2006 International Joint Conference on Neural Networks
Sheraton Vancouver Wall Centre Hotel, Vancouver, BC, Canada
July 16-21, 2006

Construction of RBF Classifiers with Tunable Units Using Orthogonal Forward Selection Based on Leave-One-Out Misclassification Rate

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

X. Hong
Department of Cybernetics
University of Reading, Reading RG6 6AY, U.K.
E-mail: x.hong@reading.ac.uk

Abstract— **An orthogonal forward selection (OFS) algorithm based on leave-one-out (LOO) misclassification rate is proposed for the construction of radial basis function (RBF) classifiers with tunable units. Each stage of the construction process determines a RBF unit, namely its centre vector and diagonal covariance matrix as well as weight, by minimising the LOO statistics. This OFS-LOO algorithm is computationally efficient and it is capable of constructing parsimonious RBF classifiers that generalise 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 classifier construction procedure is demonstrated using three classification benchmark examples.**

I. Introduction

A basic principle in nonlinear data modelling is that of ensuring the smallest possible model which explains the training data. This parsimonious principle is particularly relevant in the construction of radial basis function (RBF) classifiers. The key questions in constructing a RBF classifier are how many RBF units to use, the positions (centres) and shapes (variances or covariance matrices) of the RBF nodes. The objective is to obtain sparse RBF classifiers that generalise well, i.e. achieving small misclassification rate for data unseen in training. A popular approach for constructing RBF classifiers is to consider the training input data points as the RBF centres and to employ a common variance for every RBF unit. A sparse representation is then sought using for example the support vector machine (SVM) and other sparse kernel methods [1], [2], [3], [4], [5], [6], or the orthogonal forward selection (OFS) [7], [8]. The SVM is based on the structural risk minimisation principle and approximately minimises an upper bound on the generalisation error [1]. The OFS procedure of [7] incrementally maximises the Fisher ratio of class separability measure, while the OFS construction algorithm of [8] incrementally minimises the leave-one-out (LOO) misclassification rate, which is a measure of the classifier's generalisation capability [9].

In the above-mentioned methods, the value of the common RBF variance used has an important influence on the sparsity level of the classifier and its generalisation capability. Since the construction algorithms themselves do not provide this RBF variance, it has to be learnt typically via cross validation [9], [10]. A RBF classifier will have better modelling capability if its centres are tunable and each node has its own covariance matrix [11]. Thus, all the parameters of the RBF classifier, the RBF centres, variances or covariance matrices and weights, can alternatively be learnt together via nonlinear optimisation (e.g. [12]). The optimisation process associated with this nonlinear learning approach, however, is highly complex and nonconvex, and the evolutionary optimisation has been suggested to solve this type of nonlinear learning problems [13], at the cost of an increased computational complexity. The work [14] has compared several sophisticated state-of-the-art nonlinear optimisation algorithm for constructing RBF classifiers. Note that in this completed nonlinear optimisation approach, the number of RBF units to use has to be determined typically via (costly) cross validation.

We present a construction method for producing sparse RBF classifiers with tunable units. Unlike the "linear-in-the-parameters" kernel approach, RBF centres are not restricted to be the training input data and each RBF unit has an individually tuned diagonal covariance matrix. On the other hand, we do not attempt to optimise all the parameters of the RBF classifiers together, as the nonlinear optimisation approach does. Rather, we construct RBF units one by one in the OFS incremental construction process. At Each stage of the construction process, a RBF unit is tuned by determining its RBF centre and diagonal covariance matrix through minimising the LOO misclassification rate criterion. Since this optimisation task is non-convex, a guided global search algorithm, referred to as the repeated weighted boosting search (RWBS) [15], is adopted to perform this optimisation. Due to the orthogonal decomposition, the computation of the LOO misclassification rate is very efficient and this ensures a fast construction process [8], [16]. Moreover, the number of RBF units to use is automatically determined without costly additional cross validation, and the user does not need to specify a termination criterion for the construction process. Experimental results involving three benchmark classification examples are included to illustrated the effectiveness of the proposed OFS-LOO construction algorithm for the RBF clas-

0-7803-9490-9/06/$20.00/©2006 IEEE

sifier with tunable units.

