

Radial Basis Function and Bayesian Methods for the Hyperparameter Optimization of Classification Random Forests

Rommel G. Regis(✉)

Saint Joseph's University, Philadelphia, PA 19131, USA
rregis@sju.edu

Abstract. The hyperparameter optimization of a random forest (RF) is a discrete black-box optimization problem that aims to find the settings of the hyperparameters that optimize an overall out-of-bag (OOB) performance measure of the RF. This problem is computationally expensive for high-dimensional data involving many predictors and a large number of data points. This paper explores the use of surrogate-based approaches, particularly radial basis function (RBF) methods and Bayesian optimization techniques, to optimize the hyperparameters of a classification RF. The surrogates approximate the functional relationship between the hyperparameters and the overall OOB performance of the RF, and they are used to guide the search for a global optimum for the hyperparameter optimization problem. While Bayesian optimization methods have been used to tune the hyperparameters of a classification RF, RBF methods have rarely been used for this task, if at all. We compare the performance of the B-CONDOR-RBF algorithm and a Bayesian optimization method with global and local discrete random search methods to tune the discrete hyperparameters of an RF on 10 classification data sets that involve up to 753 predictor variables and up to 19K data points. The global variant of B-CONDOR-RBF obtained better overall OOB prediction error than the discrete random search methods and two Bayesian optimization algorithms given a limited budget of only 100 function evaluations. Furthermore, a local variant of B-CONDOR-RBF outperformed the random search methods and achieved comparable performance with the Bayesian optimization algorithms on the same problems.

Keywords: Random forest · classification problem · hyperparameter optimization · radial basis functions · Bayesian optimization

1 Introduction

Among the most popular predictive modeling techniques is the *random forest (RF)* [3], which is an ensemble of decision trees built from bootstrap samples of the training dataset. It is well-known that the predictive performance of a random forest partly depends on the setting of its hyperparameters, which are the

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2023
O. Gervasi et al. (Eds.): ICCSA 2023 Workshops, LNCS 14105, pp. 508–525, 2023.
https://doi.org/10.1007/978-3-031-37108-0_33

parameters that control the learning process. This paper considers the problem of tuning four hyperparameters of a random forest for classification tasks: number of decision trees in the RF (ntree), number of predictors drawn at random for each split (mtry), minimum number of samples in a node (min.node.size), and the fraction of the training set drawn at random with replacement for constructing each decision tree (sample.frac). We wish to find the settings of these hyperparameters that optimize an overall out-of-bag (OOB) measure of performance of the RF on classification problems. For example, we may wish to determine the RF hyperparameter values that minimize the overall OOB prediction error of the RF or maximize its overall OOB F1 score. A lower OOB prediction error or a higher OOB F1 score is generally expected to lead to better predictive ability of the RF on new or unseen instances.

The hyperparameter optimization of machine learning models for classification can be formulated as a black-box optimization problem where the decision variables are the hyperparameters and the objective function is an estimate of the predictive performance of the model. In this paper, we only consider discrete ordinal settings for the hyperparameters. However, the methods used here can be extended to deal with continuous and non-ordinal hyperparameter settings. More precisely, the hyperparameter optimization problem is to search for an approximate optimal solution to the following discrete optimization problem:

$$\begin{aligned} & \min_{\Theta \in \mathbb{R}^d} \ \mathcal{E}(\Theta) \qquad (\text{or } \max_{\Theta \in \mathbb{R}^d} \ \mathcal{P}(\Theta)) \\ \text{subject to} \quad & \\ & \theta^{(i)} \in D_i \subset \mathbb{R} \qquad \text{for } i = 1, \ldots, d \end{aligned} \tag{1}$$

Here, $\Theta = (\theta^{(1)}, \ldots, \theta^{(d)})$, where $\theta^{(i)}$ is the ith hyperparameter, and D_i is the finite set of all possible ordinal settings of $\theta^{(i)}$. For example, the hyperparameter sample.frac in RFs can take the values in $\{0.5, 0.55, 0.6, \ldots, 0.95, 1\}$. The objective function $\mathcal{E}(\Theta)$ to be minimized is an estimate of the prediction error of the model such as the *overall out-of-bag (OOB) prediction error*, which is the fraction (or percentage) of misclassified instances in the OOB procedure. The alternative objective function $\mathcal{P}(\Theta)$ to be maximized is a measure of predictive ability of the model such as the *overall OOB accuracy*, *OOB F1 score* or *OOB AUC*. The last two metrics are suitable for unbalanced classification data sets. The *F1 score* is the harmonic mean of the *precision* and *recall* of the model on binary classification tasks. The *precision* of a model is the fraction of true positives among instances that the model labeled as positive while the *recall* is the fraction of true positives among all instances that are actually positive. The *AUC metric* is the area under the *ROC curve* for binary classification problems. This curve is a plot of the true positive rate (recall) vs the false positive rate.

The objective functions $\mathcal{E}(\Theta)$ and $\mathcal{P}(\Theta)$ are black-box since analytical formulas for these functions in terms of the hyperparameters are not available. Moreover, $\mathcal{E}(\Theta)$ and $\mathcal{P}(\Theta)$ are typically random variables, which give rise to stochastic optimization problems. For example, the overall OOB prediction error $\mathcal{E}(\Theta)$ in RFs depends on the bootstrap samples used and the random sample of predictors considered at each of the splits of the decision trees forming the RF. In this

case, we minimize an estimate of the expected value of $\mathcal{E}(\Theta)$ over the same search space. For simplicity, we treat problem (1) as deterministic by fixing the random seed when calculating $\mathcal{E}(\Theta)$ or $\mathcal{P}(\Theta)$. The case where the objective function is a random variable will be treated in future work.

Grid search and random search [2] are popular methods for hyperparameter optimization of machine learning models. However, the objective function $\mathcal{E}(\Theta)$ can become computationally expensive when a machine learning model is trained on a large data set with many thousands or millions of instances and hundreds of predictors. In this case, a complete grid search might not be possible while uniform random search might be inefficient in finding optimal hyperparameters. Instead, a natural approach is to utilize an optimization method that uses a surrogate [21] for the objective function. Examples of surrogate-based methods that have been used for hyperparameter optimization include Bayesian optimization techniques [1,7,12,20] and radial basis function (RBF) methods [5,16]. The surrogates approximate the functional relationship between the hyperparameters and the objective function, and so, a surrogate-based method is expected to outperform uniform random search. In fact, Bayesian optimization was shown to be empirically superior to random search [20] on hyperparameter optimization problems on several machine learning models.

This paper explores the use of surrogate-based methods, particularly RBF and Bayesian optimization methods, to optimize the hyperparameters of a classification RF. We focus on hyperparameters that take values on an ordinal finite set. While Bayesian optimization methods have been used for hyperparameter optimization of RFs (e.g., [12]), RBFs have rarely been used for this purpose. In fact, B-CONDOR-RBF [16] is among the few, if not the only RBF method, that has been used to optimize the hyperparameters of an RF. However, this RBF method was only applied to regression tasks in [16]. In this paper, we apply B-CONDOR-RBF and a discrete Bayesian optimization algorithm to optimize the hyperparameters of an RF for classification tasks. Moreover, this study used data sets with much larger numbers of predictors than in [16]. We utilize a global and a local variant of B-CONDOR-RBF both of which use a cubic RBF model augmented by a linear polynomial. We also utilize two variants of the Bayesian optimization method, one of which uses a Gaussian kernel while the other uses the Matérn 5/2 kernel. We compare the performance of these surrogate-based approaches with the widely used uniform random search method and with a local discrete random search approach on 10 classification data sets that involve up to 753 predictor variables and up to about 19K data points. The results show that the global variant of B-CONDOR-RBF obtained better overall OOB prediction error than the two discrete random search methods and the two Bayesian optimization algorithms given a limited budget of only 100 function evaluations on the hyperparameter optimization problems. Furthermore, a local variant of B-CONDOR-RBF outperformed the random search methods and achieved comparable performance with the Bayesian optimization algorithms on the same problems. These results suggest that RBF methods, particularly B-CONDOR-RBF, are promising alternatives to Bayesian optimization for the hyperparameter optimization of classification RFs and possibly other models.

