Radial basis function classifier construction using particle swarm optimisation aided orthogonal forward regression

Radial Basis Function Classifier Construction Using Particle Swarm Optimisation Aided Orthogonal Forward Regression

S. Chen, X. Hong and C.J. Harris

Abstract— **We develop a particle swarm optimisation (PSO) aided orthogonal forward regression (OFR) approach for constructing radial basis function (RBF) classifiers with tunable nodes. At each stage of the OFR construction process, the centre vector and diagonal covariance matrix of one RBF node is determined efficiently by minimising the leave-one-out (LOO) misclassification rate (MR) using a PSO algorithm. Compared with the state-of-the-art regularisation assisted orthogonal least square algorithm based on the LOO MR for selecting fixed-node RBF classifiers, the proposed PSO aided OFR algorithm for constructing tunable-node RBF classifiers offers significant advantages in terms of better generalisation performance and smaller model size as well as imposes lower computational complexity in classifier construction process. Moreover, the proposed algorithm does not have any hyperparameter that requires costly tuning based on cross validation.**

I. Introduction

Various methods for constructing nonlinear radial basis function (RBF) classifiers can be divided into the two approaches based on nonlinear learning and linear learning, respectively. In a nonlinear learning approach, all the parameters of a RBF network, including the RBF centre vectors and variances or covariance matrices as well as the RBF weights, are learned together via nonlinear optimisation. Generally, learning based on such a nonlinear approach is computationally expensive and may encounter the problem of local minima. Additionally, the classifier structure or the number of RBF nodes has to be determined via other means, typically based on cross validation. A most popular approach for constructing RBF classifiers however is to formulate the problem as a linear learning problem by considering the training input data points as candidate RBF centres and employing a common variance for every RBF node. A parsimonious RBF classifier can be selected efficiently using the regularisation assisted orthogonal least squares (ROLS) algorithm based on the leave-one-out (LOO) misclassification rate (MR) [1], [2]. Similarly, the support vector machine (SVM) and other sparse kernel modelling methods [3]–[6] also fix the kernel centres to the training input data points and adopt a common kernel variance for every kernel. A sparse kernel classifier 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 modelling methods, additionally some learning algorithm's hyperparameters also have to be determined by cross validation. Our previous experimental results obtained in [1], [2] show that the ROLS algorithm based on the LOO-MR compares favourably with many other existing sparse kernel modelling methods for selecting fixed-node RBF classifiers, in terms of model sparsity and generalisation performance.

An alternative method for constructing RBF classifiers with tunable nodes was proposed in [7], [8], which can be viewed as combining both the linear and nonlinear learning approaches. In this novel approach, each RBF unit has a tunable centre vector as well as an adjustable diagonal covariance matrix, just as in a nonlinear learning approach. However, the algorithm does not attempt to optimise all the RBF units together, which could be a too large and complicated nonlinear optimisation task. Rather, an orthogonal forward regression (OFR) procedure is employed to optimise the RBF units one by one by minimising the LOO MR. The determination of the RBF centre vector and diagonal covariance matrix at each stage of the construction is carried out by a search algorithm known as the repeated weighted boosting search (RWBS) [9]. This construction procedure automatically determines the number of RBF units to use, and the learning algorithm does not have hyperparameter that requires tuning based on costly cross validation. Our experimental results confirm that this OFR-LOO algorithm for constructing tunable-node RBF classifiers outperforms the existing methods for selecting fixed-node RBF classifiers, in terms of sparsity and generalisation performance of the classifier. A drawback of the algorithm [7], [8] for constructing tunable-node RBF classifiers is that it may require more computation in classifier construction than the algorithm [1], [2] for selecting fixed-node RBF classifiers.

In this contribution we propose to apply the particle swarm optimisation (PSO) algorithm for adding the OFR procedure for constructing tunable-node RBF classifiers. PSO [10], [11] is a population based stochastic optimisation technique, inspired by the social behaviour of bird flocks or fish schools. The algorithm commences with a random initialisation of a swarm of individuals, referred to as particles, within the problem's search space. It then endeavours to find a globally optimum solution by gradually adjusting the trajectory of each particle towards its own best location and towards the best position of the entire swarm at each evolutionary optimisation step. The PSO method is popular owing to its simplicity in implementation, ability to rapidly converge to a "reasonably good" solution and to "steer clear" of local minima. It has been successfully applied to wide-ranging optimisation problems [12]–[17]. Because of the simplicity and

S. Chen and C.J. Harris are with School of Electronics and Computer Science, University of Southampton, Southampton SO17 1BJ, UK. E-mails: {sqc,cjh}@ecs.soton.ac.uk

X. Hong is with School of Systems Engineering, University of Reading, Reading RG6 6AY, UK. E-mail: x.hong@reading.ac.uk

efficiency of the PSO method, the proposed PSO aided OFR algorithm based on the LOO MR for constructing tunable-node RBF classifiers not only produces smaller RBF models with better generalisation performance but also require less computation in classifier construction, in comparison with the efficient ROLS-LOO algorithm of [1], [2] for selecting fixed-node RBF classifiers.

