COST-AWARE ADVERSARIAL BEST ARM IDENTIFICATION

Nikita Ivkin, Zohar Karnin, Valerio Perrone, Giovanni Zappella[*]
Amazon Web Services
{ivkin, zkarnin, vperrone, zappella}@amazon.com

ABSTRACT

We tackle the adversarial best arm identification problem, in its extension where different arms are associated with different costs. This extension is a natural abstraction of the cost-aware multi-fidelity hyperparameter optimization (CAMF-HPO) problem, where the costs, e.g. monetary or wall-clock time, of evaluations with different hyperparameter configurations are not uniform. We provide an algorithm with rigorous guarantees, matching a provable lower bound up to logarithmic factors. To the best of our knowledge, this is the first algorithm for this important problem that provides provable guarantees. The provided analysis has an application to the stochastic best arm identification problem. The core of our algorithm extends the Sequential Halving (SH) algorithm, and our novel view proves that information does not have to be discarded between rungs, closing a gap between practical implementations of SH and the theory behind it.

1 INTRODUCTION

Hyperparameter optimization (HPO) and Neural Architecture Search (NAS) aim to find the global minimizer of an expensive black-box function $f : \mathcal{X} \to \mathbb{R}$, where $\mathcal{X} \subset \mathbb{R}^d$ is the space of valid HP configurations and $f(\mathbf{x})$ is the metric to optimize. For example, $f(\mathbf{x})$ is the validation error of a machine learning model trained using the HPs $\mathbf{x} \in \mathcal{X}$. While it is common to minimize the number of queries to f, this implicitly assumes that each hyperparameter evaluation has the same cost (Jones et al., 1998; Snoek et al., 2012; Shahriari et al., 2016; Ju et al., 2017; Frazier, 2018a). This is not true in practice. For instance, the number of layers in a neural network, or maximum number of leaves in a tree ensemble drastically affect training time. Typically, the evaluation of $f(\mathbf{x})$ can be done at different fidelities of increasing costs and accuracies. The observation is that promising hyperparameter configurations (configs), outperform poor configs even at lower fidelities (e.g., a small number of epochs). Multi-fidelity HPO is today table-stakes for tuning deep networks (Jamieson & Talwalkar, 2015; Li et al., 2016; Klein et al., 2017a; Falkner et al., 2018; Klein et al., 2020).

In this paper we combine both of the above concepts and tackle cost-aware multi-fidelity HPO (CAMF-HPO). We extend the known Sequential Halving (SH) and HyperBand (HB) algorithms to handle CAMF-HPO, without the need of knowing the cost of an evaluation in advance. Our extensions, called Cost Aware Sequential Halving (CASH, Algorihtm 1) and Cost Aware HyperBand (CAHB, Algorithm 2) respectively, emit rigorous guarantees of optimality up to logarithmic factors, under very mild assumptions. To our knowledge, we are the first to provide a solution with a provable guarantee to this highly important problem. Our analysis extends to the stochastic multi-armed bandit (MAB) identification problem. We analyze CASH in the stochastic best arm identification setting and provide rigorous guarantees. At the core of our analysis is a high probability bound on the time point from which the empirical estimate of the mean of a suboptimal arm will forever be inferior to that of the optimal arm. This bound might be interesting in its own right and could help in other MAB settings. Our novel perspective simplifies the analysis of SH, and shows that there is no need to forget the information from past rungs, as suggested in Karnin et al. (2013). This closes a gap between the analysis and practical implementations that use all historical information from previous rungs. In fact, in some combinations of arm values, our analysis is asymptotically better than in Karnin et al. (2013), and matches the lower bound up to a constant factor. Other than theoretical analysis, we compare CAHB with the standard HB in experiment. We show the benefits of CAHB in the highly common scenario where one wishes to run HPO with a limited budget of time or dollars.

[*]Authors in alphabetical order.

```
Input: Budget $B$, number of configs $n$, cost vector $c$, maximum resource $R$, expansion factor $\eta > 1$ ;
Set the number of rungs $S = \left\lceil \min\{\log_\eta\left(\frac{\sum_i c_i}{\min_i c_i}\right), \log_\eta(R)\} \right\rceil$ ;
Define the surviving config set $A_1 = [n]$;
for $s = 1, 2, \ldots, S$ do
    Set the rung budget $B_s = \lfloor B/S \rfloor$ ;
    Query the configs in $A_s$ in a round-robin manner until the budget $B_s$ is completed;
    Update the estimate $\hat{\mu}$ with the results of the above queries;
    Let $i_1, i_2, \ldots, i_{|A_s|}$ be such that $\hat{\mu}_{i_1} \geq \ldots \geq \hat{\mu}_{i_{|A_s|}}$;
    Let $j$ be the largest index for which $\sum_{\ell=1}^{j} c_{i_\ell} \leq \frac{1}{\eta} \sum_{\ell=1}^{|A_s|} c_{i_\ell}$;
    Set $A_{s+1} = \{i_1, i_2, \ldots, i_j\}$
end
return the item of $A_S$ with the maximum empirical value, $\text{argmax}_i\{\hat{\mu}_i | i \in A_S\}$
```

