Reproducibility of the NeurIPS 2020 Black-Box Optimization Challenge: Re-evaluating Competition Submissions

Ivan Raytchinov
RWTH Aachen University
Aachen, Germany
ivan.raytchinov@rwth-aachen.de

Marie Anastacio
RWTH Aachen University
Aachen, Germany
Leiden University
Leiden, The Netherlands
anastacio@aim.rwth-aachen.de

Anja Jankovic
RWTH Aachen University
Aachen, Germany
jankovic@aim.rwth-aachen.de

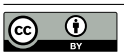
Odd Erik Gundersen
Norwegian University of Science and Technology
Trondheim, Norway
odderik@ntnu.no

Holger H. Hoos
RWTH Aachen University
Aachen, Germany
Leiden University
Leiden, The Netherlands
hh@aim.rwth-aachen.de

Abstract

Competitions are widely used to assess the performance of AI software, but the relevance of their results relies on the transparency of the protocols used for running them and the reproducibility of their results. We present a case study of the NeurIPS 2020 Black-Box Optimization (BBO) Challenge and assess whether its published outcomes can be successfully reproduced. Because the original hidden benchmark problems are unavailable, we focus on structural reproducibility: the recovery of broad performance tiers, stable strengths and weaknesses, and positive rank agreement, even when exact scores cannot be matched. We rebuilt the evaluation pipeline described in the original paper and ran five top entries (HEBO, JetBrains, Optuna, TuRBO variants, RAPIDS components), together with a Random Search baseline across 14 tasks from five benchmark families (`Bayesmark`, `BBOB`, `LCBench`, `HPO-B`, and `MechBench`). Unfortunately, the original evaluation benchmark was not provided by the competition organizers (even upon request). While broad performance trends can be reproduced on our benchmarks, an exact replication of the leaderboard ranking could not be achieved. Several factors contribute to these findings, such as the non-availability of the original benchmark problems, dependency drift, hardware-specific pipelines and variations in hardware and software environments. Our results show that post-hoc reproduction often yields only partial ranking stability rather than exact score matching. We provide concrete design recommendations for future competitions to facilitate reproducibility of their results.

ACM Reference Format:
Ivan Raytchinov, Marie Anastacio, Anja Jankovic, Odd Erik Gundersen, and Holger H. Hoos. 2026. Reproducibility of the NeurIPS 2020 Black-Box Optimization Challenge: Re-evaluating Competition Submissions. In *ACM Conference on Reproducibility and Replicability (ACM REP '26), July 20–22, 2026, Delft, Netherlands.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3820002.3828590



This work is licensed under a Creative Commons Attribution 4.0 International License.
ACM REP '26, Delft, Netherlands
© 2026 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-2778-8/2026/07
https://doi.org/10.1145/3820002.3828590

1 Introduction

In many domains, competitions are a major driving force for algorithmic innovation, offering standardized platforms for comparative evaluation. Competitions inherently produce value for participants through shared benchmarks, community engagement and direct algorithmic comparison. The broader scientific contribution of competitions is substantially strengthened when third parties can independently verify the published outcomes. When reproducibility is limited, the scientific impact of a competition is at least diminished. As software environments drift and test sets remain proprietary and hidden, validating claims about winners becomes increasingly difficult. Recent studies on reproducibility in machine learning highlight the fragility of published outcomes and the importance of transparent reporting on experiments [34], especially regarding datasets [15]. Competition design choices that partition hidden test sets are motivated by the structural risk of leaderboard overfitting [35, 45], but leaving those test sets inaccessible even after the competition results are published prevents later verification. Since rankings are dependent on the choice of problem instances on which algorithms are evaluated, keeping this underlying data hidden in the context of competitions for which the final ranking is a key contribution lowers the usefulness of the published results.

Black-box optimization, and specifically hyperparameter optimization (HPO), is a critical component of modern machine learning pipelines. Because evaluating model performance is computationally expensive and gradients are unavailable, sample-efficient strategies such as Bayesian optimization are heavily relied upon for finding good configurations within tight computational budgets [39]. Despite this, empirical surveys have found that only a small fraction of papers at major conferences used Bayesian optimization at the time of the NeurIPS 2020 BBO Challenge [3], underscoring the need for robust and accessible HPO tools. This competition explicitly aimed at benchmarking general-purpose black-box optimizers for HPO at scale [43]. The BBO Challenge had a particularly large impact on the visibility and adoption of its top-performing entries:

the winning algorithm, HEBO [4], became one of the most widely cited Bayesian optimization packages in the years following the competition, and other entries, such as TuRBO [8], were integrated into major optimization frameworks. Because the competition outcomes directly shaped the perceived state of the art in HPO, the reproducibility of its results carries outsized importance for the community.

Building on these observations about the interplay between competition design, reproducibility challenges and the practical importance of HPO benchmarks, in this work, we revisit the NeurIPS 2020 BBO Challenge several years after its conclusion and investigate three primary research questions, each paired with a concrete contribution. On reproducibility: Can the original leaderboard rankings be independently reproduced, using the earliest available public repositories, given that the hidden test benchmark problems are no longer available? To answer this, we built a unified, optimizer-agnostic evaluation framework that executes historically heterogeneous codebases in isolated runtime environments through a shared black-box protocol, and we quantified ranking stability against the official challenge ordering on Bayesmark, a public benchmark family that contains problems from the same underlying distribution as the hidden test problem set, which serves as a proxy that most closely resembles undisclosed problems. On generalization: Do the relative performance rankings hold when evaluated out-of-domain on benchmark families beyond the original Bayesmark setting (*e.g.,* numerical optimization, constrained engineering tasks)? To this end, we conducted a cross-domain empirical study over 14 tasks from five benchmark families, reporting aggregate results on a statistically discriminative subset of nine benchmarks. On competition design: What practical lessons can be drawn towards engineering future competitions for maximum legacy value and robust reproducibility? We translate our findings into operational design recommendations emphasizing archival quality, hardware transparency, and statistically robust reporting of results.

Code and Data Availability. We provide our code and data in a GitHub repository: https://github.com/ADA-research/bbocomp-reproducibility.

The remainder of the paper is organized as follows: in Section 2, we introduce relevant background on black-box optimization, reproducibility in machine learning and automated machine learning, and reproducibility in AI competitions. We outline our setup of experiments in Section 3 and present our results in Section 4. We critically discuss our findings with respect to our three research questions in Section 5 and provide concluding remarks in Section 6.

2 Background

In this section, we review the key concepts and prior work that underpin our study: the field of black-box optimization and the NeurIPS 2020 BBO Challenge itself (Section 2.1), the broader discussion around reproducibility in machine learning and AutoML (Section 2.2), and lessons drawn from the design of other AI competitions (Section 2.3).

