Soft Computing (2021) 25:12591–12603
https://doi.org/10.1007/s00500-021-05684-7

FOCUS

Green machine learning via augmented Gaussian processes and multi-information source optimization

Antonio Candelieri[1] · Riccardo Perego[2] · Francesco Archetti[2]

Accepted: 9 February 2021 / Published online: 10 March 2021

Abstract
Searching for accurate machine and deep learning models is a computationally expensive and awfully energivorous process. A strategy which has been recently gaining importance to drastically reduce computational time and energy consumed is to exploit the availability of different information sources, with different computational costs and different "fidelity," typically smaller portions of a large dataset. The multi-source optimization strategy fits into the scheme of Gaussian Process-based Bayesian Optimization. An Augmented Gaussian Process method exploiting multiple information sources (namely, AGP-MISO) is proposed. The Augmented Gaussian Process is trained using only "reliable" information among available sources. A novel acquisition function is defined according to the Augmented Gaussian Process. Computational results are reported related to the optimization of the hyperparameters of a Support Vector Machine (SVM) classifier using two sources: a large dataset—the most expensive one—and a smaller portion of it. A comparison with a traditional Bayesian Optimization approach to optimize the hyperparameters of the SVM classifier on the large dataset only is reported.

Keywords Green AI · Green machine learning · Multi information source optimization · Bayesian optimization · Gaussian processes

1 Introduction

1.1 The Green AI challenge

Machine Learning (ML) models are computationally hungry: this is particularly true in the case of Deep Neural Networks (DNNs) in fields like computer vision (Bianco et al. 2020) and Natural Language Processing (NLP) (Kulkarni and Shivananda 2019): an approximate quantification of the financial and environmental costs for training and validating some of the neural network models in the NLP domain is reported in Strubell et al. (2019) and Hao (2019) showing the amazing amount of energy consumed for training and validating a neural network model for NLP, which can generate the emission of an amount of carbon dioxide approximately five times the lifetime emissions of an average American car. No surprise that Green Machine Learning (Green-ML) and Green Artificial Intelligence (Green AI) (Schwartz et al. 2019; Yang et al. 2020) have recently emerged as new research topics.

This paper is focused on the issue of hyperparameter optimization (HPO), where *hyper-parameters* are all the parameters of a model which are not updated during the learning and are used to configure either the model (e.g., number of layers of a deep neural network, etc.) or characterize the algorithm used in the training phase (learning rate for gradient descent algorithm, etc.) and even to include the choice of optimization algorithm itself and also the data features which are fed into the ML model.

HPO can be regarded as an *optimization outer loop* on top of ML model learning (*inner loop*) to find the set of

Communicated by Marcello Sanguineti.

✉ Antonio Candelieri
antonio.candelieri@unimib.it

Riccardo Perego
riccardo.perego@unimib.it

Francesco Archetti
francesco.archetti@unimib.it

[1] Department of Economics, Management and Statistics, University of Milano-Bicocca, Milan, Italy

[2] Department of Computer Science, Systems and Communication, University of Milano-Bicocca, Milan, Italy

hyperparameters leading to the lowest error on a validation set. This two-tier optimization structure has several implications. First, the evaluation of the objective function of the outer loop is very expensive, as it requires learning a model and evaluating its performance on a validation set. This is usually repeated k times in a k-fold-cross-validation procedure. Moreover, the objective function is unknown and can only be observed pointwise with typically noisy evaluations. Secondly, the average value of the loss function does not reflect the true distribution of the data (which leads to the generalization error) and due to the relatively small size of the validation set, the variance of the average estimate obtained by cross-validation can be high. Ignoring this uncertainty can result in suboptimal configuration of hyperparameters. One must also consider that the performance of the model is evaluated with some error, and thus finding the true optimum with a high precision is usually not critical: this fits nicely into in the Bayesian Optimization (BO) framework that is very sample efficient and yields an acceptable solution with relatively few function evaluations.

The outer loop optimization algorithm can be passive, like grid or pure random search, or "educated" to learn, from previous evaluations, the structure of the objective function, and to actively search where most interesting solutions are. Indeed, BO is a framework to model the learning process and to yield a principled quantification of uncertainty (Frazier 2018; Candelieri and Archetti 2019). BO has become the main approach to handle all the relevant steps in finding an accurate ML model: *Algorithm selection*, *Hyperparameter Optimization*, both recently integrated in the more general setting named CASH: *Combining Algorithm Selection and Hyperparameter optimization* (Kotthoff et al. 2017). This led to the definition of Automated Machine/Deep Learning (AutoML/AutoDL) (Hutter et al. 2019) and Neural Architecture Search (NAS) (Hutter et al. 2019; Lindauer and Hutter 2019), showing that different algorithms and values of its hyperparameters can result in significantly different performances (Wolpert 2002; Melis et al. 2017).

Although the *active learning* inherent in BO and the ensuing sample efficiency are usually associated with the search for the best algorithm and its configuration (Shahriari et al. 2016), in terms of accuracy, they translate into significant cost and energy savings. For instance, the BERT (Bidirectional Encoder Representation from Transformer) model, now available in the Google Cloud, aimed at contextual representation in NLP, can require 4 days training sessions (with 110 million of DNN's parameters to be learned) (Strubell et al. 2019) which makes the NAS performed in the outer loop awfully expensive. Sample efficiency requires some assumption on the objective function and a model of learning from observations.

Probabilistic models commonly used in BO are Gaussian Processes (GPs) (Williams and Rasmussen 2006) and Random Forests (RFs) (Ho 1995) (here we do not discuss their relative merits in different problem classes). GPs are a powerful framework for reasoning about an unknown function f given partial knowledge of its behavior obtained through function evaluations. GP leverages a principled estimate of predictive uncertainty toward a careful balance of *exploration* (increasing one's knowledge about f) and *exploitation* (focusing on the best points found so far).

The global hyperparameter optimization problem is usually defined as:

$$\min_{x \in \mathcal{X} \subset \mathbb{R}^d} f(x) \tag{1}$$

where the search space $\mathcal{X}$ is generally box-bounded, f is the loss function and x are the values of the hyperparameters. We remark that f is analytically unknown (also called *latent*) and only pointwise, usually noisy, evaluations can be obtained by querying it. We refer to this situation as black-box optimization.

BO leverages the fact that conditioning the GP on previous observations provides versatile regressors of the objective function. BO starts from a GP prior over f, encoded with parametric mean and kernel. The available observations are used to build the posterior distribution which is used to determine the learning policy, balancing exploration (high GP variance) and exploitation (low GP mean value).

Given the cost of evaluating the objective function, trial-and-error methods like random or grid search are not useful. Compared to a simple grid search, BO can identify a better solution for HPO, given the same number of configurations to evaluate. Given its modeling flexibility, BO can build a relatively cheap probabilistic surrogate of f, take advantage of related tasks (Swersky et al. 2013) or use problem specific priors (De Ath et al. 2020).

The strategy we follow here is to mitigate the high cost of hyperparameter optimization enabling the BO algorithm to trade-off the value of information gained from the evaluation of a hyperparameter configuration against its cost. In Swersky et al. (2013) and Klein et al. (2017) BO is used to evaluate models trained on randomly chosen subsets of data to obtain more, but less informative, evaluations. Two strategies aiming at the same target, which we do not consider here, are *curriculum learning*, which leverages a data-centric view training the model on increasingly larger datasets, and *continuation learning*, which leverages a model-centric view building a sequence of loss functions $L_1 \ldots L_r$, in which each L_{i+1} is more difficult to optimize than L_i and one can view each L_i as a regularized version of L_{i+1} (Aggarwal 2018).

