Practical Multi-fidelity Bayesian Optimization of Iterative Machine Learning Algorithms

Jian Wu
Operations Research
& Information Eng.
Cornell University
Ithaca, NY 14850

Saul Toscano-Palmerin
Operations Research
& Information Eng.
Cornell University
Ithaca, NY 14850

Peter I. Frazier
Operations Research
& Information Eng.
Cornell University
Ithaca, NY 14850

Andrew Gordon Wilson
Courant Institute
of Mathematical Sciences
New York University
New York, NY 10003

Abstract

Bayesian optimization is popular for optimizing time-consuming black-box objectives. Nonetheless, for hyperparameter tuning in deep neural networks, the time required to evaluate the validation error for even a few hyperparameter settings remains a bottleneck. Multi-fidelity optimization promises relief using cheaper proxies to such objectives — for example, validation error for a network trained using a subset of the training points or fewer iterations than required for convergence. We propose a highly flexible and practical approach to multi-fidelity Bayesian optimization, focused on efficiently optimizing hyperparameters for iteratively trained supervised learning models. We introduce a new acquisition function, the trace-aware knowledge-gradient, which efficiently leverages both multiple continuous fidelity controls and trace observations — values of the objective at a sequence of fidelities, available when varying fidelity using training iterations. We provide a provably convergent method for optimizing our acquisition function and show it outperforms state-of-the-art alternatives for hyperparameter tuning of deep neural networks and large-scale kernel learning.

1 INTRODUCTION

In hyperparameter tuning of machine learning models, we seek to find hyperparameters $\mathbf{x}$ in some set $A \subseteq \mathbb{R}^d$ to minimize the validation error $f(\mathbf{x})$, i.e., to solve

$$\min_{\mathbf{x} \in A} f(\mathbf{x}) \tag{1.1}$$

Evaluating $f(\mathbf{x})$ can take substantial time and computational power (Bergstra and Bengio, 2012) and may not provide gradient evaluations. Bayesian optimization, which requires relatively few function evaluations, provides a compelling approach to such optimization problems (Jones et al., 1998; Snoek et al., 2012).

As the computational expense of training and testing a modern deep neural network for a single set of hyperparameters has grown, researchers have sought to supplant some evaluations of $f(\mathbf{x})$ with computationally inexpensive low-fidelity approximations. Conceptually, an algorithm can use low-fidelity evaluations to quickly identify a smaller set of promising hyperparameter settings, and then later focus more expensive high-fidelity evaluations within this set to refine its estimates.

Pioneering multi-fidelity approaches focused on hyperparameter tuning for deep neural networks include the Bayesian optimization methods FaBOLAS (Klein et al., 2017a, 2015), Freeze-Thaw Bayesian Optimization (Swersky et al., 2014), BOHB (Falkner et al., 2018), BOCA (Kandasamy et al., 2017), predictive entropy search for a single continuous fidelity (McLeod et al., 2017), early-stopping SMAC (Domhan et al., 2015), and Hyperband (Li et al., 2018). This work builds on earlier multi-fidelity and multi-information source optimization approaches (Huang et al., 2006; Lam et al., 2015; Poloczek et al., 2017) not focusing on hyperparameter tuning.

These validation error approximations perform the same training and testing steps as in standard Bayesian optimization but control fidelity by reducing training iterations, training data points, or validation data points. These approximations present unique opportunities not typically considered in the multifidelity literature, even within the portion focused on hyperparameter tuning. First, we observe a full *trace* of performance with respect to training iterations, rather than just a single performance value at the chosen fidelity: while training with $\mathbf{s}$ iterations, we also calculate as a byproduct a sequence of models fitted with fewer training iterations. Second, by caching state after completing $\mathbf{s}$ iterations, we can significantly

reduce computation time when later training with $\mathrm{s}' > \mathrm{s}$ iterations. This allows quickly evaluating low-fidelity approximations to the validation error using few training iterations for many hyperparameter settings, then later obtaining more accurate observations for the most promising ones by adding iterations. Third, we may simultaneously alter fidelity along several continuous dimensions (iterations, training data, validation data), rather than modifying one continuous fidelity control or choosing from among a discrete set of ambiguously related fidelities.

The knowledge-gradient (KG) approach to acquisition function design (Frazier et al., 2009; Frazier, 2018) is well-suited to multi-fidelity optimization. First, KG overcomes a shortcoming of expected improvement (EI) (Mockus, 1989). EI considers the predictive distribution at the sampled hyperparameter and fidelity only, not the information that sample provides about other fidelities and hyperparameters. However, the entire value of sampling a low fidelity is the information provided about the highest fidelity. Thus, EI does not offer clear guidance about which fidelity to sample and may make poor choices about which hyperparameters to sample with low-fidelity evaluations. Second, KG overcomes a shortcoming of entropy search (ES) (Hennig and Schuler, 2012) and predictive entropy search (PES) (Hernández-Lobato et al., 2014). ES and PES acquisition functions are optimized with derivative-free optimization (Klein et al., 2017a, 2015; Swersky et al., 2014; McLeod et al., 2017) while KG acquisition functions can often be optimized using more efficient gradient-based optimization (Wu and Frazier, 2016).

In this paper, we propose the trace-aware knowledge gradient (taKG) for Bayesian optimization with multiple fidelities. taKG is distinctive in that it leverages both trace information and multiple fidelity controls at once, efficiently selecting training size, validation size, number of training iterations, and hyperparameters to optimize. Moreover, we provide a provably-convergent method for maximizing this acquisition function. taKG addresses the challenges presented by trace observations by considering the reduced cost of adding iterations at a previously evaluated point, and using an intelligent selection scheme to choose a subset of the observed training iterations to include in inference. Additionally, taKG can be used in batch or sequential settings, and also efficiently leverage gradient information if it is available.

We present two variants of our trace-aware knowledge-gradient acquisition function, one for when the cost of sampling is substantial for any fidelity (even when using few training points or iterations), and the other for when the cost and value of information vanish as fidelity decreases to 0. The first form we refer to simply as taKG, and the second as 0-avoiding taKG (taKG$^\emptyset$) because it avoids the tendency of other multi-fidelity methods to measure repeatedly at near-0 fidelities even when these low fidelities provide almost no useful information. Alternative approaches (McLeod et al., 2017; Klein et al., 2017a) add and tune a fixed cost per sample to avoid this issue, while taKG$^\emptyset$ does not require tuning.

Furthermore, we present a novel efficient method to optimize these acquisition functions, even though they cannot be evaluated in closed form. This method first constructs a stochastic gradient estimator which it then uses within multistart stochastic gradient ascent (SGA). We show that our stochastic gradient estimator is unbiased and thus asymptotically consistent, and the resulting SGA procedure converges to a local stationary point of the acquisition function. Multi-start SGA can then produce an approximate global optimum.

Within the literature on KG methods, only Poloczek et al. (2017) considers multi-fidelity optimization. We go beyond that work in supporting multiple continuous fidelities and trace observations and proposing 0-avoidance. All provide important benefits in hyperparameter tuning.

Our numerical experiments demonstrate significant improvement over state-of-the-art alternatives including FaBOLAS (Klein et al., 2017a, 2015), Hyperband (Li et al., 2018), and BOCA (Kandasamy et al., 2017). Our approach also applies to problems without trace observations that use continuous fidelity controls, and we show strong performance in this setting. While our presentation focuses on continuous hyperparameters, our acquisition functions (though not our SGA approach) can be extended to categorical and integer-valued hyperparameters using the method of Garrido-Merchán and Hernández-Lobato (2018). We discuss this in the supplement.

