On the Sparseness and Generalization Capability of Least Squares Support Vector Machines

Aijun YAN

College of Electronic Information and Control Engineering, Beijing University of Technology, Beijing 100124, China; Beijing Key Laboratory of Computational Intelligence and Intelligent System, Beijing 100124, China; Engineering Research Center of Digital Community, Ministry of Education, Beijing 100124, China

E-mail: yanaijun@bjut.edu.cn

Xiaoqian HUANG

College of Electronic Information and Control Engineering, Beijing University of Technology, Beijing 100124, China; Engineering Research Center of Digital Community, Ministry of Education, Beijing 100124, China; Beijing Laboratory for Urban Mass Transit, Beijing 100124, China

E-mail: hxq0407@emails.bjut.edu.cn

Hongshan SHAO

College of Electronic Information and Control Engineering, Beijing University of Technology, Beijing 100124, China

E-mail: shaohs@emails.bjut.edu.cn

Abstract Compared with standard support vector machines (SVM), sparseness is lost in the modeling process of least squares support vector machines (LS-SVM), causing limited generalization capability. An improved method using quadratic renyi-entropy pruning is presented to deal with the above problems. First, a kernel principal component analysis (KPCA) is used to denoise the training data. Next, the authors use the genetic algorithm to estimate and optimize the kernel function parameter and penalty factor. Then, pick the subset that has the largest quadratic entropy to train and prune, and repeat this process until the cumulative error rate reaches the condition requirement. Finally, comparing experiments on the data classification and regression indicates that the proposed method is effective and may improve the sparseness and the generalization capability of LS-SVM model.

Keywords LS-SVM; sparseness; generalization capability; quadratic renyi-entropy pruning

1 Introduction

The least squares support vector machine (LS-SVM) is a machine learning method following the principle of structural risk minimization through changing the constraint condition and risk function of the standard support vector machine (SVM). In fact, LS-SVM is a special form of standard SVM under quadratic loss function. When dealing with problems in the LS-SVM,

Received August 21, 2014, accepted September 11, 2014

Supported by the National Natural Science Foundation of China (61374143) and the Beijing Natural Science Foundation of China (4152010)

solving a quadratic programming problem is transformed into solving a set of linear equations by using a set of equality constraints instead of inequality constraints of SVM, which is widely used and greatly improves computational efficiency^[1]. However, LS-SVM also has some shortcomings in the modeling process, such as loss of sparseness of the standard SVM and limited fitting and generalization capability.

A review of literature showed that some fruitful researches have been done about the sparseness of LS-SVM. Pelckmans et al. used the model hierarchical strategy to regain sparseness^[2]. A new guideline that restores the sparseness of LS-SVM by assigning values to the Lagrange multiplier was proposed by Carvalho et al.^[3]. Yang et al. proposed a self-adaptive pruning algorithm that is implemented from bottom to top. In the algorithm, the learning process of increment and decrement is alternately used to form a small set of support vectors, which can cover most of the information included in the training set^[4]. Traditional pruning methods are used only in the input space. Then the method of fuzzy LS-SVM modeling using density index function in a high-dimensional feature space is proposed^[5]. These methods achieve commendable results in sparseness but another performance that cannot be ignored is generalization. To improve generalization capability when transforming the prediction of long time series of the LS-SVM into regression problems, Rubio et al. put forward a weighted K-nearest neighbour to improve the kernel function of LS-SVM, which reduces the cost of computation and acquires a better generalization capability^[6]. Karsmakers et al. pointed out that a model with high sparseness tends to have better generalization performance and computing speed in machine learning. Therefore, they proposed a general sparseness algorithm, which improves the generalization capability of the model by traversing an iterative conjugate support vector incremental group and by solving a relatively small subsystem in each iteration process^[7]. Later, an improved adaptive pruning algorithm is proposed, which can improve the efficiency of dealing with large-scale problems and ensure the generalization performance as well^[8]. According to the above analysis, how to improve the sparseness and generalization capability of the LS-SVM model has important research value^[9].