2.1 Black-Box Optimization and the NeurIPS 2020 BBO Challenge

Black-box optimization seeks to find an optimal solution for an objective function $f(x)$ where $x \in \mathcal{X}$ without access to gradient information. Black-box optimization algorithms iteratively query the black-box function (often in parallel batches) and use historical observations to propose new search points, balancing exploration and exploitation.

A broad class of black-box optimization methods are model-free, in that they maintain no explicit surrogate of the objective function f, and instead rely on population-level heuristics or stochastic search rules. A prominent example is the CMA-ES (Covariance Matrix Adaptation Evolution Strategy) [21], which maintains a population of candidate solutions and adapt a multivariate Gaussian distribution over search directions based on the relative fitness of evaluated points, allowing the search to orient itself along promising dimensions of the space without gradient information.

In contrast, model-based black-box optimization methods construct an explicit probabilistic surrogate of the objective function f, and use it to reason about where evaluations would be most informative or rewarding. Bayesian optimization [11, 12, 27, 31] is a prominent example of this family of methods. Bayesian optimization fits a probabilistic model, most commonly a Gaussian process, to the observed pairs of candidate solutions x and function values $f(x)$, yielding a posterior distribution over f that quantifies both the predicted performance and the uncertainty across the decision space $\mathcal{X}$. An acquisition function, *e.g.,* Expected Improvement [30], Upper Confidence Bound [10] or Thompson Sampling [42], then trades off between querying points predicted to be good (exploitation) and points where the model is uncertain (exploration). Because each evaluation updates the surrogate, Bayesian optimization is particularly well-suited to use-cases where the objective function f is expensive to evaluate, such as HPO, materials discovery or drug design, and where sample efficiency is crucial. Extensions such as trust-region Bayesian optimization (TuRBO) [8] and batch Bayesian optimization [13] address the challenges of high-dimensional spaces and parallel evaluation budgets, respectively, broadening the practical applicability of the approach.

Formally, after t observations $\mathcal{D}_t = \{(\mathbf{x}_i, f(\mathbf{x}_i))\}_{i=1}^{t}$, Bayesian optimization selects the next evaluation point by maximizing an acquisition function $a_t(\mathbf{x})$ derived from the posterior surrogate model: $\mathbf{x}_{t+1} \in \arg\max_{\mathbf{x}\in\mathcal{X}} a_t(\mathbf{x})$. For example, expected improvement chooses points with high expected gain over the best observed value, while uncertainty-aware criteria such as upper confidence bounds combine posterior mean and uncertainty to trade off exploitation and exploration.

The NeurIPS 2020 BBO Challenge was a one-time competition organized via the NeurIPS Datasets and Benchmarks track by the Bayesmark team [44], with a focus on general-purpose optimizers for hyperparameter tuning of `scikit-learn` models. Participants submitted optimizers that were evaluated using the `Bayesmark` benchmarking framework on a mixture of benchmark problems. A crucial design choice was that the final leaderboard was determined using a hidden set of benchmark problems in order to encourage genuine generalization and to mitigate leaderboard overfitting [43]. Those were never released publicly.

2.2 Reproducibility in Machine Learning Research and AutoML

Reproducibility has become a central concern in the machine learning community, with fundamental principles now well established [14], leading to formalized reporting guidelines, badges, and dedicated reproducibility challenges [6, 17, 18, 25, 29, 34]. Reproducibility is facilitated through transparency at multiple levels: making code, data, runtime environments, and initialization states accessible to the community. The sources of irreproducibility are numerous and varied [16], and code availability alone does not guarantee that results can be replicated.

Foundational work demonstrates that, in the context of solver and classifier comparisons, point-estimate leaderboards are fragile due to statistical noise and sensitivity to initial seeds [5, 33, 47]. Without rigorous resampling or sufficient transparency regarding the setup of experiments, tuning methodology [40], computational budget [6], exact replication of specific leaderboard outcomes over time can be very challenging [2].

Automated Machine Learning (AutoML) faces unique reproducibility hurdles because the optimization process itself spans massive combinatorial search spaces. Analyses of the ChaLearn AutoML challenge series found that progressive disclosure and post-competition open-sourcing of full evaluation pipelines were critical for maintaining scientific impact and enabling independent verification [19]. As competitions increasingly rely on computational ensembling and complex pipelines, evaluating bare algorithmic contributions becomes more difficult. These observations strengthen the case for containerized, robustly designed benchmarks that level the playing field and preserve long-term reproducibility [4, 23].

2.3 Reproducibility in AI Competitions

The NeurIPS 2020 BBO Challenge, while innovative in its focus on general-purpose optimizers for machine learning hyperparameter tuning, exemplifies systemic issues that have appeared in many AI competitions. Looking at other major competitions reveals strategies that have successfully enhanced reproducibility as well as persistent challenges.

The ChaLearn AutoML Challenge Series (2015–2018) represents one of the most methodologically rigorous approaches to competition design in this area. Unlike the BBO Challenge, the ChaLearn organizers implemented a progressive disclosure framework where hidden test problems were gradually released to participants during different phases, and crucially, the complete evaluation infrastructure was made publicly available after the competition concluded [19]. This approach enabled independent verification of results and facilitated post-competition research that built directly upon the challenge methodology.

Similarly, the NeurIPS Reproducibility Challenges adopted a paradigm in which the primary objective is to reproduce published results through documented code and protocols for running experiments, rather than to achieve the highest score on a hidden benchmark [6, 34]. By having organizers or volunteers re-run submissions in controlled settings using held-out test data, these challenges demonstrate a form of "reproducibility-as-a-service" that directly addresses the black-box evaluation problem seen in the BBO Challenge.

The ImageNet Large Scale Visual Recognition Challenge (ILSVRC), while historically significant, illustrates the dangers of leaderboard-focused competition design without adequate safeguards against overfitting. Subsequent analyses documented extensive leaderboard probing and adaptive overfitting, where teams submitted large numbers of model variants to optimize performance on a specific test set rather than developing genuinely generalizable methods [26]. This experience motivated more robust evaluation protocols in later competitions, including multiple held-out test sets, delayed result reporting, and explicit penalties for excessive submissions.

Traditional point-estimate leaderboards are also highly sensitive to random noise, which is especially problematic in stochastic black-box optimization. Robust ranking of solvers requires proper significance testing and resampling across multiple datasets to distinguish true algorithmic superiority from lucky initializations or particular test-set draws [5]. Recent initiatives like the ML Reproducibility Challenge have further refined best practices by encouraging environment snapshots (*e.g.,* Docker), requiring detailed documentation of computational requirements, and providing clear guidance on statistical testing and trial repetitions [23, 34]. Other work on AI competitions goes further by applying bootstrap-based resampling directly to competition run data to derive confidence intervals and statistically robust group rankings, revealing frequent ties and rank inversions compared to official leaderboards [9, 36].