II. Construction of tunable-node RBF classifier

For the notational simplification, we restrict to the two-class case. Consider the two-class classification problem with a given training data set $D_N \triangleq \{(\mathbf{x}_k, y_k)\}_{k=1}^N$, 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 D_N is used to construct the RBF classifier of the form

$$\tilde{y}_k = \text{sgn}(\hat{y}_k) \quad \text{with} \quad \hat{y}_k = f^{[M]}(\mathbf{x}_k) = \sum_{i=1}^{M} w_i g_i(\mathbf{x}_k), \quad (1)$$

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

$$\text{sgn}(y) = \begin{cases} -1, & y \le 0, \\ +1, & y > 0. \end{cases} \quad (2)$$

Define the modelling error as $e_k = y_k - \hat{y}_k$. Then the classification model can be written in the regression form

$$y_k = \hat{y}_k + e_k = \sum_{i=1}^{M} w_i g_i(\mathbf{x}_k) + e_k = \mathbf{g}_M^T(k)\mathbf{w}_M + e_k, \quad (3)$$

where $\mathbf{w}_M = [w_1 \ w_2 \cdots w_M]^T$ is the M-unit RBF weight vector and $\mathbf{g}_M(k) = [g_1(\mathbf{x}_k) \ g_2(\mathbf{x}_k) \cdots g_M(\mathbf{x}_k)]^T$ is the corresponding regressor vector. We consider the general RBF unit of the form

$$g_i(\mathbf{x}) = K\left(\sqrt{(\mathbf{x}-\boldsymbol{\mu}_i)^T \boldsymbol{\Sigma}_i^{-1} (\mathbf{x}-\boldsymbol{\mu}_i)}\right), \quad (4)$$

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

$$\mathbf{g}_k = [g_k(\mathbf{x}_1) \ g_k(\mathbf{x}_2) \cdots g_k(\mathbf{x}_N)]^T, \ 1 \le k \le M, \quad (5)$$

the regression model (3) over the training data set can be written in the matrix form

$$\mathbf{y} = \mathbf{G}_M \mathbf{w}_M + \mathbf{e}. \quad (6)$$

Note that $\mathbf{g}_k$ denotes the kth column of $\mathbf{G}_M$ while $\mathbf{g}_M^T(k)$ is the kth row of $\mathbf{G}_M$.

Let an orthogonal decomposition of the regression matrix $\mathbf{G}_M$ be $\mathbf{G}_M = \mathbf{P}_M \mathbf{A}_M$, where $\mathbf{A}_M$ is the upper triangular matrix with unity diagonal elements

$$\mathbf{A}_M = \begin{bmatrix} 1 & \alpha_{1,2} & \cdots & \alpha_{1,M} \\ 0 & 1 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \alpha_{M-1,M} \\ 0 & \cdots & 0 & 1 \end{bmatrix} \quad (7)$$

and

$$\mathbf{P}_M = [\mathbf{p}_1 \ \mathbf{p}_2 \cdots \mathbf{p}_M] \quad (8)$$

with the orthogonal columns that satisfy $\mathbf{p}_i^T \mathbf{p}_j = 0$, if $i \ne j$. The regression model (6) can alternatively be expressed as

$$\mathbf{y} = \mathbf{P}_M \boldsymbol{\theta}_M + \mathbf{e}, \quad (9)$$

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

$$\hat{y}_k = \mathbf{p}_M^T(k) \boldsymbol{\theta}_M, \quad (10)$$

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

It is highly desirable to construct the RBF classifier (1) by directly optimising the classifier's generalisation capability. Cross validation criteria are metrics that measure a model's generalisation capability. One commonly used version of cross validation is the LOO cross validation [18], [19]. Denote the n-unit RBF classifier, identified using the entire training data set D_N, as $f^{[n]}(\bullet)$. Let $f^{[n,-k]}(\bullet)$ be the n-unit RBF classifier identified using the LOO data set $D_N \setminus (\mathbf{x}_k, y_k)$, namely, the data set D_N with its kth data point $(\mathbf{x}_k, y_k)$ being removed. The test output of this n-unit RBF classifier at the kth data point not used in training is computed by [18], [19]

$$\hat{y}_k^{[n,-k]} = f^{[n,-k]}(\mathbf{x}_k). \quad (11)$$

Define the associated LOO signed decision variable as

$$s_k^{[n,-k]} = y_k \hat{y}_k^{[n,-k]}. \quad (12)$$

Then the LOO MR for the n-term RBF classifier can be computed as

$$J_n = \frac{1}{N} \sum_{k=1}^{N} \mathcal{I}_d\left(s_k^{[n,-k]}\right), \quad (13)$$

where the indicator function is defined by

$$\mathcal{I}_d(y) = \begin{cases} 1, & y \le 0, \\ 0, & y > 0. \end{cases} \quad (14)$$

This LOO MR is a measure of the classifier's generalisation capability [18], [19].

