Elastic-Net Prefiltering for Two-Class Classification

Xia Hong, Sheng Chen, *Fellow, IEEE*, and Chris J. Harris

***Abstract*—A two-stage linear-in-the-parameter model construction algorithm is proposed aimed at noisy two-class classification problems. The purpose of the first stage is to produce a prefiltered signal that is used as the desired output for the second stage which constructs a sparse linear-in-the-parameter classifier. The prefiltering stage is a two-level process aimed at maximizing a model's generalization capability, in which a new elastic-net model identification algorithm using singular value decomposition is employed at the lower level, and then, two regularization parameters are optimized using a particle-swarm-optimization algorithm at the upper level by minimizing the leave-one-out (LOO) misclassification rate. It is shown that the LOO misclassification rate based on the resultant prefiltered signal can be analytically computed without splitting the data set, and the associated computational cost is minimal due to orthogonality. The second stage of sparse classifier construction is based on orthogonal forward regression with the D-optimality algorithm. Extensive simulations of this approach for noisy data sets illustrate the competitiveness of this approach to classification of noisy data problems.**

***Index Terms*—Cross-validation (CV), elastic net (EN), forward regression, leave-one-out (LOO) errors, linear-in-the-parameter model, regularization.**

I. Introduction

In most supervised learning algorithms using input/output data sets, system input/output mappings are constructed using parametric models, such as neural networks, kernel regression, and classification models. The two-class classification problem can be configured into a regression framework that solves a separating hyperplane for the two classes, with the known class labels being used as the system output examples for model training. Models are identified according to some objective criteria; parsimonious models are preferable in engineering applications since a models' computational complexity scales with its model complexity. Moreover, a parsimonious model is easier to interpret from the viewpoint of knowledge extraction. Consequently, a practical nonlinear modeling principle is to find the smallest model that generalizes well, i.e., having the capability to accurately approximate the system output for unseen input data. Fundamental to the evaluation of model generalization capability is the concept of cross-validation (CV) [1], which can be used either in parameter estimation (e.g., tuning regularization parameter [2], [3] and forming new parameter estimates [4]) or to derive model selection criteria based on information theoretic principles [5], which regularizes model structure. Note that information-based criteria of model generalization, such as the Akaike information criterion [6], often include a penalty term to avoid an oversized model which may tend to overfit to the training data set.

Modeling techniques using model construction/selection ideas have been widely studied, e.g., support vector machine (SVM), relevance vector machine, and orthogonal forward regression (OFR) [7]–[10]. The highly utilized orthogonal least squares algorithm [11] was developed as a practical linear-in-the-parameter model construction algorithm. A large class of nonlinear representations, e.g., radial basis function (RBF) networks and SVM, can be classified as the linear-in-the-parameter models. The orthogonal forward selection (OFS) procedure can be applied to construct parsimonious two-class classifiers incrementally by maximizing the Fisher ratio of class separability measure [12], [13] or by minimizing misclassification rate [14].

The regularization-assisted OLS approaches have been proposed based on minimizing the leave-one-out (LOO) criteria for regression, classification, and probability density estimation [15]. In particular, each RBF unit has a tunable center vector as well as an adjustable diagonal covariance matrix [15]. Specifically, at each forward-regression stage of the model construction procedure, one RBF unit's center vector and diagonal covariance matrix are optimized using a particle-swarm-optimization (PSO) algorithm. The PSO [16], [17] constitutes a population-based stochastic optimization technique, which was inspired by the social behavior of bird flocks or fish schools. The algorithm commences with random initialization of a swarm of individuals, referred to as particles, within the specific problem's search space. It then endeavors to find a globally optimum solution by gradually adjusting the trajectory of each particle toward its own best location and toward the best position of the entire swarm at each optimization step. The PSO method is popular, owing to its simplicity in implementation and ability to rapidly converge to a "reasonably good" solution and to "steer clear" of local minima. The PSO has been successfully applied to wide-ranging optimization problems [18]–[22].

Regularization methods are developed to carry out parameter estimation and model structure selection simultaneously [23], [24]. It has been shown [25], [26] that l^2 norm parameter regularization is equivalent to a maximized *a posteriori* probability estimate of parameters from Bayesian viewpoint by adopting a Gaussian prior for parameters. The regularization [2], [3] uses a penalty function on l^2 norms of the parameters. From the powerful Bayesian learning viewpoint, the l^2 norm

Manuscript received October 4, 2011; revised January 10, 2012, March 22, 2012, May 23, 2012, and May 31, 2012; accepted June 19, 2012. Date of publication July 18, 2012; date of current version January 11, 2013. This work was supported by the Engineering and Physical Sciences Research Council in the U.K. This paper was recommended by Editor X. Wang.

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

S. Chen is with the School of Electronics and Computer Science, University of Southampton, SO17 1BJ Southampton, U.K., and also with the Faculty of Engineering, King Abdulaziz University, Jeddah 21589, Saudi Arabia (e-mail: sqc@ecs.soton.ac.uk).

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

Digital Object Identifier 10.1109/TSMCB.2012.2205677

regularization parameter is equivalent to the ratio of the related hyperparameter to the noise parameter, lending to an iterative evidence procedure for solving the optimal regularization parameters [23], [26].

Alternatively, the model sparsity can be achieved by minimizing the l^1 norm of the parameters. The l^1 norm minimization is fundamental to the basis pursuit or least absolute shrinkage and selection operator (LASSO) [27], [28]. The least angle regression (LAR) procedure [29] is developed for solving the problem efficiently. The Bayesian interpretation for LASSO is simply by adopting a Laplacian prior for parameters. The advantage of LASSO is that it can achieve much sparser models by forcing more parameters to zero than models derived from the minimization of the l^p norm (as most l^p norms will produce small, but nonzero, parameter values). Unfortunately, introducing the nondifferentiable l^1 norm in the cost function introduces difficulties of model parameter estimation and finding an appropriate l^1 regularizer. Another disadvantage of using l^1 optimization is that a group of correlated terms cannot be selected together, which is not desirable if model interpretability is important. Alternatively, the use of l^2 will improve model generalization but cannot be used for model selection by itself. Combining a locally regularized orthogonal least squares model selection [30] with D-optimality experimental design enhances model robustness [25].

Recently, a promising concept, the elastic net (EN), has been proposed by minimizing both the l^1 and l^2 norms of the model parameters [24]. The EN retains the model sparsity of LASSO, while strongly correlated terms tend to be in or out of the model together. It is shown [24] that the EN problem can be transformed into an equivalent LASSO problem on augmented data, based on which the LAR procedure is applicable [24]. Now, as there are two regularization parameters in the EN, the CV has to be performed over a 2-D space. The tenfold CV was used [24] in choosing two regularization parameters by searching over a grid of l^2 norm regularization parameter values. Specifically for each fixed l^2 norm regularization parameter, the algorithm LAR produces the entire solution path of the EN, which is used to select l^1 norm regularization parameter by tenfold CV. Clearly, this may not yield the optimal parameters if the grid search is set at a coarse level, but increasing the grid search at a very fine level would inevitably increase the computational cost. It would be desirable that the two regularization parameters can be optimized simultaneously based on CV as well as in an efficient manner.

In this paper, we propose a novel two-stage linear-in-the-parameter classifier construction algorithm for a two-class noisy classification problem in order to avoid training data overfitting. The fundamental idea is that a sparse classifier is constructed using a prefiltered signal, rather than the original class label vector, as the desired output. The EN regularization is applied to produce the prefiltered signal in the first stage. A two-level algorithm is introduced, aimed at maximizing a model's generalization capability. At the lower level, a new EN model identification algorithm is employed based on significant eigenvectors using the regression matrix, and the two regularization parameters are optimized using a PSO algorithm at the upper level by minimizing the LOO misclassification rate using the prefiltered signal. It is shown that the LOO misclassification rate can be analytically computed without actually splitting the data set, and the associate computation cost is minimal due to the orthogonality. The second stage of sparse classifier construction is based on OFR with D-optimality algorithm [10].

This paper is organized as follows. Section II formulates the proposed two-stage two-class classifier construction algorithm. In Section III, we introduced the LOO misclassification rate formula based on the resultant prefiltered signal, which is used as the metric for optimizing the two EN regularization parameters using PSO. This completes the derivation of the proposed two-level learning algorithm for the first stage of classifier construction. In Section IV, the simulation results have been employed to demonstrate the effectiveness of proposed approaches, leading to a discussion on the merits of this algorithm. Finally, some conclusions are given in Section V.

II. Two-Stage Classifier Using EN Prefiltering

In this section, we initially outline the concept of linear-in-the-parameter classifier and then introduce the proposed construction algorithm for a two-stage classifier, as shown in Fig. 1. The proposed model construction algorithm includes stage one of initial generation of the prefiltered signal using EN regularization based on singular value decomposition (SVD), followed by stage two of a two-class classifier construction using the OFR with D-optimality algorithm [10].