These approaches could be interpreted as optimization problems in which multiple information sources are available, with every source approximating the actual black-box and expensive (loss) function, with a different cost for querying each information source. This setting is known as Multi-Information Source Optimization (MISO), or multi-fidelity optimization in the special case that the "fidelity" of each source is known a priori and independent on the value of the hyperparameters.

1.2 Multi information sources optimization: related works

This problem was initially studied under the name of multi-fidelity optimization in which rather than a single objective f, we have a collection of information sources denoted with $f_1(x), \ldots, f_S(x)$. Each source has its own cost, $c_1, \ldots, c_S$, where $c_\int > 0 \forall s = 1, .., S-1$, which controls the fidelity with lower s giving higher fidelity: increasing the fidelity gives a more accurate estimate but at a higher cost. In the case of cross-validation, the fidelity can be related to the number of iterations of the learning algorithm, the amount of data used in the training or the number of folds in the cross-validation. In MISO, the goal is to solve (1) while reducing the overall cost along the optimization process. MISO requires specific approaches to choose both the next location and source to evaluate, leading to a sequence $\left\{\left(s^{(1)}, x^{(1)}\right), \ldots, \left(s^{(N)}, x^{(N)}\right)\right\}$. It is always possible to sort sources such that $c_s > c_{s+1}$; in the case that also $f(x)$ can be queried, then it is the most expensive source, so we can set $f(x) = f_1(x)$ without loss of generality.

In the early work about multi-fidelity $f_s(x)$ were assumed to be ordered in terms of accuracy and cost: in more general problems of multi-information source optimization, we only assume the function $f(x)$ taking a design input x, the objective and $f_s(x)$ being the sources with different biases, different amounts of noise and different costs.

MISO has been gaining increasing attention in the last years, also beyond ML. An example in engineering design is the finite element method, where models with cost and fidelity can be obtained using different mesh values. Cheap approximations do not represent accurately the optimization targets, but still can offer an indication of the sensitivity of the output to changes in the parameters. Also, output data from physical prototypes can be integrated in the optimization framework as an additional information source, with fidelity depending on the application and the experimental setting. The application domain which has first exploited the advantages offered by multi-fidelity and multi-information source optimization is aerodynamics: in Chaudhuri et al. (2019) and Lam et al. (2015) is presented an approach that adaptively updates a multi-fidelity surrogate on multiple information sources and without any assumption about hierarchical relations among them.

In a seminal paper (Swersky et al. 2013), the use of small datasets to quickly optimize the hyperparameters of a ML model for large datasets has been proposed. The method shows that it is possible to transfer the knowledge gained from previous optimizations to new tasks in order to speed up k-fold cross-validation. The algorithm dynamically chooses which dataset to query in order to yield the most information per unit cost. In Kandasamy et al. (2016) a multi-fidelity bandit optimization based on Gaussian Process (GP) approximations of all the sources is proposed. The algorithm is named Multi-Fidelity Gaussian Process Upper Confidence Bound (MF-GP-UCB): it explores the search space using first the lower fidelity sources and then the higher ones in successively smaller regions, converging to the optimum. FABOLAS (FAst Bayesian Optimization on LArge dataSets) (Klein et al. 2017) is an approach for HPO on large datasets: at each iteration, it selects an hyperparameters configuration and a dataset size to use for optimizing hyperparameters for the entire dataset. Results are reported for HPO of Support Vector Machines (SVM) and DNNs, with FABOLAS often providing good solutions significantly faster than "vanilla" BO-based HPO on the full dataset. The approach in Poloczek et al. (2017) uses a GP with a kernel working on a space consisting of both the search space (spanned by the hyperparameters to optimize) and the information sources. In Ghoreishi and Allaire (2019) an approach incorporating correlations both within and among information sources is proposed. This allows to exploit the information collected over all the sources and then fusing them in a unique fused GP. Furthermore, the constrained setting is considered, where also constraints can be queried on multiple information sources.

A different approach has been proposed in Ariafar et al. (2020) Importance-based Bayesian Optimization (IBO), which models a distribution over the location of optimal hyperparameter configuration and allocates experimental budget according to cost adjusted expected reduction in entropy (Hennig and Schuler 2012). Higher fidelity observations provide a larger reduction in entropy, albeit at a higher evaluation cost.

To properly quantify predictive uncertainty, it is important for a learning system to recognize different types of uncertainty arising in the modeling process (Liu et al. 2019). Two types of uncertainty must be considered: aleatoric and epistemic. Aleatoric arises due to the stochastic variability of the data generating process, imperfect sensors, and epistemic arises due to our lack of knowledge about the data generating mechanism. A model epistemic uncertainty can be reduced by collecting more data and takes two forms: parametric uncertainty that is

uncertainty associated with estimating the model parameters under the current model specification and structural uncertainty that reflects the measure in which a model is sufficient to describe the data, i.e. whether there exists a systematic discrepancy.

1.3 Our contributions

The main contributions of this paper can be summarized as follows:

- A new GP called augmented GP which does not require a kernel working in the x, s space of hyperparameters and sources. Relations among sources are captured by a simplified and computationally cheap discrepancy measure (related to the epistemic error), used to select "reliable" evaluations to fit the proposed GP and included into a new acquisition function.
- A new acquisition function based on U/LCB but implementing a sparsification strategy. Indeed, the proposed GP results *sparse*, reducing the computational cost for fitting it (i.e., the number of evaluations raised power of three).
- The new GP mitigates the computational problems in estimating nonparametric regression which is inherently difficult in high dimensions with known lower bounds depending exponentially on dimension.
- Making MISO energy-efficient itself by selecting a subset of "reliable" evaluations among all those performed over all the sources. Only this subset is used to fit a GP differently from the fused GP in Ghoreishi and Allaire (2019).
- Demonstrating, empirically, the benefit provided by our approach on an HPO task aimed at optimally tuning a Support Vector Machine classifier on a large dataset.

2 Background

2.1 Gaussian processes

One way to interpret a Gaussian Process (GP) regression model is to think of f as a latent function defining a distribution over functions, and with inference taking place directly in the space of functions (i.e., *function-space view*) (Williams and Rasmussen 2006). A GP is a collection of random variables, any finite number of which have a joint Gaussian distribution. A GP is completely specified by its mean function $\mu(x)$ and covariance function $cov(f(x),f(x')) = k(x,x')$:

$$\begin{aligned} \mu(x) &= \mathbb{E}[f(x)] \\ cov(f(x),f(x')) &= k(x,x') = \mathbb{E}[(f(x)-\mu(x))(f(x')-\mu(x'))] \end{aligned} \tag{2}$$

, and the GP will be written as:

$$f(x) \sim GP\left(\mu(x), k\left(x,x'\right)\right) \tag{3}$$

Usually, for notational simplicity, we will take the prior of the mean function to be zero, although this is not necessary.

