Towards Dynamic Priors in Bayesian Optimization for Hyperparameter Optimization

Lukas Fehring[1] **Marcel Wever**[1] **Maximilian Spliethöver**[1] **Henning Wachsmuth**[1] **Marius Lindauer**[1]

[1]Institute of AI, Leibniz University Hannover, Hannover, Germany

Abstract Hyperparameter optimization (HPO), as part of automated machine learning, successfully supports users in designing models well-suited for a given dataset. However, HPO still sometimes lacks acceptance due to its black-box nature and a lack of options to intervene in the process. Although first approaches have been proposed to initialize HPO methods with expert knowledge, they do not allow for repeated expert intervention *during* the optimization process. In this paper, we introduce a novel method that enables repeated online interaction with an HPO approach via user input, specifying expert knowledge and user preferences in the form of prior distributions at runtime of the HPO process. To this end, we generalize an existing method, πBO, by stacking priors on top of each other whenever provided by the user. Including these user priors in the acquisition function of Bayesian optimization (BO), we are able to model dynamic user inputs into BO-based HPO and guide the optimization accordingly. To avoid negative effects due to misleading priors, we additionally propose a prior-disregarding mechanism based on the surrogate's understanding of the optimization problem. In our experimental evaluation, we demonstrate that our method can effectively incorporate multiple priors. It leverages helpful priors, whereas misleading priors can be rejected or overcome without significantly deteriorating anytime performance.

1 Introduction

Hyperparameter optimization (HPO) is concerned with automatically optimizing the hyperparameters of machine learning (ML) algorithms tailored to a given task (Hutter et al., 2019). HPO is effective on classical ML (Eggensperger et al., 2021; Bansal et al., 2022; Pfisterer et al., 2022) as well as on deep neural networks and transformers for computer-vision and NLP (Müller et al., 2023; Wang et al., 2024; Rakotoarison et al., 2024; Pineda Arango et al., 2024). Despite this success, potential (expert) users often disregard HPO methods in favor of making manual decisions (Van der Blom et al., 2021) since existing HPO methods limit opportunities for monitoring and control. At the same time, explainability methods (Wang et al., 2019; Sass et al., 2022; Zöller et al., 2023) empower users to gain insights into the optimization process. Still, online steering of the optimization process remains largely unexplored. Motivated by the need for a rapid prototyping workflow and the call for user-centric AutoML (Lindauer et al., 2024), we propose enabling users to inject knowledge on potentially well-suited hyperparameter configurations in the form of user priors at any point during the process, thereby enabling users to steer the optimization process. Our proposed approach adapts BO by incorporating optional user priors during the optimization process, see Fig. 1.

2 Related Work

Transfer learning from previous BO experiments leverages knowledge from previous optimization tasks (Swersky et al., 2013; Wistuba et al., 2015; van Rijn and Hutter, 2018; Feurer et al., 2022), such as hyperparameter optimization across different datasets or at various development stages (Stoll et al., 2020). Exemplary insights include previously high-performing candidates (Brochu et al., 2010; Feurer et al., 2015), or search space constraints to promising regions (Perrone et al., 2019).

 © 2025 the authors, released under CC BY 4.0

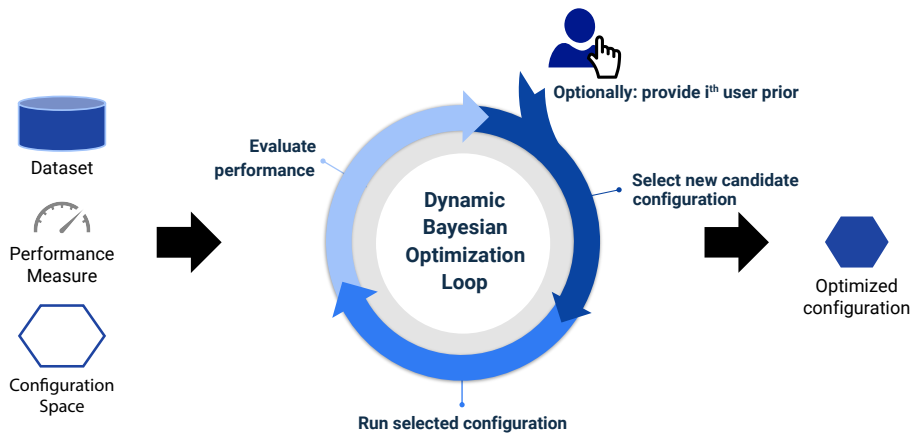


Figure 1: Overview of the proposed dynamic Bayesian optimization method, DynaBO. Provided a dataset, a performance measure, and an optional initial prior, the DynaBO loop iteratively selects new hyperparameter configurations. At each step, a candidate configuration is evaluated, and its performance is assessed until a budget is depleted. The framework allows users to steer the optimization process by dynamically adding priors at runtime.

Similarly, BO methods can incorporate structural priors about the behavior of the objective function, such as log-transformations (used in SMAC (Lindauer et al., 2022) and irace (López-Ibáñez et al., 2016)), monotonicity constraints (Li et al., 2018), partially defined solutions (Baudart et al., 2020), and non-stationary covariance modeling (Snoek et al., 2014).

Examples of user-priors on the optimum include fixed priors over the search space (Bergstra et al., 2011), posterior-driven models (Bergstra et al., 2011; Souza et al., 2021a), and search space warping (Ramachandran et al., 2020). More recently, Hvarfner et al. (2022) and Mallik et al. (2023) introduced πBO and PriorBand, extending BO and Hyperband (Li et al., 2017) respectively, thereby addressing limitations of preceding approaches, most importantly through robustness and flexibility.

