Learning multiple defaults for machine learning algorithms

Pfisterer, F.; Rijn, J.N. van; Probst, P.; Müller, A.C.; Bischl, B.; Chicano, F.

Learning Multiple Defaults for Machine Learning Algorithms

Florian Pfisterer
Ludwig-Maximilians-University
Munich, Germany

Jan N. van Rijn
LIACS, Leiden University
Leiden, Netherlands

Philipp Probst
Benediktbeuern, Germany

Andreas C. Müller
Microsoft
Sunnyvale, U.S.A.

Bernd Bischl
Ludwig-Maximilians-University
Munich, Germany

ABSTRACT

Modern machine learning methods highly depend on their hyperparameter configurations for optimal performance. A widely used approach to selecting a configuration is using default settings, often proposed along with the publication of a new algorithm. Those default values are usually chosen in an ad-hoc manner to work on a wide variety of datasets. Different automatic hyperparameter configuration algorithms which select an optimal configuration per dataset have been proposed, but despite its importance, tuning is often skipped in applications because of additional run time, complexity, and experimental design questions. Instead, the learner is often applied in its defaults. This principled approach usually improves performance but adds additional algorithmic complexity and computational costs to the training procedure. We propose and study using a set of complementary default values, learned from a large database of prior empirical results as an alternative. Selecting an appropriate configuration on a new dataset then requires only a simple, efficient, and embarrassingly parallel search over this set. To demonstrate the effectiveness and efficiency of the approach, we compare learned sets of configurations to random search and Bayesian optimization. We show that sets of defaults can improve performance while being easy to deploy in comparison to more complex methods.

CCS CONCEPTS

• **Computing methodologies** → **Supervised learning by classification**.

KEYWORDS

AutoML, Hyperparameter Optimization, Metalearning

ACM Reference Format:
Florian Pfisterer, Jan N. van Rijn, Philipp Probst, Andreas C. Müller, and Bernd Bischl. 2021. Learning Multiple Defaults for Machine Learning Algorithms. In *2021 Genetic and Evolutionary Computation Conference Companion (GECCO '21 Companion), July 10–14, 2021, Lille, France.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3449726.3459523

GECCO '21 Companion, July 10–14, 2021, Lille, France

ACM ISBN 978-1-4503-8351-6/21/07.
https://doi.org/10.1145/3449726.3459523

1 INTRODUCTION

Hyperparameter settings for machine learning algorithms are often optimized via hyperparameter optimization e.g. using random search, Bayesian optimization, or meta learning. While not tuning parameters at all can be detrimental, defaults provide a simple and fast fall-back, that is easy to implement and use while providing strong anytime performance. We describe a general, learner-agnostic procedure, to (meta-)learn not one, but a (sequential) list of default configurations, which complement each other. These sets are ordered so that the earlier elements in the sequence provide greater benefits on average.[1] While traditional optimization methods are to be preferred when time and expertise are available, we conjecture that sets of defaults work well across a large variety of datasets. We leverage a large set of historic performance results of prior experiments that are available on OpenML [4]. Several approaches attempt to combine the paradigms of meta-learning and hyperparameter optimization, for example by warm starting hyperparameter optimization methods [2, 5]. While all these methods yield convincing results, they are by no means easy to deploy. Similar to our work, Wistuba et al. [6] learn a set of defaults from a fixed grid of evaluations, requiring hyperparameters evaluated on a grid across several datasets scaling exponentially with hyperparameter dimensionality. This is practically infeasible when there are large numbers of hyperparameters.

