Global Journal of Computer Science and Technology: D
Neural & Artificial Intelligence
Volume 21 Issue 2 Version 1.0 Year 2021
Type: Double Blind Peer Reviewed International Research Journal
Publisher: Global Journals
Online ISSN: 0975-4172 & Print ISSN: 0975-4350

Machine Learning Model Optimization with Hyper Parameter Tuning Approach

By Md Riyad Hossain & Dr. Douglas Timmer

UTRGV

Abstract- Hyper-parameters tuning is a key step to find the optimal machine learning parameters. Determining the best hyper-parameters takes a good deal of time, especially when the objective functions are costly to determine, or a large number of parameters are required to be tuned. In contrast to the conventional machine learning algorithms, Neural Network requires tuning hyper-parameters more because it has to process a lot of parameters together, and depending on the fine tuning, the accuracy of the model can be varied in between 25%-90%.

A few of the most effective techniques for tuning hyper-parameters in the Deep learning methods are: Grid search, Random forest, Bayesian optimization, etc. Every method has some advantages and disadvantages over others. For example: Grid search has proven to be an effective technique to tune hyper-parameters, along with drawbacks like trying too many combinations, and performing poorly when it is required to tune many parameters at a time. In our work, we will determine, show and analyze the efficiencies of a real-world synthetic polymer dataset for different parameters and tuning methods.

Keywords: machine learning, hyper parameter optimization, grid search, random search, BO-GP.

GJCST-D Classification: H.5.4

Strictly as per the compliance and regulations of:

Machine Learning Model Optimization with Hyper Parameter Tuning Approach

Md Riyad Hossain [α] & Dr. Douglas Timmer [σ]

Abstract- Hyper-parameters tuning is a key step to find the optimal machine learning parameters. Determining the best hyper-parameters takes a good deal of time, especially when the objective functions are costly to determine, or a large number of parameters are required to be tuned. In contrast to the conventional machine learning algorithms, Neural Network requires tuning hyper-parameters more because it has to process a lot of parameters together, and depending on the fine tuning, the accuracy of the model can be varied in between 25%-90%.

A few of the most effective techniques for tuning hyper-parameters in the Deep learning methods are: Grid search, Random forest, Bayesian optimization, etc. Every method has some advantages and disadvantages over others. For example: Grid search has proven to be an effective technique to tune hyper-parameters, along with drawbacks like trying too many combinations, and performing poorly when it is required to tune many parameters at a time. In our work, we will determine, show and analyze the efficiencies of a real-world synthetic polymer dataset for different parameters and tuning methods.

Keywords: machine learning, hyper parameter optimization, grid search, random search, BO-GP.

I. Introduction

In the era of Machine learning, performance (based on accuracy and computing time) is very important. The growing number of tuning parameters associated with the Machine learning models is tedious and time-consuming to set by standard optimization techniques. Researchers working with ML models often spend long hours to find the perfect combination of hyper-parameters [1]. If we think w, x, y, z as the parameters of the model, and if all of these parameters are integers ranging from 0.0001 to say 5.00, then hyperparameter tuning is the finding the best combinations to make the objective function optimal.

One of the major difficulties in working with the Machine learning problem is tuning hyperparameters. These are the design parameters that could directly affect the training outcome. The conversion from a non-tuned Machine learning model to a tuned ML model is like learning to predict everything accurately from predicting nothing correctly [2]. There are two types of parameters in ML models: Hyperparameters, and Model parameters. Hyperparameters are arbitrarily set by the user even before starting to train the model, whereas, the model parameters are learned during the training.

The quality of a predictive model mostly depends on the configuration of its hyperparameters, but it is often difficult to know how these hyperparameters interact with each other to affect the final results of the model [14]. To determine accuracy and make a comparison between two models it is always better to make comparisons between two models with both of the models' parameters tuned. It would be unfair to compare a Decision Tree model with the best parameter against an ANN model whose hyperparameters haven't been optimized yet.

