Kernel Classifier Construction Using Orthogonal Forward Selection and Boosting With Fisher Ratio Class Separability Measure

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

***Abstract*—A greedy technique is proposed to construct parsimonious kernel classifiers using the orthogonal forward selection method and boosting based on Fisher ratio for class separability measure. Unlike most kernel classification methods, which restrict kernel means to the training input data and use a fixed common variance for all the kernel terms, the proposed technique can tune both the mean vector and diagonal covariance matrix of individual kernel by incrementally maximizing Fisher ratio for class separability measure. An efficient weighted optimization method is developed based on boosting to append kernels one by one in an orthogonal forward selection procedure. Experimental results obtained using this construction technique demonstrate that it offers a viable alternative to the existing state-of-the-art kernel modeling methods for constructing sparse Gaussian radial basis function network classifiers that generalize well.**

***Index Terms*—Boosting, classification, Fisher ratio of class separability, forward selection, kernel classifier, orthogonal least square, radial basis function network.**

I. Introduction

A fundamental principle in practical nonlinear data modeling is the parsimonious principle of ensuring the smallest possible model that explains the training data. Recently, the state-of-the-art kernel modeling techniques, such as the support vector machine (SVM) and relevant vector machine (RVM) [1]–[3], have widely been adopted in classification applications to construct sparse classifiers that generalize well. Alternatively, a greedy technique can be applied to construct parsimonious classifiers by incrementally maximizing Fisher ratio of class separability measure in an orthogonal forward selection (OFS) procedure [4], [5]. In most of the existing sparse kernel construction techniques, a fixed common variance is used for all the kernels and the kernel centers or means are placed at the training input data.

We present a flexible construction method for parsimonious classifier modeling. The proposed algorithm tunes both the mean vector and diagonal covariance matrix of individual kernel by incrementally maximizing the Fisher ratio of class separability measure in an OFS procedure. To incrementally append the classifier's kernels one by one, a weighted optimization search algorithm is developed, which is based on the idea from boosting [6]–[8]. Because kernel means are not restricted to the training input data and each kernel term has an individually tuned diagonal covariance matrix, our method can produce very sparse classifiers. The proposed weighted optimization algorithm is simple, robust, and easy to implement. Experimental results are included to demonstrate the effectiveness of this incremental OFS construction algorithm with boosting (OFSwB) optimization for constructing Gaussian radial basis function network classifiers.

Manuscript received March 24, 2005; accepted February 10, 2006.

S. Chen and C. J. Harris are with the School of Electronics and Computer Science, University of Southampton, Southampton SO17 1BJ, U.K. (e-mail: sqc@ecs.soton.ac.uk; cjh@ecs.soton.ac.uk).

X. X. Wang is with the Institute of Human Genetics, University of Newcastle, Newcastle upon Tyne NE1 3BZ, U.K. (e-mail: xunxian.wang@ncl.ac.uk).

X. Hong is with the Department of Cybernetics, University of Reading, Reading RG6 6AY, U.K. (e-mail: x.hong@reading.ac.uk).

Digital Object Identifier 10.1109/TNN.2006.881487

II. Orthogonal Forward Selection for Classifier Construction

Consider the kernel classifier of the form

$$\hat{c}_l = \text{sgn}(y_l) \text{ with } y_l = \sum_{i=1}^{M} w_i g_i(\mathbf{x}_l) \tag{1}$$

where $\mathbf{x}_l$ is an m-dimensional pattern vector with its associated class label $c_l \in \{\pm 1\}$, y_l is the classifier output for input $\mathbf{x}_l$, and $\hat{c}_l$ is the estimated class label for $\mathbf{x}_l$; $w_i, 1 \le i \le M$, denote the classifier weights, M is the number of kernels, and $g_i(\cdot), 1 \le i \le M$, denote the classifier kernels. We allow the kernel function to be chosen as the general Gaussian function $g_i(\mathbf{x}) = G(\mathbf{x}; \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$ with

$$G(\mathbf{x}; \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i) = \exp\left(-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu}_i)^T \boldsymbol{\Sigma}_i^{-1} (\mathbf{x} - \boldsymbol{\mu}_i)\right) \tag{2}$$

where the diagonal covariance matrix has the form of $\boldsymbol{\Sigma}_i = \text{diag}\{\sigma_{i,1}^2, \ldots, \sigma_{i,m}^2\}$. Obviously, our method is equally applicable to other kernel functions. We will adopt an OFS procedure to build up the classifier (1) by appending kernels one by one.