2 METHOD

Consider a target variable y, a feature vector $\boldsymbol{x}$, and an unknown joint distribution P on $(\boldsymbol{x}, y)$, from which we have sampled a i.i.d dataset $\mathcal{D}$. A machine learning *algorithm* $\mathcal{A}_{\boldsymbol{\lambda}}(\mathcal{D})$ learns a prediction model $\hat{f}(\boldsymbol{x})$. $\mathcal{A}_{\boldsymbol{\lambda}}$ is controlled by a multi-dimensional hyperparameter configuration $\boldsymbol{\lambda} \in \Lambda$ of length D, where Λ_j is usually a bounded real or integer interval, or a finite set of categorical values. We are interested in estimating the expected risk of the inducing algorithm w.r.t. $\boldsymbol{\lambda}$ on new data, also sampled from $\mathcal{P}$:$R_{\mathcal{P}}(\boldsymbol{\lambda}) = E_{\mathcal{P}}(L(y, \mathcal{A}_{\boldsymbol{\lambda}}(\mathcal{D})(\boldsymbol{x})))$, where the expectation above is taken over all data sets $\mathcal{D}$ from $\mathcal{P}$ and the test observation $(\boldsymbol{x}, y)$. Thus, $R_{\mathcal{P}}(\boldsymbol{\lambda})$ quantifies the expected predictive performance associated with a hyperparameter configuration $\boldsymbol{\lambda}$ for a given data distribution, learning algorithm and performance measure. In practice, given K different data sets we define K hyperparameter risk mappings: $R_k(\boldsymbol{\lambda}) = E_{\mathcal{P}_k}(L(y, \mathcal{A}_{\boldsymbol{\lambda}}(\mathcal{D})(\boldsymbol{x})))$ and the average risk of $\boldsymbol{\lambda}$ over K data sets: $R(\boldsymbol{\lambda}) = \frac{1}{K}\sum_{k=1}^{K} R_k(\boldsymbol{\lambda})$. Our goal now is to find a fixed-size set Λ_{def} of size T, that works well over a variety of datasets, in the sense that for each dataset $\mathcal{D}$, Λ_{def} contains at least one configuration that works well on $\mathcal{D}$. The risk of a set of configurations Λ_{def} of size T, aggregation function h (e.g.

[1] Full version of this article: https://arxiv.org/abs/1811.09409

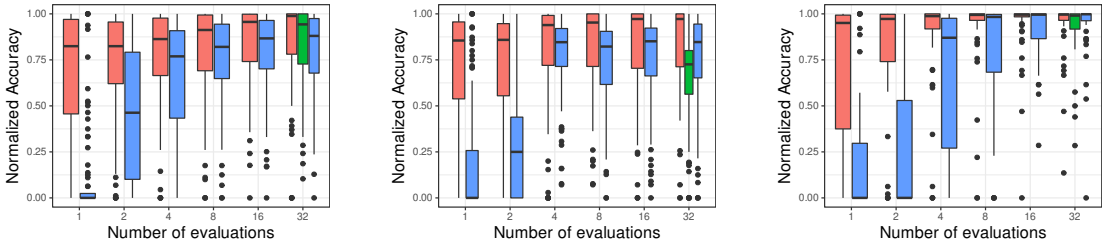

Figure 1: *Defaults* (red), *random search* (blue) and *Bayesian optimization* (green) across several budgets for Adaboost (left), Random Forest (middle) and SVM (right)

mean) and datasets $1, \ldots, K$ is then given by:

$$G(\Lambda_{\text{def}}) = h\left(\min_{j=1,\ldots,n} R_1(\boldsymbol{\lambda}_t), \ldots, \min_{t=1,\ldots,T} R_K(\boldsymbol{\lambda}_t))\right)$$

Finding an optimal subset Λ_{def} defines a (meta)-learning problem, that can be solved exactly or using a greedy approximation.

The exact version can be formulated as an instance of Mixed Integer Programming. In order to obtain a set of n defaults, the goal is to minimize

$$\sum_{k=1}^{K} \sum_{m=1}^{M} \Psi_{k,m} \cdot R_k(\boldsymbol{\lambda}_m) \tag{1}$$

subject to

$$\sum_{m=1}^{M} \phi_m = n \qquad \forall k : \forall m : \Psi_{k,m} \geq 0$$

$$\forall k : \forall m : \Psi_{k,m} \geq \phi_m - \sum_{s \in Q(k,m)} \phi_s \qquad \forall k : \sum_{m=1}^{M} \Psi_{k,m} = 1$$

