Towards Meta-learned Algorithm Selection using Implicit Fidelity Information

Aditya Mohan [*]   MOHAN@TNT.UNI-HANNOVER.DE
Tim Ruhkopf [*]   RUHKOPF@TNT.UNI-HANNOVER.DE
Marius Lindauer   LINDAUER@TNT.UNI-HANNOVER.DE
Leibniz Universität Hannover

Abstract

Automatically selecting the best performing algorithm for a given dataset or ranking multiple of them by their expected performance supports users in developing new machine learning applications. Most approaches for this problem rely on dataset meta-features and landmarking performances to capture the salient topology of the datasets and those topologies that the algorithms attend to. Landmarking usually exploits cheap algorithms not necessarily in the pool of candidate algorithms to get inexpensive approximations of the topology. While somewhat indicative, hand-crafted dataset meta-features and landmarks are likely insufficient descriptors, strongly depending on the alignment of the geometries the landmarks and candidates search for. We propose `IMFAS`, a method to exploit multi-fidelity landmarking information directly from the candidate algorithms in the form of non-parametrically non-myopic meta-learned learning curves via LSTMs in a few-shot setting during testing. Using this mechanism, `IMFAS` jointly learns a dataset's topology and algorithms' inductive biases, without the need to expensively train them to convergence. Our approach produces informative landmarks, easily enriched by arbitrary meta-features at a low computational cost, capable of producing the desired ranking using cheaper fidelities. We additionally show that `IMFAS` is able to beat Successive Halving with at most 50% of the fidelity sequence during test time.

1. Introduction

Selecting well-performing algorithms for a given task in an automatic and data-driven manner, i.e. the Algorithm Selection problem (Rice, 1976), is considered one of the major challenges in AutoML, potentially rendering it unnecessary to manually test suitable algorithms to figure out either a single best algorithm (Leite and Brazdil, 2010) or recommend multiple algorithms ranked by their performances on a new dataset (van Rijn et al., 2015; Mohr and van Rijn, 2021). The standard approach is to characterize the datasets and algorithms using meta-features (e.g., size of the dataset) and learn patterns from them, indicating high and low performing co-occurrences (e.g., SVMs for small datasets and DNNs for large datasets). During meta-test time, the same meta-features are recomputed and thus, the learned model can be used to predict which algorithms would be suited for a dataset. However, it is hard to construct pre-computed and handcrafted meta-features that adequately characterize the likely performances of algorithms (Leite and Brazdil, 2004; van Rijn et al., 2015). Landmarking meta-features, while usually cheap to evaluate and helpful

---

*. These authors equally contributed to the work

in elucidating some of the dataset's topology, fall short in sufficiently boosting the quality of choosing the most applicable candidate algorithm for datasets. This likely originates from the dependence on the alignment of the geometries for which the landmark and the candidate algorithms search.

Complementary to Algorithm Selection, methods in the domain of multi-fidelity optimization (Kandasamy et al., 2016; Li et al., 2017; Falkner et al., 2018) use different types of fidelities such as dataset subset-sizes or number of epochs, summarized in the unrolling of algorithm's learning curves, in order to cheaply evaluate candidate algorithms at various granularities. These fidelities are used to cheaply probe algorithms and draw conclusions about their final performances. There are, however, two major hurdles when utilizing fidelity information: (i) Acting on a fixed fidelity level rather than a learning curve requires that the relative performances need to be sufficiently indicative on that level. This pronounces the question of the reliability of the fidelity signal. (ii) A learning mechanism acting on multiple fidelities needs to be non-myopic to avoid fallacies such as a bias towards early performing algorithms or it might not recover.[1] While meta-learning (Vanschoren, 2018) can alleviate the myopia, it still requires the formulation of a meta-model and a mechanism to continually update it based on incoming evidence. Approaches that are less explicit in the use of their meta-prior form strong assumptions regarding the functional form of the learning curves, such as concavity or that the learning curve unfolds very similar in shape to previously observed ones (Leite and Brazdil, 2004). These assumptions are difficult to justify and require theoretical as well as experimental rigor (Mohr and van Rijn, 2022).

