Sparse Support Vector Regression Algorithm with Piecewise Loss Function

Gensheng Hu
Department of Computer ScienceShangqiu Normal
College Shangqiu, China
hugs2906@163.com

***Abstract*—Applying sparse algorithm can improve the prediction speed of support vector regression effectively. This paper solves sparse support vector regression with piecewise loss function based on iterative reweight method. By reducing support vector number, the length of regression function expansion and the prediction time of regression function for new samples are decreased. Comparing with sparse LS-SVR, the method has the advantages of suiting different noise data、steady prediction performance and good generalization performance.**

Keywords-support vector regression; sparse algorithm; Iterative Reweight Method

I. INTRODUCTION

General support vector regression (SVR) algorithm can be reduced to solve a convex quadratic programming. The solution obtained is global optimal, avoiding the local extremum problem of artificial neural network. There are some shortcomings for practical applications of support vector regression, such as the calculation burden of solving large-scale quadratic programming, the length of regression function expansion being in proportion to the number of training samples, storing large-scale kernel matrix, and so on. These problems are becoming bottleneck for support vector regression to solve large-scale problem.

Sparse support vector regression is an effective method for solving these problems. It uses few sample data to construct regression function. Thereby improves the prediction speed of support vector regression. The first sparse support vector regression algorithm was proposed by Vapnik[1]. Vapnik proposed ε -insensitive loss function of SVR. Suykens et al proposed sparse least squares support vector machine regression algorithm[2]. Engel et al constructed sparse solutions through kernel recursive least-squares algorithm[3,4]. Chen et al obtained sparse kernel density estimation by optimizing generalization performance of model[5]. Carley et al constructed sparse LS-SVR using LOO cross-validation[6]. Pelckmans et al obtained sparse representations of LS-SVR using fusion method[7]. Si et al proposed density weighted pruning method for sparse least squares support vector machines[8]. Zhao et al proposed an improved pruning algorithms for sparse least squares support vector regression machine[9].

This paper solves sparse SVR with piecewise loss function based on iterative reweight method[10]. By reducing support vector number, the length of regression function expansion and the prediction time of regression function for new samples are decreased. The sparse solutions have good generalization performance.

II. SPARSE LEAST SQUARES SUPPORT VECTOR REGRESSION ALGORITHM

Least squares support vector machines are introduced by Suykens et al[2,11]. Such regression problem can be stated as the following constrained optimization problem:

$$\min \frac{1}{2}\|W\|^2 + \frac{1}{2} r \sum_{i=1}^{N} e_i^2 \quad （1）$$

$$\text{s.t.} < W, \Phi(x_i) > +b = y_i - e_i, i = 1,...,N \quad （2）$$

Using Lagrange multiplier method, we get the following linear equations:

$$\begin{bmatrix} \mathbf{1}^T & 0 \\ K + r^{-1}I & \mathbf{1} \end{bmatrix} \begin{bmatrix} \alpha \\ b \end{bmatrix} = \begin{bmatrix} 0 \\ y \end{bmatrix} \quad （3）$$

Here $K = \{K_{i,j} = K(x_i, x_j)\}_{i,j=1}^{N}$ is kernel function. $\mathbf{1} = (1,...,1)^T$, $y = (y_1,..., y_N)^T$, $\alpha = (\alpha_1,...,\alpha_N)^T$ is Lagrange multipliers of constrains conditions.

According to the representation theorem, the optimization problem has the following solution:

$$f(x) = \sum_{i=1}^{N} \alpha_i K(x, x_i) + b \quad （4）$$

Least squares support vector regression (LS-SVR) uses the square of the error term instead of ε -insensitive loss function in the objective function. It publishes all deviated data, and therefore most of the coefficients of kernels are not zero. LS-SVR has lost the advantage of sparseness.

Suykens et al obtained the sparse approximation of LS-SVR by using the following algorithm[2]: training LS-SVR on the whole training set, resulting the weight coefficients vector α of predictive function, discarding part of the data associated with small α_i (5% of total data), re-training on remaining data, repeating this process until obtaining a satisfactory length of the kernel expansion.

Carley et al proposed an improved sparse least-squares support vector machines, Which trade-off constant is divided by the number of training data to solving the problem of generalization balance being destroyed[12].