Efficient and flexible multifidelity optimization is of crucial practical importance, as evidenced by growing momentum in this research area. Although Bayesian optimization has shown great promise for tuning hyperparameters of machine learning algorithms, computational bottlenecks have remained a major deterrent to mainstream adoption. With taKG, we leverage crucial trace information, while simultaneously providing support for several fidelity controls, providing remarkably efficient optimization of expensive objectives. This work is intended as a step towards the renewed practical adoption of Bayesian optimization for machine learning.

2 THE taKG AND taKG$^\emptyset$ ACQUISTION FUNCTIONS

In this section we define the trace-aware knowledge-gradient acquisition function. §2.1 defines our formula-

tion of multi-fidelity optimization with traces and continuous fidelities, along with our inference procedure. §2.2 describes a measure of expected solution quality possible after observing a collection of fidelities within a trace. §2.3 uses this measure to define the taKG acquisition function, and §2.4 defines an improved version, taKG$^{\emptyset}$, appropriate for settings in which the the cost and value of information vanish together (for example, as the number of training iterations declines to 0). §2.5 then presents a computational approach for maximizing the taKG and taKG$^{\emptyset}$ acquisition functions and theoretical results justifying its use. §2.6 discusses warm-starting previously stopped traces, and §2.7 briefly discusses generalizations to batch and derivative observations.

2.1 Problem Setting

We model our objective function and its inexpensive approximations by a real-valued function $g(\mathbf{x}, \mathbf{s})$ where our objective is $f(\mathbf{x}) := g(\mathbf{x}, 1)$ and $\mathbf{s} \in [0,1]^m$ denotes the m fidelity-control parameters. (Here, $\mathbf{1}$ in $g(\mathbf{x}, \mathbf{1})$ is a vector of 1s. Boldface denotes vectors throughout.) We assume that our fidelity controls have been re-scaled so that 1 is the highest fidelity and 0 the lowest. $g(\mathbf{x}, \mathbf{s})$ can be evaluated, optionally with noise, at a cost depending on $\mathbf{x}$ and $\mathbf{s}$.

We let $T(\mathbf{s})$ be the additional fidelities observable from already-collected data when observing fidelity $\mathbf{s}$. (T denotes "trace".) Although our framework can be easily generalized, we assume that $T(\mathbf{s})$ is a cross product of sets of the form $[0, s_i]$ (*trace fidelities*) or $\{s_i\}$ (*non-trace fidelities*). We let m_1 denote the number of trace fidelities and m_2 the number of non-trace fidelities. We also assume that the cost of evaluation is non-decreasing in each component of the fidelity.

For example, consider hyperparameter tuning with $m = 2$ fidelities: first is the number of training iterations; second is the amount of training data. Each is bounded between 0 and some maximum value: $s_1 \in [0,1]$ specifies training iterations as a fraction of this maximum value, and $s_2 \in [0,1]$ specifies the number of training data points similarly. Then, $T(\mathbf{s}) = [0, s_1] \times \{s_2\}$ because when training up to a fraction s_1 of the maximum number of iterations, it is easy to simultaneously compute the validation error for fewer iterations. If the amount of validation data is another trace fidelity, we would have: $T(\mathbf{s}) = [0, s_1] \times \{s_2\} \times [0, s_3]$.

We model g using Gaussian process regression jointly over $\mathbf{x}$ and $\mathbf{s}$, assuming that observations are perturbed by independent normally distributed noise with mean 0 and variance σ^2. Each evaluation consists of $\mathbf{x}$, a vector of fidelities $\mathbf{s}$, and a noisy observation of $g(\mathbf{x}, \mathbf{s}')$ for each fidelity $\mathbf{s}'$ in $T(\mathbf{s})$. For computational tractability, in our inference, we will choose to retain and incorporate observations only from a subset $\mathcal{S} \subseteq T(\mathbf{s})$ of these fidelities with each observation. After n such evaluations, we will have a posterior distribution on g that will also be a Gaussian process, and whose mean and kernel we refer to by μ_n and K_n. The supplement describes this inference framework in more detail.

We model the logarithm of the cost of evaluating g using a separate Gaussian process, updated after each evaluation, and let $c_n(\mathbf{x}, \mathbf{s})$ be the predicted cost after n evaluations. We assume for now that the cost of evaluation does not depend on previous evaluations, and then discuss later in §2.6 an extension to warm-starting evaluation at higher fidelities using past lower-fidelity evaluations.

2.2 Valuing Trace Observations

KG acquisition functions (Frazier et al., 2009) value observations by how it improves expected solution quality. They are like EI, except that the "improvement" need not occur at the sampled point. To support defining taKG and taKG$^{\emptyset}$, we define a function L_n that quantifies expected solution quality to (1.1). Given any $\mathbf{x}$ and set of fidelities S, $L_n(\mathbf{x}, S)$ will be the expected loss (with respect to the posterior after n samples) of our final solution to (1.1) if we are allowed to first observe $\mathbf{x}$ at all fidelities in S.

To define this more formally, let $\mathbb{E}_n$ indicate the expectation with respect to the posterior $\mathbb{P}_n$ after n evaluations. Let $\mathbf{y}(\mathbf{x}, S) = (g(\mathbf{x}, \mathbf{s}) + \epsilon(\mathbf{x}, \mathbf{s}) : \mathbf{s} \in S)$ be a random vector comprised of observations of $g(\mathbf{x}, \mathbf{s})$ for all $\mathbf{s} \in S$ with additive independent Gaussian noise $\epsilon(\mathbf{x}, \mathbf{s})$. Then, the conditional expected loss from choosing a solution $\mathbf{x}'$ to (1.1) after this observation is $\mathbb{E}_n\left[g(\mathbf{x}', \mathbf{1}) \mid \mathbf{y}(\mathbf{x}, S)\right]$. This quantity is a function of $\mathbf{x}$, S, $\mathbf{y}(\mathbf{x}, S)$, and the first n evaluations, and can be computed explicitly using formulas from GP regression given in the supplement.

We would choose the solution for which this is minimized, giving a conditional expected loss of $\min_{\mathbf{x}'} \mathbb{E}_n\left[g(\mathbf{x}', \mathbf{1}) \mid \mathbf{y}(\mathbf{x}, S)\right]$. This is a random variable under $\mathbb{P}_n$ whose value depends on $\mathbf{y}(\mathbf{x}, S)$. We finally take the expected value under $\mathbb{P}_n$ to obtain $L_n(\mathbf{x}, S)$:

$$
\begin{aligned}
L_n(\mathbf{x}, S) &:= \mathbb{E}_n\left[\min_{\mathbf{x}'} \mathbb{E}_n\left[g(\mathbf{x}', \mathbf{1}) \mid \mathbf{y}(\mathbf{x}, S)\right]\right] \\
&= \int \mathbb{P}_n\left(\mathbf{y}(\mathbf{x}, S) = \mathbf{y}\right) \min_{\mathbf{x}'} \mathbb{E}_n\left[g(\mathbf{x}', 1) \,|\, \mathbf{y}(\mathbf{x}, S) = y\right] d\mathbf{y},
\end{aligned}
$$

where the integral is over all $\mathbf{y} \in \mathbb{R}^{|S|}$.

