Beyond Accuracy: A Cost-Aware Perspective on Hyperparameter Optimization — A Survey and Gap Analysis

Venkata Raghu Vara Prasad Nori
Independent Researcher

ABSTRACT
Hyperparameter optimization (HPO) methods are usually compared by the best score they reach after a fixed number of trials. That number says nothing about how quickly a method improves as time, money or energy is spent. This paper surveys HPO methods with cost in mind, points out that the score-versus-cost record of a search is rarely reported, and suggests reporting it as a Performance-per-Cost Curve (PpCC) with a few summary numbers. To see whether this matters, four methods (grid search, random search, Gaussian-process Bayesian optimization and the tree-structured Parzen estimator) were run on three classification tasks, ten times each. The two Bayesian methods got the highest cross-validation scores, and the tree-structured Parzen estimator beat grid and random search by a clear margin on two of the three datasets. On held-out test data, though, the differences were small and mostly within chance. On one dataset the ranking of the methods changed depending on the budget. Equal numbers of trials also did not mean equal cost: total search time differed by up to about 13% between methods. The extra computation of the Bayesian methods was under 1% of total cost here. The results argue for reporting the whole score-versus-cost record of a search and not just its final score.

General Terms
Machine Learning, Algorithms, Performance, Measurement

Keywords
Hyperparameter optimization, AutoML, Bayesian optimization, cost-aware machine learning, model selection, benchmarking

1. INTRODUCTION
Hyperparameters are the settings of a learning method that are chosen before training and are not learned from the data. Examples are the learning rate, the strength of regularization, the depth of a tree, or the shape of a neural network. They are usually chosen by an outer search that tries many settings and scores each one on held-out data, most often by cross-validation [1], [2].

Choosing them well matters. A poorly tuned model and a well tuned one can differ by as much as two different learning algorithms do [3]. HPO has therefore grown into a research area of its own and is closely tied to automated machine learning (AutoML). The search strategies in use range from grid and random search [4] to Bayesian optimization [5], to multi-fidelity methods such as Hyperband [6] and BOHB [7], and to methods that carry over what was learned on earlier tasks [8].

Most HPO comparisons ask one question: with a fixed budget of N trials (or T seconds), which method ends with the best validation score? That question folds two separate things into one. One is the best score reachable with a generous budget. The other is how fast the score improves as cost grows. The second often matters more in practice, particularly when compute, energy or waiting time is the real limit, as it increasingly is for large models [9].

The paper has three parts. Section 2 surveys HPO methods, grouped by how each one deals with the cost of trying a setting. Section 3 describes the reporting gap: the score reached per unit of cost is not reported in any standard form, which affects decisions and reproducibility. Section 4 tests how much this matters, using four methods, three datasets and ten repetitions each, and reports the results with the proposed Performance-per-Cost Curve (PpCC) and four summary measures. Sections 5 to 7 cover what follows from this, open questions and conclusions.

2. A COST-ORIENTED SURVEY OF HPO METHODS
The survey is organized by how each method relates to evaluation cost, that is, whether and how the design of the method takes into account what it costs to learn something new about a setting. This differs from the usual grouping by type of search space.

2.1 Cost-agnostic search: grid and random search
Grid search tries every combination of a fixed list of values for each hyperparameter [4]. Its cost grows exponentially with the number of hyperparameters, and it spends the same effort everywhere, whatever it has already seen. Random search draws settings independently from a chosen distribution. Bergstra and Bengio [4] showed that it beats grid search at equal budget when only a few hyperparameters really matter. Neither method uses the cost or the outcome of earlier trials to choose the next one.

2.2 Sequential model-based optimization
Bayesian optimization (BO) fits a statistical model of the score as a function of the hyperparameters and uses an acquisition rule such as Expected Improvement (EI) to pick the next setting. The model is commonly a Gaussian process (GP) [5], [10], [11], or, for larger or conditional spaces, the tree-structured Parzen estimator (TPE) [12] or a random-forest model (SMAC) [13]. BO adapts to outcomes, but in its standard form it does not adapt to cost: the acquisition rule ignores that some settings are much slower to evaluate than others (for example, a forest with many trees).

Cost-aware versions exist. One divides the acquisition value by an estimated cost so that cheap but informative trials are preferred [11], [14]. These versions are still uncommon and are rarely the default in widely used libraries.

2.3 Multi-fidelity and bandit-based methods

Hyperband [6] and its model-based extension BOHB [7] build cost into the method. They give a small budget (few training epochs or a small data subset) to many settings and then give larger budgets to the most promising ones, following a successive-halving schedule [15]. These are the closest existing relatives of a cost-aware search. Even so, the usual reported result is still the best score found within a total budget B, which reduces the cost dimension to one number.