Given the N pairs of training data $\{\mathbf{x}_l, c_l\}_{l=1}^N$, let us define the modeling residual as $e_l = c_l - y_l$. Then the classifier model(1) over the training data set can be expressed as

$$\mathbf{c} = \mathbf{G}\mathbf{w} + \mathbf{e} \tag{3}$$

where $\mathbf{c} = [c_1 c_2 \cdots c_N]^T$, $\mathbf{e} = [e_1 e_2 \cdots e_N]^T$, the kernel matrix

$$\mathbf{G} = [\mathbf{g}_1 \mathbf{g}_2 \cdots \mathbf{g}_M] \tag{4}$$

with $\mathbf{g}_k = [g_k(\mathbf{x}_1) g_k(\mathbf{x}_2) \cdots g_k(\mathbf{x}_N)]^T$, and the classifier weight vector $\mathbf{w} = [w_1 w_2 \cdots w_M]^T$. Let an orthogonal decomposition of $\mathbf{G}$ be

$$\mathbf{G} = \mathbf{P}\mathbf{A} \tag{5}$$

where

$$\mathbf{A} = \begin{bmatrix} 1 & a_{1,2} & \cdots & a_{1,M} \\ 0 & 1 & \ddots & \vdots \\ \vdots & \ddots & \ddots & a_{M-1,M} \\ 0 & \cdots & 0 & 1 \end{bmatrix} \tag{6}$$

and

$$\mathbf{P} = [\mathbf{p}_1 \mathbf{p}_2 \cdots \mathbf{p}_M] = \begin{bmatrix} p_{1,1} & p_{1,2} & \cdots & p_{1,M} \\ p_{2,1} & p_{2,2} & \cdots & p_{2,M} \\ \vdots & \vdots & \vdots & \vdots \\ p_{N,1} & p_{N,2} & \cdots & p_{N,M} \end{bmatrix} \tag{7}$$

with orthogonal columns that satisfy $\mathbf{p}_i^T \mathbf{p}_j = 0$, if $i \ne j$. The model (3) can alternatively be expressed as

$$\mathbf{c} = \mathbf{P}\boldsymbol{\theta} + \mathbf{e} \tag{8}$$

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

A sparse k-term classifier model can be selected by incrementally maximizing a class separability measure in an OFS procedure [4], [5]. Define the two class sets $\mathcal{C}_\pm = \{\mathbf{x}_l : c_l = \pm 1\}$, and let the numbers of points in $\mathcal{C}_\pm$ be $N_\pm$, respectively, with $N_+ + N_- = N$. The means and

1045-9227/$20.00 © 2006 IEEE

variances of training samples belonging to classes $\mathcal{C}_{\pm}$ in the direction of basis $\mathbf{p}_k$ are given by

$$m_{+,k} = \frac{1}{N_+}\sum_{l=1}^{N}\delta(c_l - 1)p_{l,k},$$
$$\sigma^2_{+,k} = \frac{1}{N_+}\sum_{l=1}^{N}\delta(c_l - 1)(p_{l,k} - m_{+,k})^2 \quad (9)$$
$$m_{-,k} = \frac{1}{N_-}\sum_{l=1}^{N}\delta(c_l + 1)p_{l,k},$$
$$\sigma^2_{-,k} = \frac{1}{N_-}\sum_{l=1}^{N}\delta(c_l + 1)(p_{l,k} - m_{-,k})^2 \quad (10)$$

respectively, where $\delta(x) = 1$ for $x = 0$ and $\delta(x) = 0$ for $x \neq 0$. Fisher ratio,[1] defined as the ratio of the interclass difference and the intraclass spread, in the direction of $\mathbf{p}_k$ is given by [9]

$$F_k = \frac{(m_{+,k} - m_{-,k})^2}{\sigma^2_{+,k} + \sigma^2_{-,k}}. \quad (11)$$

