Sparse support vector regressors based on forward basis selection

Muraoka, Shigenori

Abe, Shigeo

(Citation)
Neural Networks, 2009. IJCNN 2009. International Joint Conference on:2183-2187

(Issue Date)
2009-06

(Resource Type)
conference paper

(Version)
Version of Record

(URL)
https://hdl.handle.net/20.500.14094/90000929

Sparse Support Vector Regressors Based on Forward Basis Selection

Shigenori Muraoka and Shigeo Abe

***Abstract*— Support Vector Regressors (SVRs) usually give sparse solutions but as a regression problem becomes more difficult the number of support vectors increases and thus sparsity is lost. To solve this problem, in this paper we propose sparse support vector regressors (S-SVRs) trained in the reduced empirical feature space. First by forward selection we select the training data samples, which minimize the regression error estimated by kernel least squares. Then in the reduced empirical feature space spanned by the selected, mapped training data, we train the SVR in the dual form. Since the mapped support vectors obtained by training the S-SVR are expressed by the linear combination of the selected, mapped training data, the support vectors, in the sense that form a solution, are selected training data. By computer simulation, we compare performance of the proposed method with that of the regular SVR and that of the sparse SVR based on Cholesky factorization.**

I. Introduction

In a function approximation problem, the unknown target function is estimated using input-output pairs. Support Vector Regressors (SVRs) [1] are one of the successful methods for function approximation that realize high generalization ability. The main features of SVRs are mapping of the input space into the high dimensional feature space and the use of kernel tricks to avoid explicit treatment of variables in the feature space. The solution is expressed by a small number of training data called support vectors.

However the major drawback of the SVRs is that training becomes difficult for large size problems, because in training an SVR the number of variables is twice the number of training data. In addition, as a function approximation problem becomes difficult, the number of support vectors increases. This leads to a slower function evaluation for a given input.

To solve this problem many sparse techniques have been developed. In [2], [3], least squares (LS) SVRs are trained in the reduced empirical feature space, which is obtained by the Cholesky factorization. In [4], fixed-size LS-SVMs (FS-SVMs) are proposed, in which fixed number of support vectors are selected from the training data. FS-SVMs are extended to regression problems in [5], [6]. In [7], sparse SVR based on orthogonal forward selection is proposed. In [8] basis vectors are selected by forward selection for LS-SVMs. And to speed up training for large sized problems, basis vectors are selected from a random subset of training data.

Shigenori Muraoka and Shigeo Abe are with Graduate School of Engineering, Kobe University, Kobe, Japan (email: 087t260t@stu.kobe-u.ac.jp, abe@kobe-u.ac.jp).

In this paper, we proposed Sparse Support Vector Regressors (S-SVRs) by forward selection of basis vectors. Starting from the empty set, we select the training data one at a time that minimizes the approximation error estimated by kernel least squares (KLS). We set a threshold value for terminating the selection procedure determined by the KLS error. Then in the reduced empirical feature space spanned by selected, mapped training data, we train the SVR in the dual form. Since the mapped support vectors for the trained SVR are expressed by the linear combination of the selected, mapped training data, the selected training data, not the support vectors of the obtained SVR, become support vectors, in the sense that constitute the solution. Thus, if the number of selected training data is smaller than the support vectors obtained by training the SVR, we obtain the sparse SVR. We call this S-SVR.

We compare performance of the proposed S-SVR with that of the sparse SVR whose basis vectors are selected by the Cholesky factorization and the regular SVR.

The rest of the paper is organized as follows. In Section II we discuss architectures of sparse SVR and in Section III we discuss two selection methods of basis vectors: namely selection by the Cholesky factorization and forward selection. In Section IV we compare the simulation results using some benchmark datasets. Finally in Section V we conclude our work.

II. Architecture of Sparse Support Vector Regressors

In this section we discuss the architecture of S-SVRs trained in the reduced empirical feature space. The difference between regular SVRs and S-SVRs is that whether we map the input space into the feature space or the reduced empirical feature space.

