Towards Exploiting Early Termination for Multi-Fidelity Hyperparameter Optimization

Helena Graf[1] **Lukas Fehring**[1] **Alexander Tornede**[1] **Tanja Tornede**[1] **Marcel Wever**[1,2] **Marius Lindauer**[1,2]

[1]Institute of Artificial Intelligence, Leibniz University Hannover, Germany
[2]L3S Research Center, Hannover, Germany

Abstract Hyperparameter Optimization (HPO) is crucial to fully leverage the potential of the vast majority of machine learning algorithms. Although Bayesian Optimization (BO) is considered a canonical approach to HPO, determining the runtime limit for BO is challenging, for example, we observed that a runtime on the order of days may be allocated even though retrospectively, the best solution could be found within a few minutes. To alleviate this situation, an early termination strategy based on approximating the statistical error and the remaining optimization potential has been proposed recently. However, many state-of-the-art HPO approaches are based on multi-fidelity optimization, aiming to leverage cheaper evaluation proxies. In this work, we bridge these concepts with the first HPO framework, generalizing automatic termination to multi-fidelity HPO approaches. This allows us to not only prematurely stop the HPO process but also stop the evaluation of uninformative intermediate fidelity stages. In an experimental study, we demonstrate the potential of our framework to substantially reduce the computational footprint by stopping the optimization 40% earlier, at the expense of minor performance degradation.

1 Introduction

Selecting hyperparameter configurations has a large impact on the performance of machine learning algorithms. HPO supports practitioners in finding optimal hyperparameter values for specific tasks (Feurer and Hutter, 2019; Bischl et al., 2023). However, HPO often remains computationally expensive, and termination strategies are usually built upon a pre-defined budget or performance threshold, ignoring whether any solution better than the current incumbent configuration is likely to be found. Makarova et al. (2022) alleviate this situation with their automatic termination strategy for Bayesian optimization (BO) (Mockus, 1989; Frazier, 2018), suggesting termination if the statistical error of the incumbent dominates the estimated regret of early termination. But they only consider black-box HPO and thus are not applicable to state-of-the-art multi-fidelity HPO approaches. In this paper, we propose the first framework to efficiently terminate state-of-the-art multi-fidelity approaches based on Makarova et al. (2022), saving practitioners not only compute resources but also time. In particular, we enable the application of the proposed stopping criterion for BOHB (Falkner et al., 2018), as well as propose its utilization to disregard uninformative fidelities.

2 Related Work

Automated stopping methods are typically either rule-based or dynamic. Rule-based strategies stop when, for example, the incumbent does not improve after a set number of trials (Ribeiro et al., 2011), improvements fall below a threshold (Lorenz et al., 2015), or acquisition function values drop to low (Lorenz et al., 2015; Florea and Andonie, 2022). More recently, Makarova et al. (2022) suggest a dynamic approach that halts optimization when the potential gain no longer outweighs

 © 2025 the authors, released under CC BY 4.0

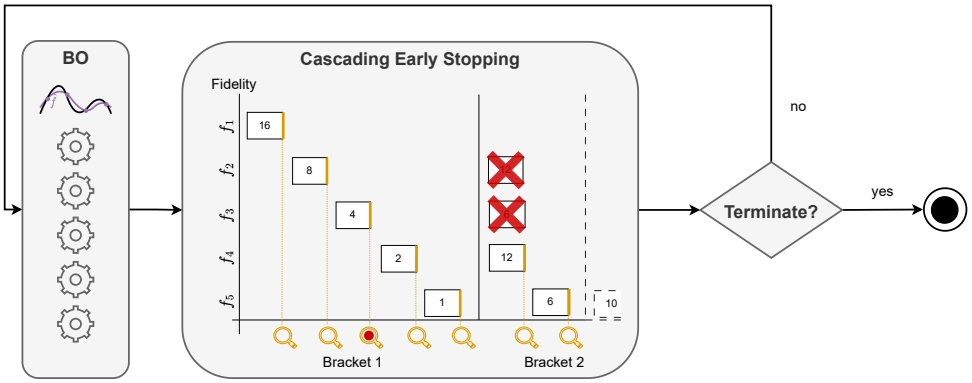

Figure 1: **Visualization of cascading early stopping**. The sampled configurations are evaluated. Whenever a configuration is evaluated, the stopping criterion is checked, indicated by the magnifying glass. When the stopping criterion is triggered, indicated by the red circle, fidelities are removed, and the brackets are adjusted accordingly.