Algorithm 1: Cost-Aware Sequential Halving

2 Cost-aware Sequential Halving

Consider the following game. A player must find the best config out of n possible configurations. In every round the player can query the value of a config and obtain a noisy estimate of the true value of the config. The objective is to find the best (or ϵ-best) config while minimizing the overall cumulative cost of queries. A few notations: we use the index i to determine the identity of a config. The vector $\mu \in \mathbb{R}^n$ contains the unknown true values of the configs, meaning that μ_i is the value of config i. We define a cost vector $c \in \mathbb{R}^n$ with positive entries, where c_i is the cost of config i. Namely, the cost of running config i for a single fidelity unit is c_i. We use the index t to index the round number. In every round t we choose a config i_t to query. By doing so, we improve our estimate of μ, denoted by $\hat{\mu}^{(t)}$ (when it is clear from context we will omit the superscript t) and pay a cost of c_i. When querying config i the estimate of μ is improved only on the i'th coordinate. We do not posit any assumptions about the convergence of our estimate of μ other than the assumption that as we query config i more, our estimate of μ_i becomes more accurate, and that as the number of queries for config i tend to infinity, the error of our estimate of μ_i tends to zero. In particular this guarantees that for every suboptimal config after a finite amount of queries its estimate will be worse than that of the optimal config. We make this formal in definition 2.1 by providing a quantitive version of this assumption.

Definition 2.1. *Let i^* be the optimal configuration and let i be a suboptimal config. Let m_i be the smallest integer such that for any integer $m \geq m_i$, the estimates $\hat{\mu}_i, \hat{\mu}_{i^*}$ obtained with m queries to i, i^* are such that $\hat{\mu}_i > \hat{\mu}_{i^*}$ ($\hat{\mu}_i < \hat{\mu}_{i^*}$) in a minimization (maximization) objective.*

For ease of reading we assume throughout that the objective is maximization. Algorithm 1 describes our proposed technique. The algorithm splits its budget equally between R rungs. In each rung, there is a set of survived configs that are queried uniformly with the budget of the rung. In the first rung, all the configs are in the survived group meaning that the configs are queried in a round-robin manner. At the end of each rung we eliminate a sufficient amount of configs so that the sum of costs is shrunk by a factor of η (a hyperparameter). Once the queries are done, we output the config in the surviving set with the best empirical value. Note that the algorithm, as written, requires knowledge of the cost vector c to determine the number of rungs S, which can be mitigated in several ways: (1) In most reasonable scenarios, the number of configs will be large enough so that $\sum_i c_i / \min_i c_i > R$, in which case we do not need to know the costs in advance. (2) If we first run each of the configs in the minimum fidelity, we get an accurate estimate of the different c_i's and can determine S according to it. (3) Surrogate models for costs such as runtime are easy to train. Before analyzing the algorithm, we begin with a lower bound for the budget required to identify the best config.

Observation 2.2. *In order to identify the best config, one must spend a budget of at least $\sum_{i=1}^n c_i m_i$, where $m_1 = m_2$.*

Proof. Without spending the mentioned budget there must be some config i queried at most $t_i < m_i$ times. In this case we are not guaranteed that $\hat{\mu}_i < \hat{\mu}_1$ and we cannot distinguish between the cases where config 1 or i are optimal. In other words, it is impossible to distinguish between the setting of the reward vector μ and μ' obtained by changing only μ_i to be $\mu'_i > \mu_1$, where the estimates all configs other than i are identical and that of i is identical until the time t_i. □

We now begin the analysis of Algorithm 1, with the following helpful definition of indices of interest.

Definition 2.3. *Let $C = \sum_{i=1}^n c_i$. Assume w.l.o.g. that $i = 1$ is the optimal config and $m_1 = m_2 \geq m_3 \geq \ldots \geq m_n$. Define $j_0 = n$, and for $s \geq 1$ let j_s be the maximal index for which $\sum_{i=1}^{j_s} c_i \leq \frac{1}{\eta} \sum_{i=1}^{j_{s-1}} c_i$.*

We now provide an upper bound to the budget required by Algorithm 1. The bound is given as a function of the cost vector and these j_s indices. In what follows we will provide a simpler representation tying this bound to the lower bound of Observation 2.2.

Lemma 2.4. *If $B \geq SC \max_{s=1}^{S} m_{j_s} \eta^{1-s}$ then the returned config is guaranteed to be the optimal.*
We are now ready to tie the expression of the upper bound to the intuitive lower bound of the budget required to identify the best config.

Lemma 2.5. $SC \max_{s=1}^{S} m_{j_s} \eta^{1-s} \leq \eta S \left(\sum_{i=1}^{n} m_i c_i + \frac{m_1 \max_i c_i}{1-1/\eta} \right)$.