At the kth stage of incremental modeling, the kth term is selected to maximize the Fisher ratio (11). However, unlike the original OFS procedure [4], [5], which restricts the choice of the kernel center $\boldsymbol{\mu}_k$ to the training data points and uses a fixed common variance, the maximization here is with respect to the mean vector $\boldsymbol{\mu}_k$ and the diagonal covariance matrix $\boldsymbol{\Sigma}_k$ of the kth kernel term. The forward selection procedure is terminated at the kth stage if

$$\frac{F_k}{\sum_{i=1}^{k} F_i} < \xi \quad (12)$$

is satisfied, where the small positive scalar ξ is a chosen tolerance that determines the sparsity of the selected classifier model. The appropriate value for ξ is problem dependent and has to be found empirically. Alternatively, cross-validation can be employed to terminate the OFS procedure. The least square solution for the corresponding sparse classifier weight vector $\mathbf{w}_k$ is readily available given the least square solution of $\boldsymbol{\theta}_k$.

III. Weighted Optimization With Boosting

It can be seen that at each incremental modeling stage, the basic task is to maximize the Fisher ratio criterion $F_k(\mathbf{u})$ over $\mathbf{u} \in \mathcal{U}$, where the vector $\mathbf{u}$ contains the kernel mean vector $\boldsymbol{\mu}$ and the diagonal covariance matrix $\boldsymbol{\Sigma}$. This task may be carried out by some global optimization methods, such as the genetic algorithm [10], [11] and adaptive simulated annealing [12], [13]. These standard global optimization methods are, however, computationally very expensive. Let us consider a simple search method to perform this optimization. Given s points of $\mathbf{u}, \mathbf{u}_i$ for $1 \leq i \leq s$, let $\mathbf{u}_{\rm best} = \arg\max\{F_k(\mathbf{u}_i), 1 \leq i \leq s\}$ and $\mathbf{u}_{\rm worst} = \arg\min\{F_k(\mathbf{u}_i), 1 \leq i \leq s\}$. An $(s+1)$th point $\mathbf{u}_{s+1}$ is first generated by a weighted convex combination of $\mathbf{u}_i, 1 \leq i \leq s$. An $(s+2)$th value $\mathbf{u}_{s+2}$ is then generated as the mirror image of $\mathbf{u}_{s+1}$, with respect to $\mathbf{u}_{\rm best}$, along the direction defined by $\mathbf{u}_{\rm best} - \mathbf{u}_{s+1}$. The best of $\mathbf{u}_{s+1}$ and $\mathbf{u}_{s+2}$ then replaces $\mathbf{u}_{\rm worst}$. The process is repeated until it converges. A simple illustration is depicted in Fig. 1 for a one-dimensional case, where there are $s = 3$ points, $\mathbf{u}_1, \mathbf{u}_2$ and $\mathbf{u}_3$, and $\mathbf{u}_{\rm best} = \mathbf{u}_2$ and $\mathbf{u}_{\rm worst} = \mathbf{u}_3$. The fourth value $\mathbf{u}_4$ is a weighted combination of $\mathbf{u}_1, \mathbf{u}_2$, and $\mathbf{u}_3$, and $\mathbf{u}_5$ is the mirror image of $\mathbf{u}_4$ with respect to $\mathbf{u}_2$. As $\mathbf{u}_4$ is better than $\mathbf{u}_5$, it will replace $\mathbf{u}_3$.

[1]In this paper, we restrict to the two-class classification problem. The definition of Fisher ratio, however, is applicable to the multiclass case, and the algorithm presented in this paper can be extended to the multiclass classification problem. Also see [4].