2.4 Meta-learning and transfer

Another group of methods uses what was learned on earlier tasks to start or guide a new search. Examples are learned starting points for BO [8], cheap surrogate benchmarks [16], and early stopping of poor runs by extrapolating learning curves [17]. They lower the effective cost by sharing it across tasks. The gains are reported either as the final score reached within a budget or as the budget needed to reach a fixed score. These are two related but different views, and the literature does not always separate them.

2.5 Summary

Table 1 lists the method families along three questions: does the method adapt to observed outcomes, does it adapt to cost, and what result is usually reported.

Table 1. Method families by outcome and cost adaptivity

Method family	Adapts to outcomes?	Adapts to cost?	Typical reported result
Grid search	No	No	Best score over the full grid
Random search	No	No	Best score over N trials
Bayesian optimization (GP, TPE, SMAC)	Yes	Rarely	Best score over N trials or wall-clock time T
Hyperband / BOHB	Yes	Yes (in the schedule)	Best score within total budget B
Meta-learning / transfer	Yes	Indirectly	Speed-up to reach a target score

Table 1 shows a pattern. The more a method uses cost internally, the more cost-related wording appears in how results are reported ("within budget B", "speed-up"). Still, none of these conventions gives the full record of score against cost in a standard form that can be compared across studies. Section 3 takes up this gap.

3. GAP ANALYSIS: THE MISSING PERFORMANCE-PER-COST AXIS

The gap shows up in three ways.

3.1 "Best achievable" versus "best per unit of cost"

Suppose a paper says Method X reached 97.5% accuracy after 30 trials and Method Y reached 96.7%. Most readers will take X to be better. That is only fair if the 30 trials cost about the same for both, and often they do not. Bayesian methods spend time fitting their model and choosing the next setting, and that time grows as observations pile up [5]; grid and random search spend none. Trials also differ in cost because settings differ: a forest with 200 trees takes far longer to train than one with 10. A method that hits 97.4% by trial 7 and then stalls can look identical in a table to one that only reaches 97.5% at trial 30, yet the first is better at any budget below the cost of 30 trials.

3.2 No standard reporting of the search trajectory

In deep learning, curves of score against training epoch are standard. The HPO counterpart, score against the cumulative cost of the search, is rarely a main result. Where such curves do appear (for example in the Hyperband and BOHB papers [6], [7]), the horizontal axis is usually the number of settings tried and not a cost such as seconds, floating-point operations or money. That brings back the problem from Section 3.1 whenever cost per setting varies, which is nearly always.

3.3 Practical consequences

Three consequences follow. A practitioner who picks a method from published "best score" tables may end up with one that only leads at budgets far larger than the one available. Comparisons are also harder to reproduce when total cost is neither controlled nor reported. Wall-clock times cannot be compared across papers because hardware differs, but comparing score per cost on the same hardware inside one paper is possible and is simply not done. Finally, as models and training runs grow [9], the search itself takes a large share of a project's compute and of its environmental footprint [18]. At that point the path of the search matters, and not only where it ends.

3.4 The proposed Performance-per-Cost Curve and summary measures

This paper proposes that HPO comparisons report a Performance-per-Cost Curve (PpCC) as a standard figure. For one run, let $c(i)$ be the cumulative cost after trial i, counting both the time to evaluate the setting and the time the optimizer needed to propose it, and let $b(i)$ be the best validation score found up to trial i. The PpCC is the step curve of b against c. Cost may be measured in seconds on stated hardware, or in floating-point operations when independence from hardware is wanted. This follows the idea of time-to-accuracy reporting in training benchmarks [19], [20]. Four numbers summarize the curve:

(a) Score at a budget fraction. The best score reached at 25%, 50%, 75% and 100% of a reference cost T, set here to the median total cost of the four methods on the same dataset.

(b) Time to 99% of the best. The cost at which the best-so-far score first reaches 99% of the best score found by any method in the same repetition. A shared target is used so that methods are compared on the same goal.

(c) PpCC area. The area under the PpCC divided by the best score and by the longest total cost, so a value close to 1 means good scores were found early.

(d) Overhead share. The optimizer's own time as a percentage of total cost.

These are reported next to the usual final score, not instead of it.

4. EXPERIMENTAL STUDY

The experiment shows how the usual summary and the cost-aware summary can lead to different conclusions. It is a small study with a fast model and a four-dimensional search space. It is not a broad benchmark, and it is not intended to show that one method is better in general.

4.1 Setup

4.1.1 Datasets and model

Three classification datasets were used (Table 2): Digits (1,797 samples, 64 features, 10 classes) and Breast Cancer (569 samples, 30 features, 2 classes), both from scikit-learn [21], and a synthetic 3-class problem with 2,000 samples and 20 features (10 informative, 4 redundant) made with scikit-learn's *make_classification*. In every repetition the data were split 75/25 into a training and a held-out test set with stratification. The model is a Random Forest [22] with four tuned hyperparameters: number of trees (10–200), maximum depth (2–30), minimum samples to split (2–20), and the fraction of features tried at each split (0.1–1.0). Each setting was scored by 3-fold cross-validation accuracy on the training set.

