Neurocomputing 144 (2014) 174–183

Contents lists available at ScienceDirect

Neurocomputing

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

Feature selection for least squares projection twin support vector machine

Jianhui Guo [a,e,*], Ping Yi [b], Ruili Wang [c], Qiaolin Ye [d], Chunxia Zhao [a]

[a] School of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing, China
[b] School of Instrument Science and Engineering, Southeast University, Nanjing, China
[c] School of Engineering and Advanced Technology, Massey University, Auckland, New Zealand
[d] School of Information Technology, Nanjing Forestry University, Nanjing, China
[e] Jiangsu Key Laboratory of Image and Video Understanding for Social Safety, Nanjing University of Science and Technology, Nanjing, China

ARTICLE INFO

Article history:
Received 14 September 2013
Received in revised form
4 May 2014
Accepted 4 May 2014
Communicated by W.S. Hong
Available online 2 June 2014

Keywords:
Twin Support Vector Machine
Least Squares Projection Twin Support Vector Machine
Feature selection

ABSTRACT

In this paper, we propose a new feature selection approach for the recently proposed Least Squares Projection Twin Support Vector Machine (LSPTSVM) for binary classification. 1-norm is used in our feature selection objective so that only non-zero elements in weight vectors will be chosen as selected features. Also, the Tikhonov regularization term is incorporated to the objective of our approach to reduce the singularity problems of Quadratic Programming Problems (QPPs), and then to minimize its 1-norm measure. This approach leads to a strong feature suppression capability, called as Feature Selection for Least Squares Projection Twin Support Vector Machine (FLSPTSVM). The solutions of FLSPTSVM can be obtained by solving two smaller QPPS arising from two primal QPPs as opposed to two dual ones in Twin Support Vector Machine (TWSVM). Thus, FLSPTSVM is capable of generating sparse solutions. This means that FLSPTSVM can reduce the number of input features for a linear case. Our linear FLSPTSVM can also be extended to a nonlinear case with the kernel trick. When a nonlinear classifier is used, the number of kernel functions required for the classifier is reduced. Our experiments on publicly available datasets demonstrate that our FLSPTSVM has comparable classification accuracy to that of LSPTSVM and obtains sparse solutions.

1. Introduction

Support Vector Machines (SVM) [1,2] is a very useful machine learning method, which is developed on the basis of the statistical learning theory and structural risk minimization [3,4]. Compared with other machine learning approaches such as Artificial Neutral Networks (ANNS), SVM implements the structural risk minimization rather than the empirical risk minimization principle, thus SVM minimizes the upper bound of the generalization error. As a powerful tool for supervised learning, SVM can handle small samples, nonlinear, dimension disaster, over learning and local minimum problems [5]. It has been successfully applied to a variety of real-world problems such as face recognition [6–8], text categorization [9], bioinformatics [10–12], time series prediction [13], and regression estimation [14,15].

However, one of the main challenges for the classic SVM is the high computational complexity, and its computational complexity is at most O (N^3), where N is the number of training samples [16]. This drawback restricts the application of SVM to large-scale problems. In the last few years, a series of modified SVM algorithms have been developed to improve computational efficiency. Proximal Support Vector Machines (PSVM) [17] solve a set of linear equations rather than convex optimization problems, and its time complexity is O (n^3), where n is the dimensions of the samples. In essence, PSVM classifies the samples by two parallel hyperplanes on the premise of guaranteeing the maximum margin. Generalized Proximal SVM (GEPSVM) [18] is an extension of PSVM, which aims at generating two nonparallel hyperplanes so that each hyperplane is closer to its class and is as far as possible from the other classes. Subsequently, Jayadeva et al. [19] proposed the Twin Support Vector Machines (TWSVM). This algorithm seeks two nonparallel planes by solving two related SVM-type problems, each of which is smaller than a classic SVM. Compared with a classic SVM, the major advantage of TWSVM is that the speed of TWSVM is 4 times faster than that of the classic SVM. In order to further reduce the computational cost of TWSVM, the Least Squares version of TWSVM is proposed (LSTSVM) [20]. LSTSVM possesses extremely faster training speed since their separating hyperplanes are determined by solving a set of linear equations. Essentially, it solves two related PSVM type of problems. Shao et al. [21] proposed a modified TWSVM to improve the performance of classification, named the Twin Bounded Support Vector Machines

* Corresponding author. Tel.: +86 18120185368.
E-mail addresses: guojianhui@njust.edu.cn (J. Guo), greenapple8189@163.com (P. Yi).

http://dx.doi.org/10.1016/j.neucom.2014.05.040
0925-2312/

(TBSVM). Ye et al. [22] proposed a multi-plane learning approach called Localized Twin SVM via Convex Minimization (LCTSVM). LIPSVM uses the selectively generated points to train the classifier. The major advantage of LCTSVM is that it is resistant to outliers.

Different from TWSVM that improves GEPSVM by using SVM-type formulation, the Multi-weight Vector Projection Support Vector Machines (MVSVM) [23] were proposed to enhance the performance of GEPSVM by seeking one weight vector instead of a hyper-plane for each class so that the samples of one class are closest to its class mean while the samples of different classes are separated as far as possible. The weight vectors of MVSVM can be found by solving a pair of eigenvalue problems.

Further, inspired by MVSVM and TWSVM, Chen et al. [16] proposed the Projection Twin Support Vector Machines (PTSVM). PTSVM solves two related SVM-type problems in order to obtain the two projection directions whereas MVSVM needs to solve two generalized eigenvalue problems. Both PTSVM and TWSVM are implemented by solving two smaller QPPs. Also, by using the proposed recursive algorithm, PTSVM can generate multiple projection directions. Experimental results in [16] show that PTSVM has comparable or better performance comparing with the other four state-of-the-art Multiple-surface Classification (MSC) algorithms (*i.e.* GEPSVM, TWSVM, LSTSVM and MVSVM). In order to further enhance the performance of PTSVM, Shao et al. [24] proposed a least squares version of PTSVM, called Least Squares Projection Twin Support Vector Machine (LSPTSVM). LSPTSVM solves two modified primal problems by solving two sets of linear equations whereas PTSVM needs to solve two QPPs along with two sets of linear equations.

In the past few years, the research in automatic feature selection optimization has attracted more and more attention. In particle for online applications, feature selection is a critical step for data classification, especially when the classification problems have high dimensional spaces. It is well known that the 1-norm SVM has the advantages over the 2-norm SVM since 1-norm SVM can generate sparse solutions and it thus makes its classifier easier to store and faster to compute and suppress features [25,26]. In such way, only non-zero elements in weight vectors will be chosen as selected features. Therefore, the 1-norm SVM can automatically select relevant features by estimating their coefficients. Especially, when there are many noise variables in input features, the 1-norm SVM has significant advantages over the 2-norm SVM due to that the latter does not select significant variables [27]. A feature selection method for nonparallel plane support vector machine classification (FLSTSVM) [28] is proposed for strong feature suppression, in which a Tikhonov regularization term is incorporated to the objective of LSTSVM so that FLSTSVM can minimizes its 1-norm measure. The solution of FLSTSVM can be obtained by solving two smaller QPPs arising from two primal QPPs so that there is no need to solve two dual ones and thus results in sparse solutions. This is different from that in TWSVM.

In this paper, we propose a feature selection algorithm for outputting sparse solutions to LSPTSVM. The 2-norm regularization terms are replaced with the 1-norm ones in the primal problems of LSPTSVM, thereby helping suppress the input features of LSPTSVM. FLSPTSVM obtains two planes by solving two primal rather than dual QPPs, each of which is smaller than that of PTSVM and comparable to that of LSPTSVM. This paper is organized as follows: Section 2 briefly discusses TWSVM, MVSVM, PTSVM, and LSPTSVM. Section 3 proposes our FLSPTSVM and experimental results are described in Section 4. Finally, concluding remarks are given in Section 5.

2. Related work

In this section, we give a brief description of TWSVM, MVSVM, PTSVM and LSPTSVM.

We consider a binary classification problem in the n-dimensional real space R^n. The set of training data points is represented by $X=\{(x_j^{(i)}) \mid i=1,2, \quad j=1,2,\ldots,m_i\}$, where $x_j^{(i)} \in R^n$, the$j$th input belongs to class i and $m=m_1+m_2$, $y_j \in \{+1,-1\}$ are corresponding outputs. We further organize the m_1 inputs of Class$+1$ by matrix A in $R^{m_1 \times n}$ and the m_2 inputs of Class-1 by matrix B in $R^{m_2 \times n}$. The 2-norm of x is denoted by $\|x\|$, and 1-norm of x is denoted by $\|x\|_1$.