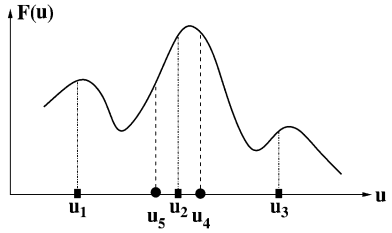

Fig. 1. Illustration of a simple weighted search optimization process.

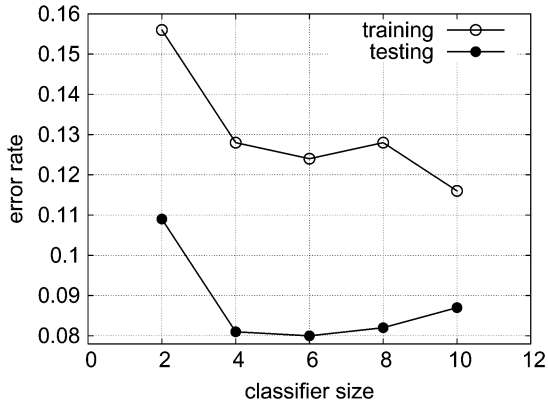

Fig. 2. Synthetic data: training and test error rates versus size of selected classifier.

Clearly, how the weighted combination is performed is crucial. The weightings for $\mathbf{u}_i, 1 \leq i \leq s$, should reflect the "goodness" of $\mathbf{u}_i$ and, moreover, the process should be capable of self-learning or adapting these weightings. This is exactly the basic idea of boosting [6]–[8]. Specifically, we combine the AdaBoost algorithm of [6] with the aforementioned simple search strategy to form a weighted search routine. This weighted search routine performs a guided random search with a population of s members $\mathbf{u}_i, 1 \leq i \leq s$, and the solution obtained may depend on the initial choice of population. To derive a robust algorithm that is independent of the initial choices of population and to improve the probability of achieving a global optimal solution, we adopt a strategy of repeating the weighted search routine with an elitist initialization of the population, namely, each repeat or generation of the weighted search will start by retaining the solution found in the previous generation and filling the rest of the population randomly. The resulting weighted optimization algorithm, referred to as the OFSwB, is summarized as follows, given the training data $\{\mathbf{x}_l, c_l\}_{l=1}^{N}$ at the kth stage of modeling.

A. Outer Loop: Number of Generations—$l = 1 : L_{\max}$

1) Initialization:

a) Set $\mathbf{u}_1 = \mathbf{u}_{\rm best}^{(l-1)}$ and randomly generate the rest of the population members $\mathbf{u}_i, 2 \leq i \leq s$, where $\mathbf{u}_{\rm best}^{(l-1)}$ denotes the solution found in the previous generation. If $l - 1 = 0$, $\mathbf{u}_1$ is also randomly chosen.

b) Set the inner loop iteration index $t = 0$ and the initial weightings $d_i^{(t)} = (1/s)$ for $1 \leq i \leq s$.

c) For $1 \leq i \leq s$, generate $\mathbf{g}_k^{(i)}$ from $\mathbf{u}_i$, the s candidates for the kth model column, and orthogonalize them

$$\alpha_{j,k}^{(i)} = \frac{\mathbf{p}_j^T \mathbf{g}_k^{(i)}}{\mathbf{p}_j^T \mathbf{p}_j} \quad \text{for} \quad 1 \leq j < k$$
$$\mathbf{p}_k^{(i)} = \mathbf{g}_k^{(i)} - \sum_{j=1}^{k-1} \alpha_{j,k}^{(i)} \mathbf{p}_j.$$