We compute this quantity using simulation. To create one replication of this simulation we first simulate $(g(\mathbf{x}, \mathbf{s}) : \mathbf{s} \in S)$ from $\mathbb{P}_n$. We then add simulated noise to this quantity to obtain a simulated $\mathbf{y}(\mathbf{x}, S)$. We then update our posterior distribution on g using this simulated

data, allowing us to compute $\mathbb{E}_n\left[g(\mathbf{x}', \mathbf{1}) \mid \mathbf{y}(\mathbf{x}, S)\right]$ for any given $\mathbf{x}'$ as a predicted value from GP regression. We then use a continuous optimization method designed for inexpensive evaluations with gradients (e.g., multi-start L-BFGS) to optimize this value, giving one replication of $\min_{\mathbf{x}'} \mathbb{E}_n\left[g(\mathbf{x}', \mathbf{1}) \mid \mathbf{y}(\mathbf{x}, S)\right]$. We then average many replications to give an unbiased and asymptotically consistent estimate of $L_n(\mathbf{x}, S)$.

We also define $L_n(\emptyset) = \min_{\mathbf{x}'} \mathbb{E}_n\left[g(\mathbf{x}', \mathbf{1})\right]$. This is the minimum expected loss we could achieve by selecting a solution without observing any additional information. This is equal to $L_n(\mathbf{x}, \emptyset)$ for any $\mathbf{x}$.

The need to compute $L_n(\mathbf{x}, S)$ via simulation will present a challenge when optimizing taKG and taKG$^{\emptyset}$ (which are defined in terms of it). Below, in §2.5 we will overcome this challenge via a novel method for simulating unbiased estimators of the gradient of $L_n(\mathbf{x}, S)$ with respect to $\mathbf{x}$ and the components of S. First, however, we define the taKG and taKG$^{\emptyset}$ acqisition functions.

2.3 Trace-aware Knowledge Gradient (taKG)

The taKG acquisition function will value observations of a point and a collection of fidelities according to the ratio of the reduction in expected loss (as measured using L_n) that it induces, to its computational cost.

While evaluating $\mathbf{x}$ at a fidelity $\mathbf{s}$ in principle provides observations of $g(\mathbf{x}, \mathbf{s}')$ at all $\mathbf{s}' \in T(\mathbf{s})$, we choose to retain and include in our inference only a subset of the observed fidelities $S \subseteq T(\mathbf{s})$. This reduces computational overhead in GP regression. In our numerical experiments, we take the cardinality of S to be either 2 or 3, though the approach also allows increased cardinality.

With this in mind, the taKG acquisition function at a point $\mathbf{x}$ and set of fidelities S at time n is

$$\text{taKG}_n(\mathbf{x}, \mathcal{S}) := \frac{L_n(\emptyset) - L_n(\mathbf{x}, S)}{c_n(\mathbf{x}, \max S)},$$

where we also refer to the numerator as the *value of information* (Howard, 1966), $\text{VOI}_n(\mathbf{x}, S) := L_n(\emptyset) - L_n(\mathbf{x}, S)$. Thus, taKG quantifies the value of information per unit cost of sampling.

We model the cost of observing at all fidelity vectors in S to be the cost of evaluating g at a single fidelity vector equal to the elementwise maximum, $\max \mathcal{S} := (\max_{\mathbf{s} \in \mathcal{S}} s_i : 1 \le i \le m)$. This is the least expensive fidelity at which we could observe S.

The taKG algorithm chooses to sample at the point $\mathbf{x}$, fidelity $\mathbf{s}$, and additional lower-fidelity point(s) $S \setminus \{\mathbf{s}\}$ to retain that jointly maximize the taKG acquisition function, among all fidelity sets S with limited cardinality ℓ.

$$\max_{\mathbf{x}, \mathbf{s}, S: S \subseteq T(\mathbf{s}), |S| = \ell, \mathbf{s} \in S} \text{taKG}_n\left(\mathbf{x}, S\right). \tag{2.1}$$

This is a continuous optimization problem whose decision variable is described by $d + \ell m_1 + m_2$ real numbers. d describe $\mathbf{x}$, $m = m_1 + m_2$ describe $\mathbf{s}$, and $(\ell - 1)m_1$ describe $S \setminus \{\mathbf{s}\}$.

2.4 0-avoiding taKG (taKG$^{\emptyset}$)

The taKG acquisition function uses the value of information per unit cost of sampling. When the value of information and cost become small simultaneously, as when we shrink training iterations or training data to 0, this ratio becomes sensitive to misspecification of the GP model on g. We first discuss this issue, and then develop a version of taKG for these settings.

To understand this issue, we first observe in Proposition 1 that the value of information for sampling $g(\mathbf{x}, \mathbf{s})$, for any $\mathbf{s}$, is strictly positive when the kernel has strictly positive entries. Proofs for all results are in the supplement.

Proposition 1 *If the kernel function $K_n((\mathbf{x}, \mathbf{s}), (\mathbf{x}', 1)) > 0$ for any $\mathbf{x}, \mathbf{x}' \in A$, then for any $\mathbf{x} \in A$ and any $\mathbf{s} \in [0, 1]^m$, $\text{VOI}_n(\mathbf{x}, \{\mathbf{s}\}) > 0$.*

Proposition 1 holds even if $\mathbf{s}$ has some or all components set to 0. Thus, if the estimated cost at such extremely low fidelities is small relative to the (strictly positive) value of information there, taKG may be drawn to sample them, even though the value of information is small. We may even spend a substantial portion of our budget evaluating $g(\mathbf{x}, \mathbf{0})$ at different $\mathbf{x}$. This is usually undesirable.

For example, with training iterations as our single fidelity, fidelity 0 corresponds to training a machine learning model with no training iterations. This would return the validation error on initial model parameter estimates. While this likely provides *some* information about the validation error of a fully trained model, specifying a kernel over g that productively uses this information from a large number of hyperparameter sets $\mathbf{x}$ would be challenging.

This issue becomes even more substantial when considering training iterations and training data together: cost nearly vanishes as *either* fidelity vanishes, creating many fidelities at each $\mathbf{x}$ that we may be drawn to oversample. This issue also harms entropy search methods (Klein et al., 2017a; McLeod et al., 2017; Klein et al., 2017b) using the ratio of information gain to cost.

To deal with this issue, Klein et al. (2017a); McLeod et al. (2017) artificially inflate the cost of evaluating at fidelity 0 to penalize low fidelity evaluations. Similarly, Klein et al.

(2017b) recommends adding a fixed cost to all evaluations motivated by the overhead of optimizing the acquisition function, but then recommends setting this to the same order of magnitude as a full-fidelity evaluation even though the overhead associated with optimizing a BO acquisition function using well-written code and efficient methodology will usually be substantially smaller. As a result, any fixed cost must be tuned to the application setting to avoid oversampling at excelssively small fidelities while still allowing sampling at moderate fidelities.

We propose an alternate solution that works well without tuning when the cost of evaluation becomes small as the smallest component in $\mathbf{s}$ approaches 0.

We first define $Z(\mathbf{s}) = \cup_{i=1}^m \{\mathbf{s}' : s'_i = 0, s'_j = s_i \; \forall j \neq i\}$ to be the set of fidelities obtained by replacing one component of $\mathbf{s}$ by 0. (Z stands for "zero".) We then let $Z(S) = \cup_{\mathbf{s}\in S} Z(\mathbf{s})$. For example, suppose s_1 is a trace fidelity (say, training iterations), s_2 is a non-trace fidelity (say, training data size), and $S = \{(1/2, 1), (1, 1)\}$, corresponding to an evaluation of g at $\mathbf{s} = (1, 1)$ and retention of the point $(1/2, 1)$ from the trace $T((1, 1))$. Then $Z(S) = \{(0, 1), (1/2, 0), (1, 0)\}$.