A. Linear-in-the-Parameter Classifier

Consider an approximately balanced two-class noisy training data set $D_N = \{\mathbf{x}(k), y(k)\}_{k=1}^N$, in which $y(k) \in \{1, -1\}$ denotes the class type for each data sample $\mathbf{x}(k) \in \Re^n$. Let a linear-in-the-parameter classifier $f(\mathbf{x}) : \Re^n \to \{1, -1\}$ be formed using the data set, given by

$$\hat{y}(k) = \operatorname{sgn}\left(f(k)\right), \qquad \text{with } f(k) = \sum_{i=1}^{L} \theta_i \phi_i\left(\mathbf{x}(k)\right) \tag{1}$$

with

$$\operatorname{sgn}(s) = \begin{cases} 1, & \text{if } s \geq 0 \\ -1, & \text{if } s < 0 \end{cases} \tag{2}$$

where $\phi_i(\bullet)$ denotes the classifier kernels with a known nonlinear basis function, such as RBF. θ_i denotes the model parameters, and L is the number of regressors (kernels). We initially consider an overparameterized model where L classifier kernels may be constructed using all or part of the training data set as centers for RBF kernels. $\hat{y}(k)$ is the model-predicted class label for $\mathbf{x}(k)$. By letting $\boldsymbol{\phi}_i = [\phi_i(\mathbf{x}(1)), \ldots, \phi_i(\mathbf{x}(N))]^{\mathrm{T}}$, for $1 \leq i \leq L$, and defining

$$\mathbf{y} = \begin{bmatrix} y(1) \\ \vdots \\ y(N) \end{bmatrix} \qquad \boldsymbol{\Phi} = [\,\boldsymbol{\phi}_1, \ldots, \boldsymbol{\phi}_L\,]$$

$$\boldsymbol{\theta} = \begin{bmatrix} \theta_1 \\ \vdots \\ \theta_L \end{bmatrix} \qquad \mathbf{e} = \begin{bmatrix} e(1) \\ \vdots \\ e(N) \end{bmatrix}$$

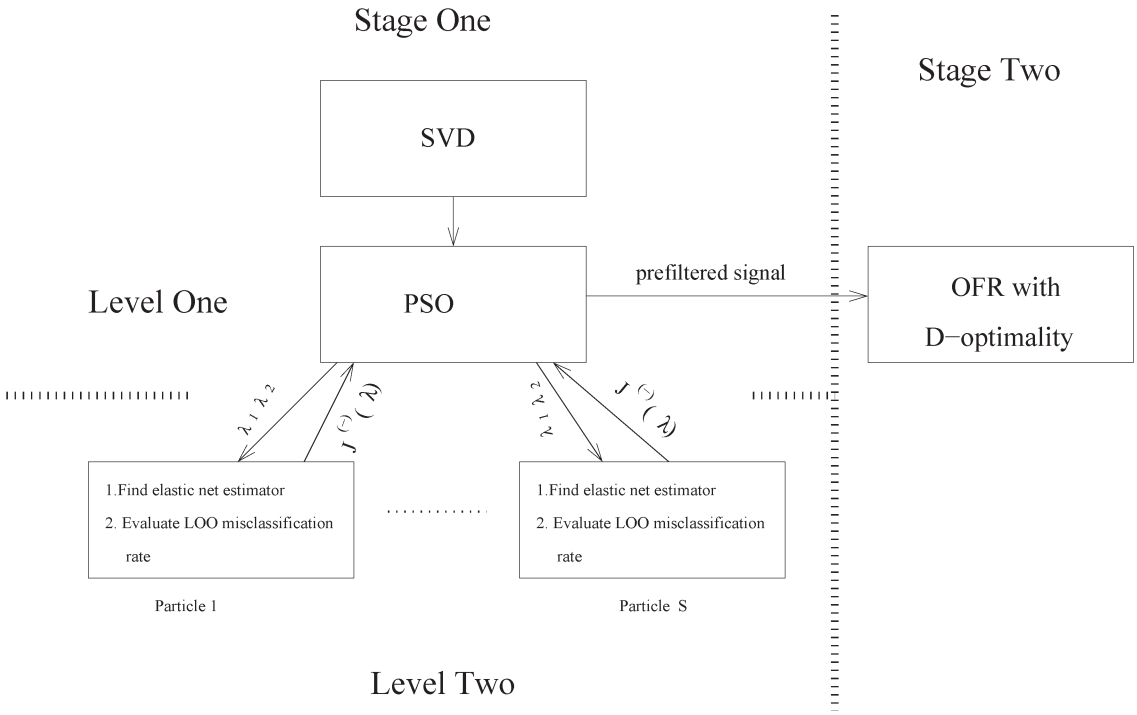

Fig. 1. Schematic diagram of two-stage classifier using EN prefiltering.

the regression model (1) can be written in the matrix form

$$\mathbf{y} = \mathbf{\Phi}\boldsymbol{\theta} + \mathbf{e}. \tag{3}$$

Geometrically, the hyperplane defined by

$$\sum_{i=1}^{L} \theta_i \phi_i(\mathbf{x}) = 0 \tag{4}$$

divides the data into two classes.

B. Prefiltering Using SVD-Based EN Regularization

The aim of prefiltering is to define a robust classification boundary over the training data set which can be used for classifier construction as target. Consider the SVD $\mathbf{\Phi} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^{\mathrm{T}}$, where $\mathbf{\Sigma} = \mathrm{diag}[s_1, \ldots, s_{n_s}, 0, \ldots, 0] \in \Re^{L\times L}$, $s_1 \geq s_2 > \cdots \geq s_{n_s} > 0$, are the resultant n_s nonzero singular values. $\mathbf{U} = [\mathbf{u}_1, \ldots, \mathbf{u}_L] \in \Re^{N\times L}$, and $\mathbf{V} = [\mathbf{v}_1, \ldots, \mathbf{v}_L] \in \Re^{L\times L}$, satisfying $\mathbf{U}^{\mathrm{T}}\mathbf{U} = \mathbf{I}_L$ and $\mathbf{V}^{\mathrm{T}}\mathbf{V} = \mathbf{I}_L$, in which $\mathbf{I}_p$ denotes the p-dimensional identity matrix. The regression model (3) can alternatively be expressed as

$$\mathbf{y} = \mathbf{U}_r\mathbf{g} + \mathbf{e} \tag{5}$$

where $\mathbf{U}_r = [\mathbf{u}_1, \ldots, \mathbf{u}_{n_s}] \in \Re^{N\times n_s}$ and $\mathbf{g} = [g_1, \ldots, g_{n_s}]^{\mathrm{T}}$. Denote the row vectors of $\mathbf{U}_r$ as $\mathbf{u}_r(k)$, $k = 1, \ldots, N$. Clearly $\mathbf{U}_r^{\mathrm{T}}\mathbf{U}_r = \mathbf{I}_{n_s}$. The SVD maps the original L-dimensional space spanned by $\mathbf{\Phi}$ to a low-dimensional space spanned by $\mathbf{U}_r$ which represents the true dimension in the sense that $\mathbf{\Phi} = \sum_{i=1}^{n_s} s_i \mathbf{u}_i \mathbf{v}_i^{\mathrm{T}}$.

We note that solving (4) by minimizing $\|\mathbf{y} - \mathbf{\Phi}\boldsymbol{\theta}\|^2$ is an ill-posed problem; thus, some structural regularization is needed to emphasize the smoothness of the decision boundary in order to avoid overfitting to the noise. For example, for any fixed positive λ_1 and λ_2, the naive EN (NEN) criterion is defined as [24]

$$L(\lambda_1, \lambda_2, \boldsymbol{\theta}) = \|\mathbf{y} - \mathbf{\Phi}\boldsymbol{\theta}\|^2 + \lambda_2\|\boldsymbol{\theta}\|^2 + \lambda_1\|\boldsymbol{\theta}\|_1 \tag{6}$$

where $\|\bullet\|$ denotes the Euclidean norm and $\|\boldsymbol{\theta}\|_1 = \sum_{i=1}^{L} |\theta_i|$. The NEN estimator is the minimizer of

$$\hat{\boldsymbol{\theta}}_{\mathrm{NEN}} = \arg\min_{\boldsymbol{\theta}} \{L(\lambda_1, \lambda_2, \boldsymbol{\theta})\}. \tag{7}$$

This can be transformed into an equivalent LASSO problem on augmented data, based on which the LAR procedure is applicable, referred to as LAR-EN [24]. The EN has some desirable properties, as it maintains the model sparsity of LASSO, but is not as aggressive as LASSO in excluding correlated terms in the model. This is because these terms tend to be in or out of the model together as a result of the l^2 norm regularization [24]. Note that there is no analytical solution to (6) unless the model terms are orthogonal.