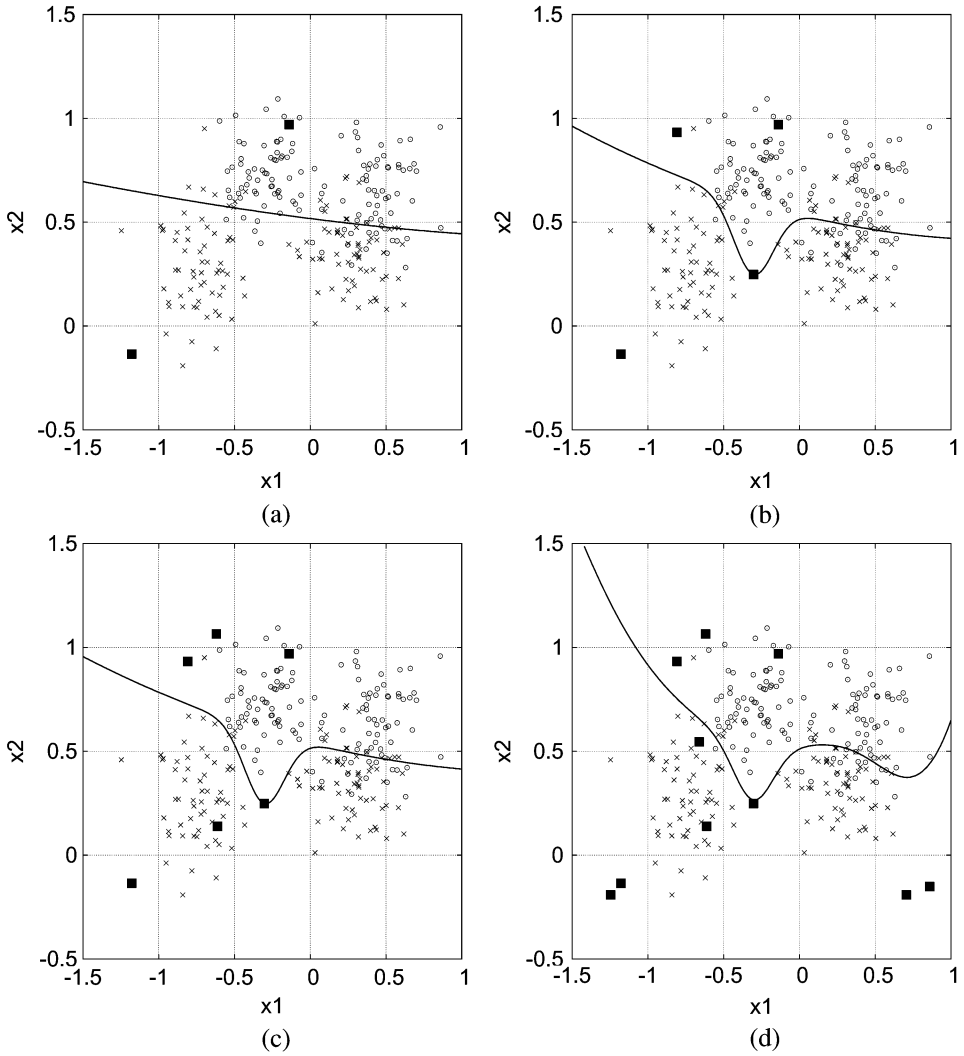

Fig. 3. Synthetic data: (a) decision boundary of the two-term classifier, (b) decision boundary of the four-term classifier, (c) decision boundary of the six-term classifier, and (d) decision boundary of the ten-term classifier. The 250 samples of the training data are shown as crosses and circles for the two classes, respectively, and the kernel centers are depicted by the dark-filled squares.

d) For $1 \leq i \leq s$, calculate the loss of each point, namely

$$J_k^{(i)} = \frac{1}{F_k^{(i)}}$$

where $F_k^{(i)}$ denotes the Fisher ratio (11) calculated in the direction of $\mathbf{p}_k^{(i)}$.

B. Inner Loop: Weighted Search Routine

Step 1) Boosting.