Combining multi-fidelity optimization and landmarking into a meta-learning framework allows probing the inductive biases of the candidates directly, thus, foregoing the need to rely on unreliable non-candidate algorithms. We propose a meta-learning method, called `IMFAS` Implicit Multi-Fidelity Algorithm Selection that exploits existing correlations between the observed learning curves of these new landmarks non-parametrically. `IMFAS` implicitly learns how the candidates' inductive biases relate to one another and how they unfold over the fidelities, alleviating myopia with an informed point-prior. Moreover, we contextualize the trajectory of their learning curves with the observed dataset's meta-features, implicitly conditioning on them.

In summary our contributions are the following: (i) We pose the Algorithm Selection problem as a multi-fidelity meta-learning few-shot ranking problem, which allows us to cheaply gather directly applicable and useful information about the new datasets' topology, bypassing former vicarious landmarking features. (ii) We use LSTMs (Hochreiter and Schmidhuber, 1997) and a differentiable ranking mechanism (Blondel et al., 2020), to characterize a dataset's topology jointly by pre-computed meta-features, the candidate algorithms' low fidelity approximations, and the meta experience of their relative progressions – thus jointly and non-parametrically learning the learning curves of algorithms on unseen datasets.

2. Background and Related Work

Classical Algorithm Selection approaches learn mappings between datasets and algorithms w.r.t. performance based on pre-computed meta-features, but these fail to sufficiently char-

---

1. As in the case of e.g. Successive Halving

acterize the datasets and algorithms. Since landmarking approaches have shown considerable performance improvement (Pfahringer et al., 2000), using a combination of pre-computed meta-features and landmarks is conceptually appealing. Landmarks are not always superior since it is non-trivial to relate the landmarks' inductive biases to those of the candidates (Pfahringer et al., 2000). Our approach sidesteps this by landmarking datasets directly based on the candidates algorithms' inductive biases.

The idea of exploiting partial learning curves has previously been explored in other settings. SAM (Leite and Brazdil, 2004) non-parametrically matched partial learning curves using k-NN to the closest in terms of shape from the observed meta-learning curves, while van Rijn et al. (2015) reduced the cost of cross-validation by exploiting the similarity of the partially-observed rankings with those of the meta-datasets and using the most similar learning curves as surrogates. Mohr and van Rijn (2021) extended this by terminating less promising candidates early on due to the abundance/redundancy of data by exploiting semi-parametric learning curves, under the assumption of concavity. They remain unbiased by not modeling meta-knowledge. Leite and Brazdil (2010) used a graph-based approach on a handcrafted performance distance metric between datasets and fidelities to account for ordinal rankings that originate from their repeated pairwise comparisons of algorithms across fidelities. In essence, they pursue a cheaper and meta-informed alternative to cross-validation that evaluates the collection of fidelities using SAM, and then actively schedules them in a cost-aware manner.

Two approaches orthogonal to ours, although similar in terms of setup, are Meta-REVEAL (Nguyen et al., 2021) and MetaBu (Rakotoarison et al., 2021). Meta-REVEAL focuses on scheduling the algorithms and fidelities through a Reinforcement Learning perspective by modeling it as a REVEAL game. The agent acts on a discrete action space that does not account for the correlations between learning curves. Consequently, using the estimates produced by `IMFAS` can provide a pre-processed action space for their problem. MetaBu extends the classical Algorithm Selection idea by relating dataset meta-features and algorithms' hyperparameters using a learned optimal transport map. They, however, ignore that the dataset meta-features cannot sufficiently characterize the dataset, thus, subsequently falling victim to the same fallacies as the methods mentioned before.

3. Method

We follow the Algorithm Selection set up with a portfolio $\mathcal{A}$ of algorithms from which one or a group of algorithms is to be recommended for a given dataset. We aim to learn a meta-model learning $\mathcal{D} \mapsto r(f_n)$ that enriches the dataset meta-features $\mathcal{D}$ implicitly with landmarks of the collection of partially-observed algorithm performances $\{f_k\}_{k=1,\ldots,g}$ where $g \ll n$ and can predict the rank of the final performances f_n using the function r.