To solve the loss of sparseness and weak generalization performance in the process of LS-SVM modeling, a pruning algorithm based on quadratic entropy is proposed in this paper. First, a kernel principal component analysis (KPCA) is used to denoise the training data. Then the quadratic entropy is introduced as the basis of the pruning algorithm, that is, according to the relative importance of the training data to remove data with minimum error. The algorithm guarantees good generalization capability of the model while improving sparseness. In addition, the effectiveness of the method is illustrated by using historical data of the furnace roasting process and examples of classification data set.

The rest of the paper is organized as follows. Section 2 introduces the LS-SVM modeling process, evaluation metrics, and problem analysis. Section 3 discusses the improved LS-SVM modeling algorithm. Section 4 verifies the effectiveness of the presented method via two experiments. Conclusions and further work are then discussed in Section 5.

2 Related Work

2.1 Modeling Process

Without loss of generality, assume that $S = \{(x_i, y_i), y_i \in \{-1, +1\}\}$, $i = 1, 2, \dots, l$, is the training set of binary classification problem, where $x_i \in R^n$ is the i th input vector, y_i is the class label of x_i , and l denotes the number of samples. Solving the binary classification problem determines an LS-SVM classification function $y(x)$ to distinguish class labels of different samples. By solving the following constrained optimization problem:

$$\begin{cases} \min_{w, b, e} & Q(w, b, e) = \frac{1}{2} \|w\|^2 + \frac{\gamma}{2} \sum_{i=1}^l e_i^2 \\ \text{s.t.} & y_i(w^T \phi(x_i) + b) = 1 - e_i, \quad i = 1, 2, \dots, l \end{cases} \quad (1)$$

where $\gamma > 0$ is the penalty factor, e_i is the classification error, and $\phi(\cdot)$ is the nonlinear mapping that the input space maps to high-dimension feature space, that is, $\phi : x \rightarrow \phi(x)$. In addition, the coefficient vector w and the bias term b are the unknown parameters. Assuming that x is the input vector to be classified, the classification function can be derived by constructing a Lagrange function:

$$y(x) = \text{sign} \left(\sum_{i=1}^l a_i y_i K(x, x_i) + b \right) \quad (2)$$

where $\text{sign}(\cdot)$ is a sign function, whose output corresponds to the class label (1 or -1). a_i ($i = 1, 2, \dots, l$) and b represent the Lagrange multiplier and the bias term, respectively. $K(x, x_i)$ denotes a kernel function satisfying Mercer's condition, and usually let $K(x, x_i)$ be the Gaussian RBF kernel function^[10]:

$$K(x, x_i) = \langle \phi(x), \phi(x_i) \rangle = \exp \left(- \frac{\|x - x_i\|^2}{\sigma^2} \right) \quad (3)$$

where σ is the width of the Gaussian kernel function which reflects the characteristics of the training data set and plays an important role in improving the generalization capability of the system.

When LS-SVM is used to solve the regression problem, the training set will become $S = \{(x_i, y_i)\}$, $i = 1, 2, \dots, l$, where y_i is the target value of x_i . This is different from solving a pattern classification problem. Therefore, the fitting function is expressed as follows^[11]:

$$y(x) = \sum_{i=1}^l a_i K(x, x_i) + b \quad (4)$$

2.2 Evaluation Metrics

The performance of the model LS-SVM established is generally evaluated by calculating the sparseness, function approximation error and classification accuracy, etc. When the model is applied to the classification problem, classification accuracy is commonly used to measure the performance of the model, namely the ratio of the number of samples correctly classified divided by that of the entire test set. The higher the ratio is, the greater the classification accuracy is. According to the description of the sparseness of the model, it can be calculated by

$$s = \left(1 - \frac{l_v}{l} \right) \times 100\% \quad (5)$$

where s represents the sparseness of the model, l_v is the number of support vectors, and l is the total number of samples. Obviously, the fewer the number of support vectors is, the higher the sparseness will be.

The performance of function approximation is measured by means of the loss function. The average loss of the entire test set is calculated by the relative root mean square error:

$$e_{\text{ms}} = \sqrt{\frac{\sum_{i=1}^d (y_i - y_i')^2}{d}} \quad (6)$$

where d is the number of test set and y_i' is the output of the model. However, the presence of outliers has a great impact on the calculation of the mean square error. Hence, the relative squared error can be used to measure the total loss of the entire test set:

$$e_{\text{rs}} = \frac{\sum_{i=1}^d (y_i - y_i')^2}{\sum_{i=1}^d (y_i - \bar{y})^2} \quad (7)$$

where $\bar{y}$ means the average value of model output on the entire set of tests.