1) Find

$$\mathbf{u}_{\text{best}} = \arg\min\{J_k^{(i)}, 1 \leq i \leq s\}$$
$$\mathbf{u}_{\text{worst}} = \arg\max\{J_k^{(i)}, 1 \leq i \leq s\}.$$

2) Normalize the loss

$$\bar{J}_k^{(i)} = \frac{J_k^{(i)}}{\sum_{l=1}^{s} J_k^{(l)}}, 1 \leq i \leq s.$$

3) Compute a weighting factor β_t according to

$$\beta_t = \frac{\epsilon_t}{1-\epsilon_t} \text{ with } \epsilon_t = \sum_{i=1}^{s} d_i^{(t)} \bar{J}_k^{(i)}.$$

4) Update the weighting vector

$$d_i^{(t+1)} = \begin{cases} d_i^{(t)} \beta_t^{\bar{J}_k^{(i)}} & \text{for } \beta_t \leq 1 \\ d_i^{(t)} \beta_t^{1-\bar{J}_k^{(i)}} & \text{for } \beta_t > 1, \end{cases} \quad 1 \leq i \leq s.$$

5) Normalize the weighting vector

$$d_i^{(t+1)} = \frac{d_i^{(t+1)}}{\sum_{l=1}^{s} d_l^{(t+1)}}, 1 \leq i \leq s.$$

Step 2) Parameter updating.

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

$$\mathbf{u}_{s+1} = \sum_{i=1}^{s} d_i^{(t+1)} \mathbf{u}_i.$$

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

$$\mathbf{u}_{s+2} = \mathbf{u}_{\text{best}} + (\mathbf{u}_{\text{best}} - \mathbf{u}_{s+1}).$$

3) Orthogonalize these two candidate model columns and compute their losses.

4) Choose a better (smaller loss value) point from $\mathbf{u}_{s+1}$ and $\mathbf{u}_{s+2}$ to replace $\mathbf{u}_{\text{worst}}$.

TABLE I
OFS PROCEDURE WITH BOOSTING FOR THE SYNTHETIC DATA SET

step k	mean vector $\boldsymbol{\mu}_k$		diagonal covariance $\boldsymbol{\Sigma}_k$		weight w_k	Fisher ratio F_k
1	-1.17849e+0	-1.35250e-1	8.48630e+1	6.49353e+0	-1.35897e+1	2.07539e+0
2	-1.39026e-1	9.70182e-1	5.78028e+1	2.03028e+1	5.10463e+1	2.18588e+0
3	-8.07991e-1	9.32838e-1	8.83254e+1	7.99372e+1	-3.79584e+1	9.85976e-3
4	-3.03289e-1	2.48324e-1	1.70513e-2	9.97627e+1	7.84764e-1	9.65163e-2
training error rate 12.8% and testing error rate 8.1%						

Repeat from Step 1) with $t = t + 1$ until the $(s+1)$th value changes very little compared with the last round or a preset maximum number of iterations has been reached.

C. End of Inner Loop

From the converged population of s points, find

$$i_k = \arg\min\{J_k^{(i)}, 1 \le i \le s\}$$

and select $\alpha_{j,k} = \alpha_{j,k}^{(i_k)}, 1 \le j < k$ and

$$\mathbf{p}_k = \mathbf{p}_k^{(i_k)} = \mathbf{g}_k^{(i_k)} - \sum_{j=1}^{k-1} \alpha_{j,k} \mathbf{p}_j.$$

This determines the solution of the lth generation, denoted as $\mathbf{u}_{\text{best}}^{(l)}$. Repeat from outer loop until $l = L_{\max}$.

D. End of Outer Loop

This determines the kth kernel's mean vector and diagonal covariance matrix or selects the kth kernel term.

The important algorithmic parameters that need to be chosen appropriately are the population size s and the number of generations $L_{\max}$. The population size depends on the dimension of $\mathbf{u}$ and the objective function to be optimized. This is very similar to the choice of population size in the genetic algorithm. The number of generations should be chosen sufficiently large for the algorithm to search for a global minimum but not too large, which may incur unnecessary computation. Again, the appropriate value for $L_{\max}$ depends on the dimension of $\mathbf{u}$ and how hard is the objective function to be optimized. Also the choice of s has some influence on the choice of $L_{\max}$. Generally, these two algorithmic parameters have to be found empirically.