Let the M input-output pairs be $(\mathbf{x}_i, y_i)$ $(i = 1, ..., M)$ and the mapping function that maps the input vector $\mathbf{x}$ into the high dimensional feature space be $\mathbf{g}(\mathbf{x})$. According to [9], [2], [3] we can define the mapping function to the empirical feature space, which is equivalent to the feature space. But, by this method, we need to calculate the eigenvalues and eigenvectors of the kernel matrix. Thus, we use the following mapping function:

$$\mathbf{h}(\mathbf{x}) = (H(\mathbf{x}_{i1}, \mathbf{x}), ..., H(\mathbf{x}_{iN}, \mathbf{x}))^T. \quad (1)$$

where $N\,(\leq M)$ is the dimension of the empirical feature space and is equivalent to the rank of the kernel matrix, $H(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{g}^T(\mathbf{x}_i)\,\mathbf{g}(\mathbf{x}_j)$ is the kernel for the feature space,

978-1-4244-3553-1/09/$25.00 ©2009 IEEE

and $\mathbf{x}_{i1}, \ldots, \mathbf{x}_{iN}$ are linearly independent training data that span the empirical feature space.

We can reduce the dimension N so long as generalization ability of the regressor does not deteriorate. We call the subspace whose dimension is smaller than the empirical feature space reduced empirical feature space. But if there is no confusion, we call the reduced empirical feature space simply empirical feature space.

We formulate the SVR in the empirical feature space. The approximation function $f(\mathbf{x})$ in the empirical feature space is given by

$$f(\mathbf{x}) = \mathbf{w}^T \mathbf{h}(\mathbf{x}) + b, \tag{2}$$

where $\mathbf{w}$ is the weight vector and b is the bias term.

The loss function is given by

$$L(y, f(\mathbf{x})) = \begin{cases} 0 & \text{if } |y - f(\mathbf{x})| \leq \varepsilon, \\ |y - f(\mathbf{x})| - \varepsilon & \text{otherwise,} \end{cases} \tag{3}$$

where ε is a user-defined error threshold. Now the SVR defined in the empirical feature space is given by

$$\text{minimize} \quad \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{M}(\xi_i + \xi_i^*) \tag{4}$$

$$\text{subject to} \quad \begin{cases} y_i - \mathbf{w}^T\mathbf{h}(\mathbf{x}_i) - b \leq \varepsilon + \xi_i, \\ \mathbf{w}^T\mathbf{h}(\mathbf{x}_i) + b - y_i \leq \varepsilon + \xi_i^*, \\ \xi_i \geq 0, \quad \xi_i^* \geq 0 \\ \qquad\qquad \text{for} \quad i = 1, ..., M, \end{cases} \tag{5}$$

where C is the margin parameter and ξ_i and ξ_i^* are the slack variables for $\mathbf{x}_i$.

We solve the above optimization problem in the dual form as follows:

$$\text{maximize} \quad -\frac{1}{2}\sum_{i,j=1}^{M}(\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*)H_{\mathrm{e}}(\mathbf{x}_i, \mathbf{x}_j) - \varepsilon\sum_{i=1}^{M}(\alpha_i + \alpha_i^*) + \sum_{i=1}^{N} y_i(\alpha_i - \alpha_i^*) \tag{6}$$

$$\text{subject to} \quad \begin{cases} \sum_{i=1}^{M}(\alpha_i - \alpha_i^*) = 0, \\ 0 \leq \alpha_i, \alpha_i^* \leq C, \end{cases} \tag{7}$$

where $H_{\mathrm{e}}(\mathbf{x}, \mathbf{x}') = \mathbf{h}^T(\mathbf{x})\ \mathbf{h}(\mathbf{x}')$ is the kernel for the empirical feature space, α_i and α_i^* are the Lagrange multipliers associated with $\mathbf{x}_i$. The resulting approximation function becomes

$$f(\mathbf{x}) = \sum_{i=1}^{M}(\alpha_i - \alpha_i^*)\, H_{\mathrm{e}}(\mathbf{x}_i, \mathbf{x}) + b. \tag{8}$$

