Journal of Systems Engineering and Electronics
Vol. 34, No. 4, August 2023, pp.827 – 838

Robust least squares projection twin SVM and its sparse solution

ZHOU Shuisheng[1], ZHANG Wenmeng[1], CHEN Li[2,3,*], and XU Mingliang[3]

1. School of Mathematics and Statistics, Xidian University, Xi'an 710126, China;
2. School of Physical Education, Zhengzhou University, Zhengzhou 450001, China;
3. School of Computer and Artificial Intelligence, Zhengzhou University, Zhengzhou 450001, China

Abstract: Least squares projection twin support vector machine (LSPTSVM) has faster computing speed than classical least squares support vector machine (LSSVM). However, LSPTSVM is sensitive to outliers and its solution lacks sparsity. Therefore, it is difficult for LSPTSVM to process large-scale datasets with outliers. In this paper, we propose a robust LSPTSVM model (called R-LSPTSVM) by applying truncated least squares loss function. The robustness of R-LSPTSVM is proved from a weighted perspective. Furthermore, we obtain the sparse solution of R-LSPTSVM by using the pivoting Cholesky factorization method in primal space. Finally, the sparse R-LSPTSVM algorithm (SR-LSPTSVM) is proposed. Experimental results show that SR-LSPTSVM is insensitive to outliers and can deal with large-scale datasets fastly.

Keywords: outliers, robust least squares projection twin support vector machine (R-LSPTSVM), low-rank approximation, sparse solution.

DOI: 10.23919/JSEE.2023.000103

1. Introduction

Support vector machine (SVM) [1,2] as a powerful tool for supervised learning, is widely used in classification and regression from various aspects such as face detection [3], financial applications [4], disease diagnosis [5], and image classification [6]. The principle of SVM is to maximize the distance between two parallel hyperplanes, which is achieved by solving a quadratic programming problem (QPP). Although SVM owns better generalization performance compared with other machine learning methods, solving QPP is time-consuming and its computational complexity is $O(m^3)$, where m is the number of training samples. Twin SVM (TWSVM) [7] based on two nonparallel proximal planes is emerged in recent years. In TWSVM, each proximal plane is close to one class of samples and far away from another. Compared to SVM, TWSVM solves two smaller sized QPPs and the computational complexity of TWSVM is only one fourth of that of SVM. Therefore, TWSVM works faster than SVM whereas the performance of TWSVM is similar to that of SVM [8].

Projection twin support vector machine (PTSVM) [9] is an improvement of TWSVM. It searches for two optimal projection directions, so that the projected samples of one class are closer to its projection centroid, and those of another class are farther away from it as much as possible. PTSVM has been studied from various perspectives. Specifically, in order to reduce the time complexity, least squares PTSVM (LSPTSVM) [10,11] was proposed, which solves two linear equations instead of QPPs. However, LSPTSVM has two limitations, one is that it is sensitive to outliers, the other is that its solution lacks sparsity. Therefore, LSPTSVM is difficult to solve large-scale datasets, especially the dataset containing outliers.

Many works have been presented to improve the sparsity of solution of least square models, Suykens et al. [12] proposed a pruning method to improve the sparsity of the dual LSSVM (D-LSSVM) by iteratively discarding 5% samples with the smallest absolute values of present solution. Jiao et al. [13] proposed a fast sparse approximation algorithm (FSA-LSSVM) for D-LSSVM by iteratively constructing an approximation classification function by adding the basis function from a kernel-based dictionary one after another until the termination condition is reached. Zhou [14] presented a pivoted Choleskian of primal LSSVM (PCP-LSSVM) by using low rank approximation of the kernel matrix. Zhou et al. [15] proposed a revised least angle regression LSSVM (RLARS-LSSVM)

Manuscript received April 13, 2022.

*Corresponding author.

This work was supported by the National Natural Science Foundation of China (61772020;62202433;62172371;62272422;62036010), the Natural Science Foundation of Henan Province (22100002), and the Postdoctoral Research Grant in Henan Province (202103111).

and got its sparse solution by solving a least absolution shrinkage and selection operator problem. Cheng et al. [16] introduced an iterative pruning error minimization algorithm and L_0-norm minimization algorithm to improve the sparsity of LSSVM. Sun et al. [17] introduced a localized generalization error model to prune the support vectors in LSSVM. Ma et al. [18] proposed a global-representation-based sparse LSSVM (GRS-LSSVM) based on the selection of globally representative points according to the density and dispersion of the points in the feature space.

In terms of enhancing the robustness of models to outliers, Suykens et al. [19] proposed a weighted LSSVM (W-LSSVM) by setting weights on samples based on the error distribution. Wei et al. [20] proposed an LSSVM with linear programming method (LSSVM-LP) based on the idea of basis pursuit (BP) in the whole feasible region. Wen et al. [21] presented a recursive outlier elimination-based LSSVM (ROELS-SVM) algorithm by employing a criterion derived from robust linear regression. Yang et al. [22] proposed a novel robust LSSVM (R-LSSVM) by using the truncated least squares loss function, and Chen et al. [23] proved the robustness of R-LSSVM in theory. Ye et al. [24] proposed a L_p-norm LSSVR by using the L_p-norm regularization term and the absolute constraint. Lu et al. [25] proposed a robust LSSVM by minimizing both the mean and variance of the modeling errors.

In this paper, we propose a robust LSPTSVM by using the truncated least squares loss to solve the defect of LSPTSVM being sensitive to outliers, and derive the sparse solution of the robust LSPTSVM by using the low-rank approximation of the kernel matrix. The sparse robust LSPTSVM algorithm is proposed to handle large-scale dataset with outliers.

The paper is organized as follows. Section 2 briefly introduces PTSVM and LSPTSVM. Section 3 proposes our robust LSPTSVM and derives its sparse solution. Furthermore, we interpret the robustness of the robust LSPTSVM and analyze the convergence and complexity of the sparse robust LSPTSVM algorithm. Experimental results and conclusions are given in the last two sections.

2. Background

In this section, notations used throughout the paper, PTSVM [9], and LSPTSVM [10] are introduced.

2.1 Notations

Consider a binary classification problem in n-dimensional real space $\mathbf{R}^n$. The training set denotes as $\boldsymbol{T}=\{\boldsymbol{x}_s^{(i)},y^{(i)}|i=1,2;s=1,2,\cdots,m_i\}$ where $\boldsymbol{x}_s^{(i)}\in\mathbf{R}^n$ is the sth sample belonging to the ith class and $m_1+m_2=m$, $y^{(1)}=1$, $y^{(2)}=-1$. Let matrices $\boldsymbol{A}\in\mathbf{R}^{m_1\times n}$ and $\boldsymbol{B}\in\mathbf{R}^{m_2\times n}$ represent the samples belonging to class $+1$ and -1 respectively. Let row vectors $\bar{\boldsymbol{A}}$ and $\bar{\boldsymbol{B}}$ denote the sample mean of class $+1$ and -1, respectively.

2.2 PTSVM

PTSVM aims to seek two nonparallel projection directions such that in each projection direction the within-class variance of its own class instances is minimized while the other class projection instances are scattered as far as possible. Let

$$\boldsymbol{S}_1=\sum_{i=1}^{m_1}\left(\boldsymbol{x}_i^{(1)}-\bar{\boldsymbol{A}}\right)\left(\boldsymbol{x}_i^{(1)}-\bar{\boldsymbol{A}}\right)^{\mathrm{T}},$$

$$\boldsymbol{S}_2=\sum_{i=1}^{m_2}\left(\boldsymbol{x}_i^{(2)}-\bar{\boldsymbol{B}}\right)\left(\boldsymbol{x}_i^{(2)}-\bar{\boldsymbol{B}}\right)^{\mathrm{T}}.$$

The PTSVM model is expressed as

$$\begin{aligned}&\min_{\boldsymbol{w}_1,\boldsymbol{\xi}}\ \frac{1}{2}\boldsymbol{w}_1^{\mathrm{T}}\boldsymbol{S}_1\boldsymbol{w}_1+c_1\boldsymbol{e}_2^{\mathrm{T}}\boldsymbol{\xi}\\&\text{s.t.}\ \ \boldsymbol{B}\boldsymbol{w}_1-\bar{\boldsymbol{A}}\boldsymbol{w}_1+\boldsymbol{\xi}\geqslant\boldsymbol{e}_2,\ \boldsymbol{\xi}\geqslant 0,\end{aligned}\tag{1}$$

$$\begin{aligned}&\min_{\boldsymbol{w}_2,\boldsymbol{\eta}}\ \frac{1}{2}\boldsymbol{w}_2^{\mathrm{T}}\boldsymbol{S}_2\boldsymbol{w}_2+c_2\boldsymbol{e}_1^{\mathrm{T}}\boldsymbol{\eta}\\&\text{s.t.}\ \ -(\boldsymbol{A}\boldsymbol{w}_2-\bar{\boldsymbol{B}}\boldsymbol{w}_2)+\boldsymbol{\eta}\geqslant\boldsymbol{e}_1,\boldsymbol{\eta}\geqslant 0,\end{aligned}\tag{2}$$

where $c_1,c_2>0$ are penalty parameters.

Denote $\boldsymbol{\alpha},\boldsymbol{\beta}$ as the solutions of the dual problems of (1) and (2). The projection axes can be constructed as

$$\boldsymbol{w}_1=\boldsymbol{S}_1^{-1}\boldsymbol{E}^{\mathrm{T}}\boldsymbol{\alpha},\tag{3}$$

$$\boldsymbol{w}_2=\boldsymbol{S}_2^{-1}\boldsymbol{F}^{\mathrm{T}}\boldsymbol{\beta},\tag{4}$$

where $\boldsymbol{E}=\boldsymbol{B}-\boldsymbol{e}_2\bar{\boldsymbol{A}}$, $\boldsymbol{F}=\boldsymbol{A}-\boldsymbol{e}_1\bar{\boldsymbol{B}}$, and $\boldsymbol{e}_1,\boldsymbol{e}_2$ are vectors whose elements are all 1.