IV. EXPERIMENTAL RESULTS

The synthetic two-class problem and Diabetes in Pima Indians, taken from [14], were used to investigate the proposed OFSwB algorithm.[2]

A. Synthetic Data

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 around 8%. With a population size $s = 21$ and the number of generations $L_{\max} = 20$, we applied the OFSwB algorithm to the 250-sample training set. Fig. 2 depicts the training and test error rates versus the size of the selected classifier. The decision boundaries of the two-term, four-term, six-term, and ten-term classifiers are illustrated in Fig. 3(a)–(d), respectively. The decision boundary of the eight-term classifier, not shown here, is almost identical to that of the four-term classifier. The result of Fig. 2 indicates that the four-term classifier is sufficient, and the selection procedure for this four-term classifier is summarized in Table I. Note that the four-term classifier constructed by the OFSwB algorithm achieved the optimal Bayes classification performance. Tipping [3] applied the SVM and RVM to this data set and only used 100 random selected samples from the 250-points training data set in training. The results given in [3] are compared with our result in Table II.

[2]The data sets were obtained from http://www.stats.ox.ac.uk/PRNN/

TABLE II
COMPARISON OF CLASSIFICATION FOR THE SYNTHETIC DATA SET

	SVM	RVM	OFSwB
classifier size	38	4	4
test error rate	10.6%	9.3%	8.1%

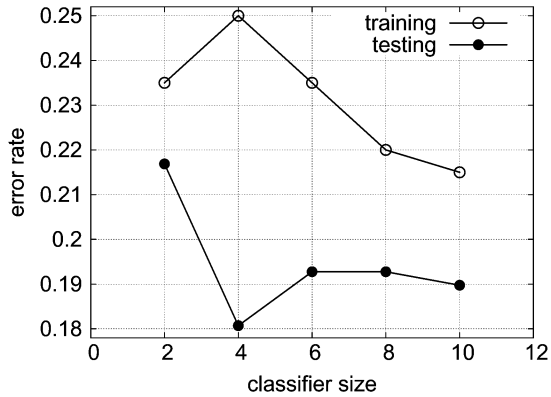

Fig. 4. Pima Diabetes data: training and test error rates versus size of selected classifier.

TABLE III
COMPARISON OF CLASSIFICATION FOR THE PIMA DIABETES DATA SET

	SVM	RVM	OFSwB
classifier size	109	4	4
test error rate	20.1%	19.6%	18.1%

B. Pima Diabetes Data

The dimension of the input space was $m = 7$, the training data set contained 200 samples and the test data set had 332 samples. With a population size $s = 61$ and the number of generations $L_{\max} = 20$ for the OFSwB algorithm, Fig. 4 shows the training and test error rates versus the size of the selected classifier, which clearly indicates that a four-term classifier is sufficient. Table III compares the performance of the selected four-term classifier with those obtained by the SVM and RVM methods, quoted from [3].

V. CONCLUSION

A novel algorithm has been proposed for the construction of parsimonious kernel classifiers using the orthogonal forward selection with boosting based on Fisher ratio for class separability measure. The proposed algorithm has the ability to tune both the mean vector and diagonal covariance matrix of individual kernel to incrementally maximize Fisher ratio for class separability measure. A weighted optimization

search method has been developed based on boosting to append classifier kernels one by one in an orthogonal forward regression procedure. Experimental results presented have demonstrated the effectiveness of the proposed technique.

REFERENCES

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

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

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

[4] K. Z. Mao, "RBF neural network center selection based on Fisher ratio class separability measure," *IEEE Trans. Neural Netw.*, vol. 13, no. 5, pp. 1211–1217, 2002.

[5] S. Chen, L. Hanzo, and A. Wolfgang, "Kernel-based nonlinear beamforming construction using orthogonal forward selection with Fisher ratio class separability measure," *IEEE Signal Process. Lett.*, vol. 11, no. 5, pp. 478–481, 2004.

[6] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *J. Comput. Syst. Sci.*, vol. 55, no. 1, pp. 119–139, 1997.

[7] R. E. Schapire, "The strength of weak learnability," *Machine Learn.*, vol. 5, no. 2, pp. 197–227, 1990.