3 Setup of Experiments

This section describes our evaluation protocol (Section 3.1), the optimizers and benchmarks used (Section 3.2 and Section 3.3), and the infrastructure on which all experiments were run (Section 3.4). In particular, we also discuss mechanisms deployed to ensure fair and reproducible comparison in Section 3.5.

3.1 Protocol and Relation to the Original Challenge

We closely follow the protocol of the original NeurIPS 2020 BBO Challenge [43]. Each algorithm was given a budget of 16 iterations with a batch size of 8, for a total of 128 objective evaluations per task. This mirrors the fixed-budget setting of the competition, where optimizers must balance exploration and exploitation under tight evaluation constraints. To approximate the historical behavior of the submissions, we used the earliest publicly available versions of the optimizers' repositories that were released immediately after the competition, thereby avoiding contamination from post-competition improvements.

The original challenge leaderboard was computed on a hidden set of `Bayesmark` benchmark problems that has not been released publicly [43] and could not be provided to us upon inquiry. As a result, we could not evaluate submissions on the exact benchmark problems used to determine the official rankings. Instead, we used public benchmark problems drawn from the same distribution and additional benchmark families (described in Section 3.3) to study how well the original leaderboard structure can be recovered in a broader setting.

3.2 Included Optimizers

We evaluated six methods that collectively represent the top competition entries: HEBO [4], JetBrains' space-decay optimizer [37], Optuna's MKSTuRBO [32], and the three un-ensembled components used by the NVIDIA RAPIDS team (RAPIDS-TuRBO, RAPIDS-SkOpt, RAPIDS-PySOT) [28]. Moreover, we included the strongest baseline used in the competition, TuRBO [8], and a simple baseline, Random Search. We configured each optimizer using the competition-era settings documented in their repositories wherever possible. Dependency drift since 2020 renders many of these codebases mutually incompatible in a single modern Python environment. To address this, we ran each optimizer in its own virtual environment as a subprocess and communicated with the central harness via JSON over `stdout`. This design ensures that legacy packages do not crash modern components, at the cost of introducing a thin inter-process communication layer that may cause minor deviations from the original monolithic execution.

Notably, the HEBO (Heteroscedastic and Evolutionary Bayesian Optimisation) algorithm [4] provides a comprehensive framework for handling non-stationary noise. Furthermore, the second-place entry by the NVIDIA RAPIDS team relied on a GPU-accelerated ensemble of black-box optimizers [28]. We did not reproduce the full RAPIDS ensemble, because the publicly available competition-era code did not provide a self-contained CPU-only execution path for the complete ensemble configuration, and because the ensemble construction procedure was tied to GPU-oriented infrastructure and hardware-specific optimizations that were not available in our reproduction environment. We therefore evaluated the individual publicly available RAPIDS components under the same CPU-only budget as the other optimizers. This does not reproduce the full RAPIDS submission, but it helps distinguish the standalone behavior of its constituent optimizers from the additional benefit of large-scale ensembling.

3.3 Benchmarks

Because the exact hidden benchmark problems from the NeurIPS 2020 BBO Challenge were unavailable and could not be provided upon request, we constructed a comprehensive 14-task evaluation suite that is as faithful as possible to the original domain while enabling broader generalization. We included public `Bayesmark` [44] benchmark problems from the same distribution used during challenge development and tuning, which serve as the closest proxy for the undisclosed test set. To test cross-domain behavior, we augmented these with four other benchmark families: continuous noiseless numerical functions from `BBOB` [20], surrogate empirical ML benchmarks from `LCBench` [48] and `HPO-B` [1], and structural engineering constrained objectives from `MechBench` [41]. Recent benchmarking studies highlight that tabular and surrogate-based benchmarks such as `HPO-B` and `LCBench` provide realistic search landscapes for HPO and are well-suited for reproducible evaluation [1, 7].

For aggregate ranking, we did not use all 14 tasks indiscriminately. The purpose of this filtering step was not to reduce computational cost, since all 14 tasks were already executed, but to avoid aggregating ranks from benchmarks on which the optimizers could not be statistically distinguished. We first ran each optimizer–benchmark pair for three independent trials and applied a non-parametric Friedman test over optimizers on each benchmark [5]. Benchmarks where the test failed to reject the null hypothesis of equal optimizer performance at $\alpha = 0.05$ were considered non-discriminative and were excluded from aggregate rank computation. The remaining 9 benchmarks, which exhibited statistically significant differences among optimizers, form our discriminative subset for average ranks and cross-dataset summaries.

3.4 Infrastructure

All experiments were executed on a SLURM-managed high-performance computing cluster, using identical CPU-only nodes (`c23ms` partition, 4 CPUs, 16 GB RAM). Each optimizer ran in its own Python virtual environment under these constraints, enforcing a uniform hardware budget across methods. This differs from the original challenge setting, where some submissions, notably the RAPIDS ensemble, relied on multi-GPU systems, such as NVIDIA DGX-1, for large-scale ensembling [28]. Our CPU-only setup should therefore be interpreted as a controlled reproduction design rather than as aiming to support a claim that GPU acceleration is scientifically irrelevant. It allowed us to compare the available optimizer implementations under a common and broadly accessible hardware budget, but it also changed the execution regime for methods whose original competitive performance depended on GPU-enabled pipelines or ensemble construction. As a result, our results measure reproducibility under a uniform CPU budget rather than a full reproduction of hardware-accelerated submissions, such as the complete RAPIDS ensemble.

3.5 Environment Isolation and Compatibility

To ensure fair and reproducible comparison across legacy submissions, we implemented a single orchestration layer in which each optimizer is launched in its own environment and interacts with benchmarks only through a standardized request/response interface. The optimizer process suggests a batch of points, the benchmark service evaluates them, and observations are returned in a common JSON format. This design decouples optimizer internals from benchmark implementation details and prevents dependency conflicts across methods.

Several competition-era repositories are not natively compatible with modern Python ecosystems. We therefore used subprocess-isolated execution with per-optimizer virtual environments and an inter-process communication bridge over `stdout`/JSON. This compatibility layer was necessary to run all methods under one harness and reduced failures caused by dependency conflicts across legacy environments. At the same time, execution-layer adaptations are themselves a reproducibility hazard: prior work identifies software environments, implementation details, platform differences, and experimental infrastructure as sources of irreproducibility [16, 22]. We therefore treat the subprocess bridge as a transparent validity consideration rather than as invisible implementation noise.