2.3 Problem Analysis

As a special form of standard SVM, LS-SVM greatly improves the efficiency of solving problem by using a set of equality constraints instead of inequality constraints of SVM. Nevertheless, to obtain (2) and (4), it is necessary to deal with a quadratic programming problem, which leads to a large amount of calculation as well as a slow learning speed. Furthermore, almost all of the training data are used as support vector in the LS-SVM modeling process, so that all training samples should be used again to solve problems when adding a new input. This causes a significant decline of the sparseness defined in (5) and a substantial increase of computation. In addition, the sparseness will be worse while the amount of data increases, which affects the overall running time and the generalization capability of the model. Therefore, it is important to identify the effective support vectors.

Next, the penalty factor γ in (1) determines the degree of complexity of the model as well as the degree of penalty of fitting offset, and the width of the Gaussian kernel function σ in (3) reflects the characteristic of the training data set. The two parameters should be determined in advance because they play an important role in improving the generalization capability of the system.

Moreover, the quality of data will directly affect the generalization capability of the model for a fixed model. The training error will increase when the data contain a lot of noise. Consequently, the training data denoising can reduce the training error and further improve the generalization capability of the model.

Therefore, sparseness and generalization should be considered in the process of LS-SVM modeling. Specifically, dealing with the sample data, selecting the penalty factor γ and the width of the Gaussian kernel function σ , and determining support vectors are significant to improve the performance of the LS-SVM model.

3 An Improved LS-SVM Algorithm

Based on the above analysis, the method of LS-SVM modeling is introduced in detail: data processing, parameter estimation, quadratic renyi-entropy pruning, and algorithm steps.

3.1 Data Processing

Through nonlinear mapping $\varphi(\cdot)$, the training data set S is mapped to the feature space, denoted as $\varphi(x_i)$, the covariance matrix in the feature space is

$$C = \frac{1}{l} \sum_{i=1}^l \varphi(x_i) \varphi^T(x_i) \quad (8)$$

Let the eigenvalue and eigenvector of matrix C be λ and V , then $\lambda V = CV$. The following equation can be obtained after standardization of the covariance matrix C in the feature space:

$$\lambda \sum_{i=1}^l \alpha_i \langle \varphi(x_k), \varphi(x_i) \rangle = \frac{1}{l} \sum_{i=1}^l \alpha_i \left\langle \varphi(x_k), \sum_{j=1}^l \varphi(x_j) \right\rangle \langle \varphi(x_j), \varphi(x_i) \rangle, \quad k = 1, 2, \dots, l \quad (9)$$

where α_i is a coefficient. Define a core matrix $K_{l \times l}$ according to the nuclear function shown by (3), then the above equation becomes

$$l\lambda\alpha = K\alpha \quad (10)$$

where $\alpha_i = (\alpha_1, \alpha_2, \dots, \alpha_l)^T$. Thus, the problem of seeking the eigenvalue λ and eigenvector V of matrix C has been turned into seeking these values of matrix K . Therefore, to extract the principal components in the feature space, only the calculation of the mapping of $\varphi(x)$ on V^k is needed:

$$h_k(x) = \langle V^k, \varphi(x) \rangle = \sum_{i=1}^l \alpha_i^k \langle \varphi(x_i), \varphi(x) \rangle = \sum_{i=1}^l \alpha_i^k K(x_i, x) \quad (11)$$

where $h_k(x)$ is the k -th nonlinear principal component of the corresponding nonlinear mapping, and feature value λ_k corresponds to $h_k(x)$, the size of which reflects the degree of the influence of noise. The main element is usually determined by defining the cumulative contribution rate, if the following formula is established:

$$\frac{\sum_{i=1}^p \lambda_i}{\sum_{i=1}^l \lambda_i} \geq 85\% \quad (12)$$

It shows that the former p components are the main elements in the feature space, and the other $l - p$ components are caused by noise.