the variability in performance. This automatic stopping significantly reduces runtime with minimal performance loss. However, existing methods generally remain either simplistic or tailored exclusively to black-box, rather than multi-fidelity (grey-box), optimization.

Alternatively, the fidelity schedule itself can be adapted. HyperJump (Mendes et al., 2021) extends Hyperband (HB) (Li et al., 2017) by allowing early jumps to higher fidelities based on expected accuracy loss. DyHPO (Wistuba et al., 2022) uses a joint surrogate model across budgets and dynamically chooses between exploring new configurations or promoting existing ones. Moving beyond static fidelity levels, BOCA (Kandasamy et al., 2017) drops the assumption of fixed fidelity levels and works in a setting where the resource is assumed to be continuous. Recently, Bohdal et al. (2023) propose PASHA, an extension of ASHA (Li et al., 2020), that automatically stops the HPO process if an approximate, soft ranking across two following fidelity levels does not change. Similarly, Brandt et al. (2024) propose a SHA extension that adapts the maximum budget dynamically by adding additional fidelities while preserving theoretical guarantees. However, none of these fidelity schedule adaptation techniques can be used to terminate HPO runs.

3 Early Termination for Multi Fidelity Hyperparameter Optimization

In this paper, we generalize the stopping criterion from Makarova et al. (2022) to multi-fidelity optimization. In particular, we adapt BOHB (Falkner et al., 2018) to support both early stopping and adapting a schedule of fidelity levels $f_1, \ldots, f_k$. After revisiting the original criterion, we present to variants: A basic version that stops the entire optimization process, and a more fine-grained method that also prunes uninformative fidelities to accelerate the optimization process.

Black Box Termination Criterion. Instead of relying on a fixed evaluation budget, Makarova et al. (2022) propose a dynamic stopping criterion that decides when to halt optimization based on the confidence in the current best solution. The idea is to stop if further search is unlikely to lead to substantially better results because the potential improvement is smaller than the uncertainty (or "noise") in the performance estimates. Formally, for a set of previously evaluated configurations $G_t = \{\lambda_1, \ldots, \lambda_{t-1}\}$ and the configuration space Λ, optimization is stopped at configuration λ_t if:

$$\underbrace{\left(\min_{\lambda_i \in G_t} ucb_t(\lambda_i) - \min_{\lambda \in \Lambda} lcb_t(\lambda)\right)}_{\text{Estimated optimization gain } \bar{r}} \leq \sqrt{\mathrm{Var}[\hat{f}(\lambda_t)]} \quad \text{with } \mathrm{Var}[\hat{f}(\lambda_t)] = \left(\frac{1}{k} + \frac{D_i}{D_{-i}}\right) s_{cv}^2(\lambda_t). \tag{1}$$

Here, ucb_t and lcb_t represent upper and lower confidence bounds on performance. If the estimated optimization gain $\bar{r}$ is less than the noise in performance estimation, then continuing the search is unlikely to be worthwhile; thus, the optimization process is terminated. The variance $\mathrm{Var}[\hat{f}(\lambda_t)]$ is typically estimated from cross-validation where $s^2_{cv}(\lambda_t)$ is the sample variance from cross-validation, k is the number of folds, and D_i/D_{-i} reflects the relative size of validation and training data splits.

