Neurocomputing 161 (2015) 243–253

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Householder transformation based sparse least squares support vector regression

Yong-Ping Zhao [a,*], Bing Li [a], Ye-Bo Li [b], Kang-Kang Wang [a]

[a] School of Mechanical Engineering, Nanjing University of Science and Technology, Nanjing 210094, China
[b] AVIC Aeroengine Control Research Institute, Wuxi 214063, China

ARTICLE INFO

Article history:
Received 27 June 2014
Received in revised form
5 January 2015
Accepted 11 February 2015
Communicated by W.S. Hong
Available online 5 March 2015

Keywords:
Least squares support vector machine
Kernel method
Sparseness
Householder transformation

ABSTRACT

Sparseness is a key problem in modeling problems. To sparsify the solution of normal least squares support vector regression (LSSVR), a novel sparse method is proposed in this paper, which recruits support vectors sequentially by virtue of Householder transformation, here HSLSSVR for short. In HSLSSVR, there are two benefits. On one hand, a recursive strategy is adopted to solve the linear equation set instead of solving it from scratch. During each iteration, the training sample incurring the maximum reduction on the residuals is recruited as support vector. On the other hand, in the process of solving the linear equation set, its condition number does not deteriorate, so the numerical stability is guaranteed. The reports from experiments on benchmark data sets and a real-world mechanical system to calculate the inverse dynamics of a robot arm demonstrate the effectiveness and feasibility of the proposed HSLSSVR.

1. Motivation

In the past two decades, support vector machine (SVM) [1–4] has been widely studied and applied due to its excellent performance on classification, regression and probability density estimation. As a member of SVM family, since least square support vector machine (LSSVM) [5,6] was proposed, it has drawn much attention due to its simple format and fast training speed. That is, LSSVM only needs to solve a set of linear equations compared to the quadratic programming in the normal SVM, so the training burden is cut down obviously. In the regression case, there is corresponding least square support vector machine (LSSVR) to solve function approximation problems. However, compared to the normal SVM, LSSVM/LSSVR is in lack of sparseness [7]. Sparse models are preferable in engineering applications since a model's computational complexity scales with its model complexity. Moreover, a sparse model is easier to interpret from the viewpoint of knowledge extraction [8]. Hence, a lot of efforts have been made to sparsify the solution of LSSVM/LSSVR.

Firstly, Suykens et al. [9] investigated imposing sparseness on LSSVM by pruning support values (PSVLSSVM) from the sorted support value spectrum which results from the solution to the linear equation. Different from omitting samples that has a small error in the previous pass [9], De Kruif et al. [10] introduced a procedure in which the training sample introducing smallest approximation error when it is omitted will be pruned, i.e., pruning error minimization in LSSVM (PEMLSSVM). Subsequently, Kuh et al. [11] accelerated PEMLSSVM by adding the regularization. Zeng et al. [12] introduced the sequential minimal optimization method into pruning process, instead of determining the pruning samples by errors, which omits the samples that will introduce minimum changes to a dual objective function. Combined with the reduced technique [13,14], reduced LSSVR (RLSSVR) is also of sparseness. To sparsify the solution of LSSVM, a fast sparse approximation (FSALSSVM) was proposed [15], which iteratively builds the decision function by adding one basis function from a kernel-based dictionary at one time. By adding the bias term to the objective function, LSSVM can be sparsified with forward least squares approximation [16]. Based on the Nyström approximation and quadratic Rényi entropy, a fixed-size LSSVM (FSLSSVM) was proposed to realize the sparse representation in primal weight space and applied successfully to electric load forecasting [17–19]. As we know, LSSVR boils down to solving a set of linear equations. If we are able to sparsify this linear equation set in the least square sense, then a sparse LSSVR is obtained equivalently. Hence, in this paper, Householder transformation [20,21] is used to orthogonalize this linear equation set and recruit so-called support vectors sequentially, thus obtaining a sparse solution. Naturally, this proposed algorithm is named Householder transformation based sparse LSSVR (HSLSSVR for short). To confirm the effectiveness and feasibility of the proposed HSLSSVR, ten benchmark data sets and a

* Corresponding author.
E-mail address: y.p.zhao@163.com (Y.-P. Zhao).

http://dx.doi.org/10.1016/j.neucom.2015.02.037
0925-2312/

real-world mechanical system are utilized to do experiments. From these reports, it is easily got that HSLSSVR is effective and feasible.

The remainder of this paper is organized as follows. In Section 2, the normal least square support vector regression is briefly introduced and its drawback on lack of sparseness is pointed out. In the following section, to sparsify the solution of normal LSSVR, HSLSSVR is elaborated on and its detailed procedure is listed. The experimental results and analyses are presented in Section 4. Finally, conclusions are given in Section 5.