4.1.2 Methods

Grid search used five values per hyperparameter (625 combinations). Because the budget is only 30 trials, the grid was shuffled with the repetition's seed and the first 30 combinations were used. This is a truncated grid, which is how grid search is normally run under a small budget, and it is not the full grid. **Random search** drew each setting uniformly from the ranges above. **GP-EI** used a Gaussian process with a Matérn kernel ($\nu = 2.5$), 5 random starting points, and Expected Improvement maximized over 200 random candidate points at every step. **TPE** used the implementation in Optuna [23] with 5 random starting trials. Every method had a budget of 30 trials.

4.1.3 Cost accounting

Cost is wall-clock time. For each trial it is the sum of the time to train and cross-validate the setting and the time the optimizer needed to propose it (model fitting and acquisition for GP-EI; sampler time for TPE). Counting evaluation time alone would hide the overhead of the Bayesian methods. All runs used one thread per process.

4.1.4 Protocol and statistics

Each dataset was run with ten repetitions, using ten different seeds. The seed fixes the train/test split, the model's random state and every method's random numbers, so the repetitions differ in split as well as in search. After the 30 trials, the best setting found was refit on the full training set and scored on the test set. Means are reported with 95% confidence intervals based on the t distribution over the ten repetitions. Methods were compared in pairs with the Wilcoxon signed-rank test [24] across repetitions; with ten repetitions the smallest possible two-sided p is 0.002. The p values are not corrected for multiple comparisons; a Bonferroni threshold for the six pairs per dataset and measure is 0.0083.

The experiments ran on a shared 2-core Intel Xeon (2.1 GHz) virtual machine with Python 3.11, scikit-learn 1.8.0 and Optuna 5.0.0. Two experiment processes ran at the same time, so absolute times are specific to this machine and are useful only for comparing methods with each other.

Table 2. Datasets and reference cost *T*

Dataset	Samples	Features	Classes	Test set size	Reference cost T (s)
Digits	1,797	64	10	450	43.3
Breast Cancer	569	30	2	143	19.3
Synthetic	2,000	20	3	500	56.3

4.2 Results

4.2.1 The usual summary

Table 3 gives the usual summary (best cross-validation score after 30 trials, and test accuracy) next to the cost-aware columns. On cross-validation score, TPE was highest on all three datasets (Digits 0.9709, Breast Cancer 0.9587, Synthetic 0.8679), followed by GP-EI, and grid and random search were lowest. On Digits and Synthetic, TPE was significantly better than every other method (all $p \leq 0.0078$, which is below the Bonferroni threshold; Table 6). On Breast Cancer the differences were small (about 0.3 points between the best and the worst method) and only TPE versus random search reached $p = 0.047$, which would not survive a correction for multiple comparisons. A reader who looked only at this part of Table 3 would conclude that TPE is the best method.

Table 3. Usual and cost-aware summaries, mean ± 95% CI over ten repetitions (30 trials per method)

Dataset	Method	Best CV accuracy	Test accuracy	Total cost (s)	Overhead (s)	Time to 99% of best (s)	PpCC area
Digits	Grid	0.9641 ± 0.0041	0.9749 ± 0.0033	42.2	0.00	14.5	0.948
	Random	0.9633 ± 0.0036	0.9671 ± 0.0073	45.7	0.00	15.5	0.962
	GP-EI	0.9668 ± 0.0039	0.9751 ± 0.0039	42.9	0.18	10.5	0.960
	TPE	0.9709 ± 0.0024	0.9753 ± 0.0045	43.0	0.13	8.8	0.967
Breast Cancer	Grid	0.9561 ± 0.0043	0.9622 ± 0.0103	18.8	0.00	3.1	0.953
	Random	0.9566 ± 0.0046	0.9559 ± 0.0127	19.6	0.00	2.5	0.968
	GP-EI	0.9580 ± 0.0053	0.9615 ± 0.0059	18.5	0.15	1.5	0.969
	TPE	0.9587 ± 0.0052	0.9629 ± 0.0088	18.4	0.12	1.5	0.968
Synthetic	Grid	0.8601 ± 0.0076	0.8718 ± 0.0121	53.4	0.00	21.6	0.946
	Random	0.8624 ± 0.0077	0.8688 ± 0.0109	56.9	0.00	15.1	0.964
	GP-EI	0.8643 ± 0.0071	0.8710 ± 0.0113	56.2	0.17	16.4	0.966
	TPE	0.8679 ± 0.0064	0.8716 ± 0.0127	60.6	0.13	14.2	0.969

4.2.2 Test accuracy

The advantage in cross-validation score did not carry over to the test set. On Digits, test accuracy was almost the same for grid search (0.9749), GP-EI (0.9751) and TPE (0.9753), while random search was lower (0.9671); random search differed significantly from the other three ($p = 0.031$, 0.016 and 0.020, uncorrected). On Breast Cancer and Synthetic, no pair of methods differed significantly (all $p \geq 0.17$). Test sets are small

(Table 2): one Breast Cancer test sample is 0.7 percentage points, so differences of this size are within what chance can produce. Overall, a gain of about half a point in cross-validation score gave no reliable gain on new data.