2.3 LSPTSVM

Different from PTSVM, LSPTSVM adopts least squares loss function and equality constraints. Moreover, it introduces maximum margin regularization $\frac{c_3}{2}\|\boldsymbol{w}_1\|^2$ and $\frac{c_4}{2}\|\boldsymbol{w}_2\|^2$. The model of LSPTSVM is given by

$$\begin{aligned}&\min_{\boldsymbol{w}_1,\boldsymbol{\xi}}\ \frac{1}{2}\boldsymbol{w}_1^{\mathrm{T}}\boldsymbol{S}_1\boldsymbol{w}_1+\frac{c_1}{2}\boldsymbol{\xi}^{\mathrm{T}}\boldsymbol{\xi}+\frac{c_3}{2}\|\boldsymbol{w}_1\|^2\\&\text{s.t.}\ \ \boldsymbol{B}\boldsymbol{w}_1-\bar{\boldsymbol{A}}\boldsymbol{w}_1+\boldsymbol{\xi}=\boldsymbol{e}_2,\end{aligned}\tag{5}$$

$$\begin{aligned}&\min_{\boldsymbol{w}_2,\boldsymbol{\eta}}\ \frac{1}{2}\boldsymbol{w}_2^{\mathrm{T}}\boldsymbol{S}_2\boldsymbol{w}_2+\frac{c_2}{2}\boldsymbol{\eta}^{\mathrm{T}}\boldsymbol{\eta}+\frac{c_4}{2}\|\boldsymbol{w}_2\|^2\\&\text{s.t.}\ \ \boldsymbol{A}\boldsymbol{w}_2-\bar{\boldsymbol{B}}\boldsymbol{w}_2+\boldsymbol{\eta}=\boldsymbol{e}_1,\end{aligned}\tag{6}$$

where $c_3,c_4>0$ are regularization parameters.

Setting the gradient of the objective functions with respect to $\boldsymbol{w}_1$ and $\boldsymbol{w}_2$ to zero, we obtain the projection axes in primal space as

$$\boldsymbol{w}_1 = \left(\frac{\boldsymbol{S}_1}{c_1} + \boldsymbol{E}^{\mathrm{T}}\boldsymbol{E} + \frac{c_3}{c_1}\boldsymbol{I}\right)^{-1} \boldsymbol{E}^{\mathrm{T}}\boldsymbol{e}_2, \tag{7}$$

$$\boldsymbol{w}_2 = -\left(\frac{\boldsymbol{S}_2}{c_2} + \boldsymbol{F}^{\mathrm{T}}\boldsymbol{F} + \frac{c_4}{c_2}\boldsymbol{I}\right)^{-1} \boldsymbol{F}^{\mathrm{T}}\boldsymbol{e}_1, \tag{8}$$

where $\boldsymbol{I}$ is an identity matrix.

3. Sparse R-LSPTSVM algorithm

In this section, we propose a robust LSPTSVM (R-LSPTSVM), model and the sparse R-LSPTSVM (SR-LSTPSVM) algorithm to achieve the sparse solution of R-LSPTSVM. Then, we analyze convergence and complexity of SR-LSPTSVM. Finally, we discuss the robustness of R-LSPTSVM from a weighted perspective.

3.1 R-LSPTSVM

Because the value of least squares loss function $L_{\mathrm{sq}}(\boldsymbol{\xi})$ increases infinitely as $\boldsymbol{\xi}$ increases and the $\boldsymbol{\xi}$ value of outlier is usually large, LSPTSVM is sensitive to outliers. To overcome this defect, we introduce the truncated least squares loss function $L_\tau(\boldsymbol{\xi}) = \min(\tau^2, \boldsymbol{\xi}^2)/2$ and propose R-LSPTSVM, where τ is the truncated parameter that can limit the value of loss function.

The truncated least squares loss function $L_\tau(\boldsymbol{\xi})$ has the following good properties. It is bounded and non-convex, whereas it can be decomposed into the difference between two convex functions. Therefore, the model with $L_\tau(\boldsymbol{\xi})$ is simple and we can use the convex concave procedure (CCCP) to solve it iteratively. Zhou et al. discussed other loss functions with similar performance in [26]. Among those functions, $L_\tau(\boldsymbol{\xi})$ is a simpler and effective function.

According to the above analysis, we choose the bounded loss function $L_\tau(\boldsymbol{\xi})$ to improve the robustness of LSPTSVM. The proposed R-LSPTSVM model is represented as follows:

$$\min_{\boldsymbol{w}_1,\xi_k} \frac{1}{2}\boldsymbol{w}_1^{\mathrm{T}}\boldsymbol{S}_1\boldsymbol{w}_1 + \frac{c_3}{2}\|\boldsymbol{w}_1\|^2 + \frac{c_1}{m_2}\sum_{k=1}^{m_2} L_\tau(\xi_k)$$
$$\text{s.t.} \begin{cases} 1-\left(\boldsymbol{w}_1^{\mathrm{T}}\boldsymbol{x}_k^{(2)} - \bar{\boldsymbol{A}}\boldsymbol{w}_1\right) = \xi_k \\ k = 1,2,\cdots,m_2 \end{cases}, \tag{9}$$

$$\min_{\boldsymbol{w}_2,\boldsymbol{\eta}_k} \frac{1}{2}\boldsymbol{w}_2^{\mathrm{T}}\boldsymbol{S}_2\boldsymbol{w}_2 + \frac{c_4}{2}\|\boldsymbol{w}_2\|^2 + \frac{c_2}{m_1}\sum_{k=1}^{m_1} L_\tau(\boldsymbol{\eta}_k)$$
$$\text{s.t.} \begin{cases} 1+\boldsymbol{w}_2^{\mathrm{T}}\boldsymbol{x}_k^{(1)} - \bar{\boldsymbol{B}}\boldsymbol{w}_2 = \boldsymbol{\eta}_k \\ k = 1,2,\cdots,m_1 \end{cases}. \tag{10}$$

Fig. 1 shows $L_\tau(\xi)$ with $\tau = 1$. In Fig. 1, $L_\tau(\xi)$ represents truncated least squares loss function, $L_{\mathrm{sq}}(\xi)$ represents least squares loss function, $L_\tau^{\mathrm{smooth}}(\boldsymbol{\xi})$ represents smooth truncated least squares loss function. $L_2(\boldsymbol{\xi})$ can be calculated by $L_\tau(\boldsymbol{\xi}) = L_{\mathrm{sq}}(\boldsymbol{\xi}) - L_2(\boldsymbol{\xi})$. The calculations of $L_\tau^{\mathrm{smooth}}(\boldsymbol{\xi})$ and $L_2(\boldsymbol{\xi})$ are shown in Subsection 3.2.2. Obviously, the value of truncated least squares loss function is limited by $\tau^2/2$ and thus the impact of outliers is reduced. The robustness of R-LSPTSVM will be further interpreted from a weighed perspective in Subsection 3.3.

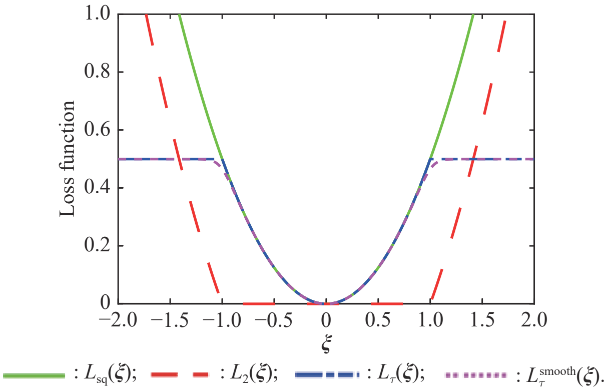


Fig. 1 Loss functions $L_\tau(\xi)$, $L_\tau^{\mathrm{smooth}}(\xi)$, $L_{\mathrm{sq}}(\xi)$, and $L_2(\xi)$

3.2 Sparse solution of R-LSPTSVM

3.2.1 Primal R-LSPTSVM

To obtain sparse solutions to problems in (9) and (10), we first represent R-LSPTSVM in primal space by representer theorem [27,28].

Theorem 1 (Representer theorem) Suppose we are given a non-empty set χ, a mapping φ from χ to a Hilbert space, training samples $(\boldsymbol{x}_1,y_1),(\boldsymbol{x}_2,y_2),\cdots,(\boldsymbol{x}_m,y_m) \in \chi \times \mathbf{R}$, a monotonically nondecreasing real-valued function $g: \mathbf{R}_+ \rightarrow \mathbf{R}$ and an arbitrary cost function $f: \mathbf{R}^m \rightarrow \mathbf{R}$. Then, minimize the regularized risk function:

$$f(\langle \boldsymbol{w}, \varphi(\boldsymbol{x}_1)\rangle, \langle \boldsymbol{w}, \varphi(\boldsymbol{x}_2)\rangle, \cdots, \langle \boldsymbol{w}, \varphi(\boldsymbol{x}_m)\rangle) + g(\|\boldsymbol{w}\|)$$

admits a representation as $\boldsymbol{w} = \sum_{i=1}^{m} \boldsymbol{\alpha}_i \varphi(\boldsymbol{x}_i)$.