2 Surrogate-Based Methods for Hyperparameter Optimization of Machine Learning Models

For large datasets involving many instances or predictors, the objective function $\mathcal{E}(\Theta)$ or $\mathcal{P}(\Theta)$ is computationally expensive to evaluate, so we consider two methods that use surrogates of the objective function to optimize the hyperparameters of an RF. Both methods are designed to solve the discrete black-box optimization problem in (1) where the hyperparameters take values from an ordinal finite set. In the case of classification RFs in this study, $\mathcal{E}(\Theta)$ represents the overall OOB prediction error of the RF for a given setting of the hyperparameters $\Theta = (\theta^{(1)}, \ldots, \theta^{(d)})$, which is a point in the search space. However, the objective function may be replaced by a different metric $\mathcal{P}(\Theta)$ to be maximized such as the overall OOB F1 score or the overall OOB AUC score of the model. Below, we focus on minimizing the objective function $\mathcal{E}(\Theta)$ since maximizing $\mathcal{P}(\Theta)$ is equivalent to minimizing its negative. The two algorithms begin by evaluating $\mathcal{E}(\Theta)$ at initial points randomly selected uniformly from the finite search space. The resulting objective function values at the initial points are then used to fit the initial surrogate for $\mathcal{E}(\Theta)$.

2.1 Bayesian Hyperparameter Optimization

In Bayesian optimization, we build a probabilistic model for the objective function $\mathcal{E}(\Theta)$ that yields a surrogate of the objective and a measure of uncertainty that are used to guide the selection of sample points for function evaluation. In particular, we assume that $\mathcal{E}(\Theta)$ is modeled by a Gaussian Process $\mathcal{GP}(\mu(\Theta), k(\Theta, \Theta'))$ where $\mu(\Theta)$ is the mean function and $k(\Theta, \Theta')$ is the covariance function. In this study, we consider a constant mean function $\mu(\Theta) = \mu$ and two types of covariance functions:

$$k(\Theta, \Theta') = \sigma^2 \exp\left(\frac{-\|\Theta - \Theta'\|^2}{2\ell^2}\right) \quad \text{(Gaussian)}$$

$$k(\Theta, \Theta') = \sigma^2 \left(1 + \frac{\sqrt{5}\|\Theta - \Theta'\|}{\ell} + \frac{5\|\Theta - \Theta'\|^2}{3\ell^2}\right) \exp\left(\frac{-\sqrt{5}\|\Theta - \Theta'\|}{\ell}\right) \quad \text{(Matérn 5/2)}$$

where σ^2 and ℓ are parameters to be determined.

Assume that objective function has been evaluated at n points, yielding $\{(\Theta_1, \xi_1), (\Theta_2, \xi_2), \ldots, (\Theta_n, \xi_n)\}$, where $\xi_i = \mathcal{E}(\Theta_i)$ for all $i = 1, \ldots, n$. The Gaussian Process is fit using maximum likelihood estimation, and the objective function value at a new point Θ^* will also have a Gaussian distribution with mean and variance given by [14]:

$$\widehat{\xi}(\Theta^*) = \widehat{\mu} + k_*^T K^{-1}(\xi - \widehat{\mu})$$
$$s^2(\Theta^*) = k(\Theta^*, \Theta^*) - k_*^T K^{-1} k_*$$

where $\widehat{\mu}$ is the MLE of μ, $\xi = [\xi_1, \ldots, \xi_n]^T$, $k_* = [k(\Theta_1, \Theta^*), \ldots, k(\Theta_n, \Theta^*)]^T$, and K is the $n \times n$ covariance matrix whose (i, j) entry is $k(\Theta_i, \Theta_j)$. The predicted value for $\mathcal{E}(\Theta^*)$ is then given by its mean $\widehat{\xi}(\Theta^*)$ while the variance $s^2(\Theta^*)$ provides a measure of uncertainty at the predicted value for $\mathcal{E}(\Theta^*)$.

In this paper, we utilize *Discrete-EI* [16], which is a Bayesian optimization algorithm for bound-constrained black-box optimization problem with discrete decision variables. Discrete-EI uses the Gaussian Process surrogate model described above and the *expected improvement (EI)* criterion [19] as its acquisition function for selecting the next sample point for objective function evaluation. The EI criterion was used in the original Efficient Global Optimization (EGO) method [6]. In each iteration of Discrete-EI, the next sample point is chosen to be a point Θ that maximizes $\mathrm{EI}(\Theta)$ approximately over the search space where

$$\mathrm{EI}(\Theta) = (\xi_{min} - \widehat{\xi}(\Theta))\Phi\left(\frac{\xi_{min} - \widehat{\xi}(\Theta)}{s(\Theta)}\right) + s(\Theta)\phi\left(\frac{\xi_{min} - \widehat{\xi}(\Theta)}{s(\Theta)}\right), \quad (2)$$

if $s(\Theta) > 0$ and $\mathrm{EI}(\Theta) = 0$ if $s(\Theta) = 0$. Here, Φ and ϕ are the cdf and pdf of the standard normal distribution, respectively. Also, ξ_{min} is the current best objective function value.

2.2 The B-CONDOR Algorithm for Hyperparameter Optimization

For the *B-CONDOR* algorithm [16], each iteration generates a large number of random trial points within the search space according to some probability distribution that may adapt as the search progresses. In the *global variant* of B-CONDOR, the trial points are generated uniformly at random throughout the finite search space. In the *local variant*, the trial points are generated in some neighborhood of the current best point. In particular, the trial points in the local variant are obtained by modifying some of the variable settings of the current best point and this process is controlled by two parameters: p_{pert} and $depth_{\mathrm{nbhd}}$. To obtain these trial points, each coordinate of the current best point is modified with probability p_{pert} and the percentage of the number of settings that a variable is allowed to increase or decrease is given by the neighborhood depth parameter $depth_{\mathrm{nbhd}}$. More details can be obtained from [16].

Once the trial points have been generated, the next sample point is selected to be the best trial point according to a weighted ranking of two criteria as in [15]: predicted value of the objective function (*surrogate criterion*), and minimum distance between the trial point and the previous sample points (*distance criterion*). We then evaluate the objective function at the best trial point, which then provides a new data point that is used to update the surrogate model of the objective function for the next iteration. The iterations continue until we reach the maximum number of function evaluations, denoted by $n_{\max}$.

Below is an outline of the B-CONDOR algorithm for problem (1).

The B-CONDOR Algorithm

(1) *(Evaluate Initial Points)* Evaluate $\mathcal{E}(\Theta)$ at space-filling points $\mathcal{I}_0$ from the search space. Set $n \leftarrow |\mathcal{I}_0|$.

