Construction of Tunable Radial Basis Function Networks Using Orthogonal Forward Selection

Sheng Chen, Xia Hong, Bing L. Luk, and Chris J. Harris

***Abstract*—An orthogonal forward selection (OFS) algorithm based on leave-one-out (LOO) criteria is proposed for the construction of radial basis function (RBF) networks with tunable nodes. Each stage of the construction process determines an RBF node, namely, its center vector and diagonal covariance matrix, by minimizing the LOO statistics. For regression application, the LOO criterion is chosen to be the LOO mean-square error, while the LOO misclassification rate is adopted in two-class classification application. This OFS-LOO algorithm is computationally efficient, and it is capable of constructing parsimonious RBF networks that generalize well. Moreover, the proposed algorithm is fully automatic, and the user does not need to specify a termination criterion for the construction process. The effectiveness of the proposed RBF network construction procedure is demonstrated using examples taken from both regression and classification applications.**

***Index Terms*—Classification, leave-one-out (LOO) statistics, orthogonal forward selection (OFS), radial basis function (RBF) network, regression, tunable node.**

I. Introduction

THE RADIAL basis function (RBF) network is a popular artificial neural network architecture that has found wide applications in diverse fields of engineering [1]–[14]. The parameters of the RBF network include its center vectors and variances or the covariance matrices of the basis functions as well as the connecting weights from the RBF nodes to the network output. All the parameters of an RBF network can be learned together via nonlinear optimization using the gradient-based algorithms [15]–[18], the evolutionary algorithms [19]–[21], or the expectation-maximization algorithm [22], [23]. Generally, learning based on such a nonlinear approach is computationally expensive and may encounter the problem of local minima. Additionally, the network structure or the number of RBF nodes has to be determined via other means, typically based on cross-validation. Alternatively, clustering algorithms can be applied to find the RBF center vectors as well as the associated basis function variances [24]–[27]. This leaves the RBF weights to be determined by the usual linear least squares solution. Again, the number of the clusters has to be determined via other means, such as cross-validation.

One of the popular approaches for constructing RBF networks for regression is to formulate the problem as a linear learning one by considering the training input data points as candidate RBF centers and employing a common variance for every RBF node. A parsimonious RBF network is then identified using the orthogonal least squares (OLS) algorithm [28]–[32]. Similarly, the support vector machine (SVM), the relevance vector machine (RVM), and other sparse kernel modeling methods [33]–[39] also fix the kernel centers to the training input data points and adopt a common kernel variance for every kernel. A sparse kernel representation is then sought. Since the common variance is not provided by the learning algorithms in this *linear* learning approach, it must be treated as a hyperparameter and determined via cross-validation. For the kernel modeling methods, additionally, some learning algorithm's hyperparameters also have to be determined by cross-validation. For example, for the SVM algorithm with the ε insensitive cost function [34], the regularization parameter and the value of ε must be specified. The experimental results obtained in [32] show that the locally regularized OLS algorithm based on the leave-one-out (LOO) mean-square error (mse) criterion (LROLS-LOO) compares favorably with many other existing sparse kernel modeling methods for regression modeling, in terms of model sparsity and generalization performance.

The sparse kernel modeling methods [33]–[39] are equally applicable to classification. A recent work [40] has developed an orthogonal forward selection (OFS) algorithm based on minimizing the LOO misclassification rate for two-class classification application using RBF classifiers. Because of orthogonal decomposition, the LOO misclassification rate can be computed efficiently, just as in the case of the LOO mse for regression [32], and this ensures a fast RBF classifier construction. As with other sparse kernel modeling methods, the RBF variance is treated as a hyperparameter and determined via cross-validation. This RBF classifier construction algorithm can be viewed as the LROLS-LOO algorithm for classification. One aspect of the *linear* learning approach for RBF models, which deserves consideration, is the true computational cost. Given the RBF variance, the LROLS-LOO algorithm is computationally very efficient. However, the true computational cost should include hyperparameter learning, which is typically via a grid-search-based cross-validation. For sparse kernel modeling methods, such as the SVM with two or three hyperparameters, the total computational requirements can become very costly.

This paper proposes a construction algorithm for the tunable RBF network, where each RBF node has a tunable center

Manuscript received March 31, 2008; revised July 3, 2008. First published December 16, 2008; current version published March 19, 2009. This paper was recommended by Associate Editor Q. Zhao.

S. Chen and C. J. Harris are with the School of Electronics and Computer Science, University of Southampton, SO17 1BJ Southampton, U.K.

X. Hong is with the School of Systems Engineering, University of Reading, RG6 6AY Reading, U.K.

B. L. Luk is with the Department of Manufacturing Engineering and Engineering Management, City University of Hong Kong, Kowloon, Hong Kong, China.

Digital Object Identifier 10.1109/TSMCB.2008.2006688

vector and an adjustable diagonal covariance matrix. An OFS procedure is developed to append the RBF units one by one by minimizing the LOO statistics. For constructing RBF classifiers, the LOO criterion is chosen to be the LOO misclassification rate, while the LOO mse is used for selecting RBF networks in regression application. Because the RBF centers are not restricted to the training input points and each RBF node has an adjusted covariance matrix, the proposed OFS-LOO algorithm can produce sparser representations with excellent generalization capability, in comparison with the existing sparse kernel modeling methods. In addition, our algorithm does not have hyperparameters that must be learning via costly cross-validation. Our method is also very different from those RBF learning methods based on nonlinear optimization, as we do not attempt to optimize all the RBF units together, which could be a too large and complicated nonlinear optimization task. Rather, we optimize one RBF node at each stage of construction based on the LOO criterion. The determination of the RBF center vector and diagonal covariance matrix at each stage can readily be carried out by an efficient global search algorithm called the repeated weighted boosting search (RWBS) [41]. Moreover, because the LOO criterion is "locally convex" with respect to the model size [32], [42], the construction process is fully automatic, and there is no need for the user to specify additional termination criterion in order to determine the size of the RBF network.

Finally, we emphasize the novelty of our proposed approach by highlighting the differences of this algorithm with our previous LROLS-LOO algorithm (for regression [32] and classification [40]). The LROLS-LOO algorithm places candidate RBF centers at all the training data points and employs a common RBF variance for every RBF unit. The model is then selected from the resulting big candidate model set using the OFS procedure. The common RBF variance is treated as a hyperparameter and determined via costly cross-validation. The proposed OFS-LOO algorithm learns the RBF units, which have tunable center vectors and diagonal covariance matrices, one by one via an OFS procedure. Although determining a tunable RBF unit may cost more than selecting a fixed RBF unit, the algorithm constructs fewer RBF units, compared with the LROLS-LOO algorithm. Moreover, the proposed OFS-LOO algorithm does not have any hyperparameter and achieves substantial saving in cross-validation. The overall computational cost of this OFS-LOO algorithm is not necessarily more than that of the LROLS-LOO algorithm. Our experimental results demonstrate that the novel OFS-LOO algorithm compares favorably with several sparse RBF or kernel modeling methods, including the LROLS-LOO algorithm.

II. Regression Using the Tunable RBF Network

Consider the regression problem of approximating the N pairs of training data $\{(\mathbf{x}_k, y_k)\}_{k=1}^N$ with the RBF network defined in

$$y_k = \hat{y}_k + e_k = \sum_{i=1}^{M} w_i g_i(\mathbf{x}_k) + e_k = \mathbf{g}^{\mathrm{T}}(k)\mathbf{w} + e_k \quad (1)$$

where the input $\mathbf{x}_k \in \mathcal{R}^m$, the desired output $y_k \in \mathcal{R}$, $\hat{y}_k$ denotes the RBF model output, $e_k = y_k - \hat{y}_k$ is the modeling error, M is the number of RBF units, $\mathbf{w} = [w_1 \quad w_2 \quad \cdots \quad w_M]^{\mathrm{T}}$ is the RBF weight vector, $g_i(\bullet)$ for $1 \leq i \leq M$ denote the RBF regressors, and $\mathbf{g}(k) = [g_1(\mathbf{x}_k) \quad g_2(\mathbf{x}_k) \quad \cdots \quad g_M(\mathbf{x}_k)]^{\mathrm{T}}$. We will consider the general RBF regressor of the form

$$g_i(\mathbf{x}) = K\left(\sqrt{(\mathbf{x}-\boldsymbol{\mu}_i)^{\mathrm{T}}\boldsymbol{\Sigma}_i^{-1}(\mathbf{x}-\boldsymbol{\mu}_i)}\right) \quad (2)$$