II. Construction of the RBF classifier with tunable units

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

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

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

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

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

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

where $\mathbf{w} = [w_1\; w_2 \cdots w_M]^T$ is the RBF weight vector and $\mathbf{g}(k) = [g_1(\mathbf{x}_k)\; g_2(\mathbf{x}_k) \cdots g_M(\mathbf{x}_k)]^T$ is the "regressor" vector. We will 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 $\boldsymbol{\Sigma}_i = \mathrm{diag}\{\sigma_{i,1}^2, \cdots, \sigma_{i,m}^2\}$, 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 $\mathbf{G} = [\mathbf{g}_1\; \mathbf{g}_2 \cdots \mathbf{g}_M]$ 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

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

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

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

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

and

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

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

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

where the weight vector $\boldsymbol{\theta} = [\theta_1\; \theta_2 \cdots \theta_M]^T$ in the orthogonal model space satisfies the triangular system $\mathbf{A}\mathbf{w} = \boldsymbol{\theta}$. 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

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

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

The goal of a classifier is to minimise the misclassification or error rate. Define the signed decision variable

$$s_k = \mathrm{sgn}(y_k)\hat{y}_k = y_k \hat{y}_k = y_k f_{\mathrm{RBF}}^{(M)}(\mathbf{x}_k). \quad (11)$$

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

$$\mathcal{M}_r = \frac{1}{N} \sum_{k=1}^{N} \mathcal{I}_d(s_k), \quad (12)$$

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

$$\mathcal{I}_d(y) = \begin{cases} 1, & y \le 0, \\ 0, & y > 0. \end{cases} \quad (13)$$

The classifier's generalisation capability however is usually measured by the test error rate over data unseen in training.

A. Orthogonal forward selection based on the leave-one-out misclassification rate

It is highly desirable to construct the RBF classifier (1) by directly optimising the classifier's generalisation capability. Cross validation criteria are metrics that measures a model's generalisation capability. One commonly used version of cross validation is the LOO cross validation [9], [10]. Let us denote the n-unit RBF classifier, identified using the entire training data set $\{(\mathbf{x}_k, y_k)\}_{k=1}^N$, as $f_{\mathrm{RBF}}^{(n)}(\bullet)$. The kth modelling error for this RBF classifier is given by

$$e_k^{(n)} = y_k - f_{\mathrm{RBF}}^{(n)}(\mathbf{x}_k) = y_k - \hat{y}_k^{(n)}. \quad (14)$$

Let $f_{\mathrm{RBF}}^{(n,-k)}(\bullet)$ 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

$$\hat{y}_k^{(n,-k)} = f_{\mathrm{RBF}}^{(n,-k)}(\mathbf{x}_k). \quad (15)$$

The associated LOO signed decision variable is defined by

$$s_k^{(n,-k)} = y_k \hat{y}_k^{(n,-k)} \quad (16)$$

and the LOO misclassification rate is computed by

$$J_n = \frac{1}{N} \sum_{k=1}^{N} \mathcal{I}_d\left(s_k^{(n,-k)}\right). \quad (17)$$

This LOO misclassification rate is a measure of the classifier's generalisation capability.

The recent work [8] has shown that the LOO signed decision variable $s_k^{(n,-k)}$ can be calculated very fast owning to the orthogonal decomposition. Therefore, the LOO misclassification rate J_n can be computed efficiently. Firstly the LOO modelling error $e_k^{(n,-k)} = y_k - \hat{y}_k^{(n,-k)}$ can be expressed as [10]

$$e_k^{(n,-k)} = \frac{e_k^{(n)}}{\eta_k^{(n)}}, \tag{18}$$

where $\eta_k^{(n)}$ is known as the LOO error weighting. Next it can be shown that [16], [17]

$$e_k^{(n)} = y_k - \sum_{i=1}^{n} \theta_i p_i(k) = e_k^{(n-1)} - \theta_n p_n(k), \tag{19}$$

and

$$\eta_k^{(n)} = 1 - \sum_{i=1}^{n} \frac{p_i^2(k)}{\mathbf{p}_i^T \mathbf{p}_i + \lambda} = \eta_k^{(n-1)} - \frac{p_n^2(k)}{\mathbf{p}_n^T \mathbf{p}_n + \lambda}, \tag{20}$$

respectively, where $\lambda \geq 0$ is a small regularisation parameter. Thus the LOO modelling error is given by

$$y_k - \hat{y}_k^{(n,-k)} = \frac{y_k - \hat{y}_k^{(n)}}{1 - \sum_{i=1}^{n} \frac{p_i^2(k)}{\mathbf{p}_i^T \mathbf{p}_i + \lambda}}. \tag{21}$$