Lastly, extending these approaches, the first methods in the direction of interactive BO integrate the user dynamically into the optimization process as safeguards (Adachi et al., 2024; Xu et al., 2024). However, these approaches do not allow users to steer the optimization process itself, but only query the user for feedback on the configurations queried.

3 Prior-Weighted Acquisition Function

Following Hvarfner et al. (2022), we integrate user-provided prior information on the location of the optimum by weighting the original acquisition function with a prior distribution. Given a surrogate model $\hat{f}$, an acquisition function α, and a user-specified prior $\pi : \Lambda \to (0, 1]$, the next point to be evaluated wrt. f at time t is selected by solving the following optimization problem:

$$\arg\max_{\lambda \in \Lambda} \alpha(\lambda) \cdot \pi^{\beta/t}(\lambda)\,,$$

where β is a scaling hyperparameter. Since with $t \to \infty$, the weight induced by π converges to 1 independent of the input λ, that is, the effect of the prior diminishes over time. Unlike the work of Hvarfner et al. (2022), we assume user-specified priors $\pi^{(m)}$ to arrive as a sequence $(t^{(1)}, \pi^{(1)}), (t^{(2)}, \pi^{(2)}), \ldots, (t^{(m)}, \pi^{(m)})$ at time points $t^{(i)}$ with $t^{(i)} \leq t$. Provided priors are then stacked. This results in a dynamically-adapted acquisition function α_{dyna} that blends the priors:

$$\alpha_{\text{dyna}}(\lambda) = \alpha(\lambda) \cdot \prod_{i=1}^{m} (\pi^{(i)})^{\beta/(t-t^{(i)})}(\lambda)\,.$$

Stacking the priors as described, we can incorporate the information given by the user at the different time steps. The priors are then faded individually based on their age, that is, older priors are considered less important as shown in Fig. 2. This flexibility sets our proposed method apart from πBO (Hvarfner et al., 2022) and Priorband (Mallik et al., 2023), which consider only a prior provided initially that decays in importance over time.

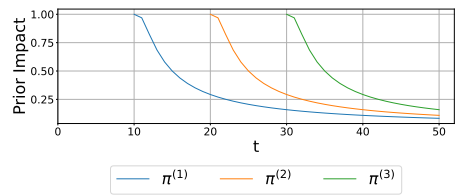


Figure 2: Impact visualisation of priors on the acquisition function for a total budget of 50 trials. The priors are provided at $t = 10, 20,$ and 30.

To handle spiky acquisition function distributions caused by priors, our method also adjusts the sampling density to capture these spikes. This ensures that promising configurations suggested by the prior are also found during acquisition function optimization, while additionally sampling random configurations for maintaining diversity.

4 Rejecting Priors

To safeguard against misleading priors, potentially slowing down DynaBO, we include a mechanism to reject priors based on their feasibility. Specifically, we assess the promisingness of a prior $\pi^{(m)}(\cdot)$ by comparing the potential of the suggested region against the region around the best previously found configuration, i.e., the current incumbent. We assess the potential of configurations $\lambda \in \Lambda$ of both regions in terms of the lower confidence bounds (LCB) acquisition function (Agrawal, 1995).

$$\mathrm{LCB}(\lambda) = (-1) \cdot (\mu(\lambda) - \kappa\sigma(\lambda)) \,,$$

Here, $\mu(\cdot)$ denotes the mean predicted with $\hat{f}$ for λ, and $\sigma(\cdot)$ the uncertainty in terms of standard deviation. Intuitively, the LCB criterion allows us to quantify the explorative potential of the prior region as opposed to the exploitative potential close to the current incumbent, since uncertainty close to the incumbent is typically low. Sampling a sufficiently-sized set of candidate configurations from normal distributions centered around the incumbent and the prior (denoted by $C_{\hat{\lambda}}$ and $C_{\pi^{(m)}}$, respectively), we compare the empirical expected values of both samples and accept the prior if it exceeds a given threshold τ:

$$\mathbb{E}_{\lambda \in C_{\pi^{(m)}}} [\mathrm{LCB}(\lambda)] - \mathbb{E}_{\lambda \in C_{\hat{\lambda}}} [\mathrm{LCB}(\lambda)] \geq \tau \,. \tag{1}$$

If τ is exceeded, the prior is accepted and vice versa. The higher τ is set, the fewer priors are accepted by DynaBO, but also the more misleading priors can be dodged effectively. Setting τ to a lower value lets DynaBO embrace priors more often, while making it more prone to be misled.

In summary, this approach ensures that optimization resources are not wasted on poor regions that are already expected to perform inferior to the current incumbent region. In a practical implementation, we envision that the user is warned against such a prior, but would have the chance to overrule its rejection.

5 Experiment Setup

In our empirical evaluation, we compare DynaBO against the state-of-the-art πBO and vanilla Bayesian optimization (Vanilla BO) using SMAC3 (Lindauer et al., 2022). πBO allows the user to provide prior information before the optimization process, whereas Vanilla BO is without any user guidance. In our experiments, we evaluate various models contained in YAHPO Gym (Pfisterer et al., 2022). A comprehensive overview of the experimental designs, along with the hardware resources utilized, is available in Appendix B. All optimization approaches are allocated a budget of 200 trials. Approaches that utilize priors receive informative, misleading, and semi-informative priors after 50 trials. In addition, DynaBO receives updated priors at 100 and 150 trials. Further details about the experimental setup can be found in Appendix A.