where $\boldsymbol{\mu}_i$ is the center vector of the ith RBF unit, the diagonal covariance matrix has the form $\boldsymbol{\Sigma}_i = \mathrm{diag}\{\sigma_{i,1}^2, \ldots, \sigma_{i,m}^2\}$, and $K(\bullet)$ is the chosen basis or kernel function. By defining $\mathbf{y} = [y_1 \quad y_2 \quad \cdots \quad y_N]^{\mathrm{T}}$, $\mathbf{e} = [e_1 \quad e_2 \quad \cdots \quad e_N]^{\mathrm{T}}$, and $\mathbf{G} = [\mathbf{g}_1 \quad \mathbf{g}_2 \quad \cdots \quad \mathbf{g}_M]$ with

$$\mathbf{g}_l = [g_l(\mathbf{x}_1) \quad g_l(\mathbf{x}_2) \quad \cdots \quad g_l(\mathbf{x}_N)]^{\mathrm{T}}, \quad 1 \leq l \leq M \quad (3)$$

the regression model (1) over the training data set can be written in the matrix form

$$\mathbf{y} = \mathbf{G}\mathbf{w} + \mathbf{e}. \quad (4)$$

Here, $\mathbf{g}_k$ is the kth column of $\mathbf{G}$ while $\mathbf{g}^{\mathrm{T}}(k)$ the kth row of $\mathbf{G}$.

A. OFS Based on the LOO Mean Square Error

Let an orthogonal decomposition of $\mathbf{G}$ be $\mathbf{G} = \mathbf{P}\mathbf{A}$, where

$$\mathbf{A} = \begin{bmatrix} 1 & \alpha_{1,2} & \cdots & \alpha_{1,M} \\ 0 & 1 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \alpha_{M-1,M} \\ 0 & \cdots & 0 & 1 \end{bmatrix} \quad (5)$$

$$\mathbf{P} = [\mathbf{p}_1 \quad \mathbf{p}_2 \quad \cdots \quad \mathbf{p}_M] \quad (6)$$

with the orthogonal columns that satisfy $\mathbf{p}_i^{\mathrm{T}}\mathbf{p}_j = 0$ if $i \neq j$. The regression model (4) can alternatively be expressed as

$$\mathbf{y} = \mathbf{P}\boldsymbol{\theta} + \mathbf{e} \quad (7)$$

where $\boldsymbol{\theta} = [\theta_1 \quad \theta_2 \quad \cdots \quad \theta_M]^{\mathrm{T}}$ satisfies the triangular system $\mathbf{A}\mathbf{w} = \boldsymbol{\theta}$. Since the space spanned by the original model bases $g_i(\bullet)$, $1 \leq i \leq M$, is identical to the space spanned by the orthogonal model bases, the RBF model output is equivalently expressed by

$$\hat{y}_k = \mathbf{p}^{\mathrm{T}}(k)\boldsymbol{\theta} \quad (8)$$

where $\mathbf{p}^{\mathrm{T}}(k) = [p_1(k) \quad p_2(k) \quad \cdots \quad p_M(k)]$ is the kth row of $\mathbf{P}$.

Consider the modeling process that has produced the n-unit model. Let us denote the constructed n model columns as $\mathbf{P}_n = [\mathbf{p}_1 \quad \mathbf{p}_2 \quad \cdots \quad \mathbf{p}_n]$, the kth model output of this n-unit model identified using the entire training data set as $\hat{y}_k^{(n)} = \sum_{i=1}^{n} \theta_i p_i(k)$, and the corresponding kth modeling error as $e_k^{(n)} = y_k - \hat{y}_k^{(n)}$. If we "remove" the kth data point from the training data set and use the remaining $N-1$ data points to identify the n-unit RBF network instead, the "test" error of the resulting model can be calculated on the data point removed

from training. This LOO modeling error, denoted as $e_k^{(n,-k)}$, is given by [43]

$$e_k^{(n,-k)} = e_k^{(n)}/\eta_k^{(n)} \tag{9}$$

where $\eta_k^{(n)}$ is the LOO error weighting [43]. The LOO mse for the n-unit RBF network is then defined by

$$J_n = \frac{1}{N}\sum_{k=1}^{N}\left(e_k^{(n,-k)}\right)^2. \tag{10}$$

J_n is a measure of the model generalization capability [43], [44]. For (8), the computation of the LOO criterion J_n is very efficient because $e_k^{(n)}$ and $\eta_k^{(n)}$ can be computed recursively using [32], [42]

$$e_k^{(n)} = y_k - \sum_{i=1}^{n}\theta_i p_i(k) = e_k^{(n-1)} - \theta_n p_n(k) \tag{11}$$

$$\eta_k^{(n)} = 1 - \sum_{i=1}^{n}\frac{p_i^2(k)}{\mathbf{p}_i^{\mathrm{T}}\mathbf{p}_i + \lambda} = \eta_k^{(n-1)} - \frac{p_n^2(k)}{\mathbf{p}_n^{\mathrm{T}}\mathbf{p}_n + \lambda} \tag{12}$$

where $\lambda \geq 0$ is a small regularization parameter [32].

The proposed OFS-LOO algorithm constructs the RBF units one by one by minimizing the LOO mse J_n. Specifically, at the nth stage of the construction procedure, the nth RBF node is determined by minimizing J_n with respect to the node's center vector $\boldsymbol{\mu}_n$ and diagonal covariance matrix $\boldsymbol{\Sigma}_n$

$$\min_{\boldsymbol{\mu}_n, \boldsymbol{\Sigma}_n} J_n(\boldsymbol{\mu}_n, \boldsymbol{\Sigma}_n). \tag{13}$$

The construction procedure is automatically terminated when

$$J_M \leq J_{M+1} \tag{14}$$

yielding an M-term RBF network. Note that the LOO criterion J_n is at least locally convex with respect to the model size n, i.e., there exists an "optimal" M such that, for $n \leq M$, J_n decreases as the model size n increases while condition (14) holds [42]. After the OFS-LOO model construction, we have a very small model set containing only M units. At this stage, we may apply the LROLS-LOO algorithm of [32] to further reduce the model size and to automatically update the individual regularization parameter for each weight. This refinement requires a very small amount of computation, as the regression matrix $\mathbf{G}$ is completely specified with only a few columns. Note that, in the OFS-LOO algorithm, the regularization parameter λ can simply be set to zero (no regularization) or a very small value (e.g., 10^{-6}). The refinement with the LROLS-LOO will automatically optimize each regularization parameter for individual weight [32]. Our experience shows that, for regression, this refinement involving the LROLS-LOO is beneficial but, for classification, it is unnecessary (no further reduction in model size).

An advantage of this OFS-LOO algorithm is that the model construction is based directly on optimizing the model generalization capability without involving an additional validation data set and the model size is determined automatically without the need for the user to specify additional termination criterion. Moreover, this learning algorithm does not contain any hyperparameter which requires costly cross-validation to tune. The regularization parameter λ in (12) can simply be set to zero or a very small value. This is not the case for many existing sparse RBF modeling methods. For example, to use the SVM algorithm, the user has to specify the kernel variance, regularization parameter C, and the parameter that defines the loss function [34]. These learning hyperparameters have critical influence on the algorithm's performance and must be determined via cross-validation. In fact, most of the complexity for many existing learning algorithms is due to the need of tuning these hyperparameters.

B. Positioning and Shaping an RBF Node

The task at the nth stage of the RBF network construction is to solve the optimization problem (13). Since this optimization problem is nonconvex with respect to $\boldsymbol{\mu}_n$ and $\boldsymbol{\Sigma}_n$, a gradient-based algorithm may become trapped at a local minimum. Alternatively, global search methods, such as the genetic algorithm [45], [46] and adaptive simulated annealing [47], [48], may be used to perform the optimization task (13). We adopt a simple yet efficient global search algorithm called the RWBS [41] to determine $\boldsymbol{\mu}_n$ and $\boldsymbol{\Sigma}_n$. The motivation and analysis of the RWBS algorithm as a general global optimizer are detailed in [41], and they will not be repeated here. The procedure for determining the nth RBF unit based on the RWBS algorithm is now summarized. Let $\mathbf{u}$ be the vector that contains $\boldsymbol{\mu}_n$ and $\boldsymbol{\Sigma}_n$. Give the following initial conditions:

$$\left.\begin{array}{l} e_k^{(0)} = y_k \quad \text{and} \quad \eta_k^{(0)} = 1, \qquad 1 \leq k \leq N, \\ J_0 = \frac{1}{N}\mathbf{y}^{\mathrm{T}}\mathbf{y} = \frac{1}{N}\sum_{k=1}^{N} y_k^2. \end{array}\right\} \tag{15}$$

Specify the RWBS algorithmic parameters, namely, the population size P_S, the number of generations in the repeated search N_G, and the number of weighted boosting search iterations M_I.

