Heterogeneous Ensemble-Based Infill Criterion for Evolutionary Multiobjective Optimization of Expensive Problems

Dan Guo, Yaochu Jin, *Fellow, IEEE*, Jinliang Ding, *Senior Member, IEEE*, and Tianyou Chai, *Fellow, IEEE*

***Abstract*—Gaussian processes (GPs) are the most popular model used in surrogate-assisted evolutionary optimization of computationally expensive problems, mainly because GPs are able to measure the uncertainty of the estimated fitness values, based on which certain infill sampling criteria can be used to guide the search and update the surrogate model. However, the computation time for constructing GPs may become excessively long when the number of training samples increases, which makes it inappropriate to use them as surrogates in evolutionary optimization. To address this issue, this paper proposes to use ensembles as surrogates and infill criteria for model management in evolutionary optimization. A heterogeneous ensemble consisting of a least square support vector machine and two radial basis function networks is constructed to enhance the reliability of ensembles for uncertainty estimation. In addition to the original decision variables, a selected subset of the decision variables and a set of transformed variables are used as inputs of the heterogeneous ensemble to further promote the diversity of the ensemble. The proposed heterogeneous ensemble is compared with a GP and a homogeneous ensemble for infill sampling criteria in evolutionary multiobjective optimization. Experimental results demonstrate that the heterogeneous ensemble is competitive in performance compared with GPs and much more scalable in computational complexity to the increase in search dimension.**

***Index Terms*—Feature extraction, feature selection, Gaussian process (GP), heterogeneous ensemble, multiobjective optimization, surrogate-assisted evolutionary algorithm (SAEA).**

Manuscript received July 25, 2017; revised November 21, 2017; accepted January 11, 2018. Date of publication January 31, 2018; date of current version February 14, 2019. This work was supported in part by the National Natural Science Foundation of China under Grant 61525302 and Grant 61590922, in part by the Project of Industry and Information Technology Ministry under Grant 20171122-6, and in part by the Projects of Shenyang under Grant Y17-0-004. This paper was recommended by Associate Editor K.-C. Tan. *(Corresponding authors: Yaochu Jin; Tianyou Chai.)*

D. Guo, J. Ding, and T. Chai are with the State Key Laboratory of Synthetical Automation for Process Industries, Northeastern University, Shenyang 110819, China (e-mail: guodan717@163.com; jlding@mail.neu.edu.cn; tychai@mail.neu.edu.cn).

Y. Jin is with the State Key Laboratory of Synthetical Automation for Process Industries, Northeastern University, Shenyang 110819, China, and also with the Department of Computer Science, University of Surrey, Guildford GU2 7XH, U.K. (e-mail: yaochu.jin@surrey.ac.uk).

This paper has supplementary downloadable multimedia material available at http://ieeexplore.ieee.org provided by the authors. The file contains additional simulation results. The material is 0.414 MB in size.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCYB.2018.2794503

I. Introduction

OVER the past two decades, evolutionary algorithms (EAs) have become very popular to solve various multiobjective optimization problems (MOPs). An unconstrained MOP can be generally formulated as follows:

$$\begin{aligned} \min \quad & \mathbf{f}(\mathbf{x}) = (f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots, f_M(\mathbf{x})) \\ \text{s.t.} \quad & \mathbf{x} \in D \end{aligned} \tag{1}$$

where $D \subseteq R^N$ is the feasible decision space and R^M is the objective space. Since objectives are conflicting to each other, there usually exists a set of optimal tradeoff solutions. The image formed by the tradeoff solutions (often also known as Pareto optimal solutions) in the objective space is called Pareto front (PF), and the corresponding points in D are known as Pareto optimal set.

EAs are well suited for multiobjective optimization as they can deliver multiple optimal solutions in one run for users to choose from. There are three main categories of multiobjective evolutionary algorithms (MOEAs), namely, dominance-based, decomposition-based, and performance indicator-based approaches. The dominance-based approaches include SPEA2 [1] and NSGA-II [2], among many others. MOEA/D [3] and its variants [4], [5] are the representative decomposition-based approaches. IBEA [6] and SMS-EMOA [7] belong to the performance indicator-based approaches. However, EAs typically require a large number of function evaluations (FEs). EAs become impractical to be employed for solving MOPs when FEs are expensive (either computationally intensive or experimentally costly). Expensive optimization problems are commonly seen in the real world. For example, in structural optimization, no explicit mathematical objective functions are available; instead, time-consuming finite element analysis or computational fluid dynamic simulations must be performed [8], [9].

Surrogates or metamodels are usually built based on historical data to replace in part the real expensive FEs [10], so that the computation time for evolutionary optimization of expensive problems can be reduced. Many machine learning models can be utilized to build surrogates, such as radial basis function network (RBFN) [11], support vector machine (SVM) [12], Gaussian process (GP) [13], and polynomial regression [14]. Although a large body of research on surrogate-assisted evolutionary algorithms (SAEAs) has been

reported, many challenges remain to be addressed, in particular how to reduce the computational cost for constructing surrogates in evolutionary optimization of large-scale and/or many-objective optimization problems [15].

In SAEAs, model management or evolution control determines which individuals will be evaluated by the expensive FEs [10]. Generally speaking, model management is categorized into individual-based, generation-based, population-based or a hybrid of these methods [16]. SAEAs that allow additional expensive FEs to be performed during the optimization are known as on-line SAEAs [17].

In addition to the above model management strategies, infill sampling criteria have been developed for GP-assisted evolutionary optimization. Infill criteria, including expected improvement (ExI) [18], probability of improvement [19], and lower confidence bound (LCB) [20], make use of the uncertainty information provided by the GPs. Note that GP-assisted optimization has different terminologies in different areas, e.g., Kriging or design and analysis of computer experiments [21] in geostatistics, or efficient global optimization [18] or Bayesian optimization [22] in global optimization.

GPs have been equally popular in surrogate-assisted single [23], [24] and multiobjective evolutionary optimization [25]–[28]. However, most GP-assisted EAs have been tested only on low-dimensional problems (up to ten decision variables) [15], mainly due to the fact that the computational cost of constructing the GP is $O(N^3)$, where N is the number of training data [29]. By contrast, SAEAs using neural networks and other machine learning models as the surrogates have been tested on problems with up to 30 decision variables. Most recently, a surrogate-assisted cooperative swarm optimization using the RBFN as the surrogate has been validated on single objective optimization problems with 100 decision variables [30].

Generally speaking, no one particular type of surrogate models works well for all problems. Thus, one natural idea is to use an ensemble surrogate consisting of a group of individual models [31], [32]. In machine learning, it has been theoretically proved that when a right balance between diversity and accuracy is met, an ensemble can provide more accurate predictions than any of its member alone [33]. Ensembles are often categorized into homogeneous ensembles (consisting of the same type of models typically generated by manipulating the data, like random sampling [34]) and heterogeneous ensembles (composed of different types of models [35] or different input features [36]).

It should be emphasized that uncertainty information of the predicted fitness plays an important role in model management in SAEAs, as sampling the most uncertain solutions can not only most efficiently enhance the model quality, but also explore the unexplored regions of the search space. This is one of the reasons why GPs are particularly popular in SAEAs as they can provide uncertainty information together with the predicted fitness value. Note, however, that other methods for uncertainty estimation have also been proposed. For example, Branke and Schmidt [37] utilized the distance to the training data in the neighborhood to estimate the uncertainty. In addition, it has been indicated that ensembles are also able to provide a degree of uncertainty using the discrepancies among the outputs of the ensemble members, which has been shown to be effective for model management [31], [38]. Although much work has been reported on using ensembles [39]–[41] or multiple surrogates [13] in SAEAs, most ensemble-based SAEAs use ensembles to improve the accuracy of fitness approximation and little has been done to exploit the uncertainty information provided by ensembles with few exceptions [31], [38]. Thus, use of ensemble as surrogates deserves more attention in SAEAs [42].

The main motivation of this paper is to take advantage of the uncertainty information provided by ensembles so that infill criteria can still be used for model management without using GP as the surrogate. One major benefit of using ensembles instead of GP as the surrogate is that the computational complexity for constructing ensembles (without involving a GP as a member) is relatively scalable to the increase in the number of training samples, thereby, enabling SAEAs to solve high-dimensional MOPs. To this end, we propose a framework for MOEA assisted by a heterogeneous ensemble, termed HeE-MOEA, where the uncertainty of the approximated fitness is estimated by the outputs of all ensemble members, based on which an infill criterion is adopted for model management. Different types of models with different input features are generated as the ensemble members, so that the ensemble members are highly diverse yet sufficiently accurate. Note that any infill sampling criterion developed for GPs and any MOEA can be adopted for the proposed HeE-MOEA framework.

The rest of this paper is organized as follows. Sections II and III provide the background knowledge about the construction of the heterogeneous ensemble, together with a brief account of the GPs so that this paper is self-contained. The main components of HeE-MOEA are then explained in Section IV. Simulation results are presented and discussed in Section V. Section VI concludes this paper.

II. Heterogeneous Ensembles

In this section, we will first introduce feature selection and feature extraction, two ways of manipulating features to generate different inputs for the ensemble, followed by a brief description of two machine learning models used as ensemble members in this paper, least square SVM (LSSVM) and RBFN. Finally, the generation of heterogeneous ensemble is briefly discussed.

A. Feature Selection and Feature Extraction

Feature selection and feature extraction are two popular methods for dimension reduction. While feature selection aims to select the smallest and most informative subset of the existing features, feature extraction transforms the existing features into a lower dimensional space so that the new features are nonredundant and informative.

By removing the irrelevant and redundant features, feature selection can simplify the structure of the model, improve the generalization capacity, and enhance the efficiency in training and responding [43]. Feature selection is nontrivial, as it is essentially a combinatorial optimization problem [44].

According to the way in which a feature selection strategy is combined with a learning machine, feature selection methods can be categorized into wrapper, embedded and filter approaches [45]. Although the resulting learning performance of filter methods is typically worse than that of the other two methods, they usually have the lowest computational cost, highest generality and robustness [46]. In [47], seven types of evaluation functions in filter methods are reviewed, where feature selection methods are also categorized into individual and subset evaluation methods.

Generally speaking, feature extraction can be divided into linear or nonlinear methods [48], [49]. Among linear approaches, principal component analysis (PCA) is probably the most popular, which uses an orthogonal coordinate system to redescribe the original data, and the principal components are selected according to their importance [50]. However, PCA assumes that data are of a Gaussian distribution.

B. Least Square Support Vector Machine

LSSVM [51] is a modified version of standard SVM. LSSVM has good generalization performance and low computational cost [52]. Building an LSSVM model based on data $X = [\mathbf{x}^1, \mathbf{x}^2, \ldots, \mathbf{x}^K]^T$ and $Y = [y^1, y^2, \ldots, y^K]^T$ can be regarded as an optimization problem

$$\begin{aligned} \min_{w,b,e} \quad & J(w,e) = \frac{1}{2}w^T w + \gamma \frac{1}{2}\sum_{k=1}^{K} e_k^2 \\ \text{s.t.} \quad & y^k = w^T \varphi(\mathbf{x}^k) + b + e_k, \ \ k = 1, 2, \ldots, K. \end{aligned} \tag{2}$$

The estimation of LSSVM at a new sampling point $\mathbf{x}^{\text{new}}$ can be given by

$$\hat{f}_L(\mathbf{x}^{\text{new}}) = \sum_{k=1}^{K} \lambda_k \Psi(\mathbf{x}^{\text{new}}, \mathbf{x}^k) + b \tag{3}$$

where λ_k is the Lagrange multiplier, and Ψ is the sigmoid kernel.