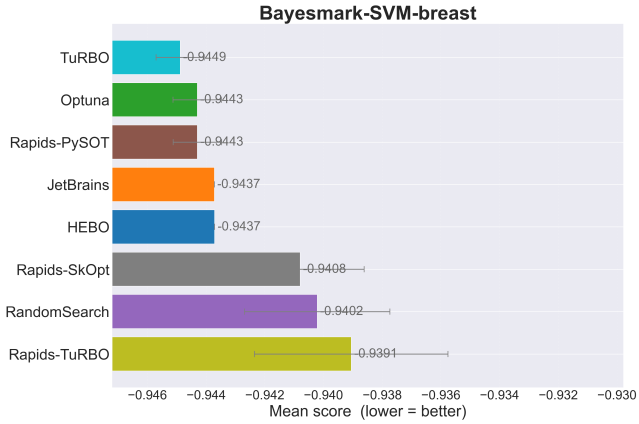


Figure 1: Per-optimizer mean score on Bayesmark-SVM-breast (retained benchmark). Error bars indicate standard error over three runs. Lower is better.

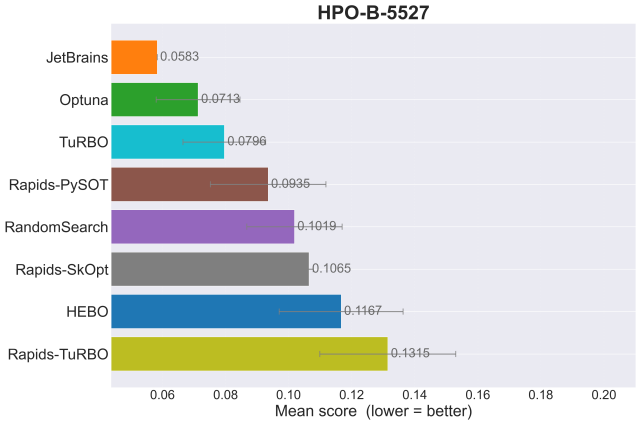


Figure 2: Per-optimizer mean score on HPO-B-5527 (retained tabular HPO benchmark). Error bars indicate standard error over three runs. Lower is better.

4 Results

We present the results of our experiments in three parts. We begin with the `Bayesmark` tasks in Section 4.1, which most closely relate to the original competition setting, questioning the reproducibility of the challenge on a similar dataset. Then, we evaluate the performance of the optimizers on out-of-domain benchmarks in Section 4.2, addressing the question of generalization to different types of functions. We also look at how the rankings change across benchmarks in Section 4.3. Finally, we aggregate the ranking and compare it to the original leaderboard in Section 4.4.

4.1 Reproducibility: Performance on `Bayesmark`

Figure 1 shows the mean performance of each optimizer on a representative retained `Bayesmark` benchmark problem, corresponding to HPO of a support vector machine (SVM) on the breast cancer dataset). These `Bayesmark` results, drawn from the same distribution as the original challenge benchmarks, represent the closest available proxy for the hidden competition test set. The ordering of methods on these tasks is broadly consistent with the original leaderboard, suggesting that the relative strengths of the top entries are at least partially recoverable on in-domain tasks. The top five performers remain the same as the original paper, but their ranking is has changed. HEBO, the winner of the original competition, is now ranked fifth. The best performing method on this task is TuRBO, the strongest available baseline of the original competition, introduced by the competition organizers.

4.2 Generalization: Out-of-Domain Performance

We show the mean performance of the optimizers on another HPO benchmark problem of the `HPO-B` set in Figure 2. While the ranking changes from the one on `Bayesmark`, we observe the stability of the top and bottom four optimizers within their respective groups, meaning that the strongest and weakest performers, respectively, stay relatively the same. On high-dimensional numerical optimization problem (`BBOB-40D`, Figure 3), performance spreads differ from

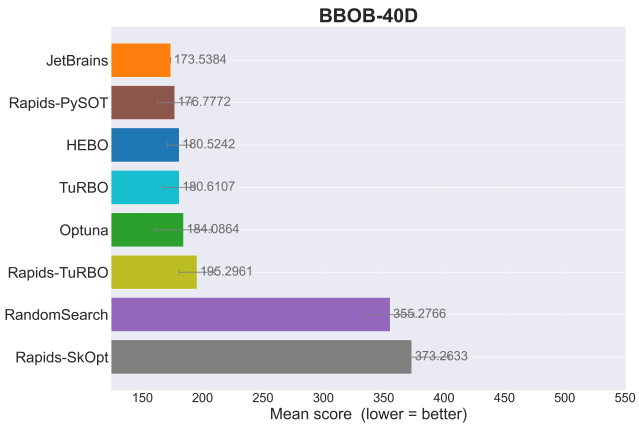


Figure 3: Per-optimizer mean score on BBOB-40D, highlighting high-dimensional numerical behavior. Lower is better.

those on `Bayesmark`, with some methods that excel on HPO tasks showing weaker behavior in purely numerical search spaces. For constrained engineering problems such as `MechBench-Spring` (Figure 4), optimizers that fail to handle large penalty regimes robustly can become either unstable or overly conservative following early poor evaluations. Across these CPU-only experiments, the individual RAPIDS components perform competitively but do not dominate; their average performance is in the middle of the pack, consistent with the hypothesis that much of the advantage of the original ensemble is derived from large-scale GPU ensembling rather than any single constituent optimizer [28].

4.3 Cross-Domain Rank Variation

To visualize how rankings change across benchmarks, Figure 5 presents a rank matrix over the 9 retained benchmarks. Each cell shows the average rank of a given optimizer on a particular task. The figure highlights pronounced cross-domain variation: methods

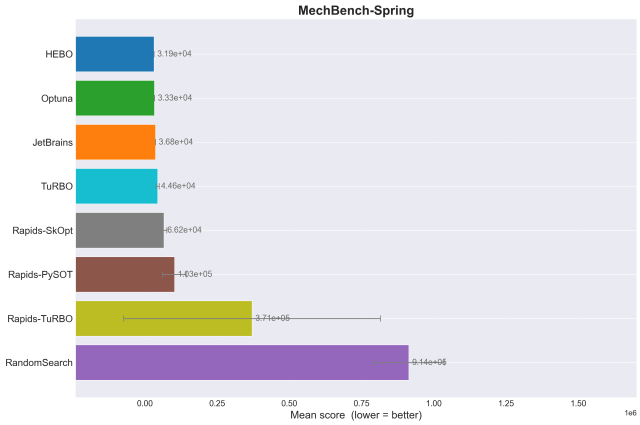


Figure 4: Per-optimizer mean score on MechBench-Spring, a constrained engineering problem. Lower is better.