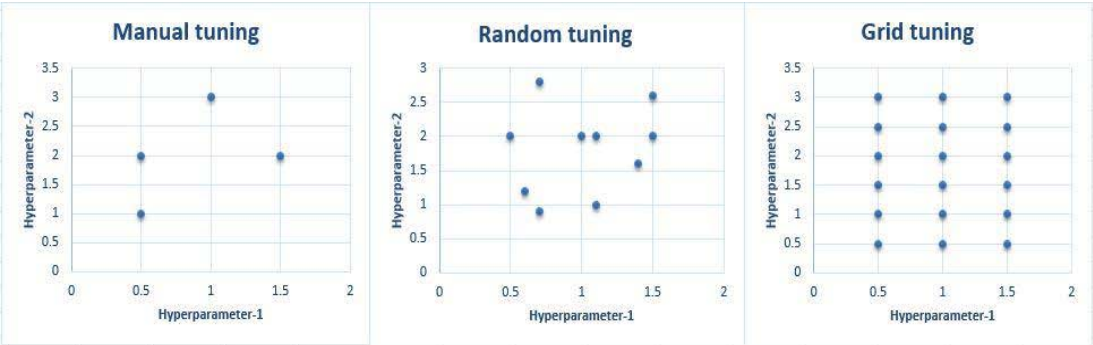

Figure 1: (a) Manual tuning (b) Random tuning (c) Grid tuning approach [From left to Right]

II. Literature Review

The hyperparameter tuning, due to its importance, has changed to a new interesting topic in the ML community. The hyperparameter tuning algorithms are either model-free or model-based.

Author α: Graduate Research Assistant, Department of Manufacturing Engineering, UTRGV, USA. e-mail: riyad35@gmail.com
Author σ: Professor, Department of Manufacturing Engineering, UTRGV, USA.

Model-free algorithms are free of using knowledge about the solution space extracted during the optimization; a few of this category includes manual search [4], random search [2, 6-7], and grid search [5]. In the Manual search categories, we assume the values of the parameters based on our previous experience. In this technique, the user allows some sets of hyperparameters based on judgments or previous experience, trains the algorithm by them, observes the performance, keeps repeating to train the model until achieving a satisfactory accuracy and then selects the best set of hyperparameters that gives the maximum accuracy. However, this technique is heavily dependent on the judgment and previous expertise and its reliability is dependent on the correctness of the previous knowledge [3]. Some of the few of the main parameters used by Random forest classifiers are: criterion, max_depth, n_estimators, min_samples_split etc.

In the Random search, we train and test our model based on some random combinations of the hyperparameters. This method is better used to identify new combinations of the parameters or to discover new hyperparameters. Although it may take more time to process, it often leads to better performance. Bergstra et al. (2012) in their work mentioned that, over the same domain, random search is able to find models that are as good as or even better in a reduced computation time. After granting the same computational budget for the random search, it was evident that random search can find better models by effectively searching for larger and less promising configuration spaces [16]. Random Search, which is developed based on grid research, sets up a grid of hyper-parameter values and selects random combinations to train the algorithm; Bergstra et al. (2011) [2].

In the grid search, the user sets a grid of hyperparameters and trains the model based on each possible combination. Amirabadi et al. (2020) proposes two novel suboptimal grid search techniques on the four separate dataset to show the efficiency of their hyperparameter tuning model and later compare it with some of the other recently published work. The main drawback of the grid search method is its high complexity. It is commonly used when there are a few numbers of hyperparameters to be tuned. In other words, grid search works well when the best combinations are already determined. Some of the similar works of grid search applications have been reported by Zhang et al. (2014) [17], Ghawi et al. (2019) [18], and Beyramysoltan et al. (2013) [19].