The LOO signed decision variable $s_k^{[n,-k]}$ can be calculated rapidly owning to the orthogonal decomposition and, therefore, the LOO-MR J_n can be computed efficiently [1], [7]. Let

$$\mathbf{P}_n = [\mathbf{p}_1 \ \mathbf{p}_2 \cdots \mathbf{p}_n] \quad (15)$$

be the orthogonal regression matrix of the n-term classifier, with $\mathbf{p}_k$ denoting the kth column of $\mathbf{P}_n$ and $\mathbf{p}_n^T(k)$ the kth row of $\mathbf{P}_n$. It can be shown [1], [7]

$$s_k^{[n,-k]} = \frac{\sum_{i=1}^n y_k \, \theta_i \, p_i(k) - \sum_{i=1}^n \frac{p_i^2(k)}{\mathbf{p}_i^T \mathbf{p}_i + \lambda}}{1 - \sum_{i=1}^n \frac{p_i^2(k)}{\mathbf{p}_i^T \mathbf{p}_i + \lambda}} \triangleq \frac{\phi_k^{[n]}}{\eta_k^{[n]}}, \quad (16)$$

where λ is a small positive regularisation parameter. Thus, the LOO error weighting $\eta_k^{[n]}$ can be computing recursively using the formula

$$\eta_k^{[n]} = \eta_k^{[n-1]} - p_n^2(k)/\left(\mathbf{p}_n^T\mathbf{p}_n + \lambda\right), \tag{17}$$

while $\phi_k^{[n]}$ can be represented using the recursive formula

$$\phi_k^{[n]} = \phi_k^{[n-1]} + y_k\,\theta_n\,p_n(k) - p_n^2(k)/\left(\mathbf{p}_n^T\mathbf{p}_n + \lambda\right), \tag{18}$$

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

The OFR-LOO algorithm [7], [8] constructs the RBF units one by one by minimising the LOO-MR J_n using the RWBS algorithm of [9]. Specifically, at the nth stage of the construction procedure, the nth RBF unit is determined by minimising J_n with respect to the RBF unit's centre vector $\boldsymbol{\mu}_n$ and diagonal covariance matrix $\boldsymbol{\Sigma}_n$

$$\min_{\boldsymbol{\mu}_n,\boldsymbol{\Sigma}_n} J_n\left(\boldsymbol{\mu}_n,\boldsymbol{\Sigma}_n\right). \tag{19}$$

The construction procedure is automatically terminated when

$$J_M \le J_{M+1}, \tag{20}$$

yielding an M-term RBF classifier. Note that the LOO-MR J_n is locally convex with respect to the classifier size n, and there exists an "optimal" M such that, for $n \le M$, J_n decreases as the model size n increases, while the condition (20) holds [1], [7].

III. PSO AIDED CLASSIFIER CONSTRUCTION

Let $\mathbf{u}$ be the parameter vector that contains $\boldsymbol{\mu}_n$ and $\boldsymbol{\Sigma}_n$. The dimension of $\mathbf{u}$ is $m^{'} = 2m$. Define the cost function as $F(\mathbf{u}) = J_n(\mathbf{u})$. We propose to use the PSO algorithm to solve the optimisation task (19), namely

$$\mathbf{u}_{\rm opt} = \arg \min_{\mathbf{u}\in\prod_{j=1}^{m^{'}}\mathsf{P}_j} F(\mathbf{u}), \tag{21}$$

where the search space

$$\prod_{j=1}^{m^{'}}\mathsf{P}_j = \prod_{j=1}^{m^{'}}[P_{j,\min},\ P_{j,\max}] \tag{22}$$

is specified by

$$P_{j,\min} = \min_{1\le k\le N}\{x_{k,j}\}, P_{j,\max} = \max_{1\le k\le N}\{x_{k,j}\}, 1 \le j \le m, \tag{23}$$

$$P_{j,\min} = \sigma_{\min}^2,\ P_{j,\max} = \sigma_{\max}^2,\ m+1 \le j \le m^{'}, \tag{24}$$

with $x_{k,j}$ denoting the jth element of $\mathbf{x}_k$, $\sigma_{\min}^2$ and $\sigma_{\max}^2$ being the chosen lower and upper bounds for the RBF variances $\sigma_{n,j}^2$, $1 \le j \le m$, respectively. With the initial conditions

$$\phi_k^{[0]} = 0,\ \eta_k^{[0]} = 1,\ 1 \le k \le N, \quad \text{and} \quad J_0 = 1, \tag{25}$$

the n-th stage of the construction procedure determines the n-th RBF unit by solving the optimisation problem (21) using the PSO algorithm, whose flowchart is depicted in Fig. 1. Specifically, a swarm of particles, $\{\mathbf{u}_i^{l)}\}_{i=1}^S$, that represent potential solutions are evolved in the search space $\prod_{j=1}^{m^{'}}\mathsf{P}_j$, where S is the swarm size and index l denotes the iteration step. The algorithm is summarised as follows.