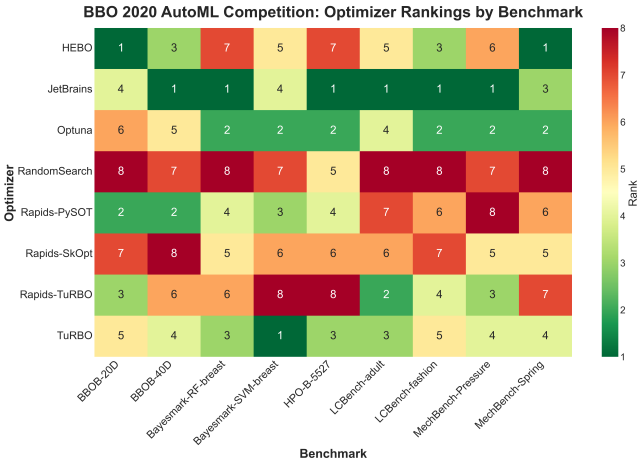


Figure 5: Per-benchmark rank matrix over the 9 retained benchmarks. Darker colors indicate better (lower) ranks. Strong cross-domain rank variation is visible, especially between HPO tasks and constrained engineering problems.

that are consistently strong on Bayesmark and tabular HPO benchmarks can drop several rank positions on MechBench or BBOB tasks, and vice versa. This domain sensitivity underscores that competition rankings derived from a single benchmark family may not generalize straightforwardly to other black-box optimization settings.

To examine when differences between optimizers emerge during the fixed-budget evaluation, Figure 6 shows the mean rank trajectories over the course of the budget. We observe that much of the rank separation between the leading methods is established within roughly the first third of the budget and is then largely preserved, rather than arising from late dramatic changes in ordering. This suggests that in low-budget competition settings, initialization behavior and early-region modeling are at least as important as asymptotic convergence properties.

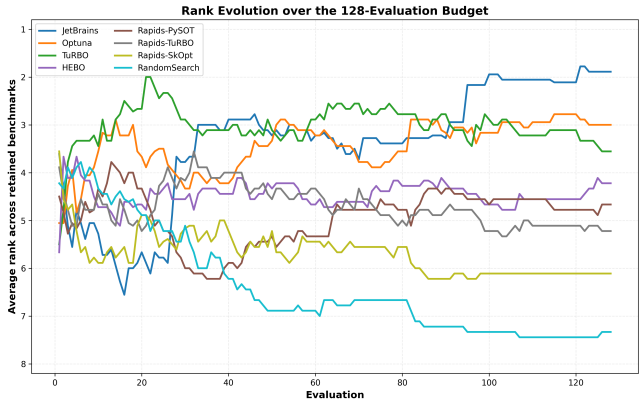


Figure 6: Mean rank trajectories of the optimizers over the 128-evaluation budget, averaged across the retained benchmarks. Lower is better. Most of the rank separation between the leading methods emerges within roughly the first third of the budget and is then largely preserved.

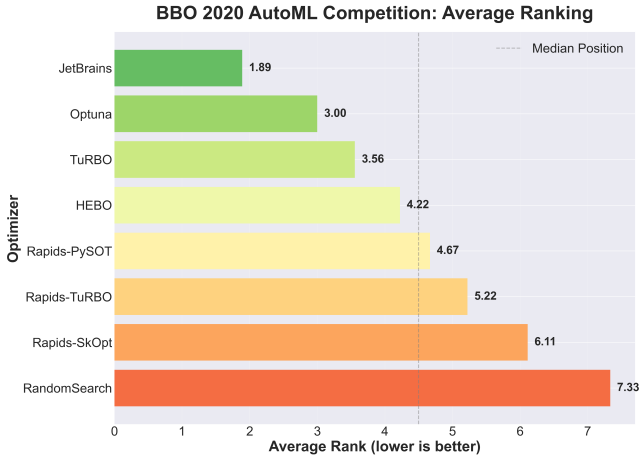


Figure 7: Average rank across the statistically discriminative 9-benchmark subset. Error bars show 95% bootstrap confidence intervals over benchmarks. Lower is better.

4.4 Aggregate Ranking and Comparison to the Official Leaderboard

Using only the 9 benchmarks that pass the Friedman test for discriminative power, we compute the average rank of each optimizer across tasks. The resulting ordering is: JetBrains (1.89), Optuna (3.00), TuRBO (3.56), HEBO (4.22), RAPIDS-PySOT (4.67), RAPIDS-TuRBO (5.22), RAPIDS-SkOpt (6.11), and Random Search (7.33). Thus, in our cross-domain reproduction JetBrains and Optuna outperform HEBO *on average* across the retained benchmark subset, even though HEBO remains strong on several HPO-style tasks. Figure 7 summarizes these averages together with confidence intervals derived from bootstrap resampling over benchmarks, following recommendations for robust competition ranking [5, 9].

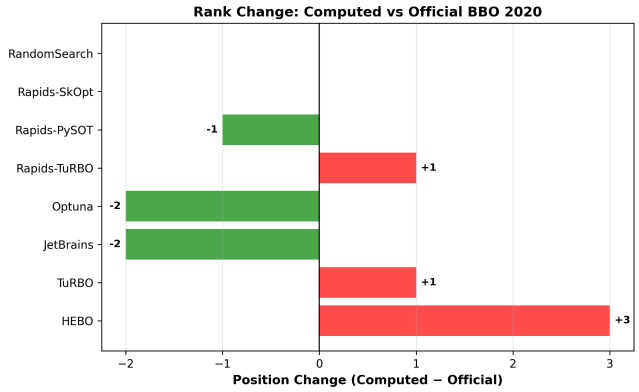


Figure 8: Rank-position change (reproduced minus official) for each optimizer relative to the NeurIPS 2020 BBO standings. Negative values indicate an improved position in our reproduction.

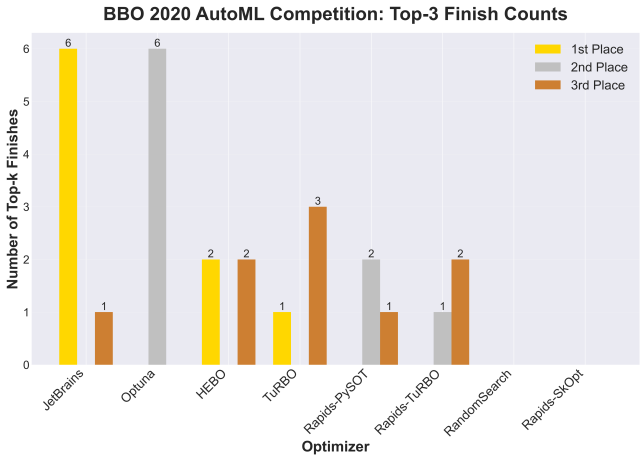


Figure 9: Top-k finish counts (1st, 2nd, 3rd) across the 9 retained benchmarks. Bars indicate how often each optimizer appears among the leading tier.