As consequence, the function values $f(x_1),\ldots,f(x_n)$ obtained at n different points $x_1,\ldots,x_n$, are jointly Gaussian. To see this, we can draw samples from the distribution of functions evaluated at any number of points; in detail, we choose a set of input points $X_{1:n} = (x_1,\ldots,x_n)^T$ and then compute the corresponding covariance matrix elementwise. This operation is usually performed by using predefined covariance functions allowing to write covariance between *outputs* as a function of *inputs* (i.e., $cov\left(f(x),f\left(x'\right)\right) = k(x,x')$). Finally, we can generate a random Gaussian vector as:

$$f(X_{1:n}) \sim \mathcal{N}(0, \mathrm{K}(X_{1:n}, X_{1:n})) \tag{4}$$

and plot the generated values as a function of the inputs. This is basically known as *sampling from prior*.

Let $\mathbf{X}_{1:n} = \left\{x^{(1)},\ldots,x^{(n)}\right\}$ denote a set of n locations into the search space $\mathcal{X}$ and $\mathbf{y} = \left\{y^{(1)},\ldots,y^{(n)}\right\}$ the associated function values, with $y^{(i)} = f\left(x^{(i)}\right)$ or, in the noisy setting, $y^{(i)} = f\left(x^{(i)}\right) + \varepsilon$ with $\varepsilon \sim \mathcal{N}\left(0,\lambda^2\right)$. Then, the GP's mean and variance are conditioned to the training set $(\mathbf{X}_{1:n},\mathbf{y})$ as follows:

$$\mu(x)|(\mathbf{X}_{1:n},\mathbf{y}) = \mathbf{k}(x,\mathbf{X}_{1:n})\left[\mathbf{K}+\lambda^2\mathbf{I}\right]^{-1}\mathbf{y} \tag{5}$$

$$\sigma^2(x)|(\mathbf{X}_{1:n},\mathbf{y}) = k(x,x) - \mathbf{k}(x,\mathbf{X}_{1:n})\left[\mathbf{K}+\lambda^2\mathbf{I}\right]^{-1}\mathbf{k}(\mathbf{X}_{1:n},x) \tag{6}$$

with k a kernel function, $\mathbf{k}(x,\mathbf{X}_{1:n})$ a vector whose ith component is $k\left(x,x^{(i)}\right)$ and $\mathbf{K}$ an $n \times n$ matrix with entries $\mathbf{K_{ij}} = k\left(x^{(i)},x^{(j)}\right)$. Finally, $\mathbf{k}(\mathbf{X}_{1:n},x)$ is the transposed version of $\mathbf{k}(x,\mathbf{X}_{1:n})$.

In the following, a simple example of five different samples is drawn at random from a GP prior and posterior, respectively (Fig. 1). The posterior is conditioned on six function observations.

A kernel function (aka covariance function) is the crucial ingredient in a GP predictor, as it encodes assumptions about the function to approximate. It is clear that the notion of similarity between data points is crucial; it is a basic assumption that points which are close are likely to have similar target values y, and thus function evaluations that

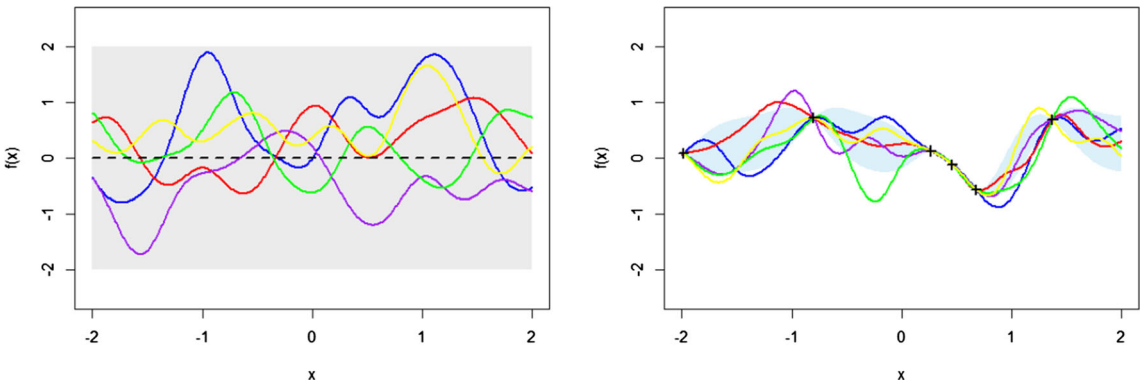

Fig. 1 Sampling from prior vs sampling from posterior (for the sake of simplicity, we consider the noise-free setting)

are near to a given point should be informative about the prediction at that point. Under the GP view it is the covariance function that defines nearness or similarity.

2.1.1 Squared Exponential (SE) kernel:

$$k_{SE}\left(x, x^{'}\right) = e^{-\frac{\|x-x'\|^2}{2\updownarrow^2}}$$

With $\updownarrow$ known as *characteristic length scale*. A large value of the length scale will map x to a narrower range of values, while a small length-scale does the opposite. Consequently, a large length-scale implies long-range correlations, whereas a short length-scale makes function values strongly correlated only if their respective inputs are very close to each other. This kernel is infinitely differentiable, meaning that the sample paths of the corresponding GP are very "smooth."

Another way to look at $\updownarrow$ is through the expected number of 0-upcrossings which is proportional to $1/\updownarrow$. Then $\updownarrow$ is proportional to the expected length before crossing 0, hence the name length scale.

SE is the most widely used kernel because it is easy to code, relatively robust to misspecification and guarantees a positive definite covariance regardless of input dimensions. One must anyway bear in mind that it is particularly liable to numerical ill conditioning of the kernel matrix. Other widely adopted kernels are reported in Appendix of this paper.

2.2 Acquisition functions

The acquisition function is the mechanism to implement the trade-off between *exploration* and *exploitation* in BO. More precisely, any acquisition function aims to guide the search of the optimum toward points with potential low values of objective function either because the prediction of $f(x)$, based on the probabilistic surrogate model, is low or the uncertainty is high (or both). Indeed, *exploiting* means to target the area providing more chance to improve the current solution (with respect to the current surrogate model), while *exploring* means to move toward less explored regions of the search space where predictions based on the surrogate model have a higher variance.

Confidence Bound—where Upper and Lower Confidence Bound (UCB and LCB) are used, respectively for maximization and minimization problems—is an acquisition function that manages exploration–exploitation by being optimistic in the face of uncertainty, in the sense of considering the best-case scenario for a given probability value (Auer 2002).

For the case of minimization, LCB is given by:

$$\mathrm{LCB}(x) = \mu(x) - \xi\sigma(x)$$

where $\xi \geq 0$ is the parameter in charge of managing the trade-off between exploration and exploitation ($\xi = 0$ is for pure exploitation; on the contrary, higher values of ξ emphasize exploration by inflating the model uncertainty). For this acquisition function there are strong theoretical results, originated in the context of multi-armed bandit problems, on achieving the optimal regret derived by Srinivas et al. (2012). For the candidate point x_n we observe instantaneous regret $r_n = f(x_n) - f(x^*)$. The cumulative regret R_N after N function evaluations is the sum of instantaneous regrets: $R_N = \sum_{n=1}^{N} r_n$. A desirable asymptotic property of an algorithm is to be no-regret: $\lim_{N\to\infty} \frac{R_N}{N} = 0$. Bounds on the average regret $\frac{R_N}{N}$ translate bounding R_N by a quantity sublinear in T, to convergence rates: $f(x^+) = \min x_{n \leq N} f(x_n)$, in the first N function evaluations, is no further from $f(x^*)$ than the average regret. Therefore, $f(x^+) - f(x^*) \to 0$, with $N \to \infty$ and so a no regret algorithm will converge to a subset of the global minimizers.