2. Least squares support vector regression

Considering a training set of l pairs of samples $\{(\boldsymbol{x}_i, d_i)\}_{i=1}^{l}$ for regression problem, where $\boldsymbol{x}_i \in \Re^n$ is the input vector and $d_i \in \Re$ is the corresponding prediction value, hence LSSVR model [5,6] is described in the following:

$$\min_{\boldsymbol{w},b,\boldsymbol{\xi}} \left\{ \frac{1}{2}\boldsymbol{w}^T\boldsymbol{w} + \frac{C}{2}\sum_{i=1}^{l}\xi_i^2 \right\}$$
$$\text{s.t.} \quad d_i = \boldsymbol{w}^T\varphi(\boldsymbol{x}_i) + b + \xi_i, \quad i = 1, \dots, l \tag{1}$$

where $\boldsymbol{w}$ is the weighted vector of the model, $\boldsymbol{\xi} = [\xi_1, \dots, \xi_l]^T$ is the residual vector, $\varphi(\boldsymbol{x}_i)$ is a nonlinear mapping which can maps $\boldsymbol{x}_i$ in the original space into a high-dimensional feature space, C is the regularization parameter which makes a tradeoff between the model complexity and the fitting errors, b is the learning bias. From (1), the Lagrangian function is found as

$$L(\boldsymbol{w}, b, \boldsymbol{\xi}, \boldsymbol{\alpha}) = \frac{1}{2}\boldsymbol{w}^T\boldsymbol{w} + \frac{C}{2}\sum_{i=1}^{l}\xi_i^2 + \sum_{i=1}^{l}\alpha_i(d_i - \boldsymbol{w}^T\varphi(\boldsymbol{x}_i) - b - \xi_i) \tag{2}$$

where $\boldsymbol{\alpha} = [\alpha_1, \dots, \alpha_l]^T$ is the Lagrangian multiplier vector.

The optimal conditions are got as

$$\begin{cases} \dfrac{\partial L}{\partial \boldsymbol{w}} = 0 \rightarrow \boldsymbol{w} = \sum_{i=1}^{l}\tilde{\alpha}_i\varphi(\boldsymbol{x}_i) & \text{(a)} \\ \dfrac{\partial L}{\partial b} = 0 \rightarrow \sum_{i=1}^{l}\tilde{\alpha}_i = 0 & \text{(b)} \\ \dfrac{\partial L}{\partial \xi_i} = 0 \rightarrow \tilde{\alpha}_i = C\xi_i & \text{(c)} \\ \dfrac{\partial L}{\partial \alpha_i} = 0 \rightarrow \boldsymbol{w}^T\varphi(\boldsymbol{x}_i) + \tilde{b} + \xi_i - d_i = 0 & \text{(d)} \end{cases} \tag{3}$$

Eliminating $\boldsymbol{w}$ and ξ_i $(i = 1, \dots, l)$ from Eqs. (3a)–(3d), the following system of linear equations is got

$$\begin{bmatrix} 0 & \boldsymbol{1}^T \\ \boldsymbol{1} & \boldsymbol{K} + \boldsymbol{I}/C \end{bmatrix} \begin{bmatrix} \tilde{b} \\ \tilde{\boldsymbol{\alpha}} \end{bmatrix} = \begin{bmatrix} 0 \\ \boldsymbol{d} \end{bmatrix} \tag{4}$$

where $\boldsymbol{d} = [d_i, \dots, d_l]^T$, $\boldsymbol{1}$ is a column vector of all ones with appropriate dimension, $\boldsymbol{I}$ is an identity matrix of appropriate dimension, the elements of the kernel matrix $\boldsymbol{K}$ are determined by $\boldsymbol{K}_{ij} = \varphi(\boldsymbol{x}_i)^T\varphi(\boldsymbol{x}_j) = k(\boldsymbol{x}_i, \boldsymbol{x}_j)$, here $k(\cdot,\cdot)$ represents the kernel function.

The Lagrangian multipliers $\tilde{\alpha}_i$ $(i = 1, \dots, l)$ and the bias $\tilde{b}$ are got through solving Eq. (4), so the LSSVR model is obtained as

$$f_{\text{LSSVR}}(\boldsymbol{x}) = \sum_{i=1}^{l}\tilde{\alpha}_i k(\boldsymbol{x}_i, \boldsymbol{x}) + \tilde{b} \tag{5}$$

From Eq. (3c), it is easily understood that the Lagrangian multiplier $\tilde{\alpha}_i$ is in direct proportion to the residual error ξ_i. Hence, if $\xi_i \neq 0$, $\tilde{\alpha}_i$ will not equal zero. Generally speaking, it is very difficult that ξ_i is exactly equal to zero, so almost every $\tilde{\alpha}_i \neq 0$. In other words, almost every training sample is support vector in LSSVR. Since LSSVR has a *dense* solution, the real time in the testing phase will be impaired. Commonly, engineers prefer sparse models to *dense* ones, because sparse models usually signify less computational complexity and faster responses. Therefore, it is necessary and important to develop methods of sparsifying the solution of LSSVR.