To quantify how closely our reproduction matches the original challenge ordering, we compute the Spearman rank correlation between the official BBO standings [43] and our reproduced aggregate ranks. On the 9-benchmark subset, the correlation is positive and statistically significant ($\rho = 0.714$, $p = 0.0465$), indicating partial structural agreement rather than arbitrary reshuffling. Figure 8 shows per-optimizer rank differences (reproduced minus official), making clear where methods move up or down relative to their original positions.

Finally, Figure 9 reports the number of first-, second-, and third-place counts for each optimizer across the discriminative benchmarks. These top-k counts complement average ranks by showing how often each method appears among the leading tier, rather than only its mean position. Together, the aggregate metrics and visualizations support the conclusion that broad competitive tiers are reproducible, while exact rank identities, especially in the middle of the table, are not.

5 Discussion

We articulate our discussion around the three questions formulated in the introduction: on reproducibility in Section 5.1, on generalization in Section 5.2, and on competition design in Section 5.3. Last, we critically discuss limitations of our approach in Section 5.4.

5.1 Can the Competition Be Reproduced?

Based on the information provided on the challenge website and the content of the published report, reproduction of the NeurIPS 2020 BBO Challenge is currently impossible. Despite our best efforts, due to the unavailability of the objective functions used, and severe dependency issues in the original software stack, we were unable to reproduce the original competition results, even on similar benchmark problems. Furthermore, hardware dependence renders the 2nd place approach non-reproducible in typical HPO benchmarking setups: requiring a DGX-1 with 8 GPUs for exhaustive ensembling restricts independent verification and aligns poorly with "Green AI" resource awareness constraints [38].

5.2 Do Rankings Generalize?

Overall, the relative ranking of methods exhibits partial stability across domains, but domain-specific inductive biases (*e.g.,* categorical *vs* numerical spaces, constraints) cause substantial reshuffling.Algorithms tuned aggressively for categorical hyperparameter spaces (`Bayesmark`) frequently crashed or underperformed on continuous, highly constrained engineering problems (`MechBench`). In the aggregate ranking from our experiments, JetBrains and Optuna achieve the best average ranks across the retained 9-benchmark subset (Figure 7), while HEBO remains a robust method in the sense of delivering relatively consistent competitive performance across multiple families rather than specializing strongly for a single type of search space. We do note that, while HEBO does exhibit performance lower than Random Search for some HPO problems (Figure 2), it is top ranked for constrained optimization problems (Figure 4).

5.3 Recommendations for Future Competitions

The findings above, *i.e.,* partial ranking recovery, domain-dependent reshuffling, and infrastructure-driven performance shifts, point to concrete improvements for future competition design. Drawing on these lessons and our empirical results, we propose several recommendations for future black-box optimization competitions.

First, competitions should adopt a hybrid evaluation model that combines immediate performance assessment with post-competition verification. Organizers should maintain control over the final evaluation infrastructure while providing participants with representative proxy benchmarks during the competition phase. After the competition concludes, the complete evaluation suite—including hidden objectives and exact software environments—should be archived in permanent repositories (*e.g.,* Zenodo) or institutional archives to support independent replication.

Second, organizers should implement hardware-aware evaluation tiers that acknowledge resource constraints while maintaining

scientific rigor. Rather than allowing essentially unlimited computational resources, competitions can establish multiple tracks: one CPU-only evaluation track accessible to most researchers, and optional GPU-accelerated tracks with clearly documented hardware requirements and energy footprints. This approach preserves the scientific value of resource-intensive methods while ensuring baseline reproducibility for the broader community [38].

Third, competitions should integrate ensemble-aware scoring that distinguishes between fundamental algorithmic innovation and brute-force computational ensembling. The second place achieved in the BBO Challenge by the NVIDIA RAPIDS ensemble, while technically impressive, primarily demonstrates the power of large-scale GPU resources and ensemble search rather than a single, novel optimization algorithm [28]. Our analysis of the RAPIDS components supports the hypothesis that ensemble-focused leaderboards can obscure progress on individual algorithmic approaches. This problem is not new: the SAT competition community faced a similar challenge with the rise of portfolio-based solvers such as SATzilla [46], and responded by introducing separate solver and portfolio tracks to preserve visibility of individual algorithmic contributions [24]. Future competitions could follow this precedent by implementing scoring systems that reward per-algorithm efficiency or require ablation studies demonstrating the contribution of individual ensemble components.

Finally, we recommend explicit post-competition sustainability requirements. Winning teams could be required to maintain their codebases for a minimum period or to integrate their methods into established frameworks such as Optuna or Ax, possibly under the guidance of organizers or maintainers. This addresses the observed pattern where competition-winning algorithms often become abandoned research artifacts rather than sustainable, community-supported tools [4, 19].

5.4 Limitations

An important observation is the trajectory of these methods post-competition. Algorithms like HEBO and TuRBO were aggressively developed; TuRBO became integrated into massive frameworks like Meta's `Ax` and `Optuna`, while HEBO transitioned into a standalone library. Conversely, other entries remained static research artifacts without subsequent updates. This highlights an often overlooked aspect of competition outcomes: community adoption and sustainability. While code availability requirements have improved, post-publication verification remains anecdotal and often sketchy. When running historical repositories "as they were," users frequently encounter bugs that were later fixed in modern integrations. This is a discrepancy that can materially affect experimental outcomes, potentially yielding results that differ from those originally reported in the competition.

The reproduction effort required approximately three months of work over a period of three months. The largest sources of effort were reconstructing legacy software environments, implementing the subprocess-based execution bridge, resolving optimizer-specific failure modes, and re-running experiments under a unified benchmark harness. This effort estimate is itself part of the reproducibility finding: even when code is publicly available, reproducing competition submissions several years later can require substantial engineering work.

A second limitation concerns statistical power: with three trials per optimizer-benchmark pair, uncertainty in rank estimates is a potentially serious factor for methods with similar performance. We therefore interpret differences between the performance of the optimizers ranked in the middle tier with caution, and as directional evidence rather than definitive separation. This is precisely why we include non-parametric benchmark filtering and rank-based aggregation, and why we avoid claims of exact leaderboard recreation.

A third limitation is protocol adaptation itself. The subprocess bridge and environment isolation are methodologically necessary to run legacy stacks, but they may induce execution-level differences compared to original challenge infrastructure, the magnitude of which is difficult to bound precisely. Our position is that transparent, reproducible approximation is scientifically preferable to unverifiable exactness claims.