A wide analysis of the convergence rate of R_N in the case of Matérn kernel, for different values of d and ν, is given in Vakili et al. (2020).

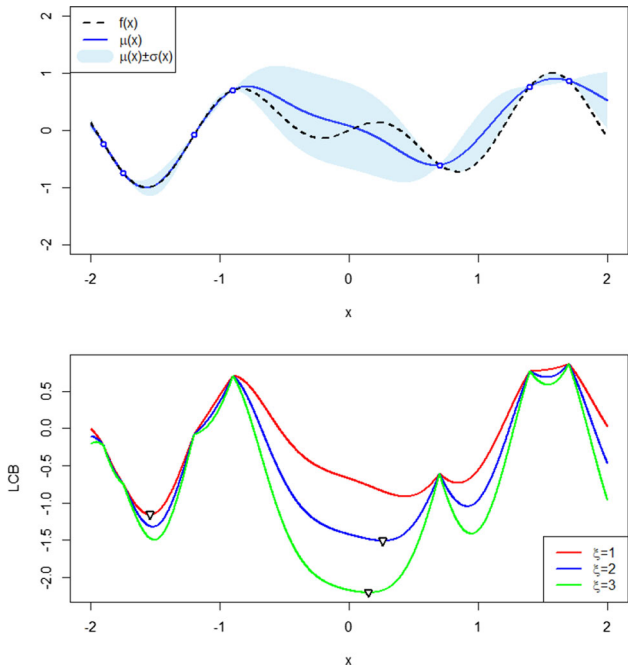

Fig. 2 GP trained depending on seven observations (top), LCB with respect to different values of ξ and min values corresponding to the next point to evaluate (bottom)

Figure 2 shows how the selected points change depending on ξ.

Finally, the next point to evaluate is chosen according to $x^{(n+1)} = \underset{x \in X}{\operatorname{argmin}} LCB(x)$, in the case of a minimization problem, or $x^{(n+1)} = \underset{x \in X}{\operatorname{argmax}} UCB(x)$ in the case of a maximization problem.

From the perspective of BO a particularly interesting bandit problem is the kernelized continuum armed bandit-problem (Srinivas et al. 2010). Here, f is assumed to be in the closure of functions on X expressible as a linear combination of a feature embedding parametrized by a kernel k. The properties of the functions in the resulting space referred as the RKHS of k, are determined by the choice of the kernel. For a SE kernel the RKHS contains only infinitely differentiable functions.

The optimization of the acquisition function leads to the next location to be queried, $x^{(n+1)}$, and, consequently, to a sequence of locations generated $\{x^{(1)}, \ldots, x^{(N)}\}$ over the BO process, with N the overall number of function evaluations at the end of the process. In this paper we use Lower Confidence Bound, largely adopted in GP-based BO and with a convergence proof under an appropriate scheduling of the internal parameter $\beta^{(n)}$ (Srinivas et al. 2012) which balances between exploration and exploitation:

$$LCB^{(n)}(x) = \mu^{(n)}(x) - \sqrt{\beta^{(n)}}\sigma^{(n)} \tag{7}$$

where the apex related to the current iteration n has been included to highlight that the value of β changes over BO iterations, as well as the conditioned GP's mean and standard deviation. Confidence Bound has been successfully applied in MISO, such as in Kandasamy et al. (2016). Wilson et al. (2018) point out that the shape of the acquisition function may have large flat regions which, in particular in high-dimensional spaces, make its optimization problematic and propose a Monte Carlo evaluation of acquisition function amenable to gradient-based optimization and identify a family of acquisition functions, including UCB, whose characteristics allow using greedy approaches for their maximization.

A specific problem in MISO is related to the acquisition function. According to Poloczek et al. (2017) and Ghoreishi and Allaire (2019), Knowledge Gradient, Entropy Search and Predictive Entropy Search can be applied. However, their computation and optimization are computationally expensive: for this reason, in this paper we consider L/UCB and build on it a new acquisition function specifically designed for MISO.

3 The proposed multi information source optimization—augmented Gaussian process (MISO-AGP)

3.1 Augmented GP

The MISO approach proposed in this paper is based on the idea of training a GP on a "reliable" subset of all the function evaluations performed so far over all the information sources. We refer to this GP as *Augmented Gaussian Process* (AGP) and consequently named our approach MISO-AGP. The term "augmented" is used to highlight that the set of function evaluations to train the AGP starts from those performed on the most expensive source and then it is "augmented" by selecting evaluations performed on some other source. Before explaining how the selection process is performed, we introduce some useful notations.

Let $D_s = \left\{\left(x^{(i)}, y_s^{(i)}\right)\right\}_{i=1,\ldots,n_s}$ denote the n_s function evaluations performed so far on the source $\int$. For each source s a specific GP, $\mathcal{G}_s$, is trained on the current D_s. Let us introduce a *model discrepancy* measure, $\eta(x, \mathcal{G}, \mathcal{G}')$, between two GPs. Differently from other papers, such as Poloczek et al. (2017) and Ghoreishi et al. (2019), we compute it simply as:

$$\eta(x, \mathcal{G}, \mathcal{G}') = |\mu(x)|D_s - \mu'(x)|D_{s'}| \tag{8}$$

with $\mu(x)$ and $\mu'(x)$ the conditioned mean functions of the two GPs. It is also important to note that $\eta(x, \mathcal{G}, \mathcal{G}')$ depends on x. Indeed, in MISO we do not know a-priori the fidelity of each source and it could be not constant over $\mathcal{X}$.

Assume that $f(x)$ can be queried at the highest cost, that is $f(x) = f_1(x)$. Thus, the set of evaluations to train the AGP consists of D_1 "augmented" by:

$$\tilde{D} = \left\{(\tilde{x}, \tilde{y}) : \exists \ddagger : (\tilde{x}, \tilde{y}) \in D_{\ddagger} \wedge \eta(x, \mathcal{G}_1, \mathcal{G}) < m\sigma_1(x)\right\} \quad (9)$$

with m a technical parameter of the MISO-AGP algorithm. We used $m = 1$ (i.e., around 68% of observations normally distributed are in the interval mean $\pm$ standard deviation). Thus, function evaluations on cheaper sources, having a discrepancy lower than the threshold given in (9), are considered "reliable" to be merged with those collected on the most expensive source. Let $\widehat{D}$ denote the augmented set of function evaluations, such that $\widehat{D} = D_1 \cup \widetilde{D}$, the AGP $\widehat{\mathcal{G}}$ is trained on $\widehat{D}$, leading to $\widehat{\mu}(x)$ and $\widehat{\sigma}(x)$, computed according to (5–6). An example is reported in Fig. 3.

3.2 Acquisition function in MISO-AGP algorithm

Following the training of the AGP, an acquisition function must be used to choose the next pair *source-location* to query, that is (s', x'). We consider the framework of U/LCB:

$$(s', x') = \underset{\substack{x \in X \subset \mathbb{R}^d \\ s = 1, \ldots, S}}{\operatorname{argmax}} \left\{ \frac{y^+ - \left(\widehat{\mu}(x) - \sqrt{\beta^{(n)}}\widehat{\sigma}(x)\right)}{c_s\left(1 + \eta\left(x, \widehat{\mathcal{G}}, \mathcal{G}_s\right)\right)} \right\} \quad (10)$$

where n is the number of function evaluations into $\widehat{D}$ and $y^+ = \min_{(x,y)\in\widehat{D}} \{y\}$ is the best observed value into $\widehat{D}$. The numerator is the most optimistic improvement with respect to the AGP's LCB, penalized by the cost of the source $\int$ and the model discrepancy between the AGP $\widehat{\mathcal{G}}$ and $\mathcal{G}_s$, at the location x.