4.2.3 Score against cost

Figure 1 shows the conventional view, the best cross-validation score against the trial number. Figure 2 shows the same runs on the proposed cost axis, the PpCC. In both views the Bayesian methods rise faster than grid and random search on Digits and Synthetic. On Breast Cancer the curves overlap within their confidence bands.

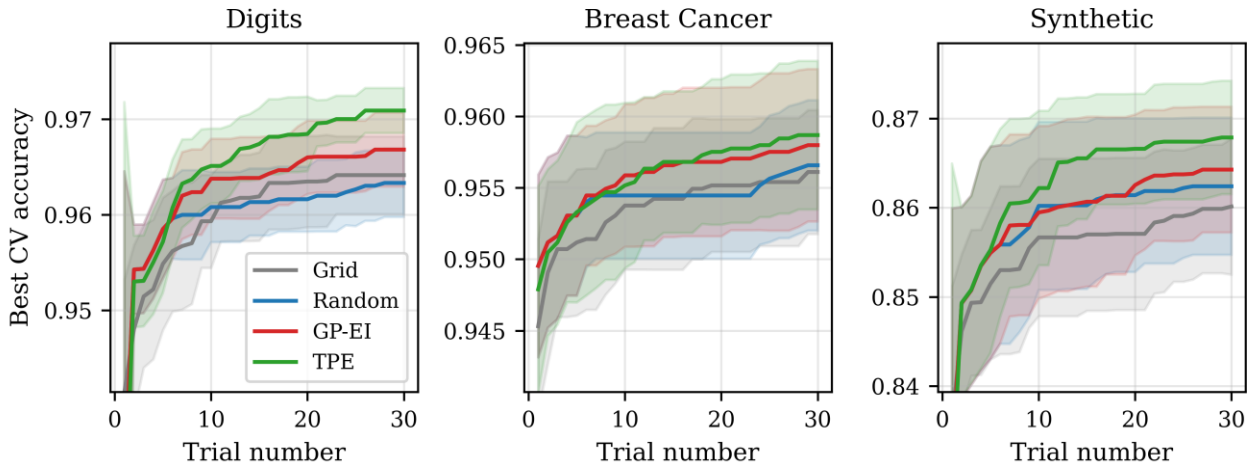


Fig 1: Best cross-validation accuracy against trial number (the usual view). Lines are means over ten repetitions; bands are 95% confidence intervals.

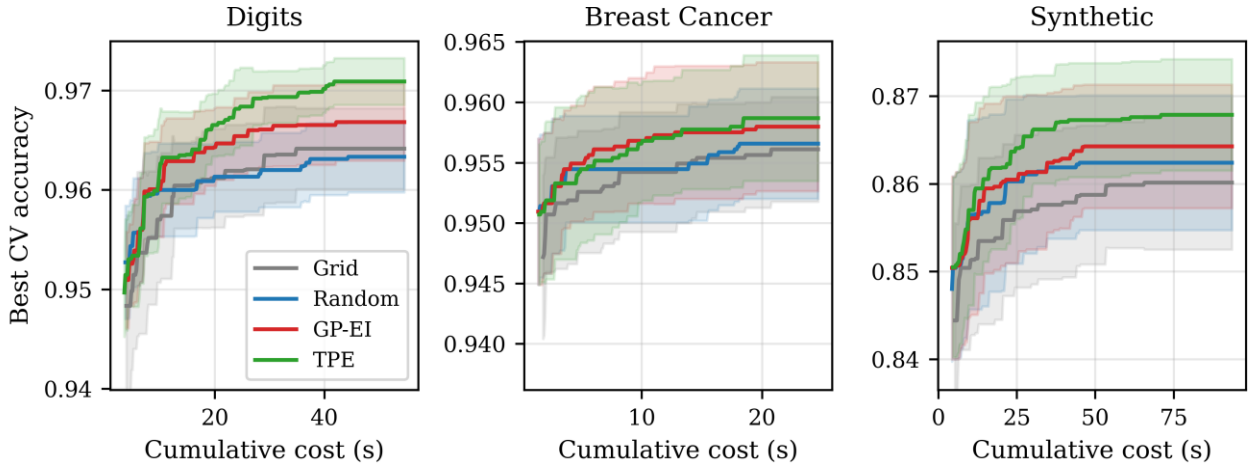


Fig 2: Performance-per-Cost Curves: best cross-validation accuracy against cumulative search cost in seconds, including optimizer overhead. Lines are means over ten repetitions, drawn where all repetitions have a value; bands are 95% confidence intervals.