(2) *(Iterate)* While $n < n_{\max}$ do:

(a) *(Build Surrogate)* Build/update the surrogate for the objective, $s_n^{\mathcal{E}}(\Theta)$.

(b) *(Generate Trial Points)* Generate random trial points within the search space according to some probability distribution.
(c) *(Evaluate Surrogate at Trial Points)* Evaluate $s_n^{\mathcal{E}}(\Theta)$ at each trial point and compute the min and max value of $s_n^{\mathcal{E}}(\Theta)$ among the trial points.
(d) *(Compute Minimum Distance of Each Trial Point from Previous Sample Points)* Compute $\Delta_n(\Theta)$ at each trial point and calculate the min and max of $\Delta_n(\Theta)$ among all trial points.
(e) *(Compute Ranks)* For each trial point Θ, compute the *Surrogate Rank* $\mathcal{R}_n^S(\Theta) \in [0,1]$, the *Minimum Distance Rank* $\mathcal{R}_n^{\Delta}(\Theta) \in [0,1]$ and the *Weighted Rank*

$$\mathcal{R}_n(\Theta) = w_S \mathcal{R}_n^S(\Theta) + (1 - w_S)\mathcal{R}_n^{\Delta}(\Theta)$$

(f) *(Select Sample Point)* Choose the next sample point Θ_{n+1} to be the trial point Θ that minimizes $\mathcal{R}_n(\Theta)$.
(g) *(Evaluate Selected Point)* Evaluate $\mathcal{E}(\Theta_{n+1})$, update information, and reset $n \leftarrow n+1$.
end.
(3) *(Return Best Point Obtained)* Return the best solution obtained Θ_n^* and the value $\mathcal{E}(\Theta_n^*)$.

As in [16], the B-CONDOR method is implemented using an RBF surrogate model, giving rise to *B-CONDOR-RBF*. Given n distinct points and their objective function values, $(\Theta_1, \mathcal{E}(\Theta_1)), (\Theta_2, \mathcal{E}(\Theta_2)), \ldots, (\Theta_n, \mathcal{E}(\Theta_n))$, we use the RBF model of the form [11]:

$$s_n(\Theta) = \sum_{i=1}^{n} \lambda_i \phi(\|\Theta - \Theta_i\|) + c^T\Theta + c_0,\ \Theta \in \mathbb{R}^d,$$

where $\|\cdot\|$ is the 2-norm, $\lambda_i \in \mathbb{R}$ for $i = 1, \ldots, n$, $c \in \mathbb{R}^d$ and $c_0 \in \mathbb{R}$. Note that each point Θ_i is a center and $c^T\Theta + c_0$ is simply a linear polynomial in d variables. Moreover, the basis functions can take various forms such as the cubic ($\phi(r) = r^3$), thin plate spline ($\phi(r) = r^2 \log r$), and Gaussian ($\phi(r) = \exp(-\gamma r^2)$ forms. Training this model is fast and is relatively straightforward since it simply involves solving a linear system with good numerical properties [11].

3 Computational Experiments

3.1 Real-World Classification Problems Used for Hyperparameter Optimization of Random Forests

All computational experiments are carried out in R [13] through RStudio [18] on an Intel(R) Core(TM) i7-7700T CPU @ 2.90 GHz, 2904 Mhz, 4 Core(s) Windows-based machine. The RFs are trained and evaluated on the data sets using the *ranger* R package [22], which provides a fast implementation of RFs for high-dimensional data. Specifically, the ranger() function is used to calculate the OOB prediction error of the RF on a dataset.

Table 1. Characteristics of the classification problems for RF hyperparameter optimization. The training set consists of 70% of the data set and is fixed in this study.

Tuning problem	Number of predictors	Number of classes	Number of data points	Size of search space
Job Scheduling	7	4	4331	7700
Gamma Telescope	10	2	19020	11000
Red Wine Quality	11	6	1599	12100
White Wine Quality	11	7	4898	12100
Forest Types	27	4	523	29700
Alzheimer	130	2	333	143000
Urban Land Cover	147	9	675	161700
Hepatic	376	3	281	413600
Madelon	500	2	2000	550000
Parkinson	753	2	756	828300

The B-CONDOR-RBF and Discrete-EI algorithms are applied to optimize the hyperparameters of an RF on 10 classification data sets with no missing values. The characteristics of these hyperparameter optimization problems are given in Table 1. The *High Performance Computing (HPC) Job Scheduling*, *Alzheimer's Disease*, and *Hepatic Injury* data sets are part of the *AppliedPredictiveModeling* R package [8,9]. The *Gamma Telescope*, *Red Wine Quality*, *White Wine Quality*, *Forest Types*, *Urban Land Cover*, *Madelon*, and *Parkinson's Disease* data sets come from the UC Irvine Machine Learning Repository [4].

As mentioned earlier, we tune four hyperparameters of a classification RF: (1) number of decision trees, denoted by *ntree*; (2) number of predictors drawn at random for each split, denoted by *mtry*; (3) minimum number of samples in a node, denoted by *min.node.size*; and (4) fraction of the training set drawn at random with replacement for constructing each tree, denoted by *sample.frac*. This results in a 4-dimensional bound-constrained discrete black-box optimization problem. The typical recommended settings of these hyperparameters for classification RFs [12] are: ntree $= 500$, mtry $= \lceil \sqrt{p} \rceil$ (where p is the number of predictors), min.node.size $= 1$, and sample.frac $= 1$ (since we are sampling with replacement). These recommended settings are included in the possible values of the hyperparameters in the optimization problem: ntree $\in \{100, 200, \ldots, 1000\}$, mtry $\in \{1, 2, \ldots, p\}$, min.node.size $\in \{1, 2, \ldots, 10\}$, and sample.frac $\in \{0.50, 0.55, 0.60, \ldots, 0.95, 1.0\}$. Hence, the total number of distinct solutions in the search space for the optimization problem is $10 \times p \times 10 \times 11 = 1100p$.

Before any experiments are performed, we split each data set into a training set (70%) and a test set (30%) only once and the *objective function* is defined as the *overall out-of-bag (OOB) prediction error of the RF on the training set*. We use the same training set and fix the seed in the ranger() function when calculating the OOB prediction error for any setting of the RF hyperparameters. This ensures that each RF tuning problem becomes deterministic. Moreover, all the decision variables of the RF tuning problem are rescaled to range from 0 to 1. After the different optimization methods are used to minimize the objective

function for each data set, the best RFs (in terms of OOB prediction error on the training set) obtained by each algorithm are evaluated on the fixed test set.

3.2 Experimental Setup

We run six algorithms for the hyperparameter optimization of the RF on the 10 classification problems: four surrogate-based methods and two random search methods. Among the surrogate-based methods are two variants of B-CONDOR-RBF [16]: B-CONDOR-RBF-Local and B-CONDOR-RBF-Global, which both use a cubic RBF model augmented by a linear polynomial. The cubic type of RBF was chosen due to its simplicity and previous good results when used with RBF methods (e.g., [5,15]). Each iteration of the RBF methods generates $100d$ random trial points from the search space and the RBF surrogate is used to select the most promising trial solution using a weighted combination of the surrogate and distance criteria of 0.95 and 0.05, respectively, as in [15]. In the B-CONDOR-RBF-Local algorithm, the random trial points are generated in the neighborhood of the current best solution. The generation of these random trial points is controlled by two parameters namely, the perturbation probability p_{pert} and the neighborhood depth parameter $depth_{\text{nbhd}}$. In particular, the values of p_{pert} and $depth_{\text{nbhd}}$ vary through the iterations according to the cycle $\langle(1, 50\%), (0.5, 50\%), (0.5, 25\%), (0.5, 10\%), (0.5, 5\%)\rangle$. This choice of parameter values yields iterations that follow a cycle beginning with a more global search and ending with a more local search. For B-CONDOR-RBF-Global, the random trial points are simply generated uniformly over the entire discrete search space.