Outer loop: generations For $(l = 1; l \leq N_G; l = l + 1)$ {
Generation initialization: Initialize the population by setting $\mathbf{u}_1^{[l]} = \mathbf{u}_{\text{best}}^{[l-1]}$ and randomly generating the rest of the population members $\mathbf{u}_i^{[l]}$, $2 \leq i \leq P_S$, where $\mathbf{u}_{\text{best}}^{[l-1]}$ denotes the solution found in the previous generation. If $l = 1$, $\mathbf{u}_1^{[l]}$ is also randomly chosen.
Weighted boosting search initialization: Assign the initial distribution weightings $\delta_i(0) = 1/P_S$, $1 \leq i \leq P_S$, for the population. Then

1) For $1 \leq i \leq P_S$, generate $\mathbf{g}_n^{i)}$ from $\mathbf{u}_i^{[l]}$, the candidates for the nth model column, and orthogonalize them

$$\alpha_{j,n}^{i)} = \mathbf{p}_j^{\mathrm{T}}\mathbf{g}_n^{i)} \Big/ \mathbf{p}_j^{\mathrm{T}}\mathbf{p}_j, \qquad 1 \leq j < n \tag{16}$$

$$\mathbf{p}_n^{i)} = \mathbf{g}_n^{i)} - \sum_{j=1}^{n-1}\alpha_{j,n}^{i)}\mathbf{p}_j \tag{17}$$

$$\theta_n^{i)} = \left(\mathbf{p}_n^{i)}\right)^{\mathrm{T}}\mathbf{y} \Big/ \left(\left(\mathbf{p}_n^{i)}\right)^{\mathrm{T}}\mathbf{p}_n^{i)} + \lambda\right). \tag{18}$$

2) For $1 \le i \le P_S$, calculate the LOO cost for each $\mathbf{u}_i^{[l]}$

$$e_k^{(n)}(i) = e_k^{(n-1)} - p_n^{i)}(k)\theta_n^{i)}, \qquad 1 \le k \le N \tag{19}$$

$$\eta_k^{(n)}(i) = \eta_k^{(n-1)} - \frac{\left(p_n^{i)}(k)\right)^2}{\left(\mathbf{p}_n^{i)}\right)^{\mathrm{T}} \mathbf{p}_n^{i)} + \lambda}, \qquad 1 \le k \le N \tag{20}$$

$$J_n^{i)} = \frac{1}{N}\sum_{k=1}^{N}\left(\frac{e_k^{(n)}(i)}{\eta_k^{(n)}(i)}\right)^2 \tag{21}$$

where $p_n^{i)}(k)$ is the kth element of $\mathbf{p}_n^{i)}$.

Inner loop: weighted boosting search For $(t = 1; t \le M_I; t = t + 1)$ {

Step 1: Boosting

1) Find

$$i_{\text{best}} = \arg \min_{1 \le i \le P_S} J_n^{i)}$$

$$i_{\text{worst}} = \arg \max_{1 \le i \le P_S} J_n^{i)}.$$

Denote $\mathbf{u}_{\text{best}}^{[l]} = \mathbf{u}_{i_{\text{best}}}^{[l]}$ and $\mathbf{u}_{\text{worst}}^{[l]} = \mathbf{u}_{i_{\text{worst}}}^{[l]}$.

2) Normalize the cost function values

$$\bar{J}_n^{i)} = \frac{J_n^{i)}}{\sum_{j=1}^{P_S} J_n^{j)}}, \qquad 1 \le i \le P_S.$$

3) Compute a weighting factor β_t according to

$$\xi_t = \sum_{i=1}^{P_S} \delta_i(t-1)\bar{J}_n^{i)} \qquad \beta_t = \frac{\xi_t}{1-\xi_t}.$$

4) Update the distribution weightings for $1 \le i \le P_S$

$$\delta_i(t) = \begin{cases} \delta_i(t-1)\beta_t^{\bar{J}_n^{i)}}, & \text{for } \beta_t \le 1 \\ \delta_i(t-1)\beta_t^{1-\bar{J}_n^{i)}}, & \text{for } \beta_t > 1 \end{cases}$$

and normalize them

$$\delta_i(t) = \frac{\delta_i(t)}{\sum_{j=1}^{P_S}\delta_j(t)}, \qquad 1 \le i \le P_S.$$

Step 2: Parameter updating

1) Construct the $(P_S + 1)$th point using

$$\mathbf{u}_{P_S+1} = \sum_{i=1}^{P_S} \delta_i(t)\mathbf{u}_i^{[l]}.$$

2) Construct the $(P_S + 2)$th point using

$$\mathbf{u}_{P_S+2} = \mathbf{u}_{\text{best}}^{[l]} + \left(\mathbf{u}_{\text{best}}^{[l]} - \mathbf{u}_{P_S+1}\right).$$

3) Calculate $\mathbf{g}_n^{P_S+1)}$ and $\mathbf{g}_n^{P_S+2)}$ from $\mathbf{u}_{P_S+1}$ and $\mathbf{u}_{P_S+2}$, orthogonalize these two candidate model columns (as in (16), (17) and (18)), and compute their corresponding LOO cost function values $J_n^{i)}$, $i = P_S + 1, P_S + 2$ (as in (19), (20) and (21)). Then, find

$$i_* = \arg \min_{i=P_S+1, P_S+2} J_n^{i)}.$$

$(\mathbf{u}_{i_*}, J_n^{i_*)})$ then replaces $(\mathbf{u}_{\text{worst}}^{[l]}, J_n^{i_{\text{worst}})})$ in the population

} **End of inner loop** The solution found in the lth generation is $\mathbf{u} = \mathbf{u}_{\text{best}}^{[l]}$.

} **End of outer loop** This yields the solution $\mathbf{u} = \mathbf{u}_{\text{best}}^{[N_G]}$, i.e., $\boldsymbol{\mu}_n$ and $\boldsymbol{\Sigma}_n$ of the nth RBF node, the nth model column $\mathbf{g}_n$, the orthogonalization coefficients $\alpha_{j,n}, 1 \le j < n$, the corresponding orthogonal model column $\mathbf{p}_n$, and the weight θ_n, as well as the n-term modeling errors $e_k^{(n)}$ and associated LOO modeling error weightings $\eta_k^{(n)}$ for $1 \le k \le N$.

The appropriate values of P_S, N_G, and M_I depend on the dimension of $\mathbf{u}$ and how hard is the objective function to be optimized. Generally, these algorithmic parameters have to be found empirically, and some general rules are discussed in [41]. For example, in the inner optimization loop, there is no need for every member of the population to converge to a (local) minimum, and it is sufficient to locate where the minimum lies. Thus, the maximum number of iterations M_I for the inner optimization loop can be set to a relatively small value. This makes the search efficient, achieving convergence with a small number of the cost function evaluations. The population size P_S and the number of generations N_G should be set sufficiently large so that the parameter space will be sampled sufficiently.

C. Computational Complexity Comparison

The computational requirements of the proposed OFS-LOO algorithm can be characterized by the number of the LOO cost function evaluations and associated model column orthogonalizations. This number can readily be shown to be

$$\begin{aligned} \text{Comp(OFS-LOO)} &= M\left(N_G(P_S + 2M_I) - (N_G - 1)\right) \\ &\approx MN_G(P_S + 2M_I) \end{aligned} \tag{22}$$

where M is the constructed model size, P_s is the population size, N_G is the number of generations, and M_I is the number of weighted boosting search iterations. The number of LOO cost evaluations and associated model column orthogonalizations for the LROLS-LOO algorithm with a given RBF variance, on the other hand, is given by

$$\text{Comp(LROLS-LOO)} = \sum_{i=1}^{M'}(N - (i-1)) \approx M'N \tag{23}$$

where the approximation is arrived because the selected model size M' is usually much smaller than the training data size N.

Typically, Comp(OFS-LOO) $\gg$ Comp(LROLS-LOO). For instance, for the Boston Housing regression example considered in Section IV-A, on average, we have Comp(OFS-LOO) $=$ 162 085 while Comp(LROLS-LOO) $=$ 26 904. However, the

complexity (23) is for a given RBF variance. This hyperparameter has to be determined by a line search based on cross-validation. Let us make an optimistic assumption that, at each point of the line search, the algorithm produces the same model size M'. The true complexity of the LROLS-LOO algorithm will be $L_S \times \text{Comp(LROLS-LOO)}$, where L_S is the total points of the line search. Again, consider the Boston Housing example. If the line search needs to be performed for $L_S \geq 8$, the total cost of the LROLS-LOO algorithm will be more than that of the proposed OFS-LOO algorithm.

