML algorithms. Data preparation includes separate tasks such as data reduction, data cleansing, normalization, and feature engineering. Data reduction, in this study, refers to using a subset of the dataset rather than the entire dataset. Filtering and sampling are among the most common ways of data reduction. Since we are only interested in students of computing programs, we filter the data and only include the students who enrolled at some point in one of the following majors: computer science, computer engineering, software engineering, computer programming, information technology, and computing and information sciences (CIP=11). In the data cleaning step, which is a step for removing the corrupt and not applicable data, we remove the features with more than 60% missing values. We also normalized or standardized the range of features of data. After the preprocessing step is the

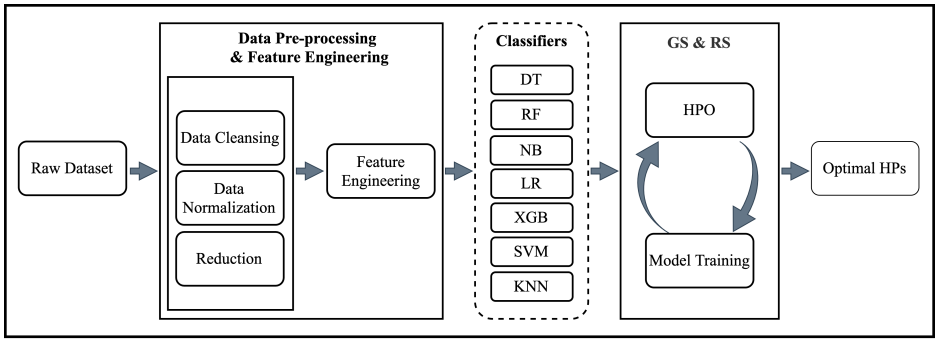


Fig. 1: Flowchart of the research methodology

feature engineering step in which we create some new features from raw features that we think might be useful for the power of prediction. Additionally, in this step, we apply one-hot encoding (setting up dummy variables for encoding categorical data) so that we can feed them into ML algorithms. Next, we feed the data to different ML models, including, Decision Tree (DT), Random Forest (RF), Naive Bayes (NB), Logistic Regression (LR), XGBoost (XGB), Support Vector Machine (SVM), and K-Nearest Neighbor(KNN). These models operate as a black box, and therefore, no additional information about building them is required.
The next step is the automated HPO method which is an iterative process of choosing HPs. The most commonly used performance metric, accuracy, is used in our experiments as the classifier performance. Accuracy is the proportion of correctly classified data. We also recorded the tuning time for further considerations. We leveraged GS and RS for finding the optimal set of HPs. For each experiment on the dataset, shuffle 3-fold cross-validation is applied in the training process (to prevent overfitting). Then, we test the model using testing data. In each step, the accuracy of the model is calculated. Algorithm 1 explains GRS pseudo-code starting with initializing HPs and calling RS and GS applied on models and ends with returning the final model, optimal HPs, and accuracy. In this study, the steps defined for GS were defined as 0.5 for continuous values and 1 for all discrete values except *n_ estimator* hyper-parameter in RF and XGB with steps of 5.

⋄ **Dataset** All of our experiments are conducted on MIDFIELD (Multiple-Institution Database for Investigating Engineering Longitudinal Development) [12] to provide practical examples. MIDFIELD is a longitudinal student unit-record dataset from 1988-2018 for all undergraduate, degree-seeking students at partner institutions. MIDFIELD includes everything that appears in students'

Algorithm 1 Implementation of GRS-AutoHP

Input: $List_{ML}$ of models (DT, RF, NB, LR, XGB, SVM, KNN) & raw dataset
Output: Best model along performance and optimal architecture
Call PreProcessing
for every model in the $List_{ML}$ **do**
    Call GS and RS ▷ Calculate Accuracy(Acc), HPs
    if GS.Acc larger than RS.Acc **then**
        Replace model, GS BestHP
    end if
    if RS.Acc larger than RS.Acc **then**
        Replace model, RS BestHP
    end if
end for
Return FinalModel, BestHP, Final.Acc

academic records, such as demographic data (ex. sex, age, and race/ethnicity) and information about major concentration, enrollment, graduation, school and pre-school students' performance. As previous research shows, computing majors have different patterns from other STEM majors[13], the data examined in this paper is exclusive to computing fields, with about 45k observations. MIDFIELD is used as a binary-classification problem to predict computing students' success, more specifically graduation. For our classification models, accuracy is used as the classifier performance metric. After completing each experiment, the model with the optimal ML architecture will be returned.
In this study, we compare the HPO methods (GS and RS) with the baseline is the ML model with default HPs. Since for each ML model, only a few HPs have significant impacts on the model's performance [10], in this study, we consider the main HPs of the ML classifiers for automated tuning.

