Support Vector Machine Regression Algorithm Based on Chunking Incremental Learning

Jiang Jingqing[1, 2], Song Chuyi[2], Wu Chunguo[1, 3], Marchese Maurizio[4], and Liang Yangchun[1,4,*]

[1] College of Computer Science and Technology, Jilin University, Key Laboratory of Symbol Computation and Knowledge Engineering of Ministry of Education, Changchun 130012, China
[2] College of Mathematics and Computer Science, Inner Mongolia University for Nationalities, Tongliao 028043, China
[3] The Key Laboratory of Information Science & Engineering of Railway Ministry/The Key Laboratory of Advanced Information Science and Network Technology of Beijing, Beijing Jiaotong University, Beijing 100044, China
[4] Department of Information and Communication Technology, University of Trento, Via Sommarive 14, 38050, Povo (TN) Italy

Abstract. On the basis of least squares support vector machine regression (LSSVR), an adaptive and iterative support vector machine regression algorithm based on chunking incremental learning (CISVR) is presented in this paper. CISVR is an iterative algorithm and the samples are added to the working set in batches. The inverse of the matrix of coefficients from previous iteration is used to calculate the regression parameters. Therefore, the proposed approach permits to avoid the calculation of the inverse of a large-scale matrix and improves the learning speed of the algorithm. Support vectors are selected adaptively in the iteration to maintain the sparseness. Experimental results show that the learning speed of CISVR is improved greatly compared with LSSVR for the similar training accuracy. At the same time the number of the support vectors obtained by the presented algorithm is less than that obtained by LSSVR greatly.

1 Introduction

The support vector machine (SVM) is a novel learning method that is constructed based on statistical learning theory. The support vector machine has been studied widely since it was presented in 1995. It has been applied to pattern recognition broadly and its excellent performance has been shown in function regression problems. Training a standard support vector machine requires the solution of a large-scale quadratic programming problem. This is a difficult problem when the number of the samples exceeds a few thousands. Many algorithms for training the SVM have been studied. Osuua [1] proposed a decomposition algorithm and the quadratic programming problem for standard SVM is divided into a serial small-scale quadratic programming sub-problem. Focusing on the problem of the working set selection, Joachims [2] presented a SVM^{Light} algorithm to implement the decomposition algorithm in [1] efficiently. A sequential minimal optimization algorithm (SMO) was proposed by Patt [3]. It transformed the quadratic programming problem for standard

* Corresponding author.

V.N. Alexandrov et al. (Eds.): ICCS 2006, Part I, LNCS 3991, pp. 547 – 554, 2006.

SVM to the minimization quadratic programming problem that could be solved analytically. Suykens [4] suggested a least squares support vector machine (LSSVM) in which the inequality constrains were replaced by equality constrains. By this way, solving a quadratic programming was converted into solving linear equations. The efficiency of training SVM is improved greatly and the difficulty of training SVM is cut down. Suykens [5] studied the LSSVM for function regression further. Hao [6] proposed a chunking incremental learning algorithm for LSSVM to deal with classification problem. In this paper, an adaptive and iterative support vector machine regression algorithm based on chunking incremental learning (CISVR) is presented. The support vectors are selected adaptively in the iteration to maintain the sparseness and the samples are added to working set in batches.

2 Least Squares Support Vector Machine for Regression (LSSVR)

According to [5], let us consider a given training set of l samples $\{x_i, y_i\}_{i=1}^{l}$ with the ith input datum $x_i \in R^n$ and the ith output datum $y_i \in R$. The aim of support vector machine model is to construct the decision function that takes the form:

$$f(x, w) = w^T \varphi(x) + b \tag{1}$$

where the nonlinear mapping $\varphi(\cdot)$ maps the input data into a higher dimensional feature space. In least squares support machine for function regression the following optimization problem is formulated