As mentioned earlier, the Discrete-EI method that handles ordinal discrete variables uses the expected improvement (EI) criterion from the EGO algorithm [6] to select its sample point. We run Discrete-EI using the Gaussian and Matérn 5/2 kernels for the covariance function. In Discrete-EI, the next sample point is an approximate maximizer of the EI function over the entire discrete search space obtained by discrete uniform random search for $100d$ iterations. This Discrete-EI implementation uses the km() and EI() functions from the DiceKriging [17] and DiceOptim [10] R packages, respectively.

Random search is the simplest and one of the most widely used method for the hyperparameter optimization of machine learning models. Hence, we include a *Discrete Global Random Search* method to serve as a baseline for comparison with the surrogate-based methods. In this method, the next function evaluation point is simply chosen uniformly at random in the search space of hyperparameters. Moreover, we also implement the *Discrete Local Random Search* method from [16] since it was shown to be consistently better than Discrete Global Random Search on the hyperparameter optimization of RFs for the regression data sets used in that study. In Discrete Local Random Search, the next function evaluation point is generated using the perturbation probability p_{pert} and the neighborhood depth parameter $depth_{\text{nbhd}}$ from B-CONDOR-RBF-Local and varying using the same cycle as in the algorithm except there is no RBF involved and only one point is generated in every iteration.

The surrogate-based and random search algorithms are run for 30 trials on the 10 RF hyperparameter optimization problems on classification data sets. For fair comparison, all algorithms used the same $5(d+1)$ initial points in a given trial run. However, different trial runs used different sets of $5(d+1)$ initial points where the objective function (overall OOB prediction error) is evaluated. For each set of initial points, we include the typical recommended values of the RF hyperparameters for classification problems given above as the first initial point. Then, we choose $5d+4$ points uniformly at random on the hyperparameter search space for the other initial points. Moreover, we run each algorithm for a limited budget of $20(d+1)$ function evaluations since the RF hyperparameter optimization problems can be computationally expensive especially on data sets containing many predictor variables and/or a large number of data points. The number of function evaluations is expressed in factors of $d+1$ since this is the minimum number of points needed to fit the simplest surrogate, which is a linear model. We include the recommended RF hyperparameter settings since they have been known to work well in many cases [12], so this allows us to determine how much further improvement can be made using the surrogate-based hyperparameter optimization methods when there is a very limited budget on the number of function evaluations.

3.3 Results and Discussion

Tables 2 and 3 show the mean and standard error, taken over 30 trials, of the best objective function value (OOB prediction error on the fixed training set) obtained by the surrogate-based and random search algorithms at different computational budgets on the RF hyperparameter optimization problems. In each row of these tables, the best is indicated by a solid blue box while the second best is indicated by a dashed magenta box. Moreover, Figs. 1, 2 and 3 show the plots of the mean of the best overall OOB prediction error obtained by the six algorithms on the RF hyperparameter optimization problems for the classification data sets at different computational budgets beginning with the initial design points of size $5(d+1)$ up to a maximum of $20(d+1)$ function evaluations. These plots also include error bars that represent 95% t confidence intervals for the mean.

The results in Tables 2 and 3 and in Figs. 1, 2 and 3 show that the RBF algorithms and the Bayesian optimization methods are consistently superior to Discrete Global Random Search on the 10 RF hyperparameter optimization problems on classification data sets. This confirms that surrogate-based approaches are promising for hyperparameter optimization of RFs on classification data sets. Among the surrogate-based approaches, B-CONDOR-RBF-Global achieved the best results on most of the problems. In particular, B-CONDOR-RBF-Global obtained generally better results than Discrete-EI with the Gaussian kernel on 9 of the 10 problems (all except on the Forest Types data set). It also obtained better results than Discrete-EI with the Matérn 5/2 kernel on 8 of the 10 problems (all except on the Forest Types and Madelon data sets). Moreover, B-CONDOR-RBF-Global outperformed the local variant B-CONDOR-RBF-Local on 9 of the

Table 2. Mean and standard error of best overall OOB prediction error found by different methods at various computational budgets on the RF tuning problems. The mean is taken over 30 trials. The best and second best results in each row are enclosed by a solid blue box and a dashed magenta box, respectively.

Tuning Problem	Number of evaluations	B-CONDOR-RBF-Local (cubic)	B-CONDOR-RBF-Global (cubic)	Discrete-EI (Matérn 5/2)	Discrete-EI (Gaussian)	Discrete Local Random Search	Discrete Global Random Search
HPC Job Scheduling	10(d+1)	0.1567 (1.54e-04)	0.1570 (1.68e-04)	0.1575 (1.58e-04)	0.1576 (1.57e-04)	0.1572 (1.60e-04)	0.1583 (1.42e-04)
	15(d+1)	0.1566 (1.36e-04)	0.1565 (1.50e-04)	0.1569 (1.51e-04)	0.1570 (1.48e-04)	0.1569 (1.36e-04)	0.1581 (1.51e-04)
	20(d+1)	0.1565 (1.28e-04)	0.1563 (1.37e-04)	0.1567 (1.48e-04)	0.1569 (1.40e-04)	0.1566 (1.34e-04)	0.1581 (1.50e-04)
Gamma Telescope	10(d+1)	0.1207 (5.64e-05)	0.1206 (6.01e-05)	0.1207 (5.47e-05)	0.1208 (6.62e-05)	0.1206 (6.17e-05)	0.1211 (7.50e-05)
	15(d+1)	0.1206 (5.82e-05)	0.1205 (5.58e-05)	0.1206 (4.39e-05)	0.1206 (7.52e-05)	0.1205 (5.59e-05)	0.1208 (6.35e-05)
	20(d+1)	0.1206 (5.48e-05)	0.1204 (5.10e-05)	0.1205 (4.39e-05)	0.1205 (6.13e-05)	0.1204 (5.35e-05)	0.1207 (6.10e-05)
Red Wine	10(d+1)	0.3601 (1.79e-04)	0.3595 (7.71e-05)	0.3598 (1.83e-04)	0.3598 (1.74e-04)	0.3609 (2.07e-04)	0.3630 (2.21e-04)
	15(d+1)	0.3598 (1.35e-04)	0.3593 (3.02e-05)	0.3595 (1.64e-04)	0.3596 (1.65e-04)	0.3602 (1.66e-04)	0.3626 (2.22e-04)
	20(d+1)	0.3596 (1.11e-04)	0.3593 (2.21e-05)	0.3594 (1.60e-04)	0.3595 (1.61e-04)	0.3598 (1.35e-04)	0.3621 (2.43e-04)
White Wine	10(d+1)	0.3745 (5.49e-05)	0.3743 (3.79e-05)	0.3744 (9.29e-05)	0.3746 (1.00e-04)	0.3751 (1.15e-04)	0.3765 (1.26e-04)
	15(d+1)	0.3744 (5.16e-05)	0.3743 (2.11e-05)	0.3743 (9.06e-05)	0.3744 (9.54e-05)	0.3748 (7.16e-05)	0.3764 (1.31e-04)
	20(d+1)	0.3744 (4.99e-05)	0.3742 (8.15e-06)	0.3743 (8.97e-05)	0.3743 (9.05e-05)	0.3746 (7.60e-05)	0.3762 (1.44e-04)
Forest Types	10(d+1)	0.0899 (3.55e-04)	0.0897 (2.95e-04)	0.0895 (3.63e-04)	0.0893 (4.41e-04)	0.0905 (5.36e-04)	0.0907 (4.61e-04)
	15(d+1)	0.0891 (3.10e-04)	0.0891 (2.81e-04)	0.0883 (2.73e-04)	0.0883 (3.25e-04)	0.0892 (4.99e-04)	0.0900 (4.53e-04)
	20(d+1)	0.0890 (3.12e-04)	0.0883 (2.67e-04)	0.0879 (2.30e-04)	0.0881 (2.83e-04)	0.0885 (4.06e-04)	0.0898 (3.87e-04)