Fidelities in $Z(S)$ are extremely inexpensive to evaluate and provide extremely small but strictly positive value of information. We wish to avoid sampling these fidelities even when their taKG is large.

To accomplish this, we modify our value of information $\text{VOI}_n(\mathbf{x}, S) = L_n(\emptyset) - L_n(\mathbf{x}, S)$ to suppose free observations $\mathbf{y}(\mathbf{x}, \mathbf{s}')$ will be provided of these problematic low-fidelity $\mathbf{s}'$. Our modified value of information will suppose these free observations will be provided to both the benchmark, previously set to $L_n(\emptyset)$, and to the reduced expected loss, previously set to $L_n(\mathbf{x}, S)$, achieved through observing $\mathbf{x}$ at fidelities S. The resulting modified value of information is

$$\text{VOI}^{\emptyset}_n(\mathbf{x}, S) = L_n(\mathbf{x}, Z(S)) - L_n(\mathbf{x}, S \cup Z(S))$$

We emphasize our algorithm will not evaluate g at fidelities in $Z(S)$. Instead, it will simulate these evaluations according to the algorithm in §2.2.

We define the taKG$^{\emptyset}$ acquisition function using this modified value of information as

$$\text{taKG}^{\emptyset}_n(\mathbf{x}, S) = \frac{\text{VOI}^{\emptyset}_n(\mathbf{x}, S)}{c_n(\mathbf{x}, \max S)}. \tag{2.2}$$

To find the point $\mathbf{x}$ and fidelity $\mathbf{s}$ to sample, we optimize taKG$^{\emptyset}$ over $\mathbf{x}$, fidelity $\mathbf{s}$, and additional lower-fidelity point(s) $S \setminus \{\mathbf{s}\}$ as we did in (2.1).

We refer to this VOI and acquisition function as "0-avoiding," because they place 0 value on fidelities with any component equal to 0. This prevents sampling at these points as long as the cost of sampling is strictly positive.

Indeed, suppose $\mathbf{s} = \max(S)$ has a component equal to 0. Then each element in S will have one component equal to 0, and $S \subseteq Z(S)$. Then $\text{VOI}^{\emptyset}_n(\mathbf{x}, S) = L_n(\mathbf{x}, Z(S)) - L_n(\mathbf{x}, Z(S) \cup S) = 0$. Moreover, the following proposition shows that if $\mathbf{s} = \max(S)$ has all components strictly positive and additional regularity conditions hold, then $\text{VOI}^{\emptyset}_n(\mathbf{x}, S)$ is also strictly positive.

Proposition 2 *If $\mathbf{s} = \max(S)$ has all components strictly positive, K_n is positive definite, and the hypothesis of Proposition 1 is satisfied for K_n given $\{g(\mathbf{x}, \mathbf{s}) : \mathbf{s} \in Z(S)\}$, then $\text{VOI}^{\emptyset}_n(\mathbf{x}, S)$ is strictly positive.*

Thus, taKG$^{\emptyset}$ will never sample at a fidelity $\mathbf{s}$ with a 0 component. Additionally, under other regularity conditions (see Corollary 1 in the supplement), $\text{VOI}^{\emptyset}_n(\mathbf{x}, S)$ is continuous in S, and so the property that $\text{VOI}^{\emptyset}_n(\mathbf{x}, S) = 0$ when a component of $\mathbf{s} = \max(S)$ is 0 also discourages sampling at $\mathbf{s}$ whose smallest component is *close* to 0.

2.5 Efficiently Maximizing taKG and taKG$^{\emptyset}$

The taKG and taKG$^{\emptyset}$ acquisition functions are defined in terms of a hard-to-calculate function $L_n(\mathbf{x}, S)$. Here, we describe how to efficiently maximize them using SGA with multiple restarts. The heart of this method is a simulation-based procedure for simulating a stochastic gradient of $L_n(\mathbf{x}, S)$, i.e., a random variable whose expectation is the gradient of $L_n(\mathbf{x}, S)$ with respect to $\mathbf{x}$ and the elements of S.

To construct this procedure, we first provide a more explicit expression for $L_n(\mathbf{x}, S)$. Because $L_n(\mathbf{x}, S)$ is the expectation of the minimum over $\mathbf{x}'$ of $\mathbb{E}_n[g(\mathbf{x}', \mathbf{1}) \mid \mathbf{y}(\mathbf{x}, S)]$, we begin with the distribution of this conditional expectation for a fixed $\mathbf{x}'$ under $\mathbb{P}_n$.

This conditional expectation can be calculated with GP regression from previous observations, the new point $\mathbf{x}$ and fidelities S, and the observations $\mathbf{y}(\mathbf{x}, S)$. This conditional expectation is linear in $\mathbf{y}(\mathbf{x}, S)$.

Moreover, $\mathbf{y}(\mathbf{x}, S)$ is the sum of $(g(\mathbf{x}, \mathbf{s}) : \mathbf{s} \in S)$ (which is multivariate normal under the posterior) and optional observational noise (which is independent and normally distributed), and so is itself multivariate normal. As a multivariate normal random variable, it can be written as the sum of its mean vector and the product of the Cholesky decomposition of its covariance matrix with an independent standard normal random vector, call it $\mathbf{w}$. (The coefficients of this mean vector and covariance matrix may depend on $\mathbf{x}$, S, and previously observed

data.) The dimension of $\mathbf{w}$ is the number of components in the observation, $|S|$.

Thus, the conditional expected value of the objective $\mathbb{E}_n\left[g(\mathbf{x}', \mathbf{1}) \mid \mathbf{y}(\mathbf{x}, S)\right]$ is a linear function (through GP regression) of another linear function (through the distribution of the observation) of $\mathbf{w}$.

We also have that the mean of this conditional expectation is $\mathbb{E}_n[\mathbb{E}_n[g(\mathbf{x}, \mathbf{1})|\mathbf{y}(\mathbf{x}, S)]] = \mathbb{E}_n[g(\mathbf{x}, \mathbf{1})] = \mu_n(\mathbf{x})$ by iterated conditional expectation.