2.1. Twin SVM (TWSVM)

The basic notion of TWSVM classifier is to seek two nonparallel hyperplanes in an n-dimensional input space

$$x^T w_1+b_1=0, \quad x^T w_2+b_2=0 \tag{1}$$

where superscript "T" denotes transposition and $(w_i,b_i) \in (R^n \times R)(i=1,2)$. The classifier aims at generating two nonparallel hyperplanes so that each hyperplane is closer to its class and is as far as possible from the other classes.

The formulation of TWSVM can be written as follows [19]:

$$\text{(TWSVM1)} \quad \min_{w_1} \tfrac{1}{2}(Aw_1+e_1b_1)^T(Aw_1+e_1b_1)+c_1e_2^T\xi$$
$$s.t. -(Bw_1+e_2b_1)+\xi \geq e_2, \quad \xi \geq 0 \tag{2}$$

and

$$\text{(TWSVM2)} \quad \min_{w_2} \tfrac{1}{2}(Bw_2+e_2b_2)^T(Bw_2+e_2b_2)+c_2e_1^T\xi$$
$$s.t. \quad (Aw_2+e_1b_2)+\xi \geq e_1, \quad \xi \geq 0 \tag{3}$$

where $c_1>0$ and $c_2>0$ are the penalty parameters; e_1 and e_2 are the column vectors of ones of appropriate dimensions; ξ is the slack variable. The inequality constraint to make that the distance from samples of the other class to the hyperplane is at least 1.

Let

$$H=[A \quad e_1], \quad G=[B \quad e_2] \tag{4}$$

The Wolf dual of Eqs. (2) and (3) can be shown as follows:

$$\text{(TWSVM1)} \quad \max_{\alpha} e_2^T\alpha-\tfrac{1}{2}\alpha G(H^TH)^{-1}G^T\alpha$$
$$s.t. \quad 0 \leq \alpha \leq c_1 \tag{5}$$

and

$$\text{(TWSVM2)} \quad \max_{\gamma} e_1^T\gamma-\tfrac{1}{2}\gamma H(G^TG)^{-1}H^T\gamma$$
$$s.t. \quad 0 \leq \gamma \leq c_2 \tag{6}$$

where $\alpha \in R^{m_2}$ and $\gamma \in R^{m_1}$ are the Lagrangian multipliers.

TWSVM introduced a regularization term $\varepsilon I, \quad \varepsilon>0$ to avoid the possible ill-conditioning problem of H^TH and G^TG in [19]. Here, I is an identity matrix of appropriate dimensions. Therefore, we can get the nonparallel hyperplanes from the solutions of (5) and (6) as follows:

$$\begin{bmatrix} w_1 \\ b_1 \end{bmatrix} = -(H^TH+\varepsilon I)^{-1}G^T\alpha, \quad \begin{bmatrix} w_2 \\ b_2 \end{bmatrix} = -(G^TG+\varepsilon I)^{-1}H^T\gamma \tag{7}$$

2.2. Multi-weight vector projection support vector machines (MVSVM)

Multi-weight Vector Projection Support Vector Machines (MVSVM) [23] try to find a weight vector direction for each class so that each of the two data sets is close to its own class mean and meanwhile the points sharing different labels are separated as far as possible. The optimization criterion and corresponding constraints are defined as follows [23]:

$$\text{(MVSVM1)} \quad \max_{w_1} \left(w_1^T \frac{1}{m_1} \sum_{j=1}^{m_1} x_j^{(1)} - w_1^T \frac{1}{m_2} \sum_{j=1}^{m_2} x_j^{(2)} \right)^2$$

$$-\beta\sum_{i=1}^{m_1}\left(w_1^T x_i^{(1)}-w_1^T\frac{1}{m_1}\sum_{j=1}^{m_1}x_j^{(1)}\right)^2$$
$$s.t.\quad \|w_1\|^2=1 \tag{8}$$

and

$$\text{(MVSVM2)}\quad \max_{w_2}\left(w_2^T\frac{1}{m_2}\sum_{j=1}^{m_2}x_j^{(2)}-w_2^T\frac{1}{m_1}\sum_{j=1}^{m_1}x_j^{(1)}\right)^2$$
$$-\beta\sum_{i=1}^{m_2}\left(w_2^T x_i^{(2)}-w_2^T\frac{1}{m_2}\sum_{j=1}^{m_2}x_j^{(2)}\right)^2$$
$$s.t.\quad \|w_2\|^2=1 \tag{9}$$

where β is a trade-off regularization constant. The formulation of MVSVM can avoid the singularity problem in GEPSVM by optimizing the difference instead of quotient [29]. To simplify the above formulation, let us give the following definitions:

$$S_1=\sum_{i=1}^{m_1}\left(x_i^{(1)}-\frac{1}{m_1}\sum_{j=1}^{m_1}x_j^{(1)}\right)\left(x_i^{(1)}-\frac{1}{m_1}\sum_{j=1}^{m_1}x_j^{(1)}\right)^T \tag{10}$$

$$S_2=\sum_{i=1}^{m_2}\left(x_i^{(2)}-\frac{1}{m_2}\sum_{j=1}^{m_2}x_j^{(2)}\right)\left(x_i^{(2)}-\frac{1}{m_2}\sum_{j=1}^{m_2}x_j^{(2)}\right)^T \tag{11}$$

$$S_3=\left(\frac{1}{m_2}\sum_{j=1}^{m_2}x_j^{(2)}-\frac{1}{m_1}\sum_{j=1}^{m_1}x_j^{(1)}\right)\left(\frac{1}{m_2}\sum_{j=1}^{m_2}x_j^{(2)}-\frac{1}{m_1}\sum_{j=1}^{m_1}x_j^{(1)}\right)^T \tag{12}$$

Considering the previous definitions, we convert MVSVM1 and MVSVM2 to their equivalent formulations shown as follows:

$$\text{(MVSVM1)}\quad \max_{\|w_1\|=1} w_1^T S_3 w_1-\beta w_1^T S_1 w_1 \tag{13}$$

and

$$\text{(MVSVM2)}\quad \max_{\|w_2\|=1} w_2^T S_3 w_2-\beta w_2^T S_2 w_2 \tag{14}$$

The optimal directions w_1 and w_2 are the eigenvectors corresponding to the maximum eigenvalue of matrices $S_3-\beta S_1$ and $S_3-\beta S_2$, respectively [23].

2.3. Projection twin support vector machines (PTSVM)

The main notion of PTSVM [16] is to find a projection direction for each class so that the within-class variance of the projected samples of its own class is minimized meanwhile the projected samples of the other class scatter away as far as possible. Therefore, the formulations of PTSVM are a pair of QPPs

$$\text{(PTSVM1)}\quad \min_{w_1}\tfrac{1}{2}w_1^T S_1 w_1+C_1 e_2^T\xi$$
$$s.t.\quad Bw_1-\frac{1}{m_1}e_2 e_1^T A w_1+\xi\geq e_2,\quad \xi\geq 0 \tag{15}$$

and

$$\text{(PTSVM2)}\quad \min_{w_2}\tfrac{1}{2}w_2^T S_2 w_2+C_2 e_1^T\eta$$
$$s.t.\quad -\left(Aw_2-\frac{1}{m_2}e_1 e_2^T B w_2\right)+\eta\geq e_1,\quad \eta\geq 0 \tag{16}$$

where C_1 and C_2 are both trade-off constants; $\xi$and η are all nonnegative slack variables, and S_1 and S_2 see the definitions in Eqs. (10) and (11).

The first term in the objective function of Eq. (15) makes the variance of the projected samples of Class $+1$as small as possible so that they are clustered around its mean. The second term reduces the hinge loss. The constraints in Eq. (15) require the projected samples of the other class (i.e. Class -1) to be at a distance of at least 1 from the projected center of Class $+1$. This term aims at maximizing the separation between the two classes. Slack variables ξ are used to measure the errors when the constraints are not feasible. PTSVM2 (16) is similar to PTSVM1 (15).