$$\min_{w,e} J(w, e) = \frac{1}{2} w^T w + \gamma \sum_{i=1}^{l} e_i^2 \tag{2}$$

subject to the equality constraints

$$y_i = w^T \varphi(x_i) + b + e_i, \quad i = 1, ..., l \tag{3}$$

This corresponds to a form of ridge regression. The Lagrangian is given by

$$L(w, b, e, \alpha) = J(w, e) - \sum_{i=1}^{l} \alpha_i \{ w^T \varphi(x_i) + b + e_i - y_i \} \tag{4}$$

with Lagrange multipliers α_k. The conditions for the optimality are

$$\begin{cases} \dfrac{\partial L}{\partial W} = 0 \rightarrow w = \sum_{i=1}^{l} \alpha_i \varphi(x_i) \\ \dfrac{\partial L}{\partial b} = 0 \rightarrow \sum_{i=1}^{l} \alpha_i = 0 \\ \dfrac{\partial L}{\partial e_i} = 0 \rightarrow \alpha_i = \gamma e_i \\ \dfrac{\partial L}{\partial \alpha_i} = 0 \rightarrow w^T \varphi(x_i) + b + e_i = 0 \end{cases} \tag{5}$$

for $i = 1,...,l$. After eliminating e_i and w, we could have the solution by the following linear equations

$$\begin{bmatrix} 0 & \vec{1}^T \\ \vec{1} & \Omega + \gamma^{-1} I \end{bmatrix} \begin{bmatrix} b \\ a \end{bmatrix} = \begin{bmatrix} 0 \\ y \end{bmatrix} \tag{6}$$

where $y = [y_1,..., y_l]^T, \vec{1} = [1,...,1]^T, \alpha = [\alpha_1,..., \alpha_l]^T$ and the Mercer condition

$$\Omega_{kj} = \varphi(x_k)^T \varphi(x_j) = \psi(x_k, x_j) \qquad k, j = 1,...,l \tag{7}$$

is applied. Set $A = \Omega + \gamma^{-1} I$. If A is a symmetric and positive-definite matrix, A^{-1} exists. Solving the linear equations (6) we obtain the solution

$$\alpha = A^{-1}(y - b\vec{1}) \qquad b = \frac{\vec{1}^T A^{-1} y}{\vec{1}^T A^{-1} \vec{1}} \tag{8}$$

Substituting w in Eq. (1) with the first equation of Eqs. (5) and using Eq. (7) we have

$$f(x, w) = y(x) = \sum_{i=1}^{l} \alpha_i \psi(x, x_i) + b \tag{9}$$

where α_i and b are the solution to Eqs. (6). The kernel function $\psi(\cdot)$ can be chosen as linear function $\psi(x, x_i) = x_i^T x$, polynomial function $\psi(x, x_i) = (x_i^T x + 1)^d$ or radial basis function $\psi(x, x_i) = \exp\{-\|x - x_i\|_2^2 / \sigma^2\}$.

3 Adaptive and Iterative Least Squares Support Vector Machine Regression Algorithm Based on Chunking Incremental Learning

3.1 Chunking Increment Procedure

According to Eq. (6), set

$$A_N = \Omega + \gamma^{-1} I \qquad \overline{\alpha}_N = \alpha \qquad \overline{y}_N = y \tag{10}$$

where N is the number of samples in current working set. Eq. (8) can be rewritten as

$$\overline{\alpha}_N = A_N^{-1}(\overline{y}_N - b\vec{1}) \qquad b = \frac{\vec{1}^T A_N^{-1} \overline{y}_N}{\vec{1}^T A_N^{-1} \vec{1}} \tag{11}$$

$\vec{1} = (1,...,1)^T$. When K new coming samples $(x_{N+1}, y_{N+1}),(x_{N+2}, y_{N+2}),...,(x_{N+K}, y_{N+K})$ are added to the current working set, we could calculate the parameters according to Eq. (12)

$$\overline{\alpha}_{N+K} = A_{N+K}^{-1}(\overline{y}_{N+K} - b\vec{1}) \qquad b = \frac{\vec{1}^T A_{N+K}^{-1} \overline{y}_{N+K}}{\vec{1}^T A_{N+K}^{-1} \vec{1}} \tag{12}$$