After the optimization procedure, element $\Psi_{k,m}$ will be 1 if and only if configuration Λ_m has the lowest risk on distribution i out of all the configurations that are in the set of defaults. ϕ_m is an auxiliary variable. Since the exact solution is computationally prohibitively expensive, we adopt a greedy procedure for t = $1, \ldots, T$:

$$\boldsymbol{\lambda}_{\text{def},t} := \arg\min_{\boldsymbol{\lambda} \in \Lambda} \; G(\{\boldsymbol{\lambda}\} \cup \Lambda_{\text{def},t-1}) \tag{2}$$

$$\Lambda_{\text{def},t} := \{\boldsymbol{\lambda}_{\text{def},1}, \ldots, \boldsymbol{\lambda}_{\text{def},t}\} \tag{3}$$

where $\Lambda_{def,0} = \emptyset$, and the final solution $\Lambda_{\text{def}} = \Lambda_{\text{def},T}$. It is possible to estimate $R_k(\boldsymbol{\lambda})$ empirically using cross-validation, but since this is computationally expensive, we employ *surrogate models* that predict the performance for a given hyperparameter configuration resulting in a fast approximate way to evaluate performances. This approach can be extended to a set of defaults across algorithms.

3 EXPERIMENTAL EVALUATION

We estimate the generalization performance of our approach on future datasets by running a leave-one-dataset-out CV scheme over K datasets, estimating performances for each held-out dataset using outer 10-fold CV and *nested* 5-fold CV for choosing the hyperparameter. We compare to *random search* with several budgets and *Bayesian optimization* with 32 iterations. We use ±137.000 experimental results available on OpenML [4] to evaluate the lists of defaults on three algorithms from `scikit-learn` and 100 datasets from the OpenML100 [1]. We evaluate using Adaboost (5), SVM (6), and random forest (6 hyperparameters) optimizing predictive accuracy. Hyperparameters and their respective ranges are the same as used in [3]. Figure 1 presents the results of the set of defaults obtained by our approach and baselines across 3 algorithms normalized to $[0, 1]$ per algorithm and task and aggregate using the mean. For defaults and random search more iterations strictly improves performance. As expected, random search with only 1 or 2 iterations performs poorly, while Bayesian optimization is often among the best strategies. We further observe that using only a few defaults is already competitive with Bayesian optimization and higher budget random search, often competitive with random search with $4 - 8$ times more budget. We note that using sets of defaults is especially worthwhile when either computation time or expertise on hyperparameter optimization is lacking. Especially in the regime of few function evaluations, sets of defaults seem to work well and are statistically equivalent to state-of-the-art techniques. A potential drawback is that the defaults are optimal with respect to a single metric such as accuracy or AUC, and thus might need to be used separately for different evaluation metrics. Our results can readily be implemented in machine learning software as simple, hard-coded lists of parameters. These will require less knowledge of hyperparameter optimization from the users than current methods, and lead to faster results in many cases.

Acknowledgements. This work has been funded by the German Federal Ministry of Education and Research (BMBF) under Grant No. 01IS18036A. The authors of this work take full responsibility for its content.

REFERENCES

[1] Bernd Bischl, Giuseppe Casalicchio, Matthias Feurer, Frank Hutter, Michel Lang, Rafael G Mantovani, Jan N van Rijn, and Joaquin Vanschoren. 2017. OpenML Benchmarking Suites and the OpenML100. *arXiv preprint arXiv:1708.03731v1* (2017).

[2] Matthias Feurer, Jost Tobias Springenberg, and Frank Hutter. 2015. Initializing Bayesian Hyperparameter Optimization via Meta-learning. In *Proc. AAAI* (Austin, Texas). AAAI Press, 1128–1135.

[3] Jan N. van Rijn and Frank Hutter. 2018. Hyperparameter Importance Across Datasets. In *Proc. of KDD*. ACM, 2367–2376.

[4] J. Vanschoren, J. N. van Rijn, B. Bischl, and L. Torgo. 2014. OpenML: networked science in machine learning. *ACM SIGKDD Explorations Newsletter* 15, 2 (2014), 49–60.

[5] Martin Wistuba, Nicolas Schilling, and Lars Schmidt-Thieme. 2015. Learning hyperparameter optimization initializations. In *Proc. of DSAA*. IEEE, 1–10.

[6] Martin Wistuba, Nicolas Schilling, and Lars Schmidt-Thieme. 2015. Sequential Model-Free Hyperparameter Tuning. In *Proc. of ICDM*. 1033–1038.