Letters

Reduced twin support vector regression

Mittul Singh a, Jivitej Chadha b, Puneet Ahuja b, Jayadeva c,*, Suresh Chandra a

a Department of Mathematics, IIT Delhi, Hauz Khas, New Delhi 110016, India
b Department of Computer Science & Engineering, IIT Delhi, Hauz Khas, New Delhi 110016, India
c Department of Electrical Engineering, IIT Delhi, Hauz Khas, New Delhi 110016, India

Article info
Article history:
Received 19 August 2010
Received in revised form
12 October 2010
Accepted 4 November 2010
Communicated by R.W. Newcomb
Available online 15 December 2010

Keywords:
Support vector machines
Regression
Function approximation
SVM
Kernels
Rectangular kernels

Abstract
We propose the reduced twin support vector regressor (RTSVR) that uses the notion of rectangular kernels to obtain significant improvements in execution time over the twin support vector regressor (TSVR), thus facilitating its application to larger sized datasets.

1. Introduction

The last decade has witnessed the evolution of support vector machines (SVMs) as a powerful paradigm for pattern classification and regression [1–4]. SVMs emerged from research in statistical learning theory on how to regulate the trade-off between structural complexity and empirical risk. Given a binary classification task, the classical SVM solves a constrained quadratic programming problem (QPP) to determine a separating hyperplane. The size of the QPP is the number of training patterns. The twin support vector machine (TWSVM) [5] finds two hyperplanes, one for each class, by solving two smaller sized QPPs. On an average, it is about four times faster than the standard SVM.

Recently, Peng [6] extended the idea of twin SVMs to regression, and proposed the twin support regressor. Though the TSVR is in the spirit of the twin SVM, it requires all the data points in the twin pair. This becomes a limitation when handling large data sets. In this paper, we propose the idea of Reduced TSVR, that provides significant computational savings over the TSVR and becomes attractive for large data sets.

The TSVR requires one matrix inversion and three matrix multiplications, each of which has a complexity of $O(n^3)$. We use the notion of rectangular kernels introduced by Lee and Mangasarian [7] to modify the TSVR formulation and obtain the Reduced TSVR.

This reduces the time required for the matrix multiplications from $O(2n^3)$ to $O(2nn_1^2 + n^2n_1)$, and the matrix inversion time from $O(n^3)$ to $O(n_1^3)$; here, $n$ is the number of training patterns and $n_1$ is a randomly chosen subset of patterns, typically of the size of 10% of $n$.

The remainder of the paper has been organized as follows: Section 2 gives the formulation of the TWSVM and then gives the kernel formalization of the non-linear TSVR. This leads to the formulation of the reduced twin support vector regression. Section 3 contains experimental results.

2. Reduced twin support regression

2.1. Twin support vector machines

The twin SVM finds two nonparallel hyperplanes around which the data points of the corresponding class get clustered by solving two independent optimization problems. In each of these twin QPPs, constraints involve patterns from only one class. Consequently, the twin SVM performs well even when the datasets are unbalanced, i.e. when one class has many more patterns than the other. In the classical SVM, the error term contains contributions from all patterns, and therefore the larger class tends to dominate.

The twin SVM classifier is obtained by solving the following pair of QPPs:

$$
\min_{w^{(1)},b^{(1)},q_1} \frac{1}{2}(Aw^{(1)} + e_1b^{(1)})^T(Aw^{(1)} + e_1b^{(1)}) + c_1e_1^Tq_1
$$

0925-2312/$ - see front matter © 2011 Elsevier B.V. All rights reserved.
doi:10.1016/j.neucom.2010.11.003
subject to $-(bw^{(1)}+e_2b^{(1)})+q_1 \geq e_2q_1 \geq 0$, \hspace{1cm} (1) \\
\min \frac{1}{w^{(1)}b^{(1)}x} (bw^{(2)}+e_2b^{(2)})^T(bw^{(2)}+e_2b^{(2)})+c_2e^Tq_2 \\
subject to $-(aw^{(2)}+e_1b^{(2)})+q_2 \geq e_1q_2 \geq 0$. \hspace{1cm} (2)

where $c_1, c_2 \geq 0$ are parameters; and $e_1$ and $e_2$ are vectors of ones of dimension $n_1 \times 1$ and $n_2 \times 1$, respectively. The training patterns belonging to one class (size $n_1$) are denoted by the symbol $A$ (size $n_1 \times m$ where $m$ is the number of features) and the other class (size $n_2$) by $B$ (size $n_2 \times m$); $q_1, q_2$ are non-negative slack variables. The hyperplanes passing through the patterns $A$ and $B$ are given by $w^{(1)} \cdot x + b^{(1)} = 0$ and $w^{(2)} \cdot x + b^{(2)} = 0$, respectively.