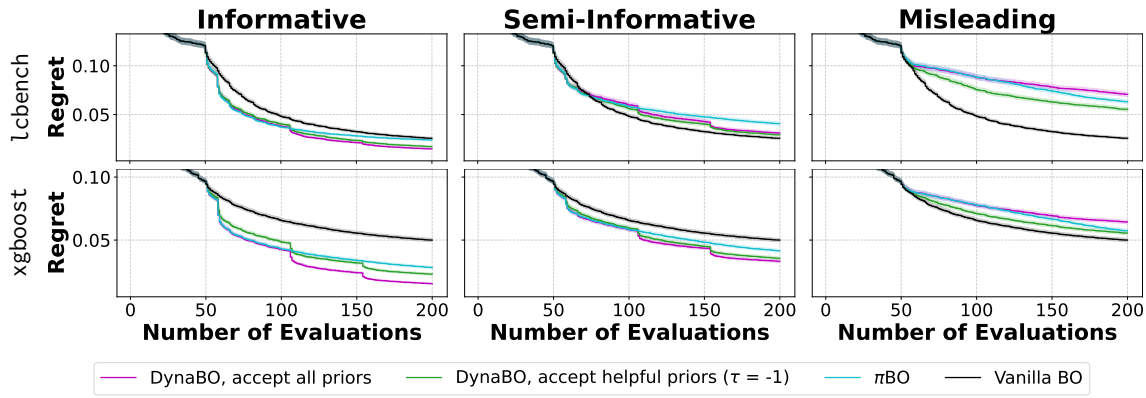


Figure 3: Mean anytime regret for `lcbench`, `rbv2_xgboost`, `rbv2_svm` and `rbv2_aknn`. The standard error is visualized as shaded areas.

6 Preliminary Results

In Fig. 3, we present anytime performance plots for the scenarios `lcbench` and `xgboost`. Additional results can be found in Appendix C. The plots show the mean regret of the best incumbents found, over the number of evaluations of the black-box function f. We consider three different kinds of priors, simulating an optimistic, more realistic, and pessimistic setting, respectively.

Informative Priors Overall, `DynaBO` variants perform superior to Vanilla BO, and equal or superior to πBO. `DynaBO` outperforms its competitors both in terms of final and anytime performance. However, false rejection of helpful priors results in decreased performance.

Semi-Informative Priors In this evaluation, both πBO and all `DynaBO` methods outperform Vanilla BO on most scenarios. While `DynaBO` without rejection performs best, `DynaBO` with rejection and πBO perform similarly. Possibly, too many informative priors are rejected.

Misleading Priors Here, Vanilla BO performs best, followed by `DynaBO` with rejection, πBO, and `DynaBO` without rejection, indicating that `DynaBO` is able to disregard misleading priors effectively.

As a general trend, `DynaBO` with and without rejection perform equal to or superior to πBO. The capability of rejecting priors, however, also comes at the risk of erroneously rejecting informative priors. As a result, the performance of `DynaBO` with prior rejection falls behind that of the variant without prior rejection, indicating that experienced users should be able to overrule the rejection.

7 Conclusion

In this work, we have presented `DynaBO`, a method that seamlessly incorporates user priors into Bayesian optimization for hyperparameter optimization. To this end, it employs a generalized acquisition function that accepts priors at any point during the optimization process. To ensure robustness, `DynaBO` decays the influence of priors over time, as well as includes a prior disregarding safeguard. Empirical results demonstrate that `DynaBO` outperforms both vanilla BO and BO with a static prior, even without the safeguard; nonetheless, the further increases robustness.

However, while our empirical evaluation considers different kinds of user priors, they are generated synthetically and provided at prespecified time points. Additionally, our proposed prior-rejection scheme sometimes falsely disregards helpful priors. Future work could focus on expanding the types of supported priors and developing improved rejection mechanisms, e.g., by evaluating prior-based configurations. Moreover, exploring the capacity of language models to encode user preferences as priors, as well as to autonomously generate priors, similar to (Chang et al., 2025), is a promising direction, particularly when informed by explainability methods (Sass et al., 2022; Wever et al., 2025) or knowledge representations such as knowledge graphs (Kostovska et al., 2022).

Acknowledgements. Lukas Fehring, 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. The authors gratefully acknowledge the computing time provided to them on the high-performance computers Noctua2 at the NHR Center PC2. 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 highperformance computing at universities (`www.nhr-verein.de/unsere-partner`).

References

Adachi, M., Planden, B., Howey, D. A., Osborne, M. A., Orbell, S., Ares, N., Muandet, K., and Chau, S. L. (2024). Looping in the human: Collaborative and explainable bayesian optimization. In Dasgupta, S., Mandt, S., and Li, Y., editors, *International Conference on Artificial Intelligence and Statistics, 2-4 May 2024, Palau de Congressos, Valencia, Spain*, volume 238 of *Proceedings of Machine Learning Research*, pages 505–513. PMLR.

Agrawal, R. (1995). Sample mean based index policies by o (log n) regret for the multi-armed bandit problem. *Advances in Applied Probability*.

Bansal, A., Stoll, D., Janowski, M., Zela, A., and Hutter, F. (2022). JAHS-bench-201: A foundation for research on joint architecture and hyperparameter search. In Koyejo, S., Mohamed, S., Agarwal, A., Belgrave, D., Cho, K., and Oh, A., editors, *Proceedings of the 36th International Conference on Advances in Neural Information Processing Systems (NeurIPS'22)*. Curran Associates.