We call the above SVR S-SVR. The difference between the regular SVR and S-SVR is that whether we use $H(\mathbf{x}, \mathbf{x}')$ or $H_{\mathrm{e}}(\mathbf{x}, \mathbf{x}')$. In the S-SVR, the mapped support vectors obtained by solving (6) and (7) are expressed by the linear combination of $\mathbf{x}_{i1}, \ldots, \mathbf{x}_{iN}$. Therefore, in the S-SVR the increase of support vectors does not increase complexity of the obtained regressor but the increase of N does. Thus, in the S-SVR, we consider $\mathbf{x}_{i1}, \ldots, H(\mathbf{x}_{iN})$ as support vectors and by reducing the value of N without deteriorating the generalization ability, we can obtain sparse solutions.

III. Selecting Basis Vectors

In training the S-SVR, first we need to select basis vectors. Here, we discuss two methods in selecting basis vectors. The first method selects basis vectors by decomposing the kernel matrix by the Cholesky factorization. And the second method selects basis vectors by forward selection based on the approximation error evaluated by KLS.

A. Sparse Support Vector Regressors by Cholesky Factorization

The dimension of the empirical feature space, N, is equivalent to the rank of the kernel matrix. Thus, by the Cholesky factorization of the kernel matrix, in which we delete the column and row vectors of the kernel matrix whose diagonal element is zero, we obtain the basis vectors that span the empirical feature space. By introducing a small positive threshold and deleting a column vector and the associated row vector of the kernel matrix if the diagonal element is smaller than the threshold value, we can control the number of basis vectors. We call the S-SVR whose basis vectors are selected by the Cholesky factorization, S-SVR (c). The procedure of Cholesky factorization is given below.

If $M \times M$ matrix H is a symmetric, positive definite matrix, H can be decomposed into the product of two triangular matrices:

$$H = L\, L^T \tag{9}$$

where L is a lower triangular matrix and each element is given by

$$L_{op} = \frac{H_{op} - \sum_{n=1}^{p-1} L_{pn} L_{on}}{L_{pp}} \quad \text{for} \quad o = 1, ..., M, p = 1, ..., o-1 \tag{10}$$

$$L_{aa} = \sqrt{H_{aa} - \sum_{n=1}^{a-1} L_{an}^2} \quad \text{for} \quad a = 1, ..., M. \tag{11}$$

To select linearly independent vectors, whose number is smaller than the rank of H, if

$$H_{aa} - \sum_{n=1}^{a-1} L_{an}^2 < \eta_{\mathrm{c}} \tag{12}$$

is satisfied, where η_{c} is a small positive value, we delete the associated column and row vectors and proceed the Cholesky factorization. When the Cholesky factorization is finished, the training data that are not deleted in factorization become basis vectors. Excluding the floating operations for the deleted data, the order of floating operations for the Cholesky factorization is $O(N\, M^2)$.

In training the S-SVR (c), we need to determine the kernel and values of the kernel parameter, C, ε, and η_c. Here, using the regular SVR, we determine the kernel and kernel parameter by cross-validation. Then we train S-SVR (c). In the following we show the algorithm to train the S-SVR (c).

Step 1
: Setting the kernel and kernel parameter value determined by training the regular SVR, determine the values of C, ε, and η_c by fivefold cross-validation.

Step 2
: Using all the training data, generate the kernel matrix for the determined kernel and kernel parameter incrementally and decompose the kernel matrix by the Cholesky factorization. If (12) is not satisfied, overwrite the current column and row vectors with those associated with the new training data and iterate factorization until all the training data are processed.

Step 3
: Construct the mapping function from the chosen columns of the kernel matrix, and train the S-SVR for the determined value of C and ε.

In Step 2, incrementally factorizing the kernel matrix, unnecessary kernel calculations that are associated with the deleted training data are avoided and the size of the kernel matrix can be smaller than M. This is especially effective when many training data are deleted.