The two views differ because trials do not cost the same. With 30 trials each, the average total cost of the four methods differed by up to 13% on the same dataset (Table 3): on Synthetic, TPE took 60.6 s against 53.4 s for grid search, and on Digits random search took 45.7 s against 42.2 s for grid search. A comparison at equal trial count is therefore not a comparison at equal cost.

4.2.4 Score at fixed fractions of the budget

Table 4 and Figure 3 give the best score reached at 25%, 50%, 75% and 100% of the reference cost *T*. On Digits and Synthetic, TPE already had the highest mean score at 25% of *T*, but the gaps at that early point are smaller than the confidence intervals, so they should be read as a tendency only. The gaps widen toward the end of the budget.

Table 4. Best cross-validation accuracy at fractions of the reference cost *T*, mean ± 95% CI

Dataset	Method	25% of T	50% of T	75% of T	100% of T
Digits	Grid	0.9574 ± 0.0063	0.9612 ± 0.0050	0.9635 ± 0.0048	0.9641 ± 0.0041
	Random	0.9600 ± 0.0047	0.9613 ± 0.0035	0.9620 ± 0.0038	0.9631 ± 0.0036
	GP-EI	0.9625 ± 0.0044	0.9647 ± 0.0037	0.9664 ± 0.0041	0.9668 ± 0.0039
	TPE	0.9633 ± 0.0046	0.9667 ± 0.0033	0.9693 ± 0.0026	0.9709 ± 0.0024
Breast Cancer	Grid	0.9526 ± 0.0051	0.9542 ± 0.0050	0.9552 ± 0.0048	0.9556 ± 0.0044

Dataset	Method	25% of T	50% of T	75% of T	100% of T
	Random	0.9545 ± 0.0044	0.9545 ± 0.0044	0.9549 ± 0.0047	0.9566 ± 0.0046
	GP-EI	0.9549 ± 0.0048	0.9568 ± 0.0052	0.9575 ± 0.0055	0.9577 ± 0.0054
	TPE	0.9542 ± 0.0061	0.9561 ± 0.0055	0.9577 ± 0.0049	0.9587 ± 0.0052
Synthetic	Grid	0.8535 ± 0.0095	0.8569 ± 0.0087	0.8579 ± 0.0085	0.8599 ± 0.0072
	Random	0.8568 ± 0.0108	0.8611 ± 0.0090	0.8619 ± 0.0081	0.8624 ± 0.0077
	GP-EI	0.8581 ± 0.0106	0.8613 ± 0.0088	0.8633 ± 0.0075	0.8643 ± 0.0071
	TPE	0.8595 ± 0.0098	0.8654 ± 0.0069	0.8673 ± 0.0065	0.8673 ± 0.0065

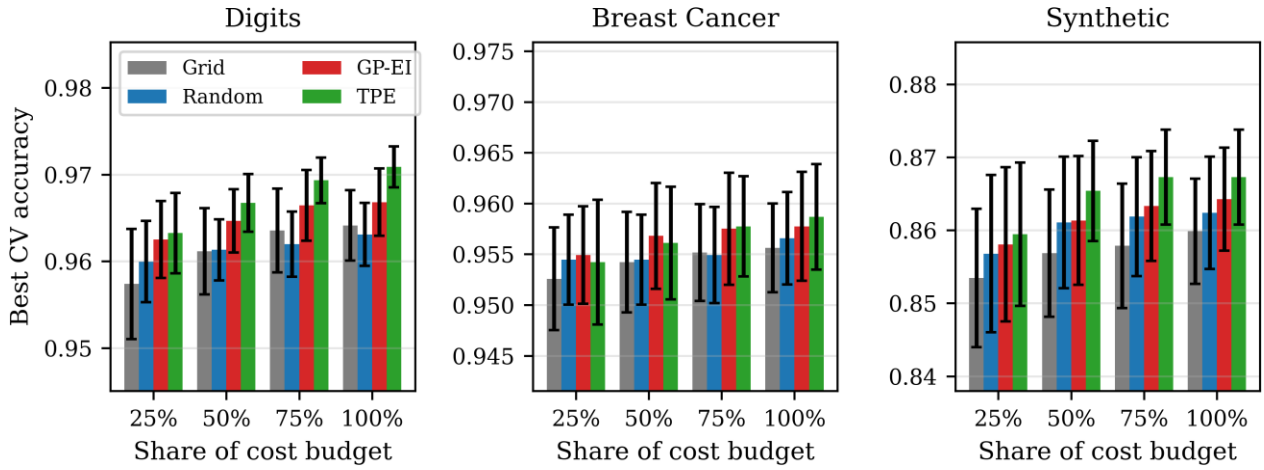


Fig 3: Best cross-validation accuracy at 25%, 50%, 75% and 100% of the reference cost. Bars are means over ten repetitions; whiskers are 95% confidence intervals.