Baudart, G., Hirzel, M., Kate, K., Ram, P., and Shinnar, A. (2020). Lale: Consistent automated machine learning. In *KDD Workshop on Automation in Machine Learning (AutoML@KDD)*.

Bergstra, J., Bardenet, R., Bengio, Y., and Kégl, B. (2011). Algorithms for hyper-parameter optimization. In Shawe-Taylor, J., Zemel, R., Bartlett, P., Pereira, F., and Weinberger, K., editors, *Proceedings of the 25th International Conference on Advances in Neural Information Processing Systems (NeurIPS'11)*, pages 2546–2554. Curran Associates.

Brochu, E., Brochu, T., and de Freitas, N. (2010). A Bayesian interactive optimization approach to procedural animation design. In Popovic, Z. and Otaduy, M., editors, *Proceedings of the 2010 Eurographics/ACM SIGGRAPH Symposium on Computer Animation (SCA'10)*, pages 103–112. Eurographics Association.

Chang, C., Azvar, M., Okwudire, C., and Kontar, R. A. (2025). Llinbo: Trustworthy llm-in-the-loop bayesian optimization. *arXiv:2505.14756 [cs.LG]*.

Eggensperger, K., Müller, P., Mallik, N., Feurer, M., Sass, R., Klein, A., Awad, N., Lindauer, M., and Hutter, F. (2021). HPOBench: A collection of reproducible multi-fidelity benchmark problems for HPO. In Vanschoren, J. and Yeung, S., editors, *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*. Curran Associates.

Feurer, M., Klein, A., Eggensperger, K., Springenberg, J., Blum, M., and Hutter, F. (2015). Efficient and robust automated machine learning. In Cortes, C., Lawrence, N., Lee, D., Sugiyama, M., and Garnett, R., editors, *Proceedings of the 29th International Conference on Advances in Neural Information Processing Systems (NeurIPS'15)*, pages 2962–2970. Curran Associates.

Feurer, M., Letham, B., Hutter, F., and Bakshy, E. (2022). Practical transfer learning for bayesian optimization. *arXiv:1802.02219v4 [stat.ML]*.

Hutter, F., Kotthoff, L., and Vanschoren, J., editors (2019). *Automated Machine Learning: Methods, Systems, Challenges*. Springer. Available for free at `http://automl.org/book`.

Hvarfner, C., Stoll, D., Souza, A., Nardi, L., Lindauer, M., and Hutter, F. (2022). πBO: Augmenting Acquisition Functions with User Beliefs for Bayesian Optimization. In *Proceedings of the International Conference on Learning Representations (ICLR'22)*. Published online: `iclr.cc`.

Kostovska, A., Jankovic, A., Vermetten, D., Nobel, J., Wang, H., Eftimov, T., and Doerr, C. (2022). Per-run algorithm selection with warm-starting using trajectory-based features. *arXiv:2204.09483 [cs.LG]*.

Li, C., Rana, S., Gupta, S., Nguyen, V., Venkatesh, S., Sutti, A., de Celis Leal, D. R., Slezak, T., Height, M., Mohammed, M., and Gibson, I. (2018). Accelerating experimental design by incorporating experimenter hunches. In *IEEE International Conference on Data Mining, ICDM 2018, Singapore, November 17-20, 2018*, pages 257–266. IEEE Computer Society.

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

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. *Journal of Machine Learning Research*, 23(54):1–9.

Lindauer, M., Karl, F., Klier, A., Moosbauer, J., Tornede, A., Müller, A., Hutter, F., Feurer, M., and Bischl, B. (2024). Position: A call to action for a human-centered automl paradigm. In *Forty-first International Conference on Machine Learning, ICML*. OpenReview.net.

López-Ibáñez, M., Dubois-Lacoste, J., Caceres, L. P., Birattari, M., and Stützle, T. (2016). The irace package: Iterated racing for automatic algorithm configuration. *Operations Research Perspectives*, 3:43–58.

Mallik, N., Hvarfner, C., Bergman, E., Stoll, D., Janowski, M., Lindauer, M., Nardi, L., and Hutter, F. (2023). PriorBand: Practical hyperparameter optimization in the age of deep learning. In *Proceedings of the 37th International Conference on Advances in Neural Information Processing Systems (NeurIPS'23)*. Curran Associates.

Müller, S., Feurer, M., Hollmann, N., and Hutter, F. (2023). PFNs4BO: In-Context Learning for Bayesian Optimization. In Krause, A., Brunskill, E., Cho, K., Engelhardt, B., Sabato, S., and Scarlett, J., editors, *Proceedings of the 40th International Conference on Machine Learning (ICML'23)*, volume 202 of *Proceedings of Machine Learning Research*. PMLR.

Papenmeier, L., Cheng, N., Becker, S., and Nardi, L. (2025). Exploring exploration in bayesian optimization. *arXiv:2502.08208 [cs.LG]*.

Perrone, V., Shen, H., Seeger, M., Archambeau, C., and Jenatton, R. (2019). Learning search spaces for bayesian optimization: Another view of hyperparameter transfer learning. In Wallach, H., Larochelle, H., Beygelzimer, A., d'Alche Buc, F., Fox, E., and Garnett, R., editors, *Proceedings of the 33rd International Conference on Advances in Neural Information Processing Systems (NeurIPS'19)*, pages 12751–12761. Curran Associates.

Pfisterer, F., Schneider, L., Moosbauer, J., Binder, M., and Bischl, B. (2022). YAHPO Gym – an efficient multi-objective multi-fidelity benchmark for hyperparameter optimization. In Guyon, I., Lindauer, M., van der Schaar, M., Hutter, F., and Garnett, R., editors, *Proceedings of the First*

International Conference on Automated Machine Learning. Proceedings of Machine Learning Research.