B. Sparse Support Vector Regressors by Kernel Least Squares

The linearly independent training data that span the empirical feature space are not unique. Namely, there are many combinations of training data that span the empirical feature space. Non-uniqueness will not be a problem since they are different sets of basis vectors that span the empirical feature space. But if we reduce the dimension of the empirical feature space, the different sets of basis vectors span different reduced empirical feature spaces. Thus, the generalization ability will differ according to the selected basis vectors. To avoid this problem, we need to select basis vectors according to the selection criterion related to the generalization ability. But it will be time consuming to train the SVR in selecting basis vectors. To accelerate basis selection, we consider using kernel least squares (KLS). Namely, starting from the empty set, we select a training data sample that realizes the minimum approximation error evaluated by KLS by forward selection. We repeat forward selection until the approximation error reaches some threshold value.

1) Kernel Least Square Methods: In this section we summarize KLS. In KLS we approximate the output y by $\hat{y}$ using the input-output pairs $\{\mathbf{x}_i, y_i\}\,(i = 1, \ldots, M)$:

$$\hat{y} = \mathbf{a}^T \mathbf{g}(\mathbf{x}) + b, \tag{13}$$

where $\mathbf{a}$ is a coefficient vector, b is a bias term. To simplify notations, we assume that the last element of $\mathbf{g}(\mathbf{x})$ is 1 and the last element of $\mathbf{a}$ is b. Then (13) is simplified as follows:

$$\hat{y} = \mathbf{a}^T \mathbf{g}(\mathbf{x}). \tag{14}$$

We determine $\mathbf{a}$ so that the sum of the squared errors is minimized:

$$\text{minimize} \quad J = \sum_{i=1}^{M} (y_i - \hat{y}_i)^2 \tag{15}$$

Assume that $\mathbf{a}$ is expressed by the linear combination of k independent vectors $\{\mathbf{g}(\mathbf{x}_i) \,|\, i \in S^k\}$ where S^k is the index set with k elements selected from $\{1, \ldots, M\}$. Then

$$\mathbf{a} = \sum_{i \in S^k} \beta_i \, \mathbf{g}(\mathbf{x}_i), \tag{16}$$

where β_i are parameters. Substituting (16) into (14), we obtain

$$\hat{y} = \sum_{i \in S^k} \beta_i \, \mathbf{g}^T(\mathbf{x}_i) \, \mathbf{g}(\mathbf{x}) = \sum_{i \in S^k} \beta_i \, H(\mathbf{x}_i, \mathbf{x}). \tag{17}$$

Substituting (16) into (15) we obtain

$$\begin{aligned} J_k &= \frac{1}{2} \sum_{i=1}^{M} \left(y_i - \sum_{j \in S^k} \beta_j H(\mathbf{x}_j, \mathbf{x}_i) \right)^2 \\ &= \frac{1}{2} (\mathbf{y} - H\boldsymbol{\beta})^T (\mathbf{y} - H\boldsymbol{\beta}). \end{aligned} \tag{18}$$

where J_k denotes that J is evaluated using the k basis vectors, $\mathbf{y} = (y_1, \ldots, y_M)^T, \boldsymbol{\beta} = \{\beta_i \,|\, i \in S^k\}$, H is an $M \times k$ matrix and is $H = \{H(\mathbf{x}_i, \mathbf{x}_j)\}$.

Partially differentiating J_k with respect to $\boldsymbol{\beta}$, we obtain

$$\frac{\partial J_k}{\partial \boldsymbol{\beta}} = -H^T (\mathbf{y} - H(\boldsymbol{\beta})) = \mathbf{0}. \tag{19}$$

The rank of H is k, because $\{\mathbf{g}(\mathbf{x}_i) \,|\, i \in S^k\}$ are linearly independent. Hence, $H^T H$ is positive definite. Thus, from (19)

$$\boldsymbol{\beta} = (H^T H)^{-1} H^T \mathbf{y}. \tag{20}$$

For k basis vectors, the order of floating operations in solving the above linear equations is $O(k^3)$. Therefore, if N is small the calculation of (20) does not require much computation.