4 Experimental Results

◇ **Classification Results** This section discusses the results of our experiments. As mentioned earlier, our first scenario is applying different ML algorithms on the data with their default HP values as our baseline method. The first column in Table 1 shows the accuracy for the baseline. As can be seen in the highlighted areas of the table, results indicate that RF performs better than other ML classifiers and, as a result, the final model in the baseline model selection step. Next, we implement the GS and RS on the same dataset. Columns two and three in Table 1 are the results for these two HPO methods, respectively.

Table 1: Comparison of baseline, previous work, extended work, GS and RS (%)

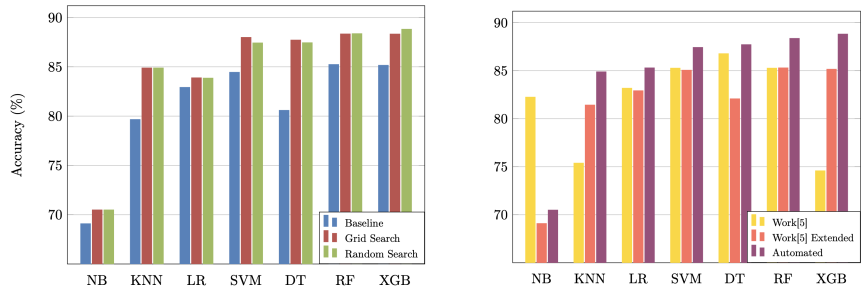
Classifier	Baseline	Work[5]	Work[5] Extended	GS	RS
NB	69.09	82.25	69.09	70.49	70.49
LR	82.92	83.18	82.92	83.89	83.86
KNN	79.66	75.38	81.43	84.89	84.89
SVM	84.45	85.27	85.06	87.99	87.43
DT	80.59	86.78	82.08	87.72	87.45
RF	**85.24**	**88.27**	**85.30**	**88.34**	88.37
XGB	85.16	74.58	85.16	**88.33**	**88.80**

The experiment results show that, regardless of the ML model, both automated HPO methods have increased the model performance. However, some models perform better than the rest. For example, NB is not a high-performing model compared to the other models. The reason is that in NB, the probability of each class given different input values requires to be calculated, and no coefficients need to be fitted by the optimization process. However, this characteristic makes the NB faster than other models. Therefore, in cases users want to use NB for their specific purpose, it can be a fast model for HP tuning (NB also has a lower number of HPs). From Table 1 we can see that after applying GS, XGB is also one of the well-performed candidates among the classifiers, while this is not the case for the baseline approach. As for the RS method, XGB is a well-performed classifier and, as a result, is the final model. This indicates that relying on the default parameters to find the final model is not always the best decision.

⋄ **Comparison with Previous Work** In this section, we compare the classification results from our experiment with previous work. In the work [5], an experiment is conducted to predict students' graduation with manual tuning using MIDFIELD (N=39k). In this study, we extended the work[5] and built models using the same configurations (N=45k).

As can be seen from the results, automated HPO beats the manual tuning even when the domain knowledge is used to tune the HPs. Automated HPO might take longer than manual tuning, but the results are promising and can guarantee performances close to the global maximum performance.

A summary of results for the experiments in this study is shown in Figure 2.



(a) Comparison with baseline approach (b) Comparison with manual tuning

Fig. 2: Comparison of automated HPO with manual and baseline approaches

5 Conclusions

GS and RS methods are applied on seven conventional ML models (DR, NB, SVM, XGB, KNN, LR, and RF) on the MIDFIELD. We use a subset of computing major students from 16 institutions across the U.S. Results show that regardless of the model leveraged, GS and RS improve the classification accuracy. Also, we see that such methods beat the manual tuning methods even when domain knowledge is used. Additionally, we observe that using HPO methods, the final selected model is XGB. However, both previous work [5] and our extended work show that RF is the well-performed method. This indicates the importance of automated HPO. We conclude that applying HPO improves the performance and helps to find the proper final model and HPs in the education field.

6 Discussion and Future Work

We demonstrated that the GS and RS methods exhibit improved prediction performance. However, these techniques spend a lot of time searching non-promising areas, leading to high tuning time. This is exacerbated when the scale of data or search space increases. Hence, such problems must be solved in a computationally efficient way to have real-time and intelligent decision-making. In our case, using large search spaces and one-hot encoding the features resulted in relatively high tuning time (especially in tree-based models and SVM). High time complexity is an issue that needs future research. Data reduction techniques using nature-inspired algorithms are being used in different applications to achieve reasonable time complexities [22]. To achieve this goal, evolutionary algorithms (EAs) have been used widely. One of the EA applications is feature selection to reduce datasets'

dimensions without decreasing the performance. Mohammadi *et al.* noted that such algorithms are robust enough to be used in different applications [23]. Going forward, we plan to examine further the impacts of using EA algorithms in educational fields to optimize the ML models' HPs.

7 Acknowledgment

All the authors would like to thank Dr. Monique Ross, Florida International University, for her inputs on deploying machine learning for *education research*.