In this paper, we propose to apply the following SVD-based EN criterion based on (5)

$$L_e(\lambda_1, \lambda_2, \mathbf{g}) = \|\mathbf{y} - \mathbf{U}_r\mathbf{g}\|^2 + \lambda_2\|\mathbf{g}\|^2 + \lambda_1\|\mathbf{g}\|_1. \tag{8}$$

It makes sense to find a robust classification boundary in the lower dimensional latent space via SVD to divide the noisy two-class data sets. There is also a clear computational advantage in that the NEN solution for $\mathbf{g}$ can be obtained by setting the subderivative $\partial L_e/\partial \mathbf{g} = \mathbf{0}$, i.e.,

$$\mathbf{U}_r^{\mathrm{T}}\mathbf{y} - \frac{\lambda_1}{2}\mathrm{sign}(\mathbf{g}) = (1 + \lambda_2)\mathbf{g} \tag{9}$$

where $\text{sign}(\mathbf{g}) = [\text{sign}(g_1), \ldots, \text{sign}(g_{n_s})]^{\text{T}}$, with

$$\text{sign}(s) \begin{cases} = & 1, & \text{if} \quad s > 0 \\ = & -1, & \text{if} \quad s < 0 \\ \in & [-1, 1], & \text{if} \quad s = 0. \end{cases} \tag{10}$$

The solution of (9) is given by

$$g_i^{(\text{NEN})} = \left(\frac{1}{1+\lambda_2} \left| g_i^{(LS)} \right| - \frac{\frac{\lambda_1}{2}}{1+\lambda_2} \right)_+ \text{sign}\left(g_i^{(LS)} \right) \tag{11}$$

with $g_i^{(LS)} = \mathbf{u}_i^{\text{T}} \mathbf{y}$, $i = 1, \ldots, n_s$, and

$$z_+ = \begin{cases} z, & \text{if } z > 0 \\ 0, & \text{if } z \leq 0. \end{cases} \tag{12}$$

Note that the cost function (8) contains sparsity inducing l^1 norm so that some parameters $g_i^{(NEN)}$ will be zeros, producing a sparse model containing $n_m (\ll n_s)$ significant singular vectors. Let $\mathbf{g}^{(\text{NEN})} = [\tilde{g}_1^{(\text{NEN})}, \ldots, \tilde{g}_{n_m}^{(\text{NEN})}] \in \Re^{n_m}$, consisting of all nonzero parameters, and let the submatrix of $\mathbf{U}_r$ consisting of columns corresponding to nonzero parameters be denoted by $\mathbf{U}_s = [\tilde{\mathbf{u}}_1, \ldots, \tilde{\mathbf{u}}_{n_m}] \in \Re^{N \times n_m}$. We construct a prefiltered signal using $\mathbf{y}_{\text{pre}} = [y_{\text{pre}}(1), \ldots, y_{\text{pre}}(N)]^{\text{T}} = \mathbf{U}_s \mathbf{g}^{(\text{NEN})}$. This means that dimension is further reduced in latent space by eliminating any term with $|g_i^{(LS)}|$ less than threshold $\lambda_1/2$. Since $|g_i^{(LS)}|$, which is subject to noise in estimation, directly measures the correlation between each singular vector and the noisy system output, this means that, if λ_1 is appropriately chosen to noise level, we can significantly reduce error propagation from the noisy system output via model parameters, thus producing smoother decision boundary.

Instead of thresholding by λ_1, the effect of λ_2 scales down parameters by multiplication, which offers another degree of freedom in controlling the parameter variance. In the original EN procedure [24], a double-shrinkage problem has been noted empirically, so a rescaling step was applied to the solutions obtained by (7). Clearly in the case of minimizing (6) where the model bases are correlated, the resultant sparse model terms are dependent on the value of λ_2; hence, the effect of λ_2 lies in the terms selected into the model as well as model parameters. However, as our proposed criterion (8) is based on orthogonal space, this means that the rescaling step, if applied, would be equivalent to setting $\lambda_2 = 0$. In order to have more flexibility in regularization control, we opt to use the so-called NEN solution, which includes $\lambda_1 = 0$ or $\lambda_2 = 0$ as special case. An efficient procedure aimed at optimizing the two regularization parameters based on CV is introduced in Section III that is different from any existent approaches.

C. Sparse Classifier Construction Using OFR With D-Optimality

The aim of sparse classifier construction stage is to identify a linear-in-the-parameter classifier as described in Section II-A, yet with good approximation and, simultaneously, a sparse representation containing only a small number of kernels. The advantages of parsimonious models are that they are computationally more efficient and easier to interpret in physical applications. We note that, although $\mathbf{y}_{\text{pre}}$ obtained previously defines a classification boundary in the latent space via SVD and can be used to generate predicted labels over the training data set, it cannot be directly used as a classifier for unseen data samples nor does it lead to a sparse kernel classifier, because each singular vector $\mathbf{u}_i$ is a linear combination of all the kernels $\phi_i(\mathbf{x}(k))$.

In order to construct a kernel classifier $f(k)$ with a minimum number of classifier kernels, the prefiltered signal $y_{\text{pre}}(k)$, $k = 1, \ldots, N$, is used as the desired output. This section outlines the OFR (OFS) with D-optimality algorithm [10], via which the classifier kernels are selected into the classifier via a forward-regression manner. The OFR, often based on the modified Gram–Schmidt (MGS) procedure, is an efficient method incorporating structure selection and parameter estimation simultaneously. The D-optimality is a model structure robustness criterion in experimental design to tackle ill conditioning in model structure.

Specifically, we can write a regression equation linking $y_{\text{pre}}(k)$ and $f(k)$ as

$$\begin{aligned} y_{\text{pre}}(k) &= f(k) + \varepsilon\left(\mathbf{x}(k)\right) \\ &= \sum_{i=1}^{L} \theta_i \phi_i\left(\mathbf{x}(k)\right) + \varepsilon\left(\mathbf{x}(k)\right) \end{aligned} \tag{13}$$

where $\varepsilon(\mathbf{x}(k))$ is the modeling error at $\mathbf{x}(k)$ between the proposed two-class kernel classifier and the prefiltered signal. Because the target $y_{\text{pre}}(k)$ is smooth and free of noise, $E[\varepsilon^2(\mathbf{x}(k))]$ should be just the approximation error which should be much smaller than $E[y_{\text{pre}}^2(\mathbf{x}(k))]$. It can then be assumed that the classification performance of the final optimal sparse model classifier $f(k)$ is close to that of the prefiltered signal $y_{\text{pre}}(k)$. For example, unless

$$\left|\varepsilon\left(\mathbf{x}(k)\right)\right| > |f(k)| \qquad \text{sgn}\left[\varepsilon\left(\mathbf{x}(k)\right)\right] \neq \text{sgn}\left[f(k)\right] \tag{14}$$

which is unlikely, the predicted class label based on the sparse classifier $f(k)$ should be the same as that of $y_{\text{pre}}(k)$.

Letting $\boldsymbol{\varepsilon} = [\varepsilon(\mathbf{x}(1)), \ldots, \varepsilon(\mathbf{x}(N))]^{\text{T}}$, the regression model (13) can be written in the matrix form

$$\mathbf{y}_{\text{pre}} = \boldsymbol{\Phi}\boldsymbol{\theta} + \boldsymbol{\varepsilon}. \tag{15}$$

The orthogonal decomposition of the matrix $\boldsymbol{\Phi}$ is

$$\boldsymbol{\Phi} = \mathbf{W}\mathbf{A} \tag{16}$$

where

$$\mathbf{A} = \begin{bmatrix} 1 & a_{1,2} & \cdots & a_{1,L} \\ 0 & 1 & \ddots & \vdots \\ \vdots & \ddots & \ddots & a_{L-1,L} \\ 0 & \cdots & 0 & 1 \end{bmatrix} \tag{17}$$

$$\mathbf{W} = [\mathbf{w}_1, \ldots, \mathbf{w}_L] \tag{18}$$

with columns satisfying $\mathbf{w}_i^{\text{T}} \mathbf{w}_j = 0$, if $i \neq j$. The regression model (15) can alternatively be expressed as

$$\mathbf{y}_{\text{pre}} = \mathbf{W}\boldsymbol{\Gamma} + \boldsymbol{\varepsilon} \tag{19}$$

where the orthogonal weight vector $\mathbf{\Gamma} = [\gamma_1, \ldots, \gamma_L]^{\mathrm{T}}$. The OFS with D-optimality algorithm adopts the following least squares criterion:

$$J(\mathbf{\Gamma}) = \boldsymbol{\varepsilon}^{\mathrm{T}}\boldsymbol{\varepsilon} \tag{20}$$

to estimate $\mathbf{\Gamma}$. $\mathbf{A}\boldsymbol{\theta} = \mathbf{\Gamma}$ can then be used to determine model parameters $\boldsymbol{\theta}$, given $\mathbf{A}$ and $\mathbf{\Gamma}$.