Naive Multi Fidelity Stopping. A naive translation of Makarova et al.'s stopping criterion to multi-fidelity optimization applies the stopping criterion only to the highest fidelity, because evaluation results on lower fidelities are not necessarily indicative of the final performance. To this end, a separate surrogate model is trained on evaluations at the highest fidelity level exclusively. Every time a hyperparameter configuration is evaluated on the highest fidelity level, the stopping criterion is applied, and if the criterion is met, the optimization process is terminated.

Cascading Early Termination Criterion. Given the promise of fidelity skipping (Mendes et al., 2021) and the observation that early stopping at lower fidelities may reflect noise rather than true performance differences, we propose using the stopping criterion to filter out low-fidelity evaluations (Fig. 1). To disregard fidelities, the stopping criterion is checked after each evaluation, using a surrogate only trained on the respective fidelity. If the criterion is met for fidelity f_i, all configurations are promoted to the next fidelity, and fidelities $f_1, ..., f_i$ are removed from consideration.

4 Preliminary Empirical Evaluation

Experimental Setup. We base our experiments on SMAC's implementation of BOHB (Falkner et al., 2018; Lindauer et al., 2022), equipped with a random forest surrogate model and, if finished within the budget, continuously re-initialized HB brackets, to tune the hyperparameters of both `Random Forest` (Breiman, 2001) and `XGBoost` (Chen and Guestrin, 2016) across a selection of OpenML tabular datasets (Bischl et al., 2023) with dataset subsets as fidelities. We compare our naive early stopping strategy (**Naive Makarova**) and cascading early stopping approach (**Cascading Makarova**) against the vanilla BOHB as a baseline (**HB+BO**). We note that all variants run multi-fidelity HPO and evaluations are based on nested cross-validation, with 10 outer for testing and 3 inner folds for validation. Each optimizer is allotted one hour. For more information, we refer to Appendix A.

Overall Results. Fig. 2 compares validation and test performance vs runtime. Both Naive and Cascading Makarova reduce runtime significantly, with Naive stopping most aggressively. It is important to note that the quality of the generated solutions differs between validation and test set. While it seems that optimizing for a longer time would be beneficial in terms of validation performance, the results on test performances reveal that HB+BO overfits on `Random Forest` and is not adding much on `XGBoost`. Cascading Makarova is in fact the best method on `Random Forest` and on par with full optimization on `XGBoost`. Additionally, uncertainty estimates differ substantially between the validation and test sets, with the validation set showing notably less variation. Overall, early stopping cuts runtime and may curb meta-overfitting. While Cascading Makarova preserves or improves quality, Naive Makarova risks notable degradation.

Anytime Performance Results. As shown in the anytime performance plots in Fig. 3 in the Appendix, Naive Makarova is consistently very aggressive, being over-confident on the highest budget and thus using only a small portion of the allocated budget, potentially contributing to a drop in performance. Compared to the originally considered black-box setting of Makarova et al. (2022), this is a surprising insight on multi-fidelity approaches. We suspect that configurations surviving the successive halving scheme to the highest budget could be fairly narrow in the configuration space and thus the random forest surrogate model in SMAC is overconfident in its uncertainty predictions because of poor extrapolation behavior. It would be open for future work to study this on other models, such as GPs or BNNs.