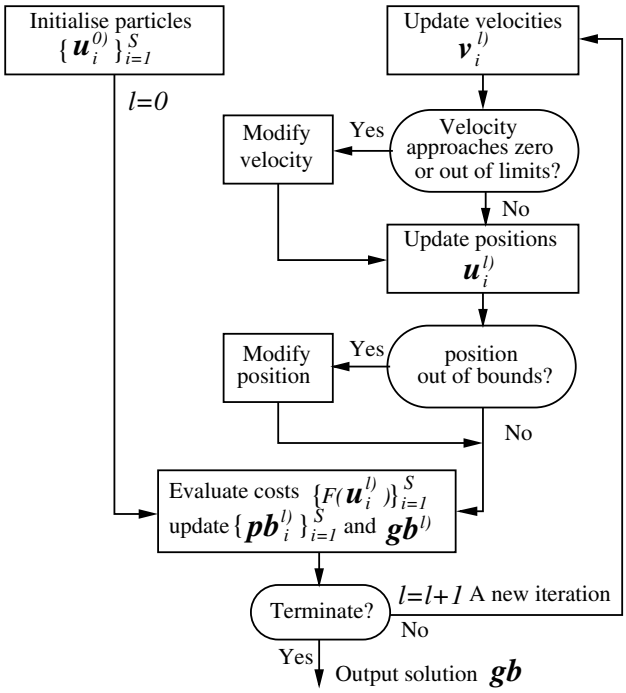

Fig. 1. Flowchart of the PSO algorithm.

A. PSO algorithm

a) The swarm initialisation. Set the iteration index $l = 0$ and randomly generate $\{\mathbf{u}_i^{l)}\}_{i=1}^S$ in the search space $\prod_{j=1}^{m^{'}}\mathsf{P}_j$.

b) The swarm evaluation. Each particle $\mathbf{u}_i^{l)}$ has a cost $F(\mathbf{u}_i^{l)})$ associated with it, which is computed as follows.

1) For $1 \le i \le S$, generate $\mathbf{g}_n^{i\}}$ from $\mathbf{u}_i^{l)}$, the candidates for the nth model column, according to (4), and orthogonalise them according to the Gram-Schmidt orthogonalisation

$$\alpha_{j,n}^{i\}} = \mathbf{p}_j^T\mathbf{g}_n^{i\}}/\mathbf{p}_j^T\mathbf{p}_j,\ 1 \le j < n, \tag{26}$$

$$\mathbf{p}_n^{i\}} = \mathbf{g}_n^{i\}} - \sum_{j=1}^{n-1}\alpha_{j,n}^{i\}}\mathbf{p}_j, \tag{27}$$

$$\theta_n^{i\}} = \left(\mathbf{p}_n^{i\}}\right)^T\mathbf{y}/\left(\left(\mathbf{p}_n^{i\}}\right)^T\mathbf{p}_n^{i\}} + \lambda\right). \tag{28}$$

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

$$\eta_k^{[n]}(i) = \eta_k^{[n-1]} - \frac{\left(p_n^{i\}}(k)\right)^2}{\left(\mathbf{p}_n^{i\}}\right)^T\mathbf{p}_n^{i\}} + \lambda}, \tag{29}$$

$$\phi_k^{[n]}(i) = \phi_k^{[n-1]} + y_k\theta_n^{i\}}p_n^{i\}}(k) - \frac{\left(p_n^{i\}}(k)\right)^2}{\left(\mathbf{p}_n^{i\}}\right)^T\mathbf{p}_n^{i\}} + \lambda}, \tag{30}$$

for $1 \le k \le N$, and

$$F(\mathbf{u}_i^{l)}) = \frac{1}{N}\sum_{k=1}^{N}\mathcal{I}_d\left(\phi_k^{[n]}(i)/\eta_k^{[n]}(i)\right). \tag{31}$$

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

Each particle $\mathbf{u}_i^{l)}$ remembers its best position visited so far, denoted as $\mathbf{pb}_i^{l)}$, which provides the cognitive information. Every particle also knows the best position visited so far among the entire swarm, denoted as $\mathbf{gb}^{l)}$, which provides the social information. The cognitive information $\{\mathbf{pb}_i^{l)}\}_{i=1}^S$ and the social information $\mathbf{gb}^{l)}$ are updated at each iteration:

For $(i = 1;\ i \leq S;\ i$++$)$
  If $(F(\mathbf{u}_i^{l)}) < F(\mathbf{pb}_i^{l)}))\quad \mathbf{pb}_i^{l)} = \mathbf{u}_i^{l)}$;
End for;
$i^* = \arg\min_{1\leq i\leq S} F(\mathbf{pb}_i^{l)})$;
If $(F(\mathbf{pb}_{i^*}^{l)}) < F(\mathbf{gb}^{l)}))\quad \mathbf{gb}^{l)} = \mathbf{pb}_{i^*}^{l)}$;

c) The swarm update. Each particle $\mathbf{u}_i^{l)}$ has a velocity, denoted as $\mathbf{v}_i^{l)}$, to direct its "flying" or search. The velocity and position of the ith particle are updated in each iteration according to

$$\begin{aligned}\mathbf{v}_i^{l+1)} &= w_{\rm I} * \mathbf{v}_i^{l)} + rand() * c_1 * (\mathbf{pb}_i^{l)} - \mathbf{u}_i^{l)}) \\ &\quad + rand() * c_2 * (\mathbf{gb}^{l)} - \mathbf{u}_i^{l)}), \qquad (32)\\ \mathbf{u}_i^{l+1)} &= \mathbf{u}_i^{l)} + \mathbf{v}_i^{l+1)}, \qquad (33)\end{aligned}$$

where $w_{\rm I}$ is the inertia weight, $rand()$ denotes the uniform random number between 0 and 1, and c_1 and c_2 are the two acceleration coefficients. In order to avoid excessive roaming of particles beyond the search space [13], a velocity space

$$\prod_{j=1}^{m'} \mathsf{V}_j = \prod_{j=1}^{m'} [-V_{j,\max},\ V_{j,\max}] \qquad (34)$$

is imposed on $\mathbf{v}_i^{l+1)}$ so that

If $(\mathbf{v}_i^{l+1)}|_j > V_{j,\max})\quad \mathbf{v}_i^{l+1)}|_j = V_{j,\max}$;
If $(\mathbf{v}_i^{l+1)}|_j < -V_{j,\max})\quad \mathbf{v}_i^{l+1)}|_j = -V_{j,\max}$;

where $\mathbf{v}|_j$ denotes the jth element of $\mathbf{v}$. Moreover, if the velocity (32) approaches zero, it is reinitialised to proportional to $V_{j,\max}$ with a small factor γ

$$\text{If } (\mathbf{v}_i^{l+1)}|_j == 0)\ \mathbf{v}_i^{l+1)}|_j = \pm rand() * \gamma * V_{j,\max}; \qquad (35)$$

Similarly, each element of $\mathbf{u}_i^{l+1)}$ is checked to ensure that it stays inside the search space:

If $(\mathbf{u}_i^{l+1)}|_j > P_{j,\max})\quad \mathbf{u}_i^{l+1)}|_j = P_{j,\max}$;
If $(\mathbf{u}_i^{l+1)}|_j < P_{j,\min})\quad \mathbf{u}_i^{l+1)}|_j = P_{j,\min}$;

d) Termination condition check. If the maximum number of iterations, $I_{\max}$, is reached, terminate the algorithm with the solution $\mathbf{gb}^{I_{\max})}$; otherwise, set $l = l + 1$ and go to Step *b)*.

When the PSO algorithm terminates, it yields the solution $\mathbf{gb}^{I_{\max})}$, i.e. the centre vector $\boldsymbol{\mu}_n$ and diagonal covariance matrix $\boldsymbol{\Sigma}_n$ of the nth RBF node. The algorithm also generates the nth model column $\mathbf{g}_n$, the orthogonalisation coefficients $\alpha_{j,n}$, $1 \leq j < n$, the corresponding orthogonal model column $\mathbf{p}_n$, and the weight θ_n (hence w_n), as well as $\phi_k^{[n]}$ and $\eta_k^{[n]}$ for $1 \leq k \leq N$. The next stage of the model construction can then commence, and the construction is automatically terminated when the condition (20) is met.

B. PSO algorithmic parameters

It was reported in [12] that using a time varying acceleration coefficient (TVAC) enhances the performance of PSO. We adopt this mechanism, in which c_1 is reduced from 2.5 to 0.5 and c_2 varies from 0.5 to 2.5 during the iterative procedure according to

$$\begin{aligned}c_1 &= (0.5 - 2.5) * l/I_{\max} + 2.5,\\ c_2 &= (2.5 - 0.5) * l/I_{\max} + 0.5.\end{aligned} \qquad (36)$$

The reason for good performance of this TVAC mechanism can be explained as follows. At the initial stages, a large cognitive component and a small social component help particles to wander around or exploit better the search space and to avoid local minima. In the later stages, a small cognitive component and a large social component help particles to converge quickly to a global minimum. We test three choices of the inertia weight, namely, $w_{\rm I} = 0$ as suggested in [12], which removes the influence of the previous velocity, $w_{\rm I}$ set to a small positive constant, and $w_{\rm I} = rand()$. The third choice of the inertia weight typically performs better than the other two choices in our application.