These arguments imply the existence of a vector-valued function $\tilde{\boldsymbol{\sigma}}_n(\mathbf{x}', \mathbf{x}, S)$ such that $\mathbb{E}_n[g(\mathbf{x}', \mathbf{1})|\mathbf{y}(\mathbf{x}, S)] = \mu_n(\mathbf{x}') + \tilde{\boldsymbol{\sigma}}_n(\mathbf{x}', \mathbf{x}, S) \cdot \mathbf{w}$ simultaneously for all $\mathbf{x}'$. In the supplement, we show $\tilde{\boldsymbol{\sigma}}_n(\mathbf{x}', \mathbf{x}, S) = K_n\left((\mathbf{x}', 1), (\mathbf{x}, S)\right)(C_n^T)^{-1}$ where $(\mathbf{x}, S) := \{(\mathbf{x}, \mathbf{s}) : \mathbf{s} \in S\}$ (abusing notation), and C_n is the Cholesky factor of the covariance matrix $K_n\left((\mathbf{x}, S), (\mathbf{x}, S)\right) + \sigma^2 I$. Thus,

$$L_n\left(\mathbf{x}, S\right) = \mathbb{E}_n\left[\min_{\mathbf{x}'} \mu_n\left(\mathbf{x}', \mathbf{1}\right) + \tilde{\boldsymbol{\sigma}}_n\left(\mathbf{x}', \mathbf{x}, S\right) \cdot \mathbf{w}\right].$$

We now take the gradient of $L_n(\mathbf{x}, S)$ with respect to $\mathbf{x}$ and the components of S, keeping the number of these components fixed. Given regularity conditions stated below, this gradient $\nabla_{\mathbf{x},S} L_n(\mathbf{x}, S)$ is

$$\begin{aligned}
&\nabla_{\mathbf{x},S}\, \mathbb{E}_n\left[\min_{\mathbf{x}'}\left(\mu_n\left(\mathbf{x}', \mathbf{1}\right) + \tilde{\boldsymbol{\sigma}}_n\left(\mathbf{x}', \mathbf{x}, S\right) \cdot \mathbf{w}\right)\right] \\
&= \mathbb{E}_n\left[\nabla_{\mathbf{x},S} \min_{\mathbf{x}'}\left(\mu_n\left(\mathbf{x}', \mathbf{1}\right) + \tilde{\boldsymbol{\sigma}}_n\left(\mathbf{x}', \mathbf{x}, S\right) \cdot \mathbf{w}\right)\right] \\
&= \mathbb{E}_n\left[\nabla_{\mathbf{x},S}\, \left(\mu_n\left(\mathbf{x}^*, \mathbf{1}\right) + \tilde{\boldsymbol{\sigma}}_n\left(\mathbf{x}^*, \mathbf{x}, S\right) \cdot \mathbf{w}\right)\right] \\
&= \mathbb{E}_n\left[\nabla_{\mathbf{x},S}\, \tilde{\boldsymbol{\sigma}}_n\left(\mathbf{x}^*, \mathbf{x}, S\right) \cdot \mathbf{w}\right],
\end{aligned}$$

where $\mathbf{x}^*$ is a global minimum (over $\mathbf{x}' \in A$) of $\mu_n(\mathbf{x}', \mathbf{1}) + \tilde{\boldsymbol{\sigma}}_n(\mathbf{x}', \mathbf{x}, S) \cdot \mathbf{w}$, and the gradient in the third and fourth lines are taken holding $\mathbf{x}^*$ fixed even though in reality its value depends on $\mathbf{x}$. Here, the interchange of expectation and the gradient is justified using infinitessimal perturbation analysis (L'Ecuyer, 1990) and ignoring the dependence of $\mathbf{x}^*$ on $\mathbf{x}$ and S when taking the gradient is justified by the envelope theorem (Milgrom and Segal, 2002). Theorem 1 formalizes this.

Theorem 1 *Suppose A is compact, μ_0 is constant, K_0 is continuously differentiable, and* $\operatorname{argmin}_{\mathbf{x}' \in A} \mu_n(\mathbf{x}', 1) + \tilde{\boldsymbol{\sigma}}_n\left(\mathbf{x}', \mathbf{x}, S\right) \cdot \mathbf{w}$ *contains a single element $\mathbf{x}^*$ (that depends on $\mathbf{w}$) almost surely. Then $\nabla_{\mathbf{x},S} L_n(\mathbf{x}, S) = \mathbb{E}_n\left[\nabla_{\mathbf{x},S}\, \tilde{\boldsymbol{\sigma}}(\mathbf{x}^*, \mathbf{x}, S) \cdot \mathbf{w}\right]$.*

With this result in place, we can obtain an unbiased estimator of $\nabla_{\mathbf{x},S} L_n(\mathbf{x}, S)$ by simulating $\mathbf{w}$, calculating $\mathbf{x}^*$, and then returning $\nabla_{\mathbf{x},S}\, \tilde{\boldsymbol{\sigma}}_n(\mathbf{x}^*, \mathbf{x}, S) \cdot \mathbf{w}$. Using this, together with the chain rule and an exact gradient calculation for $c_n(\mathbf{x}, \max S)$, we can then compute stochastic gradients for taKG and taKG$^{\emptyset}$.

We then use this stochastic gradient estimator within SGA (Kushner and Yin, 2003) to solve the optimization problem (2.1) (or the equivalent problem for maximizing taKG$^{\emptyset}$). The following theorem shows that, under the right conditions, a SGA algorithm converges almost surely to a critical point of taKG$^{\emptyset}$.

Theorem 2 *Assume the conditions of Theorem 1, A is a compact hyperrectangle, and $c_n(\max \mathcal{S})$ is continuously differentiable and bounded below by a strictly positive constant. In addition, assume that we optimize taKG$^{\emptyset}$ using a SGA method with the stochastic gradient from Theorem 1 whose stepsize sequence $\{\epsilon_t : t = 0, 1, \ldots\}$ satisfies $\epsilon_t \to 0$, $\epsilon_t \geq 0$, $\sum_t \epsilon_t = \infty$ and $\sum_t \epsilon_t^2 < \infty$. Then the sequence of points $\{\mathbf{x}_t, S_t\}_{t \geq 0}$ from SGA converges almost surely to a connected set of stationary points of taKG$^{\emptyset}$.*

2.6 Warm-starting from Partial Runs

When tuning hyperparameters using training iterations as a fidelity, we can cache the state of training after $\mathbf{s}$ iterations, for a warm start, and then continue training later up to a larger number of iterations $\mathbf{s}'$ for less than $\mathbf{s}'$ training iterations would cost at a new $\mathbf{x}$.

We assume trace fidelities can be "warm-started" while non-trace fidelities cannot. We also assume the incremental cost of evaluting fidelity vector $\mathbf{s}'$ warm-starting from $\mathbf{s}$ is the difference in the "cold-start" evaluation costs. We model the cost of cold-start evaluation as in §2.1 with a Gaussian process on log(cost). To obtain training data for this model, costs observed from warm-started evaluations are summed with those of the previous evaluations they continue to approximate what the cold-start cost would be. We set $c_n(\mathbf{x}, \mathbf{s})$ to be the difference in estimated cold-start costs if a previous evaluation would allow warm starting, and to the estimated cold start cost if not.

While our approach to choosing $\mathbf{x}$ and $\mathbf{s}$ to evaluate is to optimize taKG$^{\emptyset}$ as before, our computational approach from §2.5 (Theorem 2) required that $c_n(\mathbf{x}, \mathbf{s})$ be continuously differentiable. This requirement is not met. To address this, we modify how we optimize taKG$^{\emptyset}$. This modified approach also works for taKG.

First, we maintain a basket of size at most b of previously evaluated point, fidelity pairs, $(\mathbf{x}(j), \mathbf{s}(j))$. For each $j \leq b$, we optimize taKG$^{\emptyset}_n(\mathbf{x}(j), S)$ letting S vary over those sets satisfying: (1) $|S| = \ell$; (2) $\mathbf{s}' \geq \mathbf{s}(j)$ componentwise for each $\mathbf{s}' \in S$, with equality for non-trace fidelity components. We can use the method from §2.5 because, over this set, $c_n(\mathbf{x}(j), S)$ is continuously differentiable in S.