Table 3. Mean and standard error of best overall OOB prediction error found by different methods at various computational budgets on the RF tuning problems. The mean is taken over 30 trials. The best and second best results in each row are enclosed by a solid blue box and a dashed magenta box, respectively.

Tuning Problem	Number of evaluations	B-CONDOR-RBF-Local (cubic)	B-CONDOR-RBF-Global (cubic)	Discrete-EI (Matérn 5/2)	Discrete-EI (Gaussian)	Discrete Local Random Search	Discrete Global Random Search
Alzheimer	10(d+1)	0.1565 (8.42e-04)	0.1567 (9.14e-04)	0.1582 (1.04e-03)	0.1592 (8.32e-04)	0.1575 (1.01e-03)	0.1612 (9.13e-04)
	15(d+1)	0.1554 (8.33e-04)	0.1549 (6.93e-04)	0.1565 (8.67e-04)	0.1579 (7.25e-04)	0.1564 (8.15e-04)	0.1587 (8.60e-04)
	20(d+1)	0.1546 (7.83e-04)	0.1541 (7.23e-04)	0.1542 (7.40e-04)	0.1565 (7.05e-04)	0.1551 (8.67e-04)	0.1577 (8.21e-04)
Urban Land	10(d+1)	0.1428 (4.50e-04)	0.1425 (4.65e-04)	0.1433 (4.25e-04)	0.1437 (4.20e-04)	0.1421 (4.65e-04)	0.1441 (4.00e-04)
	15(d+1)	0.1421 (3.93e-04)	0.1415 (4.84e-04)	0.1422 (4.63e-04)	0.1429 (4.73e-04)	0.1415 (4.91e-04)	0.1433 (3.87e-04)
	20(d+1)	0.1416 (3.81e-04)	0.1407 (4.73e-04)	0.1416 (4.55e-04)	0.1423 (4.20e-04)	0.1408 (4.51e-04)	0.1423 (3.95e-04)
Hepatic	10(d+1)	0.4422 (1.53e-03)	0.4417 (1.98e-03)	0.4434 (1.70e-03)	0.4425 (1.83e-03)	0.4423 (1.70e-03)	0.4466 (1.65e-03)
	15(d+1)	0.4403 (1.37e-03)	0.4395 (1.89e-03)	0.4400 (1.52e-03)	0.4395 (1.71e-03)	0.4401 (1.53e-03)	0.4439 (1.62e-03)
	20(d+1)	0.4386 (1.31e-03)	0.4366 (1.69e-03)	0.4378 (1.46e-03)	0.4367 (1.52e-03)	0.4381 (1.48e-03)	0.4401 (1.58e-03)
Madelon	10(d+1)	0.1806 (4.01e-04)	0.1818 (4.92e-04)	0.1813 (4.75e-04)	0.1832 (5.03e-04)	0.1838 (5.75e-04)	0.1868 (7.76e-04)
	15(d+1)	0.1799 (3.27e-04)	0.1805 (3.86e-04)	0.1799 (3.89e-04)	0.1808 (4.67e-04)	0.1826 (5.99e-04)	0.1859 (7.94e-04)
	20(d+1)	0.1790 (4.16e-04)	0.1797 (4.54e-04)	0.1795 (3.10e-04)	0.1804 (4.12e-04)	0.1810 (6.14e-04)	0.1849 (7.42e-04)
Parkinsons	10(d+1)	0.1302 (4.47e-04)	0.1298 (3.18e-04)	0.1302 (4.77e-04)	0.1308 (3.78e-04)	0.1306 (4.28e-04)	0.1317 (3.55e-04)
	15(d+1)	0.1292 (3.90e-04)	0.1290 (3.13e-04)	0.1295 (4.32e-04)	0.1302 (3.78e-04)	0.1301 (4.29e-04)	0.1309 (3.62e-04)
	20(d+1)	0.1285 (4.30e-04)	0.1280 (3.73e-04)	0.1289 (4.74e-04)	0.1295 (4.32e-04)	0.1292 (4.48e-04)	0.1304 (3.14e-04)

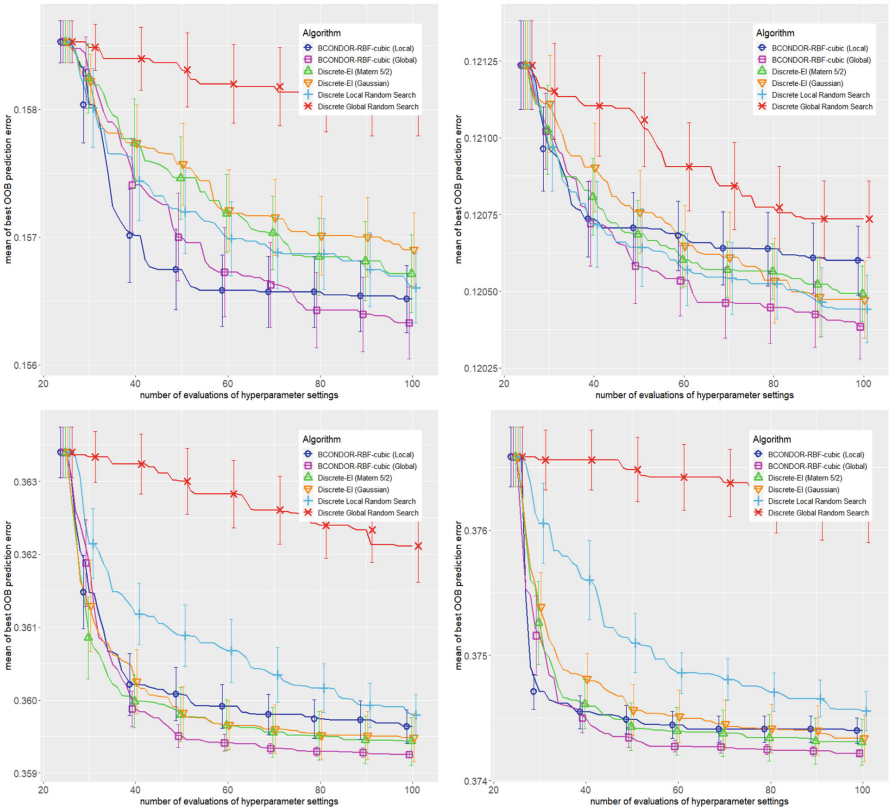

Fig. 1. Mean of the OOB prediction error obtained by the different methods at various computational budgets on four of the hyperparameter optimization problems for classification RFs: Job Scheduling (top left), Gamma Telescope (top right), Red Wine Quality (bottom left) and White Wine Quality (bottom right)

problems, but the latter obtained better results than the former on the Madelon data set involving 500 predictors and 2000 instances. It would be of interest to know for which type of data sets will the local variant yield better results.