C. Radial Basis Function Networks

RBFNs are feedforward neural networks with one hidden layer that uses radial-basis-function as the activation function. In RBFNs, input nodes are directly connected to hidden ones, and the output of the RBFN at $\mathbf{x}^{\text{new}}$ has the following expression:

$$\hat{f}_R(\mathbf{x}^{\text{new}}) = \boldsymbol{\phi} \times \mathbf{w} \tag{4}$$

which is a linear combination of the outputs of J hidden neurons. $\boldsymbol{\phi}(\mathbf{x}^{\text{new}}) = [\phi_1(\mathbf{x}^{\text{new}}), \ldots, \phi_J(\mathbf{x}^{\text{new}})]$ is a vector of basis function values and $\mathbf{w}$ is the weight vector. When Gaussian kernel is adopted as the basis function, we get

$$\phi_j\left(\mathbf{x}^{\text{new}}\right) = \exp\left(-\frac{\left\|\mathbf{x}^{\text{new}} - \boldsymbol{\mu}_j\right\|^2}{2\delta_j^2}\right), j = 1, 2, \ldots, J \tag{5}$$

where $\boldsymbol{\mu}_j$ and δ_j^2 are the center and the variance of the jth hidden neuron, respectively. Suppose that function values of K training points $X = [\mathbf{x}^1, \mathbf{x}^2, \ldots, \mathbf{x}^K]^T$ are $Y = [y^1, y^2, \ldots, y^K]^T$, one way to determine the weights is to use the least square method

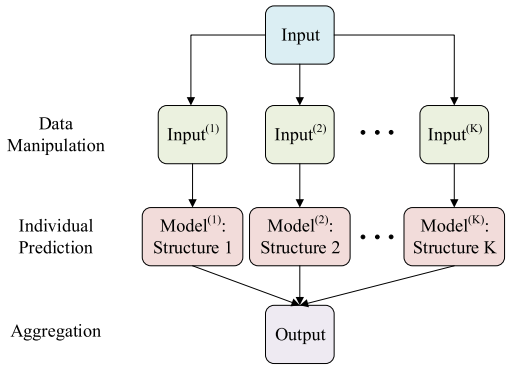

Fig. 1. Generation framework of conventional heterogeneous ensembles.

$$\mathbf{w} = (\Phi^T \Phi)^{-1} \Phi^T Y \tag{6}$$

where Φ is a $K \times J$ matrix, and the (k, j)th element of Φ is $\phi_j(\mathbf{x}^k)$, the jth basis function evaluated at the point $\mathbf{x}^k$. Backpropagation is another way to estimate the weights. If E_k represents the error of the kth training sample

$$E_k = \frac{1}{2}(y^k - \hat{f}_R(\mathbf{x}^k))^2 \tag{7}$$

then the change in weights can be formulated as

$$\Delta \mathbf{w} = -\eta \frac{\partial E_k}{\partial \mathbf{w}} \tag{8}$$

where η is the learning rate. In backpropagation, centers and variances of the hidden units are updated in a similar way as weights. In the above description, the RBFN has one output node, which can be easily extended to multiple outputs.

D. Heterogeneous Ensemble Generation

Constructing diverse and accurate ensemble members is essential and challenging. Various techniques can be used to create diversity in ensembles. These techniques are categorized into explicit and implicit methods depending on whether optimization of a diversity metric is applied in ensemble construction [53]. Most of these techniques belong to the implicit type, such as training data manipulation including bagging [34], [54], feature selection and feature extraction [55], or use of different initial parameters, different training algorithms and different architectures of learners [56], while boosting [57] and negative correlation learning [58] belong to the explicit type. Heterogeneous ensembles have the natural advantages in structural diversity by using different architectures or different learning types of models [42], [55], [59]. An illustrative example of a heterogeneous ensemble is shown in Fig. 1. Heterogeneous ensembles that use both different types of models and different input features created by feature selection and feature extraction have shown better performance than those using single type of inputs [35], [60], partly due to the inherent better ability to create diversity [36]. Therefore, this paper also employs different features and different types of models for promoting ensemble diversity without impairing accuracy,

hoping to enhance the ability of the ensemble surrogate in uncertainty estimation.

III. Infill Criteria for Gaussian Process-Assisted Optimization

A. Gaussian Processes

In GP, it is assumed that the function value $f_G(\mathbf{x}^{\text{new}}) \in R$ is an observation of the following stochastic process:

$$\mu + \epsilon(\mathbf{x}^{\text{new}}) \tag{9}$$

where $\mathbf{x}^{\text{new}} \in R^N$, μ is the mean of the stochastic process and the error term $\epsilon(\mathbf{x}^{\text{new}})$ is a Gaussian distribution of $N(0, \sigma^2)$. It is also assumed that the errors of two points, for example, $\epsilon(\mathbf{x}^i)$ and $\epsilon(\mathbf{x}^j)$, are only related to the distance between them, and the correlation can be expressed as

$$c\big[\epsilon(\mathbf{x}^i), \epsilon\big(\mathbf{x}^j\big)\big] = \exp\big[-d\big(\mathbf{x}^i, \mathbf{x}^j\big)\big]. \tag{10}$$

The weighted distance below is often used as the distance function

$$d\big(\mathbf{x}^i, \mathbf{x}^j\big) = \sum_{n=1}^{N} \theta_n |x_n^i - x_n^j|^{p_n} \tag{11}$$

where $\theta_n > 0$, $1 \le p_n \le 2$ [18]. θ_n quantifies the extent to which the nth dimension of the variables contributes to the correlation, and p_n controls the smoothness of the function in this dimension. If K training points $X = [\mathbf{x}^1, \mathbf{x}^2, \ldots, \mathbf{x}^K]^T$ and their function values $Y = [y^1, y^2, \ldots, y^K]^T$ have been collected, a $K \times K$ correlation matrix C will be formed, and the (i, j)th entry in C is $c(\mathbf{x}^i, \mathbf{x}^j)$. Then the likelihood function can be formulated as follows:

$$\text{lik}(\theta_1, \ldots, \theta_N, p_1, \ldots, p_N) = -\Big(K \ln\Big(\hat{\sigma}^2\Big) + \ln\det(C)\Big) \tag{12}$$

where

$$\hat{\sigma}^2 = \frac{\big(Y - \mathbf{1}\hat{\mu}\big)^T C^{-1} \big(Y - \mathbf{1}\hat{\mu}\big)}{K} \tag{13}$$

$$\hat{\mu} = \frac{\mathbf{1}^T C^{-1} Y}{\mathbf{1}^T C^{-1} \mathbf{1}} \tag{14}$$

$\mathbf{1}$ represents a $K \times 1$ vector of ones. By maximizing the likelihood function, the suitable parameters $\theta_1, \ldots, \theta_N, p_1, \ldots, p_N$, $\hat{\sigma}^2$ and $\hat{\mu}$ will be obtained. Let R represent the $K \times 1$ correlation vector for $\mathbf{x}^{\text{new}}$ and each element in X, and then the best linear unbiased predictor of $f(\mathbf{x}^{\text{new}})$ and its variance estimate will be expressed as

$$\hat{f}_G\big(\mathbf{x}^{\text{new}}\big) = \hat{\mu} + R^T C^{-1}\big(Y - \mathbf{1}\hat{\mu}\big) \tag{15}$$

$$\hat{s}_G^2(\mathbf{x}^{\text{new}}) = \hat{\sigma}^2 \left[1 - R^T C^{-1} R + \frac{(1 - \mathbf{1}^T C^{-1} R)^2}{\mathbf{1}^T C^{-1} \mathbf{1}}\right]. \tag{16}$$

B. Infill Sampling Criteria

Infill criteria make use of the estimates of fitness and confidence to assess the value of a solution with respect to the optimality and uncertainty. If a solution is expected to be promising according to an infill criterion, it will be selected to be evaluated using the real expensive fitness function. Although infill criteria are developed and mostly used for GPs, they are in fact applicable to any surrogate models that are able to provide uncertainty information of the predicted fitness. In this paper, we consider two infill criteria, LCB [20] and ExI [18], which have been shown to be most robust [28]. Suppose that the predicted mean and standard deviation at an unknown point $\mathbf{x}$ are $\hat{f}(\mathbf{x})$ and $\hat{s}(\mathbf{x})$, then the merit functions of LCB and ExI are as follows:

$$\text{LCB}(\mathbf{x}) = \hat{f}(\mathbf{x}) - w\hat{s}(\mathbf{x}) \tag{17}$$

where w is positive for minimization problems

$$\text{ExI}(\mathbf{x}) = (y_{\min} - \hat{f}(\mathbf{x}))\Phi \times \left(\frac{y_{\min} - \hat{f}(\mathbf{x}))}{\hat{s}(\mathbf{x})}\right) + \hat{s}(\mathbf{x})\varphi\left(\frac{y_{\min} - \hat{f}(\mathbf{x})}{\hat{s}(\mathbf{x})}\right) \tag{18}$$

where Φ and φ are standard normal distribution and probability density function, respectively, and $y_{\min}$ is the minimum of objective values of the existing real data. In (18), ExI increases as $\hat{f}_H$ decreases and $\hat{s}_H$ increases [18], so the maximum ExI is pursued. On some level, ExI and LCB both can balance the exploitation and exploration in the optimization.

C. Discussion

The computational cost for building a GP model can become prohibitively high as the number of training data increases. When the maximal likelihood function is searched, the inverse of K-dimensional square matrix C will be calculated for every set of trial hyperparameters to find the most suitable ones. Furthermore, the search process is needed in each iteration with the update of training data. The situation will become worse when the dimension of decision variables is large, because more training samples are required for higher dimensional problems. In the search for the optimal hyperparameters, some algorithms adopt a traditional mathematical optimization method to reduce the computational complexity [15], [25], whereas others use an evolutionary algorithm [7], [27] because the likelihood function is multimodal. In this research, we apply the DACE toolbox [21] to construct GPs, where the Hookes and Jeeves algorithm is employed for optimizing the hyperparameters of GPs.

IV. Heterogeneous Ensemble-Assisted MOEAs

A. Main Loop

The proposed HeE-MOEA distinguishes itself from the other surrogate-assisted MOEAs mainly in that HeE-MOEA uses a heterogeneous ensemble instead of the GP for fitness approximation. Besides, the differences among the ensemble members in HeE-MOEA are used to estimate the confidence intervals, which is important for an infill criterion to locate the solutions for expensive FEs. The ensemble surrogate used in HeE-MOEA consists of different types of models with different input features, so the accuracy of the ensemble is enhanced and the diversity of the ensemble members are promoted.