2) Selection of Basis Vectors by KLS: Starting from the empty set, we add one training data sample to the set that minimizes the approximation error approximated by KLS. In the following we show the algorithm.

Step 1
: Let the selected index set be S^k and the candidate index set be T^k. Initially we set $k = 0$, $S^0 = \emptyset$, $T^0 = \{1, \ldots, M\}$.

Step 2
: Assume that we have selected k basis vectors. Then the objective function given by (18) is J_k. Select $i \in T^k$ and evaluate J_k^i, where J_k^i means that $\mathbf{x}_i$ is temporarily add to already selected k basis vectors. If the associated matrix $H^T H$ is singular if we

add $\mathbf{x}_i$, we consider that $\mathbf{x}_i$ does not contribute in improving the approximation error and permanently delete i from T^k. We iterate the above procedure for all i in T^k.

Step 3

Compute

$$i_{\min} = \arg_{i \in T^k} J_k^i \tag{21}$$

and

$$S^{k+1} \leftarrow S^k + \{i_{\min}\}, T^{k+1} \leftarrow T^k - \{i_{\min}\}. \tag{22}$$

Step 4

Check if the obtained basis vector satisfies the stopping condition:

$$J_{k+1} \leq \eta_{\mathrm{k}} J_M, \tag{23}$$

where η_{k} is a parameter determined by cross-validation and J_M is the approximation error using all the data as basis vectors. If (23) is satisfied or $T^k = \emptyset$, terminate the algorithm. Otherwise go to Step2.

The index set S^k includes the indices of training data selected for the S-SVR (k). In this algorithm we need to calculate J_M. This increases the computation burden. But according to our experiments, it is difficult to determine stopping condition other than this. In the future study we need to improve the stopping condition.

IV. Experimental Results

In this section, using five benchmark data sets we show the effectiveness of our proposed method. First we show the performance of the proposed S-SVR (k), in comparison with those of the SVR and S-SVR (c). Then, we compare the numbers of support vectors for the SVR, S-SVR (c), and S-SVR (k). We measure the approximation error by the average absolute approximation error.

A. Comparisons with SVR and S-SVR (c)

Table I lists the data sets used in evaluation: water purification [10], Boston 14 [11], [12], pyrimidines [13], orange juice [14], and abalone [13] data sets.

TABLE I
DATA SETS

Data	Number of data	Input
Water purification	478	10
Boston 14	506	13
Pyrimidines	74	27
Orange juice	218	700
Abalone	4177	8

The water purification data set estimates the amount of chemicals for a water purification plant. The Boston14 data set predicts house price in the Boston area, which is the 14th variable in the Boston data set. But the input and output variables of the pyrimidines data set are not known. The orange juice data set estimates the level of saccharose of orange juice from observed near-infrared spectra. The abalone data set predicts the age of abalone fish from physical measurements.

In all data sets we randomly divide the set into two with almost equal sizes: training data set and test data set. And except for the abalone data set, we create 20 data set pairs and perform 20 experiments and calculate the average absolute approximation error and its standard deviation, and the average number of support vectors and its standard deviation.

In all studies we use RBF kernels:

$$H(\mathbf{x}, \mathbf{x}') = \exp(-\gamma ||\mathbf{x} - \mathbf{x}'||^2), \tag{24}$$

where $\gamma\,(> 0)$ is a parameter for slope control.

In SVRs, we determined the values of the margin parameter, kernel parameter and the error threshold by five-fold cross-validation. In both S-SVR (c) and S-SVR (k), we determined the values of the margin parameter, the error threshold and the parameters for controlling sparsity by cross-validation using the same kernel and parameter value as those of the SVR. In cross-validation, we changed $\gamma = \{0.1, 0.5, 1.0, 5.0, 10, 15, 20\}$, $\varepsilon = \{0.001, 0.01, 0.05, 0.1, 0.5, 1.0\}$, and $C = \{1, 10, 100, 1000, 5000, 10000, 100000\}$. Except for the abalone data set, we changed $\eta_{\mathrm{c}} = \{10^{-2}, 10^{-3}, 10^{-4}, 10^{-5}\}$ for the Cholesky factorization, and for the abalone data set we changed $\eta_{\mathrm{c}} = \{10^{-6}, 10^{-7}\}$. And except for the abalone data set, we changed $\eta_{\mathrm{k}} = \{0.8, 0.5, 0.2, 0.1, 0.08\}$, and for the abalone data set we changed $\eta_{\mathrm{k}} = \{0.08, 0.001\}$.