The OFR selects model terms one at a time with the final model consisting of $\mathbf{\Phi}_s$, a submatrix consisting of $\bar{n}_s$ columns selected from $\mathbf{\Phi}$. The D-optimality is defined as $\max \det\{\mathbf{\Phi}_s^{\mathrm{T}}\mathbf{\Phi}_s\}$. Since $\det\{\mathbf{\Phi}_s^{\mathrm{T}}\mathbf{\Phi}_s\} = \det\{\mathbf{W}_s^{\mathrm{T}}\mathbf{W}_s\} = \prod_{l=1}^{\bar{n}_s} \mathbf{w}_l^{\mathrm{T}}\mathbf{w}_l$, where $\mathbf{W}_s$ is the resultant orthogonal matrix from $\mathbf{\Phi}_s$ [as in (16)], the combined error reduction ratio defined as

$$[cerr]_l = \frac{\left(\mathbf{w}_l^{\mathrm{T}}\mathbf{w}_l\gamma_l^2 + \beta \log\left(\mathbf{w}_l^{\mathrm{T}}\mathbf{w}_l\right)\right)}{\mathbf{y}_{\mathrm{pre}}^{\mathrm{T}}\mathbf{y}_{\mathrm{pre}}} \tag{21}$$

was used for model term selection at the lth forward selection stage, and this is aimed at maximizing the reduction of modeling error (term $\mathbf{w}_l^{\mathrm{T}}\mathbf{w}_l\gamma_l^2$) and the log D-optimality (term $\log(\mathbf{w}_l^{\mathrm{T}}\mathbf{w}_l)$) simultaneously, where β is a fixed small positive weighting for the D-optimality cost.

Note that, at some stage, for example, the $\bar{n}_s$-th stage, the remaining unselected model terms will have $[cerr]_l \leq 0$ for $\bar{n}_s + 1 \leq l \leq L$, and this terminates the model construction process. The OFS with the D-optimality algorithm utilizing the MGS scheme is given in Appendix A. Since the D-optimality naturally penalizes overparameterization, the modeling processing can automatically terminates so as to achieve a sparse model. In this paper, we simply set β as a predetermined very small number.

III. Choosing Regularization Parameters for Prefiltering by Optimizing the LOO Misclassification Rate Using PSO

Following Section II-B, it is crucial that the prefiltered signal $y_{\mathrm{pre}}(k)$ is optimized with regard to the regularization parameters in terms of its generalization ability. Consider the general model selection problem from a set of K predictors $y_{\mathrm{pre}}(k)$ due to models generated using different settings of regularization parameters of $\boldsymbol{\lambda} = [\lambda_1, \lambda_2]^{\mathrm{T}}$ indexed by $j = 1, 2, \ldots, K$. The misclassification rate for a given two-class classifier based on (5) can be evaluated based on the misclassified data examples as

$$J(\boldsymbol{\lambda}) = \frac{1}{N}\sum_{k=1}^{N} \mathrm{Id}\left[y(k)y_{\mathrm{pre}}(k)\right] \tag{22}$$

where $\mathrm{Id}(\bullet)$ denotes the misclassification indication function for a data example and is defined as

$$\mathrm{Id}(v) = \begin{cases} 1, & \text{if } v < 0 \\ 0, & \text{if } v \geq 0. \end{cases}$$

CV criteria are metric that measure a model's generalization capability. To optimize the model generalization capability, the model selection criteria are often based on CV [1], [31]. One commonly used version of CV is the so-called LOO CV. By excluding the kth data example in the estimation data set, the output of the model for the kth data example using a model estimated by using remaining $(N-1)$ data examples is denoted as $y_{\mathrm{pre}}^{(-k)}(k)$. The associated predicted class label is calculated by

$$\hat{y}^{(-k)}(k) = \mathrm{sgn}\left(y_{\mathrm{pre}}^{(-k)}(k)\right). \tag{23}$$

It is desirable to derive a classifier with good generalization capability, i.e., to derive a classifier with a minimal misclassification error rate over a new data set that has not been used in model estimation. The LOO CV is often used to estimate generalization error for choosing among different network architectures [1]. The LOO misclassification rate is computed by

$$J^{(-)}(\boldsymbol{\lambda}) = \frac{1}{N}\sum_{k=1}^{N} \mathrm{Id}\left[\mathrm{y(k)y}_{\mathrm{pre}}^{(-k)}\mathrm{(k)}\right] = \frac{1}{\mathrm{N}}\sum_{\mathrm{k=1}}^{\mathrm{N}} \mathrm{Id}\left[\mathrm{d(k)}\right] \tag{24}$$

in which $d(k)$ denotes $y(k)y_{\mathrm{pre}}^{(-k)}(k)$. If $d(k) < 0$, then the kth data sample is misclassified, and the class label produced by the model $y_{\mathrm{pre}}^{(-k)}(k)$ is different from the actual class label $y(k)$.

Direct evaluation of the predicted class labels (23) requires extensive computational effort, so instead, it is shown in the following that the LOO misclassification rate can be evaluated without actually sequentially splitting the estimation data set.

From (9), the NEN parameter estimator based on a specified $\boldsymbol{\lambda}$ using N data points can be represented by

$$\mathbf{g}^{(\mathrm{NEN})} = \frac{1}{1+\lambda_2}\left(\mathbf{U}_s^{\mathrm{T}}\mathbf{y} - \frac{\lambda_1}{2}\mathrm{sign}\left(\mathbf{g}^{(\mathrm{NEN})}\right)\right). \tag{25}$$

The model residual is

$$\begin{aligned} e(k) &= y(k) - \left(\mathbf{g}^{(\mathrm{NEN})}\right)^{\mathrm{T}}\tilde{\mathbf{u}}_s(k) \\ &= y(k) - \frac{1}{1+\lambda_2}\left(\mathbf{y}^{\mathrm{T}}\mathbf{U}_s - \frac{\lambda_1}{2}\left[\mathrm{sign}\left(\mathbf{g}^{(\mathrm{NEN})}\right)\right]^{\mathrm{T}}\right)^{-1} \\ &\quad \times \tilde{\mathbf{u}}_s(k). \end{aligned} \tag{26}$$

If the data sample indexed at k is removed from the estimation data set, the LOO EN parameter estimator obtained by using only $(N-1)$ data points is given by

$$\begin{aligned} \mathbf{g}^{(\mathrm{NEN},-k)} &= \left\{\left[\mathbf{U}_s^{(-k)}\right]^{\mathrm{T}}\mathbf{U}_s^{(-k)} + \lambda_2\mathbf{I}_{n_m}\right\}^{-1} \\ &\quad \times \left(\left[\mathbf{U}_s^{(-k)}\right]^{\mathrm{T}}\mathbf{y}^{(-k)} - \frac{\lambda_1}{2}\mathrm{sign}\left(\mathbf{g}^{(\mathrm{NEN},-k)}\right)\right) \\ &= \left[\mathbf{H}^{(-k)}\right]^{-1}\left(\left[\mathbf{U}_s^{(-k)}\right]^{\mathrm{T}}\mathbf{y}^{(-k)}\right. \\ &\quad \left. - \frac{\lambda_1}{2}\mathrm{sign}\left(\mathbf{g}^{(\mathrm{NEN},-k)}\right)\right) \end{aligned} \tag{27}$$

where $\mathbf{U}_s^{(-k)}$, $\mathbf{y}^{(-k)}$, and $\mathbf{g}^{(\mathrm{NEN},-k)}$ denote the resultant regression matrix, output, and EN parameter estimate vector, respectively. The LOO error evaluated at k is given by

$$\begin{aligned} e^{(-k)}(k) &= y(k) - \left[\mathbf{g}^{(\mathrm{NEN},-k)}\right]^{\mathrm{T}}\tilde{\mathbf{u}}_s(k) \\ &= y(k) - \left(\left[\mathbf{y}^{(-k)}\right]^{\mathrm{T}}\mathbf{U}_s^{(-k)}\right. \\ &\quad \left. - \frac{\lambda_1}{2}\left[\mathrm{sign}\left(\mathbf{g}^{(\mathrm{NEN},-k)}\right)\right]^{\mathrm{T}}\right) \\ &\quad \times \left[\mathbf{H}^{(-k)}\right]^{-1}\tilde{\mathbf{u}}_s(k). \end{aligned} \tag{28}$$

It can be easy to verify that

$$\mathbf{H}^{(-k)} = (1+\lambda_2)\mathbf{I}_{n_m} - \tilde{\mathbf{u}}_s(k)\tilde{\mathbf{u}}_s^{\mathrm{T}}(k) \tag{29}$$

$$\left[\mathbf{y}^{(-k)}\right]^{\mathrm{T}} \mathbf{U}_s^{(-k)} = \mathbf{y}^{\mathrm{T}}\mathbf{U}_s - y(k)\tilde{\mathbf{u}}_s^{\mathrm{T}}(k). \tag{30}$$

