Training of Support Vector Regressors Based on the Steepest Ascent Method

Hirokawa, Youichi

Abe, Shigeo

TRAINING OF SUPPORT VECTOR REGRESSORS BASED ON THE STEEPEST ASCENT METHOD

Youichi Hirokawa, Shigeo Abe

Graduate School of Science and Technology, Kobe University, Kobe, Japan
(E-mail:abe@eedept.kobe-u.ac.jp)

ABSTRACT

In this paper, we propose a new method for training support vector regressors. In our method, we partition all the variables into two sets: a working set that consists of more than two variables and a set in which variables are fixed. Then we optimize the variables in the working set using the steepest ascent method. If the Hessian matrix associated with the working set is not positive definite, we calculate corrections only for the independent variable in the working set.

We test our method by two benchmark data sets, and show that by increasing the working set size, we can speed up training of support vector regressors.

1. INTRODUCTION

Support Vector Machines (SVMs) can construct classifiers and function approximators with high generalization ability [1,2]. But training of SVMs needs to solve the quadratic optimization problems, which have variables as many as the number of training data for pattern classification, and twice as many as that for function approximation. Since large matrix computations are needed, training of SVMs is slow and requires a large memory size.

To resolve these problems, the decomposition technique [3] is usually used for training for a large data set. The decomposition method partitions the variables into two sets: a working set and a fixed set. Then it optimizes the variables in the working set. Updating the working set, the above procedure is repeated until the entire optimal solution is obtained. Sequential Minimal Optimization (SMO) proposed by Platt [4] selects two variables in the working set. Since an optimization problem with two variables can be solved analytically, SMO does not require a QP solver.

In this paper, we extend SMO for Support Vector Regressors (SVRs), increasing the working set size from two and optimizing the variables in the working set by the steepest ascent method. Calculations of corrections of variables in the working set include inversion of the associated Hessian matrix. But since the Hessian matrix is not guaranteed to be positive definite, we calculate the corrections only for the linearly independent variables in the working set.

In the following, in Section 2, we explain the overview of support vector machines for function approximation. And in Section 3, we discuss the algorithm of our method. Finally in Section 4, we evaluate our method by time series data generated from the Mackey-Glass differential equation and water purification plant data.

2. SUPPORT VECTOR REGRESSORS

Let $\mathbf{x}_1, \ldots, \mathbf{x}_M$ be the inputs of the training data and let $y_1, \ldots, y_M$be the outputs. In SVRs, the original input space is mapped into the high dimensional space called feature space by a nonlinear mapping $\mathbf{x} \rightarrow \mathbf{g}(\mathbf{x})$. And in the feature space, we construct linear function

$$f(\mathbf{x}) = \mathbf{w}^t \mathbf{g}(\mathbf{x}) + b, \tag{1}$$

where $\mathbf{w}$ is a coefficient vector, and b is a threshold.

The approximation function is obtained by solving the following optimization problem:

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

$$\begin{aligned} \text{subject to} \quad & y_i - f(\mathbf{x}_i) \leq \varepsilon + \xi_i, \\ & f(\mathbf{x}_i) - y_i \leq \varepsilon + \xi_{M+i}, \\ & \xi_i, \xi_{i+M} \geq 0, \end{aligned} \tag{3}$$

where ε is a positive parameter to allow approximation errors smaller than ε, and ξ_i and ξ_{M+i} are slack variables for the data $\mathbf{x}_i$ whose errors exceed ε. Since it is difficult to solve (2) and (3), we usually solve the following dual optimization problem whose number of variables is twice the number of training data:

$$\begin{aligned}
\text{maximize} \quad & -\frac{1}{2}\sum_{i,j=1}^{M}(\alpha_i-\alpha_{M+i})(\alpha_j-\alpha_{M+j}) \\
& \times K(\mathbf{x}_i,\mathbf{x}_j)+\sum_{i=1}^{M}y_i(\alpha_i-\alpha_{M+i}) \\
& -\varepsilon\sum_{i=1}^{M}(\alpha_i+\alpha_{M+i}) \qquad (4) \\
\text{subject to} \quad & \sum_{i=1}^{M}(\alpha_i-\alpha_{M+i})=0, \qquad (5) \\
& 0 \le \alpha_i \le C, \qquad (6)
\end{aligned}$$

where $\alpha_i (i = 1, \ldots, 2M)$ are Lagrange multipliers, and $K(\mathbf{x}_i, \mathbf{x}_j)$ denotes a kernel function which is equivalent to the dot product in the feature space, namely

$$K(\mathbf{x}_i,\mathbf{x}_j)=\mathbf{g}(\mathbf{x}_i)^t\mathbf{g}(\mathbf{x}_j). \qquad (7)$$