There is the chance that x' could be too close to some previous function evaluations on s'. This behaviour arises when BO is converging to a (local/global) optimum and leads to a well-known instability issue in GP training, which is ill-conditioning in the inversion of the matrix $[\mathbf{K} + \lambda^2\mathbf{I}]$. This instability issue occurs even more frequently and quickly in the noise-free setting (i.e., $\lambda = 0$). To avoid this undesired behavior—leading to wasting evaluations without obtaining any improvement and/or risking occurring in the instability issue—we introduce the following correction.

Given (s', x') from (10), if

$$\exists\left(x^{(i)}, y^{(i)}\right) \in D_{s'} \wedge x' - x^{(i)2} < \delta$$

$$s' \leftarrow 1 \text{ and } x' = \underset{x\in\mathcal{X}\subset\mathbb{R}^d}{\operatorname{argmax}}\, \sigma_1(x) \quad (11)$$

with $\delta > 0$ the second MISO-AGP's technical parameter. In other words, we set the acceptable level of approximation, δ, in locating the optimizer and, in the case that x' is closer than δ to another evaluation on s', then we prefer to "spend our budget" in reducing uncertainty on the most expensive source.

The MISO-AGP algorithm is summarized in the following.

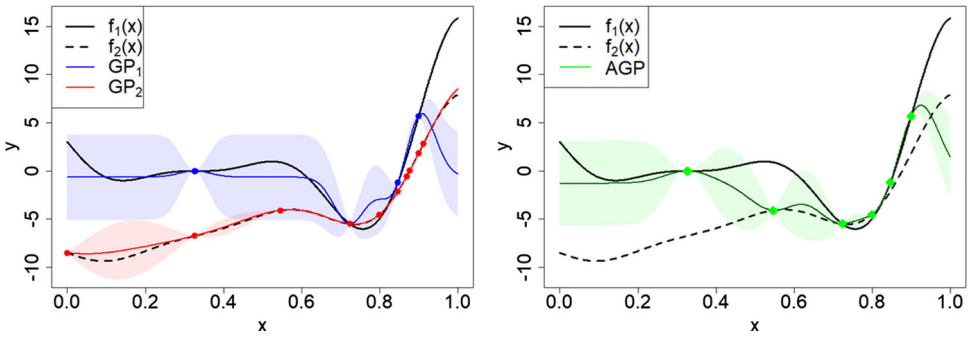

Fig. 3 An example of AGP on a one-dimensional MISO minimization problem with two information sources. (Left) the two GPs trained on each source; (right) the AGP: only three evaluations on the cheaper source (around $x = 0.5$, $x = 0.7$ and $x = 0.8$) are selected to "augment" the evaluations on the expensive one. This reduces, at the same time, the uncertainty near the global minimum of f_1 and the number of evaluations for training the AGP (six out of the 14 overall)

Algorithm: MISO-AGP

Input:

$f_1(x), \ldots, f_S(x)$; $c_1, \ldots, c_S$; $\mathcal{X}$; max cumulated cost $\bar{C}$; max iterations N;
set m and δ (MISO-AGP's technical parameters)

Initialization:

$D_s = \left\{\left(x^{(i)}, y_s^{(i)}\right)\right\}_{i=1,\ldots,n_s}$ $\forall\, s = 1, \ldots, S$ and with n_s initial evaluations on locations randomly sampled in $\mathcal{X}$

Main:

$c \leftarrow 0$; $n \leftarrow 0$;
while ($c < \bar{C}$ AND $n < N$) **do**
  train $\mathcal{G}_s$ on D_s $\forall\, s = 1, \ldots, S \Rightarrow \mu_s(x), \sigma_s(x)$
  build $\widehat{D} = D_1 \cup \widetilde{D}$ with $\widetilde{D}$ defined in (9)
  train the AGP $\hat{\mathcal{G}}$ on $\widehat{D} \Rightarrow \hat{\mu}(x), \hat{\sigma}(x)$
  choose (s', x') according to (10)
  if $\exists\, x^{(i)}: \left(x^{(i)}, y^{(i)}\right) \in D_{s'} \wedge \left\|x' - x^{(i)}\right\|^2 < \delta$ **then**
    (s', x') according to (11)
  endif
  query source s' at location x' and observe $y_{s'}$
  update $D_{s'} \leftarrow D_{s'} \cup \{(x', y_{s'})\}$
  $c \leftarrow c + c_{s'}$; $n \leftarrow n + 1$
endwhile

Output:

build $\widehat{D} = D_1 \cup \widetilde{D}$ with $\widetilde{D}$ defined in (9)
return $(x^+, y^+) \in \widehat{D}$: $y^+ = \min_{(x,y)\in\widehat{D}} \{y\}$

3.3 Computational setting

All the experiments have been performed on a Microsoft Azure virtual machine, H8 (High Performance Computing family) Standard with 8 vCPUs, 56 GB of memory, Ubuntu 16.04.6 LTS. The code has been developed in R: all the code is available upon request from the authors.

4 Experimental setting

4.1 Test problems

To validate the proposed MISO-AGP approach, we have first evaluated it on two test problems: the one-dmensional Forrester test function (Forrester et al. 2007; Bartz-Beielstein et al. 2015) and the two-dimensional Rosenbrock test function, presented in Poloczek et al. (2017) as a MISO as well as multi-fidelity optimization test case.

4.1.1 Forrester test problem

The Forrester test problem is characterized by the two following sources:

$$f_1(x) = f(x) = (6x-2)^2 \sin(12x-4)$$
$$f_2(x) = 0.5 f_1(x) + 10(x-0.5) + 5$$

with associated costs $c_1 = 1000$ and $c_2 = 1$. The two functions are considered black-box, and the search space is the interval $[0, 1]$. The solution for this problem is $x^* = 0.7572488$ with associated function value $f(x^*) = -6.02074$.

4.1.2 Rosenbrock test problem

The Rosenbrock test problem is characterized by the following two sources:

$$f_1(x) = \left(1 - x_{[1]}\right)^2 + 100\left(x_{[2]} - x_{[1]}^2\right)^2$$
$$f_2(x) = f_1(x) + 0.1 \sin\left(10x_{[1]} + 5x_{[2]}\right)$$

where $x_{[1]}$ and $x_{[2]}$ represent, respectively, the first and second components of x. The associated costs for evaluating the two sources are $c_1 = 1000$ and $c_2 = 1$. The two functions are considered black-box and the search space is $[-2, 2]^2$. The solution of this problem is $x^* = (1, 1)$ with associated function value $f(x^*) = 0$.