In HeE-MOEA, there is a cycle between the update of heterogeneous ensemble and the selection of solutions for expensive FEs, and the pseudocode of the main framework of HeE-MOEA is presented in Algorithm 1. Initially, $11N - 1$

Algorithm 1 HeE-MOEA

Input: Dimension of parameter space N; Expensive real functions $\mathbf{f}$; Maximum iterations $iter_1$; Upper limit of training data l;
Output: Solutions X and Y;
1: X are generated by Latin hypercube sampling;
2: **for** $i = 1$ to $11N - 1$ **do**
3: $Y[i] \leftarrow$ evaluate $X[i]$ by $\mathbf{f}$;
4: **end for**
5: $id \leftarrow$ Filter-feature-selection(X, Y);
6: **for** $i = 1$ to $iter_1$ **do**
7: Use PCA on X to obtain principle component coefficients r_p;
8: **if** the number of training data is less than l **then**
9: $X^{tr} \leftarrow X$, $Y^{tr} \leftarrow Y$;
10: **else**
11: Select X^{tr}, Y^{tr} from X and Y;
12: **end if**
13: $model \leftarrow$ Heterogeneous-ensemble$(X^{tr}, Y^{tr}, id, r_p)$;
14: $solutions \leftarrow$ MOEA$(model, id, r_p)$;
15: X_{new} are selected from *solutions* by k-means;
16: $X \leftarrow X \cup X_{new}$;
17: $Y_{new} \leftarrow$ evaluate X_{new} by $\mathbf{f}$;
18: $Y \leftarrow Y \cup Y_{new}$;
19: **end for**

Algorithm 2 Filter-Feature-Selection

Input: Input X and output Y of data; Input dimension N; The number of particles m; Maximum iterations $iter_2$; Threshold T;
Output: Indices of the selected features *id*;
1: Randomly initialize the position array $P_{m\times N}$ and velocity array $V_{m\times N}$ to describe all particles, and initialize an empty array *id*;
2: **while** not reach $iter_2$ **do**
3: **for** each position p in P **do**
4: $p = \begin{cases} 0, & p < T \\ 1, & p \geq T \end{cases}$
5: **end for**
6: **for** $i = 1$ to m **do**
7: **for** $j = 1$ to N **do**
8: Select the j-th dimension vector of X and put it in X_s^i when $P(i, j) = 1$;
9: **end for**
10: Calculate the fitness value of X_s^i by Equation (22) and (23), and the value is set to infinity when X_s^i is empty;
11: **end for**
12: Update personal best position *pbest* of each particle and global best position *gbest*;
13: Update V and P;
14: **end while**
15: **for** $i = 1$ to m **do**
16: **if** $gbest(i) = 1$ **then**
17: $id \leftarrow id \cup i$;
18: **end if**
19: **end for**

training data points are generated using Latin hypercube sampling. A filter method based on particle swarm optimization (PSO) [61] is used to select input features for the generation of different inputs. The selection function is defined by the redundancy of the selected features and their relevance with the function values, and Algorithm 2 details the procedure. In addition to feature selection, PCA is used to extract the principle components from input features. As a result, each member in the heterogeneous ensemble has three different sets of training data: 1) the original data; 2) the data with selected features only; and 3) the data with extracted features only. With these different data sets, different ensemble members are built. In the optimization, the ensemble surrogates replace the real objective functions to evaluate candidate solutions found by the MOEA, and LCB or ExI is used to calculate the merit values of these solutions. The goal of the optimization is to minimize (17) or to maximize (18). In the phase of selecting solutions for real expensive FEs, solutions close to the evaluated points will be first deleted, and then k-means clustering is used to pick k_m solutions. When the Euclidean distance of two points in decision space is smaller than 10^{-5}, the two points are considered to be close. After that, these new k_m data will be used to update the ensemble surrogates. This process repeats until a stopping criterion is satisfied.

HeE-MOEA takes two measures to reduce the computational complexity. One is the restriction of the number of training data. When the number of points in X exceeds an upper bound l, a selection strategy similar to [25] will be activated: the first $\lfloor l/2 \rfloor$ training data are the optimal ones chosen by nondominated sorting and crowding distance [2], and the other half are randomly sampled without replacement. The second measure is that feature selection is performed off-line only once based on the initial training data and the selected features will be used for the whole optimization process. However, feature extraction by PCA is performed in each iteration as PCA is computationally very cheap.

It should be noted that any machine learning model can replace the heterogeneous ensemble (lines 5, 7, and 13 of Algorithm 1), and any MOEA and any infill sampling criterion can be applied in selecting samples for evaluation (line 14 of Algorithm 1). In the following subsection, we will describe in more detail the filter-based feature selection method and the construction of the heterogeneous ensemble.

B. Filter-Based Feature Selection

In this filter method, the selection function is defined to minimize the redundancy and maximize the relevance to the outputs (the function values) [47]. Any measure of dependence between random vectors is able to describe the degree of redundancy in the selected features $X_s = [\mathbf{x}_s^1, \mathbf{x}_s^2, \ldots, \mathbf{x}_s^K]^T$ ($\mathbf{x}_s^k \in R^{N_1}, N_1 < N$) and the relevance of X_s to the function values $Y = [\mathbf{y}^1, \mathbf{y}^2, \ldots, \mathbf{y}^K]^T$ ($\mathbf{y}^k \in R^M$). In this paper, distance correlation suggested in [62] is adopted. Distance correlation is

based on distance covariance, which is analogous to product-moment correlation and covariance. X_s and Y can be seen as K samples of random variables $(\mathbf{x}_s, \mathbf{y})$, and then two $K \times K$ distance matrices A and B will be obtained by

$$\begin{aligned} a_{i,j} &= \|\mathbf{x}_s^i - \mathbf{x}_s^j\|^2, \ b_{i,j} = \|\mathbf{y}^i - \mathbf{y}^j\|^2 \\ A_{i,j} &= a_{i,j} - \bar{a}_{i\cdot} - \bar{a}_{\cdot j} - \bar{a}_{\cdot\cdot} \\ B_{i,j} &= b_{i,j} - \bar{b}_{i\cdot} - \bar{b}_{\cdot j} - \bar{b}_{\cdot\cdot} \\ & i, j = 1, 2, \ldots, K \end{aligned} \tag{19}$$

where $\bar{a}_{i\cdot}$ and $\bar{a}_{\cdot j}$ are the mean of ith row and jth column, respectively, and $\bar{a}_{\cdot\cdot}$ is the grand mean of matrix a. Distance covariance is actually an average of the dot product of A and B

$$\mathrm{dCov}^2(X_s, Y) = \frac{1}{K^2} \sum_{i=1}^{K} \sum_{j=1}^{K} A_{i,j} B_{i,j}. \tag{20}$$

Finally, the distance correlation of X_s and Y can be expressed by

$$\mathrm{dCor}(X_s, Y) = \frac{\mathrm{dCov}(X_s, Y)}{\sqrt{\mathrm{dCov}(X_s, X_s)\mathrm{dCov}(Y, Y)}} \tag{21}$$

where $\mathrm{dCov}(X_s, X_s)$ and $\mathrm{dCov}(Y, Y)$ have a similar expressions to (20).

The selected features should have the maximal relevance to the expensive functions and minimal redundancy to each other, so the fitness in feature selection can be defined as an aggregation function [63]

$$\min \ \mathit{Fitness} = \alpha * R_1 - (1 - \alpha) * R_2 \tag{22}$$

where R_1 and R_2 represent the redundancy in X_s and the correlation between X_s and Y, respectively. α is a constant ranging from 0 to 1, and a larger α indicates more emphasis on the relevance. Every feature of the original input set and every regression function are regarded as discrete random variables. By distance correlation, the formulations of R_1 and R_2 are

$$\begin{aligned} R_1 &= \frac{1}{N_1} \sum_{i=1}^{N_1} \mathrm{dCor}(x_i, \complement_{X_s} x_i) \\ R_2 &= \mathrm{dCor}(X_s, Y) \end{aligned} \tag{23}$$

where N_1 is the number of features in X_s, and x_i is the ith feature of X_s. R_1 calculates the mean of distance correlations between every feature in X_s and others to indicate the redundancy contained in the selected features, while R_2 uses the distance correlation of X_s and Y to denote the dependency of all selected features on all functions.

The entire feature selection process is described in Algorithm 2. The filter method makes use of a standard PSO [61] to find the best combination of features. Each particle represents a combination, and a threshold $T = 0.5$ is introduced to make every position value in particles be 0 or 1. As it is impossible that all features are redundant, infinity is assigned to a fitness value when no features are chosen. The index of every selected feature is kept in *id* and output, and then it will be applied in ensemble building and predicting.

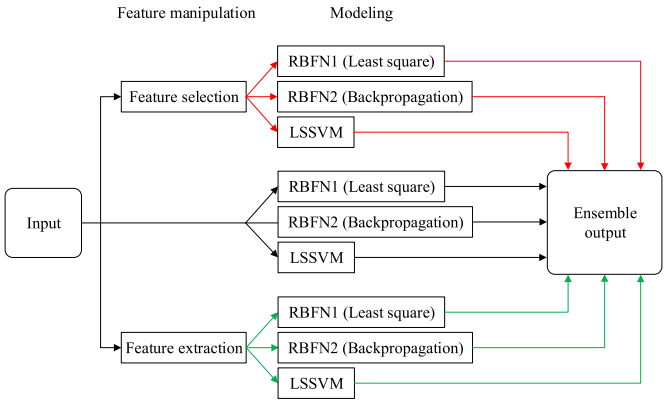

Fig. 2. Structure of the heterogeneous ensemble in HeE-MOEA.

C. Ensemble Construction

Indices *id* of the selected features and principle component coefficients r_p are obtained after feature selection and extraction. When *id* and r_p are applied to the training data (X^{tr}, Y^{tr}), there will be another two sets of data, i.e., (X_s^{tr}, Y^{tr}) and (X_e^{tr}, Y^{tr}). Three modeling methods are applied in HeE-MOEA, LSSVM, RBFN1, and RBFN2, where RBFN1 refers to the RBFN whose weights are obtained by the least square method (6), while RBFN2 refers to one using backpropagation to train weights (8). The resulting structure of the heterogeneous ensemble is shown in Fig. 2.

The heterogeneous ensemble is composed of nine members. Each ensemble member is assigned a weight w_i when estimating the objective values, and the sum of these weights is one. The estimated function value $\hat{f}_H$ and variance $\hat{s}_H^2$ of a new candidate solution $\mathbf{x}^{\mathrm{new}}$ is expressed as

$$\begin{aligned} \hat{f}_H(\mathbf{x}^{\mathrm{new}}) &= \sum_{i=1}^{N_2} w_i \hat{f}_{Hi}, \ \sum_{i=1}^{N_2} w_i = 1 \\ \hat{s}_H^2(\mathbf{x}^{\mathrm{new}}) &= \frac{1}{N_2 - 1} \sum_{i=1}^{N_2} (\hat{f}_{Hi} - \hat{f}_H)^2, \ N_2 = 9 \end{aligned} \tag{24}$$

where $\hat{f}_{Hi}$ represents the output of the ith member in the ensemble. Note that $\mathbf{x}^{\mathrm{new}}$ will first be transformed into $\mathbf{x}_s^{\mathrm{new}}$ or $\mathbf{x}_e^{\mathrm{new}}$ by *id* or r_p when the objective value is estimated by the learners that are trained on a set of reduced features. Given the estimated fitness and the variance, (17) or (18) can be used to assess the merit value of the candidate solution.

V. Numerical Experiments

In this section, the performance of HeE-MOEA is examined on two widely used test suites: 1) DTLZ [64] and 2) WFG [65]. For each test instance, 10, 20, 40, and 80 decision variables are considered while the number of objectives remains three. The heterogeneous ensemble is compared with a homogeneous ensemble and a GP using a same infill criterion and a same MOEA, which is termed HoE-MOEA and GP-MOEA, respectively. The homogeneous ensemble in HoE-MOEA is generated using bagging, and each member is an RBFN whose weights are trained by the least square method.

The homogeneous ensemble also consists of nine members. In addition, we compare HeE-MOEA with an MOEA without using surrogates to demonstrate the importance of surrogates in optimization of expensive problems. Note that any MOEA can be used in the HeE-MOEA framework to solve problems with different properties, and we take NSGA-II [2] and MOEA/D [3] as examples in this paper.

A. Performance Metrics

As both hypervolume (HV) [66] and inverted generational distance (IGD) [67] can capture diversity and convergence properties of a set of nondominated solutions, they are adopted as the performance indicators for comparisons in this paper. HV in a 2-D objective space can be interpreted as the area enclosed by a reference point and the nondominated solutions, and the higher HV values, the better the solutions. In this paper, the maximum and minimum objective values of nondominated solutions found by all compared algorithms are used to specify the reference point of HV, i.e., $\max + \delta(\max - \min), \delta = 0.01$ [25]. IGD calculates the Euclidean distance between a set of reference points sampled, typically sampled from the theoretical PF, and the nondominated solution set achieved by an algorithm. The smaller the IGD value, the better the solution set is. As the number of objectives in all benchmark tests are three, in this paper, 1000 uniformly distributed reference points along the PF are sampled for IGD calculation. We use the PlatEMO toolbox [68] to calculate HV and IGD.