Applying the matrix inversion lemma to (29) yields

$$\begin{aligned}\left[\mathbf{H}^{(-k)}\right]^{-1} &= \left[(1+\lambda_2)\mathbf{I}_{n_m} - \tilde{\mathbf{u}}_s(k)\tilde{\mathbf{u}}_s^{\mathrm{T}}(k)\right]^{-1} \\ &= \frac{1}{1+\lambda_2}\left[\mathbf{I}_{n_m} + \frac{\tilde{\mathbf{u}}_s(k)\tilde{\mathbf{u}}_s^{\mathrm{T}}(k)}{1+\lambda_2 - \tilde{\mathbf{u}}_s^{\mathrm{T}}(k)\tilde{\mathbf{u}}_s(k)}\right]\end{aligned} \tag{31}$$

$$\left[\mathbf{H}^{(-k)}\right]^{-1}\tilde{\mathbf{u}}_s(k) = \frac{1}{1+\lambda_2}\frac{\tilde{\mathbf{u}}_s(k)}{1-\tilde{\mathbf{u}}_s^{\mathrm{T}}(k)\tilde{\mathbf{u}}_s(k)/(1+\lambda_2)}. \tag{32}$$

Substituting (30) and (32) into (28) yields

$$\begin{aligned}e^{(-k)}(k) &= y(k) - \left(\mathbf{y}^{\mathrm{T}}\mathbf{U}_s - y(k)\tilde{\mathbf{u}}_s^{\mathrm{T}}(k) - \frac{\lambda_1}{2}\left[\mathrm{sign}(\mathbf{g}^{(\mathrm{NEN},-k)})\right]^{\mathrm{T}}\right) \\ &\quad \times \frac{1}{1+\lambda_2}\frac{\tilde{\mathbf{u}}_s(k)}{1-\tilde{\mathbf{u}}_s^{\mathrm{T}}(k)\tilde{\mathbf{u}}_s(k)/(1+\lambda_2)} \\ &= \frac{y(k) - \frac{1}{1+\lambda_2}\left(\mathbf{y}^{\mathrm{T}}\mathbf{U}_s - \frac{\lambda_1}{2}\left[\mathrm{sign}(\mathbf{g}^{(\mathrm{NEN},-k)})\right]^{\mathrm{T}}\right)\tilde{\mathbf{u}}_s(k)}{1-\tilde{\mathbf{u}}_s^{\mathrm{T}}(k)\tilde{\mathbf{u}}_s(k)/(1+\lambda_2)} \\ &= \frac{e(k)}{1-\tilde{\mathbf{u}}_s^{\mathrm{T}}(k)\tilde{\mathbf{u}}_s(k)/(1+\lambda_2)}\end{aligned} \tag{33}$$

$$\text{if} \quad \mathrm{sign}(\mathbf{g}^{(\mathrm{NEN},-k)}) = \mathrm{sign}\left(\mathbf{g}^{(\mathrm{NEN})}\right). \tag{34}$$

Hence

$$y(k) - y_{\mathrm{pre}}^{(-k)}(k) = \frac{y(k) - y_{\mathrm{pre}}(k)}{1-\tilde{\mathbf{u}}_s^{\mathrm{T}}(k)\tilde{\mathbf{u}}_s(k)/(1+\lambda_2)}. \tag{35}$$

Multiplying both sides of (35) with $y(k)$ and applying $y^2(k) = 1\ \forall k$ yield

$$1 - y(k)y_{\mathrm{pre}}^{(-k)}(k) = \frac{1 - y_{\mathrm{pre}}(k)y(k)}{1-\tilde{\mathbf{u}}_s^{\mathrm{T}}(k)\tilde{\mathbf{u}}_s(k)/(1+\lambda_2)} \tag{36}$$

so that

$$d(k) = y(k)y_{\mathrm{pre}}^{(-k)}(k) = \frac{y_{\mathrm{pre}}(k)y(k) - \tilde{\mathbf{u}}_s^{\mathrm{T}}(k)\tilde{\mathbf{u}}_s(k)/(1+\lambda_2)}{1-\tilde{\mathbf{u}}_s^{\mathrm{T}}(k)\tilde{\mathbf{u}}_s(k)/(1+\lambda_2)}. \tag{37}$$

We introduce a mild assumption that $(\mathrm{sign}(\mathbf{g}^{(\mathrm{NEN},-k)}) = \mathrm{sign}(\mathbf{g}^{(\mathrm{NEN})}))$ holds for most data samples. Using different settings of regularization parameters of $\boldsymbol{\lambda} = [\lambda_1, \lambda_2]^{\mathrm{T}}$ indexed by $j = 1, 2, \ldots, K$, the regularization parameter vector associated with the minimal LOO misclassification rate $J^{(-)}(\boldsymbol{\lambda})$ is chosen, i.e.,

$$\boldsymbol{\lambda}_{\mathrm{opt}} = \arg\left\{\min_{\boldsymbol{\lambda}}\left\{J^{(-)}(\boldsymbol{\lambda}) \approx \frac{1}{N}\sum_{k=1}^{N}\mathrm{Id}\left[d(k)\right] \qquad \forall j\right\}\right\} \tag{38}$$

and the resultant model is selected to produce $y_{\mathrm{pre}}(k)$.

It is simple to evaluate $J^{(-)}(\boldsymbol{\lambda})$ because of the following reasons.

1) First, the proposed EN cost function is based on parameter regularization within an orthogonal space, enabling the analytical formula of EN parameter estimator.
2) Second, we provide the aforementioned original derivation to show that the LOO misclassification rate based on models using EN estimator can be analytically evaluated without actually splitting the data by making use of the matrix inversion lemma.
3) Third, the calculation cost of evaluating $d(k)$ is very small without any matrix inversion involved due to the SVD performed.

Since the cost function $J^{(-)}(\boldsymbol{\lambda})$ is nondifferentiable and multimodal with respect to $\boldsymbol{\lambda}$, the PSO algorithm as shown in Appendix B is applied to solve (38). The algorithm has a two-layer structure, as shown in Fig. 1. The upper level is the PSO with population size of S (Appendix B). It learns the two optimal regularization parameters based on the values of LOO misclassification rate provided by the lower level of S particles. At the lower level, each particle calculates the associated LOO misclassification rate using (24) and (37).

The computational cost of the proposed algorithm comprises that of SVD on the order of $O(N^3)$, PSO on the order of $O(N)$ (scaled by $S \times I_{\max}$), and OFS with D-optimality on the order of $O(N)$ (scaled by L). The main cost is less than twice of that of SVD. This is because $S \times I_{\max}$ and $\bar{n}_s$ are much smaller than N when N is large. L can be set the same as N or smaller than N when N is very large.

IV. Modeling Examples

Numerical experiments were performed to demonstrate the modeling results of the proposed algorithm in comparison to that of several existing classifications algorithms, as published in [32]. Eight most noisy data sets were chosen and experimented: Banana, Breast Cancer, Diabetes, German, Heart, Flare Solar, Titanic, and Waveform, which are available in [33]. For the details of alternative methods used in comparison, the readers are referred to [32].

The results of the first six methods for all examples are quoted from [32] and [33]. Each data set contains 100 realizations of training and test data sets, respectively. Models are constructed over 100 training data sets, and generalization performance is evaluated using the average misclassification rate of the corresponding models over the 100 test data sets. The Gaussian kernel functions $\phi_i(\mathbf{x}) = \exp\{-(\|\mathbf{x} - \mathbf{c}_i\|^2/2\sigma^2)\}$ have been employed in the experiments. A common value σ was predetermined to derive individual models for all 100 realizations for each data set. We used a very small $\beta = 10^{-6}$ for all experiments, which leads to automatic determination of model size. A larger value of β will yield smaller model size, yet the approximation error $E[\varepsilon^2(\mathbf{x}(k))]$ will be larger, resulting unwanted higher discrepancies between between two stages. For each realization of all eight data sets, the full training data sets were used as the RBF centers to form the candidate regressor set. The performance is summarized in Tables I–VIII, respectively. The total running time of training and evaluation

TABLE I
AVERAGE MISCLASSIFICATION RATE IN PERCENT OVER 100 REALIZATIONS OF THE BANANA TEST DATA SET AND MODEL SIZE

	Misclassification rate	Model Size
RBF	10.8 ± 0.6	18
Adaboost with RBF	12.3 ± 0.7	18
AdaBoost$_Reg$	10.9 ± 0.4	18
LP$_Reg$-AdaBoost	$\mathbf{10.7} \pm 0.4$	18
QP$_Reg$-AdaBoost	10.9 ± 0.5	18
SVM with RBF kernel	11.5 ± 0.7	not available
Proposed algorithm	$\mathbf{10.7 \pm 0.5}$	28.7 ± 1.4

TABLE II
AVERAGE MISCLASSIFICATION RATE IN PERCENT OVER 100 REALIZATIONS OF THE BREAST CANCER TEST DATA SET AND MODEL SIZE