Next, as mentioned above, B-CONDOR-RBF-Local consistently outperformed Discrete Global Random Search on all 10 RF hyperparameter optimization problems. It is also better than Discrete Local Random Search on seven of the problems (all except on the Gamma Telescope, Urban Land and Hepatic data sets). Moreover, B-CONDOR-RBF-Local achieved comparable performance with the Bayesian optimization approaches. In particular, B-CONDOR-RBF-Local is better than Discrete-EI with the Gaussian kernel on five of the problems and it is competitive with the latter on another three problems. Also, B-CONDOR-RBF-Local is better than Discrete-EI with the Matérn 5/2 kernel on four of the 10 problems and it is competitive with the latter on another two problems.

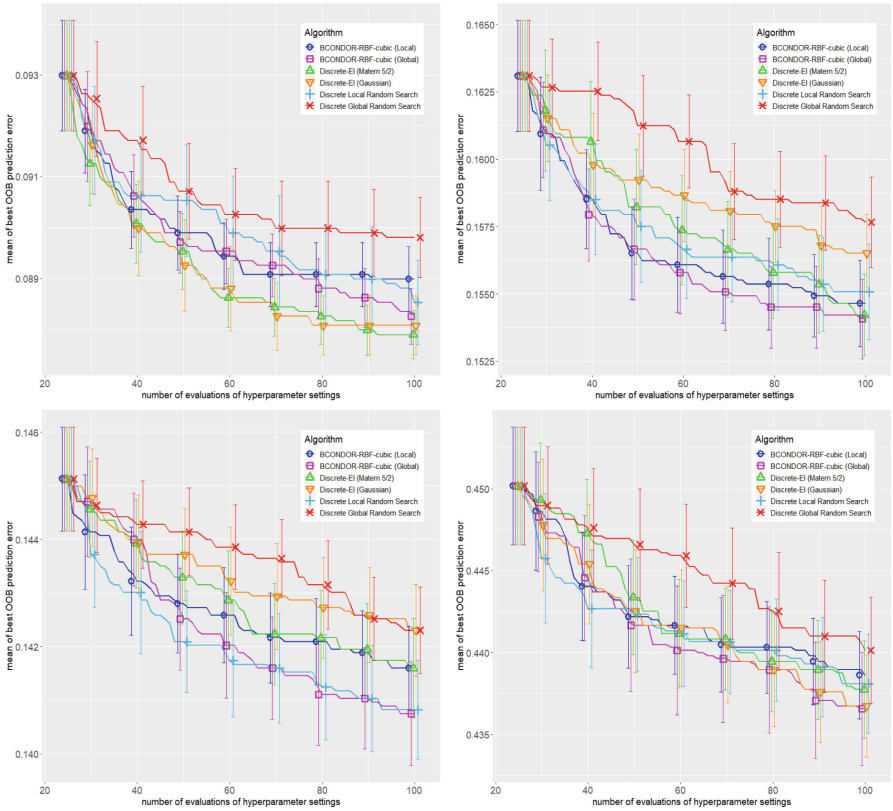

Fig. 2. Mean of the OOB prediction error obtained by the different methods at various computational budgets on four of the hyperparameter optimization problems for classification RFs: Forest Types (top left), Alzheimer's Disease (top right), Urban Land (bottom left) and Hepatic (bottom right).

These results complement those obtained in [16] on the hyperparameter optimization of RFs for regression data sets. In that paper, B-CONDOR-RBF algorithms obtained much better OOB RMSE than Discrete Global Random Search on RF hyperparameter tuning problems on seven regression data sets. They also obtained better or comparable results to Discrete-EI with the Gaussian kernel and the Discrete Local Random Search method. In that paper, Discrete-EI with the Matérn 5/2 kernel was not used.

Next, the Bayesian optimization algorithms (Discrete-EI with the Matérn 5/2 and Gaussian kernels) yielded consistently better results than Discrete Global Random Search on all 10 problems. This is consistent with the findings by Turner et al. [20]. However, compared with Discrete Local Random Search, which does not use any surrogates, the Bayesian approach does not always yield the better result. In particular, Discrete-EI with the Matérn 5/2 kernel obtained better results than Discrete Local Random Search on six of the problems while the

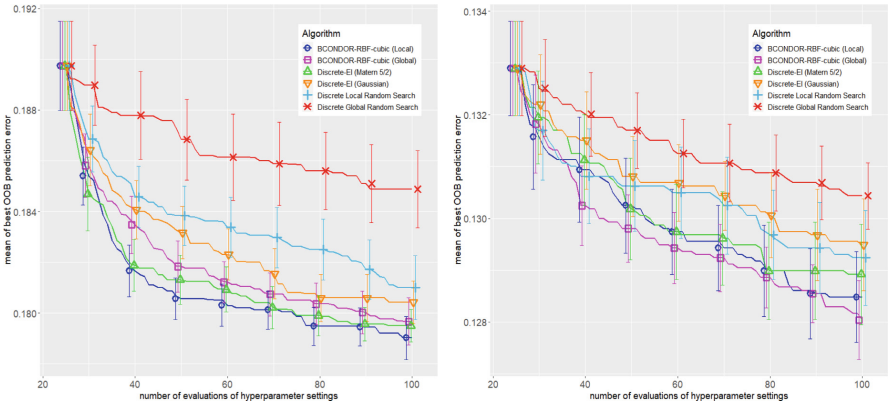

Fig. 3. Mean of the OOB prediction error obtained by the different methods at various computational budgets on two of the hyperparameter optimization problems for classification RFs: Madelon (left), Parkinson's Disease (right).

latter obtained better results than the former on three of the problems. On the Alzheimer's data set, the two methods achieved comparable results. Moreover, Discrete-EI with the Gaussian kernel obtained better results than Discrete Local Random Search on half of the problems, while the latter obtained better results than the former on the remaining half of the problems. For the Bayesian optimization methods, the Matérn 5/2 kernel yielded better results than the Gaussian kernel on seven of the problems while the latter is better than the former only on the Hepatic data set. The two Bayesian methods obtained comparable results on the remaining two problems involving the Gamma Telescope and Forest Types data sets.

As was observed in the hyperparameter optimization of RF for regression problems in [16], Discrete Local Random Search method consistently outperformed Discrete Global Random Search on all 10 RF hyperparameter optimization problems for classification data sets. Moreover, although Discrete Local Random Search does not use a surrogate, it was quite competitive with the Bayesian method Discrete-EI that uses the Gaussian kernel on the same RF tuning problems and it obtained better results than Discrete-EI with the Matern 5/2 kernel on three of the problems. Discrete Local Random Search generates its sample points in the same manner as in B-CONDOR-RBF-Local except no surrogate is involved. This again indicates that this local scheme for choosing sample points is a promising alternative for the widely used global random search for hyperparameter optimization problems, especially when there is a limited budget on function evaluations.