where $\vec{1} = (1,...,1)^T$, $\overline{\alpha}_{N+K} = (\overline{\alpha}_N, \alpha_{N+1}, ..., \alpha_{N+K})$, $\overline{y}_{N+K} = (\overline{y}_N, y_{N+1}, ..., y_{N+K})$,

$$A_{N+K} = \begin{bmatrix} A_N & Q \\ Q^T & S \end{bmatrix} \qquad Q = \begin{bmatrix} \Omega_{1,N+1} & \Omega_{1,N+2} & ... & \Omega_{1,N+K} \\ ... & ... & ... & ... \\ \Omega_{N,N+1} & \Omega_{N,N+2} & ... & \Omega_{N,N+K} \end{bmatrix}$$

$$S = \begin{bmatrix} \Omega_{N+1,N+1} & \Omega_{N+1,N+2} & ... & \Omega_{N+1,N+K} \\ ... & ... & ... & ... \\ \Omega_{N+K,N+1} & \Omega_{N+K,N+2} & ... & \Omega_{N+K,N+K} \end{bmatrix} + \gamma^{-1} I$$

According to the algorithm in [6], the matrix A_{N+K}^{-1} in Eq. (12) could be calculated from matrix A_N^{-1} and the inverse of a small $K \times K$ matrix, that is

$$A_{N+K}^{-1} = \begin{bmatrix} A_N^{-1} & 0 \\ 0 & 0 \end{bmatrix} + \begin{bmatrix} -A_N^{-1} Q \\ I \end{bmatrix} \left[S - Q^T A_N^{-1} Q \right]^{-1} \begin{bmatrix} -Q^T A_N^{-1} \\ I \end{bmatrix} \tag{13}$$

where 0 is a matrix whose elements are all zero. I is a unit matrix with K rows and K columns. In this way the calculation for the inverse of a large-scale matrix could be avoided.

3.2 Decrement Procedure

The number of support vectors will increase with the chunking increment procedure. To maintain the sparseness of support vectors, a decrement procedure is implemented after the chunking increment procedure. A support vector is omitted in this procedure. Meanwhile, a trained sample in the working set corresponding to the discarded support vector is also omitted. Form [7], $A_{l-1}^{-1} = \left(\hat{a}_{ij}\right)_{i,j \neq k}$ can be calculated from $A_l^{-1} = \left(\tilde{a}_{ij}\right)$, $A_l = \left(a_{ij}\right)$ and $A_{l-1} = \left(a_{ij}\right)_{i,j \neq k}$ in the decrement procedure, that is

$$\hat{a}_{ij} = \tilde{a}_{ij} - \frac{1}{a_{kk}} \tilde{a}_{ik} \tilde{a}_{kj}, \quad i, j \neq k \tag{14}$$

where A_{l-1} is a matrix obtained from A_l by omitting the kth row and the kth column.

3.3 Steps of CISVR Algorithm

Set the training sample set $T = \left\{ s_i \mid s_i = (x_i, y_i), x_i \in R^n, y_i \in R, i = 1,2,\cdots,l \right\}$. The form of the regression function is

$$f(x, \alpha, b) = \sum_{i \in \mathbf{W}} \alpha_i \psi(x, x_i) + b \equiv f(x)|_{\tilde{\mathbf{W}}} \tag{15}$$

where α and b are the regression parameters, W is named working set whose elements are the training samples selected to calculate the regression parameters, and $\tilde{W}$

is the regression parameters set which is decided by working set W. Set θ is the precision in training and testing, the precision in stop criterion is ε.