4.2.5 Does the ranking change with the budget?

Table 5 ranks the methods by mean score at each budget fraction. On Digits and Synthetic the order of the two Bayesian methods and the two simple methods is stable across budgets, with the only change being the order of grid and random search on Digits at 75% and 100%. On Breast Cancer the order changes: GP-EI leads at 25% and 50% of the budget and TPE at 75% and 100%, and random search falls from second place at 25% to last place at 75%. These changes are small next to the confidence intervals. They show that a ranking read at one budget is fragile; they do not show that any particular ranking is reliable.

Table 5. Ranking of methods by mean score at each budget fraction (G = Grid, R = Random, GP = GP-EI, T = TPE; best first)

Dataset	25% of T	50% of T	75% of T	100% of T
Digits	T > GP > R > G	T > GP > R > G	T > GP > G > R	T > GP > G > R
Breast Cancer	GP > R > T > G	GP > T > R > G	T > GP > G > R	T > GP > R > G
Synthetic	T > GP > R > G	T > GP > R > G	T > GP > R > G	T > GP > R > G

4.2.6 Time to 99% and PpCC area

Under the shared target, the methods reached 99% of the best score at a cost of 1.5 to 21.6 s (Table 3). This is 8% to 40% of the total cost of the same method, so a large part of each search was spent on small gains. TPE reached the target earliest on all three datasets, together with GP-EI on Breast Cancer (for example on Digits: TPE 8.8 s, GP-EI 10.5 s, grid 14.5 s, random 15.5 s). The PpCC area was lowest for grid search on every dataset (0.948, 0.953 and 0.946) and highest for TPE or GP-EI, but only the comparison of TPE with random search was significant before correction (Digits p = 0.010; Synthetic p = 0.020).

4.2.7 Statistical comparison

Table 6 gives the Wilcoxon p values for all pairs, for the best cross-validation score (the quantity the methods optimize) and for test accuracy (the quantity that matters in use). Many pairs differ on the first measure, and only a few on the second.

Table 6. Wilcoxon signed-rank *p* values over ten repetitions (uncorrected). CV = best cross-validation accuracy; Test = held-out test accuracy

Pair	Digits CV	Digits Test	Breast C. CV	Breast C. Test	Synth. CV	Synth. Test
Grid vs Random	0.715	0.031	0.875	0.172	0.492	0.379
Grid vs GP-EI	0.152	1.000	0.242	0.953	0.105	0.801
Grid vs TPE	0.002	0.926	0.094	1.000	0.002	0.912
Random vs GP-EI	0.016	0.016	0.195	0.500	0.125	0.406
Random vs TPE	0.002	0.020	0.047	0.172	0.008	0.172
GP-EI vs TPE	0.008	1.000	0.250	0.586	0.008	0.828

4.2.8 Optimizer overhead

Figure 4 shows the share of total cost spent by the optimizer itself. For GP-EI it was between 0.31% and 0.84% of the total, and for TPE between 0.21% and 0.66%. Grid and random search had practically none. In this study the overhead is therefore small. That is what one would expect with only 30 trials, a model that takes seconds to train and four hyperparameters. The share would be larger for faster models, more trials, or higher-dimensional spaces, so overhead is still worth reporting.

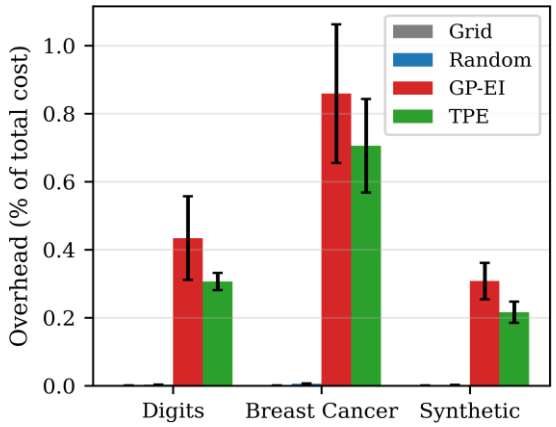


Fig 4: Optimizer overhead as a percentage of total search cost (mean, 95% CI).

4.2.9 Cross-validation score versus test accuracy

Figure 5 shows the difference between the best cross-validation score and the test accuracy of the refit model. On average the cross-validation score was below the test accuracy for all methods and datasets, except for random search on Breast Cancer where the two were almost equal; all intervals include zero except grid search and GP-EI on Digits. The likely reason is that each fold trains on two thirds of the training data, while the final model uses all of it. The selection bias described in the literature [25] is not visible here, at least at this budget. A single run on Digits had suggested that the low test accuracy of random search came from overfitting to the validation score. The ten repetitions do not back this up: random search was lower on the test set on Digits, but its gap between cross-validation and test accuracy was not larger than that of the other methods.