978-1-4244-5265-1/10/$26.00 ©2010 IEEE

III. SPARSE SUPPORT VECTOR REGRESSION ALGORITHM BASED ON ITERATIVE REWEIGHT METHOD

According to SVM theory, SVR model can be expressed as:

$$\min \|W\|^2 + \frac{C}{N}\sum_{i=1}^{N} l(\xi_i)$$

$$\text{s.t.} \left|y_i - \Phi^T(x_i)W - b\right| \le \varepsilon + \xi_i \qquad （5）$$

$$\xi_i \ge 0 \quad i=1,...,N$$

where the loss function $l(\xi_i)$ is monotone undecreasing funcion of ξ_i, satisfying $l(0)=0$.

Using sparse vector $\Phi(x_i)$ in feature space, W can be expressed as: $W=\sum_{i=1}^{n}\beta_i\Phi(x_i)$, where $n<N$. Then the last model becomes:

$$\min \sum_{i,j=1}^{n} \beta_i\beta_j K_{ij} + \frac{C}{N}\sum_{i=1}^{N} l(\xi_i)$$

$$\text{s.t.} \left|y_i - \sum_{j=1}^{n}\beta_j K_{ij} - b\right| \le \varepsilon + \xi_i \qquad （6）$$

$$\xi_i \ge 0 \quad i=1,...,N$$

Define Lagrange multiplier:

$$L=\sum_{i,j=1}^{n}\beta_i\beta_j K_{ij} + \frac{C}{N}\sum_{i=1}^{N} l(\xi_i) - \sum_{i=1}^{N}\alpha_i[(\varepsilon+\xi_i)^2$$

$$-(y_i - \sum_{j=1}^{n}\beta_j K_{ij} - b)^2] - \sum_{i=1}^{N}\mu_i\xi_i \qquad （7）$$

According to KKT conditions:

$$\frac{\partial L}{\partial \beta_i} = 2\sum_{j=1}^{n}\beta_j K_{ij} - 2\sum_{l=1}^{N}\alpha_l(y_l - \sum_{j=1}^{n}\beta_j K_{lj} - b)^2 K_{li}$$

$$=0$$

$$（8）$$

$$\frac{\partial L}{\partial b} = -2\sum_{i=1}^{N}\alpha_i(y_i - \sum_{j=1}^{n}\beta_j K_{ij} - b) = 0 \qquad （9）$$

$$\frac{\partial L}{\partial \xi_i} = \frac{C}{N}a_i - \mu_i - 2\alpha_i(\varepsilon+\xi_i) = 0, i=1,...,N \qquad （10）$$

where $a_i = \frac{\partial L(\xi_i)}{\partial \xi_i}$.

$$\mu_i, \alpha_i \ge 0,\ i=1,...,N \qquad （11）$$

$$\alpha_i[(\varepsilon+\xi_i)^2 - (y_i - \sum_{j=1}^{n}\beta_j K_{ij} - b)^2] = 0,$$

$$i=1,...,N \qquad （12）$$

$$\mu_i\xi_i = 0, i=1,...,N \qquad （13）$$

$$\left|y_i - \sum_{j=1}^{n}\beta_j K_{ij} - b\right| - \varepsilon - \xi_i \le 0, i=1,...,N \qquad （14）$$

From（8）we get

$$\Omega\beta + \Phi b = c \qquad （15）$$

where

$\Omega=(\omega_{ij})_{i,j=1}^{n}$, $\omega_{ij}=K_{ij}+\sum_{l=1}^{N}\alpha_l K_{lj}K_{li}$, $\Phi=(\Phi_i)_{i=1}^{n}$, $\Phi_i=\sum_{j=1}^{N}\alpha_j K_{ij}$, $c=(c_i)_{i=1}^{n}$, $c_i=\sum_{j=1}^{N}\alpha_j y_j K_{ij}$, $\beta=(\beta_i)_{i=1}^{n}$.

From（9）we get

$$\Phi^T\beta + \sum_{i=1}^{N}\alpha_i b = \sum_{i=1}^{n}\alpha_i y_i \qquad （16）$$

From（15）、（16）we get

$$\begin{bmatrix} \Omega & \Phi \\ \Phi^T & \sum_{i=1}^{N}\alpha_i \end{bmatrix}\begin{bmatrix} \beta \\ b \end{bmatrix} = \begin{bmatrix} c \\ \sum_{i=1}^{N}\alpha_i y_i \end{bmatrix} \qquad （17）$$

For convenience of calculation, the equation (15) is written in matrix form

$$(K_{n\times n} + K_{N\times n}^T D_\alpha K_{N\times n})\beta + K_{N\times n}^T \alpha b = K_{N\times n}^T D_\alpha y \qquad （18）$$

where $K_{n\times n}$ and $K_{N\times n}$ are $n\times n$ and $N\times n$ matrix respectively. The elementary in their i row and j column is $K(x_i, x_j)$. $D_\alpha = diag(\alpha_1,...,\alpha_N)$, $\alpha=(\alpha_i)_{i=1}^{N}$, $y=(y_i)_{i=1}^{N}$.