In the regression modeling experiments presented in Section IV-A, we also applied the ε-SVM algorithm [34] as a benchmark, and we employed a standard quadratic optimizer to solve the optimization task of the SVM learning, which has a computational complexity much higher than the other two algorithms, the LROLS-LOO and OFS-LOO. Our experimental records show that, for the Boston Housing regression example, the run time for the OFS-LOO algorithm was 50 times faster than the standard SVM. One could argue that a fast implementation of the SVM learning could be adopted, which would reduce the computational run time. However, the need to perform a 3-D grid search to determine the learning hyperparameters (kernel variance, regularization parameter, and error band parameter) makes the SVM uncompetitive in terms of the total cost, in comparison with the other two algorithms.

III. Classification Using the Tunable RBF Network

Consider the two-class classification problem with a given training data set $\{(\mathbf{x}_k, y_k)\}_{k=1}^N$, where $\mathbf{x}_k$ is an m-dimensional pattern vector and $y_k \in \{\pm 1\}$ is the class label for $\mathbf{x}_k$. The data set is used to construct the RBF classifier of the form

$$\tilde{y}_k = \text{sgn}(\hat{y}_k) \text{ with } \hat{y}_k = f_{\text{RBF}}^{(M)}(\mathbf{x}_k) = \sum_{i=1}^{M} w_i g_i(\mathbf{x}_k) \quad (24)$$

where $\tilde{y}_k$ is the estimated class label for $\mathbf{x}_k$, $f_{\text{RBF}}^{(M)}(\bullet)$ denotes the RBF classifier with M RBF units, and

$$\text{sgn}(y) = \begin{cases} -1, & y \leq 0 \\ +1, & y > 0. \end{cases} \quad (25)$$

If we define the modeling residual or error as $e_k = y_k - \hat{y}_k$, all the notations of the RBF network for regression, defined in (1)–(8) of Section II, carry over to the present classification application. The goal of a classifier is to minimize the misclassification or error rate. Define the signed decision variable

$$s_k = \text{sgn}(y_k)\hat{y}_k = y_k \hat{y}_k = y_k f_{\text{RBF}}^{(M)}(\mathbf{x}_k). \quad (26)$$

Then, the misclassification rate over $\{(\mathbf{x}_k, y_k)\}_{k=1}^N$ is evaluated as

$$\mathcal{M}_r = \frac{1}{N} \sum_{k=1}^{N} \mathcal{I}_d(s_k) \quad (27)$$

where the indication function $\mathcal{I}_d$ is defined by

$$\mathcal{I}_d(y) = \begin{cases} 1, & y \leq 0 \\ 0, & y > 0. \end{cases} \quad (28)$$

The test error rate over a data set not used in training measures how good a classifier's generalization capability is.

A. OFS Based on the LOO Misclassification Rate

Let us denote the n-unit RBF classifier, identified using the entire training data set $\{(\mathbf{x}_k, y_k)\}_{k=1}^N$, as $f_{\text{RBF}}^{(n)}(\bullet)$. The kth modeling error for this RBF classifier is given by

$$e_k^{(n)} = y_k - f_{\text{RBF}}^{(n)}(\mathbf{x}_k) = y_k - \hat{y}_k^{(n)}. \quad (29)$$

Let $f_{\text{RBF}}^{(n,-k)}(\bullet)$ be the n-unit RBF classifier identified using the data set $\{(\mathbf{x}_k, y_k)\}_{k=1}^N$ but with its kth data point being removed. The test output of this n-unit RBF classifier at the kth data point not used in training is computed by

$$\hat{y}_k^{(n,-k)} = f_{\text{RBF}}^{(n,-k)}(\mathbf{x}_k). \quad (30)$$

The associated LOO signed decision variable is then defined by

$$s_k^{(n,-k)} = y_k \hat{y}_k^{(n,-k)} \quad (31)$$

and the LOO misclassification rate is computed by

$$J_n = \frac{1}{N} \sum_{k=1}^{N} \mathcal{I}_d\left(s_k^{(n,-k)}\right). \quad (32)$$

J_n is a measure of the classifier's generalization capability, and it an be calculated efficiently owing to the orthogonal decomposition [40]. From the LOO modeling error (9), we have (also see [32])

$$y_k - \hat{y}_k^{(n,-k)} = \frac{y_k - \hat{y}_k^{(n)}}{1 - \sum_{i=1}^{n} \frac{p_i^2(k)}{\mathbf{p}_i^{\text{T}}\mathbf{p}_i + \lambda}}. \quad (33)$$

Multiplying both sides of (33) with y_k and applying $y_k^2 = 1$ yields

$$1 - s_k^{(n,-k)} = \frac{1 - y_k \hat{y}_k^{(n)}}{1 - \sum_{i=1}^{n} \frac{p_i^2(k)}{\mathbf{p}_i^{\text{T}}\mathbf{p}_i + \lambda}} \quad (34)$$

i.e.,

$$s_k^{(n,-k)} = \frac{\sum_{i=1}^{n} y_k \theta_i p_i(k) - \sum_{i=1}^{n} \frac{p_i^2(k)}{\mathbf{p}_i^{\text{T}}\mathbf{p}_i + \lambda}}{1 - \sum_{i=1}^{n} \frac{p_i^2(k)}{\mathbf{p}_i^{\text{T}}\mathbf{p}_i + \lambda}} = \frac{\phi_k^{(n)}}{\eta_k^{(n)}}. \quad (35)$$

The recursive formula for $\eta_k^{(n)}$ is given in (12), while $\phi_k^{(n)}$ can be represented using the following recursive formula:

$$\phi_k^{(n)} = \phi_k^{(n-1)} + y_k \theta_n p_n(k) - \frac{p_n^2(k)}{\mathbf{p}_n^{\text{T}}\mathbf{p}_n + \lambda}. \quad (36)$$

Just as in regression, the proposed OFS-LOO algorithm constructs the RBF units of the classifier one by one by minimizing the LOO misclassification rate (32). Specifically, at the nth stage of the construction, the nth RBF node is determined by minimizing J_n with respect to the node's center vector $\boldsymbol{\mu}_n$ and diagonal covariance matrix $\boldsymbol{\Sigma}_n$. The construction procedure is automatically terminated when $J_M \leq J_{M+1}$, yielding an M-term RBF network classifier.

B. Determining a Unit of the RBF Classifier

The nth stage of the classifier construction is to determine the nth RBF unit by minimizing the LOO misclassification rate with respect to the node's center vector $\boldsymbol{\mu}_n$ and diagonal covariance matrix $\boldsymbol{\Sigma}_n$. This optimization task is carried out by the same global search algorithm, the RWBS, presented in Section II-B with some small modifications. Specifically, the initialization (15) is replaced by

$$\phi_k^{(0)} = 0 \quad \text{and} \quad \eta_k^{(0)} = 1,\ 1 \leq k \leq N, \text{ and } \quad J_0 = 1 \tag{37}$$

while the calculation of the LOO cost function value for each $\mathbf{u}_i^{[l]}$ (19)–(21) is replaced by

$$\phi_k^{(n)}(i) = \phi_k^{(n-1)} + y_k p_n^{i)}(k)\theta_n^{i)} - \frac{\left(p_n^{i)}(k)\right)^2}{\left(\mathbf{p}_n^{i)}\right)^{\mathrm{T}} \mathbf{p}_n^{i)} + \lambda}, \qquad 1 \leq k \leq N \tag{38}$$

$$\eta_k^{(n)}(i) = \eta_k^{(n-1)} - \frac{\left(p_n^{i)}(k)\right)^2}{\left(\mathbf{p}_n^{i)}\right)^{\mathrm{T}} \mathbf{p}_n^{i)} + \lambda}, \qquad 1 \leq k \leq N, \tag{39}$$

$$J_n^{i)} = \frac{1}{N} \sum_{k=1}^{N} \mathcal{I}_d \left(\frac{\phi_k^{(n)}(i)}{\eta_k^{(n)}(i)} \right). \tag{40}$$

At the end of the outer loop, the algorithm yields the solution $\mathbf{u} = \mathbf{u}_{\text{best}}^{[N_G]}$, i.e., $\boldsymbol{\mu}_n$ and $\boldsymbol{\Sigma}_n$ of the nth RBF node, the nth model column $\mathbf{g}_n$, the orthogonalization coefficient $\alpha_{j,n}$, $1 \leq j < n$, the corresponding orthogonal model column $\mathbf{p}_n$, and the weight θ_n, as well as $\phi_k^{(n)}$ and $\eta_k^{(n)}$ for $1 \leq k \leq N$.