4.2 C-support vector classification with radial basis function kernel

To validate our MISO-AGP approach, we designed an HPO task whose goal is to optimally and efficiently tune the hyperparameters of a Support Vector Machine (SVM) classifier on a large dataset. More precisely, we consider a C-SVC with a Radial Basis Function (RBF) kernel and the "MAGIC Gamma Telescope" dataset.[1]

We chose C-SVC (i.e., C-Support Vector Classification, where C is the hyperparameter managing the trade-off between maximizing the margin and minimizing the classification error) due to its relative inefficiency on large datasets: computational complexity for training a C-SVC, on a given hyperparameters configuration, is the number of instances raised the power of three. The C-SVC's hyperparameters to optimize are the regularization term, C, and γ in the RBF kernel: $k_{RBF}(x, x') = e^{-\gamma\|x-x'\|^2}$.

The MAGIC dataset is generated by a Monte Carlo program (Heck et al. 1998), to simulate registration of high energy gamma particles in a ground-based atmospheric Cherenkov gamma telescope using the imaging technique. The overall dataset consists of 19,020 instances: 12,332 of the class "gamma (signal)" and 6,688 of the class "hadron (background)," with each instance represented by ten continuous features. We have performed a preprocessing consisting in scaling all the dataset features in $[0, 1]$.

Following the notation used in this paper, MISO-AGP will be used to minimize $f(x)$. This is straightforward for the two test problems, Forrester and Rosenbrock. As far as the hyperparameter optimization of the C-SVC is considered, $f(x)$ is the misclassification error computed on tenfold cross validation on the MAGIC dataset. The search space $\mathcal{X}$ is two-dimensional and box-bounded, spanned by the two C-SVC's hyperparameters $C \in \left[10^{-2}, 10^{2}\right]$ and $\gamma \in \left[10^{-4}, 10^{4}\right]$. We adopt a logarithmic scaling of the search space, a usual procedure suggested in AutoML for hyperparameters varying within ranges of this scale.

We have defined two different sources: the first provides the misclassification error obtained via tenfold cross-validation of a C-SVC configuration using the entire MAGIC dataset (i.e., $f_1(x) = f(x)$). The second (i.e., $f_2(x)$) performs the same computation but using a smaller portion of the data (just 5% through stratified sampling).

Energy required to perform tenfold cross validation is basically associated with the computational time, which we consider as a proxy for the sources' costs. Since computational time can also depend on the values of C-SVC's hyperparameters, we have run a sample of ten hyperparameters configurations on both the two sources and used the average computational times for estimating reference values for c_1 and c_2. More precisely, computational time required by $f_1(x)$ is, on average, 320 times that required by $f_2(x)$. Thus, we set $c_2 = 1$ and, consequently, $c_1 = 320$.

[1] http://archive.ics.uci.edu/ml/datasets/magic+gamma+telescope.

4.3 MISO-AGP setting

The kernel used to model the covariance function, for all the GPs, including the AGP, is the Squared Exponential kernel, whose hyperparameters are set via Maximum Loglikelihood Estimation during the GP training. The acquisition function (12) and, in case, the correction (11) are both optimized via L-BFGS (Limited-memory Broyden–Fletcher–Goldfarb–Shanno algorithm) algorithm.

As initialization, three hyperparameters configurations are sampled in $\mathcal{X}$ via Latin Hypercube Sampling (Huntington and Lyrintzis 1998). Then, 30 further function evaluations are used by MISO-AGP to optimize over sources. We decided not to set a limit on the cumulated cost but to use this value to make considerations on the efficiency of the proposed approach with respect to BO applied only on the most expensive source. To mitigate the effect of initial randomness, ten different runs of MISO-AGP and BO have been performed and compared: at each run, the two approaches share the same initialization.

As metrics, we consider the best function value observed so far. It is usually named "best seen" in BO and simply defined as $y_+^{(n)} = \min_{i=1,\dots,n} \left\{y^{(1)}, \dots, y^{(n)}\right\}$—because we are considering the minimization of the misclassification error. However, this definition is no more valid in the case of the AGP. Suppose that, at a certain iteration, a function evaluation on a cheaper source is selected to fit the AGP and that corresponds to the best seen up to that iteration. At the next iteration, it could not be selected and, consequently, it cannot be considered as the best seen any longer. More formally, let $\widehat{y}_+^{(n)}$ denote the "augmented best seen," $\widehat{y}_+^{(n)} = \min_{i=1,\dots,p} \left\{y^{(1)}, \dots, y^{(p)}\right\}$, with $p < n$ because only a subset of the evaluations on all the sources is used to train the AGP. In the case that $\widehat{y}_+^{(n-1)} \notin \left\{y^{(1)}, \dots, y^{(p)}\right\} \Rightarrow \widehat{y}_+^{(n)} \lesseqgtr \widehat{y}_+^{(n-1)}$; in other terms, contrary to the common "best seen," the "augmented best seen" could not be monotone over the function evaluations.

5 Results

5.1 Results on test problems

In this section we summarize the results related to the two test problems, namely Forrester and Rosenbrock. The

actual optimizer x^* is known for each one of the two test problems, so we measured the distance between the optimal solution identified by at the end of the BO and MISO-AGP processes. Distances are reported in Table 1, as average and standard deviation on the 30 different runs.

As expected, the overall cumulated cost for MISO-AGP is significantly lower than performing BO on $f_1(x) = f(x)$ only.

With respect to the Forrester test problem, the solutions identified by MISO-AGP are significantly closer to x^* than those found by BO (Wilcoxon test, p value < 0.01), with approximately half of the cumulated cost. Results are less exciting on the Rosenbrock test problem: BO solutions are in this case closer to x^* than those found by MISO-AGP (Wilcoxon test, p-value < 0.001). However, MISO-AGP cost is significantly lower than BO, around the 2%, on average. Therefore, MISO-AGP has still margin to improve by slighting increasing its cumulated cost.

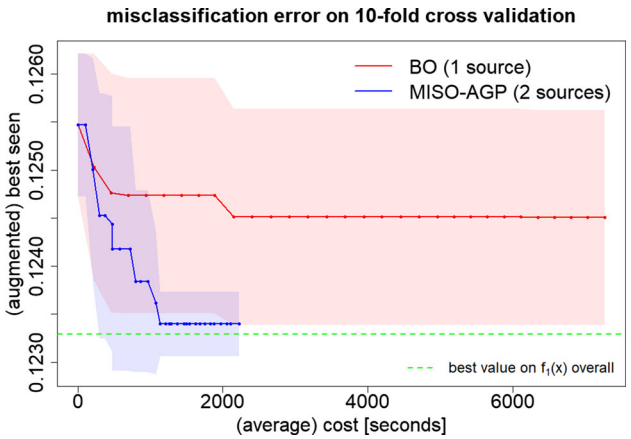

Fig. 4 HPO of C-SVC on the MAGIC dataset. Comparison between traditional BO-based HPO and MISO-AGP on two information sources. Results refer to ten independent runs

5.2 Results on hyperparameter optimization of C-SVC

Figure 4 summarizes the results obtained on a real-world application related to hyperparameter optimization of a C-SVC on the MAGIC dataset. In this case, the optimal hyperparameters configuration (i.e., x^*) is not known a priori, as well as the associated function value $f(x^*)$. The best value of the misclassification error is reported with respect to the cost cumulated over the MISO-AGP and BO iterations, separately. Solid lines represent the mean over the ten independent runs, while shaded areas represent the standard deviations. As a reference value, we have considered the best misclassification error registered, on the entire MAGIC dataset, over all the experiments performed (green dashed line). The cumulated costs—which are actual and not the nominal c_1 and c_2 used in the acquisition function—are also averaged on the ten independent runs.