Multiply the generalized inverse $(K_{N\times n}^T D_\alpha)^+$ of $K_{N\times n}^T D_\alpha$ at both side of（18）:

$$(D_{N\times n}^{+} + K_{N\times n})\beta + 1_N b = y \qquad （19）$$

where $D_{N\times n}^{+}$ is the generalized inverse of $D_{N\times n}$, $D_{N\times n} = \begin{pmatrix} D_{n\times n} \\ 0_{(N-n)\times n} \end{pmatrix}$, $D_{n\times n} = diag(\alpha_1,...,\alpha_n)$.

So（17）can be written as:

$$\begin{bmatrix} D_{N\times n}^{+} + K_{N\times n} & 1_N \\ K_{N\times n}^T \alpha & \sum_{i=1}^{N}\alpha_i \end{bmatrix}\begin{bmatrix} \beta \\ b \end{bmatrix} = \begin{bmatrix} y \\ \sum_{i=1}^{N}\alpha_i y_i \end{bmatrix} \qquad （20）$$

Form（10）to（14）, we get:

$$\alpha_i = \begin{cases} 0 & e_i \le \varepsilon \\ \frac{Ca_i}{2Ne_i} & e_i > \varepsilon \end{cases} \qquad （21）$$

where $e_i = \left| y_i - \sum_{j=1}^{n} \beta_j K_{ij} - b \right|$.

The value of α_i in（21）is associated with loss function. In this paper we use the following piecewise loss function:

$$l(e_i) = \begin{cases} 0 & e_i \in [0, \varepsilon] \\ \vdots & \vdots \\ (e_i - \varepsilon) \prod_{j=l}^{p} \frac{(e_i - \varepsilon)}{(\gamma_j - 1)\varepsilon} & e_i \in (\gamma_{l-1}\varepsilon, \gamma_l \varepsilon] \\ \vdots & \vdots \\ e_i - \varepsilon & e_i \in (\gamma_p \varepsilon, +\infty) \end{cases} \quad (22)$$

where $1 \le \gamma_1 \le \cdots \le \gamma_p < +\infty$.

In（22）, if $e_i < \gamma_{l-1}\varepsilon$, then $(e_i - \varepsilon) \prod_{j=l-1}^{p} \frac{(e_i - \varepsilon)}{(\gamma_j - 1)\varepsilon} < (e_i - \varepsilon) \prod_{j=l}^{p} \frac{(e_i - \varepsilon)}{(\gamma_j - 1)\varepsilon}$. Practical application shows that[13], loss function with third-order and above being suitable for sub-Gaussian noise, second-order loss function being suitable for Gaussian noise, and linear loss function being suitable for super-Gaussian noise.

In Bayesian model selection method for SVR, noise distribution is often assumed to have the following exponential form: $P(e_i) \propto \exp[-Cl(\xi_i)]$, here $\xi_i = \max(0, |y_i - f(x_i)| - \varepsilon)$. If we use equation (22) for $\gamma_{p-1} = 1$, when the error term falls in the quadratic loss function domain $e_i \in (\varepsilon, \gamma_p \varepsilon]$, then $P(e_i) \propto \exp[-Cl(\xi_i)] \propto \exp(-C \frac{e_i^2}{(\gamma - 1)\varepsilon})$. If the noise distribution is Gaussian with zero mean and variance r^2, then $\frac{(\gamma - 1)\varepsilon}{C} \quad 2r^2$, namely $\gamma \quad \frac{2Cr^2}{\varepsilon} + 1$. From this, it can be seen that the appropriate γ value is closely related to the variance of noise distribution.