By combining the above two lemmas (proved in the appendix) we reach the following.

Corollary 2.6. *Assuming that $\frac{m_1 \max_i c_i}{1-1/\eta} \leq \sum_{i=1}^{n} m_i c_i$, and $\frac{\sum_{i=1}^{n} c_i}{\min_i c_i} > R$, we have that when the budget B is at least $B \geq \eta 2 \log_\eta(R) \sum_{i=1}^{n} m_i c_i$, Algorithm 1 returns the optimal arm. In particular this means that the required budget is a multiplicative factor of $\eta 2 \log_\eta(R) = O(\ln(R))$ away from the lower bound of Observation 2.2.*
A note regarding the assumptions of Corollary 2.6. The first is not completely guaranteed but happens only in pathological examples where we compare an extremely small number of options or the costs are highly skewed. The second assumption is not a prerequisite for the algorithms success. Intuitively, if the second assumption is incorrect, it means that we will not use the lowest fidelity and in retrospect we should have redefined the lower fidelity to be higher and set R to be lower. Setting η according to the statement of Corollary 2.6, η should be chosen to minimize the expression $\eta \log_\eta(R)$. This is minimized when $\eta / \ln(\eta)$ is minimized, that in turn is minimized when $\eta = e$. Note that the derivative of the expression is defined for $\eta > 1$ and is equal to 0 only when $\eta = e$. This observation matches the common use of $\eta = 3$ in practical papers testing SH, HyperBand, or their variants.

3 STOCHASTIC MAB COST-AWARE BEST ARM IDENTIFICATION

The results above can be extended to analyze the stochastic setting. The analysis turns out to be different than the existing approaches, and in particular handles the setting where information is reused between rungs rather than being discarded. This has been known to help by practitioners, but until now the theory was not able to back up this practical improvement. The Sequential Halving (SH) algorithm was originally proposed by Karnin et al. (2013). It is similar to Algorithm 1 when all costs are equal to 1, and $R = \infty$. The difference is that in SH the information obtained in one rung is ignored in the next. To understand the guarantees we introduce some notations. Assume w.l.o.g that the means of the arms are of descending order meaning $\mu_1 > \mu_2 \geq \mu_3 \geq \ldots$. Let $\Delta_i = \mu_1 - \mu_i$ be the gap or sub-optimality of arm $i > 1$ and define $\Delta_1 = \Delta_2$. Let $H_1 = \sum_{i=1}^{n} i/\Delta_i^2$ and $H_2 = \max_i i/\Delta_i^2$. In their paper they prove that SH succeeds with probability at least $1-\delta$ when $B \geq c \log(n) \log(\log(n)/\delta) H_2$. The quantity H_2 is known to be $H_1/\log(n) \leq H_2 \leq H_1$, where both inequalities may be tight depending on the ratios between the different Δ_i's. The lower bound to the MAB problem is w.r.t. H_1. Specifically one must have a budget of $B \geq cH_1$ for some constant c, to succeed with constant probability. In what follows (Theorem 3.2) we prove that Algorithm 1 succeeds w.p. $1-\delta$ when $B \geq c \log(n) \log(1/\delta) H_1$. This is in a sense incomparable to the analysis of Karnin et al. (2013). In a setting where δ is large and $H_2 = H_1$ or is close to it, Theorem 3.2's guarantee is actually superior. However, when H_2 is closer to $H_1/\log(n)$, the inverse is true.

The high level idea of our analysis is as follows. Given the realizations of the random observations of the different arms, every suboptimal arm i has some integer m_i from which the estimate $\hat{\mu}_i < \hat{\mu}_{i_*}$ for all time indices greater or larger than m_i (definition 2.1). m_i is not arbitrarily chosen but is a random variable dependent on the realizations of the arms. We require a high probability bound for m_i, and this is given in Lemma 3.1. With that we provide an upper bound to $\sum_i m_i$, leading to the main theorem.

Lemma 3.1. *Let $x_1, \ldots, x_t, \ldots$ be realizations of a sub-Gaussian distribution with zero mean and unit variance. Let $\mu_t = (1/t) \sum_{i=1}^{t} x_t$ be the empirical mean at time t. Let $1 > \epsilon > 0$ and m be the minimal integer for which $|\mu_t| < \epsilon$ for all $t \geq m$. Then for some universal constant c it holds that $\mathbb{P}[m > c \ln(1/\delta)/\epsilon^2] \leq \delta$.*

Due to space restrictions we provide the proof in the appendix. We are now ready to prove the main theorem analyzing Algorithm 1.