IV. Experimental Results

Two regression examples and three two-class classification data sets were used to compare the performance of our OFS-LOO algorithm with several existing algorithms for constructing RBF models.

A. Regression Modeling Examples

Engine Data: This example constructed an RBF network model representing the relationship between the fuel rack position (input u_k) and the engine speed (output y_k) for a Leyland TL11 turbocharged direct-injection diesel engine. The data set, shown in Fig. 1, contained 410 samples. The first 210 data points were used in modeling and the last 200 points in model validation. The study [49] has shown that this data set can be modeled adequately as

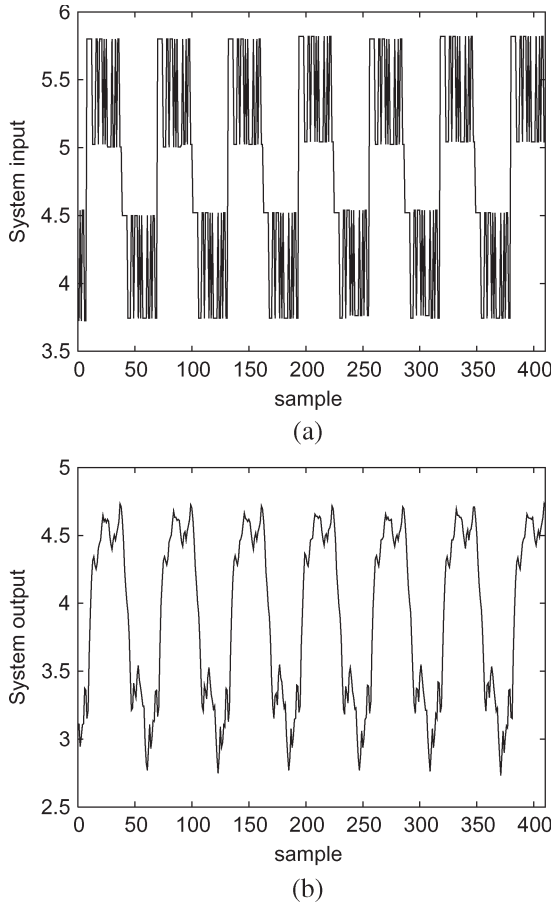

Fig. 1. Engine data set: (a) System input u_k and (b) system output y_k.

TABLE I
COMPARISON OF THE THREE GAUSSIAN RBF NETWORK MODELS OBTAINED BY THE ε-SVM, LROLS-LOO, AND PROPOSED OFS-LOO ALGORITHMS FOR THE ENGINE DATA SET

algorithm	RBF type	model size	training MSE	test MSE
ε-SVM	fixed	92	0.000447	0.000498
LROLS-LOO	fixed	22	0.000453	0.000490
OFS-LOO	**tunable**	**15**	**0.000466**	**0.000480**

$$y_k = f_s(\mathbf{x}_k) + e_k \tag{41}$$

where $f_s(\bullet)$ describes the unknown underlying system to be identified, e_k denotes the system noise, and $\mathbf{x}_k = [y_{k-1} \quad u_{k-1} \quad u_{k-2}]^{\mathrm{T}}$. Two Gaussian RBF models obtained by the ε-SVM [34] and LROLS-LOO [32] algorithms are listed in Table I. These two construction algorithms all place the RBF centers in the training input data points and use a common basis variance for every RBF node. For the LROLS-LOO algorithm, the single RBF variance was optimized via cross-validation. For the SVM algorithm, in addition to the RBF variance, two more hyperparameters, the regularization parameter C and error band ε, were also optimized via cross-validation.

We applied the proposed OFS-LOO technique to this data set, and Fig. 2 shows the LOO mse as a function of the model size during the modeling process. It is seen that the algorithm

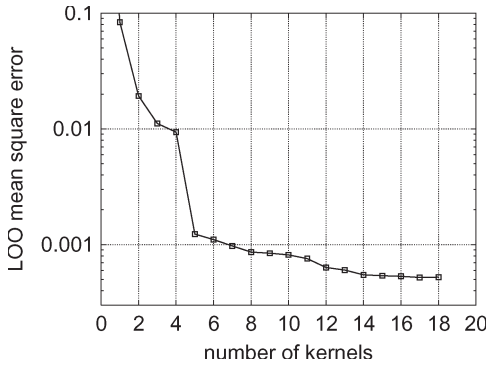

Fig. 2. Evolution of the LOO mse versus the model size for the engine data set using the OFS-LOO algorithm.

automatically constructed a 17-term RBF model, since $J_{18} > J_{17}$. The LROLS-LOO algorithm was then employed to further simplify this 17-unit RBF model, yielding a final 15-term RBF network. It was found empirically that setting the algorithmic parameters to $P_S = 37$, $M_I = 100$, and $N_G = 11$ was sufficient, and we noticed that the performance of the algorithm was not overly sensitive to the values of P_S, M_I, and N_G. This 15-term RBF network model is also listed in Table I. It can be seen that all the three models have the same excellent generalization capability, as indicated by their test mse values, but the model produced by the proposed OFS-LOO algorithm has the smallest model size, containing only 15 RBF units. The 15-unit RBF network model constructed by the OFS-LOO algorithm was used to generate the model prediction according to

$$\hat{y}_k = \hat{f}_{\mathrm{RBF}}(\mathbf{x}_k) \tag{42}$$

where $\hat{f}_{\mathrm{RBF}}(\bullet)$ denotes the constructed RBF model mapping. Fig. 3 shows the model prediction $\hat{y}_k$ and the prediction error $\hat{e}_k = y_k - \hat{y}_k$ generated by this 15-unit RBF model.

Boston Housing Data: This is a regression benchmark data set, available at the University of California, Irvine (UCI) repository [50]. The data set comprises 506 data points with 14 variables. We performed the task of predicting the median house value from the remaining 13 attributes. We randomly selected 456 data points from the data set for training and used the remaining 50 data points to form the test set. Average results were given over 100 repetitions. The RBF network with the Gaussian basis function was used, and three construction algorithms, the ε-SVM [34], the LROLS-LOO [32], and the proposed OFS-LOO, were compared. The first two algorithms place the RBF center vectors at the training input data points and employ a common RBF variance for every RBF unit, while the OFS-LOO algorithm is designed for the RBF network with tunable nodes. The RBF variance, regularization parameter, and error band of the ε-SVM algorithm were determined via cross-validation. Similarly, the RBF variance of the LROLS-LOO algorithm was optimized via cross-validation. The three optimization algorithmic parameters of the OFS-LOO algorithm were chosen empirically to be $P_S = 21$, $M_I = 200$, and $N_G = 11$. The performances of the three algorithms over the 100 realizations of the data set are compared in Table II.

The recorded average run time given hyperparameters for the LROLS-LOO method was 4 times faster than that of the

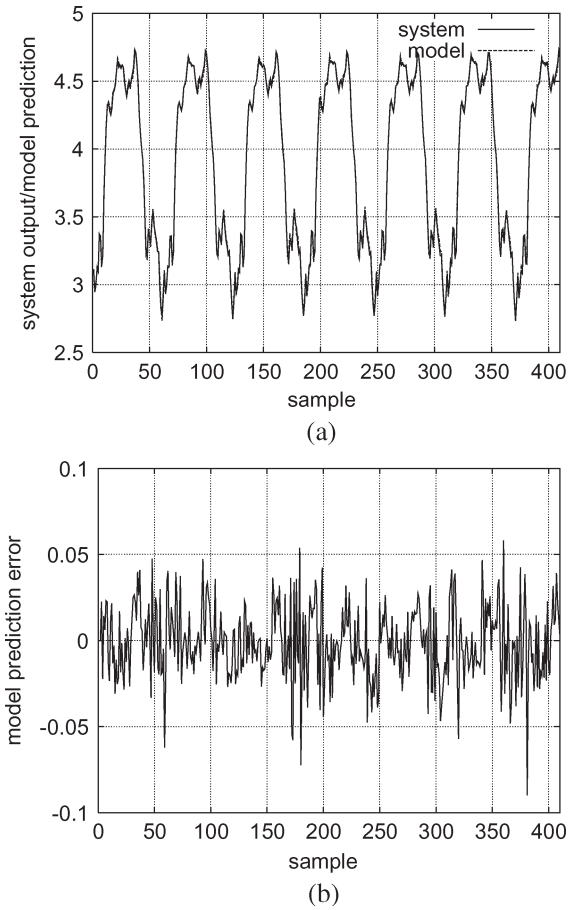

Fig. 3. Modeling of the engine data set by the 15-unit RBF network constructed using the OFS-LOO algorithm: (a) Model prediction $\hat{y}_k$ superimposed on system output y_k and (b) model prediction error $\hat{e}_k = y_k - \hat{y}_k$.