Form（22）, we get the following value in（10）:

$$a_i = \begin{cases} 0 & e_i \in [0, \varepsilon] \\ \vdots & \vdots \\ (p - l + 2) \prod_{j=l}^{p} \frac{(e_i - \varepsilon)}{(\gamma_j - 1)\varepsilon} & e_i \in (\gamma_{l-1}\varepsilon, \gamma_l \varepsilon] \\ \vdots & \vdots \\ 1 & e_i \in (\gamma_p \varepsilon, +\infty) \end{cases} \quad (23)$$

In summary, the procedures for sparse SVR with piecewise loss function based on iterative reweight method are:

1) Input training data (x_i, y_i), $i = 1, \ldots, n$ and parameters C, ε, γ_i. Initialize β, e_i.

2) For $i = 1, \ldots, n$, compute a_i according to（23）, and α_i according to（21）.

3) Compute β, b according to（20）, and $f(x_i) = \sum^{n} \beta_j K(x_i, x_j) + b$, $e_i = |y_i - f(x_i)|$. If all $e_i \le \varepsilon$, $i = 1, \ldots, n$, then go to step 4), else return to step 2).

4) Discard data associated with small β_i, and return to step 1) until remaining number of data meeting the requirements.

IV. EXPERIMENTS

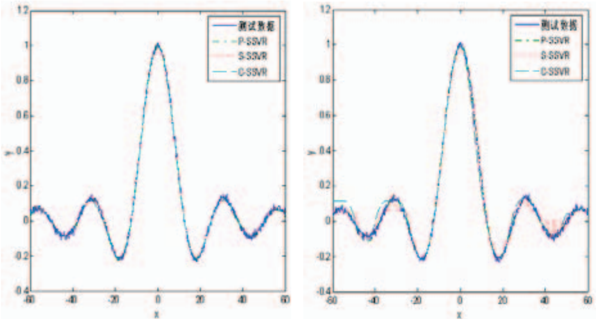
This section compares the performances of three sparse support vector regression including: Suykens's sparse least square support vector regression（S-SSVR）, Carley's improvement algorithm for sparse least square support vector regression (C-SSVR), and sparse support vector regression with piecewise loss function（P-SSVR）.

In the experiments, Gaussian kernel $K(x_i, x_j) = e^{-\frac{\|x_i - x_j\|^2}{2\sigma^2}}$ is adopted. Regularization parameter C、kernel parameter σ ,and ε are given using method of grid search.

Given data set $\{(x_i, y_i), i = 1, \ldots, 1310\}$, where x_i being random coming from interval $[-60, 60]$, $y_i = \sin(0.25x_i)/0.25x_i + n_i$, n_i being normally distributed random noise with mean 0 and standard variance of 0.01, We randomly select 210 data as training data set,100 data as validation set, and remaining 1000 data as test data set.

Based on these data set , we carry out sparse support vector regression by using P-SSVR,S-SSVR and C-SSVR respectively. The results are shown in Figure 1, where (a) is the results of reducing support vector number from 210 to 150, and (b) is the results of reducing support vector number from 210 to 50.

As can be seen from Figure 1, using sparse support vector regression algorithm to reduce the number of support vector still has high prediction accuracy. Figure 2 shows the mean absolute error $MAE = \frac{1}{n} \sum_{i=1}^{n} |\hat{y}_i - y_i|$ of the three kinds of sparse support vector regression algorithm for different support vector number, where $\hat{y}_i$ is the estimated value of y_i.



(a) (b)

Figure 1. Compare of prediction results for three sparse SVR

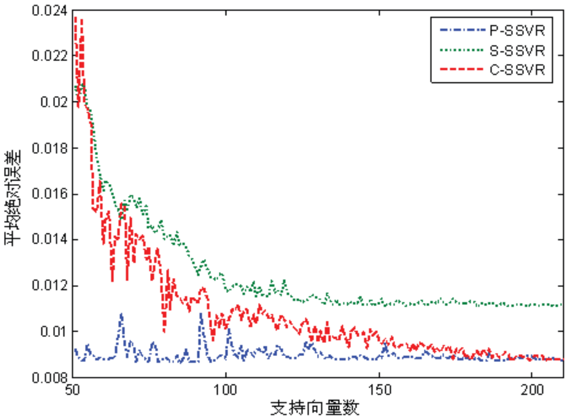


Figure 2. Compare of mean absolute error for different support vector number(Gaussian noise)