Theorem 3.2. *Algorithm 1, when applied to the stochastic, unweighted best arm classification problem, is guaranteed to provide the best arm w.p. at least $1-\delta$ as long as $B \geq c \log(n) \log(1/\delta) H_1$*

Proof. Consider the difference of realizations of queries to arm i with those to queries of arm 1. That is, we define x_t as the difference between the realization of arm i when queried the t'th time, and that of arm 1 when quried the t'th time. If either is not queried t times, define x_t as an r.v. distributed according to the difference of two new arm queries. We notice that the time at which an arm is queried is dependent on the past realizations of both it and other arms. However, since the realization

```
1 Input: Budget $B$, Generator $\Theta$ for new configs, cost oracle $c$ for configs, maximum resource $R$, expansion
    factor $\eta > 1$ ;
2 Set the number of bands $S = \lceil \log_\eta(R) \rceil + 1$;
3 for $s = 0, 1, 2, \ldots, S-1$ do
4     Draw configs from $\Theta$, whose sum of costs $C = \sum_i c_i$ is as large as possible while being bounded
        $\eta^s C \leq \frac{B}{S(S-s)}$;
5     Invoke Algorithm 1 with the configs, max budget of $R/\eta^s$, budget $B/S$, and expansion factor $\eta$. Store
        the returned best config;
6 end
7 return the best config out of those returned in the different bands;
```

Algorithm 2: CAHB: Cost-Aware HyperBand

is independent of the history the sequence of x_t defined above is i.i.d.. Given this, by fixing the randomness and the behavior of Algorithm 1, the definition of m_i applies, and it is precisely the time from which the empirical average of x_t's becomes larger than $-\Delta_i$ for all $t \geq m_i$. Since these x_i's are i.i.d sub-Gaussian, Lemma 3.1 indicates that for some constant c, $Pr[m_i > c\ln(1/\delta)/\Delta_i^2] \leq \delta$. With this in mind, we move to apply Corollary 2.6, or rather a variation of it. In the stochastic setting R, the maximum budget is infinity. However, it only appears in the algorithm in a min expression with $\sum_i c_i / \min_i c_i$, which is equal to n, the number of arms. With that in mind the statement is $B \geq \eta \log_\eta(n) \sum_{i=1} m_i$. It follows that in order to get a guarantee we must provide a high probability bound for $\sum_i m_i$. Now, according to the equation above, each m_i is at worst concentrated around zero as an exponential r.v. with mean $O(1/\Delta_i^2)$. As such, their sum is at worst concentrated as an exponential variable with mean $O(H_1)$. Note that independence is not required. Consider the extreme case where all m_i's are multiples of m_1, the argument holds even in this scenario. We conclude that w.p. of at least $1-\delta$, $\sum_i m_i < c\log(1/\delta)H_1$, and the statement follows. □

4 Cost-Aware Hyper Band

Finally, we extend CASH (Algorithm 1) to a setting where the number of configs is not pre-defined. This is the case in HPO, where there is a trade-off between trying a small number of configs while obtaining a good estimate of which one is the best one vs. trying a large number of configs but with a weaker estimate of the best amongst them. This issue was brought up in Li et al. (2016) when designing the HB algorithm, and was referred to as the n vs. B/n problem, with n being the number of configs. Rather than the number of configs n, what is relevant in our setting is the overall cost of the configs defined earlier as $\sum_i c_i = C$. In other words, we can either choose a collection of configs whose sum of costs is small, and evaluate many of them in an accurate way, or explore a larger collection whose overall cost C is large but only obtain a crude approximation for most of them. We provide our procedure in Algorithm 2. Just like Algorihtm 1, it requires as input a budget B, a scaling factor η, and a maximum resource R. Unlike Algorithm 1 it does not accept a list of configs as input, but rather a generator Θ of configs. This generator can be a process that samples a config uniformly at random. For extensions to BO, the generator could be an artifact of a surrogate function estimating the loss of each config. The algorithm divides its budget equally among $\log_\eta(R) + 1$ different rungs. In each rung we define a minimum resource r and invoke a CASH instance that will use resource levels from r to R. We use Θ to draw random configs for each CASH instance in a way that their corresponding C is as large as possible while still allowing to run a CASH instance with the required minimum budget r. Formally, this balance happens as long as $rC\log_\eta(R/r) \leq B$ where B is the budget for the rung. The CAHB algorithm returns the optimal config taken from all CASH instances. The analysis of CAHB is analogous to the one of HB. Intuitively, given a maximum fidelity of R we do not know how small the minimum fidelity should be. This translates exactly to the n vs. B/n, or cast to our setting, the C vs. B/C problem. The solution is to try all options simultaniously in a geometric grid. We try a minimum budget of $r = 1, \eta, \eta^2, \ldots$, and by doing so we lose at worst a log factor when compared to an algorithm that has prior knowledge of the best minimum budget r.

5 Conclusion

This work is the first to extend SH and HB to the cost-aware setting, factoring in the heterogeneous cost coming from different hyperparameter evaluations, such as larger or smaller architectures in NAS problems. We established a novel adversarial best arm identification problem where different arms are associated with different costs. Our algorithm comes with rigorous theoretical guarantees and a provable lower bound up to logarithmic factors.