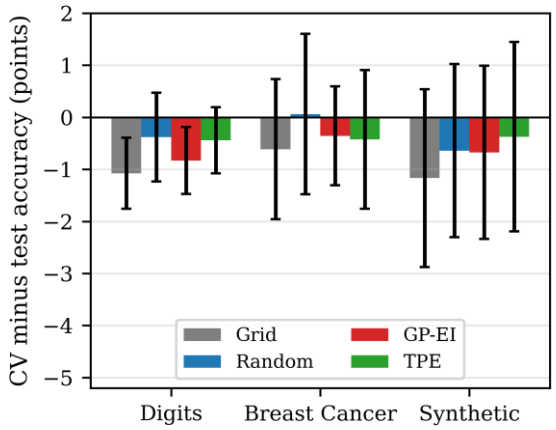


Fig 5: Best cross-validation accuracy minus test accuracy, in percentage points (mean, 95% CI). Negative values mean the test accuracy was higher.

4.3 Single run versus repeated runs

A first run on Digits (seed 42) suggested that every method reaches a plateau within about six seconds. The repeated runs do not support such a tidy picture. Plateaus do occur, but when they occur differs between repetitions and between methods, and so does the ranking at a given budget. One run can easily tell a cleaner story than the data allow.

4.4 Discussion and limits

These results do not show that TPE is the best method, nor that grid search is as good as Bayesian optimization. They show three narrower things. The usual summary (best score at 30 trials) ranks the methods clearly, but that ranking mostly disappears on held-out data, so the usual summary can make differences look bigger than they are. Equal trial counts gave unequal costs, by up to about 13%. And a ranking can depend on the budget at which it is read, as on Breast Cancer, which only the full PpCC reveals.

The study has limits. It uses one model family, three small datasets, one cost unit (wall-clock time on a shared machine), a budget of 30 trials and a truncated grid. The multi-fidelity methods from Section 2 were not run, and the test sets are small. Larger models or longer searches would need their own experiments.

5. IMPLICATIONS FOR PRACTICE AND RESEARCH

For practitioners, the question to ask is not “which method gives the best score at the full budget?” but “which method’s PpCC is highest within the budget that can actually be spent?”. The answers can differ, and the difference is only visible with the whole curve.

Benchmark suites such as HPOBench [26] and NAS-Bench [27] could report PpCCs next to their final-score tables. This costs almost nothing, since the trajectory is already recorded during a benchmark run and is normally thrown away when only the final value is kept.

For people designing methods, cost-awareness is now a niche option in Bayesian optimization [14] and a scheduling rule in multi-fidelity methods [6], [7]. It could become a central goal, for example through acquisition rules that aim at the area under the PpCC and not only at the final value.

6. A RESEARCH AGENDA FOR COST-AWARE HPO

Four open directions follow from this work.

(1) Reporting templates. Community conventions, similar to model cards [28] or reproducibility checklists, that say how a PpCC should be measured (hardware, cost unit) and reported next to the usual final-score tables.

(2) Acquisition rules judged by PpCC area. Most acquisition rules are designed and judged by final regret. How rules that target the area under the PpCC would behave, and whether it matters at different problem sizes, is not known.

(3) PpCC-aware multi-fidelity scheduling. Hyperband-type methods are cost-aware in their schedule. Whether the schedule could adapt during a run, using the shape of the PpCC so far and not a plan fixed in advance, is an open question.

(4) PpCC and generalization. The experiment shows that cross-validation score and test accuracy can disagree. A PpCC built on nested cross-validation [25] or on held-out data would be more honest, but it costs more and could add to the very cost problem it is meant to measure.

7. CONCLUSION

HPO research has produced many search strategies, but the usual way of comparing them, the best score after a fixed number of trials, leaves out how fast the score grows with cost. The survey and the experiment (four methods, three datasets, ten repetitions each) show that this omission has real effects. Methods with the same number of trials differed in cost by up to about 13%. A ranking by cross-validation score did not carry over to test accuracy. On one dataset the ranking changed with the budget. The study also found that optimizer overhead was under 1% of cost, and it found no sign that overfitting to the validation score explains the differences in test accuracy. The Performance-per-Cost Curve and its summary measures are a small addition to standard reporting. Larger models, other cost units and multi-fidelity methods still need to be tested.

Reproducibility. All experiments use scikit-learn 1.8.0 [21] and Optuna [23], ten fixed seeds (42 and 1–9), and one thread per process. The code, the raw per-trial results and the analysis scripts are available at https://github.com/prasadnori/hpo-cost-aware-experiment.

8. REFERENCES

[1] F. Hutter, L. Kotthoff, and J. Vanschoren, Eds., *Automated Machine Learning: Methods, Systems, Challenges*. Springer, 2019.

[2] R. Kohavi, “A study of cross-validation and bootstrap for accuracy estimation and model selection,” in *Proc. IJCAI*, 1995, pp. 1137–1143.