The MISO-AGP approach proved to be both more effective and efficient than traditional BO: the identified hyperparameters configurations are associated with a lower misclassification error, and within less than one-third of the time required by BO. On average, 60% of the function evaluations are performed on the cheaper source. Thus, MISO-AGP has intelligently exploited the cheaper information source, thanks to the proposed AGP, leading to an energy-efficient and green HPO task.

6 Conclusions

The GP framework can be extended to deal with multiple information sources. Relations among sources are captured by a simplified and computationally cheap discrepancy measure, which enables a sparsification strategy used to select "reliable" evaluations to fit the proposed AGP. The MISO-AGP has been empirically shown to solve a real HPO task effectively while reducing significantly computational time and consequently energy usage.

Appendix

A: Other kernels

The following are some well-known and widely adopted kernels in GP modeling.

Table 1 Distance from x^* and overall cumulated cost. Values are mean (standard deviation) on 30 independent runs. Standard deviation is not applicable in the case of BO cost because BO uses only one source, that is $f_1(x) = f(x)$

	Forrester		Rosenbrock	
	Distance	Cost [×1000]	Distance	Cost [×1000]
BO	0.093 (0.167)	30 (n.a.)	0.379 (0.067)	30 (n.a.)
MISO-AGP	0.031 (0.008)	16.833 (9.480)	0.978 (0.790)	0.633 (0.765)

Matérn kernel:

$$k_{Mat}\left(x,x^{'}\right)=\frac{2^{1-v}}{\Gamma(v)}\left(\frac{|x-x'|\sqrt{2v}}{\updownarrow}\right)^{v}K_{v}\left(\frac{|x-x'|\sqrt{2v}}{\updownarrow}\right)$$

where v and $\updownarrow$ are two kernel's hyperparameters and K_v is a modified Bessel function. Note that for $v\rightarrow\infty$ we obtain the SE kernel.

The Matérn covariance functions become especially simple when v is half-integer: $v=p+1/2$, where p is a non negative integer. The formula can be rewritten as the product of an exponential and polynomial terms of order $p-1$.

The advantages of the simplified covariance Matérn function are that there are no Bessel functions, no sum of factorials nor fraction of gammas as reported in Gramacy (2020). This is important because *the evaluation of the Bessel function can be as computationally demanding as the matrix inversion.*

The most widely adopted versions, specifically in the Machine Learning community, are $v=3/2$ and $v=5/2$:.

$$k_{v=3/2}(x,x^{'})=\left(1+\frac{|x-x'|\sqrt{3}}{\updownarrow}\right)e^{-\frac{|x-x'|\sqrt{3}}{\updownarrow}}$$

$$k_{v=5/2}(x,x^{'})=\left(1+\frac{|x-x'|\sqrt{5}}{\updownarrow}+\frac{(x-x')^{2}}{3\updownarrow^{2}}\right)e^{-\frac{|x-x'|\sqrt{5}}{\updownarrow}}$$

Choosing $p=0$ one obtains the exponential family, $p=0$ implies $v=1/2$ which is appropriate for rough surfaces.

A sample path of latent f under a GP with Matérn will be k-times differentiable iff v is larger than k.

One great advantage of Matérn is that at least for small v it creates covariance matrices that are better conditioned than SE.

The exponential kernel is also called the Laplace kernel and has a strong link with Mondrian kernels which results in Gaussian models conceptually close to Random Forests (Lévesque et al. 2017).

Rational quadratic covariance function

$$k_{RQ}\left(x,x^{'}\right)=\left(1+\frac{(x-x')^{2}}{2\alpha\updownarrow^{2}}\right)^{-\alpha}$$

where α and $\updownarrow$ are two hyperparameters. This kernel can be considered as an infinite sum (*scale mixture*) of SE kernels, with different characteristic length scales.

The aforementioned kernels are just the most widely adopted in GP regression.

More details and a most comprehensive set of covariance functions are reported in Williams and Rasmussen (2006), and Gramacy (2020) including nonstationary kernels and dot product kernels.

Some issues on kernel have been considered in recent publications, such as: kernel composition, safe optimization in relation to cognition (Schulz et al. 2018) as well as kernel learning, adaptation and sparsity in order to deal with functions that are smooth in a subset of their domain and can vary rapidly in another as analyzed in Peifer et al. (2019) from the viewpoint of computational complexity in the framework of RKHS (Reproducing Kernel Hilbert Spaces).

A space-temporal kernel has been proposed in Nyikosa et al. (2018) to allow the GP to capture all the instances of the function over time and track a temporally evolving minimum.

B: Other improvement-based acquisition functions

Confidence Bound, adopted in this paper is an optimistic acquisition function. It belongs to the family of *improvement-based* acquisition function, aimed at searching for the optimum, y^*, instead of the optimizer, x^* (the second family of acquisition functions are known as *entropy-based*). The Following are other two well-known and widely adopted improvement-based acquisition functions.

Probability of Improvement (PI) was the first acquisition function proposed in the literature (Kushner 1964):

$$PI(x)=P(f(x)\leq f(x^{+})+\xi)=\boldsymbol{\Phi}\left(\frac{f(x^{+})-\mu(x)-\xi}{\sigma(x)}\right)$$

where $f(x^+)$ is the best value of the objective function observed so far, $\mu(x)$ and $\sigma(x)$ are mean and standard deviation provided by (6) and square root of (7), and $\boldsymbol{\Phi}(\bullet)$ is the normal cumulative distribution function. The parameter ξ is introduced to modulate the balance between exploration and exploitation. More precisely, $\xi=0$ is toward exploitation while $\xi>0$ is more toward exploration.

The next point to evaluate is chosen according to:$x_{n+1}=\underset{x\in X}{\operatorname{argmax}}PI(x)$

Expected Improvement (EI), initially proposed in Močkus (1975) and then made popular in Jones et al. (1998), measures the expectation of the improvement on $f(x)$ with respect to the predictive distribution of the probabilistic surrogate model.

$$EI(x)=\begin{cases}(f(x^{+})-\mu(x)-\xi)\Phi(Z)+\sigma(x)\phi(Z) & if\ \sigma(x)>0 if\sigma(x)>0\\ 0 & if\ \sigma(x)=0\end{cases}$$

$$Z=\begin{cases}\dfrac{f(x^{+})-\mu(x)-\xi}{\sigma(x)}\dfrac{f(x^{+})-\mu(x)-\xi}{\sigma(x)} & if\sigma(x)>0\\ 0 & if\sigma(x)=0\end{cases}$$

The parameter ξ in order to actively manage the trade-off between exploration (larger values) and exploitation (smaller) should be adjusted dynamically to decrease monotonically with the function evaluations.

The next point to evaluate is chosen according to: $x_{n+1} = \underset{x \in X}{\text{argmax}} EI(x)$

Acknowledgements We greatly acknowledge the DEMS Data Science Lab, Department of Economics Management and Statistics (DEMS), for supporting this work by providing computational resources.

Funding Open access funding provided by Università degli Studi di Milano - Bicocca within the CRUI-CARE Agreement.

Compliance with ethical standards

Conflict of interest The authors have no conflicts of interest to declare.

Ethical approval This article does not contain any studies with human participants or animals performed by any of the authors.