	Misclassification rate	Model Size
RBF	27.6 ± 4.7	5
Adaboost with RBF	30.4 ± 4.7	5
AdaBoost$_Reg$	26.5 ± 4.5	5
LP$_Reg$-AdaBoost	26.8 ± 6.1	5
QP$_Reg$-AdaBoost	*25.9* ± 4.6	5
SVM with RBF kernel	26.0 ± 4.7	not available
Proposed algorithm	**25.0** ± 4.2	26.4 ± 2

TABLE III
AVERAGE MISCLASSIFICATION RATE IN PERCENT OVER 100 REALIZATIONS OF THE DIABETES TEST DATA SET AND MODEL SIZE

	Misclassification rate	Model Size
RBF	24.3 ± 1.9	15
Adaboost with RBF	26.5 ± 2.3	15
AdaBoost$_Reg$	23.8 ± 1.8	15
LP$_Reg$-AdaBoost	24.1 ± 1.9	15
QP$_Reg$-AdaBoost	25.4 ± 2.2	15
SVM with RBF kernel	*23.5* ± 1.7	not available
Proposed algorithm	$\mathbf{23.3 \pm 1.7}$	7.7 ± 1.5

TABLE IV
AVERAGE MISCLASSIFICATION RATE IN PERCENT OVER 100 REALIZATIONS OF THE GERMAN TEST DATA SET AND MODEL SIZE

	Misclassification rate	Model Size
RBF	24.7 ± 2.4	8
Adaboost with RBF	27.5 ± 2.5	8
AdaBoost$_Reg$	*24.3* ± 2.1	8
LP$_Reg$-AdaBoost	24.8 ± 2.2	8
QP$_Reg$-AdaBoost	25.3 ± 2.1	8
SVM with RBF kernel	**23.6** ± 2.1	not available
Proposed algorithm	*24.3* ± 2.2	12.8 ± 1.3

TABLE V
AVERAGE MISCLASSIFICATION RATE IN PERCENT OVER 100 REALIZATIONS OF THE HEART TEST DATA SET AND MODEL SIZE

	Misclassification rate	Model Size
RBF	17.6 ± 3.3	4
Adaboost with RBF	20.3 ± 3.4	4
AdaBoost$_Reg$	16.5 ± 3.5	4
LP$_Reg$-AdaBoost	17.5 ± 3.5	4
QP$_Reg$-AdaBoost	17.2 ± 3.4	4
SVM with RBF kernel	*16.0* ± 3.3	not available
Proposed algorithm	**15.9** ± 3.0	8.8 ± 1

for each data set using Matlab on a single computer Intel Core 2 CPU 6400 at 2.13 GHz for our approach was listed in Table IX. This is extremely cheap in comparison with the estimated two years of computing time for all the experiments by the work of Rätsch *et al.* [32] if they were carried out on a single Ultra-SPARC machine. Instead, their work had been

TABLE VI
AVERAGE MISCLASSIFICATION RATE IN PERCENT OVER 100 REALIZATIONS OF THE FLARE SOLAR DATA SET AND MODEL SIZE

	Misclassification rate	Model Size
RBF	34.4 ± 2.0	4
Adaboost with RBF	35.7 ± 1.8	4
AdaBoost$_Reg$	34.2 ± 2.2	4
LP$_Reg$-AdaBoost	34.7 ± 2.0	4
QP$_Reg$-AdaBoost	36.2 ± 1.8	4
SVM with RBF kernel	**32.4** ± 1.8	not available
Proposed algorithm	*33.26* ± 1.7	6.7 ± 0.8

TABLE VII
AVERAGE MISCLASSIFICATION RATE IN PERCENT OVER 100 REALIZATIONS OF THE TITANIC DATA SET AND MODEL SIZE

	Misclassification rate	Model Size
RBF	23.3 ± 1.3	4
Adaboost with RBF	22.6 ± 1.2	4
AdaBoost$_Reg$	22.6 ± 1.2	4
LP$_Reg$-AdaBoost	24.0 ± 4.4	4
QP$_Reg$-AdaBoost	22.7 ± 1.1	4
SVM with RBF kernel	*22.4* ± 1.0	not available
Proposed algorithm	**22.3** ± 1.0	11.1 ± 1.0

TABLE VIII
AVERAGE MISCLASSIFICATION RATE IN PERCENT OVER 100 REALIZATIONS OF THE WAVEFORM DATA SET AND MODEL SIZE

	Misclassification rate	Model Size
RBF	10.7 ± 1.1	10
Adaboost with RBF	10.8 ± 0.6	10
AdaBoost$_Reg$	**9.8** ± 0.8	10
LP$_Reg$-AdaBoost	10.5 ± 1.0	10
QP$_Reg$-AdaBoost	10.1 ± 0.5	10
SVM with RBF kernel	9.9 ± 0.4	not available
Proposed algorithm	**9.8** ± 0.4	34.1 ± 1.9

TABLE IX
TOTAL RECORDED RUNNING TIME USED IN TRAINING AND EVALUATION

Data sets	Banana	Breast Cancer	Diabetes	German
Running time (sec)	3154.37	672.97	1252.57	4011.8
Data sets	Heart	Flare Solar	Titanic	Waveform
Running time (sec)	286.84	1951.18	324.65	3644.55

carried out by 32 computers [32]. The results have shown that the proposed approach can construct parsimonious classifiers with competitive classification accuracy for these data sets with little computational cost. Although our model sizes are not generally smaller than other methods, we point out that, in the work [32], the model size for each data set was preset using CV based on their RBF-based model (the first method), except SVM for which no model size is reported. In other words, all the methods except SVM cannot perform model structure selection automatically by the algorithms. In addition, from our experience, SVMs are generally not very sparse and are expected to have larger model sizes than ours.

We also point out that the proposed algorithm is very robust in terms of that a common value σ was used for all 100 realizations for each data set, and the performances are good and insensitive to σ within a wide range. Hence, for practitioners who are looking for building classifiers with robust and superior classification performance from noisy data sets, without too much computational costs and tuning efforts, the proposed method will be a very good choice.

V. Conclusion

This paper has proposed an efficient two-stage construction algorithm for linear-in-the-parameter classifiers aimed at noisy two-class classification data sets. The first stage constructs a prefiltered signal that is then used as the desired output for the second stage, the construction of a sparse linear-in-the-parameter classifier. The prefiltering stage is a two-level algorithm aimed at maximizing a model's generalization capability. Using SVD, a new EN model identification algorithm is employed at the lower level, and the two regularization parameters are found by minimizing the LOO misclassification rate, using a PSO algorithm at the upper level. The original contributions are first to define an EN cost function based on parameters in latent space via SVD, which facilitates the automatic model structure selection process with no need of using a predetermined error tolerance to terminate the forward selection process. Second, we derived the LOO misclassification formula based on the prefiltered signal and show that its computational cost is small. As a result, a fully automated procedure is achieved without resort to any other validation data set for iterative model evaluation. The second stage of sparse classifier construction is based on OFR with D-optimality algorithm. Eight benchmark examples are included to demonstrate the competitiveness of the new approaches.

Appendix A
The OFR With D-Optimality Using the MGS Orthogonalization Procedure

The MGS orthogonalization procedure calculates the $\mathbf{A}$ matrix row by row and orthogonalizes $\mathbf{\Phi}$ as follows: At the lth stage, make the columns $\boldsymbol{\phi}_j$, $l+1 \le j \le L$, orthogonal to the lth column, and repeat the operation for $1 \le l \le L-1$. Specifically, denoting $\boldsymbol{\phi}_j^{(0)} = \boldsymbol{\phi}_j$, $1 \le j \le L$, then

$$\left.\begin{aligned} \mathbf{w}_l &= \boldsymbol{\phi}_l^{(l-1)} \\ a_{l,j} &= \frac{\mathbf{w}_l^{\mathrm{T}} \boldsymbol{\phi}_j^{(l-1)}}{\left(\mathbf{w}_l^{\mathrm{T}} \mathbf{w}_l\right)}, \qquad l+1 \le j \le L \\ \boldsymbol{\phi}_j^{(l)} &= \boldsymbol{\phi}_j^{(l-1)} - a_{l,j}\mathbf{w}_l, \qquad l+1 \le j \le L \end{aligned}\right\} \quad l = 1, 2, \ldots, L-1. \tag{39}$$

The last stage of the procedure is simply $\mathbf{w}_L = \boldsymbol{\phi}_L^{(L-1)}$. The elements of $\mathbf{\Gamma}$ are computed by transforming $\mathbf{y}_{\mathrm{pre}}^{(0)} = \mathbf{y}_{\mathrm{pre}}$ in a similar way

$$\left.\begin{aligned} \gamma_l &= \frac{\mathbf{w}_l^{\mathrm{T}} \mathbf{y}^{(l-1)}}{\left(\mathbf{w}_l^{\mathrm{T}} \mathbf{w}_l\right)} \\ \mathbf{y}_{\mathrm{pre}}^{(l)} &= \mathbf{y}_{\mathrm{pre}}^{(l-1)} - \gamma_l \mathbf{w}_l \end{aligned}\right\} 1 \le l \le L. \tag{40}$$

This orthogonalization scheme can be used to derive a simple and efficient algorithm for selecting subset models in a forward-regression manner. First, define

$$\mathbf{\Phi}^{(l-1)} = \left[\mathbf{w}_1 \cdots \mathbf{w}_{l-1} \boldsymbol{\phi}_l^{(l-1)} \cdots \boldsymbol{\phi}_L^{(l-1)}\right]. \tag{41}$$

If some of the columns $\boldsymbol{\phi}_l^{(l-1)}, \ldots, \boldsymbol{\phi}_L^{(l-1)}$ in $\mathbf{\Phi}^{(l-1)}$ have been interchanged, this will still be referred to as $\mathbf{\Phi}^{(l-1)}$ for notational convenience. The lth stage of the selection procedure is given as follows.