6 Conclusions

We have presented a systematic reproduction study of the NeurIPS 2020 Black-Box Optimization Challenge, re-evaluating 5 top-performing entries and a Random Search baseline across 14 tasks from 5 benchmark families. Our study addressed three research questions. Regarding reproducibility, we found that the original competition leaderboard cannot be exactly replicated—the hidden test objectives are unavailable, and dependency drift renders the original software stacks fragile—but broad performance tiers are recoverable on public Bayesmark tasks drawn from the same distribution, yielding a statistically significant rank correlation ($\rho = 0.714$) with the official standings. Regarding generalization, our cross-domain evaluation revealed that rankings are partially stable across benchmark families, though domain-specific factors (*e.g.*, constraint handling, search-space type) cause substantial reshuffling beyond the top positions. Regarding competition design, we distilled concrete recommendations—hybrid evaluation models with post-competition archival, hardware-aware evaluation tiers, ensemble-aware scoring inspired by precedents such as the SAT competition, and explicit sustainability requirements—aimed at maximizing the long-term scientific value of future competitions.

The central takeaway is that the most achievable and scientifically meaningful form of long-term competition reproducibility is *structural*: recovering a coherent performance hierarchy and consistent domain-dependent strengths and weaknesses, rather than matching exact numerical scores. We hope this case study encourages competition organizers to prioritize transparent archival, standardized reporting, and robust statistical evaluation, so that the scientific contributions of competitions endure well beyond the announcement of their winners.

References

[1] Sebastian Pineda Arango, Hadi S. Jomaa, Martin Wistuba, and Josif Grabocka. 2021. HPO-B: A Large-Scale Reproducible Benchmark for Black-Box HPO based on OpenML. In Proceedings of the 35th International Conference on Neural Information Processing Systems (NeurIPS'21) Track on Datasets and Benchmarks.

[2] Xavier Bouthillier, César Laurent, and Pascal Vincent. 2019. Unreproducible Research is Reproducible. In Proceedings of the 36th International Conference