It is easy to prove that the R-LSPTSVM model in (9) and (10) satisfy the representer theorem. Therefore, there exists vectors $\boldsymbol{\alpha} \in \mathbf{R}^{m_1}$ and $\boldsymbol{\beta} \in \mathbf{R}^{m_2}$ such that $\boldsymbol{w} = \sum_{i=1}^{m_1} \alpha_i \varphi(\boldsymbol{x}_i)$ and $\boldsymbol{w} = \sum_{i=1}^{m_2} \beta_i \varphi(\boldsymbol{x}_i)$ are optimal solution of in (9) and (10) respectively. Therefore, we set

$$\boldsymbol{Q}_t = \boldsymbol{K}_t\left(I - \frac{1}{m_t}\boldsymbol{e}_t\boldsymbol{e}_t^{\mathrm{T}}\right)\boldsymbol{K}_t^{\mathrm{T}} + c_{t+2}\boldsymbol{K},\ \ t = 1,2$$

where $\boldsymbol{K}_t = \boldsymbol{K}(\boldsymbol{x}, \boldsymbol{x}^{(t)})$. R-LSPTSVM in primal space without feature map $\varphi(\boldsymbol{x})$ can be expressed as

$$\min_{\alpha}\ \frac{1}{2}\boldsymbol{\alpha}^{\mathrm{T}}\boldsymbol{Q}_1\boldsymbol{\alpha}+\frac{c_1}{m_2}\sum_{k=1}^{m_2}L_\tau(\xi_k), \tag{11}$$

$$\min_{\beta}\ \frac{1}{2}\boldsymbol{\beta}^{\mathrm{T}}\boldsymbol{Q}_2\boldsymbol{\beta}+\frac{c_2}{m_1}\sum_{k=1}^{m_1}L_\tau(\eta_k), \tag{12}$$

where

$$\xi_k=1-\left(\boldsymbol{K}(\boldsymbol{x}_k^{(2)},\boldsymbol{x})-\frac{1}{m_1}\boldsymbol{e}_1^{\mathrm{T}}\boldsymbol{K}_1^{\mathrm{T}}\right)\boldsymbol{\alpha}, \tag{13}$$

$$\eta_k=1+\left(\boldsymbol{K}(\boldsymbol{x}_k^{(1)},\boldsymbol{x})-\frac{1}{m_2}\boldsymbol{e}_2^{\mathrm{T}}\boldsymbol{K}_2^{\mathrm{T}}\right)\boldsymbol{\beta}. \tag{14}$$

In (13) and (14), $\boldsymbol{K}=\varphi(\boldsymbol{x})\varphi(\boldsymbol{x})^{\mathrm{T}}$ is the kernel matrix calculated with all training samples, $\boldsymbol{K}_1$ and $\boldsymbol{K}_2$ are the kernel matrices calculated with positive and negative samples, respectively. We call (11) and (12) the primal R-LSPTSVM.

3.2.2 Solving R-LSPTSVM based on CCCP

In this subsection, we derive the sparse solution of (11), and the sparse solution of (12) can be derived similarly.

Obviously, $L_\tau(\boldsymbol{\xi})$ is non-convex, but it can be represented as the difference of two convex functions, i.e., $L_\tau(\boldsymbol{\xi})=L_{\mathrm{sq}}(\boldsymbol{\xi})-L_2(\boldsymbol{\xi})$, where $L_{\mathrm{sq}}=\boldsymbol{\xi}^2$ is the least squares loss function and

$$L_2(\xi)=\begin{cases}0, & |\boldsymbol{\xi}|\leqslant\tau\\ \dfrac{1}{2}(\boldsymbol{\xi}^2-\tau^2), & \text{otherwise}\end{cases}.$$

Then, (11) can be rewritten as a difference of convex (DC) programming:

$$\min_{\alpha}\ H_1(\boldsymbol{\alpha})-H_2(\boldsymbol{\alpha}) \tag{15}$$

where

$$H_1(\boldsymbol{\alpha})=\frac{1}{2}\boldsymbol{\alpha}^{T}\boldsymbol{Q}_1\boldsymbol{\alpha}+\frac{c_1}{m_2}\sum_{k=1}^{m_2}L_{\mathrm{sq}}(\xi_k)$$

and

$$H_2(\boldsymbol{\alpha})=\frac{c_1}{m_2}\sum_{k=1}^{m_2}L_2(\xi_k)$$

are convex functions.

Problem in (15) can be solved by CCCP [29–31], i.e., iteratively solving the following convex QPP:

$$\boldsymbol{\alpha}^{(t+1)}=\arg\min_{\boldsymbol{\alpha}\in\mathbf{R}^{m_1}}\left\{H_1(\boldsymbol{\alpha})-\left\langle\boldsymbol{\alpha},\partial H_2(\boldsymbol{\alpha}^{(t)})\right\rangle\right\}=$$
$$\arg\min_{\boldsymbol{\alpha}\in\mathbf{R}^{m_1}}\left\{H_1(\boldsymbol{\alpha})-\frac{c_1}{m_2}\sum_{k=1}^{m_2}\gamma_k^{(t)}(1-\xi_k)\right\} \tag{16}$$

where $\gamma_k^{(t)}$ is the derivative of $L_2(\xi_k)$. Calculating $\gamma_k^{(t)}$ is not easy because $L_2(\boldsymbol{\xi})$ is nondifferentiable at some points. Inspired by [32], we smooth $L_2(\boldsymbol{\xi})$ through entropy penalty function as

$$L_2^{\mathrm{smooth}}(\boldsymbol{\xi})=\frac{1}{2}\max(0,\boldsymbol{\xi}^2-\tau^2)+$$
$$\frac{1}{2p}\ln\left(1+\exp(-p|\boldsymbol{\xi}^2-\tau^2|)\right)$$

where p is a smooth parameter. When $p\to+\infty$, $L_2^{\mathrm{smooth}}\to L_2$. In practice, we can set p as a sufficiently large number, such as $p=10^4$. By using L_2^{smooth}, $L_\tau^{\mathrm{smooth}}(\boldsymbol{\xi})=L_{\mathrm{sq}}(\boldsymbol{\xi})-L_2^{\mathrm{smooth}}(\boldsymbol{\xi})$. Fig. 1 shows L_τ^{smooth} with $p=10$.

Using $L_2^{\mathrm{smooth}}(\xi_k)$, $\gamma_k^{(t)}$ can be computed as

$$\gamma_k^{(t)}=\frac{\xi_k^{(t)}\min\{1,\exp(p({\xi_k^{(t)}}^2-\tau^2))\}}{1+\exp(-p|{\xi_k^{(t)}}^2-\tau^2|)}. \tag{17}$$

Thus, the solution of R-LSPTSVM1 can be obtained by solving the following linear equations:

$$\left(\boldsymbol{Q}_1+\frac{c_1}{m_2}\boldsymbol{R}^{\mathrm{T}}\boldsymbol{R}\right)\boldsymbol{\alpha}=\frac{c_1}{m_2}\boldsymbol{R}^{\mathrm{T}}(\boldsymbol{e}_2-\gamma^{(t)}) \tag{18}$$

where $\boldsymbol{R}=\boldsymbol{K}_2^{\mathrm{T}}-\dfrac{1}{m_1}\boldsymbol{e}_2\boldsymbol{e}_1^{\mathrm{T}}\boldsymbol{K}_1^{\mathrm{T}}$.

The coefficient matrix in (18) may be low-rank if the kernel matrix is low-rank, which may make the solution of (11) sparse.

3.2.3 Sparse solution for R-LSPTSVM

In this subsection, we seek the sparse solution of (11) by the low-rank approximation of the kernel matrix.

Adopt the pivoting Cholesky decomposition method [14], and the kernel matrix can be decomposed as $\boldsymbol{K}=\boldsymbol{P}\boldsymbol{P}^{\mathrm{T}}$, where $\boldsymbol{P}\in\mathbf{R}^{m\times r}$, and r denotes the number of elements in the work set B. Let $\boldsymbol{P}=[\boldsymbol{P}_1^{\mathrm{T}},\boldsymbol{P}_2^{\mathrm{T}}]^{\mathrm{T}}$, then $\boldsymbol{K}_1^{\mathrm{T}}=\boldsymbol{P}_1\boldsymbol{P}^{\mathrm{T}}$, $\boldsymbol{K}_2^{\mathrm{T}}=\boldsymbol{P}_2\boldsymbol{P}^{\mathrm{T}}$, where $\boldsymbol{P}_1\in\mathbf{R}^{m_1\times r}$ and $\boldsymbol{P}_2\in\mathbf{R}^{m_2\times r}$.

Therefore, linear equation (18) can be rewritten as

$$\left(\boldsymbol{P}_1^{\mathrm{T}}\boldsymbol{P}_1+c_3\boldsymbol{I}_r-\frac{1}{m_1}\boldsymbol{P}_1^{\mathrm{T}}\boldsymbol{e}_1\boldsymbol{e}_1^{\mathrm{T}}\boldsymbol{P}_1+\frac{c_1}{m_2}\boldsymbol{J}_1^{\mathrm{T}}\boldsymbol{J}_1\right)\cdot$$
$$\boldsymbol{P}^{\mathrm{T}}\alpha=\frac{c_1}{m_2}\boldsymbol{J}_1^{\mathrm{T}}(\boldsymbol{e}_2-\gamma^{(t)}) \tag{19}$$

where $\boldsymbol{J}_1=\boldsymbol{P}_2-\dfrac{1}{m_1}\boldsymbol{e}_2\boldsymbol{e}_1^{\mathrm{T}}\boldsymbol{P}_1$ is an identity matrix. Furthermore, by permuting rows of matrix $\boldsymbol{P}$, we get $\boldsymbol{P}=[\boldsymbol{P}_B^{\mathrm{T}},\boldsymbol{P}_N^{\mathrm{T}}]^{\mathrm{T}}$, where $\boldsymbol{P_B}$ is full-rank and lower triangular, $\boldsymbol{P_N}$ consists of the remaining rows of $\boldsymbol{P}$. Correspondingly, $\boldsymbol{\alpha}$ can be rewritten as $\boldsymbol{\alpha}=[\boldsymbol{\alpha}_B^{\mathrm{T}},\boldsymbol{\alpha}_N^{\mathrm{T}}]^{\mathrm{T}}$, thus the sparse solution of (11) can be obtained by iteratively computing

$$\begin{cases}\boldsymbol{\alpha}_B^{(t+1)}=\dfrac{c_1}{m_2}(\boldsymbol{P}_B^{\mathrm{T}})^{-1}\boldsymbol{T}_1{}^{-1}\boldsymbol{J}_1^{\mathrm{T}}(\boldsymbol{e}_2-\gamma^{(t)})\\ \boldsymbol{\alpha}_N^{(t+1)}=0\end{cases} \tag{20}$$

where $\boldsymbol{T}_1=\boldsymbol{P}_1^{\mathrm{T}}\boldsymbol{P}_1+c_3\boldsymbol{I}_r-\dfrac{1}{m_1}\boldsymbol{P}_1^{\mathrm{T}}\boldsymbol{e}_1\boldsymbol{e}_1^{\mathrm{T}}\boldsymbol{P}_1+\dfrac{c_1}{m_2}\boldsymbol{J}_1^{\mathrm{T}}\boldsymbol{J}_1$.