Recall that in each trial run of every algorithm, the first point in the initial design of $5(d+1)$ points consists of the recommended RF hyperparameter settings for classification problems. Table 4 shows the mean and standard error (over 30 trials) of the percentage improvement in the best overall OOB Prediction Error

Table 4. Mean and standard error (over 30 trials) of the percent improvement in OOB prediction error found by the hyperparameter optimization methods after $20(d+1)$ evaluations over the one obtained by using the recommended hyperparameters on RF tuning problems. The best and second best in each row are indicated by a solid blue box and a dashed magenta box, respectively.

Tuning Problem	B-CONDOR-RBF (Local)	B-CONDOR-RBF (Global)	Discrete-EI (Matérn 5/2)	Discrete-EI (Gaussian)	Discrete Local Random Search	Discrete Global Random Search
Job Sched	1.37 (8.1e-02)	1.49 (8.6e-02)	1.25 (9.4e-02)	1.13 (8.8e-02)	1.32 (8.4e-02)	0.37 (9.5e-02)
Telescope	1.07 (4.5e-02)	1.24 (4.2e-02)	1.16 (3.6e-02)	1.17 (5.0e-02)	1.20 (4.4e-02)	0.96 (5.0e-02)
Red Wine	1.19 (3.1e-02)	1.30 (6.1e-03)	1.25 (4.4e-02)	1.23 (4.4e-02)	1.15 (3.7e-02)	0.51 (6.7e-02)
White Wine	0.66 (1.3e-02)	0.71 (2.2e-03)	0.69 (2.4e-02)	0.68 (2.4e-02)	0.62 (2.0e-02)	0.19 (3.8e-02)
Forest Types	9.54 (3.2e-01)	10.28 (2.7e-01)	10.65 (2.3e-01)	10.46 (2.9e-01)	10.00 (4.1e-01)	8.70 (3.9e-01)
Alzheimer	14.21 (4.3e-01)	14.52 (4.0e-01)	14.44 (4.1e-01)	13.17 (3.9e-01)	13.97 (4.8e-01)	12.54 (4.6e-01)
Urban Land	8.45 (2.5e-01)	9.00 (3.1e-01)	8.45 (2.9e-01)	7.99 (2.7e-01)	8.95 (2.9e-01)	7.99 (2.6e-01)
Hepatic	7.56 (2.8e-01)	7.99 (3.6e-01)	7.74 (3.1e-01)	7.96 (3.2e-01)	7.67 (3.1e-01)	7.24 (3.3e-01)
Madelon	45.04 (1.3e-01)	44.84 (1.4e-01)	44.89 (9.5e-02)	44.61 (1.3e-01)	44.43 (1.9e-01)	43.24 (2.3e-01)
Parkinsons	5.60 (3.2e-01)	5.93 (2.7e-01)	5.28 (3.5e-01)	4.86 (3.2e-01)	5.05 (3.3e-01)	4.17 (2.3e-01)

over the recommended hyperparameter settings obtained by the algorithms after $20(d+1)$ function evaluations on the RF hyperparameter optimization problems. As before, the best and second best for each problem are indicated by a solid blue box and a dashed magenta box, respectively. This table also shows that the B-CONDOR-RBF-Global algorithm has the best improvements over the recommended hyperparameters on eight of the problems (all except on the Forest Types and Madelon data sets). The next best improvements were obtained by Discrete-EI with the Matérn 5/2 kernel followed by B-CONDOR-RBF-Local and Discrete-EI with the Gaussian kernel, which have comparable results. These results are somewhat consistent with those obtained in Tables 2 and 3. Moreover, the improvements over the recommended hyperparameters are considerable with the highest mean percentage improvement at 45.04% obtained by B-CONDOR-RBF-Local on the RF tuning problem on the Madelon data set.

Finally, Table 5 shows the mean and standard error (over 30 trials) of the prediction error (percent of misclassified instances) on the test set of the RF whose hyperparameters are obtained by the various methods after $20(d+1)$ evaluations. This table shows that the algorithms that obtained the smallest OOB prediction error do not always get the smallest prediction error on the test set. In particular, B-CONDOR-RBF-Local obtained better test set prediction error than B-CONDOR-RBF-Global on six of the problems. Moreover, it obtained better test set prediction error than Discrete-EI with the Matérn 5/2 kernel on five of the problems while the results are comparable on the Alzheimer data set. These results are not necessarily inconsistent with those in the previous tables since some of the test sets have relatively few instances and the means of the prediction errors are not too far away from one another. In particular, B-CONDOR-RBF-Global, which has the best results in the previous tables, has test set prediction errors that are competitive with the other surrogate methods. In addition, it obtained better or comparable test set prediction errors than Discrete Global Random Search on six of the problems.

Table 5. Mean and standard error (over 30 trials) of the prediction error (expressed as a percentage) on the test set of the RF whose hyperparameters are obtained by the various methods after $20(d+1)$ evaluations. The best and second best in each row are indicated by a solid blue box and a dashed magenta box, respectively.

Tuning Problem	B-CONDOR-RBF (Local)	B-CONDOR-RBF (Global)	Discrete-EI (Matérn 5/2)	Discrete-EI (Gaussian)	Discrete Local Random Search	Discrete Global Random Search
Job Sched	16.49 (3.5e-02)	16.51 (3.6e-02)	16.47 (4.5e-02)	16.58 (4.6e-02)	16.56 (4.2e-02)	16.38 (5.0e-02)
Telescope	11.43 (1.7e-02)	11.46 (1.8e-02)	11.44 (1.5e-02)	11.44 (1.5e-02)	11.45 (1.7e-02)	11.46 (2.0e-02)
Red Wine	29.93 (4.7e-02)	29.78 (6.9e-03)	29.76 (3.7e-02)	29.79 (2.6e-02)	29.74 (8.6e-02)	29.74 (1.2e-01)
White Wine	33.04 (2.4e-02)	33.10 (1.5e-02)	33.12 (2.8e-02)	33.10 (3.2e-02)	33.14 (4.7e-02)	33.74 (5.4e-02)
Forest Types	13.46 (8.5e-02)	13.61 (6.5e-02)	13.69 (6.7e-02)	13.74 (7.3e-02)	13.55 (6.8e-02)	13.55 (7.4e-02)
Alzheimer	10.00 (1.9e-01)	9.80 (1.5e-01)	10.00 (1.9e-01)	10.00 (2.1e-01)	10.00 (1.9e-01)	9.97 (1.8e-01)
Urban Land	11.36 (8.5e-02)	11.36 (7.4e-02)	11.35 (1.0e-01)	11.33 (7.8e-02)	11.44 (8.7e-02)	11.44 (1.1e-01)
Hepatic	40.67 (3.6e-01)	40.86 (4.5e-01)	40.98 (4.0e-01)	40.75 (3.8e-01)	39.96 (4.2e-01)	40.39 (4.8e-01)
Madelon	17.39 (1.2e-01)	17.58 (1.1e-01)	17.59 (9.7e-02)	17.53 (9.4e-02)	17.58 (1.1e-01)	17.83 (1.1e-01)
Parkinsons	9.24 (1.5e-01)	8.96 (1.2e-01)	9.03 (1.2e-01)	9.15 (1.1e-01)	9.57 (1.7e-01)	9.49 (1.2e-01)

4 Summary and Future Work

This paper explored the use of surrogate-based approaches that handle discrete variables, particularly the RBF method B-CONDOR-RBF [16] and a Bayesian optimization method called Discrete-EI, for the hyperparameter optimization of a classification random forest (RF). A global and a local variant of B-CONDOR-RBF and two variants of Discrete-EI are compared with a discrete global random search method and a discrete local random search method on 10 RF hyperparameter optimization problems on classification data sets involving up to 753 predictor variables and up to 19K data points. The B-CONDOR-RBF algorithms used a cubic RBF augmented by a linear polynomial while the Bayesian optimization methods used the Gaussian and Matérn 5/2 kernels. The RF hyperparameter optimization problem involves tuning four RF hyperparameters, so the problem has four decision variables and the objective function to be minimized is the overall out-of-bag (OOB) prediction error of the RF on the training set, which is represented as a fraction of misclassified instances.