The optimal approximation function can be obtained by solving (4) to (6). The dual form of the approximation function is given by Lagrange multipliers, namely

$$f(\mathbf{x}_i)=\sum_{i=1}^{M}(\alpha_i-\alpha_{M+i})K(\mathbf{x}_i,\mathbf{x}_j)+b. \qquad (8)$$

3. TRAINING BY STEEPEST ASCENT METHOD

SMO solves 2-variable subproblems without using a QP solver. Our method solves subproblems with more than 2 variables by the steepest ascent method without using a QP solver.

Now we define a set V as the index set of variables that are candidates of support vectors and a set B as the index set of variables in the working set.

The rough flow of the training procedure of the SVR is as follows:

1. Add all indexes to V.
2. Select n indices from V randomly and set them to B. Selected indices are removed from V.
3. Calculate corrections of the variables in the working set by the steepest ascent method so that the objective function is maximized.
4. If a convergence condition is satisfied, finish training. Otherwise, if V is empty, add new candidates which violate the Karush-Kuhn-Tucker condition. Return to Step 2.

3.1. Subproblem Optimization

In this subsection we explain Step 3 more in detail.

Let α_B be a vector whose elements are α_i $(i \in B)$. From (5), $\alpha_s \in \alpha_B$ is expressed as follows:

$$\alpha_s = -\sum_{i=1,i\neq s}^{M}\alpha_i+\sum_{i=M+1}^{2M}\alpha_i \quad \text{if} \quad s \le M, \qquad (9)$$

$$\alpha_s = \sum_{i=1}^{M}\alpha_i-\sum_{i=M+1,i\neq s}^{2M}\alpha_i \quad \text{if} \quad s > M. \qquad (10)$$

Substituting (9) or (10) into the objective function, we eliminate constraint (5) from the dual problem. Here B' is defined as the set in which s is removed from B, namely $B' = B - \{s\}$.

Since the objective function is quadratic, the change of the objective function, $\Delta Q(\alpha_{B'})$, for the change of variables, $\Delta\alpha_{B'}$, is given by

$$\begin{aligned}
\Delta Q(\alpha_{B'}) \;&=\; \frac{\partial Q(\alpha_{B'})}{\partial \alpha_{B'}}\Delta\alpha_{B'} \\
&\quad +\frac{1}{2}\Delta\alpha_{B'}^t\frac{\partial^2 Q(\alpha_{B'})}{\partial \alpha_{B'}^2}\Delta\alpha_{B'}. \qquad (11)
\end{aligned}$$

If $\partial^2 Q(\alpha_{B'})/\partial\alpha_{B'}^2$ is positive definite, we can calculate corrections by the following formula so that $\Delta Q(\alpha_{B'})$ is maximized:

$$\Delta\alpha_{B'} = -\left(\frac{\partial^2 Q(\alpha_{B'})}{\partial \alpha_{B'}^2}\right)^{-1}\frac{\partial Q(\alpha_{B'})}{\partial \alpha_{B'}}. \qquad (12)$$

From (4), (9), and (10), the element of $\partial Q(\alpha_{B'})/\partial\alpha_{B'}$ is

$$\begin{aligned}
\frac{\partial Q(\alpha_{B'})}{\partial \alpha_i} \;&=\; p_i\{y_i-y_s-\varepsilon(p_i+q) \\
&\quad -\sum_{j=1}^{M}(\alpha_j-\alpha_{M+j})(K_{ij}-K_{sj})\} \qquad (13)
\end{aligned}$$

and the element of $\partial^2 Q(\alpha_{B'})/\partial\alpha_{B'}^2$ is

$$\frac{\partial^2 Q(\alpha_{B'})}{\partial\alpha_i\partial\alpha_j} = -p_ip_j(K_{ij}+K_{ss}-K_{is}-K_{js}), \qquad (14)$$

where p_i, p_j, and q are defined as

$$p_i = \begin{cases} +1 & \text{for} \quad i \le M, \\ -1 & \text{for} \quad i > M, \end{cases} \qquad (15)$$

$$q = \begin{cases} -1 & \text{for} \quad s \le M, \\ +1 & \text{for} \quad s > M. \end{cases} \qquad (16)$$

The solution given by (12) is obtained by solving a set of simultaneous equations.