Pineda Arango, S., Ferreira, F., A., K., F., H., and J., G. (2024). Quick-tune: Quickly learning which pretrained model to finetune and how. In *International Conference on Learning Representations (ICLR'24)*. Published online: `iclr.cc`.

Rakotoarison, H., Adriaensen, S., Mallik, N., Garibov, S., Bergman, E., and Hutter, F. (2024). In-context freeze-thaw bayesian optimization for hyperparameter optimization. In *Forty-first International Conference on Machine Learning*.

Ramachandran, A., Gupta, S., Rana, S., Li, C., and Venkatesh, S. (2020). Incorporating expert prior in Bayesian optimisation via space warping. *Knowledge-Based Systems*, 195.

Sass, R., Bergman, E., Biedenkapp, A., Hutter, F., and Lindauer, M. (2022). Deepcave: An interactive analysis tool for automated machine learning. In Mutny, M., Bogunovic, I., Neiswanger, W., Ermon, S., Yue, Y., and Krause, A., editors, *ICML Adaptive Experimental Design and Active Learning in the Real World (ReALML Workshop 2022)*.

Snoek, J., Swersky, K., Zemel, R., and Adams, R. (2014). Input warping for Bayesian optimization of non-stationary functions. In Xing, E. and Jebara, T., editors, *Proceedings of the 31th International Conference on Machine Learning, (ICML'14)*, pages 1674–1682. Omnipress.

Souza, A., Nardi, L., Oliveira, L., Olukotun, K., Lindauer, M., and Hutter, F. (2021a). Bayesian optimization with a prior for the optimum. In Oliver, N., Pérez-Cruz, F., Kramer, S., Read, J., and Lozano, J., editors, *Machine Learning and Knowledge Discovery in Databases (ECML/PKDD'21)*, volume 12975 of *Lecture Notes in Computer Science*, pages 265–296. Springer.

Souza, A., Nardi, L., Oliveira, L., Olukotun, K., Lindauer, M., and Hutter, F. (2021b). Bayesian optimization with a prior for the optimum. In Oliver, N., Pérez-Cruz, F., Kramer, S., Read, J., and Lozano, J. A., editors, *Machine Learning and Knowledge Discovery in Databases. Research Track*, volume 12975 of *Lecture Notes in Artificial Intelligence*, page 265–296. Springer-Verlag.

Stoll, D., Franke, J., Wagner, D., Selg, S., and Hutter, F. (2020). Hyperparameter transfer across developer adjustments. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M.-F., and Lin, H., editors, *Proceedings of the 34th International Conference on Advances in Neural Information Processing Systems (NeurIPS'20)*. Curran Associates.

Swersky, K., Snoek, J., and Adams, R. (2013). Multi-task Bayesian optimization. In Burges, C., Bottou, L., Welling, M., Ghahramani, Z., and Weinberger, K., editors, *Proceedings of the 27th International Conference on Advances in Neural Information Processing Systems (NeurIPS'13)*, pages 2004–2012. Curran Associates.

Tornede, T., Tornede, A., Hanselle, J., Mohr, F., Wever, M., and Hüllermeier, E. (2023). Towards green automated machine learning: Status quo and future directions. *Journal of Artificial Intelligence Research*, 77:427–457.

Van der Blom, K., Serban, A., Hoos, H., and Visser, J. (2021). Automl adoption in ml software. In *8th ICML Workshop on Automated Machine Learning (AutoML)*.

van Rijn, J. and Hutter, F. (2018). Hyperparameter importance across datasets. In Guo, Y. and Farooq, F., editors, *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'18)*, pages 2367–2376. ACM Press.

Wang, Q., Ming, Y., Jin, Z., Shen, Q., Liu, D., Smith, M., Veeramachaneni, K., and Qu, H. (2019). Atmseer: Increasing transparency and controllability in automated machine learning. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems (CHI'19)*, page 1–12. ACM Press.

Wang, Z., Dahl, G., Swersky, K., Lee, C., Mariet, Z., Nado, Z., Gilmer, J., Snoek, J., and Ghahramani, Z. (2024). Pre-trained Gaussian processes for Bayesian optimization. *J. Mach. Learn. Res.*

Wever, M., Muschalik, M., Fumagalli, F., and Lindauer, M. (2025). Hypershap: Shapley values and interactions for hyperparameter importance. *arXiv:2502.01276 [cs.LG]*.

Wistuba, M., Schilling, N., and Schmidt-Thieme, L. (2015). Hyperparameter search space pruning - A new component for sequential model-based hyperparameter optimization. In Appice, A., Rodrigues, P., Costa, V., Gama, J., Jorge, A., and Soares, C., editors, *Machine Learning and Knowledge Discovery in Databases (ECML/PKDD'15)*, volume 9285 of *Lecture Notes in Computer Science*, pages 104–119. Springer.

Xu, W., Adachi, M., Jones, C. N., and Osborne, M. A. (2024). Principled bayesian optimisation in collaboration with human experts. *CoRR*, abs/2410.10452.

Zöller, M., Titov, W., Schlegel, T., and Huber, M. F. (2023). Xautoml: A visual analytics tool for understanding and validating automated machine learning. *ACM Trans. Interact. Intell. Syst.*, 13(4):28:1–28:39.