Steps of CISVR algorithm are as follows:

Initialization: set $W = \{(x_1, y_1),...,(x_N, y_N)\}$ and calculate A^{-1} analytically. Calculate $\tilde{W}$ and $f(x)|_{\tilde{W}}$ from Eqs. (8) and (9). Set k=0.

for $i = N+1,...,l$ **do**
adaptive learning
1. read a sample $s_i = (x_i, y_i)$
2. **if** $|f(x_i)|_{\tilde{W}} - y_i| > \theta$ **and** $s_i \notin W$ **then**
3. $W = W \bigcup \{s_i\}$, k=k+1
4. **end if**
5. **if** k=K **then**
6. calculate $\tilde{W}$ by chunking increment procedure
7. find the minimization support vector $|\alpha_{i^*}| = \min_{s_i \in W} \{|\alpha_i|\}$
8. $\hat{W} = W \setminus \{s_{i^*}\}$ // $\hat{W}$ is temporary working set
9. calculate $(\hat{W}\tilde{)}$ and temporary regression function $f(x)|_{\hat{W}}$
// $(\hat{W}\tilde{)}$ is the temporary regression parameters set corresponding to the
// temporary working set $\hat{W}$
10. read a sample s_{i+1}
11. **if** $|f(x_{i+1})|_{(\hat{W}\tilde{)}} - y_{i+1}| \le \theta$ **then**
12. $W = \hat{W}$ $\tilde{W} = (\hat{W}\tilde{)}$
13. **end if**
14. k=0
15. **end if**

end for
while the stop criterion is false **do**
for $i = 1,...,l$ **do**
adaptive learning
end for
end while

The stop criterion is related to the objective value. The formulation of the objective function is $J(w, e)|_W = \frac{1}{2}\|w\|^T_{w \in \tilde{W}} + \frac{\gamma}{2}\sum_{s_i \in W} e_i^2$, where $w = \sum_{s_i \in W} \alpha_i \varphi(x_i)$, $e_i = \frac{1}{\gamma}\alpha_i$. The meaning of the defined stop criterion is that the procedure ends

when the relative error of objective values in the two adjacent iterations is smaller than a given precision ε. In the decrement procedure, the minimization support vector is omitted because it has least effect on the performance of the regression function. The matrix A^{-1} in the current iteration is obtained from that in the previous iteration in both chunking increment and decrement procedure. In this way, it is possible on one hand to avoid calculating the inverse for a large-scale matrix and on the other hand to improve the learning speed of the procedure.

4 Numerical Experiments

In order to examine the efficiency of CISVR algorithm and compare CISVR with LSSVR algorithm, numerical experiments are performed using two kinds of data sets. One kind of data set is composed of the simply elementary functions which include $f(x)=\sin(x)$ and $f(x)=x^2$. These functions are used to test the regression ability for the known function. The other kind of data set is composed of Mackey-Glass (MG) system and simple function $f(x)=\sin c(x)$. The MG system is a blood cell regulation model established in 1977 by Mackey and Glass. It is a chaos system $\frac{dx}{dt}=\frac{a\cdot x(t-\tau)}{1+x^{10}(t-\tau)}-b\cdot x(t)$ described in [8], where $\tau=17$ $a=0.2$ $b=0.1$ $\Delta t=1$ $t\in(0,400)$. The embedded dimensions are $n=4,6,8$ respectively. The sample function is $f(x)=\begin{cases}1 & x=0\\ \dfrac{\sin(x)}{x} & x\neq 0\end{cases}$. An RBF kernel function $\psi(x_i,x_j)=\exp(-\left\|x_i-x_j\right\|^2/(2\sigma^2))$ is employed in these two algorithms. The parameters γ and σ are showed in Tab.1. The other parameters are as follows: $\theta=0.01,\varepsilon=0.01$. The comparison between LSSVR

Table 1. Parameters used in algorithm

		sin	square	sinc	MG system4	MG system6	MG system8
γ		50000	30000	5000	50000	50000	50000
σ	LSSVR	1.0	1.0	2.0	2.0	2.0	2.0
	CISVR	1.0	1.0	1.0	1.0	1.0	1.0

and CISVR are showed in Tab.2, where the third column is the number of support vectors, the forth column is the seconds for training, and the fifth and seventh columns are the regression accuracy for training and testing, respectively. The regression accuracy is a ratio that is the number of samples whose relative error is smaller than θ to the number of samples in the working set (testing set). The sixth and eighth columns are the mean square error for training and testing, respectively. It can be seen from Tab.2 that the learning speed of CISVR is much faster than LSSVR. Moreover,