OFS-LOO algorithm and 200 times faster than the SVM algorithm. Thus, without counting the hyperparameter tuning, the LROLS-LOO algorithm is the fastest while the SVM algorithm is the slowest. By adopting the fast implementation of the SVM algorithm, significant reduction in run time may be achieved. It can be seen from Table II that the best modeling result was obtained by the proposed OFS-LOO algorithm. It can also be seen that the test mse of the SVM was poor. This was probably because the three learning hyperparemeters were not tuned to the optimal values. For this regression problem of input dimension of 13 and data size $N \approx 500$, the 3-D grid search required by the SVM was expensive, and the optimal hyperparameters were hard to find, compared with the smaller engine data set.

B. Classification Examples

Synthetic Data: This synthetic two-class problem was taken from [51], and we obtained the data set form [52]. The dimension of the feature space was $m = 2$. The training set contained 250 samples, and the test set had 1000 points. The optimal Bayes error rate for this example is known to be 8%. With the population size $P_S = 7$, the number of inner-loop iterations $M_I = 400$, and the number of generations $N_G = 11$, we applied the OFS-LOO algorithm to the 250-sample training set to construct the Gaussian RBF classifier with tunable nodes,

TABLE II
COMPARISON OF THE THREE GAUSSIAN RBF NETWORK MODELS OBTAINED BY THE ε-SVM, LROLS-LOO, AND OFS-LOO ALGORITHMS FOR THE BOSTON HOUSING DATA SET. THE RESULTS WERE AVERAGED OVER 100 REALIZATIONS AND QUOTED AS THE MEAN $\pm$ STANDARD DEVIATION

algorithm	RBF type	model size	training MSE	test MSE
ε-SVM	fixed	243.2 ± 5.3	6.7986 ± 0.4444	23.1750 ± 9.0459
LROLS-LOO	fixed	58.6 ± 11.3	12.9690 ± 2.6628	17.4157 ± 4.6670
OFS-LOO	**tunable**	$\mathbf{34.6 \pm 8.4}$	$\mathbf{10.0997 \pm 3.4047}$	$\mathbf{14.0745 \pm 3.6178}$

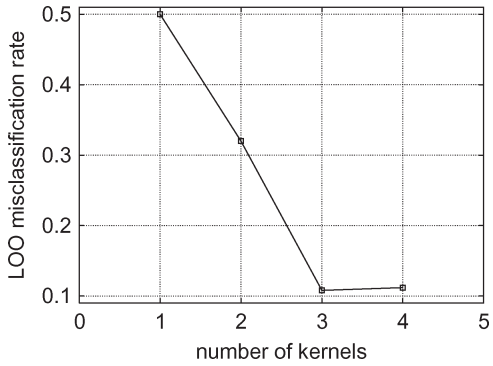

Fig. 4. Evolution of the LOO misclassification rate versus the classifier size for the synthetic data set using the OFS-LOO algorithm.

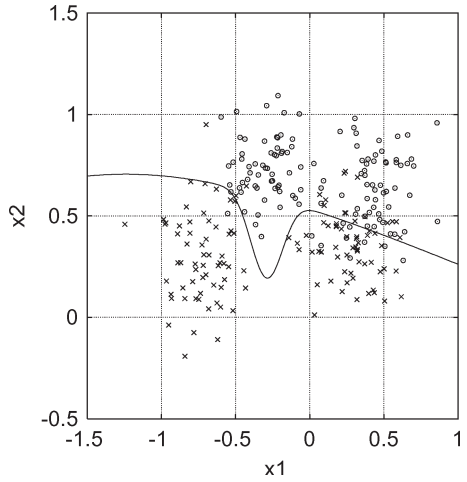

Fig. 5. Decision boundary of the three-unit RBF classifier obtained by the OFS-LOO algorithm for the synthetic data set. The 250 samples of the training data are shown as crosses and circles for the two classes.

and Fig. 4 shows the evolution of the LOO misclassification rate during the model construction. It is seen from Fig. 4 that the algorithm automatically constructed a three-unit RBF classifier, since $J_4 > J_3$. The decision boundary of this constructed three-unit Gaussian RBF network is shown in Fig. 5. The SVM and RVM algorithms were applied to construct Gaussian RBF networks for this data set in [36]. The results given in [36] are compared with our result in Table III. We also applied the LROLS-LOO classification algorithm [40] to this data set, and the result obtained is also listed in Table III. It can be seen from Table III that our proposed method not only had the sparsest model containing three RBF units but also achieved the optimal Bayes classification error rate of 8%.

Breast Cancer Data: This data set was originated in the UCI repository [50], and we obtained the data from [53]. The input dimension was $m = 9$. There were 100 realizations of this data set, each containing 200 training patterns and 77 test patterns.

TABLE III
COMPARISON OF THE FOUR GAUSSIAN RBF NETWORK CLASSIFIERS OBTAINED BY THE SVM, RVM, LROLS-LOO, AND OFS-LOO ALGORITHMS FOR THE SYNTHETIC DATA SET. THE RESULTS FOR THE SVM AND RVM ALGORITHMS ARE QUOTED FROM [36]

algorithm	RBF type	model size	test error rate
SVM	fixed	38	10.6%
RVM	fixed	4	9.3%
LROLS-LOO	fixed	5	9.0%
OFS-LOO	**tunable**	**3**	**8.0%**

TABLE IV
AVERAGE CLASSIFICATION TEST ERROR RATE IN PERCENTAGE OVER THE 100 REALIZATIONS OF THE BREAST CANCER DATA SET. THE FIRST SEVEN RESULTS WERE QUOTED FROM [53]

method	test error rate	model size
RBF-Network	27.64 ± 4.71	5
AdaBoost with RBF-Network	30.36 ± 4.73	5
LP-Reg-AdaBoost (-"-)	26.79 ± 6.08	5
QP-Reg-AdaBoost (-"-)	25.91 ± 4.61	5
AdaBoost-Reg (-"-)	26.51 ± 4.47	5
SVM with RBF-Kernel	26.04 ± 4.74	not available
Kernel Fisher Discriminant	24.77 ± 4.63	200
LROLS-LOO	25.74 ± 5.00	6.0 ± 2.0
Proposed OFS-LOO	$\mathbf{24.49 \pm 3.28}$	$\mathbf{3.1 \pm 1.2}$

In [53] and [54], seven RBF classifier construction algorithms were compared, and the performance averaged over all the 100 realizations were given. We applied the LROLS-LOO and OFS-LOO algorithms to the 100 realizations of the data set, and our results are given in Table IV, in comparison with the benchmark results quoted from [53]. For the first five methods studied in [53], the Gaussian RBF network with five optimized units was used. For the SVM with Gaussian kernel, no average model size was given in [53] but it could safely be assumed that it was larger than 50. The kernel Fisher discriminant was the nonsparse optimal classifier using all the $N = 200$ training data samples. From Table IV, it is seen that our OFS-LOO algorithm compares favorably with other benchmark RBF classifiers, both in terms of classification accuracy and model size.

Diabetes Data: This was a benchmark data set in the UCI repository [50], and we obtained the data set from [53]. The feature space dimension was $m = 8$. There were 100 realizations of the data set, each having 468 training patterns and 300 test patterns. Seven RBF classifiers were studied in [53] and [54], and the results of [53] were reproduced in Table V. For the first five methods studied in [53], the Gaussian RBF network with 15 optimized units was used. For the SVM with RBF kernel, no average model size was given in [53] but we could safely assume that it was larger than 100. We applied our OFS-LOO algorithm and the LROLS-LOO algorithm to construct the Gaussian RBF classifiers to this data set, and our results are also listed in Table V. It can be seen that our method produced the best classification accuracy with the smallest RBF classifier.