Zhang et al. (2019) [20] in their work reported a few of the drawbacks of the existing hyperparameter tuning methods. In their work, they mentioned grid search as an ad-hoc process, as it traverses all the possible combinations, and the entire procedure requires a lot of time. Andradóttir (2014) [13] shows that Random Search (RS) eradicates some of the limitations of the grid search technique to an extent. RS can reduce the overall time consumption, but the main disadvantage is that it cannot converge to the global optimal value.

The combination of randomly selected hyper-parameters can never guarantee a steady and widely acceptable result. That's why, apart from the manually tuning methods, automated tuning methods are becoming more and more popular in recent times; snoek et al. (2015) [10]. Bayesian Optimization is one of the most widely used automated hyperparameter tuning methods to find the global optimum in fewer steps. However, Bayesian optimization's results are sensitive to the parameters of the surrogate model and the accuracy is greatly depending on the quality of the learning model; Amirabadi et al. (2020) [3].

To minimize the error function of hyperparameter values, Bayesian optimization adopts probabilistic surrogate models like Gaussian processes. Through precise exploration and development, an alternative model of hyperparameter space is established; Eggensperger et al. (2013) [8]. However, probabilistic surrogates need accurate estimations of sufficient statistics of error function distribution. So, a sizable number of hyperparameters is required to evaluate the estimations and this method doesn't work well when there is to process myriad hyperparameters altogether.

III. Methodology

a) Dataset description

Denier: Denier is a weight measurement usually refers to the thickness of the threads. It is the weight (grams) of a single optical fiber for 9 kilometers. If we have a 9 km fiber weighs 1 gram, this fiber has a denier of 1, or 1D. A fiber with less than 1 gram weight calls Microfibers [22]. Microfibers become a new development trend in the synthetic polymer industry. The higher the denier is, the more thick and strong the fiber is. Conversely, less denier means that the fiber/fabric will be softer and more transparent. Fine denier fibers are becoming a new standard and are very useful for the development of new textiles with excellent performance [21].

Breaking Elongation (%): Elongation at break is one of the few main quality parameters of any synthetic fiber [24]. It is the percentage of elongation at break. Fiber elongation partly reflects the extent of stretching a filament under a certain loading condition. Fibers with high elongation at break are determined to be easily stretched under a predetermined load. Fibers showing these characteristics are known to be flexible. The elongation behavior of any single fiber can be complex because of its multiplicity of structural factors affecting it. Moreover, a cotton fiber comes up with a natural crimp, which is important for fibers to stick together while undergoing other production processes [23]. If L is the

length of the fiber, then the equation for the percentages of the breaking elongation would be:

$$E_{Breaking\ \ elongation} = \frac{\Delta L_{Break}}{L_0} * 100\%$$

Breaking elongation for the cotton fiber might be varied from 5% to 10%, which is significantly lower than that of wool fibers (25%-45%), and much lower than polyester fibers (typically over 50%).

Breaking force (cN) and Tenacity (cN/tex): Breaking tenacity is the maximum load that a single fiber can withstand before breaking. For the Polypropylene and PET staple fibers, 10 mm lengths sample filaments is drawn until failure. Breaking tenacity is measured in grams/denier. Very small forces are encountered when evaluating fiber properties, so an instrument with gram-level accuracy is required [25]. The tenacity of virgin PP fibers is about 5–8 g/den, and the elongation at break is about 100%. At the same time, the tenacity of recycled PET is about 3.5-5.7 g/den; the elongation at break usually exceeds 100%.