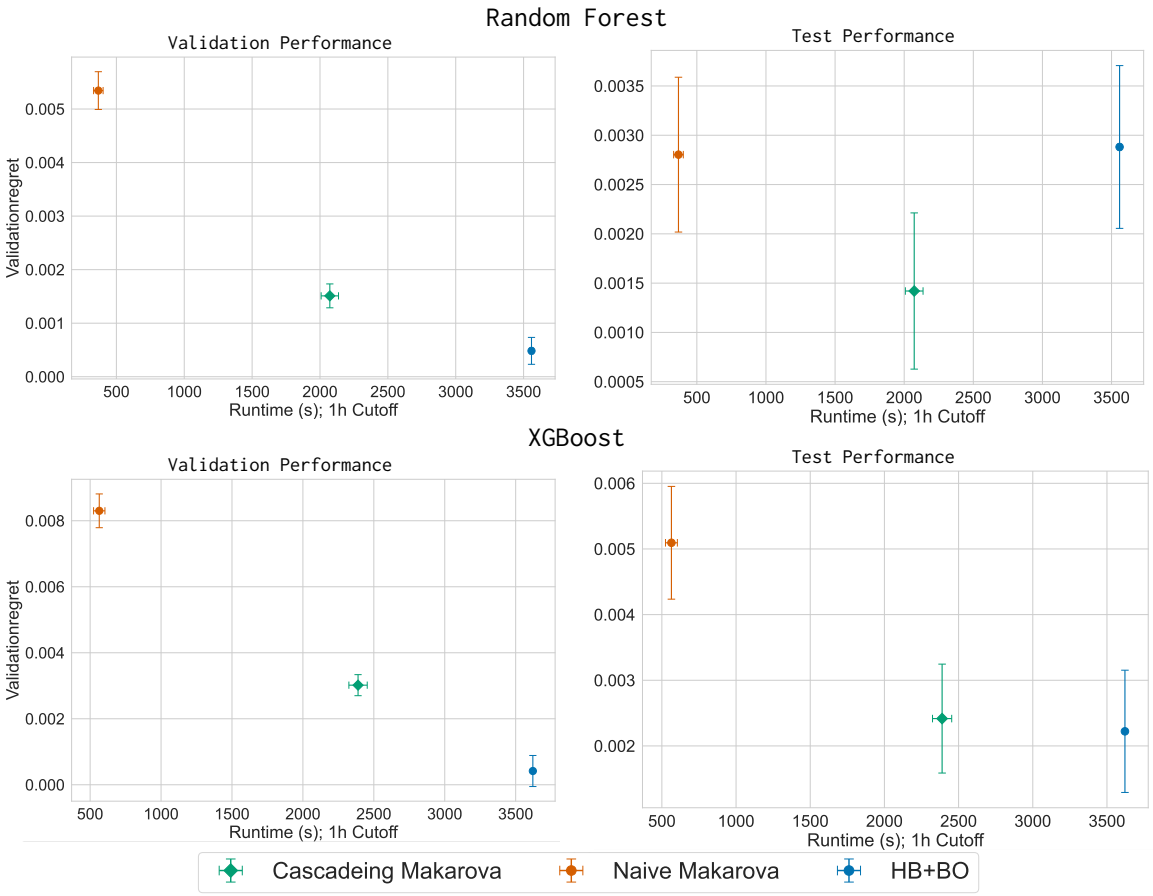

Figure 2: **Performance Comparison of Validation and Test Performance**: Exemplary results comparing our cascading early stopping method (Cascade) to SMAC's BOHB approach (HB+BO) on the validation set (left), and test set (right). Runtime is plotted on the x-axis and regret on the y-axis. Runtime and performance uncertainties are estimated using standard error.

In contrast to Naive Makarova, our cascading method generally terminates later, often resulting in improved final performance. Overall, the final performance of naive stopping is inferior, while Cascade remains largely competitive to HB+BO. Thus, the Cascade represents a reasonable trade-off due to the negligible decrease in quality, but a major decrease in runtime compared to HB+BO.

5 Conclusion

In this paper, we present a first step toward using early termination strategies for multi-fidelity optimization. We generalize the stopping criterion of Makarova et al. (2022) to multi-fidelity settings, allowing not only early stopping of the entire HPO process but also pruning of uninformative fidelity levels. However, the approach remains limited, since the stopping criterion requires inner cross-validation, which is often infeasible in costly learning scenarios. Additionally, our evaluation only considers two learners on tabular datasets. Possible future work consists of further evaluation, as well as exploring other ways of estimating the learner's variance. Example approaches include meta-learning algorithm-specific variances, as well as utilizing the differing variance of different approaches using hyperparameter optimization knowledge bases. Moreover, the development of stopping criteria that do not require cross-validation, for example, using learning curve prediction, may help a translation to state-of-the-art deep-learning models.