The search space (22) is defined by the specific problem to be solved, and the velocity limit can often be set to

$$V_{j,\max} = 0.5 * (P_{j,\max} - P_{j,\min}). \qquad (37)$$

An appropriate value of the small control factor γ in (35) for avoiding zero velocity is found to be $\gamma = 0.1$ by experiments. In the literature it is suggested that a swarm size of $S = 20$ is often appropriate for wide-ranging problems. For simple and small optimisation problems, $S = 10$ may be adequate. We have found empirically that often the maximum number of iterations can be chosen as $I_{\max} = 20$. Thus, the PSO method is generally very efficient.

C. Computational complexity comparison

Let the computational complexity of evaluating the cost function $F(\mathbf{u})$ once be $C_{\rm single}$. The complexity of the PSO algorithm in solving the optimisation problem (21) or in determining one tunable RBF unit is obviously $I_{\max} \times S \times C_{\rm single}$. The complexity of one LOO cost evaluation and the associated model column orthogonalisation, Steps *b)* and *c)*, can be shown to be the order of N, $\mathcal{O}(N)$ (also see [20]). Thus, $C_{\rm single} = \mathcal{O}(N)$, and the computational requirements of the proposed PSO-aided OFR-LOO algorithm in constructing an M-node RBF model can readily be given as

$$C_{\rm PSO-OFR} = (M + 1) \times I_{\max} \times S \times \mathcal{O}(N). \qquad (38)$$

The ROLS-LOO algorithm [1], [2] is an efficient algorithm for selecting fixed-node RBF classifiers. The complexity of the ROLS-LOO algorithm in selecting M' RBF nodes from the N-candidate set with a given RBF variance is readily determined by

$$C_{\rm ROLS} \approx \left(M' + 1\right) \times N \times \mathcal{O}(N). \qquad (39)$$

The number of cost function evaluations is proportional to the training data size N for the ROLS-LOO algorithm,

TABLE I

COMPARISON OF AVERAGE CLASSIFICATION TEST ERROR RATES IN % OVER THE 100 REALIZATIONS OF THE BREAST CANCER DATA SET OBTAINED BY NINE METHODS. THE FIRST 7 RESULTS WERE QUOTED FROM [23].

method	RBF type	test error rate	model size	complexity
RBF-Network	tunable	27.64 ± 4.71	5	NA
AdaBoost with RBF-Network	tunable	30.36 ± 4.73	5	NA
LP-Reg-AdaBoost (-"-)	tunable	26.79 ± 6.08	5	NA
QP-Reg-AdaBoost (-"-)	tunable	25.91 ± 4.61	5	NA
AdaBoost-Reg (-"-)	tunable	26.51 ± 4.47	5	NA
SVM with RBF-Kernel	fixed	26.04 ± 4.74	not available	NA
Kernel Fisher Discriminant	fixed	24.77 ± 4.63	200	NA
ROLS-LOO	fixed	25.74 ± 5.00	6.0 ± 2.0	$1400 \times \mathcal{O}(200)$
PSO OFR-LOO	tunable	23.04 ± 3.41	2.8 ± 0.9	$760 \times \mathcal{O}(200)$

TABLE II

COMPARISON OF AVERAGE CLASSIFICATION TEST ERROR RATES IN % OVER THE 100 REALIZATIONS OF THE DIABETIS DATA SET OBTAINED BY NINE METHODS. THE FIRST 7 RESULTS WERE QUOTED FROM [23].

method	RBF type	test error rate	model size	complexity
RBF-Network	tunable	24.29 ± 1.88	15	NA
AdaBoost with RBF-Network	tunable	26.47 ± 2.29	15	NA
LP-Reg-AdaBoost (-"-)	tunable	24.11 ± 1.90	15	NA
QP-Reg-AdaBoost (-"-)	tunable	25.39 ± 2.20	15	NA
AdaBoost-Reg (-"-)	tunable	23.79 ± 1.80	15	NA
SVM with RBF-Kernel	fixed	23.53 ± 1.73	not available	NA
Kernel Fisher Discriminant	fixed	23.21 ± 1.63	468	NA
ROLS-LOO	fixed	23.00 ± 1.70	6.0 ± 1.0	$3276 \times \mathcal{O}(468)$
PSO OFR-LOO	tunable	21.87 ± 1.24	3.5 ± 1.4	$900 \times \mathcal{O}(468)$

while for the PSO aided OFR-LOO algorithm, the number of cost function evaluations does not explicitly depend on N. Thus the PSO aided OFR-LOO algorithm has clearly computational advantage for large training data sets. Our experimental results show that the PSO aided OFR-LOO algorithm typically imposes lower computational complexity in classifier construction than the efficient ROLS-LOO algorithm. This computational advantage is largely due to the efficiency of the PSO method. A significant advantage of the PSO aided OFR-LOO approach for constructing tunable RBF classifiers is that the learning algorithm does not contain any hyperparameter which requires cross validation to tune. The complexity $C_{\mathrm{PSO-OFR}}$ represents the true computational requirement of the PSO-aided OFR-LOO algorithm, while the complexity C_{ROLS} is the computational requirement of the ROLS-LOO algorithm with the given RBF variance. Since this variance is not provided by the learning algorithm, it is a hyperparameter and must be determined typically based on a grid-search cross validation. Thus, the true computational advantage of the PSO-aided OFR-LOO algorithm over the ROLS-LOO algorithm is even more significant.