3.2 Parameter Estimation and Optimization

The selection of the penalty factor γ in (1) and the width of the Gaussian kernel function σ in (3) plays an important role in improving the generalization capability of the system, and σ mainly controls the distribution complexity of sample data in the high dimensional feature space and γ is a counterbalance between complexity and the fitting error, which need to be determined in advance. The two parameters γ and σ are estimated by genetic algorithms. In the process of estimation, it is necessary to define a fitness function that is conducive to the iterative learning of parameters. In classification problems, the accuracy rate can be used as the fitness function, but in function approximation problems where the performance is based on loss function, the generalization error or relative error can be considered in the fitness function. The process of parameter estimation is as follows:

Step 1 Encoding. Let the two parameters γ and σ correspond to a gene, respectively, and two genes form an individual. Then, they are coded by the 4-bit binary.

Step 2 Decoding. Translate the codes of the parameters into values.

Step 3 Generating a group $p(t)$.

Step 4 Evaluation. Calculate all of the individuals' fitness in $p(t)$ and execute the following operation of evolution.

Step 5 Selection. First, calculate the probability of each individual to be selected. Next, conduct several rounds of selection. Then, select the appropriate individuals. Besides, the individuals selected will be used as the parents to breed offspring and added to $p(t+1)$.

Step 6 Crossover. Use the single-point crossover to form two new individuals.

Step 7 Mutation. Select a number of individuals from $p(t+1)$ randomly, and inverse one of the bits in the gene randomly.

After the above operation is completed, a new generation of individuals and groups $p(t+1)$ are formed. Next, repeat the above evaluation, selection, crossover, and mutation operations until the output of the model meets the requirements or the required number of iterations are achieved.

3.3 Quadratic Renyi-Entropy Pruning

The LS-SVM model takes all of the training data as support vectors, which leads to the poor sparseness defined by (5). In addition, all training samples are required to be calculated again while a new input is added. This increases the amount of computation sharply. Therefore, the effective support vectors should be determined. In classification problems, support vectors are generally located on the boundary of two types of data. In addition, they are usually close or mixed together. In the view of information entropy, the more chaotic data have larger entropy [12], so that support vectors are most likely to appear in the maximum entropy. The quadratic entropy is defined by this principle as follows:

$$H_2 = -\log \int p^2(x)dx \quad (13)$$

where $p(x)$ is the probability density function of the data points and can be estimated according to the following formula^[13]:

$$\int p^2(x)dx \approx \int \hat{p}(x)dx = \frac{1}{l^2} \sum_{i=1}^l \sum_{j=1}^l K(x_i, x_j) \quad (14)$$

Hence, the quadratic entropy can be approximated as follows:

$$H_2 \approx -\log \left(\frac{1}{l^2} \sum_{i=1}^l \sum_{j=1}^l K(x_i, x_j) \right) \quad (15)$$

From the above calculation, a smaller data subset containing the vast majority of support vectors can be determined according to the size of the information entropy.

Next, the data subset obtained will be pruned by the method of regularization which takes into account the sample position^[14]. The approximation error will emerge when pruning the regularized data. Considering the amount of information contained in each sample, the spatial distribution of the input samples can minimize the importing error and wipe out samples containing the least comprehensive information. The formula of minimizing importing error is:

$$d(x_i) = \frac{c_i}{[A^{-1}]_{ii}} \quad (16)$$

where $d(x_i)$ is the iterative error after deleting x_i ; matrix A contains the nuclear matrix and the unit matrix; and c_i is the value of the corresponding support vector. The sample with minimum iterative error will be cut and will replace the sample with the smallest c_i by comparing x_i . The termination condition of the pruning process can be set to 80% of the performance before pruning^[14].

3.4 Algorithm Description

According to the above introduction of the quadratic entropy pruning algorithm, the training data set S can be divided by n subsets randomly. After calculating and sorting the quadratic entropy of n subsets, the subset with the maximum quadratic entropy will be trained and then pruned. The above training process is repeated until the termination condition is met. The termination condition can be the cumulative error rate:

$$s = \sum_{i=1}^l \frac{y(x_i) - y_i}{y_i} \quad (17)$$

The concrete steps of the quadratic entropy pruning algorithm are as follows:

Step 1 Let training termination conditions be the cumulative error rate, a positive constant; and pruning termination condition be the performance of the model down to 80% before pruning.

Step 2 Determine the parameters γ and σ by genetic algorithm.

Step 3 Denoise the training data by the KPCA algorithm.