We also optimize taKG$^{\emptyset}_n(\mathbf{x}, S)$ over all S with $|S| = \ell$

and all $\mathbf{x}$ (not restricting to $\mathbf{x}$ to previously evaluated points), setting $c_n(\mathbf{x}, S)$ to the cost-start cost.

Among the solutions to these at most $b + 1$ optimization problems, we select the $\mathbf{x}$ and S with the largest $\text{taKG}_n^{\emptyset}(\mathbf{x}, S)$ and evaluate g at $\mathbf{x}$ and $\max(S)$.

We then update our basket. We first add the $\mathbf{x}$ and $\max(S)$ produced by the optimization not constraining $\mathbf{x}$. If the basket size exceeds b, we then remove the $\mathbf{x}$ and $\mathbf{s}$ whose optimization over $\text{taKG}_n^{\emptyset}$ produced the smallest value. In practice, we set $b = 10$.

2.7 Batch and Derivative Evaluations

taKG and $\text{taKG}^{\emptyset}$ generalize naturally following Wu and Frazier (2016) and Wu et al. (2017) to batch settings where we can evaluate multiple point, fidelity pairs simultaneously and derivative-enabled settings where we observe gradients. The batch version uses the same acquisition functions taKG and $\text{taKG}^{\emptyset}$ defined above, but optimizes over a *set* of values for $\mathbf{x}$ and $\mathbf{s}$, each of which has an associated $S \in T(\mathbf{s})$ of limited cardinality. In the derivative-enabled setting, we incorporate (optionally noisy) gradient observations into our posterior distribution directly through GP regression. We also generalize the taKG and $\text{taKG}^{\emptyset}$ acquisition functions to allow inclusion of gradients of the objective in the set of quantities observed $\mathbf{y}(\mathbf{x}, S)$ in the definition of $L_n(\mathbf{x}, S)$.

3 NUMERICAL EXPERIMENTS

We compare sequential, batch, and derivative-enabled $\text{taKG}^{\emptyset}$ with benchmark algorithms on synthetic optimization problems (Sect. 3.1), hyperparameter optimization of neural networks (Sect. 3.2), and hyperparameter optimization for large-scale kernel learning (Sect. 3.3). The synthetic and neural network benchmarks use fidelities with trace observations, while the large-scale kernel learning benchmark does not. We integrate over GP hyperparameters by sampling 10 sets of values using the `emcee` package (Foreman-Mackey et al., 2013). We run algorithms to a maximum number of iterations, using a smaller cutoff if an algorithm fails to make progress.

3.1 Optimizing Synthetic Functions

Here, we compare $\text{taKG}^{\emptyset}$ against both the sequential and batch versions of the single-fidelity algorithms KG (Wu and Frazier, 2016; Wu, 2017) and EI (Jones et al., 1998; Wang et al., 2016), a derivative-enabled single-fidelity version of KG (Wu et al., 2017), Hyperband (Li et al., 2018), and the multi-fidelity method BOCA (Kandasamy et al., 2017). BOCA is the only previous method of which we are aware that treats multiple continuous fidelities. We do not compare against FaBOLAS (Klein et al., 2017a, 2015) because the kernel it uses is specialized to neural network hyperparameter tuning.

Following experiments in Kandasamy et al. (2017), we augment four synthetic test functions, 2-d Branin, 3-d Rosenbrock, 3-d Hartmann, and 6-d Hartmann, by adding one or two fidelity controls, as described in the supplement. We set the cost of an individual evaluation of $\mathbf{x}$ at fidelity $\mathbf{s}$ to $0.01 + \prod_i s_i$. Thinking of s_1 as mimicking training data and s_2 training iterations, the term $\prod_i s_i$ mimics a cost of training that is proportional to the number of times a datapoint is visited in training. The term 0.01 mimics a fixed cost associated with validating a trained model. We set the cost of a batch evaluation to the maximum of the costs of the individual evaluations, to mimic wall-clock time for running evaluations synchronously in parallel.

Fig. 1 shows results. We run versions of $\text{taKG}^{\emptyset}$ using $|S|$ set to 2 (taKG_0 2-points) and 3 (taKG_0 3-points). Hyperband is implemented using values from Li et al. (2018): $R = 1.01$, which is the maximum resource that can be consumed by a single fidelity; $\eta = 3$, the default from Algorithm 1; and $s_{\max} = 4$, the value used in Table 1.

Sequential $\text{taKG}^{\emptyset}$ performs well relative to sequential competitors (EI, KG, BOCA) for both values of $|S|$. Batch $\text{taKG}^{\emptyset}$ with batch size 8 performs well relative to batch competitors (EI, KG, Hyperband). (We consider Hyperband to be a batch method although the parallelism it leverages varies during its operation.) Using the larger $|S|$ improves the performance of $\text{taKG}^{\emptyset}$.

3.2 Optimizing Hyperparameters of Neural Nets

Here, we evaluate on hyperparameter optimization of neural networks. Benchmarks include the single-fidelity Bayesian optimization algorithms KG (Wu and Frazier, 2016) and EI (Jones et al., 1998), their batch versions, and the state-of-art hyperparameter tuning algorithms HyperBand and FaBOLAS.

$\text{taKG}^{\emptyset}$ uses two fidelity controls: the training set size and the number of training iterations. FaBOLAS is specifically designed to use the training set size as its fidelity control. We implement Hyperband also using the training set size as its fidelity (it can use only one fidelity). We run Hyperband using the parameters described in §3.1. Because Hyperband did not perform as well as expected, we also performed experiments using $s_{\max} = 2$ and $s_{\max} = 1$. $s_{\max} = 1$ is equivalent to random search, and is labeled so in plots.