Similarly, we can get the sparse solution of (12) by iteratively computing

$$\begin{cases}\boldsymbol{\beta}_B^{(t+1)}=-\dfrac{c_2}{m_1}(\boldsymbol{P}_B^{\mathrm{T}})^{-1}\boldsymbol{T}_2^{-1}\boldsymbol{J}_2^{\mathrm{T}}(\boldsymbol{e}_1-\gamma^{(t)})\\ \boldsymbol{\beta}_N^{(t+1)}=0\end{cases}\tag{21}$$

where

$$\boldsymbol{T}_2=\boldsymbol{P}_2^{\mathrm{T}}\boldsymbol{P}_2+c_4\boldsymbol{I}_r-\frac{1}{m_2}\boldsymbol{P}_2^{\mathrm{T}}\boldsymbol{e}_2\boldsymbol{e}_2^{\mathrm{T}}\boldsymbol{P}_2+\frac{c_2}{m_1}\boldsymbol{J}_2^{\mathrm{T}}\boldsymbol{J}_2,$$

$$\boldsymbol{J}_2=\boldsymbol{P}_1-\frac{1}{m_2}\boldsymbol{e}_1\boldsymbol{e}_2^{\mathrm{T}}\boldsymbol{P}_2.$$

3.2.4 SR-LSPTSVM algorithm

Based on the above analysis, SR-LSTPSVM is listed as Algorithm 1.

```
Algorithm 1   SR-LSPTSVM: sparse algorithm for R-LSPTSVM
Input: Training set $\boldsymbol{T}=\{(\boldsymbol{x}_s^{(i)},y^{(i)})|i=1,2;s=1,2,\cdots,m_i\}$, parameters $c_1,c_2,c_3,c_4>0$, the stop criterion $\delta>0$, $r=|\boldsymbol{B}|$.
Output: $\alpha_B$ and $\boldsymbol{\beta}_B$.
1. Find $\boldsymbol{P}$ and $\boldsymbol{B}$ such that $\boldsymbol{K}\approx\boldsymbol{P}\boldsymbol{P}^{\mathrm{T}}$ holds, let $\boldsymbol{\gamma}_1^{(0)}=\boldsymbol{0}\in\mathbf{R}^{m_1}$, $\boldsymbol{\gamma}_2^{(0)}=\boldsymbol{0}\in\mathbf{R}^{m_2}$.
2. Compute $\boldsymbol{T}_1$ and $\boldsymbol{T}_2$, let $t=0$.
3. Update $\boldsymbol{\alpha}_B^{(t+1)}$ and $\boldsymbol{\beta}_B^{(t+1)}$ according to (20) and (21), iteratively.
4. Let $\boldsymbol{\xi}^{(t+1)}=\boldsymbol{e}_2-\boldsymbol{J}_1\boldsymbol{P}_B^{\mathrm{T}}\boldsymbol{\alpha}_B^{(t+1)}$, $\boldsymbol{\eta}^{(t+1)}=\boldsymbol{e}_1+\boldsymbol{J}_2\boldsymbol{P}_B^{\mathrm{T}}\boldsymbol{\beta}_B^{(t+1)}$, compute $\boldsymbol{\gamma}_1^{(t+1)}$ and $\boldsymbol{\gamma}_2^{(t+1)}$ according to (17).
5. If $\|\boldsymbol{\gamma}_1^{(t+1)}-\boldsymbol{\gamma}_1^{(t)}\|<\delta$ or $\|\boldsymbol{\gamma}_2^{(t+1)}-\boldsymbol{\gamma}_2^{(t)}\|<\delta$, then
6.     stop with $\alpha_B=\boldsymbol{\alpha}_B^{(t+1)}$ and $\beta_B=\boldsymbol{\beta}_B^{(t+1)}$.
7. else
8.     $t=t+1$, go to Step 3.
9. end if
```

After obtaining the optimal $\boldsymbol{\alpha}_B$ and $\boldsymbol{\beta}_B$ by Algorithm 1, the label of a test sample $\boldsymbol{x}^{\text{new}}$ is

$$\text{label}(\boldsymbol{x}^{\text{new}})=\arg\min_{i=1,2}\{\boldsymbol{d}_i\}$$

where

$$\boldsymbol{d}_1=\boldsymbol{K}(\boldsymbol{x}^{\text{new}},\boldsymbol{x}_B)\boldsymbol{\alpha}_B-b_1,$$

$$\boldsymbol{d}_2=\boldsymbol{K}(\boldsymbol{x}^{\text{new}},\boldsymbol{x}_B)\boldsymbol{\beta}_B-b_2,$$

$$b_1=\frac{1}{m_1}\boldsymbol{e}_1^{\mathrm{T}}\boldsymbol{K}(\boldsymbol{x}^{(1)},\boldsymbol{x}_B)\boldsymbol{\alpha}_B,$$

$$b_2=\frac{1}{m_2}\boldsymbol{e}_2^{\mathrm{T}}\boldsymbol{K}(\boldsymbol{x}^{(2)},\boldsymbol{x}_B)\boldsymbol{\beta}_B.$$

Compared with the recursive algorithm of LSPTSVM in [10,11], which needs to update all samples many times to get projection axes, Algorithm 1 only needs to calculate the low rank approximation of the kernel matrix to update α_B, β_B. Therefore, our algorithm is more efficient than the LSPTSVM algorithm.

3.2.5 Convergence analysis

Based on the convergence of DC programming [33], we have the following theorem.

Theorem 2 Assume error variable $\boldsymbol{\xi}=\boldsymbol{e}_2-\boldsymbol{J}_1\boldsymbol{P}_B^{\mathrm{T}}\boldsymbol{\alpha}_B$ is bounded for all training samples $x^{(1)}$ with selected parameter $\boldsymbol{w}_1$, then limit point of sequence $\boldsymbol{\alpha}^{(1)}$ is the generalized Karush-Kuhn-Tucker (KKT) point of the optimization problem in (11).

Similar conclusion can be obtained for (12). Therefore, Algorithm 1 is convergent.

3.2.6 Complexity analysis

For SR-LSPTSVM, the computation cost of Step 1 and Step 2 are both $O(mr^2)$ $(r\ll m)$. The complexity of iteratively calculating Step 3 is $O(Nmr)$, where N is the number of iterations. Then, the overall computational complexity of SR-LSPTSVM is $O(mr^2+Nmr)$. In contrast, the computational complexity of nonlinear LSPTSVM [11] is $O(m^3)$. Therefore, our SR-LSPTSVM algorithm has lower computational complexity than the existing approach.

3.3 Robustness of R-LSPTSVM

As we know, weighting is a common method to improve robustness of the model. In this subsection, we first propose weighted LSPTSVM (W-LSPTSVM) for binary classifications, and then explain the robustness of R-LSPTSVM by analyzing the relationship between W-LSPTSVM and R-LSPTSVM.

3.3.1 W-LSPTSVM

By introducing weights in the model, W-LSPTSVM can be written as

$$\begin{gathered}\min_{w_1,\xi_k}\ \frac{1}{2}\boldsymbol{w}_1^{\mathrm{T}}\boldsymbol{S}_1\boldsymbol{w}_1+\frac{c_1}{m_2}\sum_{k=1}^{m_2}\frac{1}{2}\rho_k\xi_k^2+\frac{c_3}{2}\|\boldsymbol{w}_1\|^2\\ \text{s.t.}\begin{cases}\boldsymbol{w}_1^{\mathrm{T}}\boldsymbol{x}_k^{(2)}-\bar{\boldsymbol{A}}\boldsymbol{w}_1+\xi_k=1\\ k=1,2,\cdots,m_2\end{cases},\end{gathered}\tag{22}$$

$$\begin{gathered}\min_{w_2,\eta_k}\ \frac{1}{2}\boldsymbol{w}_2^{\mathrm{T}}\boldsymbol{S}_2\boldsymbol{w}_2+\frac{c_2}{m_1}\sum_{k=1}^{m_1}\frac{1}{2}v_k\eta_k^2+\frac{c_4}{2}\|\boldsymbol{w}_2\|^2\\ \text{s.t.}\begin{cases}-(\boldsymbol{w}_2^{\mathrm{T}}\boldsymbol{x}_k^{(1)}-\bar{\boldsymbol{B}}\boldsymbol{w}_2)+\eta_k=1\\ k=1,2,\cdots,m_1\end{cases},\end{gathered}\tag{23}$$

where $\rho_k\geqslant0$ and $v_k\geqslant0$ are the weights. When $\rho_k=1$ and $v_k=1$, (22) and (23) are equivalent to LSTPSVM models in (5) and in (6) respectively. As ρ_k and v_k decrease, the influence of samples on the model decreases. Therefore, we can set smaller weights for outliers to reduce their effect on the model.

The optimal weights ρ_k and v_k are selected based on the error variables ξ_k and η_k, respectively. According to the method in [19], we can set the weight ρ_k by the following formula:

$$\rho_k=\begin{cases}1, & |\xi_k/\hat{s}|<c_1\\ \dfrac{c_2-|\xi_k/\hat{s}|}{c_2-c_1}, & c_1\leqslant|\xi_k/\hat{s}|\leqslant c_2\\ 10^{-4}, & \text{otherwise}\end{cases}$$

where $\hat{s}$ is a robust estimate of the standard deviation of error variables ξ_k $(k=1,2,\cdots,m_1)$. $\hat{s}$ is calculated as follows:

$$\hat{s}=\frac{\mathrm{IQR}}{2\times 0.674\,5}$$

where IQR is the interquartile difference of ξ_k, i.e., the difference between the 75th and the 25th percentiles of ξ_k $(k=1,2,\cdots,m_1)$. The weight v_k in (23) can be obtained similarly. The calculation complexity of finding the optimal weight of W- LSPTSVM is $O(m)$.

3.3.2 Robustness of R-LSPTSVM

In this subsection, we prove that R-LSPTSVM is equivalent to W-LSPTSVM, hence the robustness of R-LSPTSVM is proved.

For convenience, we rewrite W-LSPTSVM in primal space as follows:

$$\min_{\alpha}\ \frac{1}{2}\boldsymbol{\alpha}^{\mathrm{T}}\boldsymbol{Q}_1\boldsymbol{\alpha}+\frac{c_1}{m_2}\sum_{k=1}^{m_2}\frac{1}{2}\rho_k\xi_k^2, \tag{24}$$

$$\min_{\beta}\ \frac{1}{2}\boldsymbol{\beta}^{\mathrm{T}}\boldsymbol{Q}_2\boldsymbol{\beta}+\frac{c_2}{m_1}\sum_{k=1}^{m_1}\frac{1}{2}v_k\eta_k^2, \tag{25}$$

where ξ_k and η_k are calculated by (13) and (14), respectively.