2.2. Twin support vector regression

Peng proposed the twin support vector regression (TSVR) [6] on the lines of the twin SVM. The non-linear TSVR regressor is obtained by solving the following pair of QPPs:

$$ \min \frac{1}{w^{(1)}b^{(1)}x} (Y-ec_1-(K(A,A^T)w^{(1)})^T(Y-ec_1-(K(A,A^T)w^{(1)})+e^{(1)})+c_1e^Tq_1 $$

subject to $Y-(K(A,A^T)w^{(1)})+q_1 \geq ec_1$, $q_1 \geq 0$. \hspace{1cm} (3)

$$ \min \frac{1}{w^{(2)}b^{(2)}x} (Y+e^{(2)}-(K(A,A^T)w^{(2)})^T(Y+e^{(2)}-(K(A,A^T)w^{(2)})+e^{(2)})) + c_2e^Tq_2 $$

subject to $(K(A,A^T)w^{(2)})+q_2 \geq ec_2$, $q_2 \geq 0$. \hspace{1cm} (4)

where $A$ is the set of training patterns of size $n$ with $m$ features, $c_1, c_2 \geq 0$ are parameters, and $e$ is a vector of ones of size $n \times 1$. And $K(A, A^T)$ is the kernel used by the non-linear paradigm.

The above formulation provides us with two functions $f_1(x) = (w^{(1)})^T K(A, x) + b^{(1)}$ and $f_2(x) = (w^{(2)})^T K(A, x) + b^{(2)}$. Each function approximates the data points within a tolerance of $\varepsilon$. The TSVR gives rise to two smaller QPPs of half the size, and is thus approximately four times faster than the classical SVM.

The Karush–Kuhn–Tucker (KKT) necessary and sufficient optimality conditions [8,9] for the problem (3) are given by

$$-K(A,A^T)^T(Y-K(A,A^T)w^{(1)})-e^{(1)}-e^{(1)}+e^{(1)}x = 0, \hspace{1cm} (5)$$

$$e^{(1)}(Y-K(A,A^T)w^{(1)})-e^{(1)}+e^{(1)}x = 0, \hspace{1cm} (6)$$

$$c_1e-x-\beta = 0, \hspace{1cm} (7)$$

$$Y-(K(A,A^T)w^{(1)}+e^{(1)})+q \geq ec_1, \hspace{1cm} q \geq 0, \hspace{1cm} (8)$$

$$\lambda^T(Y-(K(A,A^T)w^{(1)}+e^{(1)})+q-ec_1) = 0, \hspace{1cm} x \geq 0, \hspace{1cm} (9)$$

$$\beta^Tq = 0, \hspace{1cm} \beta \geq 0. \hspace{1cm} (10)$$

Simplifying, we obtain the TSVR dual problems as

$$\max \frac{1}{2} \gamma H^T H^{-1} \gamma + \gamma^T H^T H^{-1} H^T x - x^T \gamma \hspace{1cm} s.t. \hspace{1cm} 0 \leq x \leq c_1, \hspace{1cm} (11)$$

$$\max \frac{1}{2} \gamma^T H^T H^{-1} \gamma - \gamma^T H^T H^{-1} H^T \gamma \hspace{1cm} s.t. \hspace{1cm} 0 \leq \gamma \leq c_2. \hspace{1cm} (12)$$

where $H = [K(A,A^T)] \ e_1$, $e = Y-ec_1$, $h = Y+ec_2$ and $x$ and $\gamma$ are the Lagrangian multipliers corresponding to the twin pair (11) and (12) respectively. Solving these gives us the upper and lower bound functions. These can be used as follows to obtain the approximation

$$f(x) = \frac{1}{2} f_1(x) + f_2(x). \hspace{1cm} (13)$$

It is evident that the larger matrix sizes imply an increased computational cost for the matrix inversion step. In Section 2.3, we show how the sizes of (3) and (4) can be reduced by using the notion of rectangular kernels.