3. Sparsify LSSVR based on Householder transformation

3.1. Householder transformation

Given a column vector $\boldsymbol{y}$, it can be transformed into the following form in an isometric mapping mode using Householder transformation [20,21]:

$$\boldsymbol{H}\boldsymbol{y} = -\text{sign}(y_i)\|\boldsymbol{y}\|\boldsymbol{e}_i \tag{6}$$

where $\boldsymbol{e}_i$ is the ith column of identity matrix, $\|\cdot\|$ represents the Euclidean norm, y_i is the ith element of $\boldsymbol{y}$,

$$\text{sign}(y_i) = \begin{cases} 1, & y_i \geq 0 \\ -1, & y_i < 0 \end{cases} \tag{7}$$

and $\boldsymbol{H}$ is a Householder matrix satisfying

$$\boldsymbol{H} = \boldsymbol{I} - 2\frac{\boldsymbol{v}\boldsymbol{v}^T}{\boldsymbol{v}^T\boldsymbol{v}} \tag{8}$$

with

$$\boldsymbol{v} = \boldsymbol{y} + \text{sign}(y_i)\|\boldsymbol{y}\|\boldsymbol{e}_i \tag{9}$$

From Eq. (8), it is easily known that if $\boldsymbol{H}$ is a Householder matrix, it satisfies the following properties:

(1) Symmetric: $\boldsymbol{H} = \boldsymbol{H}^T$
(2) Unitary: $\boldsymbol{H}^{-1} = \boldsymbol{H}^T$
(3) Involutory: $\boldsymbol{H}^2 = \boldsymbol{I}$

3.2. Sparsify the solution of LSSVR

HSLSSVR is a sequential forward greedy algorithm, which starts with no support vectors and gradually recruits one support vector at each iteration until the stopping criterion is satisfied. In HSLSSVR, two key components must be solved: the recruitment of support vectors and the solution of subproblem. Assume that at the $(n-1)$th iteration, there are $(n-1)$ training samples recruited as support vectors, i.e., $\{\boldsymbol{x}_i | i \in S\}$, where S is the index set of support vectors. In this situation, Eq. (4) is rearranged as

$$\begin{bmatrix} 0 & \boldsymbol{1}_{|S|}^T \\ \boldsymbol{1}_l & \overline{\boldsymbol{K}}_S \end{bmatrix} \begin{bmatrix} \tilde{b}^{(n-1)} \\ \tilde{\boldsymbol{\alpha}}_S^{(n-1)} \end{bmatrix} = \begin{bmatrix} 0 \\ \boldsymbol{d} \end{bmatrix} \tag{10}$$

where $\boldsymbol{1}_{|S|}^T$ include $|S|$ ones, here $|\cdot|$ represents the cardinality of set, similarly $\boldsymbol{1}_l$ includes l ones, $\overline{\boldsymbol{K}}_S = [\overline{\boldsymbol{k}}_i, \dots, \overline{\boldsymbol{k}}_j](i, j \in S)$ with $\overline{\boldsymbol{k}}_i = \boldsymbol{k}_i + \boldsymbol{e}_i/C$, $\boldsymbol{k}_i$ is the column vector of the kernel matrix $\boldsymbol{K}$ corresponding to the index i, $\tilde{\boldsymbol{\alpha}}_S^{(n-1)}$ is a subvector consisting of elements confining to the index set S at the $(n-1)$th iteration. Evidently, Eq. (10) is an overdetermined linear equation set. Its solution in the least square sense amounts to the optimal solution of the following problem:

$$\min_{b^{(n-1)}, \boldsymbol{\alpha}_S^{(n-1)}} \left\{ G_S^{(n-1)} = \left\| \begin{bmatrix} 0 & \boldsymbol{1}_{|S|}^T \\ \boldsymbol{1}_l & \overline{\boldsymbol{K}}_S \end{bmatrix} \begin{bmatrix} b^{(n-1)} \\ \boldsymbol{\alpha}_S^{(n-1)} \end{bmatrix} - \begin{bmatrix} 0 \\ \boldsymbol{d} \end{bmatrix} \right\|_2^2 \right\} \tag{11}$$

It is easily got the optimal solution of (11) as

$$\begin{bmatrix} \tilde{b}^{(n-1)} \\ \tilde{\boldsymbol{\alpha}}_S^{(n-1)} \end{bmatrix} = \left(\begin{bmatrix} 0 & \boldsymbol{1}_{|S|}^T \\ \boldsymbol{1}_l & \overline{\boldsymbol{K}}_S \end{bmatrix}^T \begin{bmatrix} 0 & \boldsymbol{1}_{|S|}^T \\ \boldsymbol{1}_l & \overline{\boldsymbol{K}}_S \end{bmatrix} \right)^{-1} \begin{bmatrix} 0 & \boldsymbol{1}_{|S|}^T \\ \boldsymbol{1}_l & \overline{\boldsymbol{K}}_S \end{bmatrix}^T \begin{bmatrix} 0 \\ \boldsymbol{d} \end{bmatrix} \tag{12}$$