[8] R. Meir and G. Rätsch, "An introduction to boosting and leveraging," in *Advanced Lectures in Machine Learning*, S. Mendelson and A. Smola, Eds. Berlin, Germany: Springer Verlag, 2003, pp. 119–184.

[9] R. O. Duda and P. E. Hart, *Pattern Classification and Scene Analysis*. New York: Wiley, 1973.

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

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

[12] L. Ingber, "Simulated annealing: Practice versus theory," *Math. Comput. Model.*, vol. 18, no. 11, pp. 29–57, 1993.

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

[14] B. D. Ripley, *Pattern Recognition and Neural Networks*. Cambridge, U.K.: Cambridge Univ. Press, 1996.

$O(\log_2 M)$ Self-Organizing Map Algorithm Without Learning of Neighborhood Vectors

Hiroki Kusumoto and Yoshiyasu Takefuji

***Abstract*—In this letter, a new self-organizing map (SOM) algorithm with computational cost $O(\log_2 M)$ is proposed where M^2 is the size of a feature map. The first SOM algorithm with $O(M^2)$ was originally proposed by Kohonen. The proposed algorithm is composed of the subdividing method and the binary search method. The proposed algorithm does not need the neighborhood functions so that it eliminates the computational cost in learning of neighborhood vectors and the labor of adjusting the parameters of neighborhood functions. The effectiveness of the proposed algorithm was examined by an analysis of codon frequencies of *Escherichia coli (E. coli)* K12 genes. These drastic computational reduction and accessible application that requires no adjusting of the neighborhood function will be able to contribute to many scientific areas.**

***Index Terms*—Binary search, computational reduction, codon frequency, *Escherichia coli (E. coli)*, neighborhood function, self-organizing map (SOM), subdividing method.**

I. INTRODUCTION

A self-organizing map (SOM) algorithm is one of unsupervised learning methods in the artificial neural network in order to map a multidimensional input data set into two-dimensional (2-D) space according to the neighborhood function. The first SOM algorithm was originally developed by Kohonen [1] and has been used in a variety of research areas including speech or speaker recognition [2], mathematics [3], financial analysis [4], color quantization [5], identification and control of dynamical systems [6], color clustering [7], and bioinformatics [8]–[10]. Particularly in the field of bioinformatics, many researchers have adopted SOM algorithm for analysis of gene sequences as a method of clustering, visualization, or feature extraction. Wang *et al.* clustered genes according to codon usage by SOM algorithm in order to identify highly expressed and horizontally transferred genes [8]. Sultan *et al.* and Gill *et al.* applied SOM algorithm to analyze microarray data [9], [10].

When M^2 is the size of a feature map, the number of compared weight vectors for one input vector to search a winner vector by exhaustive search is equivalent to M^2. Tree-structured SOM proposed by Koikkalainen and Oja [11] and Truong [12] to improve the winner search reduces the number of searching operations to $O(M \log M)$. Kohonen proposed a new method with the total number of comparison operations by $O(M)$ [1]. Self-organizing topological tree with $O(\log M)$ was proposed by Xu and Chang [13].

In this letter, a new SOM algorithm with $O(\log_2 M)$ is proposed where it is composed of the subdividing method and the binary search method. The proposed algorithm not only reduces the computational costs but also eliminates the time-consuming parameter tuning in the neighborhood function in SOM applications. When we use SOM for practical analyses, one of the most time-consuming tasks for effective learning is to adjust the values of several parameters, particularly in neighborhood function. In addition to that, the neighborhood function has a critical effect on the performance of SOM. In the proposed algorithm, only winner vectors are trained. The proposed algorithm not to train neighborhood vectors is completely original.

Manuscript received May 4, 2005; revised April 3, 2006.

The authors are with the Keio University, Kanagawa 252-8520, Japan (e-mail: kusu@sfc.keio.ac.jp).

Digital Object Identifier 10.1109/TNN.2006.882370

1045-9227/$20.00 © 2006 IEEE