TABLE V
AVERAGE CLASSIFICATION TEST ERROR RATE IN PERCENTAGE OVER THE 100 REALIZATIONS OF THE DIABETES DATA SET. THE FIRST SEVEN RESULTS WERE QUOTED FROM [53]

method	test error rate	model size
RBF-Network	24.29 ± 1.88	15
AdaBoost with RBF-Network	26.47 ± 2.29	15
LP-Reg-AdaBoost (-"-)	24.11 ± 1.90	15
QP-Reg-AdaBoost (-"-)	25.39 ± 2.20	15
AdaBoost-Reg (-"-)	23.79 ± 1.80	15
SVM with RBF-Kernel	23.53 ± 1.73	not available
Kernel Fisher Discriminant	23.21 ± 1.63	468
LROLS-LOO	23.00 ± 1.70	6.0 ± 1.0
Proposed OFS-LOO	$\mathbf{22.16 \pm 1.47}$	$\mathbf{4.0 \pm 1.6}$

V. CONCLUSION

A novel construction algorithm has been proposed for RBF networks with tunable nodes. Unlike most of the sparse RBF or kernel modeling methods, the RBF centers are not restricted to the training input data points, and each RBF node has an individually adjusted diagonal covariance matrix. On the other hand, we do not attempt to optimize all the RBF network's parameters together using nonlinear optimization. Rather, we optimize the RBF units one by one by minimizing the LOO statistics, which is a measure of the model generalization capability. The proposed RBF network construction algorithm can be applied to both regression and classification. The RBF units are selected in a computationally efficient OFS procedure, and the orthogonal decomposition ensures a fast updating of the LOO criterion. Moreover, the RBF network construction is fully automatic, and the user does not need to specify any additional termination criterion. Our proposed method is computationally attractive, since it does not have any hyperparameter that requires costly tuning based on cross-validation. Several examples taken from both regression and classification applications have been used in our simulation experiment, and the results obtained have demonstrated that the proposed RBF network construction algorithm compares favorably with several existing benchmark RBF network construction algorithms.

REFERENCES

[1] S. Chen, S. A. Billings, C. F. N. Cowan, and P. M. Grant, "Non-linear systems identification using radial basis functions," *Int. J. Syst. Sci.*, vol. 21, pp. 2513–2539, 1990.

[2] J. A. Leonard and M. A. Kramer, "Radial basis function networks for classifying process faults," *IEEE Control Syst. Mag.*, vol. 11, no. 3, pp. 31–38, Apr. 1991.

[3] S. Chen, B. Mulgrew, and P. M. Grant, "A clustering technique for digital communications channel equalization using radial basis function networks," *IEEE Trans. Neural Netw.*, vol. 4, no. 4, pp. 570–579, Jul. 1993.

[4] A. Caiti and T. Parisini, "Mapping ocean sediments by RBF networks," *IEEE J. Ocean. Eng.*, vol. 19, no. 4, pp. 577–582, Oct. 1994.

[5] D. Gorinevsky, A. Kapitanovsky, and A. Goldenberg, "Radial basis function network architecture for nonholonomic motion planning and control of free-flying manipulators," *IEEE Trans. Robot. Autom.*, vol. 12, no. 3, pp. 491–496, Jun. 1996.

[6] M. Rosenblum and L. S. Davis, "An improved radial basis function network for visual autonomous road following," *IEEE Trans. Neural Netw.*, vol. 7, no. 5, pp. 1111–1120, Sep. 1996.

[7] J. A. Refaee, M. Mohandes, and H. Maghrabi, "Radial basis function networks for contingency analysis of bulk power systems," *IEEE Trans. Power Syst.*, vol. 14, no. 2, pp. 772–778, May 1999.

[8] R. Mukai, V. A. Vilnrotter, P. Arabshahi, and V. Jamnejad, "Adaptive acquisition and tracking for deep space array feed antennas," *IEEE Trans. Neural Netw.*, vol. 13, no. 5, pp. 1149–1162, Sep. 2002.

[9] C.-T. Su, T. Yang, and C.-M. Ke, "A neural-network approach for semiconductor wafer post-sawing inspection," *IEEE Trans. Semicond. Manuf.*, vol. 15, no. 2, pp. 260–266, May 2002.

[10] Y. Li, N. Sundararajan, P. Saratchandran, and Z. Wang, "Robust neuro-H_∞ controller design for aircraft auto-landing," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 40, no. 1, pp. 158–167, Jan. 2004.

[11] M.-J. Lee and Y.-K. Choi, "An adaptive neurocontroller using RBFN for robot manipulators," *IEEE Trans. Ind. Electron.*, vol. 51, no. 3, pp. 711–717, Jun. 2004.

[12] S. X. Ng, M.-S. Yee, and L. Hanzo, "Coded modulation assisted radial basis function aided turbo equalization for dispersive Rayleigh-fading channels," *IEEE Trans. Wireless Commun.*, vol. 3, no. 6, pp. 2198–2206, Nov. 2004.

[13] N. Acir, I. Oztura, M. Kuntalp, B. Baklan, and C. Guzelis, "Automatic detection of epileptiform events in EEG by a three-stage procedure based on artificial neural networks," *IEEE Trans. Biomed. Eng.*, vol. 52, no. 1, pp. 30–40, Jan. 2005.

[14] K. K. Tan, S. Zhao, and S. Huang, "Iterative reference adjustment for high-precision and repetitive motion control applications," *IEEE Trans. Control Syst. Technol.*, vol. 13, no. 1, pp. 85–97, Jan. 2005.

[15] S. Chen, C. F. N. Cowan, S. A. Billings, and P. M. Grant, "Parallel recursive prediction error algorithm for training layered neural networks," *Int. J. Control*, vol. 51, no. 6, pp. 1215–1228, 1990.

[16] P. E. An, M. Brown, S. Chen, and C. J. Harris, "Comparative aspects of neural network algorithms for on-line modeling of dynamic processes," *Proc. Inst. Mech. Eng., Pt. I*, vol. 207, pp. 223–241, 1993.

[17] S. McLoone, M. D. Brown, G. Irwin, and A. Lightbody, "A hybrid linear/nonlinear training algorithm for feedforward neural networks," *IEEE Trans. Neural Netw.*, vol. 9, no. 4, pp. 669–684, Jul. 1998.

[18] H. Peng, T. Ozaki, V. Haggan-Ozaki, and Y. Toyoda, "A parameter optimization method for radial basis function type models," *IEEE Trans. Neural Netw.*, vol. 14, no. 2, pp. 432–438, Mar. 2003.

[19] B. A. Whitehead and T. D. Choate, "Evolving space-filling curves to distribute radial basis functions over an input space," *IEEE Trans. Neural Netw.*, vol. 5, no. 1, pp. 15–23, Jan. 1994.

[20] B. A. Whitehead, "Genetic evolution of radial basis function coverage using orthogonal niches," *IEEE Trans. Neural Netw.*, vol. 7, no. 6, pp. 1525–1528, Nov. 1996.

[21] J. Gonzalez, I. Rojas, J. Ortega, H. Pomares, F. J. Fernandez, and A. F. Diaz, "Multiobjective evolutionary optimization of the size, shape, and position parameters of radial basis function networks for function approximation," *IEEE Trans. Neural Netw.*, vol. 14, no. 6, pp. 1478–1495, Nov. 2003.

[22] Z. R. Yang and S. Chen, "Robust maximum likelihood training of heteroscedastic probabilistic neural networks," *Neural Netw.*, vol. 11, no. 4, pp. 739–747, Jun. 1998.

[23] M.-W. Mak and S.-Y. Kung, "Estimation of elliptical basis function parameters by the EM algorithm with application to speaker verification," *IEEE Trans. Neural Netw.*, vol. 11, no. 4, pp. 961–969, Jul. 2000.

[24] J. Moody and C. J. Darken, "Fast learning in networks of locally-tuned processing units," *Neural Comput.*, vol. 1, no. 2, pp. 281–294, 1989.

[25] S. Chen, S. A. Billings, and P. M. Grant, "Recursive hybrid algorithm for non-linear system identification using radial basis function networks," *Int. J. Control*, vol. 55, no. 5, pp. 1051–1070, May 1992.

[26] S. Chen, "Nonlinear time series modeling and prediction using Gaussian RBF networks with enhanced clustering and RLS learning," *Electron. Lett.*, vol. 31, no. 2, pp. 117–118, Jan. 1995.

[27] Z. Uykan, "Clustering-based algorithms for single-hidden-layer sigmoid perceptron," *IEEE Trans. Neural Netw.*, vol. 14, no. 3, pp. 708–715, May 2003.

[28] S. Chen, S. A. Billings, and W. Luo, "Orthogonal least squares methods and their application to non-linear system identification," *Int. J. Control*, vol. 50, no. 5, pp. 1873–1896, Nov. 1989.