References

1. Eduardo Fernandes, Maristela Holanda, Marcio Victorino, Vinicius Borges, Rommel Carvalho, and Gustavo Van Erven. Educational data mining: Predictive analysis of academic performance of public school students in the capital of brazil. *Journal of Business Research*, 94:335–343, 2019.
2. Alejandro Peña-Ayala. Educational data mining: A survey and a data mining-based analysis of recent works. *Expert systems with applications*, 41(4):1432–1462, 2014.
3. Adrián Domínguez, Joseba Saenz-de Navarrete, Luis De-Marcos, Luis Fernández-Sanz, Carmen Pagés, and José-Javier Martínez-Herráiz. Gamifying learning experiences: Practical implications and outcomes. *Computers & education*, 63:380–392, 2013.
4. Shahab Boumi and Adan Vela. Application of hidden markov models to quantify the impact of enrollment patterns on student performance. *International Educational Data Mining Society*, 2019.
5. Leila Zahedi, Stephanie J Lunn, S Pouyanfar, MS Ross, and MW Ohland. Leveraging machine learning techniques to analyze computing persistence in undergraduate programs. *2020 ASEE Virtual Annual Conference Content Access*, pages 1–15, 2020.
6. Matthew W Ohland, Marisa K Orr, Richard A Layton, Susan M Lord, and Russell A Long. Introducing "stickiness" as a versatile metric of engineering persistence. In *2012 Frontiers in Education Conference Proceedings*, pages 1–5. IEEE, 2012.
7. Farid Ghareh Mohammadi, Hamid R Arabnia, and M Hadi Amini. On parameter tuning in meta-learning for computer vision. In *2019 International Conference on Computational Science and Computational Intelligence (CSCI)*, pages 300–305. IEEE, 2019.
8. Vito Walter Anelli, Tommaso Di Noia, Eugenio Di Sciascio, Claudio Pomo, and Azzurra Ragone. On the discriminative power of hyper-parameters in cross-validation and how to choose them. In *Proceedings of the 13th ACM Conference on Recommender Systems*, pages 447–451, 2019.

9. Radwa Elshawi, Mohamed Maher, and Sherif Sakr. Automated machine learning: State-of-the-art and open challenges. *arXiv preprint arXiv:1906.02287*, 2019.
10. Li Yang and Abdallah Shami. On hyperparameter optimization of machine learning algorithms: Theory and practice. *Neurocomputing*, 415:295–316, 2020.
11. Gonzalo I Diaz, Achille Fokoue-Nkoutche, Giacomo Nannicini, and Horst Samulowitz. An effective algorithm for hyperparameter optimization of neural networks. *IBM Journal of Research and Development*, 61(4/5):9–1, 2017.
12. Matthew W Ohland, Guili Zhang, Brian Thorndyke, and Timothy J Anderson. The creation of the multiple-institution database for investigating engineering longitudinal development (midfield). *Proc. Amer. Soc. Eng. Ed*, 2004.
13. Leila Zahedi, Hossein Ebrahiminejad, Monique S. Ross, Matthew W. Ohland, and Stephanie J. Lunn. Multi-institution study of student demographics and stickiness of computing majors in the usa. *2021 CoNECD*, pages 1–7, January 2021.
14. Max Kuhn, Kjell Johnson, et al. *Applied predictive modeling*, volume 26. Springer, 2013.
15. Matthias Feurer and Frank Hutter. Hyperparameter optimization. In *Automated Machine Learning*, pages 3–33. Springer, Cham, 2019.
16. Steven Abreu. Automated architecture design for deep neural networks. *arXiv preprint arXiv:1908.10714*, 2019.
17. Skogby Steinholtz Olof. A comparative study of black-box optimization algorithms for tuning of hyper-parameters in deep neural networks, 2018.
18. Marc Claesen, Jaak Simm, Dusan Popovic, Yves Moreau, and Bart De Moor. Easy hyperparameter search using optunity. *arXiv preprint arXiv:1412.1114*, 2014.
19. Iwan Syarif, Adam Prugel-Bennett, and Gary Wills. Svm parameter optimization using grid search and genetic algorithm to improve classification performance. *Telkomnika*, 14(4):1502, 2016.
20. F Hutter, R Caruana, R Bardenet, M Bilenko, I Guyon, B Kegl, and H Larochelle. Automl 2014@ icml. In *AutoML 2014 Workshop@ ICML*, 2014.
21. James Bergstra and Yoshua Bengio. Random search for hyper-parameter optimization. *Journal of machine learning research*, 13(2), 2012.
22. Farid Ghareh Mohammadi, M Hadi Amini, and Hamid R Arabnia. Evolutionary computation, optimization, and learning algorithms for data science. In *Optimization, Learning, and Control for Interdependent Complex Networks*, pages 37–65. Springer, 2020.
23. Farid Ghareh Mohammadi, M Hadi Amini, and Hamid R Arabnia. Applications of nature-inspired algorithms for dimension reduction: enabling efficient data analytics. In *Optimization, Learning, and Control for Interdependent Complex Networks*, pages 67–84. Springer, 2020.