Multiplying both sides of (21) with y_k and applying $y_k^2 = 1$ yields [8]

$$1 - s_k^{(n,-k)} = \frac{1 - y_k \hat{y}_k^{(n)}}{1 - \sum_{i=1}^{n} \frac{p_i^2(k)}{\mathbf{p}_i^T \mathbf{p}_i + \lambda}}, \tag{22}$$

that is,

$$s_k^{(n,-k)} = \frac{\sum_{i=1}^{n} y_k\, \theta_i\, p_i(k) - \sum_{i=1}^{n} \frac{p_i^2(k)}{\mathbf{p}_i^T \mathbf{p}_i + \lambda}}{1 - \sum_{i=1}^{n} \frac{p_i^2(k)}{\mathbf{p}_i^T \mathbf{p}_i + \lambda}} = \frac{\phi_k^{(n)}}{\eta_k^{(n)}}. \tag{23}$$

The recursive formula for the LOO error weighting $\eta_k^{(n)}$ is given in (20), while $\phi_k^{(n)}$ can be represented using the following recursive formula

$$\phi_k^{(n)} = \phi_k^{(n-1)} + y_k\, \theta_n\, p_n(k) - \frac{p_n^2(k)}{\mathbf{p}_n^T \mathbf{p}_n + \lambda}. \tag{24}$$

The proposed OFS-LOO algorithm constructs the RBF units of the classifier one by one by minimising the LOO misclassification rate J_n. 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 $\boldsymbol{\Sigma}_n$

$$\min_{\boldsymbol{\mu}_n, \boldsymbol{\Sigma}_n} J_n \left(\boldsymbol{\mu}_n, \boldsymbol{\Sigma}_n\right). \tag{25}$$

The construction procedure is automatically terminated when

$$J_M \leq J_{M+1}, \tag{26}$$

yielding an M-term RBF classifier. Note that the LOO criterion J_n is at least locally convex, and there exists an "optimal" M such that: for $n \leq M$ J_n decreases as the model size n increases while the condition (26) holds [16], [17].

B. Positioning and shaping a RBF unit

It can be seen that the task at the nth stage of the RBF classifier construction is to position and shape the nth RBF unit by solving the optimisation problem (25). Since this optimisation problem is non-convex, a gradient-based algorithm may become trapped at a local minimum. Alternatively, global optimisation methods, such as the genetic algorithm (GA) [18], [19] and adaptive simulated annealing (ASA) [20], [21], may be used to perform the optimisation task (25). We adopt a simply yet efficient global search algorithm called the RWBS [15] to determine $\boldsymbol{\mu}_n$ and $\boldsymbol{\Sigma}_n$. The motivation and analysis of the RWBS algorithm as a general global optimiser are detailed in [15]. A comparative study given in [15] shows that the RWBS algorithm achieves a similar convergence speed as the GA and ASA for several global optimisation applications. The RWBS algorithm has additional advantages of requiring minimum programming effort and having fewer algorithmic parameters that require to tune, in comparison with the GA and ASA. The procedure for determining the nth RBF unit based on the RWBS algorithm is summarised in the following.

Let $\mathbf{u}$ be the vector that contains $\boldsymbol{\mu}_n$ and $\boldsymbol{\Sigma}_n$. Give the initial conditions

$$\phi_k^{(0)} = 0 \text{ and } \eta_k^{(0)} = 1, \ 1 \leq k \leq N, \text{ and } J_0 = 1. \tag{27}$$

Specify the population size P_S, the number of generations in the repeated search N_G, and the accuracy for terminating the weighted boosting search ξ_B.

Outer loop: generations For $l = 1 : N_G$

Generation initialisation: Initialise the population by setting $\mathbf{u}_1^{[l]} = \mathbf{u}_{\text{best}}^{[l-1]}$ and randomly generating 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 initialisation: Assign the initial distribution weightings $\delta_i(0) = \frac{1}{P_S}$, $1 \leq i \leq P_S$, 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 orthogonalise them:

$$\alpha_{j,n}^{i)} = \frac{\mathbf{p}_j^T \mathbf{g}_n^{i)}}{\mathbf{p}_j^T \mathbf{p}_j}, \ 1 \leq j < n, \tag{28}$$

$$\mathbf{p}_n^{i)} = \mathbf{g}_n^{i)} - \sum_{j=1}^{n-1} \alpha_{j,n}^{i)} \mathbf{p}_j, \tag{29}$$

$$\theta_n^{i)} = \frac{\left(\mathbf{p}_n^{i)}\right)^T \mathbf{y}}{\left(\mathbf{p}_n^{i)}\right)^T \mathbf{p}_n^{i)} + \lambda}. \tag{30}$$