References

Majid Abdolshah, Alistair Shilton, Santu Rana, Sunil Gupta, and Svetha Venkatesh. Cost-aware multi-objective bayesian optimisation. *arXiv preprint arXiv:1909.03600*, 2019.

Tobias Domhan, Tobias Springenberg, and Frank Hutter. Extrapolating learning curves of deep neural networks. In *ICML AutoML Workshop*, 2014.

Stefan Falkner, Aaron Klein, and Frank Hutter. Bohb: Robust and efficient hyperparameter optimization at scale. In *Proceedings of the International Conference on Machine Learning (ICML)*, pp. 1436–1445, 2018.

Alexander IJ Forrester, András Sóbester, and Andy J Keane. Multi-fidelity optimization via surrogate modelling. *Proceedings of the royal society a: mathematical, physical and engineering sciences*, 463(2088):3251–3269, 2007.

Peter I. Frazier. A tutorial on bayesian optimization. *arXiv preprint arXiv:1807.02811*, 2018a.

Peter I. Frazier. Bayesian optimization. In Esma Gel and Lewis Ntaimo (eds.), *Recent Advances in Optimization and Modeling of Contemporary Problems*, pp. 255–278. INFORMS, 2018b. doi: 10.1287/educ.2018.0188.

Daniel Golovin, Benjamin Solnik, Subhodeep Moitra, Greg Kochanski, John Karro, and D Sculley. Google Vizier: A service for black-box optimization. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1487–1495, 2017.

Gauthier Guinet, Valerio Perrone, and Cédric Archambeau. Pareto-efficient Acquisition Functions for Cost-Aware Bayesian Optimization. *NeurIPS Meta Learning Workshop*, 2020.

Kevin Jamieson and Ameet Talwalkar. Non-stochastic best arm identification and hyperparameter optimization. Technical report, preprint arXiv:1502.07943, 2015.

Donald R Jones, Matthias Schonlau, and William J Welch. Efficient global optimization of expensive black-box functions. *Journal of Global optimization*, 13(4):455–492, 1998.

Shenghong Ju, Takuma Shiga, Lei Feng, Zhufeng Hou, Koji Tsuda, and Junichiro Shiomi. Designing nanostructures for phonon transport via bayesian optimization. *Physical Review X*, 7(2):021024, 2017.

Kirthevasan Kandasamy, Gautam Dasarathy, Jeff Schneider, and Barnabas Poczos. Multi-fidelity bayesian optimisation with continuous approximations. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 1799–1808. JMLR. org, 2017.

Z. Karnin, T. Koren, and O. Somekh. Almost optimal exploration in multi-armed bandits. In *Proceedings of the 30th International Conference on Machine Learning (ICML)*, pp. 1238–1246, 2013.

Aaron Klein, Stefan Falkner, Simon Bartels, Philipp Hennig, and Frank Hutter. Fast Bayesian Optimization of Machine Learning Hyperparameters on Large Datasets. In Aarti Singh and Jerry Zhu (eds.), *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, volume 54 of *Proceedings of Machine Learning Research*, pp. 528–536, 2017a.

Aaron Klein, Stefan Falkner, Jost Tobias Springenberg, and Frank Hutter. Learning curve prediction with Bayesian neural networks. In *International Conference on Learning Representations (ICLR)*, volume 17, 2017b.

Aaron Klein, Louis Tiao, Thibaut Lienart, Cedric Archambeau, and Matthias Seeger. Model-based asynchronous hyperparameter and neural architecture search. *arXiv preprint arXiv:2003.10865*, 2020.

Eric Hans Lee, Valerio Perrone, Cedric Archambeau, and Matthias Seeger. Cost-aware Bayesian optimization. In *ICML AutoML Workshop*, 2020.

L. Li, K. Jamieson, A. Rostamizadeh, E. Gonina, M. Hardt, B. Recht, and A Talwalkar. Massively parallel hyperparameter tuning. Technical Report 1810.05934v4 [cs.LG], ArXiv, 2019.

Lisha Li, Kevin Jamieson, Giulia DeSalvo, Afshin Rostamizadeh, and Ameet Talwalkar. Hyperband: A novel bandit-based approach to hyperparameter optimization. Technical report, preprint arXiv:1603.06560, 2016.

Matthias Poloczek, Jialei Wang, and Peter Frazier. Multi-information source optimization. In *Advances in Neural Information Processing Systems*, pp. 4288–4298, 2017.

Emmanuel Rio. *Théorie asymptotique des processus aléatoires faiblement dépendants*, volume 31. Springer Science & Business Media, 1999.

Bobak Shahriari, Kevin Swersky, Ziyu Wang, Ryan P Adams, and Nando de Freitas. Taking the human out of the loop: A review of Bayesian optimization. *Proceedings of the IEEE*, 104(1): 148–175, 2016.