We take R-LSPTSVM1 model in (11) as an example to analyze the relationship between W-LSPTSVM and R-LSPTSVM. Consider the following two models:

$$\min_{\alpha}\min_{0\leqslant\rho\leqslant1}\ \frac{1}{2}\boldsymbol{\alpha}^{\mathrm{T}}\boldsymbol{Q}_1\boldsymbol{\alpha}+\frac{c_1}{m_2}\sum_{k=1}^{m_2}L_\rho(\rho_k,\xi_k), \tag{26}$$

$$\min_{\alpha}\ \frac{1}{2}\boldsymbol{\alpha}^{\mathrm{T}}\boldsymbol{Q}_1\boldsymbol{\alpha}+\frac{c_1}{m_2}\sum_{k=1}^{m_2}L_\tau(\xi_k), \tag{27}$$

where $L_\rho(\rho_k,\xi_k)=\dfrac{1}{2}\rho_k\xi_k^2+\tau^2(1-\rho_k)/2$. In (26), if the value of ρ_k is fixed, the term $\tau^2(1-\rho_k)/2$ can be ignored. Therefore, the optimization problem in (26) is equivalent to that in (24). Inspired by [23], we have the following lemma.

Lemma 1 $L_\tau(\xi_k)=\min\{\tau^2,\xi^2\}/2$ can be reformulated as

$$L_\tau(\xi)=\min_{\rho_k\in R_+}\ \frac{1}{2}\rho_k\xi^2+\phi(\rho_k) \tag{28}$$

where $\phi(\rho_k)=\tau^2(1-\rho_k)_+/2$.

Lemma 1 indicates that $\min\limits_{0\leqslant\rho\leqslant1}L_\rho(\rho_k,\xi_k)=L_\tau(\xi_k)$, then we have the following theorem.

Theorem 3 The model in (26) is equivalent to the model (27), i.e.,

$$\min_{\alpha}\min_{0\leqslant\rho\leqslant1}\ \frac{1}{2}\boldsymbol{\alpha}^{\mathrm{T}}\boldsymbol{Q}_1\boldsymbol{\alpha}+\frac{c_1}{m_2}\sum_{k=1}^{m_2}L_\rho(\rho_k,\xi_k)=$$
$$\min_{\alpha}\ \frac{1}{2}\boldsymbol{\alpha}^{\mathrm{T}}\boldsymbol{Q}_1\boldsymbol{\alpha}+\frac{c_1}{m_2}\sum_{k=1}^{m_2}L_\tau(\xi_k). \tag{29}$$

Proposition 1 Any critical point of R-LSPTSVM in (11) and (12) can be obtained by iteratively solving

$$\min_{\alpha}\ \frac{1}{2}\boldsymbol{\alpha}^{\mathrm{T}}\boldsymbol{Q}_1\boldsymbol{\alpha}+\frac{c_1}{m_2}\sum_{k=1}^{m_2}\frac{1}{2}\rho_k^{(t)}\xi_k^2, \tag{30}$$

$$\min_{\beta}\ \frac{1}{2}\boldsymbol{\beta}^{\mathrm{T}}\boldsymbol{Q}_2\boldsymbol{\beta}+\frac{c_2}{m_1}\sum_{k=1}^{m_1}\frac{1}{2}v_k^{(t)}\eta_k^2, \tag{31}$$

where $\rho_k^{(t)}$ and $v_k^{(t)}$ are the values of the tth iteration of the weight ρ_k and v_k, respectively.

Proof Here, we only discuss the proof for R-LSPTSVM1, and R-LSPTSVM2 can be proved similarly.

Substituting (28) into (11), we have

$$\min_{\alpha\in R^{m_1},\rho\in R_+^{m_1}}\ \Phi(\boldsymbol{\alpha},\boldsymbol{\rho}):=\frac{1}{2}\boldsymbol{\alpha}^{\mathrm{T}}\boldsymbol{Q}_1\boldsymbol{\alpha}+$$
$$\frac{c_1}{m_2}\sum_{k=1}^{m_2}\frac{1}{2}\rho_k\xi_k^2+\frac{c_1}{m_2}\sum_{k=1}^{m_2}\phi(\rho_k). \tag{32}$$

Because $\Phi(\boldsymbol{\alpha},\boldsymbol{\rho})$ is non-convex, there may be multiple local minima for (32). We only consider one of them. Let $(\boldsymbol{\alpha}^*,\boldsymbol{\rho}^*)$ be one of the critical points of (11). By Theorem 3, there exists $\boldsymbol{\rho}^*\in\arg\min\limits_{\rho\in\mathbf{R}_+^{m_1}}\Phi(\boldsymbol{\alpha}^*,\boldsymbol{\rho})$ such that $(\boldsymbol{\alpha}^*,\boldsymbol{\rho}^*)$ is the solution of (32). On the other hand, if $(\boldsymbol{\alpha}^*,\boldsymbol{\rho}^*)$ is any stationary point of (32), then $\boldsymbol{\alpha}^*\in\arg\min\limits_{\alpha\in\mathbf{R}^{m_1}}\Phi(\boldsymbol{\alpha},\boldsymbol{\rho}^*)$ also solves (11).

Hence, we can iteratively solve (32) by alternating direction method (ADM) [33] as follows:

$$\boldsymbol{\alpha}^{(t)}\in\arg\min_{\alpha\in\mathbf{R}^{m_1}}\ \Phi(\boldsymbol{\alpha},\boldsymbol{\rho}^{(t-1)}), \tag{33}$$

$$\boldsymbol{\rho}^{(t)}\in\arg\min_{\rho\in\mathbf{R}_+^{m_1}}\ \Phi(\boldsymbol{\alpha}^{(t)},\boldsymbol{\rho}). \tag{34}$$

Obviously, the optimization problem in (34) has the closed form solution. The optimization problem in (33) is just the standard weighted LSPTSVM after removing the constant term. Therefore, the solution of R-LSPTSVM can be obtained by iteratively solving (30) and (31). □

Proposition 1 indicates that the model of R-LSPTSVM (11) and (12) are equivalent to that of W-LSPTSVM (24) and (25) respectively. Therefore, the robustness of R-LSPTSVM is proved.

4. Experimental results

In order to verify the effectiveness of the proposed R-LSPTSVM and SR-LSPTSVM, we compare classification accuracy and computing efficiency between our proposed algorithms and four state-of-the-art algorithms, including RPTSVM, LSPTSVM, FLSPTSVM, and SR-LSSVM.

(i) RPTSVM: Regularized PTSVM [34] improves PTSVM by adding a maximum margin regularization term, and expands PTSVM to the nonlinear kernel.

(ii) LSPTSVM: LSPTSVM is introduced in Subsection 2.2.

(iii) FLSPTSVM: Feature selection for LSPTSVM [35] uses 1-norm regularization to achieve feature selection.

(iv) SR-LSSVM: SR-LSSVM [23] obtains a sparse solution to R-LSSVM by the Cholesky decomposition of the kernel matrix.

In SR-LSPTSVM and SR-LSSVM, we fix the values of the smooth parameter $p = 10^4$, the stop criterion $\varepsilon = 10^{-2}$ and the truncated parameter $\tau = 1$. We adopt Gaussian kernel function $K(x_i, x_j) = \exp\left(-\sigma\|x_i - x_j\|^2\right)$ in the experiments, where $\sigma \in \{2^{-8}, 2^{-7}, \cdots, 2^{-4}\}$ is the kernel parameter. Other parameters, such as c_1, c_2, c_3, c_4, v_1, v_2, ε, and λ, are selected from the set $\{10^{-1}, 10^0, \cdots, 10^5\}$. We use five-fold cross-validation procedure and grid search to obtain optimal parameters. In the experiments, we set the trade-off parameters $c_1 = c_2$ and the regularization parameters $c_3 = c_4$, because they play the same role in (9) and (10).

All experiments are implemented in Matlab R2017a and run on a 2.40 GHz processor with 8 GB RAM.

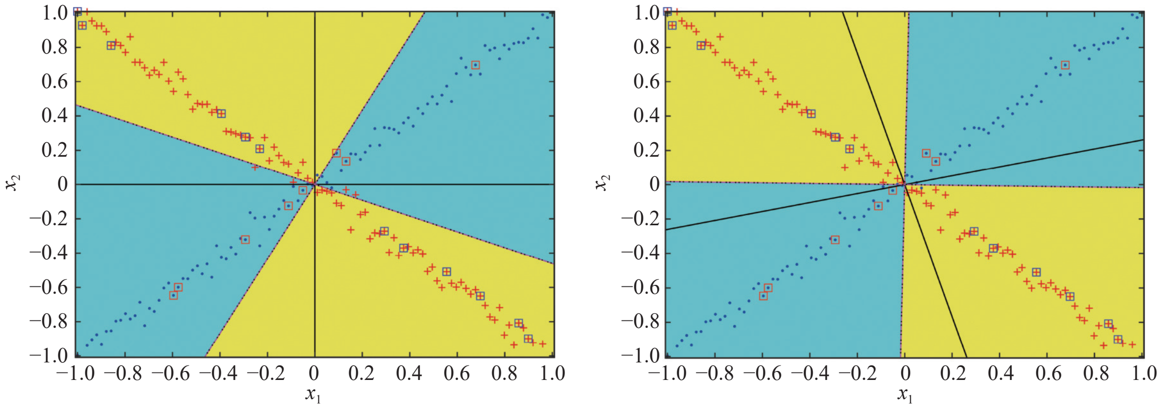
4.1 Experiments on binary classification datasets

In this section, we verify the performance of the proposed algorithms on a synthetic dataset and several real-world benchmark datasets.