Table II show the average absolute error and its standard deviation for the SVR, S-SVR (c), and S-SVR (k). For the water purification data set, the average absolute error of the SVR and that of the S-SVR (k) are comparable but for other data sets, the absolute average error of the S-SVR (k) is slightly lower than that of the SVR. But for the orange juice data set, the standard deviation of the S-SVR (k) is much smaller than that of the SVR. Thus the worst average absolute error of the S-SVR (k) is lower than that of the SVR.

TABLE II
AVERAGE ABSOLUTE ERRORS AND STANDARD DEVIATIONS

Data	SVR	S-SVR (c)	S-SVR (k)
Water purification	0.966 ± 0.049	1.029± 0.117	1.022 ± 0.051
Boston 14	2.411± 0.175	2.603 ± 0.342	2.894 ± 0.455
Pyrimidines	0.023± 0.011	0.035± 0.020	0.040± 0.030
Orange juice	7.199± 4.208	7.583± 1.038	7.700± 0.840
Abalone	2.225	2.391	2.676

Table III lists the number of support vectors and its standard deviation. For the five data sets, the numbers of support vectors for the S-SVM (k) are the smallest. And especially for the orange juice and abalone data sets, the reduction is significant.

To compare the generalization abilities of the S-SVR (c) and S-SVR (k) on the orange juice data set, we set the same

TABLE III

AVERAGE NUMBERS OF SUPPORT VECTORS AND STANDARD DEVIATIONS

Data	SVR	S-SVR (c)	S-SVR (k)
Water purification	193.5± 57.7	142.1± 73.8	134.6± 91.7
Boston14	215.2 ± 44.5	132.7 ± 67.3	105.5 ±63.8
Primidines	43.8±5.13	38.0± 9.57	34.8± 8.57
Orange juice	100.2± 23.9	25.5± 30.0	2.0± 0.0
Abalone	1761	222	4

kernel parameter values and the same numbers of support vectors. Table IV shows the results. From the table, it is clear that for the same number of support vectors of 2 and 25, the S-SVR (k) shows better performance.

TABLE IV

COMPARISON OF S-SVR (C) AND S-SVR (K) UNDER THE SAME KERNEL AND SAME NUMBER OF SUPPORT VECTORS FOR THE ORANGE JUICE DATA SET

SVs	S-SVR (c)	S-SVR (k)
2	8.246± 0.953	7.700± 0.840
25	7.666± 0.954	7.331± 0.772

For the Boston data set, the average absolute approximation error for the S-SVR is worse than that of the regular SVR. This may be caused by the fact that the same kernel parameter value is used for the S-SVR (k). Then for the number of support vectors of 50 and 100, we determine the value of γ by fivefold cross-validation. Table V shows the result. Even with 50 support vectors, the approximation performance is lower than the corresponding result shown in Table II and with 100 support vectors, the approximation performance is comparable with that of the regular SVR.

TABLE V

AVERAGE ABSOLUTE ERRORS AND STANDARD DEVIATIONS BY THE S-SVR (K) WITH THE OPTIMIZED γ FOR THE BOSTON 14 DATA SET

Data	50 SVs	100 SVs
Boston 14	2.728 ± 0.212	2.551 ± 0.197

Table VI lists the number of γ values selected for 20 trials when the γ values are optimized by cross-validation. From the table, the optimal values for the S-SVR are different from those of the SVR.