Acknowledgements. Helena Graf, Lukas Fehrig, Alexander Tornede, Marcel Wever, and Marius Lindauer acknowledge funding by the European Union (ERC, "ixAutoML", grant no.101041029). Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Research Council Executive Agency. Neither the European Union nor the granting authority can be held responsible for them. Tanja Tornede is funded by the German Federal Ministry of the Environment, Nature Conservation, Nuclear Safety and Consumer Protection (GreenAutoML4FAS project no. 67KI32007A).

The authors gratefully acknowledge the computing time provided to them on the high-performance computers Noctua2 at the NHR Center PC2 under the project hpc-prf-intexml. These are funded by the Federal Ministry of Education and Research and the state governments participating on the basis of the resolutions of the GWK for the national high performance computing at universities (www.nhr-verein.de/unsere-partner).

Lastly, the authors thank the AutoML.org Freiburg group lead by Prof. Dr. Frank Hutter and the COSEAL community for insightful discussions on preliminary ideas and results of this work.

References

Bischl, B., Binder, M., Lang, M., Pielok, T., Richter, J., Coors, S., Thomas, J., Ullmann, T., Becker, M., Boulesteix, A., Deng, D., and Lindauer, M. (2023). Hyperparameter optimization: Foundations, algorithms, best practices, and open challenges. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, page e1484.

Bohdal, O., Balles, L., Ermis, B., Archambeau, C., and Zappella, G. (2023). Pasha: Efficient hpo with progressive resource allocation. In *Proc. of ICLR'23*.

Brandt, J., Wever, M., Bengs, V., and Hüllermeier, E. (2024). Best arm identification with retroactively increased sampling budget for more resource-efficient HPO. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence*.

Breiman, L. (2001). Random forests. *MLJ*, 45:5–32.

Chen, T. and Guestrin, C. (2016). Xgboost: A scalable tree boosting system. In *Proc. of KDD'16*, pages 785–794.

Falkner, S., Klein, A., and Hutter, F. (2018). BOHB: Robust and efficient Hyperparameter Optimization at scale. In *Proc. of ICML'18*, pages 1437–1446.

Feurer, M. and Hutter, F. (2019). Hyperparameter Optimization. In Hutter, F., Kotthoff, L., and Vanschoren, J., editors, *Automated Machine Learning: Methods, Systems, Challenges*, chapter 1, pages 3 – 38. Springer. Available for free at http://automl.org/book.

Florea, A. and Andonie, R. (2022). A dynamic early stopping criterion for random search in SVM hyperparameter optimization. In *Proc. of AIAI'18*, pages 168–180.

Frazier, P. (2018). A tutorial on Bayesian optimization. *arXiv:1807.02811 [stat.ML]*.

Kandasamy, K., Dasarathy, G., Schneider, J., and Póczos, B. (2017). Multi-fidelity Bayesian Optimisation with Continuous Approximations. In Precup, D. and Teh, Y., editors, *Proc. of ICML'17*, pages 1799–1808.

Li, L., Jamieson, K., DeSalvo, G., Rostamizadeh, A., and Talwalkar, A. (2017). Hyperband: Bandit-based configuration evaluation for Hyperparameter Optimization. In *Proc. of ICLR'17*.

Li, L., Jamieson, K., Rostamizadeh, A., Gonina, E., Ben-tzur, J., Hardt, M., Recht, B., and Talwalkar, A. (2020). A system for massively parallel hyperparameter tuning. In *Proc. of MLSys'20*.

Lindauer, M., Eggensperger, K., Feurer, M., Biedenkapp, A., Deng, D., Benjamins, C., Ruhkopf, T., Sass, R., and Hutter, F. (2022). SMAC3: A versatile bayesian optimization package for Hyperparameter Optimization. *JMLR*, 23(54):1–9.

Lorenz, R., Monti, R., Violante, I., Faisal, A., Anagnostopoulos, C., Leech, R., and Montana, G. (2015). Stopping criteria for boosting automatic experimental design using real-time fMRI with bayesian optimization. *arXiv:1511.07827 [q-bio.NC]*.