Draw Ratio: The drawing ration is the ratio of the diameter of the initial blank form to the diameter of the drawn part. The limiting drawing ratio (Capstan speed/Nip reel speed) for the extruder section is between 1.6 and 2.2 [26], whereas, for the stretching section it is in between 3 and 4.

b) Hyper-parameter Optimization (HPO)

The purpose of hyperparameter optimization is to find the global optimal value x^* of the objective function *f*(**x**) can be evaluated for any arbitrary $\boldsymbol{x} \in \boldsymbol{X}$, $x^* = \arg min_{x \in X} f(x)$, and X is a hyperparameter space that can contain categorical, discrete, and continuous variables [27].In order to construct the design of different machine learning models, the application of effective hyperparameter optimization techniques can simplify the process of identifying the best hyperparameters for the models. HPO contains four major components: First, an estimator that could be a regressor or any classifier with one or more objective functions, second: a search space, Third: an optimization method to find the best combinations, and Fourth: a function to make a comparison between the effectiveness of various hyperparameter configurations [28]. Some of the common hyperparameter techniques is discussed below:

Grid Search: Grid search is a process that exhaustively searches a manually specified subset of the hyperparameter space of the target algorithm [30]. A traditional approach to finding the optimum is to do a grid search, for example, to run experiments or processes on a number of conditions, for example, if there are three factors, a 15 × 15× 15 would mean performing 3375 experiments under different conditions. [32]. Grid search is more practical when [31]: (1) the total number of parameters in the model is small, say M <10. The grid is M-dimensional, so the number of test solutions is proportional to L^M, where L is the number of test solutions along each dimension of the grid. (2) The solution is known to be within a specific range of values, which can be used to define the limits of the grid. (3) The direct problem d = g (m) can be computed quickly enough that the time required to compute L^M from them is not prohibitive. (4) The error function E (m) is uniform on the scale of the grid spacing, Δm, so that the minimum is not lost because the grid spacing is too coarse.

There are many problems with the grid search method. The first is that the number of experiments can be prohibitive if there are several factors. The second is that there can be significant experimental error, which means that if the experiments are repeated under identical conditions, different responses can be obtained; therefore, choosing the best point on the grid can be misleading, especially if the optimum is fairly flat. The third is that the initial grid may be too small for the number of experiments to be feasible, and it could lose characteristics close to the optimum or find a false (local) optimum [32].

Random Search: Random search [33] is a basic improvement on grid search. It indicates a randomized search over hyper-parameters from certain distributions over possible parameter values. The searching process continues till the predetermined budget is exhausted, or until the desired accuracy is reached. This methods are the simplest stochastic optimization and are very useful for certain problems, such as small search space and fast-running simulation. RS finds a value for each hyperparameter, prior to the probability distribution function. Both the GS and RS estimate the cost measure based on the produced hyperparameter sets. Although RS is simple, it has proven to be more effective than Grid search in many of the cases [33].

Random search has been shown to provide better results due to several benefits: first, the budget can be set independently according to the distribution of the search space, therefore, random search can work better especially when multiple hyper-parameters are not uniformly distributed [34]. Second: Because each evaluation is independent, it is easy to parallelize and allocate resources. Unlike GS, RS samples a number of parameter combinations from a defined distribution, which maximizes system efficiency by reducing the likelihood of wasting a lot of time in a small, underperforming area. In addition, this method can detect global optimum values or close to global if given a sufficient budget. Third, although getting optimal results using random search is not promising, more time consumption will lead to a greater likelihood of finding the best hyperparameter set, whereas longer search times cannot guarantee better results in Grid searches.

The use of random search is recommended in the early stages of HPO to narrow the search space quickly, before using guided algorithms to get better results. The main drawback [28] of RS and GS is that each evaluation in its iteration does not depend on previous evaluations; thus, they waste time evaluating underperforming areas of the search space.

Bayesian Optimization: Bayesian optimization (BO) is a commonly used reprocessing algorithm for HPO problems. Unlike GS and RS, BO determines future assessment levels based on the previous results. To determine the following parameters of the hyperparameter, BO uses two key factors: a surrogate model and an acquisition function. The division model aims to match all the points that are now seen in the objective function. The acquisition function determines the use of different points, balancing exploration and exploitation. The BO model balances the search and use process to identify the best possible area and avoid losing the best configuration in undeveloped areas [35].