In order to solve the constrained minimization problems in (15) and (16), the Lagrangian function is introduced and the Wolf dual problems of (15) and (16) are as follows:

$$\min_{\alpha}\tfrac{1}{2}\alpha^T\left(B-\tfrac{1}{m_1}e_2e_1^TA\right)S_1^{-1}\left(B^T-\tfrac{1}{m_1}A^Te_1e_2^T\right)\alpha-e_2^T\alpha$$
$$s.t.\quad 0\leq\alpha\leq C_1\ e_2 \tag{17}$$

and

$$\min_{\gamma}\tfrac{1}{2}\gamma^T\left(A-\tfrac{1}{m_2}e_1e_2^TB\right)S_2^{-1}\left(A^T-\tfrac{1}{m_2}B^Te_2e_1^T\right)\gamma-e_1^T\gamma$$
$$s.t.\quad 0\leq\gamma\leq C_2\ e_1 \tag{18}$$

The projection directions are given by

$$w_1=S_1^{-1}\left(B^T-\frac{1}{m_1}A^Te_1e_2^T\right)\alpha \tag{19}$$

$$w_2=S_2^{-1}\left(A^T-\frac{1}{m_2}B^Te_2e_1^T\right)\gamma \tag{20}$$

For testing, the label of a new coming sample x is determined depending on the distance between the projection of x and the projected class's mean which is expressed as

$$label(x)=\arg\min_{i=1,2}(d_i)=\arg\min_{i=1,2}\left(\left|w_i^Tx-w_i^T\frac{1}{m_i}\sum_{j=1}^{m_i}x_j^{(i)}\right|\Big/\|w_i\|\right) \tag{21}$$

2.4. Least square PTSVM (LSPTSVM)

The LSPTSVM's primal problems are the modified versions of the primal problems (15) and (16) of PTSVM using least squares. The objective functions are constructed following the idea of PTSVM proposed in [16]. Different from the primal problems (15) and (16) with the inequality constraints, the LSPTSVM primal problems have only equality constraints as follows:

$$\text{(LSPTSVM1)}\quad \min_{w_1}\tfrac{1}{2}w_1^TS_1w_1+\tfrac{C_1}{2}e_2^T\xi^2+\tfrac{C_3}{2}\|w_1\|^2$$
$$s.t.\quad Bw_1-\frac{1}{m_1}e_2e_1^TAw_1+\xi=e_2 \tag{22}$$

and

$$\text{(LSPTSVM2)}\quad \min_{w_2}\tfrac{1}{2}w_2^TS_2w_2+\tfrac{C_2}{2}e_1^T\eta^2+\tfrac{C_4}{2}\|w_2\|^2$$
$$s.t.\quad -(Aw_2-\frac{1}{m_2}e_1e_2^TBw_2)+\eta=e_1 \tag{23}$$

where $C_1>0$, $C_2>0$, $C_3>0$ and $C_4>0$ are parameters.

Note that LSPTSVM makes two modifications of PTSVM. The first one is that in the objective functions of (22) and (23), the regularization terms $1/2\ C_3\|w_1\|^2$ and $1/2\ C_4\|w_2\|^2$ are introduced. Similar to classic SVM [2] and improved TWSVM [21], this strategy will lead this method to be more theoretically sound than PTSVM. The second one is that the loss functions in (22) and (23) are the squares of 2-norm of slack variables ξ and η instead of 1-norm of $\xi$and η used in (15) and (16), which makes the constraints $\xi\geq 0$and $\eta\geq 0$ redundant [30]. This modification leads to that the solutions to QPPs (22) and (23) can be obtained by solving a set of linear equations.

On substituting the equality constraints into the objective function, QPP (22) and (23), becomes

$$\min_{w_1}\tfrac{1}{2}w_1^TS_1w_1+\frac{C_1}{2}\left\|-Bw_1+\frac{1}{m_1}e_2e_1^TAw_1+e_2\right\|^2+\frac{C_3}{2}\|w_1\|^2 \tag{24}$$

$$\min_{w_2}\tfrac{1}{2}w_2^TS_2w_2+\frac{C_2}{2}\left\|Aw_2-\frac{1}{m_2}e_1e_2^TBw_2+e_1\right\|^2+\frac{C_4}{2}\|w_2\|^2 \tag{25}$$

The projection direction is given by

$$w_1 = \left(\frac{S_1}{C_1} + \left(-B + \frac{1}{m_1} e_2 e_1^T A\right)^T \left(-B + \frac{1}{m_1} e_2 e_1^T A\right) + \frac{C_3}{C_1} I\right)^{-1} \times \left(B - \frac{1}{m_1} e_2 e_1^T A\right)^T e_2 \tag{26}$$

$$w_2 = \left(\frac{S_2}{C_2} + \left(A - \frac{1}{m_2} e_1 e_2^T B\right)^T \left(A - \frac{1}{m_2} e_1 e_2^T B\right) + \frac{C_4}{C_2} I\right)^{-1} \times \left(A - \frac{1}{m_2} e_1 e_2^T B\right)^T e_1 \tag{27}$$

Thus, LSPTSVM completely solves the classification problem with just two matrix inverses of two much smaller dimensional matrixes.

3. Feature selection least squares projection twin support vector machine (FLSPTSVM)

3.1. Linear FLSPTSVM

Different from LSPTSVM, the objective functions (22) and (23) in LSPTSVM are modified as follows:

$$\text{(FLSPTSVM1)} \quad \min_{w_1} \tfrac{1}{2} w_1^T S_1 w_1 + \varepsilon \|w_1\|_1 + \tfrac{C_1}{2} e_2^T \xi^2 \quad s.t. \quad B w_1 - \frac{1}{m_1} e_2 e_1^T A w_1 + \xi = e_2 \tag{28}$$

and

$$\text{(FLSPTSVM2)} \quad \min_{w_2} \tfrac{1}{2} w_2^T S_2 w_2 + \varepsilon \|w_2\|_1 + \tfrac{C_2}{2} e_1^T \eta^2 \quad s.t. \quad -\left(A w_2 - \frac{1}{m_2} e_1 e_2^T B w_2\right) + \eta = e_1 \tag{29}$$

Note that formulations (28) and (29) are slightly different from both (22) and (23). FLSPTSVM aims to suppress the input features. In FLSPTSVM, the squares of 2-norm regularization terms $\|w_1\|^2$ and $\|w_2\|^2$ are replaced by 1-norm ones ($\|w_1\|_1$ and $\|w_2\|_1$), respectively. This can result in sparse solutions. Here, ε acts as a feature suppression parameter instead of a regularization parameter.

By using (10) and (11) and substituting the equality constraints into the objective function, (28) and (29) become

$$\min_{w_1} \tfrac{1}{2} w_1^T \left(A - \frac{1}{m_1} e_1 e_1^T A\right)^T \left(A - \frac{1}{m_1} e_1 e_1^T A\right) w_1 + \varepsilon \|w_1\|_1 + \frac{C_1}{2} \| - B w_1 + \frac{1}{m_1} e_2 e_1^T A w_1 + e_2 \|^2 \tag{30}$$

$$\min_{w_2} \tfrac{1}{2} w_2^T \left(B - \frac{1}{m_2} e_2 e_2^T B\right)^T \left(B - \frac{1}{m_2} e_2 e_2^T B\right) w_2 + \varepsilon \|w_2\|_1 + \frac{C_2}{2} \| A w_2 - \frac{1}{m_2} e_1 e_2^T B w_2 + e_1 \|^2 \tag{31}$$

To simplify the above formulations, let us give the following definitions:

$$G = \left(A - \frac{1}{m_1} e_1 e_1^T A\right) \tag{32}$$

$$H = -B + \frac{1}{m_1} e_2 e_1^T A \tag{33}$$

$$L = \left(B - \frac{1}{m_2} e_2 e_2^T B\right) \tag{34}$$

$$M = -A + \frac{1}{m_2} e_1 e_2^T B \tag{35}$$

Considering the previous definitions, the formulations (30) and (31) can be rewritten as follows:

$$\min_{w_1} \tfrac{1}{2} w_1^T G^T G w_1 + \varepsilon \|w_1\|_1 + \frac{C_1}{2} \|H w_1 + e_2\|^2 \tag{36}$$

$$\min_{w_2} \tfrac{1}{2} w_2^T L^T L w_2 + \varepsilon \|w_2\|_1 + \frac{C_2}{2} \| - M w_2 + e_1\|^2 \tag{37}$$

Let $w_1 = p_1 - q_1$ and $w_2 = p_2 - q_2$, where $p_1 \geq 0$, $q_1 \geq 0$, $p_2 \geq 0$ and $q_2 \geq 0$. According to [31], we know minimizing $\|w_1\|_1$ equivalent to minimizing $e^T(p_1 + q_1)$. Then, the formulations (36) and (37) can be expressed with the following equivalent formulations, respectively

$$\text{(FLSPTSVM1)} \quad \min \tfrac{1}{2} (p_1 - q_1)^T G^T G (p_1 - q_1) + \varepsilon e^T (p_1 + q_1) + \frac{C_1}{2} (H(p_1 - q_1) + e_2)^T (H(p_1 - q_1) + e_2) \quad s.t. \quad p_1 \geq 0, q_1 \geq 0 \tag{38}$$

$$\text{(FLSPTSVM2)} \quad \min \tfrac{1}{2} (p_2 - q_2)^T L^T L (p_2 - q_2) + \varepsilon e^T (p_2 + q_2) + \frac{C_2}{2} (-M(p_2 - q_2) + e_1)^T (-M(p_2 - q_2) + e_1) \quad s.t. \quad p_2 \geq 0, \; q_2 \geq 0 \tag{39}$$

where $e \in R^n$. Setting $U = [p_1 q_1]^T$, formulation (38) can be rewritten as

$$\text{(FLSPTSVM1)} \quad \min \tfrac{1}{2} U^T E^T P E U + \Phi_1^T U \quad s.t. \quad U \geq 0 \tag{40}$$

where

$$E = [I_n \;\; -I_n]^T, \quad P = (G^T G + C_1 H^T H), \quad \Phi_1 = \begin{bmatrix} \varepsilon e + C_1 H^T e_2 \\ \varepsilon e - C_1 H^T e_2 \end{bmatrix}.$$

Here, I_n is an identity matrix of n dimensions. The formulation (40) is a QPP, which does not need to solve the dual problems as in a classic SVM.