Step 4 Divide the training data S into n subsets randomly; then, calculate the quadratic entropy of each subset and arrange them in descending order.

Step 5 Train each subset to obtain the support vector matrix.

Step 6 Calculate the iterative error and delete the data corresponding to the minimum error, and then retrain.

Step 7 Judge if the pruning termination condition is satisfied or not, that is, whether the performance of the model is down to the original 80% or not. If it is satisfied, move to Step 9; if not, go back to Step 5.

Step 8 Loop variable plus one and continue to train the model.

Step 9 Calculate the cumulative error rate; if the error rate meets the training termination conditions, move to Step 10. Otherwise, go back to Step 8.

Step 10 Regard the last calculated $y(x)$ as the final LS-SVM model.

4 Experimental Results

To investigate the effectiveness of the proposed method, the problem of data classification^[15] and furnace roasting process variable regression were studied. According to the results, the method of data denoising, the GA parameter estimation, the quadratic entropy pruning (denoted by KPAS-LS-SVM) in this paper were compared with other methods such as BP neural network (denoted by NN), standard LS-SVM, and traditional pruning after KPCA denoising LS-SVM (KPS-LS-SVM). Experiments were conducted on a computer with CPU Intel ® Core 2 Quad Q9550@2.83 GHz, 4GB of memory; and the algorithm software is Matlab 7.0.

4.1 Data Classification

In this experiment, classification accuracy and the sparseness of the model were studied. Four categorical data sets were selected from UCI: breast cancer diagnosis (Breast), liver function diagnostics (Liver), the wine classification (Wine), and glass species identification (Glass). Basic information of the four data sets is shown in Table 1.

Table 1 Information on the data sets

Data sets	Number of samples	Number of attributes	Number of classes
Breast	569	32	2
Liver	345	7	2
Wine	178	13	3
Glass	214	10	7

To quantify the performance of the mentioned methods, the experiment uses a ten-fold cross-validation method. The learning error of the NN was set at 0.02, and the number of iterations was set at 5000. The accuracy of classifiers and the sparseness of KPAS-LS-SVM, LS-SVM and KPS-LS-SVM were calculated after the classification test.

The accuracies for each classification model are summarized in Table 2. Notice that the accuracy of the KPS-LS-SVM model that used KPCA denoising is higher than the standard LS-SVM model and the NN model in the four data sets. Moreover, the accuracy of the KPAS-LS-SVM model that used the KPCA denoising and quadratic entropy pruning classification was further improved. Table 3 shows the sparseness for each model. Obviously, the sparseness of KPAS-LS-SVM classification model proposed in this paper is better.

Table 2 Classification accuracy of test sets

Data sets	LS-SVM (%)	KPS-LS-SVM (%)	KPAS-LS-SVM (%)	NN (%)
Breast	96.9±1.2	97.61.0	98.1±0.5	92.7±1.1
Liver	76±4.8	80.3±4.3	83.22±3.2	78.3±4.9
Wine	95.5±1.3	96.7±1.6	97.8±1.1	91.2±1.2
Glass	72.8±7.4	75.2±6.6	78.8±5.3	73.4±5.8

Table 3 Sparseness of each model

Data sets	LS-SVM (%)	KPS-LS-SVM (%)	KPAS-LS-SVM (%)
Breast	0	63.4±2.9	75.8±1.1
Liver	0	36.1±3.1	53.9±2.8
Wine	0	40.5±2.7	54.9±1.5
Glass	0	47.4±3.4	63.4±1.9

According to the experimental results, it is necessary to denoise and clean the data in the process of designing the LS-SVM classifier. In addition, the quadratic entropy pruning algorithm and GA optimization parameters can improve the classifier's generalization and sparseness.

4.2 Data Regression

Shaft furnace roasting is an important sub-process of ore mining in the metallurgical industry. The optimal control of the process has shown that with simple loop stability control, it is difficult to regulate product quality, product yield, and energy consumption. Main controlled variables that affect the above production targets should be set reasonably. The process of shaft furnace roasting is very complicated. It has the features of multivariable, nonlinear, and strong coupling. Thus, it is difficult to establish a precise mechanism model. Therefore, the process model of key variables can be established based on the method proposed in this paper.