2) For $1 \leq i \leq P_S$, calculate the LOO cost function value of each $\mathbf{u}_i^{[l]}$

$$\phi_k^{(n)}(i) = \phi_k^{(n-1)} + y_k p_n^{i)}(k) \theta_n^{i)} - \frac{\left(p_n^{i)}(k)\right)^2}{\left(\mathbf{p}_n^{i)}\right)^T \mathbf{p}_n^{i)} + \lambda}, \tag{31}$$

$$\eta_k^{(n)}(i) = \eta_k^{(n-1)} - \frac{\left(p_n^{i)}(k)\right)^2}{\left(\mathbf{p}_n^{i)}\right)^T \mathbf{p}_n^{i)} + \lambda}, \qquad (32)$$

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

$$J_n^{i)} = \frac{1}{N} \sum_{k=1}^{N} \mathcal{I}_d \left(\frac{\phi_k^{(n)}(i)}{\eta_k^{(n)}(i)} \right). \qquad (33)$$

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

Inner loop: weighted boosting search Set $t = 0$; $t = t + 1$

Step 1: Boosting

1) Find

$$i_{\text{best}} = \arg \min_{1 \leq i \leq P_S} J_n^{i)} \quad \text{and} \quad i_{\text{worst}} = \arg \max_{1 \leq i \leq P_S} J_n^{i)}.$$

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) Normalise the cost function values

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

3) Compute a weighting factor β_t according to

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

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

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

and normalise them

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

Step 2: Parameter updating

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

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

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

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

3) Calculate $\mathbf{g}_n^{P_S+1)}$ and $\mathbf{g}_n^{P_S+2)}$ from $\mathbf{u}_{P_S+1}$ and $\mathbf{u}_{P_S+2}$, orthogonalise these two candidate model columns (as in (28) to (30)), and compute their corresponding LOO cost function values $J_n^{i)}$, $i = P_S + 1, P_S + 2$ (as in (31) to (33)). Then find

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

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

If $\|\mathbf{u}_{P_S+1} - \mathbf{u}_{P_S+2}\| < \xi_B$, exit **inner loop**.

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 unit, 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, as well as $\phi_k^{(n)}$ and $\eta_k^{(n)}$ for $1 \leq k \leq N$.

The motivations and analysis of the RWBS algorithm as a global optimiser is detailed in [15]. To guarantee a global optimal solution as well as to achieve a fast convergence, P_S, N_G and ξ_B need to be set carefully. The appropriate values for these algorithmic parameters depend on the dimension of $\mathbf{u}$ and how hard the objective function to be optimised, and generally they have to be found empirically. The elitist initialisation is very useful, as it keeps the information obtained by the previous search generation, which otherwise would be lost due to the randomly sampling initialisation. In the inner loop optimisation, there is no need for every members of the population to converge to a (local) minimum, and it is sufficient to locate where the minimum lies. Thus ξ_B can be set to a relatively large value. This makes the search efficient, achieving convergence with a small number of the cost function evaluations. A sufficiently large N_G should be used to ensure that the parameter space is sampled sufficiently.

It is worth emphasising that P_S, N_G and ξ_B are not the learning hyperparameters of the proposed OFS-LOO algorithm. Rather they are the optimisation algorithmic parameters, which are similar in nature to the step size in a gradient-based optimisation algorithm or the algorithmic parameters of the particular quadratic optimiser chosen to solve the optimisation task of the SVM learning algorithm. It is important to distinguish the algorithmic parameters of a particular optimiser used to solve the optimisation task of a learning problem from the possible learning hyperparameters that may be inherent in the learning problem.

III. Classification Results

Numerical experiments were performed to demonstrate the modelling results of the proposed OFS-LOO algorithm for construction RBF classifiers with tunable units, in comparison to those of several benchmark classification algorithms as published in [14]. Three two-class data sets, Breast Cancer, Diabetes and Hhyroid, were experimented. These benchmark data sets were originated in the UCI repository [22] and we obtained the data sets from [23]. The information regarding these benchmark data sets can be found in [23]. In our experiments, the basis function $K(\bullet)$ was chosen to be Gaussian. Seven benchmark RBF classifiers were studied in [14], and the results given in [23] were reproduced in Table I to III, in comparison with the results obtained by our proposed OFS-LOO algorithm. It can be seen that our method produced the best classification accuracy with the smallest RBF classifier for all the three data sets.

TABLE I

Average classification test error rate in % over the 100 realizations of the breast cancer data set. The first 7 results were quoted from [23].

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	not available
Proposed OFS-LOO	24.49 ± 3.28	3.1 ± 1.2

TABLE II

Average classification test error rate in % over the 100 realizations of the diabetis data set. The first 7 results were quoted from [23].