The Wilcoxon rank sum is employed to conduct a significance test in comparing HeE-MOEA with one of the three compared algorithms at a significance level of 5%, where symbol "+" indicates that HeE-MOEA is statistically better, while "−" means that the compared algorithm performs statistically better, and "=" indicates that there is no significant difference between the results obtained by the two algorithms.

B. Test Problems

Two sets of three-objective test problems are used in the experimental studies, namely, DTLZ1 to DTLZ7 and WFG1 to WFG9. In the DTLZ test suite, the number of position parameters must be one less than the number of objectives, and the number of distance parameters is scalable. All decision variables of DTLZ are within [0, 1]. DTLZ1 and DTLZ3 have many local PFs, causing difficulties for MOEAs to converge to the global PF. In DTLZ4, the mapping from the Pareto set to the PF is biased, as a result, it is much challenging to achieve evenly distributed solutions in the objective space. As suggested in [64], the parameter α of DTLZ4 is set to 100. The PFs of DTLZ5 and DTLZ6 are degenerate, and DTLZ7 has disconnected Pareto optimal regions. The WFG test suite is more flexible than DTLZ, allowing many characteristics to be combined and introduced into both the objective space (e.g., linear, convex, disconnected, or mixed shape functions) and the decision space (e.g., bias, deceptive, multimodal, or nonseparable transformation functions). In WFG, the number of position parameters k must be divisible by $M-1$, where M is the number of objectives, and the number of distance parameters must be a multiple of two in WFG2 and WFG3. In our experiments associated with NSGA-II, the values of k are set to $6, 10, 10, 40$ for the dimension of search space $N = 10, 20, 40, 80$, respectively, and $k = 2$ for the experiments associated with MOEA/D. The ranges of decision variables in the WFG test problems are of different magnitudes, i.e., $[0, 2n]$, $n = 1, 2, \ldots, N$.

C. Parameter Settings

Suppose M and N represent the objective number and the number of decision variables in all test instances, respectively. We have the following settings.

1) The number of independent runs is 20 for test problems with $N = 10, 20, 40$, while it is set to 10 when $N = 80$. In each run, the initial training data are newly generated, and they are the same for all compared algorithms.
2) In Algorithm 1, the upper limit of the training data l takes the value of $11N-1+25$, and the number of FEs at each iteration $k_m = 5$.
3) In Algorithm 1, when NSGA-II is taken as the base MOEA, the population size and the generation number of NSGA-II are both 50, and the maximum iterations $\text{iter}_1 = 24$, so the maximal FEs in a run are $11N - 1+120$. For NSGA-II without surrogates, the maximum number of FEs in a run is set to $11N + 120$: the population size is $46, 34, 40, 40$ when $N = 10, 20, 40, 80$, respectively.
4) In Algorithm 1, when MOEA/D is adopted as the base MOEA, the population size of MOEA/D is 45 ($H = 8, T = 5$) and the generation number is 55, and the maximum number of expensive FEs in a run is set to 359 when $N = 20$. For MOEA/D without surrogates, the population size is also set to 45 and the maximum number of FEs in a run is 360 for $N = 20$.
5) There are two RBFNs in HeE-MOEA, i.e., RBFN1 and RBFN2. RBFN1 uses the least square method to determine its weights, and its number of hidden neurons is set by $\lceil\sqrt{M+N}+3\rceil$. Each RBFN1 in HoE-MOEA adopts the same settings. RBFN2 uses the backpropagation algorithm (a gradient method) to train the weights, and the mean square error (MSE) is set according to $c\sqrt{\sum_{i=1}^{M}(y_{i\max}-y_{i\min})^2}$, where $y_{i\max}$ and $y_{i\min}$ are the maximum and the minimum of the training data in the ith objective, and the constant $c = 0.05, 0.5, 5, 50$ are used for $N = 10, 20, 40, 80$, respectively.
6) In PSO for feature selection, the number of particles m and the number of iterations iter_2 of Algorithm 2 are 20 and 30, respectively. The constant α in (22) is set to 0.8 as recommended in [63]. The accumulative contribution of principal components is required to be no less than 95% in PCA.
7) Equal weights are assigned to ensemble members, and the parameter w in (17) is set to 2 as recommended in [19].

In our experiments, LSSVM is implemented by LIBSVM [69] and we use the *newrb* function in MATLAB to create RBFN2. All simulations are implemented using

TABLE I
MEAN (FIRST LINE) AND STANDARD DEVIATION VALUES (SECOND LINE) OF HV OBTAINED BY HEE-MOEA AND GP-MOEA USING EXI AND LCB ON 40-D DTLZ AND WFG TEST INSTANCES

Problem	ExI		LCB	
	HeE-MOEA	GP-MOEA	HeE-MOEA	GP-MOEA
DTLZ1	**0.9468** **2.3e-03**	0.9459 = 1.9e-03	**0.9319** **2.4e-03**	0.9311 = 3.3e-03
DTLZ2	**0.9637** **4.7e-03**	0.9436 + 8.0e-03	**0.9571** **5.9e-03**	0.9386 + 5.2e-03
DTLZ3	0.8864 7.3e-03	**0.8865** = **8.8e-03**	**0.8797** **5.9e-03**	0.8706 + 6.9e-03
DTLZ4	**0.8779** **2.6e-02**	0.8530 + 2.9e-02	**0.8729** **3.0e-02**	0.8391 + 3.3e-02
DTLZ5	**0.9305** **4.0e-03**	0.8852 + 9.1e-03	**0.9258** **4.9e-03**	0.8905 + 6.9e-03
DTLZ6	**0.5230** **6.9e-03**	0.5204 = 5.1e-03	**0.5174** **3.4e-03**	0.5147 + 2.0e-03
DTLZ7	0.4784 2.1e-02	**0.4949** = **2.6e-02**	0.4512 1.6e-02	**0.4765** − **1.8e-02**
WFG1	0.2008 3.0e-02	**0.2039** = **3.0e-02**	**0.1776** **2.4e-02**	0.1356 + 4.1e-02
WFG2	**0.5807** **1.5e-02**	0.4887 + 1.3e-02	**0.5319** **2.1e-02**	0.4800 + 1.1e-02
WFG3	**0.4166** **5.6e-03**	0.3934 + 5.0e-03	**0.3870** **9.4e-03**	0.3649 + 4.5e-03
WFG4	0.2499 1.0e-02	**0.2611** − **7.7e-03**	**0.2385** **1.1e-02**	0.2266 + 8.2e-03
WFG5	0.2468 7.0e-03	**0.2747** − **5.9e-03**	0.2493 4.2e-03	**0.2609** − **1.5e-02**
WFG6	0.2691 7.8e-03	**0.2864** − **8.3e-03**	**0.2568** **7.2e-03**	0.2493 + 5.4e-03
WFG7	**0.3168** **8.2e-03**	0.3051 + 4.5e-03	**0.3133** **7.3e-03**	0.2927 + 6.1e-03
WFG8	0.3024 8.9e-03	**0.3151** − **8.0e-03**	**0.3056** **7.2e-03**	0.2908 + 5.1e-03
WFG9	**0.2943** **1.7e-02**	0.2663 + 5.7e-03	**0.2916** **1.1e-02**	0.2753 + 7.2e-03
win/lose/tie	7/4/5		13/2/1	

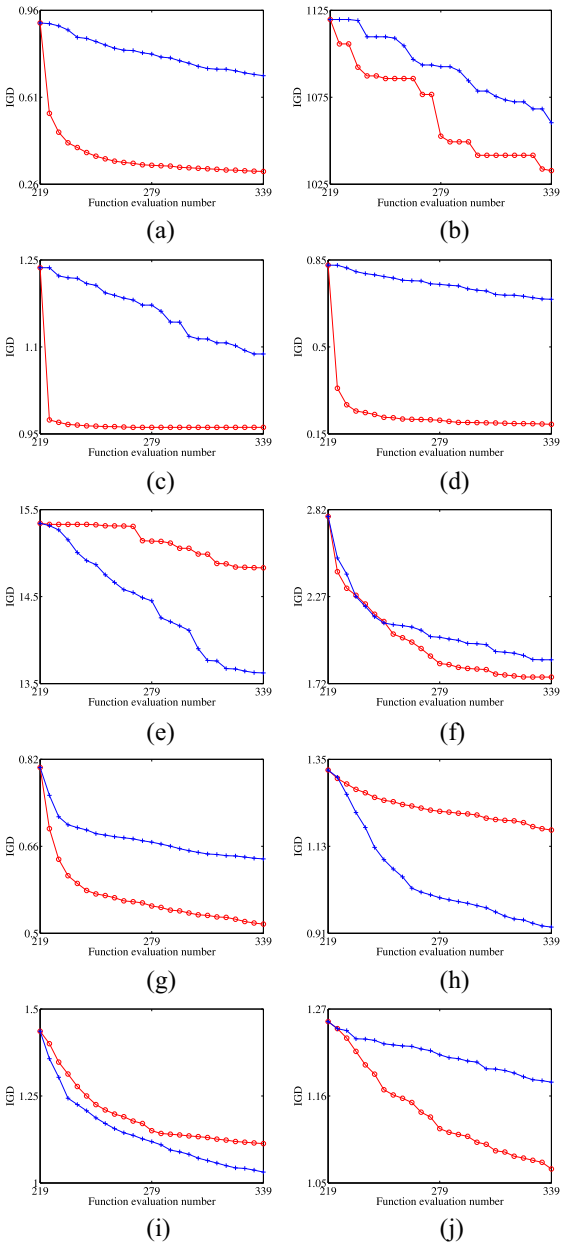

Fig. 3. Mean IGD values versus the number of FEs when number of decision variables is 20. The red lines with "o" are for HeE-MOEA-ExI, and the blue lines with + are for GP-MOEA-ExI. (a) DTLZ2. (b) DTLZ3. (c) DTLZ4. (d) DTLZ5. (e) DTLZ6. (f) WFG1. (g) WFG3. (h) WFG5. (i) WFG7. (j) WFG9.

MATLAB R2014a on an Intel Core i7 with 3.4 GHz CPU, running on the Microsoft Windows 7 Enterprize SP1 64-bit operating system.