Makarova, A., Shen, H., Perrone, V., Klein, A., Faddoul, J., Krause, A., Seeger, M., and Archambeau, C. (2022). Automatic termination for hyperparameter optimization. In *Proc. of AutoML Conf'22*, pages 7/1–21. PMLR.

Mendes, P., Casimiro, M., Romano, P., and Garlan, D. (2021). Hyperjump: Accelerating hyperband via risk modelling. *arXiv:2108.02479 [cs.LG]*.

Mockus, J. (1989). *Bayesian Approach to Global Optimization. Theory and Applications*. Kluwer Academic Publishers.

Ribeiro, C., Rosseti, I., and Souza, R. (2011). Effective probabilistic stopping rules for randomized metaheuristics: GRASP implementations. In *Proc. of LION'11*, pages 146–160.

Tornede, T., Tornede, A., Fehring, L., Gehring, L., Graf, H., Hanselle, J., Mohr, F., and Wever, M. (2023). PyExperimenter: Easily distribute experiments and track results. *JOSS*, 8(84):5149.

Wistuba, M., Kadra, A., and Grabocka, J. (2022). Supervising the multi-fidelity race of hyperparameter configurations. In *Proc. of NeurIPS'22*.

Submission Checklist

1. For all authors…

   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]

   (b) Did you describe the limitations of your work? [Yes]

   (c) Did you discuss any potential negative societal impacts of your work? [N/A]

   (d) Did you read the ethics review guidelines and ensure that your paper conforms to them? (see `https://2022.automl.cc/ethics-accessibility/`) [Yes]

2. If you ran experiments…

   (a) Did you use the same evaluation protocol for all methods being compared (e.g., same benchmarks, data (sub)sets, available resources, etc.)? [Yes]

   (b) Did you specify all the necessary details of your evaluation (e.g., data splits, pre-processing, search spaces, hyperparameter tuning details and results, etc.)? [Yes]

   (c) Did you repeat your experiments (e.g., across multiple random seeds or splits) to account for the impact of randomness in your methods or data? [Yes]

   (d) Did you report the uncertainty of your results (e.g., the standard error across random seeds or splits)? [Yes]

   (e) Did you report the statistical significance of your results? [No]

   (f) Did you use enough repetitions, datasets, and/or benchmarks to support your claims? [Yes]

   (g) Did you compare performance over time and describe how you selected the maximum runtime? [Yes]

   (h) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes]

   (i) Did you run ablation studies to assess the impact of different components of your approach? [No]

3. With respect to the code used to obtain your results…

   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results, including all dependencies (e.g., `requirements.txt` with explicit versions), random seeds, an instructive `README` with installation instructions, and execution commands (either in the supplemental material or as a URL)? [Yes]

   (b) Did you include a minimal example to replicate results on a small subset of the experiments or on toy data? [No]

   (c) Did you ensure sufficient code quality and documentation so that someone else can execute and understand your code? [Yes]

   (d) Did you include the raw results of running your experiments with the given code, data, and instructions? [No]

   (e) Did you include the code, additional data, and instructions needed to generate the figures and tables in your paper based on the raw results? [Yes]

4. If you used existing assets (e.g., code, data, models)…

(a) Did you cite the creators of used assets? [Yes]

(b) Did you discuss whether and how consent was obtained from people whose data you're using/curating if the license requires it? [N/A]

(c) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]

5. If you created/released new assets (e.g., code, data, models)…

(a) Did you mention the license of the new assets (e.g., as part of your code submission)? [Yes]

(b) Did you include the new assets either in the supplemental material or as a URL (to, e.g., GitHub or Hugging Face)? [Yes]

6. If you used crowdsourcing or conducted research with human subjects…

(a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]

(b) Did you describe any potential participant risks, with links to institutional review board (IRB) approvals, if applicable? [N/A]

(c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

7. If you included theoretical results…

(a) Did you state the full set of assumptions of all theoretical results? [N/A]