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? `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)? [Yes]

   (b) Did you specify all the necessary details of your evaluation (e.g., data splits, pre-processing, search spaces, hyperparameter tuning)? [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 variance across random seeds or splits)? [Yes]

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

   (f) Did you use tabular or surrogate benchmarks for in-depth evaluations? [Yes]

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

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 requirements (e.g., `requirements.txt` with explicit versions), random seeds, an instructive `README` with installation, 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 Artificial Priors

To evaluate the concept of dynamic priors, we design a benchmark grounded in real hyperparameter optimization (HPO) scenarios but with artificial, data-generated priors provided after 50, 100, and 150 priors. In our analysis, we consider three kinds of priors to design challenging experiments in a controlled way.

Informative Priors facilitate sampling well-performing configurations. Their center is a pre-evaluated configuration of superior performance compared to the current incumbent.

Misleading Priors are centered around configurations inferior to the current incumbent, steering the optimization process into low-performance regions.

Semi-Informative Priors combine the idea of informative and misleading priors. They simulate users generating both informative and misleading priors with probability 0.5. With this kind of prior, we assume a user to provide randomly both helpful and unhelpful information.

To construct priors, we follow the idea of a preceding exploration of the search space as proposed by Souza et al. (2021b) and Hvarfner et al. (2022). From the collected data, priors are then constructed using a normal distribution centered around the pre-evaluated incumbents, with priors becoming increasingly peaked as the optimization process progresses. During the optimization process, priors are provided with respect to the current incumbent.

Detailed Prior Generation.

1. Generate prior data: For each learner A, dataset D combination, execute explorative Bayesian optimization runs with both the rather greedy Expected Improvement (EI) and explorative Lower Confidence Bounds (LCB) (Papenmeier et al., 2025). For each acquisition function, run 10 seeds for a budget of $5,000$ iterations. Then, for every algorithm, dataset combination, concatenate the lists of incumbents and assemble a joint list sorted by ℓ.

$$I_{A,D} = [(\hat{\lambda}_1, \ell_{\hat{\lambda}_1}), (\hat{\lambda}_2, \ell_{\hat{\lambda}_2}), \ldots, (\hat{\lambda}_n, \ell_{\hat{\lambda}_n})]$$

   Note the abuse of notation: We denote with ℓ_λ the performance on the validation set

$$\ell_\lambda = \left[\frac{1}{|D_V|} \sum_{(x,y) \in D_V} \ell\left(y, h_{\lambda,D_T}(x)\right) \right]$$

   Here h_{λ,D_T} is a result of training Algorithm A on D_T using λ.

2. Filter prior data: For each algorithm and dataset combination, initialize a list for the filtered priors with the best and worst-performing incumbent. Then, greedily augment the filtered list by adding incumbents that meet a pair-wise performance difference of at least $\epsilon = 0.05$. This results in a list of incumbents for each combination of dataset and algortihm of size $m \leq n$.

$$\begin{aligned} I_{A,D}^{filtered} = \Big[& (\hat{\lambda}_{(1)}, \ell_{\hat{\lambda}_{(1)}}), (\hat{\lambda}_{(2)}, \ell_{\hat{\lambda}_{(2)}}), \ldots, (\hat{\lambda}_{(m-1)}, \ell_{\hat{\lambda}_{(m-1)}}), (\hat{\lambda}_{(m)}, \ell_{\hat{\lambda}_{(m)}}) : \\ & (\hat{\lambda}_1, \ell_{\hat{\lambda}_1}) = (\hat{\lambda}_{(1)}, \ell_{\hat{\lambda}_{(1)}}), (\hat{\lambda}_{(m)}, \ell_{\hat{\lambda}_{(m)}}) = (\hat{\lambda}_n, \ell_{\hat{\lambda}_n}) \\ & \text{and } |\ell_{\hat{\lambda}_{(i)}} - \ell_{\hat{\lambda}_{(i+1)}}| \geq \epsilon \ \forall i \in 1, ..., m-1 \Big] \subseteq I_{A,D} \end{aligned}$$

Prior Injection. To then utilize a prior when search for hyperparameter configurations for learner A on dataset D, the priors are generated dynamically as follows:

1. Sample configuration to be used as a prior: Given a current optimization run, with current incumbent configuration $\lambda^+, \ell_{\lambda^+}$ select $(\lambda^p, \ell_{\lambda^p}) \in I_{A,D}^{filtered}$ used as a prior.

- For informative priors, the sampled config is better than the current incumbent, that means $\ell_\lambda^p \geq \ell_{\lambda^+}$.
- For misleading priors, the sampled config is worse than the current incumbent, that means $\ell_\lambda^p \leq \ell_{\lambda^+}$.
- For semi-informative priors, it is either informative or misleading with a chance of 0.5.

2. Build prior: We assume for each prior provided to DynaBO, the confidence of a user would grow. In our synthetic prior generation, we therefore build the k-th prior π^k as follows:

   - For each hyperparameter in a search space of dimension d, investigate the lower bounds $\lambda_1^l, \lambda_2^l, ..., \lambda_d^l$ and upper bounds $\lambda_1^u, \lambda_2^u, ..., \lambda_d^u$
   - For each hyperparameter in a search space, set an independent normally distributed prior:

$$\pi^k = [\mu_j, \sigma_j]_{j=1}^{j=d} = [(\lambda_j^p, \frac{|\lambda_j^u - \lambda_j^l|}{k \cdot 5})]_{j=1}^{j=d}$$

   As mentioned in Section 5, we provide three priors π^1, π^2, π^3 in our experiments.