`IMFAS` landmarks datasets using the collective of candidate algorithm's inductive biases. This is inspired by work in multi-dependent-label classification for Algorithm Selection (Wang et al., 2014; Khan et al., 2020), which explicitly models the co-occurrence of algorithms (labels), conditioned on the dataset's meta-features, hinting towards patterns in the algorithms' inductive biases. `IMFAS` tries to exploit this correlation as it unfolds along the fidelities. We combine information from the pre-computed meta-features and a sequence of fidelity information using LSTMs (Hochreiter and Schmidhuber, 1997).
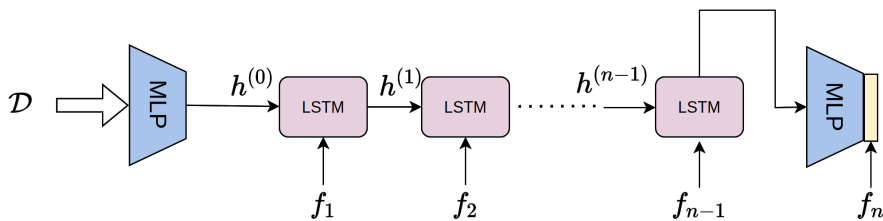
Figure 1: An overview of the method, initializing the LSTM using the encoded meta-features and contextualizing it using the algorithm's performance vectors at the respective fidelities.

Since LSTMs fundamentally combine memory-based representations with contextual information fed at every step of the sequence, we encode the dataset meta-features to initialize the hidden state. This is done by feeding the dataset meta-features $\mathcal{D}$ to an encoder Multi-Layer Perceptron (MLP). Given n fidelities going from the cheapest to full training of a model, we forward propagate the LSTM $n-1$ times. The output of the LSTM is again encoded to a vector of $|\mathcal{A}|$ dimensions using an MLP. The final output values are differentiably sorted and ranked (Blondel et al., 2020). Finally, the ranks are compared against a ranking of ground-truth performances in the final fidelity as depicted in Figure 1.

To create a loss for optimization, we calculate the spearman correlation (Zar, 2005) between the predicted ranking r_p and the ground-truth ranking r_g and then complement this value to get the loss $\mathcal{L} = 1 - \frac{Cov(r_p, r_g)}{\sigma_{r_p}\sigma_{r_g}}$, where σ_{r_p} and σ_{r_g} are the respective standard deviations of the rank variables. Minimizing this quantity amounts to maximizing the positive correlation between the predicted ranking and the ground-truth. The gradients of the differentiable ranking function are then propagated backward. During meta-test time, we repeat the same process but with the crucial difference that instead of unrolling the LSTM for all fidelities, we only unroll it up to a fraction of the fidelity level. This helps us examine whether the trained meta-model can approximate the full-fidelity rank with partial fidelity information, indicated by a low value of the test loss. Please refer to Appendix A for implementation details

Intuitively, the hidden dimensions d of LSTM are constant, the hidden representation at each step can be thought of as a point that moves in the latent space $h^{(i)} \in \mathbb{R}^d$. Thus, given an encoding of the dataset meta-features in that latent space as initialization, unrolling the LSTM each time based on the current f_i is akin to tracking a trajectory of this point. During each unroll step, the LSTM acts like a transition operator on that space, progressively conditioning on the obtained fidelity information. Applying the final decoder MLP on any of the $h^{(i)}$ amounts to the LSTM's current expectation $\hat{f}_n$ conditioned on $\{f_j\}_{j=1,...,i}$. This operator's learned weights encapsulate all the meta-knowledge about how the algorithms' trajectories correlate conditioned on the dataset's meta-features. It acts as meta-prior, similar to Ravi and Larochelle (2016), during test time which combats myopia in an exclusively data-driven and non-parametric manner. We merely need to apply this operator to condition our expectation towards $\hat{f}_n$ on the partially available fidelity information.

4

4. Experiments

Benchmarks and Baseline We evaluate our approach[2] on the YAHPO-Gym surrogate benchmark (Pfisterer et al., 2022), specifically the rbv2 algorithm instances, which is a collection of separate Hyperparameter Optimization surrogate benchmarks, each for a specific algorithm. All of the rbv2 instances consider dataset subset size as fidelity type. Crucially, we model different algorithms as separate instantiations of the hyperparameters of an algorithm, which has been shown to sufficiently alter inductive bias to be relevant to our problem (Li et al., 2021). We additionally test `IMFAS` on LCBench (Zimmer et al., 2021), an epoch-based learning curve benchmark that provides extensive meta-training data for different MLP architectures and hyperparameters. Finally, to validate whether the learned rankings are sensible, we employ a meta-ignorant, non-parametric, and myopic successive halving (Li et al., 2017) as a baseline. It is altered to produce an absolute ranking in the following fashion: The level of termination (fidelity) generates a tied ordinal ranking, where each level's ties are broken by considering the observed performances at that level. As a result, we obtain an absolute ranking, respecting the amount of available information on the levels. This predicted ranking is compared against the ground-truth.