Step 1) For $l \le j \le L$, compute

$$\left.\begin{aligned} \gamma_l^{(j)} &= \frac{\left(\boldsymbol{\phi}_j^{(l-1)}\right)^{\mathrm{T}} \mathbf{y}^{(l-1)}}{\left(\left(\boldsymbol{\phi}_j^{(l-1)}\right)^{\mathrm{T}} \boldsymbol{\phi}_j^{(l-1)}\right)} \\ [crerr]_l^{(j)} &= \left(\left(\gamma_l^{(j)}\right)^2 \left(\left(\boldsymbol{\phi}_j^{(l-1)}\right)^{\mathrm{T}} \boldsymbol{\phi}_j^{(l-1)}\right) \right. \\ &\quad \left. + \beta \log\left(\left(\boldsymbol{\phi}_j^{(l-1)}\right)^{\mathrm{T}} \boldsymbol{\phi}_j^{(l-1)}\right)\right) / \left(\mathbf{y}_{\mathrm{pre}}^{\mathrm{T}} \mathbf{y}_{\mathrm{pre}}\right). \end{aligned}\right\}$$

Step 2) Find

$$[crerr]_l = [crerr]_l^{(j_l)} = \max\left\{[crerr]_l^{(j)}, \qquad l \le j \le L\right\}.$$

Then, the j_lth column of $\mathbf{\Phi}^{(l-1)}$ is interchanged with the lth column of $\mathbf{\Phi}^{(l-1)}$, and the j_lth column of $\mathbf{A}$ is interchanged with the lth column of $\mathbf{A}$ up to the $(l-1)$th row. This effectively selects the j_lth candidate as the lth regressor in the subset model.

Step 3) Perform the orthogonalization as indicated in (39) to derive the lth row of $\mathbf{A}$ and to transform $\mathbf{\Phi}^{(l-1)}$ into $\mathbf{\Phi}^{(l)}$. Calculate γ_l, and update $\mathbf{y}_{\mathrm{pre}}^{(l-1)}$ into $\mathbf{y}_{\mathrm{pre}}^{(l)}$ in the way shown in (40).

The selection is terminated at the $\bar{n}_s$ stage, as discussed in Section II-B and, this produces a subset model containing $\bar{n}_s$ significant regressors. The algorithm described here is in its standard form; a fast implementation can however be adopted to reduce computational cost [34].

Appendix B
PSO for Choosing Regularization Parameters

In the following, we propose to apply the PSO algorithm [16], [17] and aim to solve

$$\boldsymbol{\lambda}_{\mathrm{opt}} = \arg \min_{\boldsymbol{\lambda} \in \prod_{j=1}^{2} \Lambda_j} J^{(-)}(\boldsymbol{\lambda}) \tag{42}$$

where

$$\prod_{j=1}^{2} \Lambda_j = \prod_{j=1}^{2} [0, \Lambda_{j,\max}] \tag{43}$$

defines the search space. A swarm of particles $\{\boldsymbol{\lambda}_i^{(m)}\}_{i=1}^{S}$ that represent potential solutions are "flying" in the search space $\prod_{j=1}^{2} \Lambda_j$, where S is the swarm size and index m denotes the iteration step. The algorithm is summarized as follows.

1) *Swarm initialization*. Set the iteration index $m = 0$, and randomly generate $\{\boldsymbol{\lambda}_i^{(m)}\}_{i=1}^{S}$ in the search space $\prod_{j=1}^{2} \Lambda_j$.
2) *Swarm evaluation*. The cost of each particle $\boldsymbol{\lambda}_i^{(m)}$ is obtained as $J^{(-)}(\boldsymbol{\lambda}_i^{(m)})$. Each particle $\boldsymbol{\lambda}_i^{(m)}$ remembers its best position visited so far, denoted as $\mathbf{pb}_i^{(m)}$, which provides the cognitive information. Every particle also knows the best position visited so far among the entire swarm, denoted as $\mathbf{gb}^{(m)}$, which provides the social information.

The cognitive information $\{\mathbf{pb}_i^{(m)}\}_{i=1}^{S}$ and the social information $\mathbf{gb}^{(m)}$ are updated at each iteration:

For $(i = 1; i \leq S; i++)$
  If $(J^{(-)}(\boldsymbol{\lambda}_i^{(m)}) < J^{(-)}(\mathbf{pb}_i^{(m)}))\ \mathbf{pb}_i^{(m)} = \boldsymbol{\lambda}_i^{(m)}$;
End for;
$i^* = \arg\min_{1 \leq i \leq S} J^{(-)}(\mathbf{pb}_i^{(m)})$;
  If $(J^{(-)}(\mathbf{pb}_{i^*}^{(m)}) < J^{(-)}(\mathbf{gb}^{(m)}))\mathbf{gb}^{(m)} = \mathbf{pb}_{i^*}^{(m)}$;

3) *Swarm update*. Each particle $\boldsymbol{\lambda}_i^{(m)}$ has a velocity, denoted as $\boldsymbol{\gamma}_i^{(m)}$, to direct its "flying." The velocity and position of the ith particle are updated in each iteration according to

$$\boldsymbol{\gamma}_i^{(m+1)} = \mu_0 * \boldsymbol{\gamma}_i^{(m)} + \text{rand}() * \mu_1 * \left(\mathbf{pb}_i^{(m)} - \boldsymbol{\lambda}_i^{(m)}\right) + \text{rand}() * \mu_2 * \left(\mathbf{gb}^{(m)} - \boldsymbol{\lambda}_i^{(m)}\right) \quad (44)$$

$$\boldsymbol{\lambda}_i^{(m+1)} = \boldsymbol{\lambda}_i^{(m)} + \boldsymbol{\gamma}_i^{(m+1)} \quad (45)$$

where μ_0 is the inertia weight and μ_1 and μ_2 are the two acceleration coefficients. $\text{rand}()$ denotes the uniform random number between zero and one. In order to avoid excessive roaming of particles beyond the search space [21], a velocity space

$$\prod_{j=1}^{2} \Upsilon_j = \prod_{j=1}^{2} [-\Upsilon_{j,\max}, \Upsilon_{j,\max}] \quad (46)$$

is imposed on $\boldsymbol{\gamma}_i^{(m+1)}$ so that

$$\text{If } \left(\boldsymbol{\gamma}_i^{(m+1)}|_j > \Upsilon_{j,\max}\right), \quad \boldsymbol{\gamma}_i^{(m+1)}|_j = \Upsilon_{j,\max}$$
$$\text{If } \left(\boldsymbol{\gamma}_i^{(m+1)}|_j < -\Upsilon_{j,\max}\right), \quad \boldsymbol{\gamma}_i^{(m+1)}|_j = -\Upsilon_{j,\max}$$

where $\boldsymbol{\gamma}|_j$ denotes the jth element of $\boldsymbol{\gamma}$. Moreover, if the velocity as given in (44) approaches zero, it is reinitialized proportional to $\Upsilon_{j,\max}$ with a small factor ν

$$\text{If } \left(\boldsymbol{\gamma}_i^{(m+1)}|_j == 0\right), \quad \boldsymbol{\gamma}_i^{(m+1)}|_j = \pm\text{rand}() * \nu * \Upsilon_{j,\max}. \quad (47)$$

4) *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 $m = m + 1$, and go to Step 2).

Ratnaweera and coauthors [19] reported that using a time-varying acceleration coefficient (TVAC) enhances the performance of PSO. We adopt this mechanism, in which μ_1 is reduced from 2.5 to 0.5 and μ_2 varies from 0.5 to 2.5 during the iterative procedure

$$\mu_1 = (0.5 - 2.5) * m/I_{\max} + 2.5$$
$$\mu_2 = (2.5 - 0.5) * m/I_{\max} + 0.5. \quad (48)$$

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 better exploit the search space, avoiding 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 use $\mu_0 = \text{rand}()$ at each iteration.