D. Comparison With GP-MOEA

In this section, NSGA-II is taken as the base MOEA for comparison. Table I presents the statistical results of HV obtained by HeE-MOEA and GP-MOEA on 40-D test problems using two infill sampling criteria: 1) ExI and 2) LCB, and Table II summarizes the statistical results in terms of IGD values obtained by the two algorithms using LCB. In each table, the better results are highlighted and all results are summarized as "win/lose/tie" at the end of each table. Generally speaking, the advantage of HeE-MOEA becomes clear as the search dimension increases, although HeE-MOEA has almost equal performance with the compared algorithm on 80-D instances. The statistical results of HV and IGD on 40-D problems are almost the same when the LCB criterion is used. It can be seen from the two tables that HeE-MOEA significantly outperforms GP-MOEA on DTLZ2, DTLZ5, WFG1, WFG2, WFG3, and WFG9, however, it loses on DTLZ7 and WFG5. On two multimodal problems, DTLZ1 and DTLZ3, HeE-MOEA and GP-MOEA perform similarly, except that HeE-MOEA performs better on 10-D instances of DTLZ3. The results on DTLZ6 indicate that there is no difference between the two algorithms. The results vary greatly on DTLZ4 and WFG7, so it is hard to conclude which surrogate performs better on the two problems: although GP-MOEA is able to compete with HeE-MOEA on 10-D instances, it performs worse on 40-D

TABLE II
MEAN (STANDARD DEVIATION) OF IGD OBTAINED BY HEE-MOEA AND GP-MOEA USING LCB INFILL CRITERION

Algorithm	N	DTLZ1	DTLZ2	DTLZ3	DTLZ4	DTLZ5	DTLZ6
HeE-MOEA	10	**109.60** (2.1e+01)	**0.2017** (2.7e-02)	**252.84** (5.5e+01)	0.7400 (7.0e-02)	**0.1138** (3.1e-02)	5.8597 (5.5e-01)
	20	**351.20** (4.5e+01)	**0.3623** (2.8e-02)	**1047.5** (1.3e+02)	**0.9595** (2.5e-02)	**0.2153** (3.2e-02)	14.708 (8.6e-01)
	40	**883.29** (5.6e+01)	**0.7843** (7.5e-02)	**2779.2** (1.7e+02)	**1.2197** (5.2e-02)	**0.6444** (7.2e-02)	**32.304** (5.3e-01)
	80	**2006.7** (7.7e+01)	2.0726 (2.6e-01)	**6260.5** (3.8e+02)	**2.5253** (2.4e-01)	1.8144 (3.7e-01)	**67.409** (1.0327)
GP-MOEA	10	118.27 (2.3e+01) =	0.2612 (1.7e-02) +	324.32 (5.8e+01) +	**0.6840** (9.9e-02) −	0.1424 (3.2e-02) +	**5.7859** (3.7e-01) =
	20	351.88 (3.3e+01) =	0.5415 (4.5e-02) +	1081.3 (1.3e+02) =	0.9864 (1.5e-01) =	0.4367 (5.4e-02) +	**14.532** (6.3e-01) =
	40	886.66 (5.1e+01) =	0.9798 (1.1e-01) +	2822.2 (1.6e+02) =	1.8094 (1.7e-01) +	0.8999 (1.1e-01) +	32.554 (5.5e-01) =
	80	2008.1 (8.2e+01) =	**1.9835** (2.7e-01) =	6320.8 (2.6e+02) =	2.7071 (2.2e-01) =	**1.7611** (1.9e-01) =	68.099 (5.1e-01) =

Algorithm	N	DTLZ7	WFG1	WFG2	WFG3	WFG4	WFG5
HeE-MOEA	10	2.0033 (7.4e-01)	**1.9043** (1.8e-01)	**0.6286** (7.5e-02)	0.3576 (8.2e-02)	0.9058 (1.5e-01)	0.7911 (7.9e-02)
	20	4.3576 (8.2e-01)	**1.8883** (1.2e-01)	**0.6801** (5.3e-02)	**0.4477** (3.9e-02)	1.2296 (1.8e-01)	1.1416 (6.9e-02)
	40	6.4813 (5.2e-01)	**1.8677** (1.6e-01)	**0.7947** (5.3e-02)	**0.6118** (2.6e-02)	**1.2694** (1.4e-01)	1.2119 (3.3e-02)
	80	8.3088 (7.7e-01)	**2.3588** (1.4e-01)	**1.0860** (1.1e-01)	**0.6674** (2.5e-02)	**2.1010** (7.9e-02)	1.8386 (3.3e-02)
GP-MOEA	10	**1.0787** (3.8e-01) −	2.1390 (2.0e-01) +	0.7512 (8.3e-02) +	**0.3246** (5.7e-02) =	**0.8213** (9.6e-02) −	**0.6603** (3.8e-02) −
	20	**3.4536** (6.5e-01) −	2.1305 (2.5e-01) +	0.9076 (8.0e-02) +	0.5869 (2.5e-02) +	**1.1804** (1.2e-01) =	**0.9564** (9.8e-02) −
	40	**5.9949** (6.3e-01) −	2.2466 (2.9e-01) +	0.9013 (3.0e-02) +	0.6634 (1.4e-02) +	1.2950 (1.3e-01) =	**1.0657** (1.0e-01) −
	80	**7.6487** (3.2e-01) −	2.5560 (1.2e-01) +	1.2244 (1.2e-01) +	0.8413 (1.9e-02) +	2.1548 (6.6e-02) =	**1.7100** (7.2e-02) =

Algorithm	N	WFG6	WFG7	WFG8	WFG9	win/lose/tie
HeE-MOEA	10	0.8578 (8.1e-02)	0.9152 (1.1e-01)	**0.9852** (7.6e-02)	**0.9486** (6.5e-02)	26/12/26
	20	1.0869 (5.5e-02)	**1.0111** (4.6e-02)	**1.1506** (6.4e-02)	**1.0607** (8.3e-02)	
	40	**1.1691** (3.2e-02)	**0.9590** (5.5e-02)	**1.0791** (3.2e-02)	**1.1702** (4.0e-02)	
	80	**1.5723** (2.5e-02)	**1.6098** (7.1e-04)	**1.6459** (3.9e-02)	**1.5153** (8.0e-02)	
GP-MOEA	10	**0.8004** (3.9e-02) −	**0.8028** (4.0e-02) −	1.0226 (5.3e-02) +	0.9507 (5.1e-02) =	
	20	**1.0507** (5.0e-02) −	1.0690 (3.9e-02) +	1.1796 (5.1e-02) =	1.1844 (5.8e-02) +	
	40	1.1733 (4.7e-02) =	1.0256 (5.3e-02) +	1.1360 (3.2e-02) +	1.2327 (4.4e-02) +	
	80	1.6172 (2.7e-02) =	1.7070 (8.6e-02) =	1.7803 (4.9e-02) =	1.5936 (5.4e-02) +	

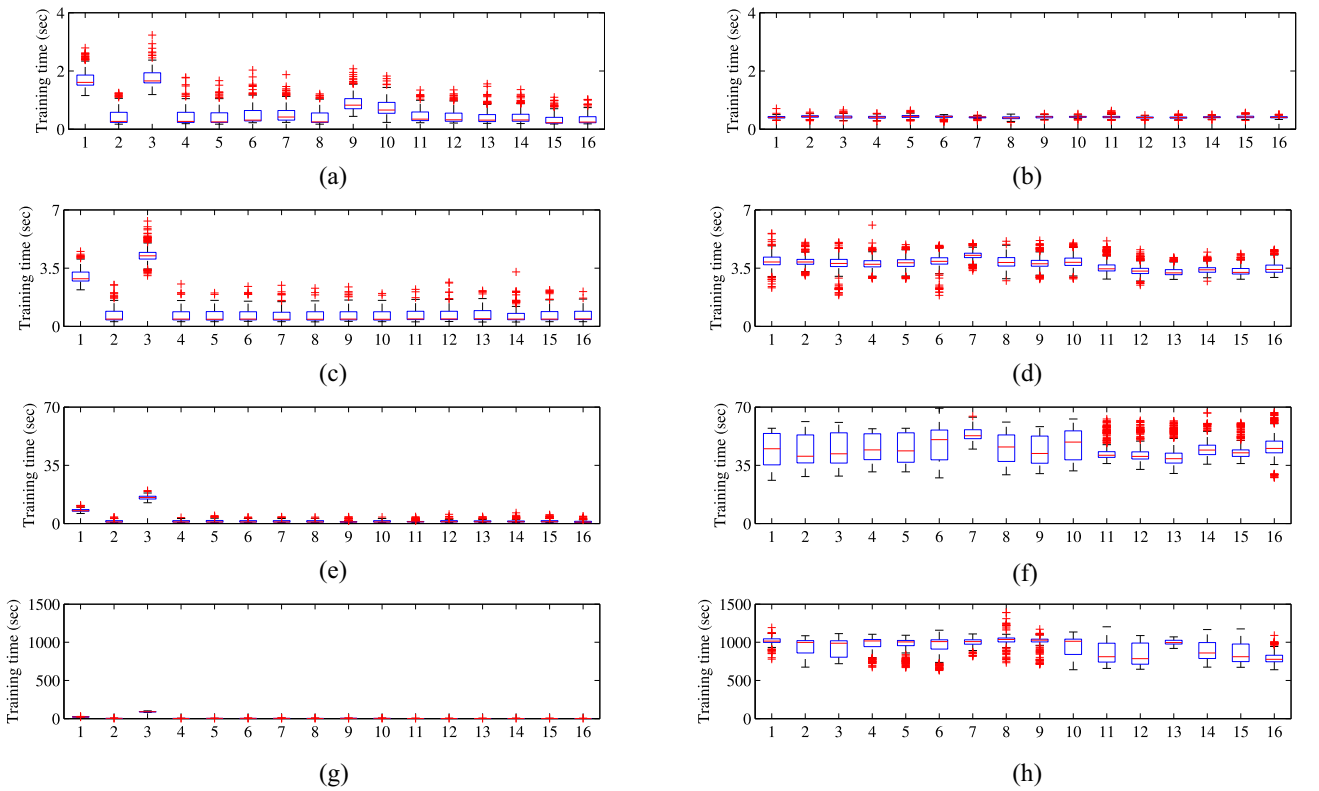

Fig. 4. Boxplot of training time of the surrogates in HeE-MOEA and GP-MOEA using LCB infill criterion. Number 1 to 16 represent the benchmark problems used in the comparisons: DTLZ1 to DTLZ7 and WFG1 to WFG9. (a) HeE-MOEA ($N = 10$). (b) GP-MOEA ($N = 10$). (c) HeE-MOEA ($N = 20$). (d) GP-MOEA ($N = 20$). (e) HeE-MOEA ($N = 40$). (f) GP-MOEA ($N = 40$). (g) HeE-MOEA ($N = 80$). (h) GP-MOEA ($N = 80$).

instances. It should be noted that the statistical results of HV in terms of ExI and LCB on some 40-D test problems are conflicting with each other, e.g., WFG4, WFG6, and WFG8. This implies that the reference point for calculating the HV might not have been properly set. On WFG4 and WFG6, GP-MOEA achieves better IGD values on low-dimensional instances while

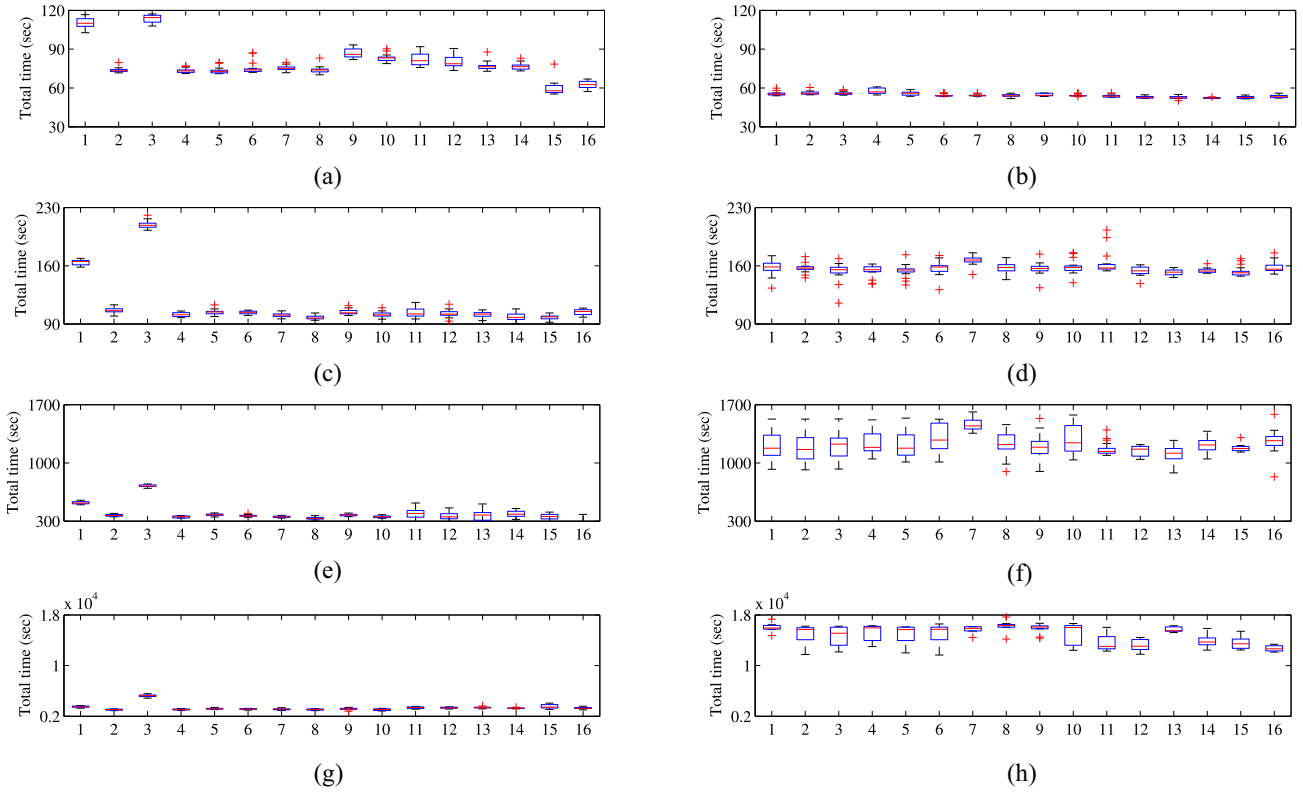

Fig. 5. Boxplot of the total time for a run in HeE-MOEA and GP-MOEA using LCB infill criterion. Number 1 to 16 represent the benchmark problems used in the comparisons: DTLZ1 to DTLZ7 and WFG1 to WFG9. (a) HeE-MOEA ($N = 10$). (b) GP-MOEA ($N = 10$). (c) HeE-MOEA ($N = 20$). (d) GP-MOEA ($N = 20$).(e) HeE-MOEA ($N = 40$). (f) GP-MOEA ($N = 40$). (g) HeE-MOEA ($N = 80$). (h) GP-MOEA ($N = 80$).