To speed up solving (12), we decompose $\partial^2 Q(\alpha_{B'})/\partial\alpha_{B'}^2$ into the upper and lower triangular matrices by the Cholesky decomposition. If the Hessian matrix is not positive definite, the Cholesky decomposition stops because the input of the root becomes nonpositive. When this happens, we solve (12) only for the variables that are decomposed so far. And we delete the associated variable and the variables that are not decomposed from the working set.

Now, we can calculate the correction of α_s. In the case of $s \le M$,

$$\Delta\alpha_s = -\sum_{i\in B', i\le M} \Delta\alpha_i + \sum_{i\in B', i>M} \Delta\alpha_i \qquad (17)$$

or in the case of $s > M$,

$$\Delta\alpha_s = \sum_{i\in B', i\le M} \Delta\alpha_i - \sum_{i\in B', i>M} \Delta\alpha_i. \qquad (18)$$

Although the solution calculated by the above-described procedure satisfies (5), it may not satisfy (6). Namely, when there are variables that cannot be corrected:

$$\begin{aligned} \Delta\alpha_i < 0 \quad &\text{when} \quad \alpha_i = 0, \\ \Delta\alpha_i > 0 \quad &\text{when} \quad \alpha_i = C, \end{aligned}$$

we remove these variables from the working set and solve again (12) for the reduced working set. This does not require much time because recalculations of the kernel functions for the reduced working set are not necessary by caching them.

Now, suppose the solution that can make some corrections for all the variables in the working set are obtained. Then, the corrections are adjusted so that all the updated variables go into the range $[0, C]$. Let $\Delta\alpha'_i$ be the allowable corrections if each variable is corrected separately. Then,

$$\Delta\alpha'_i = \begin{cases} C - \alpha_i^{old} & \text{if} \quad \alpha_i^{old} + \Delta\alpha_i > C, \\ -\alpha_i^{old} & \text{if} \quad \alpha_i^{old} + \Delta\alpha_i < 0, \\ \Delta\alpha_i & \text{otherwise.} \end{cases} \qquad (19)$$

Using $\Delta\alpha'_i$ we calculate the minimum ratio of corrections:

$$r = \min_{i\in B} \frac{\Delta\alpha'_i}{\Delta\alpha_i}. \qquad (20)$$

Then the variables are updated by

$$\alpha_i^{new} = \alpha_i^{old} + r\Delta\alpha_i. \qquad (21)$$

Clearly the updated variables satisfy (6).

3.2. Convergence Condition and Working Set Selection

After the update of the variables in the working set, we check whether training should be finished or not. Namely, in Step 4, only when V becomes empty, the following KKT conditions are checked.

1. For α_i $(i = 1, \ldots, M)$

$$\begin{aligned} y_i - f(\mathbf{x}) > \varepsilon \quad &\text{and} \quad \alpha_i < C, \\ y_i - f(\mathbf{x}) < \varepsilon \quad &\text{and} \quad \alpha_i \ne 0. \end{aligned} \qquad (22)$$

2. For α_{i+M} $(i = 1, \ldots, M)$

$$\begin{aligned} f(\mathbf{x}) - y_i > \varepsilon \quad &\text{and} \quad \alpha_{i+M} < C, \\ f(\mathbf{x}) - y_i < \varepsilon \quad &\text{and} \quad \alpha_{i+M} \ne 0. \end{aligned} \qquad (23)$$

The indices of the variables that violate the conditions are added to V.

If there are no data that violate the KKT conditions, the optimization problem is converged. Thus we finish training. To accelerate training, if an increase of the objective function becomes very small, we consider the solution is sufficiently near the optimal solution. Training is finished when the following inequality is satisfied for N consecutive iterations:

$$\frac{Q_n - Q_{n-1}}{Q_{n-1}} < \delta, \qquad (24)$$

where δ is a small value.

The threshold b is computed at every iteration. If there are Lagrange multipliers inside the boundaries ($0 < \alpha_i < C$), we calculate b_i by

$$b_i = y_i - \sum_{i=1}^{M}(\alpha_i - \alpha_{i+M}) - \varepsilon \quad \text{for} \quad 0 < \alpha_i < C, \qquad (25)$$

$$b_i = y_i - \sum_{i=1}^{M}(\alpha_i - \alpha_{i+M}) + \varepsilon \quad \text{for} \quad 0 < \alpha_{i+M} < C, \qquad (26)$$

and take the average of b_i and set it to b.