Setup We evaluated the agent on the aforementioned benchmarks by training it for 300 epochs on the full fidelity information. The evaluations were performed by holding out 20% of the data for testing. Additionally, the Successive Halving baseline has been computed on the test data that was held out. We report the results as the average and standard deviation across 5 seeds.

Results We summarize our results in Table 1. The metric used for the rbv2 suite is the F1 score and the one for LCBench is the validation accuracy. Notice that the Spearman correlation generally increases with an increasing amount of fidelity information. However, the differences between the final correlations produced at 50% fidelities and at 100% fidelities during test time are generally lower, indicating that the transfer helps the model in approximating a correlation close to the ground-truth using cheaper approximations. `IMFAS` is able to beat Successive Halving in most cases (highlighted in bold) at fidelities below 100%.

Table 1: Summary of resulting correlations during test time given the amount of available fidelity information in (%) of the fidelity sequence against Successive Halving. In bold are the first fidelities at which the baseline is surpassed.

Dataset	10%	20%	50%	100%	SH
rbv2_super	0.606 ± 0.021	$\mathbf{0.698 \pm 0.070}$	0.759 ± 0.021	0.716 ± 0.116	0.694 ± 0.073
rbv2_svm	0.431 ± 0.013	0.504 ± 0.182	$\mathbf{0.660 \pm 0.104}$	0.853 ± 0.135	0.641 ± 0.079
rbv2_xgboost	0.473 ± 0.054	0.596 ± 0.034	$\mathbf{0.769 \pm 0.010}$	0.882 ± 0.034	0.668 ± 0.096
rbv2_ranger	0.454 ± 0.028	0.454 ± 0.144	$\mathbf{0.552 \pm 0.092}$	0.602 ± 0.147	0.500 ± 0.084
rbv2_rpart	0.391 ± 0.342	$\mathbf{0.554 \pm 0.237}$	0.749 ± 0.086	0.802 ± 0.028	0.351 ± 0.093
rbv2_aknn	$\mathbf{0.539 \pm 0.209}$	0.633 ± 0.059	0.704 ± 0.057	0.719 ± 0.092	0.497 ± 0.068
lcbench	0.616 ± 0.045	0.710 ± 0.041	$\mathbf{0.759 \pm 0.031}$	0.763 ± 0.027	0.641 ± 0.079

2. The implementation of `IMFAS` can be found on GitHub `https://github.com/automl/IMFAS`

Figure 2 shows an exemplary plot for rbv2_xgboost dataset and we see that the model is able to approximate the performance of the 100% fidelities by only unrolling for 50%, as compared to when it is fed only 10% or 20% of the sequence during test time. This shows positive transfer, since the higher fidelities can be approximated from the cheaper.
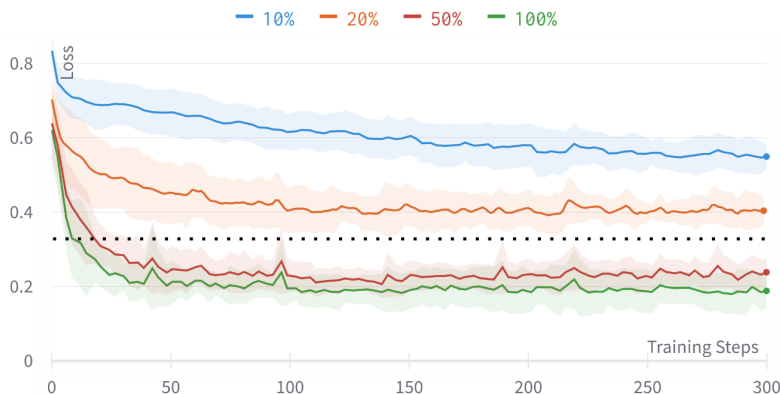

Figure 2: The loss progression in terms of a learning curve for `rbv2_xgboost`. By only using 50% of the fidelity sequence, our model approximates the the final ranking possible with the full sequence. The dotted line shows the value produced by Successive Halving for this dataset