the two algorithms behave almost the same on higher dimensional problems. On WFG8, HeE-MOEA exhibits absolute advantages over GP-MOEA on 10-D and 40-D instances. Overall, we can conclude that the performance of HeE-MOEA is comparable to or better than GP-MOEA.

Fig. 3 shows the change of the mean IGD of HeE-MOEA-ExI and GP-MOEA-ExI over the number of FEs for $N = 20$. The mean IGD of HeE-MOEA-ExI improves faster than GP-MOEA-ExI on almost all problems except for DTLZ6, WFG5, and WFG7. The training time of the surrogates in each iteration and the total time of an optimization run for HeE-MOEA and GP-MOEA are plotted in Figs. 4 and 5, respectively. Although both algorithms are implemented in MATLAB, the computation time of the two algorithms for the same search dimension are not directly comparable due to different programming techniques, and the increase in computation time over the increase in search dimension is more important. We can observe that the training time of GP-MOEA increases much more rapidly than that of HeE-MOEA as the number of decision variables increases, so does the total time of the optimization runs. Four examples (DTLZ2, DTLZ3, WFG2, and WFG3) of the training time of the heterogeneous ensemble and GP versus the number of decision variables are plotted in Fig. 6. Thus, we can conclude that heterogeneous ensembles are computationally more efficient and scalable than GPs as we have discussed in Section I. The training time of the heterogeneous ensemble in DTLZ1 and DTLZ3 are slightly more than that in other test problems, which can be attributed

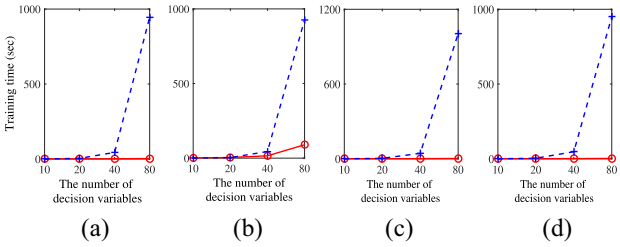

Fig. 6. Mean of training time of the surrogates in HeE-MOEA-LCB (the red solid lines with o) and GP-MOEA-LCB (the blue dashed lines with +) versus the number of decision variables on (a) DTLZ2; (b) DTLZ3; (c) WFG2; and (d) WFG3.

to the increase in training time of RBFN2. We surmise that many local PFs may bring difficulties in the convergence of weights in RBFN2.

E. Discussion

1) Influence of the Ensemble Aggregation Method: Here, we investigate three aggregation methods for calculating the final output of the ensemble, i.e., simple averaging, weighted aggregation (termed WTA1) [32], and partial least squares regression (PLSR) [70]. Both WTA1 and PLSR determine the weights by the estimated fitness values of each ensemble member for all training data. However, the root MSE is used in WTA1 while latent components are involved in PLSR. The comparative results in terms of IGD are provided in Table I in

TABLE III
MEAN (STANDARD DEVIATION) VALUES OF HV OBTAINED BY HEE-MOEA, HOE-MOEA, AND NSGA-II USING EXI INFILL CRITERION

Algorithm	N	DTLZ1	DTLZ2	DTLZ3	DTLZ4
HeE-MOEA	10	**0.9846** (2.9e-03)	**0.9341** (4.8e-03)	**0.9828** (5.2e-03)	0.7805 (6.0e-02)
	20	0.9705 (2.6e-03)	**0.9672** (3.5e-03)	0.9334 (7.4e-03)	**0.7642** (2.5e-02)
	40	0.9403 (2.6e-03)	**0.9621** (4.9e-03)	0.8702 (8.3e-03)	0.8770 (2.6e-02)
	80	0.9187 (7.2e-04)	0.9357 (7.9e-03)	0.7972 (6.2e-03)	0.8615 (2.5e-02)
HoE-MOEA	10	0.9724 (4.7e-03) $^+$	0.9137 (1.1e-02) $^+$	0.9577 (1.1e-02) $^+$	0.7725 (6.5e-02) $^=$
	20	0.9612 (3.3e-03) $^+$	0.9476 (5.5e-03) $^+$	0.9033 (8.8e-03) $^+$	0.7640 (2.3e-02) $^=$
	40	0.9356 (2.2e-03) $^+$	0.9528 (6.2e-03) $^+$	0.8572 (6.9e-03) $^+$	**0.8795** (2.8e-02) $^=$
	80	0.9161 (1.2e-03) $^+$	**0.9368** (6.0e-03) $^=$	0.7929 (5.5e-03) $^=$	**0.8634** (1.7e-02) $^=$
NSGA-II	10	0.9819 (7.0e-03) $^=$	0.8803 (1.7e-02) $^+$	0.9634 (1.4e-02) $^+$	**0.7998** (8.9e-02) $^=$
	20	**0.9714** (5.2e-03) $^=$	0.8987 (1.8e-02) $^+$	**0.9485** (9.7e-03) $^-$	0.7265 (1.0e-01) $^=$
	40	**0.9488** (5.5e-03) $^-$	0.8884 (2.1e-02) $^+$	**0.9000** (1.2e-02) $^-$	0.8706 (8.9e-02) $^=$
	80	**0.9326** (4.2e-03) $^-$	0.8448 (1.3e-02) $^+$	**0.8566** (1.1e-02) $^-$	0.8558 (7.4e-02) $^=$
Algorithm	N	DTLZ5	DTLZ6	DTLZ7	win/lose/tie
HeE-MOEA	10	0.8329 (6.4e-03)	**0.5581** (2.7e-02)	**0.6974** (4.5e-02)	
	20	**0.9047** (3.8e-03)	**0.5204** (1.1e-02)	**0.5595** (4.5e-02)	
	40	**0.9305** (4.0e-03)	**0.5249** (6.9e-03)	**0.4779** (2.1e-02)	
	80	**0.9242** (5.3e-03)	0.5028 (1.1e-03)	**0.3562** (1.9e-02)	
HoE-MOEA	10	**0.8345** (4.3e-03) $^=$	0.4900 (5.2e-03) $^+$	0.5699 (2.0e-02) $^+$	19/0/9
	20	0.8893 (8.1e-03) $^+$	0.4988 (3.3e-03) $^+$	0.4738 (1.6e-02) $^+$	
	40	0.9269 (7.8e-03) $^=$	0.5023 (1.5e-03) $^+$	0.4046 (1.2e-02) $^+$	
	80	0.9155 (9.4e-03) $^=$	0.4980 (1.0e-03) $^+$	0.3373 (4.3e-03) $^+$	
NSGA-II	10	0.7817 (1.6e-02) $^+$	0.5288 (2.6e-02) $^+$	0.5567 (3.5e-02) $^+$	15/5/8
	20	0.8055 (1.9e-02) $^+$	0.5157 (2.0e-02) $^=$	0.4533 (2.3e-02) $^+$	
	40	0.8437 (1.3e-02) $^+$	0.5012 (1.8e-02) $^+$	0.3973 (1.3e-02) $^+$	
	80	0.8244 (1.6e-02) $^+$	**0.5035** (9.0e-03) $^=$	0.3380 (1.1e-02) $^+$	

supplementary materials, which show that there is hardly any significant difference in the performances for the three aggregation methods. Thus in the following studies, we will use the simple averaging method only for calculating the output of the ensemble in HeE-MOEA.

2) Influence of the Number of the Initial Samples: Not much work has been reported on the influence of the allocation of computational budget (expensive FEs) between the offline training of the surrogate and the online sampling of the points found by SAEAs. Shi *et al.* [71] concluded that $3N$ (N is the search dimension) initial samples for offline training were sufficient, while in [72], it was found that more samples for offline training were necessary to obtain good performance and $32N$ initial samples were used. Here, we conduct some comparative experiments on the performance of HeE-MOEA using different initial samples to investigate the influence of the number of the initial samples. The results are listed in Table II in supplementary materials. From these results, we can conclude that the differences among the performances of different cases are small. Thus, in this paper, we use $11N - 1$ samples as recommended in many other SAEAs [25], [27], [28].

3) Performance Differences Between HeE-MOEA and GP-MOEA: Theoretically, if the HeE can estimate the fitness and uncertainty information exactly as the GP, the final optimization results should be very similar. From the above-presented comparative results, however, it is clear that the effectiveness of HeE and GP varies on different test problems. To gain deeper insights into the reason for the performance differences, we conduct additional experiments on four DTLZ test problems to examine the performance of the two models for fitness and uncertainty estimation. For this purpose, the two models are trained using the same $11N - 1$ training data and the estimated function values $\hat{f}$ and variances $\hat{s}^2$ of 500 randomly generated points are used to compute the MSE and the standard deviation of $\hat{s}$, respectively.

The experimental results are plotted in Fig. 1 in supplementary materials. From Fig. 1, we can see that the standard deviation of $\hat{s}$ of GP is smaller in four test instances, indicating that the estimated variances of GP for different sample points do not change as dramatically as those of HeE. We surmise that the ineffective uncertainty estimation of GP leads to the poor performance of GP-MOEA on some MOPs, such as DTLZ5. By contrast, the MSE of HeE is larger than that of GP, so the inaccurate fitness estimates may be the reason why GP-MOEA sometimes performs better than HeE-MOEA. For example, although the standard deviation of $\hat{s}$ obtained by GP on DTLZ7 is almost zero, its MSE also approaches to zero. This may explains why HeE-MOEA is outperformed by GP-MOEA on DTLZ7. Note that the infill criteria assess the value of a solution according to both its fitness estimates and variance. On DTLZ1 and DTLZ3, HeE-MOEA has poorer approximation performance than GP while GP has poor uncertainty estimation, and this may lead to the fact that HeE-MOEA and GP-MOEA using ExI or LCB perform similarly on these two test instances.