The results showed that the B-CONDOR-RBF algorithms (global and local variants) and Discrete-EI algorithms (using the Matérn 5/2 and Gaussian kernels) obtained consistently much better overall OOB prediction error than the Discrete Global Random Search method on all RF hyperparameter optimization problems. Among the four surrogate-based methods, the global variant of B-CONDOR-RBF obtained the best overall OOB prediction error on most of the problems. Moreover, the local variant of B-CONDOR-RBF is an improvement over Discrete Local Random Search and it is better or competitive with the Bayesian optimization algorithm Discrete-EI with the Gaussian kernel on most of the problems. Also, it is competitive with Discrete-EI with the Matérn 5/2 kernel. Among the two Bayesian algorithms, the one that uses the Matérn 5/2 kernel obtained generally better results than the one using the Gaussian kernel. In addition, between the two random search methods, the Discrete Local

Random Search method consistently outperformed the Discrete Global Random Search method. It is also worth noting that the Discrete Local Random Search approach, which does not use a surrogate, was competitive with the Bayesian optimization algorithm that uses the Gaussian kernel on the test problems considered. Compared to random search methods and Bayesian optimization methods, RBF methods are not as popular for the hyperparameter optimization of machine learning models. In fact, this paper is among the few, if not the only one, that used RBF surrogates for the hyperparameter optimization of RFs for classification problems.

Previous work showed that B-CONDOR-RBF works well on regression data sets compared to alternatives such as Bayesian optimization with the Gaussian kernel. This paper showed that it is also promising for the hyperparameter optimization of RF on classification data sets. In future work, we consider other metrics such as the OOB F1 score or OOB AUC score for the objective function of the RF hyperparameter optimization problem on classification data sets. Moreover, we consider other surrogate-based discrete optimization methods that use a different strategy for the selection of sample points. In particular, one can use a nonlinear discrete optimization algorithm to perform a more thorough search on the surrogate surface to determine the next sample point for function evaluation. We can also run B-CONDOR-RBF using other types of RBF such as the thin plate spline or the Gaussian and compare its performance when using the cubic type. When the data set involves thousands of predictor variables and hundreds of thousands or millions of data points, the hyperparameter optimization problem becomes even more computationally expensive. However, the number of decision variables, which is the number of hyperparameters to tune, remains the same. It would be interest to develop more efficient algorithms when dealing with much larger data sets. Finally, it is also of interest to compare the performance of the B-CONDOR-RBF algorithms with the Bayesian methods on hyperparameter tuning problems involving deep neural networks, support vector machines and other machine learning models.

References

1. Archetti, F., Candelieri, A.: Bayesian Optimization and Data Science. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-24494-1
2. Bergstra, J., Bengio, Y.: Random search for hyper-parameter optimization. J. Mach. Learn. Res. **13**, 281–305 (2012)
3. Breiman, L.: Random forests. Mach. Learn. **45**, 5–32 (2001)
4. Dua, D., Graff, C.: UCI Machine Learning Repository. University of California, School of Information and Computer Science, Irvine (2019). http://archive.ics.uci.edu/ml
5. Ilievski, I., Akhtar, T., Feng, J., Shoemaker, C.: Efficient hyperparameter optimization for deep learning algorithms using deterministic RBF surrogates. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 31, no. 1 (2017)
6. Jones, D.R., Schonlau, M., Welch, W.J.: Efficient global optimization of expensive black-box functions. J. Glob. Optim. **13**, 455–492 (1998)

7. Joy, T.T., Rana, S., Gupta, S., Venkatesh, S.: Hyperparameter tuning for big data using Bayesian optimisation. In: 2016 23rd International Conference on Pattern Recognition (ICPR), pp. 2574–2579 (2016). https://doi.org/10.1109/ICPR.2016.7900023
8. Kuhn, M., Johnson, K.: Applied Predictive Modeling. Springer, New York (2013). https://doi.org/10.1007/978-1-4614-6849-3
9. Kuhn, M., Johnson, K.: AppliedPredictiveModeling: Functions and Data Sets for 'Applied Predictive Modeling'. R package version 1.1-7 (2018). https://CRAN.R-project.org/package=AppliedPredictiveModeling
10. Picheny, V., Ginsbourger, D., Roustant, O.: DiceOptim: Kriging-Based Optimization for Computer Experiments. R package version 2.1.1 (2021). https://CRAN.R-project.org/package=DiceOptim
11. Powell, M.J.D.: The theory of radial basis function approximation in 1990. In: Light, W. (ed.) Advances in Numerical Analysis, Volume 2: Wavelets, Subdivision Algorithms and Radial Basis Functions, pp. 105–210. Oxford University Press, Oxford (1992)
12. Probst, P., Wright, M.N., Boulesteix, A.-L.: Hyperparameters and tuning strategies for random forest. WIREs Data Min. Knowl. Discov. **9**(3), e1301 (2019)
13. R Core Team: R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria (2021). https://www.R-project.org/
14. Rasmussen, C.E., Williams, C.K.I.: Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning). The MIT Press (2006)
15. Regis, R.G.: Stochastic radial basis function algorithms for large-scale optimization involving expensive black-box objective and constraint functions. Comput. Oper. Res. **38**(5), 837–853 (2011)
16. Regis, R.G.: Hyperparameter tuning of random forests using radial basis function models. In: Nicosia, G., et al. (eds.) LOD 2022. LNCS, vol. 13810, pp. 309–324. Springer, Cham (2023). https://doi.org/10.1007/978-3-031-25599-1_23
17. Roustant, O., Ginsbourger, D., Deville, Y.: DiceKriging, DiceOptim: two R packages for the analysis of computer experiments by kriging-based metamodeling and optimization. J. Stat. Softw. **51**(1), 1–55 (2012)
18. RStudio Team: RStudio: Integrated Development for R. RStudio, PBC, Boston, MA (2020). http://www.rstudio.com/
19. Schonlau, M.: Computer experiments and global optimization. Ph.D. thesis, University of Waterloo, Canada (1997)
20. Turner, R., et al.: Bayesian optimization is superior to random search for machine learning hyperparameter tuning: analysis of the black-box optimization challenge 2020. In: Proceedings of the NeurIPS 2020 Competition and Demonstration Track, in Proceedings of Machine Learning Research, vol. 133, pp. 3–26 (2021)
21. Vu, K.K., D'Ambrosio, C., Hamadi, Y., Liberti, L.: Surrogate-based methods for black-box optimization. Int. Trans. Oper. Res. **24**, 393–424 (2017)
22. Wright, M.N., Ziegler, A.: Ranger: a fast implementation of random forests for high dimensional data in C++ and R. J. Stat. Softw. **77**(1), 1–17 (2017)