5. Conclusion and Future work

We present a unifying outlook towards Algorithm Selection through a multi-fidelity lens and present a method to capture the fidelity information implicitly and non-parametrically using LSTMs with a differentiable ranking loss. We test this approach on diverse benchmarks and show that lower fidelities captured in this manner are often sufficient to implicitly produce a ranking comparable to the ground-truth. We finally present an analysis of how the sufficient fidelities depend on the benchmark being tested. Our approach, however, has certain limitations which we plan to address in future work: (i) The current methodology does not completely support hyperparameter settings, albeit their features, which vicariously describe the inductive biases, are captured implicitly by the learning curves. Exploiting this information allows extending `IMFAS` to Hyperparameter Optimization across multiple algorithms, thus addressing the full CASH problem (Thornton et al., 2013). (ii) The requirement of a full grid of algorithms and datasets in the meta-dataset can be reduced using an interpolation strategy based on partially overlapping dataset-algorithm combinations, thus, improving scalability. (iii) Estimates regarding uncertainty resulting from the fidelity spacing and varying degree of contained information can be propagated to potentially better inform the fidelity roll-out.

References

M. Blondel, O. Teboul, Q. Berthet, and J. Djolonga. Fast differentiable sorting and ranking. In *International Conference on Machine Learning*, pages 950–959. PMLR, 2020.

S. Falkner, A. Klein, and F. Hutter. BOHB: Robust and efficient hyperparameter optimization at scale. In J. Dy and A. Krause, editors, *Proceedings of the 35th International Conference on Machine Learning (ICML'18)*, volume 80, pages 1437–1446. Proceedings of Machine Learning Research, 2018.

S. Hochreiter and J. Schmidhuber. Long Short-Term Memory. *Neural Computation*, 9(8): 1735–1780, 1997. Based on TR FKI-207-95, TUM (1995).

K. Kandasamy, G. Dasarathy, J. Oliva, J. Schneider, and B. Póczos. Gaussian Process Bandit Optimisation with Multi-fidelity Evaluations. In D. Lee, M. Sugiyama, U. von Luxburg, I. Guyon, and R. Garnett, editors, *Proceedings of the 29th International Conference on Advances in Neural Information Processing Systems (NeurIPS'16)*, pages 992–1000, 2016.

I. Khan, X. Zhang, M. Rehman, and R. Ali. A literature survey and empirical study of meta-learning for classifier selection. *IEEE Access*, 8:10262–10281, 2020.

R. Leite and P. Brazdil. Predicting relative performance of classifiers from samples. In R. Greiner, editor, *Proceedings of the 21st International Conference on Machine Learning (ICML'04)*. Omnipress, 2004.

R. Leite and P. Brazdil. Active testing strategy to predict the best classification algorithm via sampling and metalearning. In H. Coelho, R. Studer, and M. Wooldridge, editors, *Proceedings of the Nineteenth European Conference on Artificial Intelligence (ECAI'10)*, pages 309–314. IOS Press, 2010.

L. Li, K. Jamieson, G. DeSalvo, A. Rostamizadeh, and A. Talwalkar. Hyperband: Bandit-based configuration evaluation for hyperparameter optimization. In *Proceedings of the International Conference on Learning Representations (ICLR'17)*, 2017. Published online: `iclr.cc`.

M. Li, E. Grant, and T. Griffiths. Meta-learning inductive biases of learning systems with gaussian processes. In M. Ranzato, A. Beygelzimer, K. Nguyen, P. Liang, J. Vaughan, and Y. Dauphin, editors, *Fifth Workshop on Meta-Learning at the Conference on Neural Information Processing Systems*, 2021.

F. Mohr and J. van Rijn. Fast and informative model selection using learning curve cross-validation. *arXiv preprint arXiv:2111.13914*, 2021.

F. Mohr and J. van Rijn. Learning curves for decision making in supervised machine learning–a survey. *arXiv preprint arXiv:2201.12150*, 2022.

M. Nguyen, N. Grinsztajn, I. Guyon, and L. Sun-Hosoya. MetaREVEAL: RL-based Meta-learning from Learning Curves. In G. Krempl, V. Lemaire, D. Kottke, B. Hammer, and A. Holzinger, editors, *Workshop on Interactive Adaptive Learning*, 2021.

A. Paszke, S. Gross, F. Massa, A. Lerer, et al. PyTorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Proceedings of the 32nd International Conference on Advances in Neural Information Processing Systems (NeurIPS'19)*, pages 8024–8035, 2019.

B. Pfahringer, H. Bensusan, and C. Giraud-Carrier. Meta-learning by landmarking various learning algorithms. In P. Langley, editor, *Proceedings of the Seventeenth International Conference on Machine Learning (ICML'00)*, pages 743–750. kaufmann, 2000.