The search space as given in (43) is defined by the specific problem to be solved, and the velocity limit $\Upsilon_{j,\max}$ is empirically set. An appropriate value of the small control factor ν in (47) for avoiding zero velocity is empirically found to be $\nu = 0.1$ for our application.

ACKNOWLEDGMENT

The authors would like to thank the reviewers for their constructive comments.

REFERENCES

[1] M. Stone, "Cross validatory choice and assessment of statistical predictions," *J. Roy. Statist. Soc. Ser. B*, vol. 36, no. 2, pp. 117–147, 1974.
[2] 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.
[3] M. J. L. Orr, "Regularization in the selection of radial basis function centers," *Neural Comput.*, vol. 7, no. 3, pp. 606–623, May 1995.
[4] X. Hong and S. A. Billings, "Parameter estimation based on stacked regression and evolutionary algorithms," *Proc. Inst. Elect. Eng.—Control Theory Appl.*, vol. 146, no. 5, pp. 406–414, Sep. 1999.
[5] L. Ljung and T. Glad, *Modelling of Dynamic Systems*. Englewood Cliffs, NJ: Prentice-Hall, 1994.
[6] H. Akaike, "A new look at the statistical model identification," *IEEE Trans. Autom. Control*, vol. AC-19, no. 6, pp. 716–723, Dec. 1974.
[7] V. Vapnik, *The Nature of Statistical Learning Theory*. New York: Springer-Verlag, 1995.
[8] M. E. Tipping, "Sparse Bayesian learning and the relevance vector machine," *J. Mach. Learn. Res.*, vol. 1, pp. 211–244, Sep. 2001.
[9] B. Scholkopf and A. J. Smola, *Learning With Kernels: Support Vector Machine, Regularization, Optimization and Beyond*. Cambridge, MA: MIT Press, 2002.
[10] X. Hong and C. J. Harris, "Nonlinear model structure design and construction using orthogonal least squares and D-optimality design," *IEEE Trans. Neural Netw.*, vol. 13, no. 5, pp. 1245–1250, Sep. 2002.
[11] S. Chen, S. A. Billings, and W. Luo, "Orthogonal least squares methods and their applications to non-linear system identification," *Int. J. Control*, vol. 50, no. 5, pp. 1873–1896, 1989.
[12] 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, Sep. 2002.
[13] S. Chen, X. X. Wang, X. Hong, and C. J. Harris, "Kernel classifier construction using orthogonal forward selection and boosting with Fisher ratio class separability," *IEEE Trans. Neural Netw.*, vol. 17, no. 6, pp. 1652–1656, Nov. 2004.
[14] X. Hong, S. Chen, and C. J. Harris, "A fast kernel 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.
[15] S. Chen, X. Hong, and C. J. Harris, "Particle swarm optimization aided orthogonal forward regression for unified data modelling," *IEEE Trans. Evol. Comput.*, vol. 14, no. 4, pp. 477–499, Aug. 2010.
[16] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proc. IEEE Int. Conf. Neural Netw.*, Perth, Australia, Nov. 27–Dec. 1, 1995, vol. 4, pp. 1942–1948.
[17] J. Kennedy and R. C. Eberhart, *Swarm Intelligence*. Waltham, MA: Morgan Kaufmann, 2001.
[18] D. W. van der Merwe and A. P. Engelbrecht, "Data clustering using particle swarm optimization," in *Proc. CEC*, Cabberra, Australia, Dec. 8–12, 2003, pp. 215–220.
[19] A. Ratnaweera, S. K. Halgamuge, and H. C. Watson, "Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients," *IEEE Trans. Evol. Comput.*, vol. 8, no. 3, pp. 240–255, Jun. 2004.
[20] M. G. H. Omran, "Particle swarm optimization methods for pattern recognition and image processing," Ph.D. dissertation, Univ. Pretoria, Pretoria, South Africa, 2005.
[21] S. M. Guru, S. K. Halgamuge, and S. Fernando, "Particle swarm optimisers for cluster formation in wireless sensor networks," in *Proc. Int. Conf. Intell. Sens., Sens. Netw. Inf. Process.*, Melbourne, Australia, Dec. 5–8, 2005, pp. 319–324.

[22] 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. Veh. Technol.*, vol. 56, no. 5, pp. 3006–3013, Sep. 2007.
[23] 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.
[24] H. Zou and T. Hastie, "Regularization and variable selection via the elastic net," *J. Roy. Stastist. Soc. B*, vol. 67, no. 2, pp. 301–320, 2005.
[25] S. Chen, "Locally regularised orthogonal least squares algorithm for the construction of sparse kernel regression models," in *Proc. 6th Int. Cof. Signal Process.*, Beijing, China, 2002, pp. 1229–1232.
[26] D. J. C. MacKay, "Bayesian Methods for Adaptive Models," Ph.D. thesis, California Inst. Technol., Pasadena, CA, 1991.
[27] S. S. Chen, D. L. Donoho, and M. A. Saunders, "Atomic decomposition by basis pursuit," *SIAM J. Sci. Comput.*, vol. 43, no. 1, pp. 129–159, 1998.
[28] R. Tibshirani, "Regression shrinkage and selection via the lasso," *J. Roy. Statist. Soc. Ser. B*, vol. 58, no. 1, pp. 267–288, 1996.
[29] B. Efron, I. Johnstone, T. Hastie, and R. Tibshirani, "Least angle regression," *Ann. Statist.*, vol. 32, pp. 407–451, 2004.
[30] S. Chen, "Local regularization assisted orthogonal least squares regression," *Neurocomputing*, vol. 69, no. 4–6, pp. 559–585, Jan. 2006.
[31] L. Ljung, *System Identification: Theory for the User*. Upper Saddle River, NJ: Prentice-Hall, 1987.
[32] G. Rätsch, T. Onoda, and K. R. Müller, "Soft margins for AdaBoost," *Mach. Learn.*, vol. 42, no. 3, pp. 287–320, Mar. 2001.
[33] G. Rätsch. [Online]. Available: http://www.fml.tuebingen.mpg.de/members/raetsch/benchmark
[34] S. Chen and J. Wigger, "Fast orthogonal least squares algorithm for efficient subset selection," *IEEE Trans. Signal Process.*, vol. 43, no. 7, pp. 1713–1715, Jul. 1995.

Xia Hong received the B.Sc. and M.Sc. degrees in automatic control from the National University of Defense 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.

In 1987–1993, she was a Research Assistant with Beijing Institute of Systems Engineering, Beijing, China. In 1997–2001, she was a Research Fellow with the Department of Electronics and Computer Science, University of Southampton, Southampton, U.K. She is currently a Reader with the School of Systems Engineering, University of Reading, Reading, U.K. She is actively engaged in research in nonlinear system 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 a recipient of the Donald Julius Groen Prize from the Institution of Mechanical Engineers in 1999.

Sheng Chen (M'90–SM'97–F'08) received the B.Eng. degree in control engineering from the East China Petroleum Institute, Dongying, China, in 1982 and the Ph.D. degree in control engineering from the City University London, London,U.K., in 1986.

Since September 1999, he has been with the University of Southampton, Southampton, U.K. He is also with the Faculty of Engineering, King Abdulaziz University, Jeddah, Saudi Arabia. He previously held research and academic appointments at the University of Sheffield, Sheffield, U.K., The University of Edinburgh, Edinburgh, U.K., and University of Portsmouth, Portsmouth, U.K. He has published over 400 research papers. His recent research works include adaptive nonlinear signal processing, modeling and identification of nonlinear systems, neural network research, finite-precision digital controller design, evolutionary computation methods, and optimization.

Chris J. Harris received the B.Sc. degree from the University of Leicester, Leicester, U.K., the M.A. degree from the University of Oxford, Oxford, U.K., and the Ph.D. degree from the University of Southampton, Southampton, U.K.

He is currently with the School of Electronics and Computer Science, University of Southampton. He previously held appointments at the University of Hull, Hull, U.K., University of Manchester Institute of Science and Technology, Manchester, U.K., University of Oxford, and Cranfield University, Cranfield, U.K., as well as was employed by the U.K. Ministry of Defence. He has authored or coauthored 12 books and over 400 research papers, and he was an Associate Editor of numerous international journals, including *Automatica, Engineering Applications of Artificial Intelligence*, *International Journal of General Systems Engineering*, *International Journal of Systems Science*, and *International Journal of Mathematical Control and Information*. His research interests are in the areas of intelligent and adaptive system theory and its application to intelligent autonomous systems, management infrastructures, intelligent control and estimation of dynamic processes, multisensor data fusion, and system integration.

Dr. Harris was elected to the Royal Academy of Engineering in 1996 and was a recipient of the Institution of Electrical Engineers (IEE) Senior Achievement Medal in 1998 for his work on autonomous systems and the highest international award in IEE, the IEE Faraday Medal in 2001 for his work on intelligent control and neurofuzzy system.