Jasper Snoek, Hugo Larochelle, and Ryan P. Adams. Practical Bayesian optimization of machine learning algorithms. In *Proceedings of the 26th Conference on Neural Information and Processing Systems*, pp. 2951–2959, 2012.

Kevin Swersky, Jasper Snoek, and Ryan P Adams. Multi-task bayesian optimization. In *Advances in neural information processing systems*, pp. 2004–2012, 2013.

Kevin Swersky, Jasper Snoek, and Ryan Prescott Adams. Freeze-thaw Bayesian optimization. Technical report, preprint arXiv:1406.3896, 2014.

Jian Wu, Saul Toscano-Palmerin, Peter I Frazier, and Andrew Gordon Wilson. Practical multi-fidelity bayesian optimization for hyperparameter tuning. In *Proceedings of the Thirtieth Conference on Uncertainty in Artificial Intelligence*, 2019.

APPENDIX

A RELATED WORK

Multi-fidelity HPO. A number of HPO techniques have been developed to leverage low-fidelity approximations of the objective function $f(\mathbf{x})$, with early stopping being the simplest approach. Early stopping ends the evaluation of $\mathbf{x}$ at fidelity r if this is predicted to score worse at r_{max} than some earlier (or parallel) configuration $\mathbf{x}'$. The median rule is a simple way to achieve this Golovin et al. (2017). Other methods extrapolate learning curves beyond the current r Domhan et al. (2014); Klein et al. (2017b). These techniques are particularly suited to asynchronous parallel execution: whenever a job is stopped, its resources can be used to evaluate the next hyperparameter configuration $\mathbf{x}$. An alternative to stopping configurations is to pause and resume them Swersky et al. (2014).

Building on early stopping, another successful multi-fidelity technique is successive halving (SH) Karnin et al. (2013); Jamieson & Talwalkar (2015). At each round, $f(\mathbf{x}, r_{\text{min}})$ is evaluated for n number randomly sampled configurations $\mathbf{x}$. Then, $f(\mathbf{x}, 2r_{\text{min}})$ is run for the top $n/2$ of configurations, while the bottom half are discarded. This filtering step is repeated until r_{max} is reached. SH is simple to implement and can be an efficient baseline, especially compared to full-fidelity HPO. However, selecting r_{min} and n is not easy in practice. This is tackled in Hyperband Li et al. (2016), which adds an outer iteration over different values of r_{min}, n. In their standard form, both SH and Hyperband are synchronous algorithms: all configurations have to be evaluated at a certain level r before being promoted, so that processors can spend idle time at every synchronization point to wait for the slowest job. ASHA Li et al. (2019) extends SH and Hyperband to the asynchronous and any-time case. By any-time we mean that ASHA does not require a budget of evalutions but will run evaluations until it is stopped. It does so by promoting configs after a certain minimum amount of configs in the rung have been evaluated. Assuming that the configs are chosen at random and are not close to being decreasing in terms of their loss, roughly the same number of configs are promoted as in the setting of SH. These techniques measure the obtained loss at a certain number of fidelity units. A single fidelity unit is an epoch of a training job, or a training job on a certain dataset size. It is not related to wall-clock time, monetary cost, or energy efficiency. In contrast, we solve the multi-fidelity problem in a cost-efficient way.

Bayesian Optimization (BO) is an extremely common technique for HPO. In a nutshell, the idea is to learn a surrogate function approximating f while we try to minimize (or maximize) f. When observing an evaluation of f we update a posterior distribution for f, which is used to choose the next config $\mathbf{x}$ from regions that are more promising Frazier (2018b). Although classically BO was used for single fidelity HPO, several papers combined the use of multi-fidelity with BO. The multi-fidelity optimizers mentioned above use SH to schedule hyperparameter evaluations, new configurations $\mathbf{x}$ are chosen at random, while BO techniques are able to choose new configs more wisely. BOHB Klein et al. (2017a); Falkner et al. (2018) combines synchronous Hyperband with model-based HPO, which can outperform the random sampling variants, while Asynchronous BOHB Klein et al. (2020) combines ASHA with Bayesian optimization. As with the non-BO multi-fidelity case, all these techniques and their variants, such as Forrester et al. (2007); Kandasamy et al. (2017); Poloczek et al. (2017); Wu et al. (2019), are not cost aware, and minimize the fidelity units rather than the cost. In this paper we do not add a BO component to our algorithm. Indeed, in a real system adding BO to select the configs is highly recommended, but the purpose of this paper is not to provide a complete system, rather deepen the scientific knowledge of multi-fidelity HPO. We note however that the benefits from adding a BO-based mechanism for selecting the configs is orthogonal to our contributions, and we expect the same improvement seen from the mentioned papers to translate to the cost-aware setting.