2.3. Reduced twin support vector regression

In the present case the $(n \times (n+1))$-sized matrix $H = [K(A,A^T)] e_1$ contains a square kernel matrix of size $n \times n$. In the TSVR, the kernel matrix $K(A,A^T)$ is a square $n \times n$ sized matrix in which the $(ij)$th entry is given by $K(A_i, A_j)$, where $A_i$ and $A_j$ denote the ith and jth patterns, respectively. A rectangular kernel is a $n \times n_1$ sized matrix in which the $(ij)$th entry is given by $K(A_i, A_j)$, where $A_i$ is a $n_1$-sized randomly chosen subset of the training samples. In practice, $n_1$ may be 10% of $n$, or even smaller.

As a result, the size of $H = [K(A,A^T)] e_1$ becomes $n \times (n + 1)$. From (11) and (12), we observe that the matrix $(H^T H)$ needs to be inverted. The size of this matrix in the TSVR is $(n + 1) \times (n + 1)$. The use of rectangular kernels causes the size of $(H^T H)$ to reduce to $(n_1 + 1) \times (n_1 + 1)$. Since the worst case computational complexity of inverting a $n \times n$ matrix is $O(n^3)$, this can lead to significant computational savings. For example, if $n = 1000$ and $n_1 = 100$, this implies computational savings of by a factor of 1000.

Determining the quadratic terms in (11) and (12), viz. $H^T H^{-1} H^T$ requires additional matrix multiplications. The use of rectangular kernels reduces the matrix multiplication time from $O(2n^3)$ to $O(2n_1 n + n^2 n_1)$; with the above mentioned numbers, this implies a factor of about 10. However, there is no decrease in the size of the QPP. Hence, total computational time reduces due to the reduction in the sizes of the matrices involved.

The time taken to calculate the quadratic terms, i.e. $H^T H^{-1} H^T$ for the TSVR and $H^T H^{-1} H^T$ for the RTSVR has been defined as the setup time. Thus we can see that setup time is actually a part of the total computation time. In this paper, the total computation time is also referred to as the execution time and comprises the time taken to setup the TSVR and also the time to solve it. Hence a decrease in setup time will lead to decrease in total time. The use of rectangular kernels also leads to reduced memory requirements to store the kernel $K(A, A^T)$.

3. Experiments and results

We compared the reduced TSVR with the TSVR on several datasets from the UCI repository [10], as well as on some standard datasets.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>RTSVR</th>
<th>TSVR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sin(x)</td>
<td>0.0100 ± 0.0101</td>
<td>3E-5 ± 5E-5</td>
</tr>
<tr>
<td>Cos(x)</td>
<td>0.4331 ± 0.0395</td>
<td>8E-5 ± 2E-4</td>
</tr>
<tr>
<td>Sin(x)</td>
<td>0.4704 ± 0.0686</td>
<td>0.0001 ± 0.0001</td>
</tr>
<tr>
<td>Auto-Mpg</td>
<td>0.0638 ± 0.0159</td>
<td>0.0346 ± 0.0081</td>
</tr>
<tr>
<td>Boston housing</td>
<td>0.0374 ± 0.0138</td>
<td>0.0056 ± 0.0027</td>
</tr>
<tr>
<td>Concrete CS</td>
<td>0.0432 ± 0.0098</td>
<td>0.0060 ± 0.0019</td>
</tr>
<tr>
<td>CPU machine</td>
<td>0.0034 ± 0.0041</td>
<td>0.0013 ± 0.0025</td>
</tr>
<tr>
<td>Flares</td>
<td>0.0137 ± 0.0050</td>
<td>0.0132 ± 0.0034</td>
</tr>
<tr>
<td>Forest fires</td>
<td>0.0034 ± 0.0064</td>
<td>0.0034 ± 0.0064</td>
</tr>
<tr>
<td>Parkinson</td>
<td>0.0762 ± 0.0110</td>
<td>0.0385 ± 0.0038</td>
</tr>
<tr>
<td>Servo</td>
<td>0.0346 ± 0.0146</td>
<td>0.0076 ± 0.0042</td>
</tr>
<tr>
<td>Tic data 2000</td>
<td>0.0596 ± 0.0095</td>
<td>0.0616 ± 0.0111</td>
</tr>
</tbody>
</table>
functions. In the UCI datasets, rows with missing attributes were removed. The RTSVR employed a rectangular kernel using only 10% of the dataset. The codes were written in MATLAB 7.7 2008b and executed on an Intel i7 processor (2.53 GHz) with 4 GB RAM, and the Windows 7 operating system. The values of $c_1$ and $c_2$ were kept at 0.1 for all the experiments.