F. Comparison With HoE-MOEA and MOEA Without Surrogates

Two MOEAs, NSGA-II, and MOEA/D, and ExI infill criterion are taken as examples in this section. The statistics of the HV values of the solutions obtained by HeE-MOEA,

TABLE IV
MEAN (STANDARD DEVIATION) VALUES OF IGD OBTAINED BY HEE-MOEA AND MOEA/D USING EXI INFILL CRITERION ON 20-D WFG TEST INSTANCES

Problem	HeE-MOEA	MOEA/D
WFG1	**2.0431** (1.2e-01)	2.3831 (2.2e-01) $^+$
WFG2	**0.8040** (5.4e-02)	0.8663 (6.3e-02) $^+$
WFG3	**0.6497** (4.5e-02)	0.7307 (6.1e-02) $^+$
WFG4	**0.5434** (3.2e-02)	0.6812 (9.9e-02) $^+$
WFG5	**0.7037** (4.0e-02)	0.7833 (3.8e-02) $^+$
WFG6	**0.8496** (3.3e-02)	0.9253 (5.1e-02) $^+$
WFG7	**0.7112** (1.7e-02)	0.7870 (5.3e-02) $^+$
WFG8	**0.7934** (1.9e-02)	0.9043 (6.1e-02) $^+$
WFG9	**0.9411** (3.6e-02)	0.9700 (8.4e-02) $^=$
win/lose/tie	8/0/1	

HoE-MOEA, and NSGA-II on DTLZ test suite are listed in Table III. In HeE-MOEA and HoE-MOEA of Table III, NSGA-II is the base MOEA. We also use win/lose/tie to summarize the overall results. HeE-MOEA shows the most competitive performance on DTLZ2, DTLZ6 and DTLZ7, and NSGA-II shows the best performance on DTLZ1 and DTLZ3. This observation agrees with the results found in [15] showing that a surrogate-assisted MOEA performs worse than the MOEA itself on DTLZ1 and DTLZ3 probably because of the difficulties in approximating the fitness landscapes for multimodal problems. Compared with NSGA-II, HeE-MOEA is also significantly better on DTLZ5, and is no worse on DTLZ4. HoE-MOEA is significantly outperformed by HeE-MOEA on DTLZ1 and DTLZ3, while the performances of the two algorithms on DTLZ4 and DTLZ5 are similar. The nondominated solutions obtained by HeE-MOEA and HoE-MOEA are provided in Fig. 2 in supplementary materials, from which we can see that the nondominated solutions obtained by HeE-MOEA show a better distribution than those obtained by HoE-MOEA. Table IV presents the statistical results in terms of IGD obtained by MOEA/D and HeE-MOEA (which uses MOEA/D as the base MOEA) on the 20-D WFG test problems. HeE-MOEA shows significant advantages over MOEA/D on almost all test instances. Overall, HeE-MOEA shows the best performance among the four compared algorithms on the test problems for the dimensions studied in this paper.

VI. CONCLUSION

In many real-world optimization problems, it is common that FEs are time-consuming, and online SAEAs are often resorted to for efficient search of optimal solutions. While GPs based on infill criteria have shown to be particularly competitive on low-dimensional problems, their computational complexity may become prohibitive for high-dimensional problems as more training samples are required to train the GP. In this paper, we propose to use heterogeneous ensembles to replace GPs as surrogates in that ensembles are also able to estimate the uncertainty in approximating the objective values and are computationally more scalable to the increase in training samples. To enhance the diversity and accuracy of the ensemble, both feature selection and extraction are adopted to manipulate inputs, and different types of machine learning models, LSSVM and RBFN, are used to build ensemble members in this paper. Our empirical results confirm that the heterogeneous ensemble assisted MOEAs are comparable to or better than the GP assisted MOEAs, and the computation time of the former remains very attractive when the search dimension increases.

Although the computational time for the heterogeneous ensemble is computationally more efficient than GP on high-dimensional problems, PSO-based feature selection can become computationally intensive. Our next step will be to develop a more efficient ensemble construction method that does not rely on feature selection. In addition, it might of interest to develop more effective infill criteria that take into account the more properties of ensembles.

REFERENCES

[1] E. Zitzler, M. Laumanns, and L. Thiele, "SPEA2: Improving the strength Pareto evolutionary algorithm," in *Proc. EUROGEN Evol. Methods Design Optim. Control Appl. Ind. Problems*, 2001, pp. 95–100.
[2] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Trans. Evol. Comput.*, vol. 6, no. 2, pp. 182–197, Apr. 2002.
[3] Q. Zhang and H. Li, "MOEA/D: A multiobjective evolutionary algorithm based on decomposition," *IEEE Trans. Evol. Comput.*, vol. 11, no. 6, pp. 712–731, Dec. 2007.
[4] H.-L. Liu, F. Gu, and Q. Zhang, "Decomposition of a multiobjective optimization problem into a number of simple multiobjective subproblems," *IEEE Trans. Evol. Comput.*, vol. 18, no. 3, pp. 450–455, Jun. 2014.
[5] K. Li, Q. Zhang, S. Kwong, M. Li, and R. Wang, "Stable matching-based selection in evolutionary multiobjective optimization," *IEEE Trans. Evol. Comput.*, vol. 18, no. 6, pp. 909–923, Dec. 2014.
[6] E. Zitzler and S. Künzli, "Indicator-based selection in multiobjective search," in *Proc. Int. Conf. Parallel Problem Solving Nat.*, Birmingham, U.K., 2004, pp. 832–842.
[7] N. Beume, B. Naujoks, and M. Emmerich, "SMS-EMOA: Multiobjective selection based on dominated hypervolume," *Eur. J. Oper. Res.*, vol. 181, no. 3, pp. 1653–1669, 2007.
[8] Y. Jin and B. Sendhoff, "A systems approach to evolutionary multiobjective structural optimization and beyond," *IEEE Comput. Intell. Mag.*, vol. 4, no. 3, pp. 62–76, Aug. 2009.
[9] Y. S. Ong, P. B. Nair, and A. J. Keane, "Evolutionary optimization of computationally expensive problems via surrogate modeling," *AIAA J.*, vol. 41, no. 4, pp. 687–696, 2003.
[10] Y. Jin, "A comprehensive survey of fitness approximation in evolutionary computation," *Soft Comput.*, vol. 9, no. 1, pp. 3–12, 2005.
[11] H. Nakayama, K. Inoue, and Y. Yoshimori, "Approximate optimization using computational intelligence and its application to reinforcement of cable-stayed bridges," in *Proc. Integr. Intell. Syst. Eng. Design*, 2006, pp. 289–304.
[12] W. Kong, T. Chai, S. Yang, and J. Ding, "A hybrid evolutionary multiobjective optimization strategy for the dynamic power supply problem in magnesia grain manufacturing," *Appl. Soft Comput.*, vol. 13, no. 5, pp. 2960–2969, 2013.
[13] D. Lim, Y. Jin, Y.-S. Ong, and B. Sendhoff, "Generalizing surrogate-assisted evolutionary computation," *IEEE Trans. Evol. Comput.*, vol. 14, no. 3, pp. 329–355, Jun. 2010.
[14] K. Rasheed, X. Ni, and S. Vattam, "Comparison of methods for developing dynamic reduced models for design optimization," *Soft Comput.*, vol. 9, no. 1, pp. 29–37, 2005.
[15] T. Chugh, Y. Jin, K. Miettinen, J. Hakanen, and K. Sindhya, "A surrogate-assisted reference vector guided evolutionary algorithm for computationally expensive many-objective optimization," *IEEE Trans. Evol. Comput.*, to be published.
[16] Y. Jin, "Surrogate-assisted evolutionary computation: Recent advances and future challenges," *Swarm Evol. Comput.*, vol. 1, no. 2, pp. 61–70, 2011.
[17] H. Wang, Y. Jin, and J. O. Janson, "Data-driven surrogate-assisted multiobjective evolutionary optimization of a trauma system," *IEEE Trans. Evol. Comput.*, vol. 20, no. 6, pp. 939–952, Dec. 2016.

[18] D. R. Jones, M. Schonlau, and W. J. Welch, "Efficient global optimization of expensive black-box functions," *J. Glob. Optim.*, vol. 13, no. 4, pp. 455–492, 1998.

[19] M. T. M. Emmerich, K. C. Giannakoglou, and B. Naujoks, "Single-and multiobjective evolutionary optimization assisted by Gaussian random field metamodels," *IEEE Trans. Evol. Comput.*, vol. 10, no. 4, pp. 421–439, Aug. 2006.

[20] V. Torczon and M. Trosset, "Using approximations to accelerate engineering design optimization," in *Proc. 7th AIAA/USAF/NASA/ISSMO Symp. Multidiscipl. Anal. Optim.*, 1998, p. 4800.

[21] S. N. Lophaven, H. B. Nielsen, and J. Søndergaard, "DACE: A MATLAB Kriging toolbox, version 2.0," Tech. Univ. Denmark, Kongens Lyngby, Denmark, Tech. Rep. IMM-TR-2002-12, 2002.

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

[23] M. Emmerich, A. Giotis, M. Uezdenir, T. Baeck, and K. Giannakoglou, "Metamodel—Assisted evolution strategies," in *Parallel Problem Solving From Nature* (Lecture Notes in Computer Science). Heidelberg, Germany: Springer, 2002, pp. 371–380.

[24] L. Willmes, T. Back, Y. Jin, and B. Sendhoff, "Comparing neural networks and kriging for fitness approximation in evolutionary optimization," in *Proc. IEEE Congr. Evol. Comput.*, Canberra, ACT, Australia, 2003, pp. 663–670.

[25] J. Knowles, "ParEGO: A hybrid algorithm with on-line landscape approximation for expensive multiobjective optimization problems," *IEEE Trans. Evol. Comput.*, vol. 10, no. 1, pp. 50–66, Feb. 2006.

[26] W. Ponweiser, T. Wagner, D. Biermann, and M. Vincze, "Multiobjective optimization on a limited budget of evaluations using model-assisted $\mathcal{S}$-metric selection," in *Proc. Int. Conf. Parallel Problem Solving Nat.*, Dortmund, Germany, 2008, pp. 784–794.

[27] Q. Zhang, W. Liu, E. Tsang, and B. Virginas, "Expensive multiobjective optimization by MOEA/D with Gaussian process model," *IEEE Trans. Evol. Comput.*, vol. 14, no. 3, pp. 456–474, Jun. 2010.

[28] D. Horn, T. Wagner, D. Biermann, C. Weihs, and B. Bischl, "Model-based multi-objective optimization: Taxonomy, multi-point proposal, toolbox and benchmark," in *Proc. Int. Conf. Evol. Multi Criterion Optim.*, Guimarães, Portugal, 2015, pp. 64–78.

[29] D. Büche, N. N. Schraudolph, and P. Koumoutsakos, "Accelerating evolutionary algorithms with Gaussian process fitness function models," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 35, no. 2, pp. 183–194, May 2005.

[30] C. Sun, Y. Jin, R. Cheng, J. Ding, and J. Zeng, "Surrogate-assisted cooperative swarm optimization of high-dimensional expensive problems," *IEEE Trans. Evol. Comput.*, vol. 21, no. 4, pp. 644–660, Aug. 2017.

[31] Y. Jin and B. Sendhoff, "Reducing fitness evaluations using clustering techniques and neural network ensembles," in *Proc. Genet. Evol. Comput. Conf.*, Seattle, WA, USA, 2004, pp. 688–699.

[32] T. Goel, R. T. Haftka, W. Shyy, and N. V. Queipo, "Ensemble of surrogates," *Struct. Multidiscipl. Optim.*, vol. 33, no. 3, pp. 199–216, 2007.

[33] G. Brown, J. L. Wyatt, and P. Tiňo, "Managing diversity in regression ensembles," *J. Mach. Learn. Res.*, vol. 6, pp. 1621–1650, Dec. 2005.

[34] L. Breiman, "Bagging predictors," *Mach. learn.*, vol. 24, no. 2, pp. 123–140, 1996.

[35] S. Gu and Y. Jin, "Multi-train: A semi-supervised heterogeneous ensemble classifier," *Neurocomputing*, vol. 249, pp. 202–211, Aug. 2017.

[36] W. A. Albukhanajer, Y. Jin, and J. A. Briffa, "Classifier ensembles for image identification using multi-objective Pareto features," *Neurocomputing*, vol. 238, pp. 317–327, May 2017.