F. Pfisterer, L. Schneider, J. Moosbauer, M. Binder, and B. Bischl. Yahpo gym-an efficient multi-objective multi-fidelity benchmark for hyperparameter optimization. In *First Conference on Automated Machine Learning (Main Track)*, 2022.

H. Rakotoarison, L. Milijaona, A. RASOANAIVO, M. Sebag, and M. Schoenauer. Learning meta-features for autoML. In *Proceedings of the International Conference on Learning Representations (ICLR'22)*, 2021. Published online: `iclr.cc`.

S. Ravi and H. Larochelle. Optimization as a model for few-shot learning. In *Proceedings of the International Conference on Learning Representations (ICLR'16)*, 2016. Published online: `iclr.cc`.

J. Rice. The algorithm selection problem. *Advances in Computers*, 15:65–118, 1976.

C. Thornton, F. Hutter, H. Hoos, and K. Leyton-Brown. Auto-WEKA: combined selection and hyperparameter optimization of classification algorithms. In I. Dhillon, Y. Koren, R. Ghani, T. Senator, P. Bradley, R. Parekh, J. He, R. Grossman, and R. Uthurusamy, editors, *The 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'13)*, pages 847–855. acm, 2013.

J. van Rijn, S. Abdulrahman, P. Brazdil, and J. Vanschoren. Fast algorithm selection using learning curves. In E. Fromont, T. De Bie, and M. van Leeuwen, editors, *Proceedings of the 14th International Symposium on Intelligent Data Analysis*. Springer, 2015.

J. Vanschoren. Meta-learning: A survey. *CoRR*, abs/1810.03548, 2018.

G. Wang, Q. Song, X. Zhang, and K. Zhang. A generic multilabel learning-based classification algorithm recommendation method. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 9(1):1–30, 2014.

J. Zar. Spearman rank correlation. *Encyclopedia of biostatistics*, 7, 2005.

L. Zimmer, M. Lindauer, and F. Hutter. Auto-PyTorch Tabular: Multi-Fidelity MetaLearning for Efficient and Robust AutoDL. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 3079 – 3090, 2021.

Appendix A. Implementation Details

LSTM Details The general hyperparameter configuration of the LSTM has been shown in Table 2. The LSTM was implemented using the pytorch(Paszke et al., 2019) library, and the differentiable ranking losses have been implemented using torchsort (Blondel et al., 2020). The training was performed on a 16-core CPU. The parameters that changed within tests were the `batch size` and `learning rate`, which have been reported in Table 3. The hyperparameters were tuned by elucidating the configuration space using Hyperband (Li et al., 2017) and then checking the ones that work from the reduced set of configurations.

Table 2: Table showing the general hyperparameter configuration of lstm

Hyperparameter	Value
Number of Layers	2
Hidden dimensions of Encoder	$(300, 200)$
Hidden dimensions of readout	$(200, |\mathcal{A}|)$

Table 3: Table showing the general hyperparameter configuration of LSTM

Dataset	Batch Size	Learning Rate
rbv2_super	10	0.001
rbv2_svm	10	0.0005
rbv2_xgboost	10	0.0005
rbv2_ranger	10	0.001
rbv2_rpart	10	0.001
rbv2_aknn	10	0.0005
lcbench	8	0.001

YAHPO Benchmark Yahpo consists of multiple Hyperparameter Optimization benchmarks of which we chose those named in the form rbv2_<algorithm> including <algorithm>: svm, xgboost, aknn, ranger, rpart, super. Noteworthy, 'super' is a full ML pipeline, joining the search spaces of rpart, glmnet, ranger, xgboost hierarchically, thereby encapsulating a CASH problem. Since Yahpo is a surrogate benchmark, we sample 50 hyperparameter configurations for each of the rbv2 instances respectively using a Latin Hypercube Design. These configurations are held constant for each instance across all its datasets. With the exception of 'super', sampling a hyperparameter space in this way likely will introduce correlating learning curves. The 'super' instance combines the latter with heterogeneous algorithms, likely differing in their inductive biases.

LCBench Benchmark Using the raw LCBench, which reports its real evaluations, we sample the table of hyperparameters in a manner, that produces a set of diverse and known to be performant configurations. We do this by considering each dataset's top-k performing algorithms and building the union over those resulting configurations. Using $k = 3$, we arrive at 58 candidate algorithms.