4.1.1 Experiments on synthetic dataset

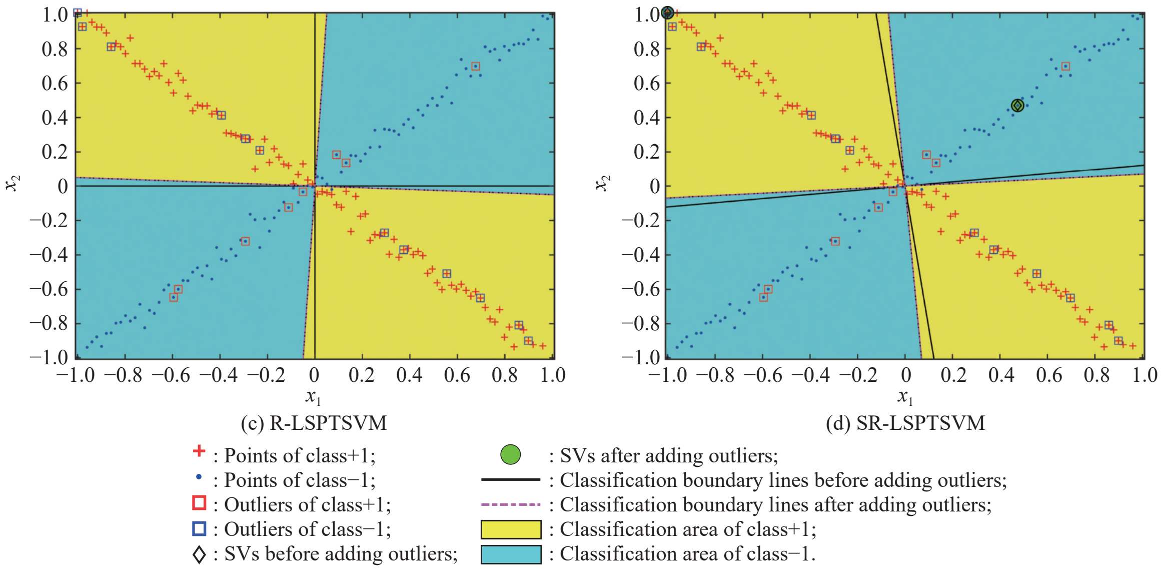
To compare the robustness of four algorithms (LSPTSVM, FLSPTSVM, R-LSPTSVM and SR-LSPTSVM), we experiment on a 2D-"Cross Planes " dataset [36] is generated perturbing points lying on two intersecting lines. The dataset includes 200 training samples and 600 test samples. To simulate outliers, we randomly choose 10% of samples and flip their labels. The numbers of the outliers of positive and negative classes are equal.

Experimental results and the dataset are shown in Fig. 2. The horizontal and vertical coordinates of the data points are both within the interval [−1,1]. It can be seen from Fig. 2 that the classification boundary lines of R-LSPTSVM and SR-LSPTSVM are almost unchanged before and after adding outliers. In comparison, the classification boundary lines of LSPTSVM and FLSPTSVM change greatly after adding outliers. Therefore, R-LSPTSVM and SR-LSPTSVM are insensitive to outliers. As for the accuracy, LSPTSVM, FLSPTSVM, R-LSPTSVM, and SR-LSPTSVM has the accuracy of 98.17%, 97.83%, 98.17%, and 98.17% without outliers, and 90.17%, 97.67%, 98.17%, and 98.17% with outliers. Therefore, R-LSPTSVM and SR-LSPTSVM own the highest accuracy among the comparison algorithms, and their accuracy is stable before and after adding outliers. The accuracy of FLSPTSVM before adding outliers (97.83%) is the lowest, and after adding outliers, the test accuracy of LSPTSVM (90.17%) is the lowest.



(a) LSPTSVM

(b) FLSPTSVM



(c) R-LSPTSVM

(d) SR-LSPTSVM

Fig. 2 Comparison of four algorithms

In terms of the sparsity, the numbers of support vectors are only two for SR-LSPTSVM before and after adding outliers, while almost all of the training samples are support vectors for LSPTSVM, FLSPTSVM, and R-LSPTSVM. Therefore, SR-LSPTSVM is robust and sparse.

4.1.2 Experiments on real-world datasets

In order to further investigate the efficiency of the proposed algorithms, we apply algorithms to eight real-world datasets from University of California Irvine (UCI) machine learning repository [37]. All attributes of the datasets are normalized to [−1, 1]. We randomly choose 10% of samples and flip their labels to simulate outliers. The datasets information and experimental results are shown in Table 1. Satimage, United States Postal Service (USPS), and Shuttle are multi-class datasets. We only classify two classes of samples for them. The detailed information is listed as follows. The best results are marked in bold. “—” represents that the running result is not obtained, because computer memory is insufficient due to much memory consumed by computing kernel matrix.

(i) Satimage: It is comprised by six classes. The task of this experiments is to classify Class 1 versus Class 6.

(ii) USPS: It is a multi-class dataset with 10 classes. A binary classification is to separate Class 1 from Class 2.

(iii) Shuttle: It is a multi-class dataset with seven classes. We only classify Class 4 versus Class 5 here.

Experiments are repeated for ten times, and the accuracy refers to the mean of ten times testing results. For SR-LSPTSVM and SR-LSSVM, we set $r = |\boldsymbol{B}| = 0.05$ m. Table 1 gives the experimental results of eight real datasets before and after adding 10% outliers. In Table 1, the average values and the standard deviations of experimental results are given outside and inside the brackets, respectively. m and l are the numbers of training and testing samples respectively, n is the number of features. ‘nSVs’ is the average number of support vectors.

Table 1 Comparison of different algorithms on UCI datasets before and after adding 10% outliers

Data	Algorithm	0% outliers			10% outliers		
		nSVs	Time/s	Accuracy/%	nSVs	Time/s	Accuracy/%
	RPTSVM	95(0)	0.02(0.01)	**90.32**(0)	69.7(14.5)	0.03(0.01)	85.81(0.04)
Hepatitis	LSPTSVM	124(0)	0.02(0.01)	89.25(0.03)	124(0)	0.01(0.01)	89.03(0.03)
$m = 124$	FLSPTSVM	22.6(19.9)	0.07(0.01)	86.45(0.04)	26.7(42.7)	0.07(0.01)	80.64(0.09)
$l = 31$	SR-LSSVM	**2.4**(2.6)	**0.00**(0.00)	**90.32**(0)	4.8(2.7)	**0.00**(0.00)	89.03(0.02)
$n = 19$	R-LSPTSVM	62(56.8)	0.03(0.01)	**90.32**(0)	82(32.1)	0.03(0.03)	90.32(0.04)
	SR-LSPTSVM	4.8(2.7)	**0.00**(0.00)	**90.32**(0)	**1.3**(1.8)	0.01(0.02)	**90.97**(0.01)
	RPTSVM	173.2(11.6)	0.05(0.02)	70.82(0.02)	162.5(8.4)	0.04(0.02)	70.00(0.02)
Haberman	LSPTSVM	245(0)	0.02(0.01)	71.80(0.02)	245(0)	0.02(0.01)	71.47(0.04)
$m = 245$	FLSPTSVM	35(5.5)	0.18(0.08)	70.82(0.03)	72.6(21.0)	0.17(0.07)	70.00(0.02)
$l = 61$	SR-LSSVM	**7.2**(6.6)	**0.01**(0.00)	72.13(0)	8.6(5.0)	**0.00**(0.00)	72.13(0)
$n = 3$	R-LSPTSVM	166.6(28.6)	0.11(0.01)	**73.77**(0)	89.1(79.0)	0.03(0.03)	**73.11**(0.02)
	SR-LSPTSVM	10(0)	**0.01**(0.00)	73.44(0.01)	**4.3**(3.1)	0.01(0.00)	72.62(0.01)

Continued

Data	Algorithm	0% outliers			10% outliers		
		nSVs	Time/s	Accuracy/%	nSVs	Time/s	Accuracy/%
	RPTSVM	123.2(86.4)	0.10(0.01)	97.06(0.01)	269.7(67.4)	0.10(0.01)	94.19(0.01)
Breast	LSPTSVM	547(0)	0.22(0.00)	97.79(0.01)	547(0)	0.23(0.10)	97.50(0.01)
$m = 547$	FLSPTSVM	62(0)	0.64(0.05)	**98.53**(0)	**3.6**(4.8)	0.64(0.56)	97.94(0.01)
$l = 136$	SR-LSSVM	27(0)	0.01(0.00)	**98.53**(0)	27(0)	**0.00**(0.00)	97.20(0.01)
$n = 10$	R-LSPTSVM	197(0)	0.07(0.00)	**98.53**(0)	226.5(129.0)	0.16(0.09)	**98.53**(0.01)
	SR-LSPTSVM	**8**(0)	**0.01**(0.00)	97.79(0)	21.7(8.6)	0.01(0.02)	97.72(0.00)
	RPTSVM	**47.4**(5.0)	3.60(0.11)	99.89(0)	1 053.8(164.0)	3.03(1.30)	99.68(0.00)
Satimage	LSPTSVM	2 110(0)	0.98 (0.06)	99.81(0.00)	2 110(0)	1.10(0.18)	99.74(0.00)
$m = 2\,110$	FLSPTSVM	59.4(54.3)	20.43(1.75)	99.76(0.00)	115.4(201.1)	21.37(1.84)	99.70(0.00)
$l = 931$	SR-LSSVM	105(0)	**0.06**(0.01)	99.77(0.00)	**105**(0)	0.14(0.08)	99.63(0.00)
$n = 36$	R-LSPTSVM	1 073.4(54.3)	1.23(0.05)	**99.92**(0.00)	1 887.6(664.6)	1.67(0.20)	**99.79**(0.00)
	SR-LSPTSVM	105(0)	0.07(0.00)	99.77(0.00)	111(11.3)	**0.08**(0.03)	99.64(0.00)
	RPTSVM	**14**(0)	4.06(0.84)	99.36(0)	681.9(208.6)	4.25(0.89)	98.65(0.01)
USPS	LSPTSVM	2 199(0)	3.17(0.60)	**99.52**(0)	2 199(0)	3.52(0.75)	98.74(0.01)
$m = 2\,199$	FLSPTSVM	59.5(16.3)	26.79(0.05)	99.44(0.00)	**34.4**(51.4)	20.43(0.40)	98.62(0.01)
$l = 623$	SR-LSSVM	109(0)	1.01(0.03)	**99.52**(0)	109(0)	0.48(0.06)	99.12(0.00)
$n = 256$	R-LSPTSVM	1 044(0)	11.47(0.22)	99.36(0)	907.7(1 130.2)	2.08(1.64)	98.65(0.00)
	SR-LSPTSVM	109(0)	**0.84**(0.10)	99.20(0)	65.2(35.8)	**0.33**(0.03)	**99.15**(0.01)
	RPTSVM	1 219(1 707.0)	88.65(0.35)	**100**(0)	3 157.0(2 788.0)	114.67(23.44)	99.88(0.00)
Shuttle	LSPTSVM	9 206(0)	39.25(0.35)	**100**(0)	9 206(0)	39.39 (0.50)	99.87(0.00)
$m = 9\,206$	FLSPTSVM	152.4(65.2)	1 796.35(0.57)	99.95(0.00)	182.3(70.4)	1 821.20(40.05)	99.87(0.01)
$l = 2\,964$	SR-LSSVM	**50.2**(5.0)	**0.04**(0.00)	99.97(0)	126.8(84.3)	0.33 (0.31)	99.89(0.00)
$n = 9$	R-LSPTSVM	4 596(0)	48.25(1.17)	99.93(0)	4 002.3(687.5)	36.42(0.24)	99.91(0.00)
	SR-LSPTSVM	138.2(34.1)	0.32(0.09)	**100**(0)	**88.2**(12.0)	**0.09** (0.02)	**99.93**(0)
	RPTSVM	—	—	—	—	—	—
Ijcnn1	LSPTSVM	—	—	—	—	—	—
$m = 35\,000$	FLSPTSVM	—	—	—	—	—	—
$l = 91\,701$	SR-LSSVM	**1 750**(0)	28.28(1.10)	**90.50**(0)	**1 750**(0)	28.68(1.67)	**90.50**(0)
$n = 22$	R-LSPTSVM	—	—	—	—	—	—
	SR-LSPTSVM	**1 750**(0)	**26.78**(3.33)	**90.50**(0)	**1 750**(0)	**26.39**(0.23)	**90.50**(0)
Skin-nonskin	RPTSVM	—	—	—	—	—	—
	LSPTSVM	—	—	—	—	—	—
$m = 61\,265$	FLSPTSVM	—	—	—	—	—	—
$l = 183\,792$	SR-LSSVM	3 061(2.31)	128.39(0.86)	97.67(0.01)	3 063(0)	115.25(0.26)	94.49(0.01)
$n = 3$	R-LSPTSVM	—	—	—	—	—	—
	SR-LSPTSVM	**2 532.2**(21.4)	**115.23**(4.84)	**98.02**(0.00)	**2 661**(9.4)	**114.28**(0.60)	**95.17**(0)