[37] J. Branke and C. Schmidt, "Faster convergence by means of fitness estimation," *Soft Comput. Fusion Found. Methodol. Appl.*, vol. 9, no. 1, pp. 13–20, 2005.

[38] H. Wang, Y. Jin, and J. Doherty, "Committee-based active learning for surrogate-assisted particle swarm optimization of expensive problems," *IEEE Trans. Cybern.*, vol. 47, no. 9, pp. 2664–2677, Sep. 2017.

[39] A. Rosales-Pérez, C. A. C. Coello, J. A. Gonzalez, C. A. Reyes-Garcia, and H. J. Escalante, "A hybrid surrogate-based approach for evolutionary multi-objective optimization," in *Proc. IEEE Congr. Evol. Comput.*, Cancún, Mexico, 2013, pp. 2548–2555.

[40] S. Z. Martínez and C. A. C. Coello, "MOEA/D assisted by RBF networks for expensive multi-objective optimization problems," in *Proc. Genet. Evol. Comput. Conf.*, Amsterdam, The Netherlands, 2013, pp. 1405–1412.

[41] N. Azzouz, S. Bechikh, and L. B. Said, "Steady state IBEA assisted by MLP neural networks for expensive multi-objective optimization problems," in *Proc. Genet. Evol. Comput. Conf.*, Vancouver, BC, Canada, 2014, pp. 581–588.

[42] Y. Ren, L. Zhang, and P. N. Suganthan, "Ensemble classification and regression-recent developments, applications and future directions," *IEEE Comput. Intell. Mag.*, vol. 11, no. 1, pp. 41–53, Feb. 2016.

[43] B. Tran, B. Xue, and M. Zhang, "Overview of particle swarm optimisation for feature selection in classification," in *Proc. Asia–Pac. Conf. Simulat. Evol. Learn.*, Dunedin, New Zealand, 2014, pp. 605–617.

[44] S. Gu, R. Cheng, and Y. Jin, "Feature selection for high-dimensional classification using a competitive swarm optimizer," *Soft Comput.*, pp. 1–12, 2016, doi: 10.1007/s00500-016-2385-6.

[45] J. R. Vergara and P. A. Estévez, "A review of feature selection methods based on mutual information," *Neural Comput. Appl.*, vol. 24, no. 1, pp. 175–186, 2014.

[46] G. Chandrashekar and F. Sahin, "A survey on feature selection methods," *Comput. Elect. Eng.*, vol. 40, no. 1, pp. 16–28, 2014.

[47] V. Bolón-Canedo, N. Sánchez-Maroño, and A. Alonso-Betanzos, "A review of feature selection methods on synthetic data," *Knowl. Inf. Syst.*, vol. 34, no. 3, pp. 483–519, 2013.

[48] D. R. Hardoon, S. R. Szedmak, and J. R. Shawe-Taylor, "Canonical correlation analysis: An overview with application to learning methods," *Neural Comput.*, vol. 16, no. 12, pp. 2639–2664, 2004.

[49] B. Schölkopf, A. Smola, and K.-R. Müller, "Kernel principal component analysis," in *Proc. Int. Conf. Artif. Neural Netw.*, 1997, pp. 583–588.

[50] J. Shlens. (2014). *A Tutorial on Principal Component Analysis*. [Online]. Available: https://arxiv.org/abs/1404.1100

[51] J. A. K. Suykens and J. Vandewalle, "Least squares support vector machine classifiers," *Neural Process. Lett.*, vol. 9, no. 3, pp. 293–300, 1999.

[52] Z. Mustaffa and Y. Yusof, "LSSVM parameters tuning with enhanced artificial bee colony," *Int. Arab J. Inf. Technol.*, vol. 11, no. 3, pp. 236–242, 2014.

[53] G. Brown, J. Wyatt, R. Harris, and X. Yao, "Diversity creation methods: A survey and categorisation," *Inf. Fusion*, vol. 6, no. 1, pp. 5–20, 2005.

[54] R. Tibshirani, "A comparison of some error estimates for neural network models," *Neural Comput.*, vol. 8, no. 1, pp. 152–163, 1996.

[55] S. Gu, R. Cheng, and Y. Jin, "Multi-objective ensemble generation," *Interdiscipl. Rev. Data Min. Knowl. Disc.*, vol. 5, no. 5, pp. 234–245, 2015.

[56] A. J. Sharkey, *Combining Artificial Neural Nets: Ensemble and Modular Multi-Net Systems*. London, U.K.: Springer, 2012.

[57] H. Drucker, "Improving regressors using boosting techniques," in *Proc. Int. Conf. Mach. Learn.*, vol. 97, 1997, pp. 107–115.

[58] H. Chen and X. Yao, "Multiobjective neural network ensembles based on regularized negative correlation learning," *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 12, pp. 1738–1751, Dec. 2010.

[59] Y. Jin and B. Sendhoff, "Pareto-based multiobjective machine learning: An overview and case studies," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 38, no. 3, pp. 397–415, May 2008.

[60] S. Gu and Y. Jin, "Heterogeneous classifier ensembles for EEG-Based motor imaginary detection," in *Proc. 12th U.K. Workshop Comput. Intell.*, Edinburgh, U.K., 2012, pp. 1–8.

[61] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proc. IEEE Int. Conf. Neural Netw.*, vol. 4. Perth, WA, Australia, 1995, pp. 1942–1948.

[62] G. J. Székely, M. L. Rizzo, and N. K. Bakirov, "Measuring and testing dependence by correlation of distances," *Ann. Stat.*, vol. 35, no. 6, pp. 2769–2794, 2007.

[63] L. Cervante, B. Xue, M. Zhang, and L. Shang, "Binary particle swarm optimisation for feature selection: A filter based approach," in *Proc. IEEE Congr. Evol. Comput.*, Brisbane, QLD, Australia, 2012, pp. 1–8.

[64] K. Deb, L. Thiele, M. Laumanns, and E. Zitzler, "Scalable multi-objective optimization test problems," in *Proc. IEEE Congr. Evol. Comput.*, vol. 1. Honolulu, HI, USA, 2002, pp. 825–830.

[65] S. Huband, P. Hingston, L. Barone, and L. While, "A review of multiobjective test problems and a scalable test problem toolkit," *IEEE Trans. Evol. Comput.*, vol. 10, no. 5, pp. 477–506, Oct. 2006.

[66] L. While, P. Hingston, L. Barone, and S. Huband, "A faster algorithm for calculating hypervolume," *IEEE Trans. Evol. Comput.*, vol. 10, no. 1, pp. 29–38, Feb. 2006.

[67] P. A. N. Bosman and D. Thierens, "The balance between proximity and diversity in multiobjective evolutionary algorithms," *IEEE Trans. Evol. Comput.*, vol. 7, no. 2, pp. 174–188, Apr. 2003.

[68] Y. Tian, R. Cheng, X. Zhang, and Y. Jin, "PlatEMO: A MATLAB platform for evolutionary multi-objective optimization," *IEEE Comput. Intell. Mag.*, vol. 12, no. 4, pp. 73–87, Nov. 2017.
[69] C.-C. Chang and C.-J. Lin, "LIBSVM: A library for support vector machines," *ACM Trans. Intell. Syst. Technol.*, vol. 2, no. 3, p. 27, 2011.
[70] S. Li, P. Wang, and L. Goel, "A novel wavelet-based ensemble method for short-term load forecasting with hybrid neural networks and feature selection," *IEEE Trans. Power Syst.*, vol. 31, no. 3, pp. 1788–1798, May 2016.
[71] L. Shi, R. J. Yang, and P. Zhu, "A method for selecting surrogate models in crashworthiness optimization," *Struct. Multidiscipl. Optim.*, vol. 46, no. 2, pp. 159–170, 2012.
[72] P. van Heiningen, B. van Stein, and T. Bäck, "A framework for evaluating meta-models for simulation-based optimisation," in *Proc. IEEE Symp. Comput. Intell. (SSCI)*, Athens, Greece, 2016, pp. 1–8.

Dan Guo received the B.S. degree in automation from Northeastern University, Shenyang, China, in 2012, where she is currently pursuing the Ph.D. degree with the State Key Laboratory of Synthetical Automation for Process Industry.

Her current research interests include surrogate-assisted multiobjective evolutionary optimization and machine learning.

Yaochu Jin (M'98–SM'02–F'16) received the B.Sc., M.Sc., and Ph.D. degrees from Zhejiang University, Hangzhou, China, in 1988, 1991, and 1996, respectively, and the Dr.-Ing. degree from Ruhr University Bochum, Bochum, Germany, in 2001.

He is a Professor in computational intelligence with the Department of Computer Science, University of Surrey, Guildford, U.K., where he heads the Nature Inspired Computing and Engineering Group. He is also a Finland Distinguished Professor funded by the Finnish Agency for Innovation (Tekes) and a Changjiang Distinguished Visiting Professor appointed by the Ministry of Education, China. His current research interests include data-driven surrogate-assisted evolutionary optimization, evolutionary multiobjective optimization, evolutionary learning, interpretable and secure machine learning, and evolutionary developmental systems. He has (co)authored over 250 peer-reviewed journal and conference papers and been granted eight patents on evolutionary optimization. He has delivered 30 invited keynote speeches at international conferences.

Dr. Jin was a recipient of the 2018 IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION Outstanding Paper Award, the 2015 and 2017 *IEEE Computational Intelligence Magazine* Outstanding Paper Award, and the Best Paper Award of the 2010 IEEE Symposium on Computational Intelligence in Bioinformatics and Computational Biology. He is the Editor-in-Chief of the IEEE TRANSACTIONS ON COGNITIVE AND DEVELOPMENTAL SYSTEMS and the Co-Editor-in-Chief of *Complex and Intelligent Systems*. He is an IEEE Distinguished Lecturer from 2013 to 2015 and from 2017 to 2019 and the past Vice President for Technical Activities of the IEEE Computational Intelligence Society from 2014 to 2015.

Jinliang Ding (SM'14) received the Ph.D. degree in control theory and control engineering from Northeastern University, Shenyang, China, in 2012.

He is a Professor with the State Key Laboratory of Synthetical Automation for Process Industry, Northeastern University. He has authored or co-authored over 90 refereed journal papers and refereed papers at international conferences. He is also the Inventor or Co-Inventor of 17 patents. His current research interests include modeling, plant-wide control and optimization for the complex industrial systems, stochastic distribution control, and multiobjective evolutionary algorithms and its application.

Dr. Ding was a recipient of the Young Scholars Science and Technology Award of China in 2016, the National Science Fund for Distinguished Young Scholars in 2015, the National Technological Invention Award in 2013, two First-Prize of Science and Technology Award of the Ministry of Education in 2006 and 2012, respectively, and the Best Paper Award of 2011–2013 for Control Engineering Practice.

Tianyou Chai (M'90–SM'97–F'08) received the Ph.D. degree in control theory and engineering from Northeastern University, Shenyang, China, in 1985.

He is the Founder and the Director of the Center of Automation, which became a National Engineering and Technology Research Center and a State Key Laboratory. He became a Professor with Northeastern University in 1988. He is the Director of the Department of Information Science, National Natural Science Foundation of China. He has authored 150 peer-reviewed international journal papers. He has developed control technologies with applications to various industrial processes. His current research interests include modeling, control, optimization, and integrated automation of complex industrial processes.

Dr. Chai was a recipient one of three Best Papers for the Control Engineering Practice Paper Prize for 2011–2013 for his paper entitled *Hybrid Intelligent Control for Optimal Operation of Shaft Furnace Roasting Process*, Four Prestigious Awards from the National Science and Technology Progress and National Technological Innovation for his contributions, and the 2007 Industry Award for Excellence in Transitional Control Research from the IEEE Multiple Conference on Systems and Control. He is a member of the Chinese Academy of Engineering and is an International Federation of Automatic Control Fellow.