[3] M. Feurer and F. Hutter, “Hyperparameter optimization,” in *Automated Machine Learning*. Springer, 2019, pp. 3–33.

[4] J. Bergstra and Y. Bengio, “Random search for hyper-parameter optimization,” *J. Mach. Learn. Res.*, vol. 13, pp. 281–305, 2012.

[5] B. Shahriari, K. Swersky, Z. Wang, R. P. Adams, and N. de Freitas, “Taking the human out of the loop: A review of Bayesian optimization,” *Proc. IEEE*, vol. 104, no. 1, pp. 148–175, 2016.

[6] L. Li, K. Jamieson, G. DeSalvo, A. Rostamizadeh, and A. Talwalkar, “Hyperband: A novel bandit-based approach to hyperparameter optimization,” *J. Mach. Learn. Res.*, vol. 18, no. 185, pp. 1–52, 2018.

[7] S. Falkner, A. Klein, and F. Hutter, “BOHB: Robust and efficient hyperparameter optimization at scale,” in *Proc. ICML*, 2018, pp. 1437–1446.

[8] M. Feurer, J. T. Springenberg, and F. Hutter, “Initializing Bayesian hyperparameter optimization via meta-learning,” in *Proc. AAAI*, 2015, pp. 1128–1135.

[9] E. Strubell, A. Ganesh, and A. McCallum, “Energy and policy considerations for deep learning in NLP,” in *Proc. ACL*, 2019, pp. 3645–3650.

[10] C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning*. MIT Press, 2006.

[11] J. Snoek, H. Larochelle, and R. P. Adams, “Practical Bayesian optimization of machine learning algorithms,” in *Proc. NIPS*, 2012, pp. 2951–2959.

[12] J. Bergstra, R. Bardenet, Y. Bengio, and B. Kégl, “Algorithms for hyper-parameter optimization,” in *Proc. NIPS*, 2011, pp. 2546–2554.

[13] F. Hutter, H. H. Hoos, and K. Leyton-Brown, “Sequential model-based optimization for general algorithm configuration,” in *Proc. LION*, 2011, pp. 507–523.

[14] K. Swersky, J. Snoek, and R. P. Adams, “Freeze-thaw Bayesian optimization,” arXiv:1406.3896, 2014.

[15] K. Jamieson and A. Talwalkar, “Non-stochastic best arm identification and hyperparameter optimization,” in *Proc. AISTATS*, 2016, pp. 240–248.

[16] K. Eggensperger, F. Hutter, H. H. Hoos, and K. Leyton-Brown, “Efficient benchmarking of hyperparameter optimizers via surrogates,” in *Proc. AAAI*, 2015, pp. 1114–1120.

[17] T. Domhan, J. T. Springenberg, and F. Hutter, “Speeding up automatic hyperparameter optimization of deep neural networks by extrapolation of learning curves,” in *Proc. IJCAI*, 2015, pp. 3460–3468.

[18] R. Schwartz, J. Dodge, N. A. Smith, and O. Etzioni, “Green AI,” *Commun. ACM*, vol. 63, no. 12, pp. 54–63, 2020.

[19] C. Coleman et al., “DAWNBench: An end-to-end deep learning benchmark and competition,” in *NIPS ML Systems Workshop*, 2017.

[20] P. Mattson et al., “MLPerf training benchmark,” in *Proc. MLSys*, 2020.

[21] F. Pedregosa et al., “Scikit-learn: Machine learning in Python,” *J. Mach. Learn. Res.*, vol. 12, pp. 2825–2830, 2011.

[22] L. Breiman, “Random forests,” *Mach. Learn.*, vol. 45, no. 1, pp. 5–32, 2001.

[23] T. Akiba, S. Sano, T. Yanase, T. Ohta, and M. Koyama, “Optuna: A next-generation hyperparameter optimization framework,” in *Proc. KDD*, 2019, pp. 2623–2631.

[24] F. Wilcoxon, “Individual comparisons by ranking methods,” *Biometrics Bulletin*, vol. 1, no. 6, pp. 80–83, 1945.

[25] G. C. Cawley and N. L. C. Talbot, “On over-fitting in model selection and subsequent selection bias in performance evaluation,” *J. Mach. Learn. Res.*, vol. 11, pp. 2079–2107, 2010.

[26] K. Eggensperger et al., “HPOBench: A collection of reproducible multi-fidelity benchmark problems for HPO,” in *Proc. NeurIPS Datasets and Benchmarks Track*, 2021.

[27] C. Ying, A. Klein, E. Christiansen, E. Real, K. Murphy, and F. Hutter, “NAS-Bench-101: Towards reproducible neural architecture search,” in *Proc. ICML*, 2019, pp. 7105–7114.

[28] M. Mitchell et al., “Model cards for model reporting,” in *Proc. ACM FAT**, 2019, pp. 220–229.