(b) Did you include complete proofs of all theoretical results? [N/A]

A Evaluation Setup Details

A.1 Technical Setup

For our experiments, we utilize an HPC cluster. Each compute node is equipped with two Intel Xeon Gold Skylake containing 40 cores, and a 192GiB of RAM. Each optimization was run in an independent compute job equipped with 4 CPUs and 4 GB RAM. To log experiment results, we utilize the PyExperimenter (Tornede et al., 2023) library. To consider test-performance in our final performance plots **??** and Appendix B, we conducted an additional evaluation round, where runtime was impacted by cluster problems - we only consider datasets where all 10 test-folds were evaluated successfully. Our anytime performance plots in Fig. 3 portray the validation performance evaluated preceding these cluster issues.

B Further Experimental Results

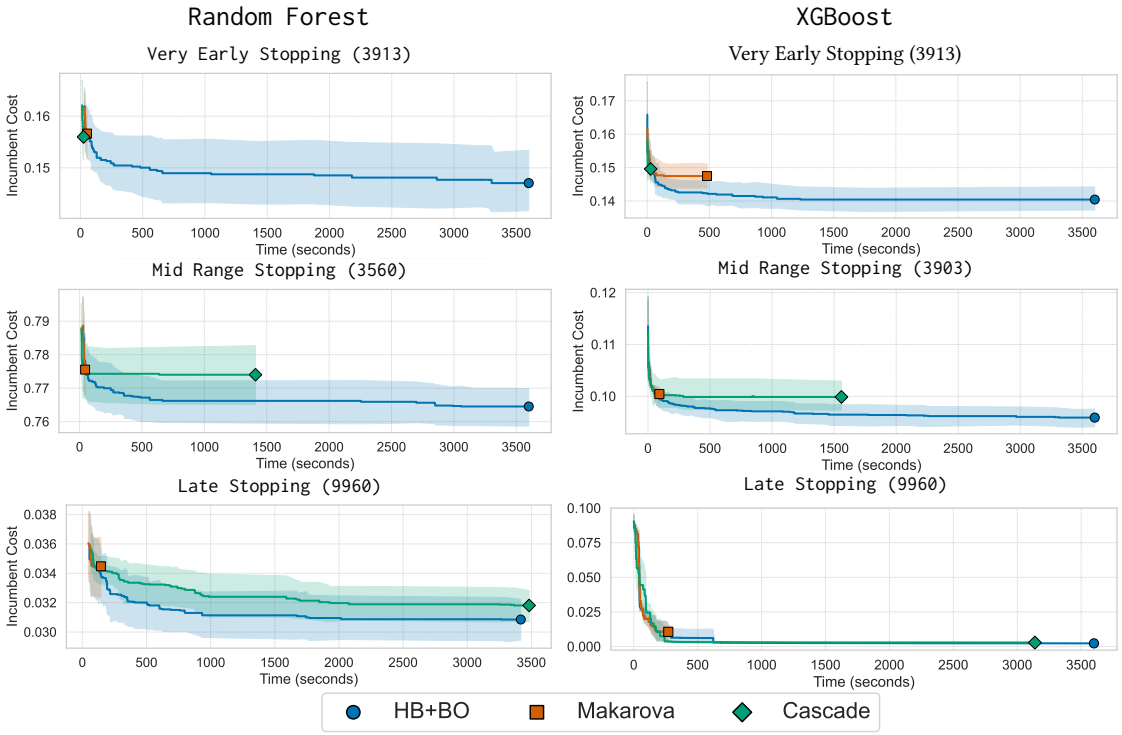

Figure 3: **Anytime Validation Performance Plots**: Exemplary results on different datasets (ID in brackets) comparing our naive early stopping approach (Makarova), cascading early stopping (Cascade), and SMAC's BOHB method (HB+BO) evaluated on optimizing the `Random Forest` and `XGBoost` hyperparameters. Runtime in seconds (capped at the mean endpoint) is plotted vs regret. The plots portray the mean performance, error bars show the standard deviation.