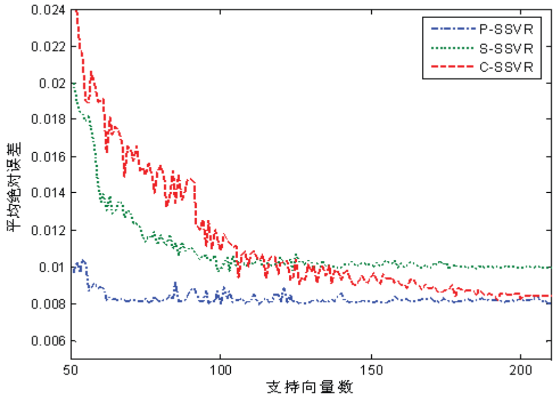


Figure 3. Compare of mean absolute error for different support vector number(exponential distribution noise)

As can be seen from Figure 2, the MAE is low for these three sparse support vector regression algorithm for more than 100 support vector number, but below 100 or so support vector number, the MAE for S-SSVR and C - SSVR algorithm increases rapidly, while the P-SSVR algorithm increases slowly. So in general, the performance of P-SSVR is superior to S-SSVR and C-SSVR algorithm.

The following experiment revises Gaussian noise to obey exponential distribution with parameter 0.01. Figure 3 shows the MAE of the three kinds of sparse support vector regression algorithm for different support vector number given exponential distribution noise.

As can be seen from Figure 3, under the exponential distribution noise condition, C-SSVR isn't superior to S-SSVR, but P-SSVR has been relatively stable whose mean absolute error is lowest. So for non-Gaussian noise case, P-SSVR algorithm is superior to C-SSVR and S-SSVR .

V. CONCLUSION

Reducing the number of support vector can reduce the size of the kernel matrix, thereby reduce the prediction time for new sample. Using this method we can get sparse support vector regression algorithm.

In this paper, we discuss sparse support vector regression with piecewise loss function by using iterative reweight method. Comparing with sparse LS-SVR method, the method proposed in this paper has advantages of suitable dealing with different noise data, stable prediction performance and strong generalization performance.

REFERENCES

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

[2] J. A. K. Suykens, J. Brabanter, L. Lukas, et al, "Weighted least squares support vector machines: Robustness and sparse approximation", Neurocomputing, vol.48, No.(1-4), 2002, pp. 85-105.

[3] Y. Engel, S. Mannor, R. Meir, "The kernel recursive least-squares algorithm", IEEE Transactions on Signal Processing, vol.52, No.8, 2004, pp 2275 – 2285.

[4] T. Downs, K. Gates, A. Masters, "Exact Simplification of Support Vector Solutions", Journal of Machine Learning Research, vol.2, No.2, 2002, pp 293-297.

[5] S. Chen, X. Hong, C. J. Harris, "Sparse kernel density construction using orthogonal forward regression with leave-one-out test score and local regularization", IEEE Transactions on Systems, Man, and Cybernetics, Part B. Cybernetics, vol.34, No.4, 2004, pp 1708-1717.

[6] G. C. Cawley, N. L. C. Talbot, "Fast exact leave-one-out cross-validation of sparse least-squares support vector machines", Neural Networks, vol.17, No.10, 2004, pp 1467-1475.

[7] K. Pelckmans, J. A. K. Suykens, B. D. Moor, "Building sparse representations and structure determination on LS-SVM substrates", Neurocomputing, vol.64, No.1-4, 2005, pp 137-159.

[8] G. Si, H. Cao, Y. Zhang,et al, "Density Weighted Pruning Method for Sparse Least Squares Support Vector Machines", Journal of Xi an Jiaotong University, vol.43, No.10, 2009, pp 11-15.

[9] Y. Zhao, J. Sun, "Improved pruning algorithms for sparse least squares support vector regression machine", Systems Engineering-Theory & Practice, vol.29, No.6, 2009, pp 166-171.

[10] F. Perez-cruz, A. Navia-vazquez, P. L. Alarcon-diana, et al, An IRWLS procedure for SVR, Proc of the European Signal Processing Conf. Tampere, Finland. 2000.

[11] J. A. K. Suykens, J. Vandewalle, "Least Squares Support Vector Machine Classifiers", Neural Processing Letters, vol.9, No.3, 1999, pp 293- 300

[12] G. C. Cawley, N. L. C. Talbot, "Improved sparse least-squares support vector machines", Neurocomputing, vol.48, 2002, pp 1025-1031

[13] J. L. Rojo-Alvarez, M. Martinez-Ramon, M. Prado-Cumplido, et al, "Support Vector Method for Robust ARMA System Identification", IEEE Transactions on Signal Processing, vol.52, No.1, 2004, pp 155-164