In an entirely similar way, setting $V = [p_2 q_2]^T$, the formulation (39) can be shown as

$$\text{(FLSPTSVM2)} \quad \min \tfrac{1}{2} V^T E^T Q E V + \Phi_2^T V \quad s.t. \quad V \geq 0 \tag{41}$$

where

$$Q = (L^T L + C_2 M^T M), \quad \Phi_2 = \begin{bmatrix} \varepsilon e - C_2 M^T e_1 \\ \varepsilon e + C_2 M^T e_1 \end{bmatrix}.$$

Thus, FLSPTSVM solves the classification problem by computing two QPPs with smaller sizes. Once the solutions U and V of (40) and (41) are obtained, we can get the projection directions w_1 and w_2.

After the optimal projection directions are known, the class of unknown data point $x \in R^n$ is determined depending on the distance between the projection of x and the projected class means. The class of unknown data point is expressed as

$$label(x) = \arg\min_{i=1,2} \left(\left| w_i^T x - w_i^T \frac{1}{m_i} \sum_{j=1}^{m_i} x_j^{(i)} \right| \Big/ \|w_i\| \right) \tag{42}$$

From the above procedure, we can find a single direction for each class rather than an approximate hyperplane to make the corresponding projected samples well separated. In order to further enhancing the performance of FLSPTSVM, FLSPTSVM can be extended to obtain multiple orthogonal directions similar to PTSVM [16] and LSPTSVM [24].

Algorithm 1: the linear FLSPTSVM algorithm

Given a training point set of two classes as follows:

$T=\{(x_j^{(i)})|i=1,2,\quad j=1,2,\ldots,m_i\}$, where $x_j^{(i)}\in R^n$ are jth inputs belongs to class i and $m=m_1+m_2$, samples of class $+1$ denoted by the $m_1\times n$ matrix A, samples of class -1 denoted by the $m_2\times n$ matrix B.

(1) Replace the 2-norm regularization terms in the objectives functions of (22) and (23) with the 1-norm ones and construct the optimizations (28) and (29);
(2) obtain the QPPs (40) and (41) by simplifying (38) and (39);
(3) solve U and V by the QPPS (40) and (41), then compute projection directions w_1 andw_2; and
(4) classify the unknown data according to formulation (42).

3.2. Nonlinear FLSTWSVM

In this section, we show that our linear FLSPTSVM can also be extended to the nonlinear case with the kernel trick [23,32–34], which was ignored in [24]. In order to construct our nonlinear version, the Mapping ϕ can be considered as $R^d\to H$ (KFS: Kernel Feature Space). Note that every $w\in H$ (KFS) can be written as an expansion in terms of mapped training data, thus

$$w_1=\sum_{i=1}^{m}u_{1i}\phi(x_i)=\phi(X)u_1$$

where

$$\phi(X)=(\phi(x_1),\phi(x_1),\ldots,\phi(x_m))$$

$$u_1=(u_{11},u_{12},\ldots,u_{1m})^T..$$

Let $K(C,C^T)=\phi(X)=\phi(C)=(\phi(x_1),\phi(x_1),\ldots,\phi(x_m))$ be the class matrix in KFS, where $C=\begin{bmatrix}A\\B\end{bmatrix}$ and K is an appropriately chosen kernel. Using a similar solution to the linear case, we introduce the 1-norm term into the objective functions in (30) and (31), and get

$$\min_{u_1}\tfrac{1}{2}u_1^T(K(A,C^T)-\frac{1}{m_1}e_1e_1^TK(A,C^T))^T(K(A,C^T)-\frac{1}{m_1}e_1e_1^TK(A,C^T))u_1+\varepsilon\|u_1\|_1+\frac{C_1}{2}\|-K(B,C^T)u_1+\frac{1}{m_1}e_2e_1^TK(A,C^T)u_1+e_2\|^2 \tag{43}$$

$$\min_{u_2}\tfrac{1}{2}u_2^T(K(B,C^T)-\frac{1}{m_2}e_2e_2^TK(B,C^T))^T(K(B,C^T)-\frac{1}{m_2}e_2e_2^TK(B,C^T))u_2+\varepsilon\|u_2\|_1+\frac{C_2}{2}\|K(A,C^T)u_2-\frac{1}{m_2}e_1e_2^TK(B,C^T)u_2+e_1\|^2 \tag{44}$$

setting

$$g=(K(A,C^T)-\frac{1}{m_1}e_1e_1^TK(A,C^T)) \tag{45}$$

$$h=-K(B,C^T)+\frac{1}{m_1}e_2e_1^TK(A,C^T) \tag{46}$$

$$l=(K(B,C^T)-\frac{1}{m_2}e_2e_2^TK(B,C^T)) \tag{47}$$

$$m=-K(A,C^T)+\frac{1}{m_2}e_1e_2^TK(B,C^T) \tag{48}$$

$$u_1=r_1-s_1,\ r_1\ge 0,\ s_1\ge 0$$

$$u_2=r_2-s_2,\ r_2\ge 0,\ s_2\ge 0$$

Then, the formulations (43) and (44) become

$$\text{(KFLSPTSVM1)}\quad \min\tfrac{1}{2}(r_1-s_1)^Tg^Tg(r_1-s_1)+\varepsilon\,e^T(r_1+s_1)+\frac{C_1}{2}(h(r_1-s_1)+e_2)^T(h(r_1-s_1)+e_2)\quad s.t.\quad r_1\ge 0,s_1\ge 0 \tag{49}$$

$$\text{(KFLSPTSVM2)}\quad \min\tfrac{1}{2}(r_2-r_2)^Tl^Tl(r_2-r_2)+\varepsilon\,e^T(r_2+s_2)+\frac{C_2}{2}(-m(r_2-s_2)+e_1)^T(-m(r_2-s_2)+e_1)\quad s.t.\quad r_2\ge 0,s_2\ge 0 \tag{50}$$

where $e\in R^m$.

In the same way, the formulations (49) and (50) are rewritten as two QPPs as follows:

$$\text{(KFLSPTSVM1)}\quad \min\tfrac{1}{2}u^TE^TREu+\varphi_1^Tu\quad s.t.\quad u\ge 0 \tag{51}$$

$$\text{(KFLSPTSVM2)}\quad \min\tfrac{1}{2}v^TE^TSEv+\varphi_2^Tv\quad s.t.\quad v\ge 0 \tag{52}$$

where

$$u=[r_1s_1]^T,\quad E=[I_m\ \ -I_m]^T,\quad R=(g^Tg+C_1h^Th),\quad \varphi_1=\begin{bmatrix}\varepsilon e+C_1h^Te_2\\ \varepsilon e-C_1h^Te_2\end{bmatrix},$$

$$v=[r_2s_2]^T,\quad S=(l^Tl+C_2m^Tm),\quad \Phi_2=\begin{bmatrix}\varepsilon e-C_2m^Te_1\\ \varepsilon e+C_2m^Te_1\end{bmatrix}.$$

Once the solutions u and v to the problems (51) and (52) are obtained, we can get the projection directions u_1 and u_2. The label of a new coming data point x is determined in a manner similar to the linear case, which can be expressed as follows:

$$label(x)=\arg\min_{i=1,2}\left(\left|K(x,C^T)u_i-\frac{1}{m_i}e_iK(X_i,C^T)u_i\right|\Big/\|u_i\|\right)$$

where $X_1=A$ and $X_2=B$.

4. Experimental results

In this section, in order to demonstrate the performance of our approach, we report results on publicly available data sets: the UCI Repository [35], and two artificial datasets “Cross Plane” and “Complex Xor” [18,23], as well as David Musicant's NDC Data Generator [36] datasets.