B Further Experimental Design

For each model and dataset, we repeat each HPO run 30 times with different seeds. Given the same seed DynaBO and πBO will sample identical (first) priors. We utilize 200 trials, a reasonable amount of trials for affordable models, and three priors after 50, 100, and 150 trials. Following the recommendations by Hvarfner et al. (2022), β is initialized as $N/10$ with N denoting the number of overall trials. In our experimental results, we report the mean performance and standard error. All experiments were run executed on HPC nodes equipped with Intel Xeon Skylake 6148 @2.4GHz and 192GiB RAM, of which 2 CPU cores and 16GB RAM were allocated per run.

Our experiments are scheduled, and the results are logged in a MySQL database using the PyExperimenter library (Tornede et al., 2023). For the YAHPO Gym data-generation runs, totalled to 900 CPU days. For the YAHPO Gym experiment runs, we invested 760 CPU days. For the PD1 experiments, data generation took approximately five CPU days, and the final experiments approximately one CPU day.

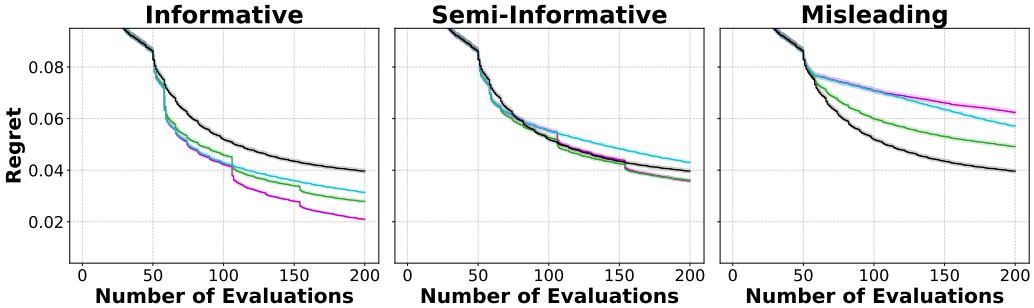
B.1 Summary of the Benchmark

A learner, searchspace, and dataset overview is provided in Table 1. To focus on challenging scenarios among YAHPO Gym datasets and reduce computational load, we preselect them by conducting one exploration run for Expected Improvement and another for Confidence Bounds, storing the optimal performance and the list of incumbents. To achieve this, we assess when the performance of either run first reaches within 5fj of the final performance, and we retain only those datasets that required more than 500 trials to reach this threshold for both runs.

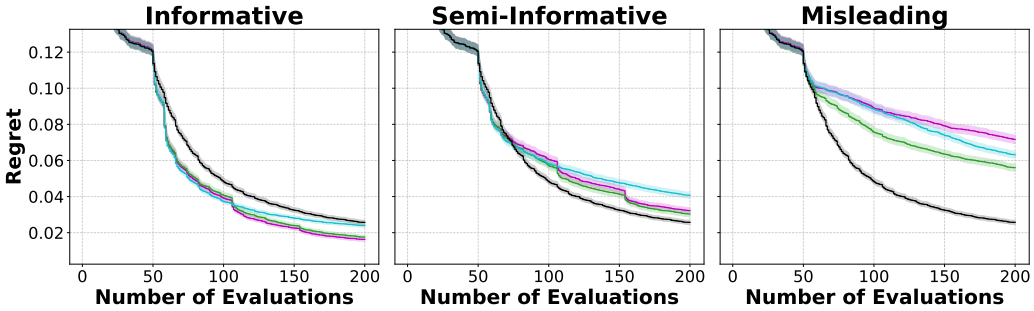
Table 1: Table of scenarios with search space type and number of datasets

Scenario	SearchSpace	#Datasets
rbv2_super	38D: Mixed	68
rbv2_svm	6D: Mixed	14
rbv2_rpart	5D: Mixed	20
rbv2_aknn	6D: Mixed	25
rbv2_glmnet	3D: Mixed	12
rbv2_ranger	8D: Mixed	45
rbv2_xgboost	14D: Mixed	66
lcbench	7D: Numeric	14

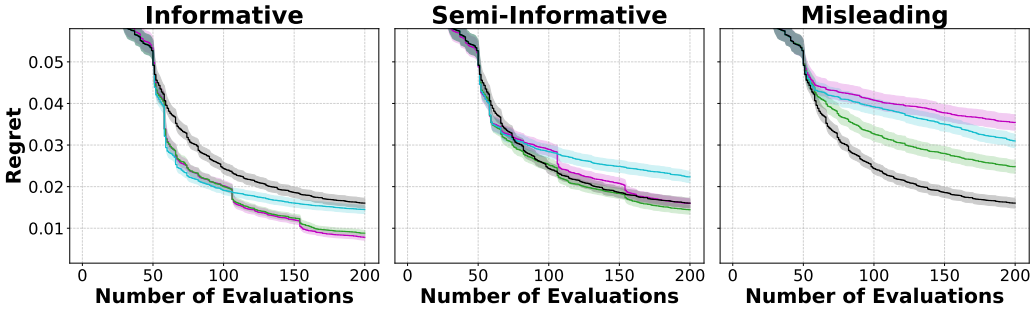
C Additional Empirical Results



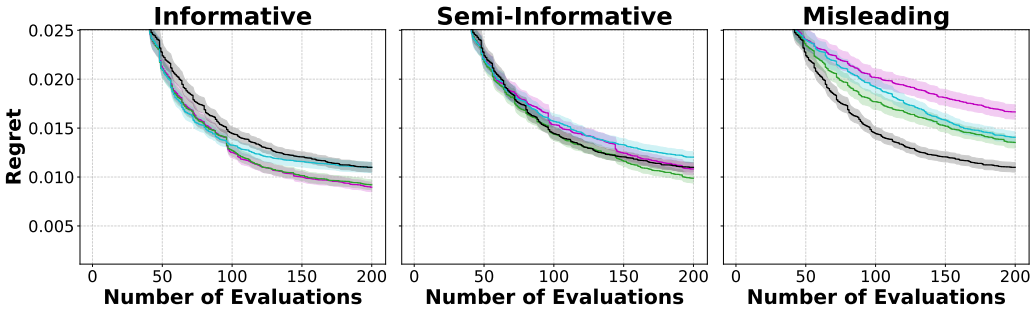
(a) Evaluation results averaged over all learner, dataset combinations.