Table 1 compares the error rates obtained by running the TSVR and the RTSVR on various functions and UCI datasets. The first three rows of the Table 1 are functions $\sin(x)$, $\cos(x)$ and $\sin(x)$. All these three were approximated on a range of $-50 \leq x \leq 50$. The step size for the intervals in all these functions were 0.2. The rest of the datasets is standard UCI datasets [10].

Table 2 compares the TSVR and the RTSVR with regard to the setup and total time taken. As mentioned earlier, the time taken to calculate the quadratic terms, i.e. $H^{T}H$ for the TSVR and $H^{T}H + H^{-1}H^{-T}$ for the RTSVR has been defined as the setup time. The total computation time is also referred to as the execution time and comprises the time taken to setup the QPP and also the time to solve it. The sizes of datasets have been mentioned along the dataset names in Table 2.

The RTSVR shows significantly reduced setup and total execution times in comparison to the TSVR, at the expense of a slight increase in the error rate. Note that the setup time is a significant component of the total time, and justifies the use of rectangular kernels even though the QPP sizes are unchanged. The results show that even on large datasets, the RTSVR displays high test set accuracies.

4. Conclusion

In this paper, we used the notion of rectangular kernels to obtain the reduced twin SVR, a variant of the twin SVR, that can yield comparable error rates at significantly reduced computation costs.

References


Mittal Singh obtained the Integrated M.Tech. in Mathematics and Computing from the Department of Mathematics, Indian Institute of Technology, Delhi in 2010. He is currently a Ph.D. student at the Saarbrucken Graduate School of Computer Science. He received the Central Board of Secondary Education's Merit Scholarship from the years 2005–2009. He was awarded the Army Education Merit Scholarship for the year 2006. He was a guest speaker at an Advanced Matlab Workshop at IIT Delhi. His research interests include distributed computing, datamining, machine learning and optimization.

Jivitej Chadha completed his Bachelor’s and Master’s degree from IIT-Delhi, where he worked in the areas of machine learning and approximation algorithms. He worked on the problem of “Job Scheduling on multiple machines to minimize the flow time” for his master’s thesis. Currently, he works in the area of computational finance at Tower Research Capital. His work includes application of econometric and machine learning techniques for financial modeling.

Puneet Ahuja, received his Bachelor’s and Master’s degree in Computer Science and Engineering from IIT-Delhi in 2009. He is currently working at Tower Research Capital. His work involves designing and implementing a high-frequency trading platform. His research interest include optimization, econometrics and machine learning.
Jayadeva obtained his B.Tech and Ph.D. degrees from the Department of Electrical Engineering, IIT Delhi. He is currently a Professor in the same department. He has been a speaker on the IEEE Computer Society Distinguished Visitor Programme, a recipient of the Young Engineer Award from the Indian National Academy of Engineering, the Young Scientist Award from the Indian National Science Academy, and the BOYSCAST Fellowship from the Department of Science and Technology, Government of India. He was a URSI Young Scientist at the General Assembly in Lille, France (1996), and received the Sir J.C. Bose Young Scientist title from the Indian Council of the URSI. He visited the Department of Brain and Cognitive Sciences, Massachusetts Institute of Technology in 1997 as a BOYSCAST fellow. He spent a sabbatical year as a visiting Researcher at IBM India Research Laboratory, from July 2006. One of his papers in Neurocomputing was listed on the Top 25 hot list; he is a recipient of best paper awards from the IETE Journal of Research, and three other conference papers. He holds a US Patent on A/D conversion, and is the co-author of the book Numerical Optimization and Applications. His research interests include machine learning, optimization, and VLSI. He is the co-author of a book with Profs. Suresh Chandra and Aparna Mehra, entitled “Numerical Optimization with Applications”.

Suresh Chandra received the M.S. degree in Mathematical Statistics from Lucknow University, and his Ph.D. from the Indian Institute of Technology, Kanpur. He is currently a Professor at the Department of Mathematics, Indian Institute of Technology, Delhi, India. He has authored and co-authored more than 100 publications in refereed journals and international conferences and co-authored two books, one on Fuzzy Mathematical Programming and Fuzzy Matrix Games and the other on Principles of Optimization Theory. His research interests include numerical optimization, mathematical programming, generalized convexity, fuzzy optimization, fuzzy games, neural networks, machine learning and financial mathematics. He is a Member of the Editorial Board for the International Journal of Management and Systems, and the Journal of Decision Sciences. Also he is a Senior Member of the Operational Research Society of India, and a Member of the International Working Group on Generalized Convexity and Applications.