We focus on the comparisons between the proposed algorithms and some state-of-the-art multiple-surface classification methods, including GEPSVM [18], TWSVM [19], LSTSVM [21], MVSVM [23], LSPTSVM [24] and FLSTSVM [28]. All the methods are implemented using MATLAB 7.10.0 R2012a on a PC with an Intel(R) Core(TM) i5 M480 processor (2.67 GHz 2.67 GHz) with 2 GB RAM. The Mosek optimization toolbox ⟨http://www.mosek.com⟩, which implements the fast interior point based algorithms, was used for solving the dual QPPs occurring in TWSVM, FLSTSVM and FLSPTSVM.

The eigenvalue problems in GEPSVM and MVSVM are solved by the MATLAB function “eig.m”. As for LSTSVM and LSPTSVM, the involved set of linear equations is realized by MATLAB operation“\”. The “Accuracy” used to evaluate all the compared methods is defined as follows:$Acc=(TP+TN)/(TP+TN+FP+FN)$, where TP, TP, TN, FP and FN are the numbers of true positive, true negative, false positive and false negative, respectively. As for the problem of selecting hyper-parameters, we employ the standard 10-fold cross-validation technique for all the datasets except the NDC Datasets. Furthermore, the parameters for the all methods are selected from the set $\left\{2^i|i=-7,-6,\ldots,6,7\right\}$. For FLSTSVM and our FLSPTSVM, the components of $w_i, i=1,2$ less than $1e-8$ in

absolute are discarded. The remaining features are used to classify the unknown samples. The projection direction in our experiments for each class is set to 1 in both LSPTSVM and FLSPTSVM.

Finally, the mean training accuracy, mean training time and mean testing accuracy with standard deviation across ten tests are adopted to measure the performances of these algorithms. Also, the best accuracy is shown by bold figures.

4.1. Toy example

Two artificial data sets, termed as "Crossplane" and "Complex-XOR" are designed to visually illustrate the effectiveness of the proposed method on XOR datasets. Figs. 1 and 2 illustrate their distributions.

The average results of GEPSVM, TWSVM, LSTSVM, MVSVM, LSPTSVM, FLSTSVM and our LSPTSVM on these two data sets are reported in Table 1. From Table 1, one can observe that FLSPTSVM gains the similar performance comparing with LSPTSVM. However, all the other five algorithms cannot yield good accuracy on complex XOR datasets except FLSPTSVM and LSPTSVM. The test accuracies of FLSPTSVM are 98.47% and 95.56%, respectively for Crossplane and Complex XOR. In contrast, our FLSPTSVM is also suitable for both Crossplane and Complex XOR problems and gains a superior performance similar to LSPTSVM in comparison with its competitors.

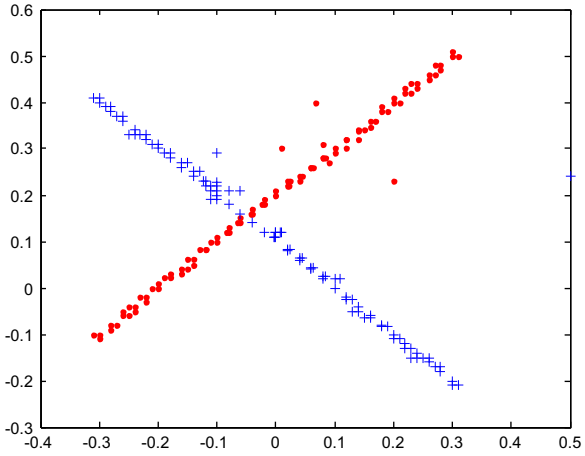


Fig. 1. Crossplane datasets.

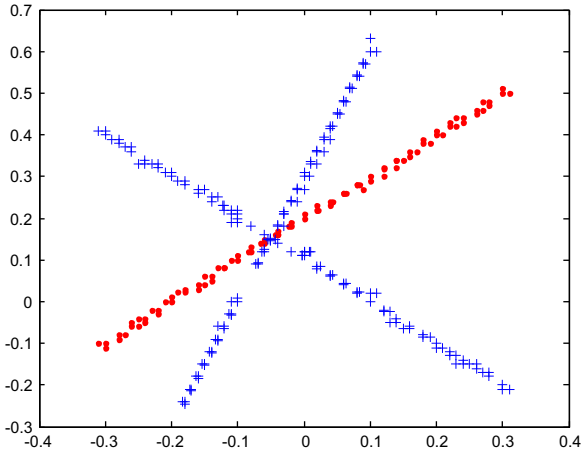


Fig. 2. Complex XOR datasets.

4.2. UCI datasets

In order to compare the behavior of our linear FLSPTSVM with other six algorithms, we apply the seven algorithms GEPSVM, TWSVM, LSTSVM, MVSVM, LSPTSVM, FLSTSVM and our FLSPTSVM to 11 selected UCI benchmark datasets and report the results on these datasets in Table 2.

The feature suppression ability of our methods FLSPTSVM and FLSTSVM [28] using the formula $(F_1+F_2)/2$, where F_1 and F_2 ($F_1, F_2 < n$, where n is the dimensions of the dataset) are the numbers of features used in FLSPTSVM1 (or FLSTSVM1) and FLSPTSVM2 (or FLSTSVM2) in the linear case, respectively. In the nonlinear case, F_1 and F_2 ($F_1, F_2 < n$) are the numbers of kernel functions used in KFLSPTSVM1 (or FLSTSVM1) and KFLSPTSVM2 (or FLSTSVM2), respectively. We computed the means and standard derivations of the 10-folds and performed paired t-tests comparing GEPSVM, TWSVM, LSTSVM, MVSVM, LSPTSVM and FLSTSVM to our FLSPTSVM. The significance level was set to 0.05. When the p-value is less than 0.05, this denotes a greater difference between two classifications accuracy values. If the p-value is "NaN", this denotes the completely same between two classifications' accuracy values.

Table 2 shows that the classification results of FLSPTSVM are comparable to TWSVM, LSTSVM, LSPTSVM and there are little statistical differences in average classification accuracy as only two or three test cases in which p-values are below 0.05. In general, when the p-value is below 0.05, the average classification results of FLSPTSVM are significantly better than others. The number of test cases in which p-values are below 0.05 by comparing our FLSPTSVM with GEPSVM is 6 out of 12 cases, with MVSVM is 4 out of 12 cases, with FLSTSVM is 6 out of 12 cases, respectively. Thus, FLSPTSVM generalization capabilities are better than that of GEPSVM, MVSVM and FLSTSVM on many of the datasets considered. Actually, it is easy to check that FLSPTSVM is better than FLSTSVM in classification accuracy on some datasets. In the Semeion Handwritten Digit dataset, the dimension of feature is 256. The average selective features of FLSPTSVM is 122.6 and 86.2 in the "0 vs 1" and "1 vs 2" cases, respectively. In other words, the feature suppression rates are 47.89% and 33.67%, correspondingly. When the feature dimension is higher, the effectiveness of feature suppression is better. Table 2 shows that FLSPTSVM obtains effective feature suppression which is similar with FLSTSVM in the linear cases.

Table 3 shows the comparisons in the nonlinear case with kernel trick. Note that in [23] LSPTSVM ignored the nonlinear cases. Thus, we only compare our nonlinear FLSPTSVM with GEPSVM, TWSVM, LSTSVM, MVSVM and FLSTSVM. The Gaussian kernel $K(x_i, x_j) = \exp(-\gamma\|x_i - x_j\|^2)$ is used. The ratios in Table 3 indicate the feature suppression rates of FLSTSVM and FLSPTSVM. $Ratio = (F_1+F_2)/(2n)$, where n is the dimensions of kernel functions. Table 3 shows that FLSPTSVM has a comparable generalization performance to both TWSVM and LSTSVM, while has a better generalization performance than GEPSVM, MVSVM and FLSTSVM. Also, FLSPTSVM and FLSTSVM have fewer kernel functions and exhibit a comparable feature suppression performance. This is obvious because we give its sparse formulation by using the 1-norm measure incorporated in them. The feature suppression ratios of FLSPTSVM are less than 50% except the Heart-statlog dataset. The feature suppression ratios of FLSPTSVM for the Semeion Handwritten Digit dataset which has 256 dimensions are 42.44% and 36.12% in the "0 vs 1" and "1 vs 2" cases, respectively.

4.3. NDC datasets

To further investigate how the computing costs of all these algorithms scale with respect to the number of data points, we also conducted experiments on large NDC datasets [36] to compare the training time of the linear GEPSVM, TWSVM, MVSVM, LSTSVM, LSPTSVM, FLSTSVM and our FLSPTSVM. Table 4 gives a

Table 1
Classification accuracy (%) on crossplane and complex XOR datasets.