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	not available
Proposed OFS-LOO	22.16 ± 1.47	4.0 ± 1.6

TABLE III

Average classification test error rate in % over the 100 realizations of the thyroid data set. The first 7 results were quoted from [23].

method	test error rate	model size
RBF-Network	4.52 ± 2.12	8
AdaBoost with RBF-Network	4.40 ± 2.18	8
LP-Reg-AdaBoost (-”-)	4.59 ± 2.22	8
QP-Reg-AdaBoost (-”-)	4.35 ± 2.18	8
AdaBoost-Reg (-”-)	4.55 ± 2.19	8
SVM with RBF-Kernel	4.80 ± 2.19	not available
Kernel Fisher Discriminant	4.20 ± 2.07	not available
Proposed OFS-LOO	3.21 ± 1.35	3.9 ± 0.8

IV. Conclusions

A novel construction algorithm has been proposed for RBF classifiers with tunable units. Unlike most of the sparse RBF modelling methods, the RBF centres are not restricted to the training input data points and each RBF unit has an individually adjusted diagonal covariance matrix. On the other hand, we do not attempt to optimise all the RBF classifier's parameters together using nonlinear optimisation. Rather we optimise the RBF units one by one by minimising the LOO misclassification rate, which is a measure of the model generalisation capability. The RBF units are selected in a computationally efficient OFS procedure, and the orthogonal decomposition ensures a fast updating of the LOO misclassification rate criterion. Moreover, the RBF classifier construction is fully automatic and the user does not need to specify any additional termination criterion. Three classification benchmark examples have been used in our simulation experiment, and the results obtained have demonstrated that the proposed RBF classifier construction algorithm compares favourably with several existing state-of-the-art classifier construction algorithms.

Acknowledgements

S. Chen wish to thank the support of the United Kingdom Royal Academy of Engineering.

References

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

[2] S. Gunn, “Support vector machines for classification and regression,” *Technical Report*, ISIS Research Group, Department of Electronics and Computer Science, University of Southampton, UK, May 1998.

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

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

[5] P. Vincent and Y. Bengio, “Kernel matching pursuit,” *Machine Learning*, vol.48, no.1, pp.165–187, 2002.

[6] G.R.G. Lanckriet, N. Cristianini, P. Bartlett, L.E Ghaoui and M.I. Jordan, “Learning the kernel matrix with semidefinite programming,” *J. Machine Learning Research*, vol.5, pp.27–72, 2004.

[7] K.Z. Mao, “RBF neural network center selection based on fisher ratio class separability measure,” *IEEE Trans. Neural Networks*, vol.13, no.5, pp.1211–1217, 2002.

[8] 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,” submitted to *Int. J. Systems Science*, 2006.

[9] M. Stone, “Cross validation choice and assessment of statistical predictions,” *Applied Statistics*, vol.36, pp. 117–147, 1974.

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

[11] S. Lee and R.M. Kil, “A Gaussian potential function network with hierarchically self-organizing learning,” *Neural Networks*, vol.4, no.2, pp.207–224, 1991.

[12] Z.R. Yang and S. Chen, “Robust maximum likelihood training of heteroscedastic probabilistic neural networks,” *Neural Networks*, vol.11, pp.739–747, 1998.

[13] 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 Networks*, vol.14, no.6, pp.1478–1495, 2003.

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

[15] 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.

[16] X. Hong, P.M. Sharkey and K. Warwick, “Automatic nonlinear predictive model construction algorithm using forward regression and the PRESS statistic,” *IEE Proc. Control Theory and Applications*, vol.150, no.3, pp.245–254, 2003.

[17] S. Chen, X. Hong, C.J. Harris and P.M. Sharkey, “Sparse modelling using orthogonal forward regression with PRESS statistic and regularization,” *IEEE Trans. Systems, Man and Cybernetics, Part B*, vol.34, no.2, pp.898–911, 2004.

[18] D.E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*. Reading, MA: Addison Wesley, 1989.

[19] K.F. Man, K.S. Tang and S. Kwong, *Genetic Algorithms: Concepts and Design*. London: Springer-Verlag, 1998.

[20] L. Ingber, “Simulated annealing: practice versus theory,” *Mathematical and Computer Modeling*, vol.18, no.11, pp.29–57, 1993.

[21] S. Chen and B.L. Luk, “Adaptive simulated annealing for optimization in signal processing applications,” *Signal Processing*, vol.79, no.1, pp.117–128, 1999.

[22] http://www.ics.uci.edu/∼mlearn/MLRepository.html

[23] http://ida.first.fhg.de/projects/bench/benchmarks.htm