The basic BO method works as follows: (i) Building a reduced-order probabilistic model (ROM) of the objective function. (ii) Finding the best hyperparameter values in the ROM model. (iii) Applying those optimal values to the objective function. (iv) Updating the ROM model with the new set of results. (v) Repeating above steps until achieving maximum number of iterations.

BO is more efficient than GS and RS because it can detect optimal combinations of hyperparameters by analyzing previously tested values, and running the surrogate model is usually much cheaper than running the objective function as a whole. However, because Bayesian optimization models are run based on previously tested values, it is difficult to belong to them with parallel sequential methods; but they are generally able to detect optimal close hyperparameter combinations in a few iterations [36]. Common substitution models for BO include the Gaussian process (GP) [37], random forest (RF) [38], and Parzen estimator (TPE) [39]. Therefore, there are three main BO algorithms based on their substitution models: BO-GP, BO-RF, BO-TPE. GP is an attractive reduced order model of BO that can be used to quantify forecast uncertainty. This is not a parametric model and the number of its parameters depends only on the input points. With the right kernel function, your GP can take advantage of the data structure. However, the GP also has disadvantages. For example, it is conceptually difficult to understand with BO theory. In addition, its low scalability with large dimensions or a large number of data points is another important issue [36].

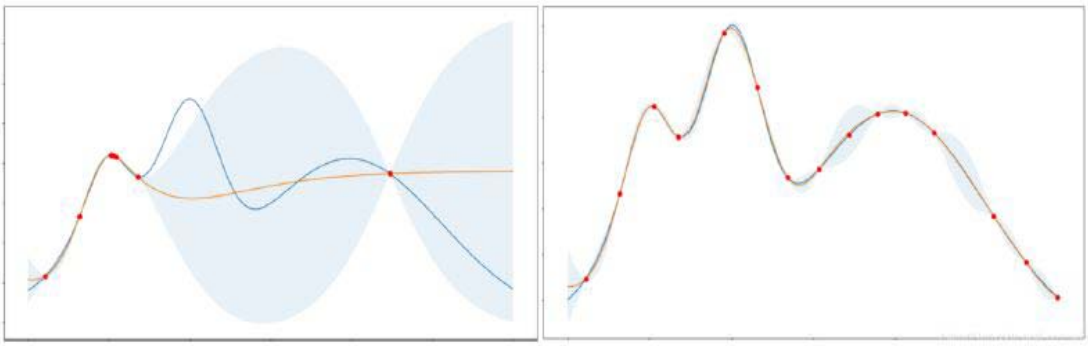

Figure 2: Exploration-based (left) and exploitation-based Bayesian optimization (right); the shadow indicates uncertainty (Yang and Shami, 2020)

IV. Applying HPO in ML Models

In order to put the theory into practice, several experiments have been performed on an industrial-based synthetic polymer model. This section describes experiments with four different HPO techniques on three general and representative ML algorithms. In the first part of the section, we discussed the experimental setup and the main HPO process. In the second part, we compare and analyze the results of the application of different HPO methods.

Table 1: An overview of common ML models we used in this work, their hyper-parameters are listed below:

ML Model	Hyper-parameter
RF Regressor	n_estimators, max_depth, min_samples_split, min_samples_leaf, criterion, max_features
SVM Regressor	C, kernel, epsilon
KNN Regressor	n neighbors

Table 2: Performance evaluation of applying HPO methods to the regressor on the synthetic polymer dataset

Model	Optimization Algorithm	MSE	Cycle time (s)
RF Regressor	Default HPs	0.42	0.07
	GS	0.32	3.62
	RS	0.65	2.8
	BO-GP	0.44	20.5
SVM Regressor	Default HPs	36.54	0.08
	GS	0.93	4
	RS	8.29	1
	BO-GP	2.21	12
KNN	Default HPs	31.01	0.1
	GS	29.45	0.3
	RS	31.02	0.43
	BO-GP	33.44	0.7
ANN	Default HPs	6.72	0.5
	GS	0.47	2
	RS	0.97	2.8
	BO-GP	3.58	30