Cost Aware HPO has been explored mostly in the BO setting. In that context, a common heuristic is to divide the acquisition function by cost to penalize expensive hyperparameter evaluations Snoek et al. (2012); Lee et al. (2020); Guinet et al. (2020). Other instances of cost-aware BO arise in the multi-task and multi-objective setting, respectively through cost-aware, multi-task extensions of entropy search (Swersky et al., 2013), and by incorporating cost preferences to find cheap solutions (Abdolshah et al., 2019). The emphasis of the mentioned papers is on the selection of new configs to be evaluated. The idea is to optimize not the gain from a single evaluation but the gain for a single cost unit. This objective is orthogonal to what we achieve. Our method provides better control over the total cost used, and a better strategy for allocating resources to different fidelities when compared to cost-oblivious approaches. It follows that the gain observed from these config selection methods over cost-oblivious techniques will also be seen in possible future works that combine our improved promotion strategy with these cost-aware config selection methods.

B PROOFS FOR THE NON-STOCHASTIC SETTING

Proof of Lemma 2.4 At round s, we are guaranteed that $\sum_{i \in A_s} c_i \leq C/\eta^{s-1}$. It follows that all configs in A_s will be queried at least

$$\frac{\eta^{s-1}B}{SC} \geq m_{j_s}$$

It follows that any config i for which $\hat{\mu}_i \geq \hat{\mu}_1$ must have $m_i > m_{j_s}$. Since the sum of costs for these i's is less than C/η^s, we must have that $i^* \in A_{s+1}$. Since this holds for all s we must have that $i^* \in A_{S+1}$ and it remains to prove that i^* is indeed the config that is chosen when the algorithm terminates.

In the setting where $S = \left\lceil \log_\eta \left(\frac{\sum_i c_i}{\min_i c_i} \right) \right\rceil$ we must have that the sum of costs

$$\sum_{i \in A_{S+1}} c_i \leq \eta^{-S} \sum_{i \in A_1} c_i \leq \min_i c_i \; .$$

Since the costs are positive this must mean that $|A_{S+1}| = 1$ and the algorithm is successful. Otherwise, in round S we queried the configs with the maximum budget R meaning that for all configs in A_S, therefor also in A_{S+1}, $\hat{\mu}_i = \mu_i$, indicating that the maximizer of $\hat{\mu}_i$ is the optimal config i^*.

Proof of Lemma 2.5 By the definition of j_s, specifically it's maximality we know that for $s > 0$,

$$\sum_{i=1}^{j_s} c_j \geq \frac{1}{\eta} \sum_{i=1}^{j_{s-1}} c_j - c_{j_s+1},$$

where $j_0 = n$. We get that

$$\begin{aligned}
\sum_{i=1}^{j_s} c_j &\geq \frac{1}{\eta} \sum_{i=1}^{j_{s-1}} c_j - c_{j_s+1} \\
&\geq \frac{1}{\eta^2} \sum_{i=1}^{j_{s-2}} c_j - \frac{1}{\eta} c_{j_{s-1}+1} - c_{j_s+1} \\
&\geq \eta^{-s} C - (c_{j_s+1} + \frac{1}{\eta} c_{j_{s-1}+1} + \ldots) \\
&\geq \eta^{-s} C - \frac{\max_i c_i}{1 - 1/\eta}
\end{aligned}$$

and it follows that

$$\begin{aligned}
m_{j_s} C \eta^{1-s} &\leq \eta m_{j_s} \left(\sum_{i=1}^{j_s} c_j + \frac{\max_i c_i}{1 - 1/\eta} \right) \\
&\leq \eta \left(\sum_{i=1}^{j_s} m_i c_i + \frac{m_{j_s} \max_i c_i}{1 - 1/\eta} \right) \\
&\leq \eta \left(\sum_{i=1}^{n} m_i c_i + \frac{m_1 \max_i c_i}{1 - 1/\eta} \right)
\end{aligned}$$

C PROOF FOR THE STOCHASTIC SETTING

Lemma 3.1 Let $x_1, \ldots, x_t, \ldots$ be realizations of a sub-Gaussian distribution with zero mean and unit variance. Let $\mu_t = (1/t) \sum_{i=1}^{t} x_t$ be the empirical mean at time t. Let $1 > \epsilon > 0$ and m be the minimal integer for which $|\mu_t| < \epsilon$ for all $t \geq m$. Then for some universal constant c it holds that

$$\mathbb{P}[m > c \ln(1/\delta)/\epsilon^2] \leq \delta$$

Proof. According to standard concentration bounds for Gaussian variables we know that for t

$$\mathbb{P}[|\mu_t| \geq \epsilon] \leq 2\exp(-\epsilon^2 t/2)$$

Let $t_0 \geq 1/\epsilon^2$ be an integer and consider the sequence $u_i = t_0 + 2i/\epsilon^2$. We notice first that

$$\begin{aligned} Pr\left[\exists i \geq 0, |\mu_{t_{u_i}}| \geq \epsilon\right] &\leq \sum_{i=0}^{\infty} \exp(-\epsilon^2 t_0/2 - i) \\ &\leq \frac{\exp(-\epsilon^2 t_0/2)}{1 - 1/e} \end{aligned} \tag{1}$$