Datasets	GEPSVM train (%) Test ± std (%)	TWSVM train (%) Test ± std (%)	LSTSVM train (%) Test ± std (%)	MVSVM train (%) Test ± std (%)	LSPTSVM train (%) Test ± std (%)	FLSTSVM train (%) Test ± std (%)	FLSPTSVM train (%) Test ± std (%)
Crossplane 196 × 2	99.49 **99.47 ± 1.57**	98.07 97.97 ± 3.34	95.75 94.95 ± 8.94	99.49 98.97 ± 2.05	97.62 98.50 ± 2.29	95.75 94.95 ± 8.94	99.04 98.47 ± 2.3325
Xor 270 × 2	73.75 75.56 ± 7.62	90.45 87.78 ± 6.83	66.30 65.93 ± 6.37	77.28 78.52 ± 8.5747	97.53 95.50 ± 3.07	66.26 65.56 ± 6.83	96.50 **95.56 ± 3.22**

Table 2
10-fold testing percentage test set accuracy (%) of linear classifiers on UCI datasets.

Datasets	GEPSVM Acc ± std (%) *p*-Value time (s)	TWSVM Acc ± std(%) *p*-Value time (s)	LSTSVM Acc ± std (%) *p*-Value time (s)	MVSVM Acc ± std (%) *p*-Value time (s)	LSPTSVM Acc ± std (%) *p*-Value time (s)	FLSTSVM Acc ± std (%), features *p*-Value time (s)	FLSPTSVM Acc ± std (%), features *p*-Value time (s)
Votes 435 × 16	94.01 ± 2.38 0.336 0.087	94.02 ± 2.09 0.1904 1.532	94.96 ± 3.76 0.8079 0.081	**95.17 ± 2.98** 0.5161 0.092	94.25 ± 2.58 0.5932 0.111	94.02 ± 2.12, 10.10 0.3357 0.745	94.71 ± 2.05, 14.0 0.737
Monk3 432 × 6	80.14 ± 3.76 0.0021 0.088	**88.44 ± 3.27** 0.2041 1.697	80.89 ± 7.92 0.169 0.070	80.51 ± 4.06 0.000085 0.089	86.82 ± 4.56 1.0 0.109	80.68 ± 1.70, 3.75 0.0043 0.787	86.82 ± 4.62, 3.2 0.693
Spect 267 × 44	74.10 ± 5.44 0.1341 0.139	76.47 ± 8.97 0.1385 1.102	78.70 ± 7.31 0.6167 0.081	77.56 ± 7.85 0.3445 0.104	77.91 ± 7.56 0.3639 0.098	73.11 ± 10.61, 24.95 0.0503 0.789	**79.79 ± 7.95, 10.9** 0.734
Pima 768 × 8	74.74 ± 4.05 0.822 0.095	**77.22 ± ± 4.23** 0.011 3.488	74.87 ± 3.57 0.7962 0.084	72.40 ± 4.22 0.1043 0.091	74.74 ± 4.79 0.5056 0.144	69.54 ± 7.59, 7.1 0.0138 0.753	74.48 ± 4.35, 5.2 0.668
Heart-statlog 270 × 13	74.07 ± 8.14 0.00097 0.087	82.59 ± 8.77 0.011 0.94	**84.29 ± 5.98** 0.6193 0.065	73.33 ± 9.19 0.0025 0.087	73.96 ± 8.61 0.0005 0.096	71.11 ± 11.45, 11.95 0.0267 0.708	83.33 ± 6.46, 12.2 0.625
Tic-tac-toe 958 × 9	65.87 ± 3.17 0.0463 0.091	66.49 ± 3.57 0.2823 4.988	57.91 ± 10.56 0.0193 0.064	60.97 ± 5.93 0.0187 0.087	64.78 ± 4.75 0.046 0.187	56.46 ± 8.73, 9.0 0.0034 0.727	**68.58 ± 3.29, 4.35** 0.719
Cleveland 296 × 13	73.97 ± 5.31 0.0084 0.089	**82.02 ± 7.45** 0.3821 1.006	80.68 ± 6.97 0.8835 0.073	76.36 ± 7.14 0.1947 0.085	71.95 ± 8.13 0.00091 0.097	71.23 ± 11.24, 11.35 0.0474 0.723	80.36 ± 5.78, 11.0 0.630
Ionosphere 351 × 34	76.93 ± 6.54 0.0019 0.121	88.30 ± 5.20 0.1679 1.228	85.50 ± 8.03 0.4088 0.076	83.75 ± 7.46 0.0894 0.096	**88.87 ± 4.70** 0.1942 0.117	72.36 ± 14.24, 27.5 0.0132 0.801	87.17. ± 3.68 28.6 0.707
Sonar 208 × 60	74.12 ± 8.13 0.916 0.173	72.57 ± 8.33 0.8289 0.986	75.45 ± 7.89 0.7631 0.085	**79.86 ± 5.45** 0.09 0.117	77.45 ± 8.95 0.3781 0.221	71.74 ± 10.78, 34.6 0.6562 0.978	73.67 ± 11.25, 24.30 0.929
Wpbc 198 × 33	75.42 ± 12.84 0.4596 0.096	77.97 ± 8.17 0.3436 0.966	**80.58 ± 11.31** 0.234 0.085	75.45 ± 13.35 0.7386 0.096	76.37 ± 8.76 0.9623 0.104	76.75 ± 10.51, 20.6 1.0 0.789	76.42 ± 10.36, 8.0 0.707
Semeion 1593 × 256 0 vs. 1	98.08 ± 1.403 0.3514 4.13	98.13 ± 2.07 0.0527 2.12	95.66 ± 4.65 0.0133 0.34	**99.69 ± 0.94** 0.008 0.8	99.38 ± 1.25 0.343 0.472	**99.69 ± 0.94, 97.6** NaN 8.114	**99.69 ± 0.94, 122.6** 8.111
Semeion 1593 × 256 1 vs. 2	94.10 ± 4.01 0.017 4.16	94.08 ± 6.01 0.0577 1.708	82.58 ± 5.34 0.00004 0.221	97.50 ± 3.36 0.3657 0.723	96.57 ± 2.95 0.3434 0.342	96.26 ± 2.72, 157.2 0.168 7.421	**97.51 ± 2.33, 86.2** 7.883

Table 3
10-fold testing percentage test set accuracy (%) of nonlinear classifiers with Gaussian kernel on UCI datasets.