IV. CLASSIFICATION RESULTS

Breast cancer data [21], [22]. The input space dimension was $m = 9$. The data set contained 100 realizations, each having 200 training patterns and 77 test patterns. We applied the ROLS-LOO algorithm [1], [2] to select sparse fixed-node Gaussian RBF classifiers and the PSO-aided OFR-LOO algorithm to construct small tunable Gaussian RBF classifiers, and the results obtained are listed in Table I, in comparison with the seven benchmark results quoted from [23]. For the PSO-aided OFR-LOO, we used $S = 10$, $I_{\max} = 20$ and $w_{\mathrm{I}} = rand()$. For the methods evaluated in [23], the first 5 algorithms optimised the 5-node Gaussian RBF network using various nonlinear optimisation methods, while the kernel Fisher discriminant was the optimal non-sparse method that placed a kernel on every training data sample. For the SVM method with the Gaussian kernel, no average model size was given in [23]. It can be seen from Table I that the PSO aided OFR-LOO algorithm compared favourably with other benchmark methods, in terms of classification accuracy and model size. The PSO aided OFR-LOO algorithm is also seen to impose lower complexity in model construction than the ROLS-LOO algorithm.

Diabetes data [21], [22]. The feature space dimension was $m = 8$. There were 100 realisations of the data set, each having 468 training patterns and 300 test patterns. We applied both the ROLS-LOO and PSO-aided OFR-LOO algorithms to the data set. For the PSO-aided OFR-LOO, we again adopted $S = 10$, $I_{\max} = 20$ and $w_{\mathrm{I}} = rand()$. The results obtained are listed Table II, in comparison with the seven benchmark RBF classifiers studied in [23]. For the first 5 methods of [23], the Gaussian RBF network with 15 optimised nonlinear RBF units was used. For the SVM with RBF kernel, no average model size was given in [23]. It can be seen from Table II that the PSO-aided OFR-LOO method

TABLE III

COMPARISON OF AVERAGE CLASSIFICATION TEST ERROR RATES IN % OVER THE 100 REALIZATIONS OF THE THYROID DATA SET OBTAINED BY NINE METHODS. THE FIRST 7 RESULTS WERE QUOTED FROM [23].

method	RBF type	test error rate	model size	complexity
RBF-Network	tunable	4.52 ± 2.12	8	NA
AdaBoost with RBF-Network	tunable	4.40 ± 2.18	8	NA
LP-Reg-AdaBoost (-"-)	tunable	4.59 ± 2.22	8	NA
QP-Reg-AdaBoost (-"-)	tunable	4.35 ± 2.18	8	NA
AdaBoost-Reg (-"-)	tunable	4.55 ± 2.19	8	NA
SVM with RBF-Kernel	fixed	4.80 ± 2.19	not available	NA
Kernel Fisher Discriminant	fixed	4.20 ± 2.07	140	NA
ROLS-LOO	fixed	4.80 ± 2.20	4.6 ± 1.0	$784 \times \mathcal{O}(140)$
PSO OFR-LOO	tunable	2.48 ± 1.41	3.5 ± 0.8	$1800 \times \mathcal{O}(140)$

produced the best classification accuracy with the smallest RBF classifier. It is also seen to impose lower complexity in model construction than the ROLS-LOO method.

Thyroid data [21], [22]. The input space dimension was $m = 5$. There were 100 realizations of this data set, each containing 140 training patterns and 75 test patterns. Nine RBF classifiers are compared in Table III, with the first seven quoted from [23]. Again it is seen that the PSO-aided OFR-LOO method produced the best classification accuracy with the smallest RBF classifier. The PSO algorithmic parameters were empirically set to $S = 20$, $I_{\max} = 20$ and $w_{\rm I} = rand()$. For this example, the complexity of the PSO aided OFR-LOO algorithm is seen to be higher than that of the ROLS-LOO algorithm when the latter's RBF variance was given. However, several values of RBF variance needed to be tested via grid search for the ROLS-LOO algorithm and, therefore, its true complexity was likely to be higher than that of the PSO aided OFR-LOO algorithm.