References

Aggarwal CC (2018) Neural networks and deep learning. Springer, vol 10, pp 978–983

Ariafar S, Mariet Z, Elhamifar E, Brooks D, Dy J, Snoek J (2020) Weighting is worth the wait: Bayesian optimization with importance sampling. arXiv preprint http://arxiv.org/abs/2002.09927

Auer P (2002) Using confidence bounds for exploitation-exploration trade-offs. J Mach Learn Res 3:397–422

Bartz-Beielstein T, Jung C, Zaefferer M (2015) Uncertainty management using sequential parameter optimization. In: Uncertainty management in simulation-optimization of complex systems. Springer, pp 79–99

Bianco S, Buzzelli M, Ciocca G, Schettini R (2020) Neural architecture search for image saliency fusion. Inf Fusion 57:89–101

Candelieri A, Archetti F (2019) Bayesian optimization and data science. Springer International Publishing

Chaudhuri A, Marques AN, Lam R, Willcox KE (2019) Reusing information for multifidelity active learning in reliability-based design optimization. In: AIAA Scitech 2019 Forum 1222

De Ath G, Fieldsend JE, Everson RM (2020) What do you mean? The role of the mean function in Bayesian optimisation. arXiv preprint http://arxiv.org/abs/2004.08349

Forrester AI, Sobester A, Keane AJ (2007) Multi-fidelity optimization via surrogate modelling. Proc R Soc Math Phys Eng Sci 463(2088):3251–3269

Frazier PI (2018) Bayesian optimization. In: INFORMS tutorials in operations research, pp 255–278

Ghoreishi SF, Allaire D (2019) Multi-information source constrained Bayesian optimization. Struct Multidiscip Optim 59(3):977–991

Gramacy RB (2020) Surrogates: Gaussian process modeling, design, and optimization for the applied sciences. CRC Press

Hao K (2019) Training a single AI model can emit as much carbon as five cars in their lifetimes. Deep learning has a terrible carbon footprint. MIT TECHNOLOGY REVIEW

Heck D, Schatz G, Knapp J, Thouw T, Capdevielle JN (1998) CORSIKA: a Monte Carlo code to simulate extensie air showers (No. FZKA-6019)

Hennig P, Schuler CJ (2012) Entropy search for information-efficient global optimization. J Mach Learn Res 13(Jun):1809–1837

Ho TK (1995) In: Proceedings of the 3rd international conference on document analysis and recognition. Random decision forests, pp 278–282

Huntington DE, Lyrintzis CS (1998) Improvements to and limitations of Latin hypercube sampling. Probab Eng Mech 13(4):245–253

Hutter F, Kotthoff L, Vanschoren J (2019) Automated machine learning. Springer, New York, NY, USA

Jones DR, Schonlau M, Welch WJ (1998) Efficient global optimization of expensive black-box functions. J Global Optim 13(4):455–492

Kandasamy K, Dasarathy G, Oliva JB, Schneider J, Póczos B (2016) Gaussian process bandit optimisation with multi-fidelity evaluations. In: Advances in neural information processing systems, pp 992–1000

Klein A, Falkner S, Bartels S, Hennig P, Hutter F (2017) Fast Bayesian optimization of machine learning hyperparameters on large datasets. In: Artificial intelligence and statistics, pp 528–536

Kotthoff L, Thornton C, Hoos HH, Hutter F, Leyton-Brown K (2017) Auto-WEKA 2.0: automatic model selection and hyperparameter optimization in WEKA. J Mach Learn Res 18(1):826–830

Kulkarni A, Shivananda A (2019) Deep learning for NLP. In: Natural language processing recipes, pp 185–227. Apress, Berkeley, CA

Kushner HJ (1964) A new method of locating the maximum point of an arbitrary multipeak curve in the presence of noise. J Basic Eng 86(1):97–106

Lam R, Allaire DL, Willcox KE (2015) Multifidelity optimization using statistical surrogate modeling for non-hierarchical information sources. In: 56th AIAA/ASCE/AHS/ASC structures, structural dynamics, and materials conference 0143

Lévesque JC, Durand A, Gagné C, Sabourin R (2017) Bayesian optimization for conditional hyperparameter spaces. In: 2017 International joint conference on neural networks (IJCNN). IEEE, pp 286–293

Lindauer M, Hutter F (2019) Best practices for scientific research on neural architecture search. arXiv preprint http://arxiv.org/abs/1909.02453

Liu J, Paisley J, Kioumourtzoglou MA, Coull B (2019) Accurate uncertainty estimation and decomposition in ensemble learning. In: Advances in neural information processing systems, pp 8950–8961

Melis G, Dyer C, Blunsom P (2017) On the state of the art of evaluation in neural language models. arXiv preprint http://arxiv.org/abs/1707.05589

Močkus J (1975) On Bayesian methods for seeking the extremum. In: Optimization techniques IFIP technical conference. Springer, Berlin

Nyikosa FM, Osborne MA, Roberts SJ (2018) Bayesian optimization for dynamic problems. arXiv preprint http://arxiv.org/abs/1803.03432

Peifer M, Chamon LF, Paternain S, Ribeiro A (2019) Sparse learning of parsimonious reproducing kernel Hilbert space models. In: ICASSP 2019–2019 IEEE international conference on acoustics, speech and signal processing (ICASSP). IEEE, pp 3292–3296

Poloczek M, Wang J, Frazier P (2017) Multi-information source optimization. In: Advances in neural information processing systems, pp 4288–4298

Schulz E, Speekenbrink M, Krause A (2018) A tutorial on Gaussian process regression: modelling, exploring, and exploiting functions. J Math Psychol 85:1–16

Schwartz R, Dodge J, Smith NA, Etzioni O (2019) Green AI. https://arxiv.org/abs/1907.10597

Shahriari B, Swersky K, Wang Z, Adams RP, De Freitas N (2016) Taking the human out of the loop: a review of Bayesian optimization. Proc IEEE 104(1):148–175

Srinivas N, Krause A, Kakade S, Seeger M (2010) Gaussian process optimization in the bandit setting: no regret and experimental design. In: Proceedings of the 27th international conference on international conference on machine learning. Omnipress, pp 1015–1022

Srinivas N, Krause A, Kakade SM, Seeger MW (2012) Information-theoretic regret bounds for gaussian process optimization in the bandit setting. IEEE Trans Inf Theory 58(5):3250–3265

Strubell E, Ganesh A, McCallum A (2019) Energy and policy considerations for deep learning in NLP. In: Proceedings of the 57th annual meeting of the association for computational linguistics, pp 3645–3650

Swersky K, Snoek J, Adams RP (2013) Multi-task Bayesian optimization. In: Advances in neural information processing systems, pp 2004–2012

Vakili S, Picheny V, Durrande N (2020) Regret bounds for noise-free Bayesian optimization. arXiv preprint http://arxiv.org/abs/2002.05096

Williams CK, Rasmussen CE (2006) Gaussian processes for machine learning, 2(3). MIT press, Cambridge, MA

Wilson J, Hutter F, Deisenroth M (2018) Maximizing acquisition functions for Bayesian optimization. In: Advances in neural information processing systems, pp 9884–9895

Wolpert DH (2002) The supervised learning no-free-lunch theorems. In: Soft computing and industry, pp 25–42. Springer, London

Yang X, Hua S, Shi Y, Wang H, Zhang J, Letaief KB (2020) Sparse optimization for green edge AI inference. J Commun Inf Netw 5(1):1–15