Following Li et al. (2018), we set the cost to the number

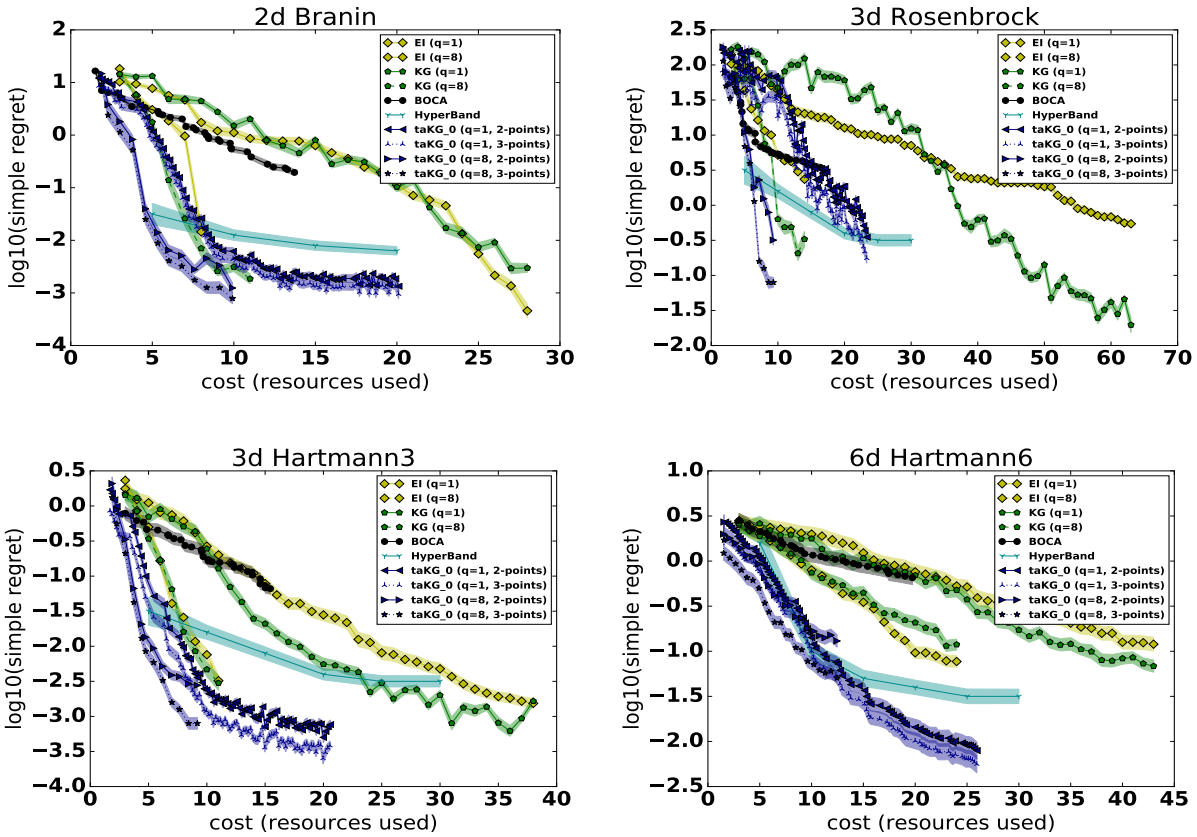

Figure 1: Optimizing synthetic functions: Plots show simple regret over 40 independent runs for synthetic functions with trace observations and one or two continuous fidelity controls for 2-d Branin, 3-d Rosenbrock, 3-d Hartmann, and 6-d Hartmann problems. q indicates batch size for fixed batch-size methods. taKG0 outperforms competitors in both sequential and batch settings.

of training examples used in training, normalized by the number used in training with full fidelity. For example, if full fidelity uses 10^5 training examples per epoch over 100 epochs, then the cost of evaluating a set of hyperparameters using 10^4 sub-sampled training examples per epoch over 10 epochs is $10^4 \times 10/(10^5 \times 100) = 0.01$.

Feedforward Neural Nets on MNIST We tune a fully connected two-layer neural network on MNIST. The maximum number of epochs allowed is 20. We optimize 5 hyperparameters: learning rate, dropout rate, batch size and the number of units at each layer.

Fig. 2 shows that sequential taKG$^{\emptyset}$ outperforms the sequential methods KG, EI and the multi-fidelity hyperparameter optimization algorithm FaBOLAS. taKG$^{\emptyset}$ with batch size 4 substantially improves over batch versions of KG and EI and the batch method Hyperband.

Convolutional Neural Nets on CIFAR-10 and SVHN We tune convolution neural networks (CNNs) on CIFAR-10 and SVHN. Our CNN consists of 3 convolutional blocks and a softmax classification layer. Each convolutional block consists of two convolutional layers with the same number of filters followed by a max-pooling layer. There is no dropout or batch-normalization layer. We split the CIFAR-10 dataset into 40,000 training samples, 10,000 validation samples and 10000 test samples. We split the SVHN training dataset into 67,235 training samples and 6,000 validation samples, and use the standard 26,032 test samples. We apply standard data augmentation: horizontal and vertical shifts, and horizontal flips. We optimize 5 hyperparameters to minimize the classification error on the validation set: the learning rate, batch size, and number of filters in each convolutional block. We set the maximum number of training epochs for all algorithms to 50 for CIFAR-10 and 40 for SVHN. Because of the computational expense of training CNNs, we leave out some benchmarks, dropping the single-fidelity method EI in favor of the structurally similar single-fidelity method KG, and performing batch evaluations for only some methods.

Fig. 2 shows that sequential taKG$^{\emptyset}$ outperforms its competitors (including the batch method Hyperband) on both problems. Using batch evaluations with taKG$^{\emptyset}$ on CIFAR-

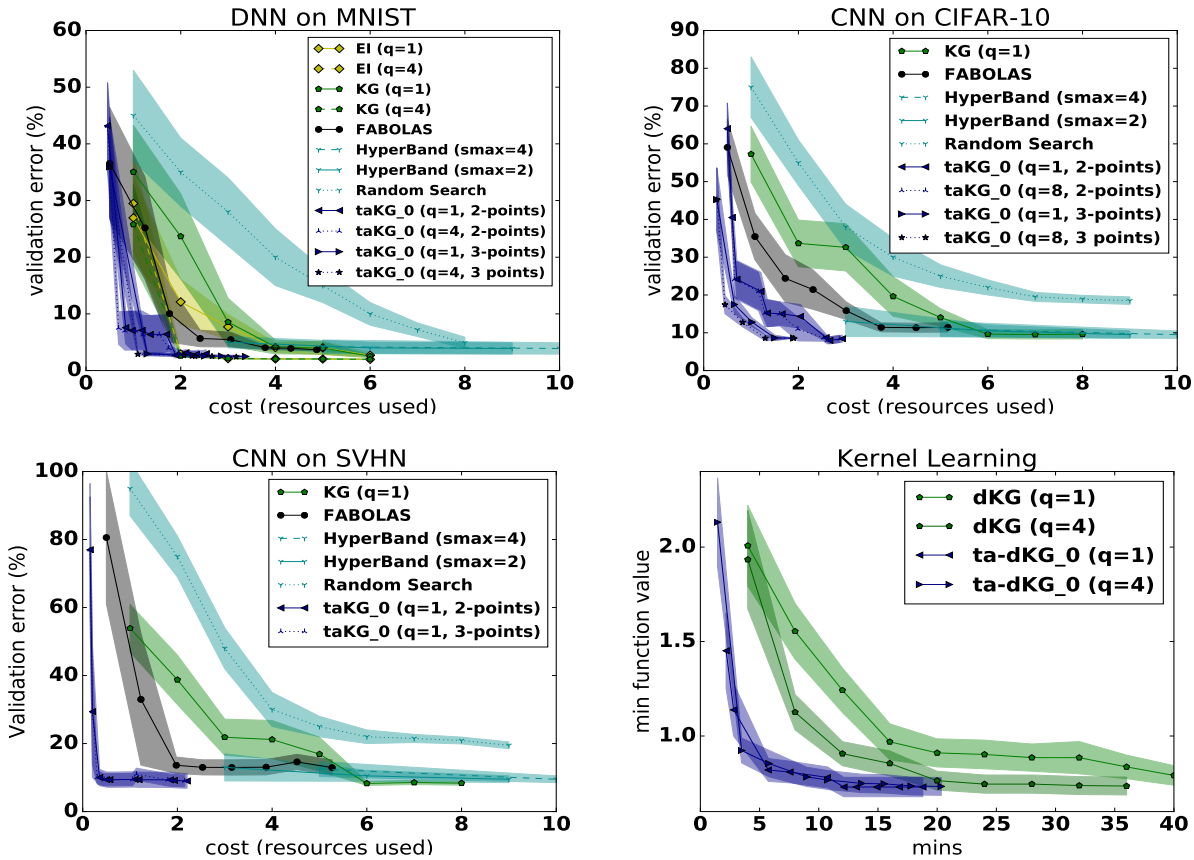

Figure 2: We show the validation error for tuning feedforward neural networks on MNIST (each with 20 runs); tuning convolutional neural networks on CIFAR-10 and SVHN (each with 10 runs); for KISS-GP kernel learning we show -log marginal likelihood divided by the number of datapoints. q indicates batch size for fixed batch-size methods. taKG$^{\emptyset}$ outperforms competitors in both sequential and batch settings.