Datasets	GEPSVM Acc ± Std (%) *p*-Value time (s)	TWSVM Acc ± Std (%) *p*-Value time (s)	LSTSVM Acc ± Std (%) *p*-Value time (s)	MVSVM Acc ± Std (%) *p*-Value time (s)	FLSTSVM Acc ± Std (%), features, ratio *p*-Value time (s)	FLSPTSVM Acc ± Std (%), features, ratio *p*-Value time (s)
Votes 435 × 16	83.21 ± 4.03 0.00005 33.642	93.31 ± 3.68 0.1184 2.522	94.02 ± 2.98 0.2446 0.896	91.29 ± 5.09 0.0661 3.756	86.42 ± 4.57,187.0,47.83% 0.0001 23.623	**95.17 ± 2.38, 63.25, 16.16%** 23.485
Monk3 432 × 6	87.49 ± 16.88 0.1160 3.9051	97.83 ± 1.36 0.4434 3.962	**98.73 ± 1.63** 0.0241 1.356	68.02 ± 25.78 0.0071 8.425	98.37 ± 1.71,193.0,38.76% 0.1379 46.368	97.47 ± 1.84, 245.3, 49.2% 44.944
Spect 267 × 44	73.43 ± 10.97 0.0482 3.076	80.17 ± 3.21 0.3478 1.064	75.71 ± 4.87 0.0119 0.451	79.03 ± 8.80 0.5816 1.102	80.20 ± 7.35,60.5, 25.18% 0.6992 6.171	**81.28 ± 4.06, 69.3, 28.84%** 5.611
Pima 768 × 8	75.14 ± 4.95 0.822 86.405	**77.22 ± 3.65** 0.020 9.58	76.04 ± 3.30 0.0659 3.14	68.12 ± 5.98 0.0521 29.06	74.09 ± 4.16,134.1,19.41% 0.8794 120.42	74.22 ± 4.18, 114.05,16.50% 117.74
Heart-statlog 270 × 13	67.04 ± 7.84 0.00017 2.918	82.22 ± 6.58 0.6470 0.784	80.37 ± 8.12 0.2228 0.297	62.96 ± 13.04 0.0037 2.030	82.22 ± 5.92,73.5,30.25% 0.4679 7.705	**83.33 ± 6.46, 180.1, 74.14%** 7.189
Tic-tac-toe 958 × 9	75.15 ± 5.11 0.0 317.147	99.01 ± 1.07 0.0051 18.252	**99.06 ± 0.86** 0.0153 5.028	80.03 ± 24.20 0.008 35.639	77.66 ± 3.85,580.5,67.34% 0 212.45	96.97 ± 1.72, 405.5, 46.99% 221.12
Cleveland 296 × 13	72.53 ± 3.63 0.5036 80.475	**80.75 ± 5.08** 0.0056 0.964	74.81 ± 8.25 0.4498 0.388	60.64 ± 12.32 0.0646 1.741	69.98 ± 9.74, 91.2, 34.31% 0.8304 10.118	70.30 ± 9.41, 66.05, 24.77% 8.753
Ionosphere 351 × 34	90.02 ± 8.21 0.019 6.631	91.24 ± 8.99 0.2072 1.653	86.67 ± 13.91 0.0909 0.617	90.37 ± 8.07 0.1886 1.658	92.87 ± 4.65, 86.0, 27.3% 0.0745 14.104	**95.71 ± 4.98, 44.95, 14.23%** 12.937
Sonar 208 × 60	71.19 ± 16.16 0.0468 1.602	**87.98 ± 4.89** 0.1746 0.620	**87.98 ± 4.89** 0.1746 0.327	64.10 ± 13.17 0.00013 0.685	82.69 ± 9.14, 157.5, 84.2% 0.4694 3.673	84.64 ± 8.21, 91.0, 48.6% 3.409
Wpbc 198 × 33	72.42 ± 17.80 0.8641 1.683	**81.08 ± 11.23** 0.0121 0.514	79.47 ± 10.09 0.0053 0.209	65.37 ± 10.18 0.0841 1.310	76.97 ± 11.89, 69.3, 39.7% 0.151 3.179	73.29 ± 7.86,4.85, 2.78% 3.784
Semeion 1593 × 256 0 vs 1	99.70 ± 0.909 0.3434 7.095	**100.0 ± 0.0** NaN 2.088	**100.0 ± 0.0** NaN 2.206	99.06 ± 2.01 0.1934 3.336	**100.0 ± 0.0, 135.4, 46.5%** NaN 11.21	**100.0 ± 0.0, 123.4, 42.44%** 11.49
Semeion 1593 × 256 1 vs 2	90.31 ± 5.83 0.0007 7.029	**99.06 ± 1.432** 0.4433 2.644	98.75 ± 2.07 0.6783 2.321	92.50 ± 3.75 0.0004 4.593	98.44 ± 2.52,191.4, 66.2% 1.0 11.28	98.44 ± 2.52, 104.4, 36.12% 11.01

description of the NDC datasets. The NDC datasets are divided into a training set and a prediction set. We report the training accuracy and prediction accuracy, respectively. For experiments with the NDC datasets, we fixed the penalty parameters of all algorithms to be 1.

The results of Table 5 show that FLSPTSVM takes a considerably less computational time than TWSVM, and its computing efficiency is similar to FLSTSVM. Also, FLSPTSVM has the same order of magnitude of the computational time compared with LSTSVM and LSPTSVM. And GEPSVM and MVSVM require less computational time than others with lower accuracies in the linear case of NDC datasets.

5. Conclusion

We have improved LSPTSVM and derived a new algorithm for binary classification, which is termed as FLSPTSVM (Feature Selection LSPTSVM) in this paper. Our FLSPTSVM is an effective algorithm to find two projection directions by solving two smaller QPPs. Similar to the other multi-plane classifiers, FLSPTSVM is capable of dealing with XOR examples. Experimental results on the synthetic datasets, UCI datasets and NDC datasets demonstrate that our FLSPTSVM obtains classification accuracy comparable to that of TWSVM, LSTSVM and LSPTSVM, as in most of cases, *p*-value is higher than 0.05. This reflects that the proposed method is

Table 4
Description of NDC datasets.

Dataset	Training data	Testing data	Features
NDC-500	500	50	32
NDC-700	700	70	32
NDC-1k	1000	100	32
NDC-2k	2000	200	32
NDC-3k	3000	300	32
NDC-5k	5000	500	32
NDC-10k	10,000	1000	32
NDC-50k	50,000	5000	32
NDC-100k	100,000	10,000	32
NDC-1000k	1,000,000	100,000	32

Table 5
Comparison on large NDC datasets in linear case.

Datasets	GEPSVM Train (%) Test (%) Train time (s)	TWSVM Train (%) Test (%) Train time (s)	LSTSVM Train (%) Test (%) Train time (s)	MVSVM Train (%) Test (%) Train time (s)	LSPTSVM Train (%) Test (%) Train time (s)	FLSTSVM Train (%) Test (%) Train time (s)	FLSPTSVM Train (%) Test (%) Train time (s)
NDC-500	82.6 76.0 0.0036	**89.2** 82.0 0.1658	88.4 80.0 0.0214	77.2 64.0 0.0015	88.6 84.0 0.0011	86.4 82.0 0.0211	88.8 **88.0** 0.0200
NDC-700	83.57 80.0 0.0038	**86.86** 85.71 0.3521	86.71 85.71 0.0217	71.71 65.71 0.0017	86.57 87.14 0.0014	85.57 87.14 0.0234	86.0 **88.57** 0.0211
NDC-1k	83.00 84.00 0.0038	**86.80** 84.00 0.8447	86.30 86.00 0.0220	73.20 76.00 0.0022	**86.80** 85.00 0.0020	86.20 **87.00** 0.0244	**86.80** 85.00 0.0255
NDC-2k	85.35 82.00 0.0052	86.85 85.50 5.3239	86.85 85.50 0.0042	70.35 68.00 0.0027	86.70 85.5 0.0036	**87.10** **86.50** 0.0272	86.80 85.00 0.0258
NDC-3k	84.40 82.67 0.0056	86.53 **84.33** 16.1351	86.80 84.00 0.0094	69.87 65.00 0.0045	**86.83** **84.33** 0.0051	86.77 84.00 0.0263	**86.83** **84.33** 0.0281
NDC-5k	83.32 83.68 0.0082	85.62 **87.40** 68.4448	**86.18** 86.80 0.0113	68.80 68.40 0.0073	85.66 87.00 0.0133	86.08 86.60 0.0315	85.58 87.00 0.0302
NDC-10k	84.86 84.40 0.0136	*	86.26 **86.80** 0.0169	70.26 69.30 0.0140	86.30 86.70 0.0327	86.28 **86.80** 0.0515	**86.32** 86.60 0.0574

*Experiments were stopped due to too much computational time.

significantly similar to the three methods discussed here. However, the numbers of test cases, in which *p*-values are below 0.05, are about half of 12 test cases by comparing our FLSPTSVM with GEPSVM, MVSVM and FLSTSVM. In general, when the *p*-value is below 0.05, the average classification results of FLSPTSVM are great better than others.

Significantly, the similar as FLSTSVM, our FLSPTSVM can automatically selecting input features, and obtains comparable sparse solutions with higher classification accuracy comparing with FLSTSVM. In the UCI Semeion Handwritten Digit dataset which has 256 dimensions, the feature suppression rates of FLSPTSVM are 47.89% and 33.67% in the "0 vs 1" case and "1 vs 2" cases, respectively in linear classification. In nonlinear classification, the feature suppression ratios of FLSPTSVM are less than 50% except the Heart-statlog dataset, while the feature suppression ratios of FLSPTSVM for the Semeion Handwritten Digit dataset are 42.44% and 36.12% in the "0 vs 1" case and "1 vs 2" cases, respectively. When the feature dimension is higher, the effectiveness of feature suppression is better. Furthermore, compared with FLSTSVM, FLSPTSVM has a better generalization performance. Therefore, FLSPTSVM has the advantage of reducing feature/kernel. Thus, it can be applied to large-scale datasets with many dimensions. Especially, when feature selection is needed, FLSPTSVM is a plausible solution to binary classification.

The experimental results show that our FLSPTSVM has a comparable generalization performance to TWSVM and LSTSVM, while has a better generalization performance than GEPSVM, MVSVM and FLSTSVM. Furthermore, it exhibits a comparable feature suppression performance comparing with FLSTSVM, in both linear and nonlinear cases. Our further work will try to apply our FLSPTSVM to some real world problems. Also, we intend to extend our FLSPTSVM to multi-category classification problems.