[29] S. Chen, C. F. N. Cowan, and P. M. Grant, "Orthogonal least squares learning algorithm for radial basis function networks," *IEEE Trans. Neural Netw.*, vol. 2, no. 2, pp. 302–309, Mar. 1991.

[30] S. Chen, Y. Wu, and B. L. Luk, "Combined genetic algorithm optimization and regularized orthogonal least squares learning for radial basis function networks," *IEEE Trans. Neural Netw.*, vol. 10, no. 5, pp. 1239–1243, Sep. 1999.

[31] S. Chen, X. Hong, and C. J. Harris, "Sparse kernel regression modeling using combined locally regularized orthogonal least squares and D-optimality experimental design," *IEEE Trans. Autom. Control*, vol. 48, no. 6, pp. 1029–1036, Jun. 2003.

[32] S. Chen, X. Hong, C. J. Harris, and P. M. Sharkey, "Sparse modeling using orthogonal forward regression with PRESS statistic and regularization," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 34, no. 2, pp. 898–911, Apr. 2004.
[33] V. Vapnik, *The Nature of Statistical Learning Theory*. New York: Springer-Verlag, 1995.
[34] S. Gunn, "Support vector machines for classification and regression," ISIS Res. Group, Dept. Electron. Comput. Sci., Univ. Southampton, Southampton, U.K., May 1998. Tech. Rep..
[35] S. S. Chen, D. L. Donoho, and M. A. Saunders, "Atomic decomposition by basis pursuit," *SIAM Rev.*, vol. 43, no. 1, pp. 129–159, 2001.
[36] M. E. Tipping, "Sparse Bayesian learning and the relevance vector machine," *J. Mach. Learn. Res.*, vol. 1, pp. 211–244, Sep. 2001.
[37] B. Schölkopf and A. J. Smola, *Learning With Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. Cambridge, MA: MIT Press, 2002.
[38] P. Vincent and Y. Bengio, "Kernel matching pursuit," *Mach. Learn.*, vol. 48, no. 1, pp. 165–187, 2002.
[39] G. R. G. Lanckriet, N. Cristianini, P. Bartlett, L. E. Ghaoui, and M. I. Jordan, "Learning the kernel matrix with semidefinite programming," *J. Mach. Learn. Res.*, vol. 5, pp. 27–72, Dec. 2004.
[40] X. Hong, S. Chen, and C. J. Harris, "A fast linear-in-the-parameters classifier construction algorithm using orthogonal forward selection to minimize leave-one-out misclassification rate," *Int. J. Syst. Sci.*, vol. 39, no. 2, pp. 119–125, 2008.
[41] S. Chen, X. X. Wang, and C. J. Harris, "Experiments with repeating weighted boosting search for optimization signal processing applications," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 35, no. 4, pp. 682–693, Aug. 2005.
[42] X. Hong, P. M. Sharkey, and K. Warwick, "Automatic nonlinear predictive model-construction algorithm using forward regression and the PRESS statistic," *Proc. Inst. Elect. Eng.—Control Theory Appl.*, vol. 150, no. 3, pp. 245–254, May 2003.
[43] R. H. Myers, *Classical and Modern Regression With Applications*, 2nd ed. Boston, MA: PWS-KENT, 1990.
[44] M. Stone, "Cross-validatory choice and assessment of statistical predictions," *J. R. Stat. Soc. Ser. B*, vol. 36, no. 2, pp. 111–147, 1974.
[45] D. E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*. Reading, MA: Addison-Wesley, 1989.
[46] K. F. Man, K. S. Tang, and S. Kwong, *Genetic Algorithms: Concepts and Design*. London, U.K.: Springer-Verlag, 1998.
[47] L. Ingber, "Simulated annealing: Practice versus theory," *Math. Comput. Model.*, vol. 18, no. 11, pp. 29–57, 1993.
[48] S. Chen and B. L. Luk, "Adaptive simulated annealing for optimization in signal processing applications," *Signal Process.*, vol. 79, no. 1, pp. 117–128, Nov. 1999.
[49] S. A. Billings, S. Chen, and R. J. Backhouse, "The identification of linear and non-linear models of a turbocharged automotive diesel engine," *Mech. Syst. Signal Process.*, vol. 3, no. 2, pp. 123–142, 1989.
[50] [Online]. Available: http://www.ics.uci.edu/~mlearn/MLRepository.html
[51] B. D. Ripley, *Pattern Recognition and Neural Networks*. Cambridge, U.K.: Cambridge Univ. Press, 1996.
[52] [Online]. Available: http://www.stats.ox.ac.uk/PRNN/
[53] [Online]. Available: ida.first.fhg.de/projects/bench/benchmarks.htm
[54] G. Rätsch, T. Onoda, and K. R. Müller, "Soft margins for AdaBoost," *Mach. Learn.*, vol. 42, no. 3, pp. 287–320, Mar. 2001.

Sheng Chen received the B.Eng. degree in control engineering from Huadong Petroleum Institute, Dongying, China, in 1982, the Ph.D. degree in control engineering from the City University, London, U.K., in 1986, and the D.Sc. degree from the University of Southampton, Southampton, U.K., in 2005.

He has been with the School of Electronics and Computer Science, University of Southampton, since September 1999. He previously held research and academic appointments with the University of Sheffield, Sheffield, U.K., the University of Edinburgh, Edinburgh, U.K., and the University of Portsmouth, Portsmouth, U.K. He has published over 350 research papers. In the database of the world's most highly cited researchers, compiled by the Institute for Scientific Information of the USA, he is on the list of the highly cited researchers in the engineering category. His research interests include wireless communications, machine learning and neural networks, finite-precision digital controller design, and evolutionary computation methods.

Xia Hong received the B.Sc. and M.Sc. degrees in automatic control from the National University of Defence Technology, Changsha, China, in 1984 and 1987, respectively, and the Ph.D. degree in automatic control from the University of Sheffield, Sheffield, U.K., in 1998.

She was a Research Assistant with the Beijing Institute of Systems Engineering, Beijing, China, from 1987 to 1993. She was a Research Fellow with the Department of Electronics and Computer Science, University of Southampton, Southampton, U.K., from 1997 to 2001. She is currently a Lecturer with the School of Systems Engineering, University of Reading, Reading, U.K. She is actively engaged in research into nonlinear systems identification, data modeling, estimation and intelligent control, neural networks, pattern recognition, learning theory, and their applications. She has published over 100 research papers and coauthored a research book.

Dr. Hong was the recipient of a Donald Julius Groen Prize by IMechE in 1999.

Bing L. Luk received the B.Sc. degree in electrical and electronic engineering from Portsmouth Polytechnic, Portsmouth, U.K., in 1985, the M.Sc. degree in digital computer systems from Brunel University, Uxbridge, U.K., in 1986, and the Ph.D. degree in robotics from the University of Portsmouth, Portsmouth, in 1991.

He has been with the Department of Manufacturing Engineering and Engineering Management, City University of Hong Kong, Kowloon, Hong Kong, China, since 2000. He previously held research and academic appointments with the University of Portsmouth and engineering consultant positions with Portsmouth Technology Consultant, Ltd. and also with other industrial companies. His recent research works include mobile robotics, telemedicine research, nondestructive test methods, machine learning, and evolutionary computation methods.

Chris J. Harris received the Ph.D. and D.Sc. degrees from the University of Southampton, Southampton, U.K., in 1972 and 2001, respectively.

He previously held appointments with the University of Hull, Hull, U.K., the University of Manchester Institute of Science and Technology, Manchester, the University of Oxford, Oxford, U.K., and the University of Cranfield, Cranfield, U.K. He was also with the U.K. Ministry of Defence. He returned to the University of Southampton as the Lucas Professor of Aerospace Systems Engineering in 1987 to establish the Advanced Systems Research Group and, later, he jointed the School of Electronics and Computer Science to establish Image, Speech and Intelligent Systems Group. He has authored and coauthored 12 research books and over 400 research papers, and he is the Associate Editor of numerous international journals. His research interests lie in the general area of intelligent and adaptive systems theory and its application to intelligent autonomous systems such as autonomous vehicles, management infrastructures such as command and control, intelligent control, and estimation of dynamic processes, multisensor data fusion, and systems integration.

Dr. Harris was elected as a Fellow of the Royal Academy of Engineering in 1996 and was awarded the IEE Senior Achievement medal in 1998 for his work in autonomous systems and the highest international award in IEE, the IEE Faraday medal, in 2001 for his work in intelligent control and neurofuzzy systems.