10 improves performance even further. When we train using optimized hyperparameters on the full training dataset for 200 epochs, test data classification error is $\sim 12\%$ for CIFAR-10 and $\sim 5\%$ for SVHN.

3.3 Optimizing Hyperparameters for Large-scale Kernel Learning

We test derivative-enabled taKG$^{\emptyset}$ (ta-dKG$^{\emptyset}$) in large-scale kernel learning: the 1-d demo for KISS-GP (Wilson and Nickisch, 2015) on the GPML website (Rasmussen and Nickisch, 2016). In this example, we optimize 3 hyperparameters (marginal variance, length scale, and variance of the noise) of a GP with an RBF kernel on 1 million training points to maximize the log marginal likelihood. We evaluate both the log marginal likelihood and its gradient using the KISS-GP framework. We use two continuous fidelity controls: the number of training points and the number of inducing points. We set the maximum number of inducing points to $m = 1000$.

We compare ta-d-KG to the derivative-enabled knowledge gradient (d-KG) (Wu et al., 2017), using both algorithms in the sequential setting and with a batch size of 4. We leave out methods unable to utilize derivatives (including Hyperband and FaBOLAS), as these are likely to substantially underperform. Fig. 2 shows ta-dKG$^{\emptyset}$ finds a good solution faster than d-KG in sequential and batch settings.

4 CONCLUSION

We propose a novel multi-fidelity acquisition function, the trace aware knowledge-gradient, that leverages special structure provided by trace observations, is able to handle multiple simultaneous continuous fidelities, and generalizes to batch and derivative settings. This acquisition function uses traces to outperform state-of-the-art hyperparameter tuning algorithms.

Acknowledgements

PIF was supported by NSF CAREER CMMI-1254298, NSF CMMI-1536895, and AFOSR FA9550-15-1-0038. AGW was supported by NSF IIS-1563887, an Amazon Research Award and a Facebook Research Award.

References

Bergstra, J. and Bengio, Y. (2012). Random search for hyper-parameter optimization. *Journal of Machine Learning Research*, 13(Feb):281–305.

Domhan, T., Springenberg, J. T., and Hutter, F. (2015). Speeding up automatic hyperparameter optimization of deep neural networks by extrapolation of learning curves. In *International Joint Conferences on Artificial Intelligence*.

Falkner, S., Klein, A., and Hutter, F. (2018). BOHB: Robust and efficient hyperparameter optimization at scale. In *International Conference on Machine Learning*, pages 1436–1445.

Foreman-Mackey, D., Hogg, D. W., Lang, D., and Goodman, J. (2013). emcee: the MCMC hammer. *Publications of the Astronomical Society of the Pacific*, 125(925):306.

Frazier, P., Powell, W., and Dayanik, S. (2009). The knowledge-gradient policy for correlated normal beliefs. *INFORMS Journal on Computing*, 21(4):599–613.

Frazier, P. I. (2018). A tutorial on Bayesian optimization. *arXiv preprint arXiv:1807.02811*.

Garrido-Merchán, E. C. and Hernández-Lobato, D. (2018). Dealing with categorical and integer-valued variables in bayesian optimization with gaussian processes. *arXiv preprint arXiv:1805.03463*.

Hennig, P. and Schuler, C. J. (2012). Entropy search for information-efficient global optimization. *Journal of Machine Learning Research*, 13(Jun):1809–1837.

Hernández-Lobato, J. M., Hoffman, M. W., and Ghahramani, Z. (2014). Predictive entropy search for efficient global optimization of black-box functions. In *Advances in Neural Information Processing Systems*.

Howard, R. (1966). Information Value Theory. *IEEE Transactions on Systems Science and Cybernetics*, 2(1):22–26.

Huang, D., Allen, T., Notz, W., and Miller, R. (2006). Sequential kriging optimization using multiple-fidelity evaluations. *Structural and Multidisciplinary Optimization*, 32(5):369–382.

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

Kandasamy, K., Dasarathy, G., Schneider, J., and Poczos, B. (2017). Multi-fidelity Bayesian optimisation with continuous approximations. In *International Conference on Machine Learning*.

Klein, A., Bartels, S., Falkner, S., Hennig, P., and Hutter, F. (2015). Towards efficient Bayesian optimization for big data. In *NIPS 2015 Bayesian Optimization Workshop*.

Klein, A., Falkner, S., Bartels, S., Hennig, P., and Hutter, F. (2017a). Fast Bayesian optimization of machine learning hyperparameters on large datasets. In *Artificial Intelligence and Statistics*.

Klein, A., Falkner, S., Mansur, N., and Hutter, F. (2017b). RoBO: A flexible and robust Bayesian optimization framework in python. In *NIPS 2017 Bayesian Optimization Workshop*.

Kushner, H. and Yin, G. G. (2003). *Stochastic Approximation and Recursive Algorithms and Applications*, volume 35. Springer Science & Business Media.

Lam, R., Allaire, D., and Willcox, K. (2015). Multifidelity optimization using statistical surrogate modeling for non-hierarchical information sources. In *56th AIAA/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, page 0143.

L'Ecuyer, P. (1990). A unified view of the IPA, SF, and LR gradient estimation techniques. *Management Science*, 36(11):1364–1383.

Li, L., Jamieson, K., DeSalvo, G., Rostamizadeh, A., and Talwalkar, A. (2018). Hyperband: A novel bandit-based approach to hyperparameter optimization. *Journal of Machine Learning Research*.

McLeod, M., Osborne, M. A., and Roberts, S. J. (2017). Practical Bayesian optimization for variable cost objectives. *arXiv preprint arXiv:1703.04335*.

Milgrom, P. and Segal, I. (2002). Envelope theorems for arbitrary choice sets. *Econometrica*, 70(2):583–601.

Mockus, J. (1989). *The Bayesian approach to local optimization*. Springer.

Poloczek, M., Wang, J., and Frazier, P. I. (2017). Multi-information source optimization. In *Advances in Neural Information Processing Systems*.

Rasmussen, C. E. and Nickisch, H. (2016). Documentation for GPML Matlab code version 4.2. http://www.gaussianprocess.org/gpml/code/matlab/doc/, accessed 2019-06-30.

Snoek, J., Larochelle, H., and Adams, R. P. (2012). Practical Bayesian optimization of machine learning algorithms. In *Advances in Neural Information Processing Systems*.

Swersky, K., Snoek, J., and Adams, R. P. (2014). Freeze-thaw Bayesian optimization. *arXiv preprint arXiv:1406.3896*.

Wang, J., Clark, S. C., Liu, E., and Frazier, P. I. (2016). Parallel Bayesian global optimization of expensive functions. *arXiv preprint arXiv:1602.05149*.

Wilson, A. G. and Nickisch, H. (2015). Kernel interpolation for scalable structured Gaussian processes (KISS-GP). In *International Conference on Machine Learning*.

Wu, J. (2017). *Knowledge Gradient Methods for Bayesian Optimization*. PhD thesis, Cornell University.

Wu, J. and Frazier, P. (2016). The parallel knowledge gradient method for batch Bayesian optimization. In *Advances in Neural Information Processing Systems*.

Wu, J., Poloczek, M., Wilson, A. G., and Frazier, P. I. (2017). Bayesian optimization with gradients. In *Advances in Neural Information Processing Systems*.