Table 1 illustrates that the proposed algorithms have higher accuracy than other algorithms on all the datasets without outliers except USPS. After adding 10% outliers, the proposed algorithms have the best test accuracy on all datasets. Furthermore, the accuracy of our algorithms has little change before and after adding outliers on all datasets.

As for training efficiency, SR-LSPTSVM greatly reduces the training time of R-LSPTSVM. Compared with PTSVM based algorithms, such as RPTSVM, LSPTSVM, and FLSPTSVM, SR-LSPTSVM has faster training speed especially when the size of training set is larger than 1 000.

We also experiments on two large scale datasets IJCNN1 and Skin-nonskin, which have more than 10 000 training samples. Experimental results verify the advantage of sparse algorithms. The non-sparse algorithms RPTSVM, LSPTSVM, FLSPTSVM, and R-LSPTSVM cannot process these big-scale datasets, because computing full kernel matrix consumes much computer memory. In comparison, sparse algorithms SR-LSPTSVM and SR-LSSVM can achieve good classification results including accuracy and running time, and SR-LSPTSVM is slightly better than SR-LSSVM.

In terms of the number of support vectors (nSVs), Table 1 shows that 'nSVs' of SR-LSPTSVM is much less than LSPTSVM. 'nSVs' of FLSPTSVM is smaller than other algorithms on the Breast and USPS datasets, but its accuracy is lower than R-LSPTSVM (or SR-LSPTSVM).

4.1.3 Robustness comparison

In this subsection, we test the robustness of RPTSVM, LSPTSVM, FLSPTSVM, SR-LSSVM, and our proposed algorithms by adding 0%, 5%, 10%, 15%, and 20% outliers on Satimage dataset. The experimenal results are shown in Fig. 3. It shows that the classification accuracy of R-LSPTSVM is the highest among the compared algorithms, and it decreases less than other algorithms as the ratio of outliers increases. Therefore, R-LSPTSVM is more robust than other algorithms.

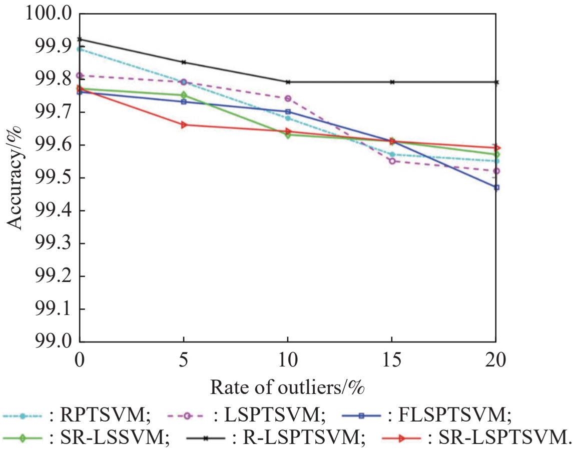
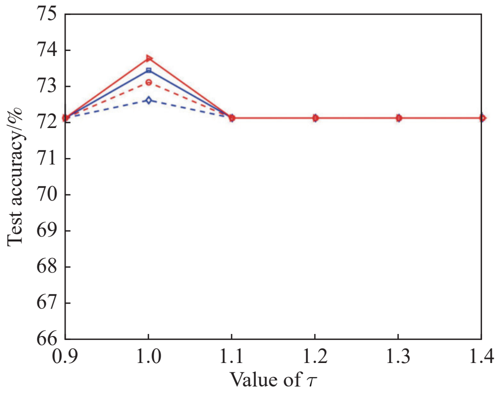


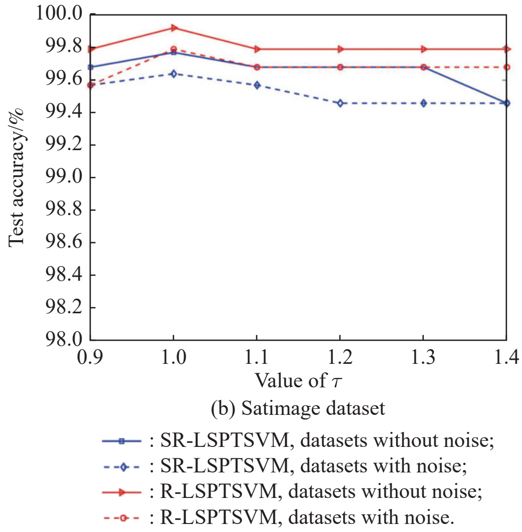
Fig. 3 Comparison of classification accuracy on Satimage dataset with different outliers

4.1.4 Parameter analysis

In order to discuss the optimal value of τ, we test the classification accuracy of R-LSPTSVM and SR-LSPTSVM on the Breast and Satimage datasets before and after adding 10% outliers. The value of τ is taken from the set $\{0.9, 1.0, 1.1, 1.2, 1.3, 1.4\}$. Fig. 4 gives the experimental results. Fig. 4 illustrates that the optimal τ values of R-LSPTSVM and SR-LSPTSVM on Satimage and Haberman datasets are 1 before and after adding outliers. Therefore, we set $\tau = 1$ in all of our experiments.



(a) Haberman dataset



(b) Satimage dataset

Fig. 4 Comparison of test accuracy on two datasets

4.2 Experiments on multi-classification datasets

Our model can also be extended to analyze multi-classification problems by using one versus one or one versus all techniques. In this subsection, we test the efficiency of our proposed algorithms on five multi-classification datasets. The information of the datasets is shown in Table 2.

Table 2 Comparison of different algorithms on multi-classification datasets

Data	Algorithm	nSVs	Time/s	Accuracy/%
Iris	RPTSVM	74.1(4.4)	0.02(0.00)	93.33(0)
$m = 120$	LSPTSVM	120(0)	0.01(0.00)	96.33(0.03)
$l = 30$	FLSPTSVM	8.8(1.7)	0.04(0.00)	92.33(0.02)
$n = 4$	R-LSPTSVM	79.1(3.7)	0.01(0.01)	**96.34**(0.02)
$k = 3$	SR-LSPTSVM	**6**(0)	**0.00**(0.00)	96.33(0.02)
Wine	RPTSVM	117.9(3.8)	0.04(0.00)	**100**(0)
$m = 143$	LSPTSVM	143(3.0)	0.02(0.00)	**100**(0)
$l = 35$	FLSPTSVM	33.0(3.5)	0.08(0.00)	98.86(0.01)
$n = 13$	R-LSPTSVM	119.1(2.8)	0.01(0.00)	**100**(0)
$k = 3$	SR-LSPTSVM	**18.3**(3.1)	**0.00**(0.00)	**100**(0)
Vehicle	RPTSVM	290.4(10.7)	0.54(0.06)	82.72(0.02)
$m = 677$	LSPTSVM	677(0)	0.38(0.01)	77.04(0.02)
$l = 169$	FLSPTSVM	49.8(3.7)	1.13(0.04)	70.60(0.02)
$n = 18$	R-LSPTSVM	579.5(12.3)	0.05(0.00)	**83.14**(0.01)
$k = 4$	SR-LSPTSVM	**33**(0)	**0.01**(0.01)	73.96(0.00)
Segment	RPTSVM	453.6(9.2)	2.98(0.05)	94.67(0.01)
$m = 1848$	LSPTSVM	1848(0)	10.8(0.16)	95.61(0.00)
$l = 462$	FLSPTSVM	137.4(3.5)	3.72(0.08)	95.26(0.00)
$n = 19$	R-LSPTSVM	1 828.7(18.0)	0.29(0.01)	**96.23**(0.01)
$k = 7$	SR-LSPTSVM	**91.9**(0.3)	**0.09**(0.00)	93.77(0.00)
Pendigits	RPTSVM	846.6(24.2)	5.54(0.24)	98.03(0.00)
$m = 7494$	LSPTSVM	7 494(0)	18.59(5.55)	97.45(0.00)
$l = 3498$	FLSPTSVM	500.4(2.0)	137.34(4.80)	86.99(0.04)
$n = 16$	R-LSPTSVM	7 342.1(21.2)	2.95(0.06)	**98.08**(0.00)
$k = 10$	SR-LSPTSVM	**374**(0)	**1.11**(0.01)	98.05(0.00)

We use the one versus all techniques on multi-classification datasets. For datasets with k classification, one versus all techniques solves k subproblems. In the ith subproblem ($1 \leqslant i \leqslant k$), samples of class i are regarded as positive points, and the rest samples are regarded as negative points. By solving k optimization problems, we obtain k decision functions. The test samples are assigned to the class corresponding to the maximum value of the decision function.