4. PERFORMANCE EVALUATION

We evaluated the performance of our method using Mackey-Glass time series data and stationary water purification plant data [5]. We investigated dependence of the calculation time on the working set size. We used the RBF kernel function with $\gamma = 10$:

$$K(\mathbf{x}_i, \mathbf{x}_j) = \exp(-\gamma\|\mathbf{x}_i - \mathbf{x}_j\|^2). \qquad (27)$$

We measured the CPU time using a Pentium III (1GHz) computer operating under Linux. In all simulations, training was finished when (24) was satisfied 10 consecutive times.

4.1. Mackey-Glass Time Series Data

Mackey-Glass time series data includes 500 training data and 500 test data with a 4-demensional input. We used $\varepsilon = 0.01$ and $C = 10000$.

Table 1 shows the performance comparison for several working set sizes. “Size” represents the working set size. Approximation performance was measured by the normalized root-mean-square error (NRMSE). The rightmost column of the table shows the average number of variables in the working set that were updated simultaneously, namely, the average size of the Hessian matrices that were positive definite.

As the working set size was increased, the average number of updated variables was increased. This led to the smaller number of training epochs and shorter training time. Against size 2 the maximum speedup of 4.9 was obtained at size 30. But for size 50, the training time was increased.

Table 1: Performance for Mackey-Glass

	Time	NRMSE			Ave. updated
Size	[sec]	Train.	Test	Epochs	variables
2	66.0	0.027	0.027	52291	1.87
3	45.3	0.027	0.027	21826	2.69
5	36.6	0.028	0.027	9728	4.69
10	19.9	0.027	0.027	2452	8.64
20	17.4	0.028	0.027	843	15.48
30	13.6	0.027	0.027	436	22.02
50	23.7	0.027	0.027	440	31.32

4.2. Water Purification Plant Data

The output of water purification plant data is the amount of coagulant that was determined by an operator. The data set includes 241 training data and 237 test data with a 10-demensional input. We used $\varepsilon = 1$ and $C = 1000$.

Table 2 shows the performance comparison for several working set sizes. Training time was shortened as the size increases. Against size 2, the maximum speedup of 1.9 was obtained for size 30.

Table 3 shows the average number of updated variables simultaneously. Comparing Tables 1 and 2, the average numbers of updated variables for the working set size are almost the same.

Table 2: Performance for water purification plant data (stationary state)

	Time	Error (train.)		Error (test)		
Size	[sec]	Ave.	Max	Ave.	Max	Epochs
2	13.6	0.69	1.41	1.03	6.92	16439
3	14.4	0.69	1.44	1.04	6.92	10613
5	11.0	0.69	1.05	1.03	6.92	4539
10	8.8	0.69	1.54	1.04	6.98	1741
15	9.5	0.68	1.11	1.02	7.09	1106
20	8.8	0.68	1.02	1.03	7.01	780
30	7.1	0.68	1.03	1.03	6.95	426

Table 3: Average number of updated variables for water purification plant data

	Ave. updated	
Size	variables	Epochs
2	1.92	16439
3	2.91	10613
5	4.78	4539
10	9.10	1741
15	13.01	1106
20	16.89	780
30	25.32	426

5. CONCLUSIONS

In this paper, we proposed the steepest ascent method of training support vector regressors, which is an extension of the sequential minimal optimization technique. The variables are partitioned into a working set, which includes more than one variable and a fixed set. The variables in the working set are optimized by the steepest ascent method. By the simulations using two benchmark data sets, we showed the speedup of training by increasing the working set size.

6. REFERENCES

[1] V. Cherkassy and F. Mulier, *Learning from Data: Concepts, Theory, and Method*, John Wiley & Sons, 1998.

[2] V. N. Vapnik, *Statistical Learning Theory*, John Wiley & Sons, 1998.

[3] E. Osuna, R. Freund, and F. Girosi, An improved training algorithm for support vector machines, *Proc. 1997 IEEE Signal Processing Society Workshop on Neural Networks for Signal Processing VII*, pp. 276–285, 1997.

[4] J. C. Platt, “Fast Training of Support Vector Machines Using Sequential Minimal Optimization,” in *Advances in Kernel Methods*, pp. 185–208, The MIT Press, 1999.

[5] S. Abe, *Pattern Classification: Neuro-Fuzzy Methods and Their Comparison*, Springer-Verlag, 2001.