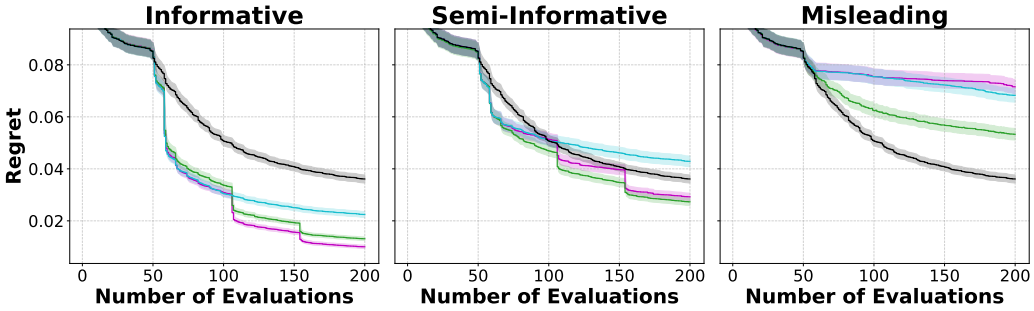
(b) Evalaution results of `lcbench`.



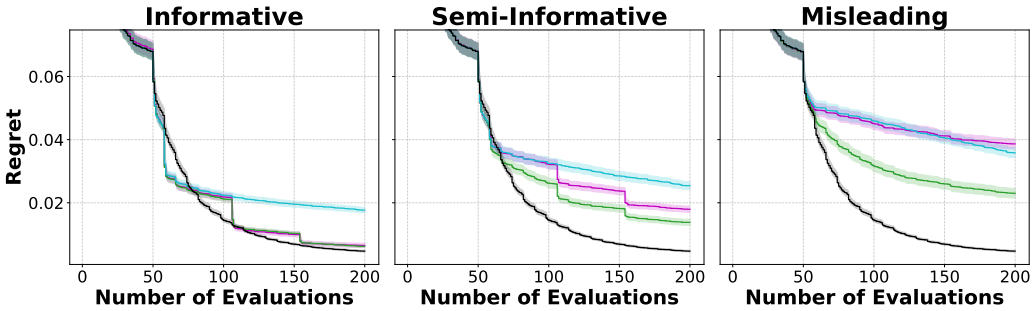
(c) Evalaution results of `rbv2_aknn`.



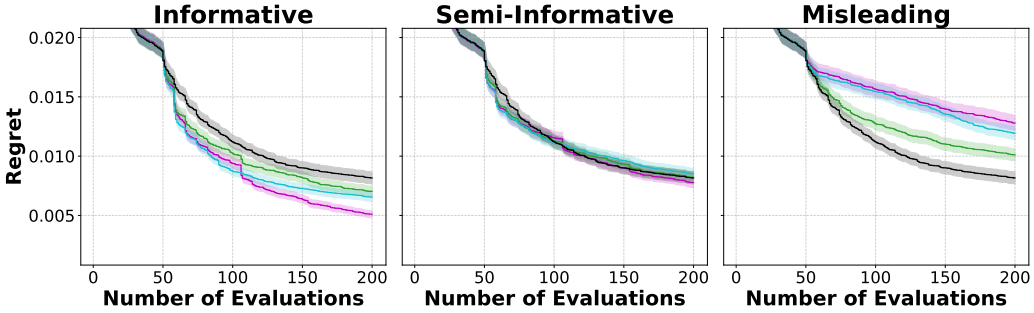
(d) Evalaution results of `rbv2_glmnet`.



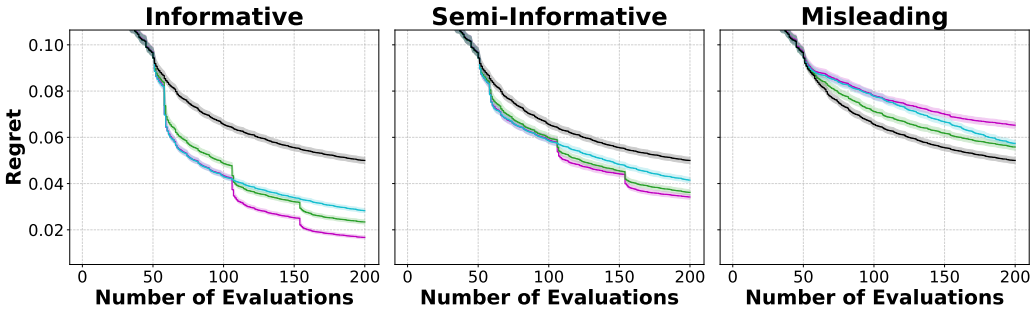
(a) Evalaution results of `rbv2_ranger`.



(b) Evalaution results of `rbv2_rpart`.



(c) Evalaution results of `rbv2_svm`.



(d) Evalaution results of `rbv2_xgboost`.