Table 2 shows the experimental results of RPTSVM, LSPTSVM, FLSPTSVM, R-LSPTSVM, and SR-LSPTSVM on multi-classification datasets. It can be seen from Table 2 that the R-LSPTSVM algorithm has the highest accuracy on all datasets. Although the accuracy of SR-LSPTSVM is lower than that of R-LSPTSVM on some datasets, the 'nSVs' and training time of SR-LSPTSVM are the lowest on all datasets. This verifies the sparsity and efficiency of SR-LSPTSVM.

5. Conclusions

LSPTSVM gives good performance on many binary classification problems, but it is sensitive to outliers and its solution lacks sparsity. In this paper, we propose R-LSPTSVM to reduce the influence of outliers to the model. We interpret its robustness from a weighted perspective. In order to get the sparse solution of R-LSPTSVM, R-LSPTSVM is further rewritten in the primal space by representer theorem and its sparse solution is obtained by applying the pivoting Cholesky factorization technique. Finally, we propose SR-LSPTSVM algorithm and analyze its convergence and computational complexity. Experimental results indicate that the proposed algorithms have robustness. SR-LSPTSVM is a sparsity algorithm, and the training speed of it is faster than other comparison algorithms. Therefore, SR-LSPTSVM is a suitable option for dealing with large-scale classification. In the future, our methods can be extended to multi-view learning or regression.

References

[1] CORTES C, VAPNIK V. Support-vector networks. Machine Learning, 1995, 20(3): 273–297.

[2] BURGES C. A tutorial on support vector machines for pattern recognition. Data Mining and Knowledge Discovery, 1998, 2(6): 121–167.

[3] OSUNA E, FREUND R, GIROSI F. Training support vector machines: an application to face detection. Proc. of the IEEE Computer Society Conference on Computer Vision & Pattern Recognition, 2000: 130–136.

[4] TRAFALIS T B, INCE H. Support vector machine for regression and applications to financial forecasting. Proc. of the IEEE International Joint Conference on Neural Networks, 2000: 348–353.

[5] CALISIR D, DOGANTEKIN E. A new intelligent hepatitis diagnosis system: PCA-LSSVM. Expert Systems with Applications, 2011, 38(8): 10705–10708.

[6] ZHANG S J, YANG R, LIU S Y, et al. Coupled compressed sensing inspired sparse spatial-spectral LSSVM for hyperspectral image classification. Knowledge-Based Systems, 2015, 79(5): 80–89.

[7] KHEMCHANDANI R, CHANDRA S. Twin support vector machines for pattern classification. IEEE Trans. on Pattern Analysis and Machine Intelligence, 2007, 29(5): 905–910.

[8] KUMAR M A, GOPAL M. Application of smoothing technique on twin support vector machines. Pattern Recognition Letters, 2008, 29(13): 1842–1848.

[9] CHEN X B, YANG J, YE Q L, et al. Recursive projection twin support vector machine via within-class variance minimization. Pattern Recognition, 2011, 44(10/11): 2643–2655.

[10] SHAO Y H, DENG N Y, YANG Z M. Least squares recursive projection twin support vector machine for classification. Pattern Recognition, 2012, 45(6): 2299–2307.

[11] DING S F, HUA X P. Recursive least squares projection twin support vector machines for nonlinear classification. Neurocomputing, 2014, 130(4): 3–9.

[12] SUYKENS J, LUKAS L, VANDEWALLE J. Sparse approximation using least squares support vector machines. Proc. of the IEEE International Symposium on Circuits and Systems, 2002: 757–760.

[13] JIAO L C, BO L F, WANG L. Fast sparse approximation for least squares support vector machine. IEEE Trans. on Neural Networks, 2007, 18(3): 685–697.

[14] ZHOU S S. Sparse LSSVM in primal using cholesky factorization for large-scale problems. IEEE Trans. on Neural Networks & Learning Systems, 2016, 27(4): 783–795.

[15] ZHOU S S, LIU M. A new sparse LSSVM method based the revised LARS. Proc. of the IEEE International Conference on Machine Vision & Information Technology, 2017: 46–51.

[16] CHENG R J, SONG Y D, CHEN D W, et al. Intelligent localization of a high-speed train using LSSVM and the online sparse optimization approach. IEEE Trans. on Intelligent Transportation Systems, 2017, 18(8): 2071–2084.

[17] SUN B B, NG W W Y, YEUNG D S. Improved sparse LSSVMS based on the localized generalization error model. International Journal of Machine Learning and Cybernetics, 2017, 8(6): 1853–1861.

[18] MA Y F, LIANG X, SHENG G, et al. Noniterative sparse LS-SVM based on globally representative point selection. IEEE Trans. on Neural Networks and Learning Systems, 2021, 32(2): 788–798.

[19] SUYKENS J A K, BRABANTER J D, LUKAS L, et al. Weighted least squares support vector machines: robustness and sparse approximation. Neurocomputing, 2002, 48(1): 85–105.

[20] WEI L W, CHEN Z Y, LI J P, et al. Sparse and robust least squares support vector machine: a linear programming formulation. Proc. of the IEEE International Conference on Grey Systems and Intelligent Services, 2007: 1134–1138.

[21] WEN W, HAO Z F, YANG X W, et al. Robust least squares support vector machine based on recursive outlier elimination. Soft Computing, 2010, 14(11): 1241–1251.

[22] YANG X W, TAN L J, HE L F. A robust least squares support vector machine for regression and classification with noise. Neurocomputing, 2014, 140(22): 41–52.

[23] CHEN L, ZHOU S S. Sparse algorithm for robust LSSVM in primal space. Neurocomputing, 2018, 275(1): 2880–2891.

[24] YE Y F, SHAO Y H, DENG N Y, et al. Robust L_p-norm least squares support vector regression with feature selection. Applied Mathematics and Computation, 2017, 305(6): 32–52.

[25] LU X J, LIU W B, ZHOU C, et al. Robust least-squares support vector machine with minimization of mean and variance of modeling error. IEEE Trans. on Neural Networks & Learning Systems, 2018, 29(7): 2909–2920.

[26] ZHOU S S, ZHOU W D. Unified SVM algorithm based on LS-DC loss. Machine Learning, 2021, 31(4): 1–28.

[27] SCHLKOPF B, HERBRICH R, SMOLA A J. A generalized representer theorem. Berlin: Springer, 2001.

[28] SHALEV-SHWARTZ S, BEN-DAVID S. Understanding machine learning: from theory to algorithms. New York: Cambridge University Press, 2014.

[29] YUILLE A L, RANGARAJAN A. The concave convex procedure. Neural Computation, 2003, 15(4): 915–936.

[30] TAO P D, LE T. Convex analysis approach to d.c. programming: theory, algorithm and applications. Acta Mathematica Vietnamica, 1997, 22(1): 289–355.

[31] WANG K N, ZHONG P. Robust non-convex least squares loss function for regression with outliers. Knowledge-Based Systems, 2014, 71(1): 290–302.

[32] ZHOU S S, CUI J T, YE F, et al. New smoothing SVM algorithm with tight error bound and efficient reduced techniques. Computational Optimization & Applications, 2013, 56(3): 599–617.

[33] HE B S, YUAN X M. On the $O(1/n)$ convergence rate of the Douglas-Rachford alternating direction method. SIAM Journal on Numerical Analysis, 2012, 50(2): 700–709.

[34] SHAO Y H, WANG Z, CHEN W J, et al. A regularization for the projection twin support vector machine. Knowledge-Based Systems, 2013, 37(2): 203–210.

[35] GUO J H, YI P, WANG R L, et al. Feature selection for least squares projection twin support vector machine. Neurocomputing, 2014, 144(11): 174–183.

[36] SHAO Y H, ZHANG C H, WANG X B, et al. Improvements on twin support vector machines. IEEE Trans. on Neural Networks, 2011, 22(6): 962–968.

[37] CHANG C C, LIN C J. LIBSVM: a library for support vector machines. http://www.csie.ntu.edu.tw/~cjlin/libsvm.

Biographies

ZHOU Shuisheng was born in 1972. He received his M.S. degree in applied mathematics and Ph.D. degree in computer science from Xidian University, Xi’an, China, in 1998 and 2005, respectively. He is currently a professor in the School of Mathematics and Statistics, Xidian University. His current research interests include optimization algorithm and its application, machine learning, pattern recognition, kernel-based learning, and support vector machines.

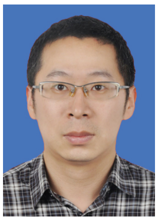
E-mail: sszhou@mail.xidian.edu.cn

ZHANG Wenmeng was born in 1996. She received her degree in the School of Mathematics and Statistics, Xidian University. Her current research interests include optimization algorithm and its application, machine learning, pattern recognition, kernel-based learning, and support vector machines.

E-mail: 3137710140@qq.com

CHEN Li was born in 1982. She received her Ph.D. degree in School of Mathematics and Statistics, Xidian University, Xi’an, China and M.S. degree in mathematics from China Agricultural University, Beijing, China, in 2019 and 2009, respectively. She is currently an associate professor in Zhengzhou University. Her current research interests include optimization algorithm and its application, machine learning, pattern recognition, and support vector machines.

E-mail: cli@zzu.edu.cn



XU Mingliang was born in 1981. He received his Ph.D. degree from the State Key Lab of Computer Aided Design and Computer Graphics, Zhejiang University, China. He is a professor in the School of Computer and Artificial Intelligence, Zhengzhou University, China. His current research interests include computer graphics, multimedia and artificial intelligence.

E-mail: iexumingliang@zzu.edu.cn