on Machine Learning (ICML'19). 725–734.
[3] Xavier Bouthillier and Gaël Varoquaux. 2020. Survey of machine learning experimental methods at NeurIPS 2019 and ICLR 2020. Research Report hal-02447823. Inria Saclay Ile de France.
[4] Alexander I. Cowen-Rivers, Wenlong Lyu, Rasul Tutunov, Zhi Wang, Antoine Grosnit, Ryan-Rhys Griffiths, Alexandre Max Maraval, Jianye Hao, Jun Wang, Jan Peters, and Haitham Bou-Ammar. 2022. HEBO: An Empirical Study of Assumptions in Bayesian Optimisation. Journal of Artificial Intelligence Research 74 (2022), 1269–1349.
[5] Janez Demšar. 2006. Statistical comparisons of classifiers over multiple data sets. Journal of Machine Learning Research 7 (2006), 1–30.
[6] Jesse Dodge, Suchin Gururangan, Dallas Card, Roy Schwartz, and Noah A Smith. 2019. Show your work: Improved reporting of experimental results. arXiv preprint arXiv:1909.03004 (2019).
[7] Katharina Eggensperger, Philipp Müller, Neeratyoy Mallik, Matthias Feurer, René Sass, Aaron Klein, Noor Awad, Marius Lindauer, and Frank Hutter. 2021. HPOBench: A Collection of Reproducible Multi-Fidelity Benchmark Problems for HPO. In Proceedings of the 35th International Conference on Neural Information Processing Systems (NeurIPS'21) Track on Datasets and Benchmarks.
[8] David Eriksson, Michael Pearce, Jacob Gardner, Ryan D Turner, and Matthias Poloczek. 2019. Scalable global optimization via local Bayesian optimization. In Proceedings of the 33rd International Conference on Neural Information Processing Systems (NeurIPS'19). 5496–5507.
[9] Chris Fawcett, Mauro Vallati, Holger H. Hoos, and Alfonso E. Gerevini. 2023. Competitions in AI: Robustly ranking solvers using statistical resampling. arXiv preprint arXiv:2308.05062 (2023).
[10] Alexander I. J. Forrester, András Sóbester, and Andy J. Keane. 2008. Engineering Design via Surrogate Modelling - A Practical Guide.
[11] Peter I Frazier. 2018. A Tutorial on Bayesian Optimization. arXiv preprint arXiv:1807.02811 (2018).
[12] Roman Garnett. 2023. Bayesian Optimization. Cambridge University Press.
[13] Javier González, Zhenwen Dai, Philipp Hennig, and Neil Lawrence. 2016. Batch Bayesian Optimization via Local Penalization. In Proceedings of the 19th International Conference on Artificial Intelligence and Statistics (AISTATS'16).
[14] Odd Erik Gundersen. 2021. The fundamental principles of reproducibility. Philosophical Transactions of the Royal Society A 379, 2197 (2021), 202–210.
[15] Odd Erik Gundersen, Odd Cappelen, Martin Mølnå, and Nicklas Grimstad Nilsen. 2025. The unreasonable effectiveness of open science in AI: A replication study. In Proceedings of the 39th AAAI Conference on Artificial Intelligence (AAAI'25), Vol. 39. 26211–26219.
[16] Odd Erik Gundersen, Kevin Coakley, Christine Kirkpatrick, and Yolanda Gil. 2022. Sources of irreproducibility in machine learning: A review. arXiv preprint arXiv:2204.07610 (2022).
[17] Odd Erik Gundersen, Yolanda Gil, and David W Aha. 2018. On reproducible AI: Towards reproducible research, open science, and digital scholarship in AI publications. AI Magazine 39, 3 (2018), 56–68.
[18] Odd Erik Gundersen, Malte Helmert, and Holger Hoos. 2024. Improving reproducibility in AI research: Four mechanisms adopted by JAIR. Journal of Artificial Intelligence Research 81 (2024), 1019–1041.
[19] Isabelle Guyon, Hugo Jair Escalante, Sergio Escalera, Damir Jajetic, Ben Lloyd, Núria Macià, William Romano, Michèle Sebag, Alexander Statnikov, Stéphane Treguer, et al. 2019. Analysis of the AutoML challenge series 2015–2018. Springer International Publishing, 177–204.
[20] Nikolaus Hansen, Steffen Finck, Raymond Ros, and Anne Auger. 2009. Real-Parameter Black-Box Optimization Benchmarking 2009: Noisy Functions Definitions. Research Report RR-6829. INRIA.
[21] Nikolaus Hansen and Andreas Ostermeier. 2001. Completely Derandomized Self-Adaptation in Evolution Strategies. Evolutionary Computation 9, 2 (2001), 159–195.
[22] Peter Henderson, Riashat Islam, Philip Bachman, Joelle Pineau, Doina Precup, and David Meger. 2018. Deep reinforcement learning that matters. In Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 32.
[23] Brandon Houghton, Stephanie Milani, Nicholay Topin, William Guss, Katja Hofmann, Diego Perez-Liebana, Manuela Veloso, and Ruslan Salakhutdinov. 2020. Guaranteeing Reproducibility in Deep Learning Competitions. arXiv preprint arXiv:arXiv:2005.06041 (2020).
[24] Matti Järvisalo, Daniel Le Berre, Olivier Roussel, and Laurent Simon. 2012. The International SAT Solver Competitions. AI Magazine 33, 1 (2012), 89–92.
[25] Sayash Kapoor, Emily M Cantrell, Kenny Peng, Thanh Hien Pham, Christopher A Bail, Odd Erik Gundersen, Jake M Hofman, Jessica Hullman, Michael A Lones, Momin M Malik, et al. 2024. REFORMS: Consensus-based Recommendations for Machine-learning-based Science. Science Advances 10, 18 (2024).
[26] Sayash Kapoor and Arvind Narayanan. 2023. Leakage and the reproducibility crisis in machine-learning-based science. Patterns 4, 9 (2023).
[27] Harold Kushner. 1964. A new method of locating the maximum point of an arbitrary multipeak curve in the presence of noise. Journal of Fluids Engineering (1964), 97–106.
[28] Jiwei Liu, Bojan Tunguz, and Gilberto Titericz. 2020. GPU-Accelerated Exhaustive Search for Optimal Ensemble of Black-Box Optimization Algorithms. arXiv preprint arXiv:2012.04201 (2020).
[29] Manuel López-Ibáñez, Juergen Branke, and Luís Paquete. 2021. Reproducibility in Evolutionary Computation. ACM Transactions on Evolutionary Learning and Optimization 1, 4 (2021).
[30] Jonas Močkus. 1975. On Bayesian methods for seeking the extremum. In Proceedings of the Technical Conference on Optimization Techniques (IFIP). 400–404.
[31] Jonas Močkus. 2012. Bayesian approach to global optimization: theory and applications. Springer Dordrecht.
[32] Optuna. 2020. MkSTurBO. https://github.com/optuna/bboc-optuna-developers
[33] Hung Viet Pham, Shangshu Qian, Jiannan Wang, Thibaud Lutellier, Jonathan Rosenthal, Lin Tan, Yaoliang Yu, and Nachiappan Nagappan. 2020. Problems and opportunities in training deep learning software systems: An analysis of variance. In Proceedings of the 35th IEEE/ACM International Conference on Automated Software Engineering (ASE'20). 771–783.
[34] Joelle Pineau, Philippe Vincent-Lamarre, Koustuv Sinha, Vincent Larivière, Alina Beygelzimer, Florence d'Alché Buc, Emily Fox, and Hugo Larochelle. 2021. Improving reproducibility in machine learning research. Journal of Machine Learning Research 22 (2021), 1–20.
[35] Rebecca Roelofs, Vaishaal Shankar, Benjamin Recht, Sara Fridovich-Keil, Moritz Hardt, John Miller, and Ludwig Schmidt. 2019. A Meta-Analysis of Overfitting in Machine Learning. In Proceedings of the 33rd International Conference on Neural Information Processing Systems (NeurIPS'19). 9179–9189.
[36] Jeroen Rook, Holger H. Hoos, and Heike Trautmann. 2024. Multi-objective Ranking using Bootstrap Resampling. In Proceedings of the Genetic and Evolutionary Computation Conference (GECCO'24) Companion. 155–158.
[37] Mikita Sazanovich, Anastasiya Nikolskaya, Yury Belousov, and Aleksei Shpilman. 2020. Solving Black-Box Optimization Challenge via Learning Search Space Partition for Local Bayesian Optimization. arXiv preprint arXiv:2012.10335 (2020).
[38] Roy Schwartz, Jesse Dodge, Noah A Smith, and Oren Etzioni. 2020. Green AI. Commun. ACM 63, 12 (2020), 54–63.
[39] Jasper Snoek, Hugo Larochelle, and Ryan P. Adams. 2012. Practical Bayesian optimization of machine learning algorithms. In Proceedings of the 26th International Conference on Neural Information Processing Systems (NIPS'12), Vol. 2. 2951–2959.
[40] Jasper Snoek, Alex Wiltschko, and Ali Rahimi. 2018. Winner's curse? On pace, progress, and empirical rigor. In Proceedings of the International Conference on Learning Representations (ICLR'18) Workshop.
[41] Ryoji Tanabe and Hisao Ishibuchi. 2020. An Easy-to-use Real-world Multi-objective Optimization Problem Suite. arXiv preprint arXiv:2009.12867 (2020).
[42] William R. Thompson. 1933. On the Likelihood that One Unknown Probability Exceeds Another in View of the Evidence of Two Samples. Biometrika 25, 3/4 (1933), 285–294.
[43] Ryan Turner, David Eriksson, Michael McCourt, Juha Kiili, Eero Laaksonen, Zhen Xu, and Isabelle Guyon. 2021. Bayesian Optimization is Superior to Random Search for Machine Learning Hyperparameter Tuning: Analysis of the Black-Box Optimization Challenge 2020. In Proceedings of the 34th International Conference on Neural Information Processing Systems (NeurIPS'20) Competition and Demonstration Track. 3–26.
[44] Uber. 2019. Bayesmark. https://github.com/uber/bayesmark
[45] Gaël Varoquaux and Veronika Cheplygina. 2022. Machine learning for medical imaging: methodological failures and recommendations for the future. NPJ Digital Medicine 5, 48 (2022).
[46] Lin Xu, Frank Hutter, Holger H. Hoos, and Kevin Leyton-Brown. 2008. SATzilla: Portfolio-based Algorithm Selection for SAT. Journal of Artificial Intelligence Research 32 (2008), 565–606.
[47] Donglin Zhuang, Xingyao Zhang, Shuaiwen Leon Song, and Sara Hooker. 2022. Randomness in neural network training: Characterizing the impact of tooling. In Proceedings of the 5th Machine Learning and Systems Conference (MLSys'22).
[48] Lucas Zimmer, Marius Lindauer, and Frank Hutter. 2021. Auto-PyTorch Tabular: Multi-Fidelity MetaLearning for Efficient and Robust AutoDL. IEEE Transactions on Pattern Analysis and Machine Intelligence 43, 9 (2021), 3079–3090.