Table 2. Comparison between CISVR algorithm and standard LSSVR

Dataset $l \times n$	Algorithm name	# of SVs	Train time (CPU s)	Accuracy (train%)	MSE (train)	Accuracy (test%)	MSE (test)
sin 3000×1	LSSVR	3000	1465.05	99.93	3.58e-009	99.87	4.19e-009
	CISVR	56	4.44	99.96	9.31e-007	99.70	1.04e-006
square 3000×1	LSSVR	3000	1463.44	99.97	2.62e-005	99.97	2.49e-005
	CISVR	132	10.65	97.76	3.29e-003	97.73	2.85e-003
sinc 3000×1	LSSVR	3000	1448.28	99.93	8.23e-011	99.80	1.14e-010
	CISVR	53	4.984	99.83	2.87e-008	99.56	3.18e-008
MG system4 6000×4	LSSVR	6000	11048.69	100	1.44e-008	100	1.49e-008
	CISVR	16	52.35	100	6.65e-007	100	6.91e-007
MG system6 6000×6	LSSVR	6000	12954.34	100	8.06e-009	100	8.48e-009
	CISVR	14	55.64	100	9.67e-007	100	1.02e-006
MG system8 6000×8	LSSVR	6000	13104.99	100	3.30e-009	100	3.28e-009
	CISVR	10	54.92	100	1.08e-006	100	1.13e-006

the number of support vectors is less than that obtained by LSSVR for the similar regression accuracy.

5 Discussion and Conclusion

In this paper we propose an adaptive and iterative support vector machine regression algorithm based on the chunking incremental learning and the least square support vector machine regression algorithm. The samples are added to the working set in batches. The support vectors are selected adaptively in the iteration and the sparseness of support vectors is maintained. Meanwhile, the inverse of matrix A in the previous iteration is used to calculate the regression parameters. Therefore, the proposed approach can avoid calculating the inverse of a large-scale matrix, and at the same time, substantially improve the learning speed compared to that of LSSVR for the similar regression accuracy.

Acknowledgment

The authors are grateful to the support of the National Natural Science Foundation of China (60433020), the science-technology development project for international collaboration of Jilin Province of China (20050705-2), the doctoral funds of the National Education Ministry of China (20030183060), Graduate Innovation Lab of Jilin University (503043), and “985” Project of Jilin University. The last two authors would like to thank the support of the EuMI School and the Erasmus Mundus programme of the European Commission.

References

1. Osuna E., Freund R., Girosi F.: An Improved Training Algorithm for Support Vector Machines. IEEE Workshop on Neural Networks and Signal Processing, Amelia Island, (1997) 276-285
2. Joachims T.: Making Large-scale Support Vector Machine Practical. In Advances in Kernel Methods-Support Vector Learning, Cambridge, Massachusetts: The MIT Press, (1999) 169-184.
3. Platt J.C.: Fast Training of Support Vector Machines Using Sequential Minimal Optimization. In Advances in Kernel Methods-Support Vector Learning, Cambridge, Massachusetts: The MIT Press, (1999) 185-208.
4. Suykens J.A.K., Vandewalle J.: Least Squares Support Vector Machine Classifiers. Neural Processing Letters, Vol.9 (1999) 293–300.
5. Suykens J.A.K., Lukas L., Wandewalle J.: Sparse Approximation Using Least Squares Support Vector Machines. In Proc. of the IEEE International Symposium on Circuits and Systems (ISCAS 2000), Geneva, Switzerland, (2000) 757-760.
6. Hao Z.F., Yu S., Yang X.W., Hu R., Zhao F., Liang Y.C.: Online LS-SVM Learning for Classification Problems Based on Incremental Chunk. Lecture Notes in Computer Science, Vol.3173 (2004) 558-564.
7. Cauwenberghs G., Poggio T.: Incremental and Decremental Support Vector Machine Learning. In Advances in Neural Information Processing Systems, Cambridge, MA: MIT Press, Vol.13 (2001) 426-433.
8. Flake G.W., Lawrence S.: Efficient SVM Regression Training with SMO. Machine Learning, Vol.46 (2002) 271–290.