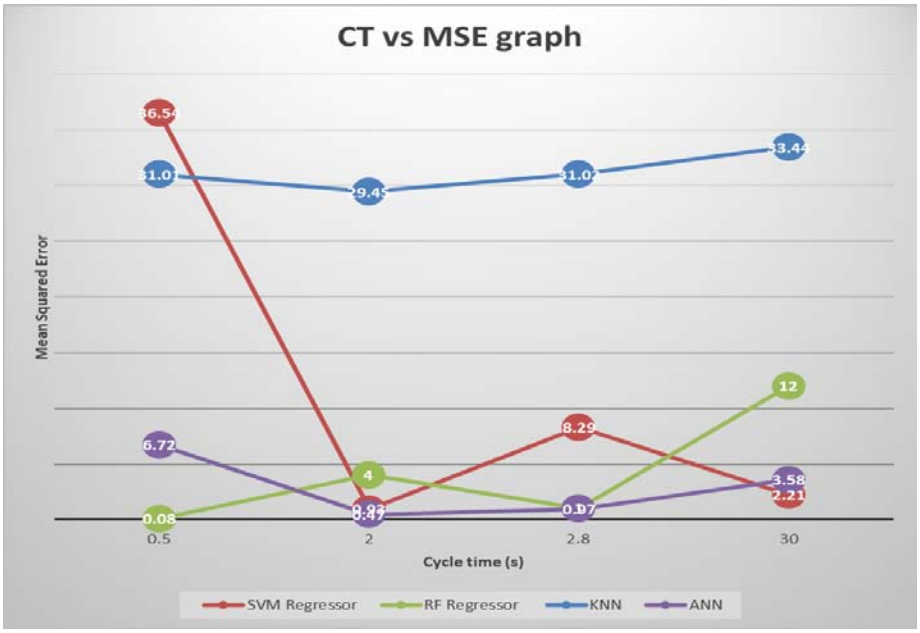

Figure 3: Cycle time vs. MSE graph

V. Discussion & Conclusion

Machine learning has become the primary strategy for dealing with data problems and is widely used in various applications. To apply ML models to practical problems, hyperparameters must be tuned to handle specific datasets. However, as the size of the generated data increases greatly in real life, and manual tuning of hyperparameters is extremely computationally expensive, it has become essential to optimize the hyperparameters by an automatic process. In this work, we used hyperparameter techniques in the ML model to find the best set of hyperparameters. Our data set was small, and in this small datset we can see that the randomly selected subsets are very representative for the given data set, as they can effectively optimize all types of hyperparameters. Our future work would be to test our model on a much larger data set and see the feedback.

Conflict of Interest: The authors whose names are listed in this work certify that they have no affiliations with or involvement in any organization or entity with any financial interest, or non-financial interest in the subject matter or materials discussed in this manuscript.

References Références Referencias

1. Cho, H., Kim, Y., Lee, E., Choi, D., Lee, Y., & Rhee, W. (2020). Basic Enhancement Strategies When Using Bayesian Optimization for Hyperparameter Tuning of Deep Neural Networks. *IEEE Access, 8*, 52588-52608. doi:10.1109/access.2020.2981072
2. J.S. Bergstra, R. Bardenet, Y. Bengio, B. Kégl, Algorithms for hyperparameter optimization, in:

Advances in Neural Information Processing Systems, 2011, pp. 2546–2554.
3. Amirabadi, M., Kahaei, M., & Nezamalhosseini, S. (2020). Novel suboptimal approaches for hyperparameter tuning of deep neural network [under the shelf of optical communication]. *Physical Communication, 41*, 101057. doi:10.1016/j.phycom.2020.101057
4. F. Hutter, J. Lücke, L. Schmidt-Thieme, Beyond manual tuning of hyperparameters, DISKI 29 (4) (2015) 329–337.
5. F. Friedrichs, C. Igel, Evolutionary tuning of multiple SVM parameters, Neurocomputing 64 (2005) 107–117.
6. R.G. Mantovani, A.L. Rossi, J. Vanschoren, B. Bischl, A.C. De Carvalho, Effectiveness of random search in SVM hyper-parameter tuning, in: 2015 International Joint Conference on Neural Networks (IJCNN), 2015, pp. 1–8.
7. L. Li, A. Talwalkar, Random search and reproducibility for neural architecture search, 2019, arXiv preprint arXiv: 1902.07638.
8. K. Eggensperger, M. Feurer, F. Hutter, J. Bergstra, J. Snoek, H. Hoos, K. Leyton-Brown, Towards an empirical foundation for assessing bayesian optimization of hyperparameters. In NIPS workshop on Bayesian Optimization in Theory and Practice (Vol. 10, 3), 2013.
9. H. Larochelle, D. Erhan, A. Courville, J. Bergstra, Y. Bengio, An empirical evaluation of deep architectures on problems with many factors of variation, in: Proceedings of the 24th International Conference on Machine Learning, ACM, 2007, pp. 473–480.
10. J. Snoek, O. Rippel, K. Swersky, R. Kiros, N. Satish, N. Sundaram, et al., Scalable bayesian optimization using deep neural networks, in: International conference on machine learning, 2015, pp. 2171-2180.
11. Jones, D.R.; Schonlau, M.; Welch, W.J. Efficient global optimization of expensive black-box functions. J. Glob. Optim. 1998, 13, 455–492.
12. Calandra, R.; Peters, J.; Rasmussen, C.E.; Deisenroth, M.P. Manifold Gaussian processes for regression. In Proceedings of the 2016 International Joint Conference on Neural Networks, Vancouver, BC, Canada, 24–29 July 2016; pp. 3338–3345.
13. Andrad_ottir, S.: A review of random search methods. In: Handbook of Simulation Optimization, pp. 277{292. Springer (2015).
14. Li, L.; Jamieson, K.; DeSalvo, G.; Rostamizadeh, A.; Talwalker, A. Hyperband: A Novel Bandit-Based Approach to Hyperparameter Optimization. Journal of Machine Learning Research 18 (2018) 1-52.
15. A. Klein, S. Falkner, J. T. Springenberg, and F. Hutter. Learning curve prediction with Bayesian neural networks. In International Conference On Learning Representation (ICLR), 2017.
16. J.S. Bergstra, Y. Bengio. Random Search for Hyper-Parameter Optimization, in: Journal of Machine Learning Research 13 (2012) 281-305
17. Zhang, H., Chen, L., Qu, Y., Zhao, G., & Guo, Z. (2014). Support Vector Regression Based on Grid-Search Method for Short-Term Wind Power Forecasting. *Journal of Applied Mathematics, 2014*, 1-11. doi:10.1155/2014/835791
18. Ghawi, R., & Pfeffer, J. (2019). *Efficient Hyperparameter Tuning with Grid Search for Text Categorization using kNN Approach with BM25 Similarity. Open Computer Science, 9(1), 160–180.* doi:10.1515/comp-2019-0011
19. Beyramysoltan, S., Rajkó, R., & Abdollahi, H. (2013). *Investigation of the equality constraint effect on the reduction of the rotational ambiguity in three-component system using a novel grid search method. Analytica Chimica Acta, 791, 25–35.* Doi: 10.1016/j.aca.2013.06.043
20. Zhang, X., Chen, X., Yao, L., Ge, C., & Dong, M. (2019). Deep Neural Network Hyperparameter Optimization with Orthogonal Array Tuning. *Communications in Computer and Information Science Neural Information Processing,* 287-295. doi:10.1007/978-3-030-36808-1_31
21. Zhang, C., Liu, Y., Liu, S. *et al.* Crystalline behaviors and phase transition during the manufacture of fine denier PA6 fibers. *Sci. China Ser. B-Chem.* 52, 1835 (2009). https://doi.org/10.1007/s11426-009-0242-5
22. Joe. (2020, May 5). *What Is Denier Rating? Why Does It Matter To You?* Digi Travelist. https://www.digitravelist.com/what-is-denier-rating/.
23. Elmogahzy, Yehia (2018). *Handbook of Properties of Textile and Technical Fibres || Tensile properties of cotton fibers. , (), 223–273.* doi:10.1016/B978-0-08-101272-7.00007-9
24. Tyagi, G.K. (2010). *Advances in Yarn Spinning Technology || Yarn structure and properties from different spinning techniques., (), 119–154.* doi:10.1533/9780857090218.1.119
25. Blair, K. (2007). Materials and design for sports apparel. *Materials in Sports Equipment,* 60-86. doi:10.1533/9781845693664.1.60
26. Swift, K., & Booker, J. (2013). Forming Processes. *Manufacturing Process Selection Handbook,* 93-140. doi:10.1016/b978-0-08-099360-7.00004-5.
27. Cho, H., Kim, Y., Lee, E., Choi, D., Lee, Y., & Rhee, W. (2020). Basic Enhancement Strategies When Using Bayesian Optimization for Hyperparameter Tuning of Deep Neural Networks. *IEEE Access, 8*, 52588-52608. doi: 10.1109/access. 2020.2981072.
28. Yang, L., &Shami, A. (2020). On hyperparameter optimization of machine learning algorithms: Theory