V. CONCLUSIONS

A PSO aided construction algorithm has been proposed for RBF classifiers with tunable units, which optimises the RBF units one by one by minimising the LOO misclassification rate based on an efficient PSO aided OFR procedure. The proposed approach does not have any hyperparameter that requires tuning based on cross validation. Compared with the state-of-the-art ROLS-LOO algorithm for selecting fixed-node RBF classifiers, the proposed PSO-aided OFR-LOO algorithm offers significant advantages in terms of better generalisation performance and smaller model size as well as imposes lower complexity in classifier construction process.

REFERENCES

[1] X. Hong, S. Chen and C.J. Harris, "Fast kernel classifier construction using orthogonal forward selection to minimise leave-one-out misclassification rate," in *Proc. 2006 Int. Conf. Intelligent Computing* (Kunming, China), Aug.16-19, 2006, pp.106–114.

[2] 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. Systems Sci.*, vol.39, no.2, pp.119–125, 2008.

[3] V. Vapnik, *The Nature of Statistical Learning Theory*. New York: Springer-Verlag, 1995.

[4] C.J.C. Burges, "A tutorial on support vector machines for pattern recognition," *Data Mining and Knowledge Discovery*, vol.2, no.2, pp.121–167, 1998.

[5] M.E. Tipping, "Sparse Bayesian learning and the relevance vector machine," *J. Machine Learning Research*, vol.1, pp.211–244, 2001.

[6] B. Schölkopf and A.J. Smola, *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press: Cambridge, MA, 2002.

[7] S. Chen, X. Hong and C. J. Harris, "Construction of RBF classifiers with tunable units using orthogonal forward selection based on leave-one-out misclassification rate," in *Proc. 2006 Int. Joint Conf. Neural Networks* (Vancouver, Canada), July 16-21, 2006, pp.6390–6394.

[8] S. Chen, X. Hong, B.L. Luk and C.J. Harris, "Construction of tunable radial basis function networks using orthogonal forward selection," *IEEE Trans. Systems, Man, and Cybernetics, Part B*, vol.39, no.2, pp.457–466, 2009.

[9] S. Chen, X.X. Wang and C.J. Harris, "Experiments with repeating weighted boosting search for optimization in signal processing applications," *IEEE Trans. Systems, Man and Cybernetics, Part B*, vol.35, no.4, pp.682–693, 2005.

[10] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proc. of 1995 IEEE Int. Conf. Neural Networks* (Perth, Australia), Nov.27-Dec.1, 1995, Vol.4, pp.1942–1948.

[11] J. Kennedy and R.C. Eberhart, *Swarm Intelligence*. Morgan Kaufmann, 2001.

[12] A. Ratnaweera, S.K. Halgamuge and H.C. Watson, "Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients," *IEEE Trans. Evolutionary Computation*, vol.8, no.3, pp.240–255, 2004.

[13] S.M. Guru, S.K. Halgamuge and S. Fernando, "Particle swarm optimisers for cluster formation in wireless sensor networks," in *Proc. 2005 Int. Conf. Intelligent Sensors, Sensor Networks and Information Processing* (Melbourne, Australia), Dec.5-8, 2005, pp.319–324.

[14] H.-M. Feng, "Self-generation RBFNs using evolutional PSO learning," *Neurocomputing*, vol.70, no.1-3, pp.241–251, 2006.

[15] K.K. Soo, Y.M. Siu, W.S. Chan, L. Yang and R.S. Chen, "Particle-swarm-optimization-based multiuser detector for CDMA communications," *IEEE Trans. Vehicular Technology*, vol.56, no.5, pp.3006–3013, 2007.

[16] W.-F. Leong and G.G. Yen, "PSO-based multiobjective optimization with dynamic population size and adaptive local archives," *IEEE Trans. Systems, Man and Cybernetics, Part B*, vol.38, no.5, pp.1270–1293, 2008.

[17] W. Yao, S. Chen, S. Tan and L. Hanzo, "Particle swarm optimisation aided minimum bit error rate multiuse transmission," in *Proc. ICC 2009* (Dresden, Germany), June 14-18, 5 pages, 2009.

[18] M. Stone, "Cross validation choice and assessment of statistical predictions," *J. Royal Statistics Society Series B*, vol.36, pp.117–147, 1974.

[19] R.H. Myers, *Classical and Modern Regression with Applications*. 2nd Edition, Boston, MA: PWS-KENT, 1990.

[20] S. Chen and J. Wigger, "Fast orthogonal least squares algorithm for efficient subset model selection," *IEEE Trans. Signal Processing*, vol.43, no.7, pp.1713–1715, 1995.

[21] `http://www.ics.uci.edu/~mlearn/MLRepository.html`

[22] `ida.first.fhg.de/projects/bench/benchmarks.htm`

[23] G. Rätsch, T. Onoda, and K.R. Müller, "Soft margins for AdaBoost," *Machine Learning*, vol.42, no.3, pp.287–320, 2001.