TABLE VI

THE NUMBER OF SELECTED γ VALUES FOR THE BOSTON 14 DATA SET

γ	SVR	S-SVR(k)-50SVs	S-SVR(k)-100SVs
0.1	3	0	0
0.5	2	0	0
1.0	1	3	3
5.0	11	11	7
10	3	6	8
20	0	0	2

V. CONCLUSIONS

We proposed sparse support vector regressors based on forward selection of basis vectors. First the training data that span the reduced empirical feature space are selected one by one by forward selection based on the approximation error evaluated by kernel least squares. Then, the support vector regressor is trained in the empirical feature space. Because the mapped support vectors of the regressor are expressed by the selected, mapped training data, the complexity of the regressor is determined by the number of the selected data.

We compared the method with the regular SVM and the sparse SVR, in which the basis vectors are selected by the Cholesky factorization, and showed that the proposed method gives sparser solutions. The reason why the approximation performance is slightly inferior to the regular SVR is that we set the same kernel parameter value as that of the regular SVR. Thus, to improve the approximation performance we need to optimize the kernel parameter value.

As the future work we need to study a new stopping condition for basis vector selection and a method for optimizing the γ value with less computation burden.

REFERENCES

[1] K. R. Müller, A. J. Smola, G. Rätsch, B. Schölkopf, J. Kohlmorgen, and V. Vapnik, "Predicting time series with support vector machines," In W. Gerstner, A. Germond, M. Hasler, and J.D. Nicoud, editors, *Artificial Neural Networks (ICANN '97) Proceedings of the Seventh International Conference, Lausanne, Switzerland*, pp. 999-1004, Springer-Verlag, Berlin, 1997.

[2] S. Abe, "Sparse least squares support vector training in the reduced empirical feature space," *Pattern Analysis and Applications*, pp. 203-214, Springer-Verlag, 2006.

[3] S. Abe and K. Onishi, "Sparse least squares support vector regressors trained in the reduced empirical feature space," In *Proceedings of the International Conference on Artificial Neural Networks (ICANN)*, pp. 529-536, Springer-Verlag, 2007.

[4] J. A. K. Suykens, T. Van Gestel, J. De Brabanter, B. De Moor, and J. Vamdewalle, "Least squares support vector machines," World Scientific, 2002.

[5] L. Hoegaerts, J. A. K. Suykens, J. Vamdewalle, B. De Moor, "Subset based least squares subspace regression in RKHS," *Neurocomputing*, Vol. 63, pp. 293-323, 2005.

[6] L. Hoegaerts, J. A. K. Suykens, J. Vamdewalle, B. De Moor, "Primal space sparse kernel partial least squares regression for large scale problems," *Proceedings of the International Joint Conference on Neural Networks (IJCNN 2004)*, pp. 561-566, 2004.

[7] X. X. Wang, S. Chen, D. Lowe, C. J. Harris, "Sparse support vector regression based on orthogonal forward selection for the generalised kernel model," *Neurocomputing*, Volume 70, pp. 462-474, 2006.

[8] L. Jiao, L. Bo, L. Wang, "Fast sparse approximation for least squares support vector machine," *IEEE Transactions on Neural Networks*, Vol. 18, No. 3, 2007.

[9] H. Xiong, M. N. S. Swamy, and M. O. Ahmad, "Optimizing the kernel in the empirical feature space," *IEEE Transactions on Neural Networks*, Vol. 16, pp. 460-474, 2005.

[10] S. Abe, *Neural Networks and Fuzzy Systems: Theory and Applications*, Kluwer, Boston, MA, 1997.

[11] D. Harrison and D. L. Rubinfeld, "Hedonic prices and the demand for clean air," *Journal of Environmental and Management*, Vol. 5, pp. 81-102, 1978.

[12] http://www.cs.toronto.edu/~delve/data/datasets.html.

[13] C. Blake and C. Merz, UCI repository of machine learning databases, http://www.ics.uci.edu/~mlearn/MLRepository.html, University of California, Irvine, Department of Information and Computer Sciences, 1998.

[14] http://www.ucl.ac.be/mlg/index.phppage=home.