Acknowledgment

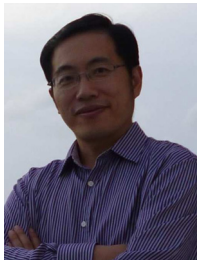
The authors would like to thank the anonymous reviewers for their constructive comments and suggestions. This work was partially supported by the Jiangsu Key Laboratory of Image and Video Understanding for Social Safety (Nanjing University of Science and Technology), Grant no. 30920130122006, the China Postdoctoral Science Foundation (Grant no. 2014M551599), and the National Natural Science Foundation of China (Grant nos. 61272220, 61101197) and the Natural Science Foundation of Jiangsu Province of China (Grant no. BK2012399).

References

[1] C. Burges, A tutorial on support vector machines for pattern recognition, Data Min. Knowl. Discov. 2 (1998) 121–167.
[2] N. Cristianini, J. Shawe-Taylor, An Introduction to Support Vector Machines: And Other Kernel-Based Learning Methods, Cambridge University Press, Cambridge, New York, 2000.
[3] V.N. Vapnik, Statistical Learning Theory, Wiley, New York, 1998.
[4] V.N. Vapnik, The Nature of Statistical Learning Theory, Springer-Verlag, New York, 1995.
[5] S. Ding, J. Yu, B. Qi, H. Huang, An overview on twin support vector machines, Artif. Intell. Rev. (2012).
[6] E. Osuna, R. Freund, F. Girosi, Training support vector machines: an application to face detection, Proc. 1997 IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (1997) 130–136.
[7] X. Zhou, W. Jiang, Y. Tian, Y. Shi, Kernel subclass convex hull sample selection method for SVM on face recognition, Neurocomputing 73 (2010) 2234–2246.
[8] X. Chen, J. Yang, Q. Mao, F. Han, Regularized least squares fisher linear discriminant with applications to image recognition, Neurocomputing 122 (2013) 521–534.
[9] T. Joachims, Text categorization with support vector machines: learning with many relevant features, Mach. Learn.: ECML 98 (1998) 137–142.
[10] G.R. Naik, D.K. Kumar, Jayadeva, hybrid independent component analysis and twin support vector machine learning scheme for subtle gesture recognition, Biomed. Tech./Biomed. Eng. 55 (2010) 301–307.
[11] G.R. Naik, D.K. Kumar, Jayadeva, twin SVM for Gesture classification using the surface electromyogram, IEEE Trans. Inf. Technol. BioMed. 14 (2) (2010) 301–308.
[12] D.D. Wang, R. Wang, H. Yan, Fast prediction of protein-protein interaction sites based on Extreme Learning Machines, fast prediction of protein–protein interaction sites based on Extreme Learning Machines, Neurocomputing 128 (27) (2014) 258–266.
[13] L. Cao, Support vector machines experts for time series forecasting, Neurocomputing 51 (2003) 321–339.
[14] X. Chen, J. Yang, J. Liang, Q. Ye, Recursive robust least squares support vector regression based on maximum correntropy criterion, Neurocomputing 97 (2012) 63–73.
[15] Y. Zhao, J. Zhao, M. Zhao, Twin least squares support vector regression, Neurocomputing 118 (2013) 225–236.
[16] X. Chen, J. Yang, Q. Ye, J. Liang, Recursive projection twin support vector machine via within-class variance minimization, Pattern Recognit. 44 (2011) 2643–2655.
[17] G. Fung, O.L. Mangasarian, Proximal support vector machine classifiers, in: F. Provost, R. Srikant, Eds., Proceedings of the Knowledge Discovery and Data Mining, 2001 pp. 77–86.
[18] O. Mangasarian, E. Wild, Multisurface proximal support vector classification via generalize eigenvalues, IEEE. Trans. Pattern. Anal. Mach. Intell. 28 (1) (2006) 69–74.
[19] R. Khemchandani Jayadeva, S. Chandra, Twin support vector machines for pattern classification, IEEE Trans. Pattern Anal. Mach. Intell. 29 (2007) 905–910.
[20] M. Arun Kumar, M. Gopal, Least squares twin support vector machines for pattern classification, Expert Syst. Appl. 36 (2009) 7535–7543.
[21] Y. Shao, C. Zhang, X. Wang, N. Deng, Improvements on Twin Support Vector Machines, IEEE Trans. Neural Netw. 22 (6) (2011) 962–968.
[22] Q. Ye, C. Zhao, N. Ye, X. Chen, Localized twin SVM via convex minimization, Neurocomputing 74 (2011) 580–587.
[23] Q. Ye, C. Zhao, N. Ye, Y. Chen, Multi-weight vector projection support vector machines, Pattern Recognit. Lett. 31 (13) (2010) 2006–2011.
[24] Y. Shao, N. Deng, Z. Yang, Least squares recursive projection twin support vector machine for classification, Pattern Recognition 45 (2012) 2299–2307.
[25] W.D. Zhou, L. Zhang, L.C. Jiao, Linear programming support vector machines, Pattern Recognit. 35 (12) (2002) 2927–2936.
[26] H. Zou, An improved 1-norm SVM for simultaneous classification and variable selection, in: Proceedings of the Eleventh International Conference on Artificial Intelligence and Statistics, 2007.
[27] J. Zhu, S. Rosset, T. Hastie, R. Tibshirani, 1-norm support vector machines, Neural Inf. Proc. Syst. (2003) 16–23.
[28] Q. Ye, C. Zhao, N. Ye, H. Zheng, X. Chen, A feature selection method for nonparallel plane support vector machine classification, Optim. Methods Softw. 27 (3) (2012) 431–443.
[29] Y. Tao, J. Yang, Quotient vs difference: comparison between the two discriminant criteria, Neurocomputing 73 (2010) 1808–1817.
[30] O. Mangasarian, D. Musicant, Lagrangian support vector machines, J. Mach. Learn. Res. 1 (2001) 61–177.
[31] G. Fung, O.L. Mangasarian, A feature selection Newton method for support vector machine classification, Comput. Optim. Appl. 28 (2) (2004) 185–202.
[32] K.R. Müller, S. Mika, G. Ratsch, An introduction to kernel-based learning algorithms, IEEE Trans. Neural Netw. 12 (2) (2001) 181–202.
[33] X.i. Peng, D. Xu, Bi-density twin support vector machines for pattern recognition, Neurocomputing 99 (2013) 134–143.
[34] Y. Fang, R. Wang, B. Dai, X. Wu, Graph-based Learning via auto-grouped sparse regularization and kernelized extension, IEEE Trans. Knowl. Data Eng. 25 (2013) 10.1109/TKDE.2014.2312322.
[35] C..Blake, C..Merz, UCI Repository of Machine Learning Databases, Department of Information and Computer Sciences, University of California, Irvine, ⟨http://www.ics.uci.edu/~mlearn/MLRepository.html⟩, 1998.
[36] D.R..Musicant, NDC: Normally Distributed Clustered Datasets, www.cs.wisc.edu/dmi/svm/ndc/, 1998.

Jianhui Guo received his B.Sc., M.Sc. and Ph.D. from the Nanjing University of Science and Technology, Nanjing, China, in 2003, 2005 and 2008, respectively. In 2008, he joined the Nanjing Institute of Electronics Technology as a senior engineer. Currently, he is an Associate Professor in the School of Computer Science and Engineering, the Nanjing University of Science and Technology. His research interests include machine learning, data mining, pattern recognition, robotics and information fusion. In these areas, he has published over 10 journal and conference papers. (Email: guojianhui@njust.edu.cn)

Ping Yi received her B.Sc. and M.Sc. both from the Nanjing University of Science and Technology, Nanjing, China, in 2003 and 2006, respectively. Currently, she is a Ph.D. student in the School of Instrument Science and Engineering, Southeast University, Nanjing, China. Her research interests include intelligent robot, machine learning and pattern recognition.



Ruili Wang received his Ph.D. in Computer Science from Dublin City University, Ireland. Currently he is a Senior Lecturer in Computer Science and Information Technology. His research interests include intelligent systems and speech processing. He has been awarded one of the most prestigious research grants in New Zealand, Marsden Fund. He is an associate editor and member of editorial boards of 5 international journals.

Qiaolin Ye received his B.Sc. in Computer Science from the Nanjing Institute of Technology, China, M.Sc. in Computer Science and Technology from the Nanjing Forestry University, China. Also, he received his Ph.D. from the Nanjing University of Science and Technology, China, in 2013. He is now an Associate Professor in Computer Science and Technology at the Nanjing Forestry University. His research interests include machine learning, data mining, pattern recognition and robotics.

Chunxia Zhao received her Ph.D. in Electronic Engineering from the Harbin Institute of Technology in 1998. Since 2000, as a full Professor, she has been with the Computer Science and Technology Department at the Nanjing University of Science and Technology, Nanjing, China. She is now a senior member of China Computer Federation. Her current research interests include pattern recognition, image processing, artificial intelligence, and mobile robots.