The regression model of key variables is obtained by using the modeling method of standard SVM, LS-SVM, and KPAS-LS-SVM, respectively. According to (5) to (7), the sparseness, root mean square error, and relative squared error of each model were calculated to compare the performance of the above three modeling methods. The results are shown in Table 4.

Table 4 Comparison of experimental results

name	root mean square error	relative squared error (%)	sparseness (%)
LS-SVM	0.0548±0.0152	0.6115±0.0127	0
SVM	0.0365±0.0101	0.5891±0.0093	65.56
KPAS-LS-SVM	0.0316±0.0087	0.5781±0.0102	64.43

KPAS-LS-SVM basically recovers the model sparseness and reduces the root mean square error as well as the relative squared error compared with the standard SVM. Compared with LS-SVM, KPAS-LS-SVM also improves sparseness, root mean square error, and relative squared error.

5 Conclusions

LS-SVM lost the sparseness of standard SVM in the modeling process. Thus, this paper describes the process of combining quadratic entropy with improved pruning algorithm to deal with the LS-SVM model, and takes advantage of KPCA to clean data and improve data quality. Genetic algorithm is applied to the parameter optimization process of the LS-SVM model. The parameters obtained ensure better overall performance of the model. These experimental results prove that this method can effectively improve the sparseness and generalization performance of LS-SVM model in the field of data classification and regression. Further work lies in LS-SVM online modeling method and the balance between the speed and accuracy of the LS-SVM model. With data denoising, pruning, and parameter optimization, the requirements of online application in the field of real-time modeling may be met.

References

- [1] Zhang W, Niu P, Li G, et al. Forecasting of turbine heat rate with online least squares support vector machine based on gravitational search algorithm. *Knowledge-Based Systems*, 2013, 39(2): 34–44.
- [2] Pelckmans K, Suykens J A K, De Moor B. Building sparse representations and structure determination on LS-SVM substrates. *Neurocomputing*, 2005, 64(1–4 SPEC. ISS.): 137–159.
- [3] Carvalho B P R, Braga A P. IP-LSSVM: A two-step sparse classifier. *Pattern Recognition Letters*, 2009, 30(16): 1507–1515.

- [4] Yang X, Lu J, Zhang G. Adaptive pruning algorithm for least squares supports vector machine classifier. *Soft Computing*, 2010, 14(7): 667–680.
- [5] Li Y, Tang W, Li M. A least squares support vector machine sparseness algorithm. *Journal of Convergence Information Technology*, 2012, 7(13): 240–248.
- [6] Rubio G, Herrera L J, Pomares H, et al. Design of specific-to-problem kernels and use of kernel weighted K-nearest neighbours for time series modeling. *Neurocomputing*, 2010, 73(10–12): 1965–1975.
- [7] Karsmakers P, Pelckmans K, De Brabanter K, et al. Sparse conjugate directions pursuit with application to fixed-size kernel models. *Machine Learning*, 2011, 85(1–2): 109–148.
- [8] Gao R, San Y. Improved adaptive pruning algorithm for least squares support vector regression. *Journal of Systems Engineering and Electronics*, 2012, 23(3): 438–444.
- [9] Wei L, Chen Z, Li J. Evolution strategies based adaptive L_p LS-SVM. *Information Sciences*, 2011, 181(14): 3000–3016.
- [10] Zanaty E A, Afifi A. Support vector machines (SVMs) with universal kernels. *Applied Artificial Intelligence*, 2011, 25(7): 575–589.
- [11] Yang L, Yang S, Zhang R, et al. Sparse least square support vector machine via coupled compressive pruning. *Neurocomputing*, 2014, 131(5): 77–86.
- [12] Liang J, Zhao X, Li D, et al. Determining the number of clusters using information entropy for mixed data. *Pattern Recognition*, 2012, 45(6): 2251–2265.
- [13] Hong X, Chen S, Harris C J. Using zero-norm constraint for sparse probability density function estimation. *International Journal of Systems Science*, 2012, 43(11): 2107–2113.
- [14] De Kruif B J, De Vries T J A. Pruning error minimization in least squares support vector machines. *IEEE Transactions on Neural Network*, 2003, 14(3): 696–702.
- [15] Frank A, Asuncion A. UCI machine learning repository. Irvine, CA: University of California, School of Information and Computer Science. Available at: <http://archive.ics.uci.edu/ml>, 2010.