and practice. *Neurocomputing, 415*, 295-316. doi:10.1016/j.neucom.2020.07.061.
29. Chan, S., &Treleaven, P. (2015). Continuous Model Selection for Large-Scale Recommender Systems. *Handbook of Statistics Big Data Analytics,* 107-124. doi:10.1016/b978-0-444-63492-4.00005-8.
30. Menke, W. (2012). Nonlinear Inverse Problems. Geophysical Data Analysis: Discrete Inverse Theory, 163-188. doi:10.1016/b978-0-12-397160-9.00009-6.
31. Brereton, R. (2009). Steepest Ascent, Steepest Descent, and Gradient Methods. Comprehensive Chemometrics, 577-590. doi:10.1016/b978-0444 52701-1.00037-5.
32. Bergstra, J., Bengio, Y., 2012. Random search for hyper-parameter optimization. J. Mach. Learn. Res. 13, 281–305. ISSN 1532-4435. URL http://dl. acm.org/citation.cfm?id=2188385.2188395
33. Yu, T., & Zhu, H. (2020). Hyper-Parameter Optimization: A Review of Algorithms and Applications, https://arxiv.org/abs/2003.05689.
34. E. Hazan, A. Klivans, and Y. Yuan, Hyperparameter optimization: a spectral approach, arXiv preprint arXiv: 1706.00764, (2017). https://arxiv.org/abs 1706.00764.
35. Hutter, F., Kotthoff, L., &Vanschoren, J. (2019). *Automated Machine Learning Methods, Systems, Challenges*. Cham: Springer International Publishing.
36. Seeger, M. (2004). Gaussian Processes For Machine Learning. International Journal of Neural Systems, 14(02), 69-106. doi:10.1142/s012906570 4001899
37. Hutter F., Hoos H.H., Leyton-Brown K. (2011) Sequential Model-Based Optimization for General Algorithm Configuration. In: Coello C.A.C. (eds) Learning and Intelligent Optimization. LION 2011. Lecture Notes in Computer Science, vol 6683. Springer, Berlin, Heidelberg. https://doi.org/10. 1007/978-3-642-25566-3_40.
38. Bergstra J, Bardenet R, Bengio Y, Kégl B (2011) Algorithms for hyper-parameter optimization. Adv Neural Inf Process Syst (NIPS) 24:2546–2554.