Now, for $t \in (u_i, u_{i+1})$ we use Bernstein's maximal inequality (see e.g. Rio (1999), Theorem B.2),

$$\begin{aligned} &\mathbb{P}\left[\max_{t\in(u_i,u_{i+1})} \left|\sum_{\ell=0}^{t-u_i} x_{u_i+\ell}\right| \geq \epsilon u_{i+1}\right] \\ &\leq \exp\left(-\frac{\epsilon^2 u_{i+1}^2}{2(u_{i+1}-u_i) + 4\epsilon u_{i+1}}\right) \end{aligned}$$

To get a more interpretable form of the above expression we split our analysis to two possible cases. In the first, $2(u_{i+1} - u_i) \geq 4\epsilon u_{i+1}$. Since $t_0 \geq 1/\epsilon^2$ we have

$$u_{i+1}/(u_{i+1} - u_i) \geq u_1/(u_1 - u_0) \geq \frac{(1+2)/\epsilon^2}{2/\epsilon^2} \geq 1$$

hence

$$\begin{aligned} &\mathbb{P}\left[\max_{t\in(u_i,u_{i+1})} \left|\sum_{\ell=0}^{t-u_i} x_{u_i+\ell}\right| \geq \epsilon u_{i+1}\right] \\ &\leq \exp\left(-\frac{\epsilon^2 u_{i+1}^2}{4(u_{i+1}-u_i)}\right) \\ &\leq \exp\left(-\frac{\epsilon^2 u_{i+1}}{4}\right) \end{aligned} \tag{2}$$

Now consider the case where $2(u_{i+1} - u_i) \leq 4\epsilon u_{i+1}$. We have

$$\begin{aligned} \mathbb{P}\left[\max_{t\in(u_i,u_{i+1})} \left|\sum_{\ell=0}^{t-u_i} x_{u_i+\ell}\right| \geq \epsilon u_{i+1}\right] \leq \exp\left(-\frac{\epsilon^2 u_{i+1}^2}{8\epsilon u_{i+1}}\right) = \\ \exp\left(-\frac{\epsilon u_{i+1}}{8}\right) \leq \exp\left(-\frac{\epsilon^2 u_{i+1}}{4}\right) \end{aligned}$$

the last inequality assumes $\epsilon \leq 1/2$. We conclude that either way

$$\mathbb{P}\left[\max_{t\in(u_i,u_{i+1})} \left|\sum_{\ell=0}^{t-u_i} x_{u_i+\ell}\right| \geq \epsilon u_{i+1}\right] \leq \exp\left(-\frac{\epsilon^2 u_{i+1}}{4}\right)$$

Taking a union bound of all $i \geq 0$ gives

$$\begin{aligned} &\mathbb{P}\left[\exists i, \max_{t\in(u_i,u_{i+1})} \left|\sum_{\ell=1}^{t} x_{u_i+\ell}\right| \geq \epsilon u_{i+1}\right] \\ &\leq \exp\left(-\frac{\epsilon^2 t_0}{4}\right) \sum_{i=0}^{\infty} \exp\left(-\frac{2i\epsilon^2/\epsilon^2}{4}\right) = \\ &\exp\left(-\frac{\epsilon^2 t_0}{4}\right) / (1 - \exp(-1/2)) \end{aligned} \tag{3}$$

Let $t \geq u_0 \geq 1/\epsilon^2$, and assume the inequalities in Equations (1), (3) are not held, meaning the inequalities or sums are smaller than the presented quantities. Let u be the largest integer in the u_i sequence such that $u \leq t$. We have by the triangle inequality that

$$\left|\sum_{\ell=1}^{t} x_\ell\right| \leq \epsilon(2u + 2/\epsilon^2) \leq 4\epsilon u$$

where the last inequality holds since $u \geq u_0 \geq 1/\epsilon^2$. It follows that

$$|\mu_t| = \left|\frac{\sum_{\ell=1}^{t} x_\ell}{t}\right| \leq \frac{4\epsilon u}{t} \leq \frac{4\epsilon u}{u} = 4\epsilon \tag{4}$$

We can now conclude the proof by setting $t_0 = c\ln(1/\delta)/\epsilon^2$ for some constant $c > 1$ and parameter $\delta > 0$. Equations (1), (3), combined with Equation (4) indicate that

$$\begin{aligned}\mathbb{P}\left[m > c\ln(1/\delta)/\epsilon^2\right] =& \mathbb{P}\left[\exists t \geq t_0,\ |\mu_t| > \epsilon\right] \\ &\leq \exp\left(-\Theta(c\ln(1/\delta))\right)\end{aligned}$$

by setting c to be a sufficiently large constant, the probability above is upper bounded by δ as required. □