Applied Intelligence (2019) 49:4258–4275
https://doi.org/10.1007/s10489-019-01505-5

An efficient regularized K-nearest neighbor structural twin support vector machine

Fan Xie[1] · **Yitian Xu**[1]

Published online: 5 June 2019

Abstract

K-nearest neighbor based structural twin support vector machine (KNN-STSVM) performs better than structural twin support vector machine (S-TSVM). It applies the intra-class KNN method, and different weights are given to the samples in one class to strengthen the structural information. For the other class, the redundant constraints are deleted by the inter-class KNN method to speed up the training process. However, the empirical risk minimization principle is implemented in the KNN-STSVM, so it easily leads to over-fitting and reduces the prediction accuracy of the classifier. To enhance the generalization ability of the classifier, we propose an efficient regularized K-nearest neighbor structural twin support vector machine, called RKNN-STSVM, by introducing a regularization term into the objective function. So there are two parts in the objective function, one of which is to maximize the margin between the two parallel hyper-planes, and the other one is to minimize the training errors of two classes of samples. Therefore the structural risk minimization principle is implemented in our RKNN-STSVM. Besides, a fast DCDM algorithm is introduced to handle relatively large-scale problems more efficiently. Comprehensive experimental results on twenty-seven benchmark datasets and two popular image datasets demonstrate the efficiency of our proposed RKNN-STSVM.

Keywords TSVM · K-nearest neighbors · Structural information · Regularization

1 Introduction

The support vector machine (SVM) approach, introduced by Vapnik [1], is an extremely powerful kernel-based machine learning technique for data classifications. At present SVM has been successfully applied in various aspects, such as pattern recognition [2], text classification [3], and imbalance classification [4]. For SVM, the target is to find a hyper-plane that can separate different classes well. Many hyper-planes can meet this requirement, and SVM finds the one which follows the maximum margin principle in statistical learning theory [1]. SVM solves a quadratic programming problem (QPP), assuring that once an optimal solution is obtained, it is the unique (global) solution. By introducing kernel trick into the dual QPP, SVM can solve nonlinear classification problems successfully and can also effectively overcome the "curse of dimension". It is well known that the training cost of SVM is $O(l^3)$, where l is the total size of the training data. As the increasing of the number of training samples, SVM costs much time.

To improve the computational speed of SVM. Recently, Jayadeva et al. [5] proposed a twin support vector machine (TSVM) for the binary classification data, which is based on the idea of the GEPSVM [6]. TSVM generates two nonparallel hyper-planes such that each hyper-plane is closer to one class and as far as possible from the other. It is implemented by solving two smaller-sized QPPs, which makes the learning speed of TSVM approximately four times faster than that of the classical SVM. Now, TSVM has been widely used because of its low computational complexity. Many variants of TSVM have been proposed, such as smooth TSVM [7], twin support vector regression [8–10], non-parallel Universum support vector machine [11]. However, each of them minimizes the empirical risk, which easily leads to the over-fitting problem. Twin bounded support vector machine (TBSVM) [12, 13] minimizes the structural risk by adding a regularization term. Wang et al. [14] proposed an improved ν-twin bounded support vector machine (Iν-TBSVM). Various algorithms [15–20] also implement the structural risk minimization principle, and have better generalization

✉ Yitian Xu
xytshuxue@126.com

[1] College of Science, China Agricultural University, Beijing 100083, China

ability. But these algorithms do not sufficiently apply the prior distribution information of samples within classes.

In fact, different classes may possess diverse structural information when dealing with real world data. This structural information may be important for classification and has a great impact on the decision function. One obvious disadvantage of TSVM is that TSVM usually pays more attention to separating two classes well but ignores the underlying structural information within classes. Based on the studies of structural SVM, such as SLMM [21], SRSVM [22]. In 2013, Qi et al. [23] proposed a novel structural twin support vector machine (S-TSVM). This S-TSVM extracts the structural information by using a hierarchical clustering method and minimizes the tightness of each cluster by using the form of the covariance matrix. To improve the computational speed of S-TSVM, Xu et al. [24] proposed the structural least square twin support vector machine (S-LSTSVM). Since S-LSTSVM only needs to solve a pair of linear equations, it greatly accelerates the calculation speed.

Although experimental results reveal that S-TSVM owns excellent generalization ability, the S-TSVM has an obvious disadvantage. It ignores that samples within different clusters have different importance, and we can improve the classification accuracy by studying the different information of samples. Ye et al. [25] proposed a weighted TSVM with local information (WLTSVM), and it introduces a novel KNN method which gives different treatments to different classes in objective function or constraints of the model. Inspired by S-TSVM and WLTSVM, Pan et al. [26] proposed a K-nearest neighbor based structural twin support vector machine (KNN-STSVM). Recently, Mir et al. [27] proposed KNN-based least squares twin support vector machine (KNN-LSTSVM), which applies the similarity information of samples into LSTSVM. And Tanveer et al. [28] proposed an efficient regularized K-nearest neighbor based weighted twin support vector regression (RKNNWTSVR). By introducing regularization terms into KNNWTSVR [10] and replacing 1-norm of slack variables with 2-norm, RKNNWTSVR not only saves computational cost, but also improves the generalization performance.

However, KNN-STSVM involves the empirical risk minimization principle, which easily leads to the over-fitting problem if outliers exist [29–31] and reduces the prediction accuracy of the classifier. RKNNWTSVR solves the above problems by introducing extra regularization terms in each objective function to implement the structural risk minimization principle. But RKNNWTSVR ignores the helpful underly structural information of the data, which may contain useful prior domain knowledge for training a model. Motivated by the studies above, we improve the primal problem of KNN-STSVM by adding regularization terms into the objective function. By doing this, our new formulation, called RKNN-STSVM, is not only singularity free but also theoretically supported by statistical learning theory [32–34]. Similar to KNN-STSVM, RKNN-STSVM constructs two nonparallel hyper-planes by solving two smaller QPPs. We also incorporate the structural information of the corresponding class into the model. In addition, we not only strengthen the structural information by giving different weights to the samples, but also delete the redundant constraints to speed up the computation process. The contributions of our paper are as follows: (1) By introducing a regularization term, the matrix is guaranteed to be reversible when solving the dual problems. However, this extra prerequisite cannot always be satisfied without a regularization term. The dual problem of our algorithm can be derived without any extra assumption and need not be modified anymore; (2) In the primal problem of KNN-STSVM, the empirical risk is minimized, whereas in our RKNN-STSVM the structural risk is minimized by adding a regularization term with the idea of maximizing the margin; (3) In order to shorten training time, an effective method (the dual coordinate descent method, DCDM) is applied to our RKNN-STSVM.

Nevertheless, like the SVM, the long training time is still one of the main challenges of our RKNN-STSVM. So far, many fast optimization algorithms have been presented to accelerate the training speed, including the decomposition method [35], sequential minimal optimization (SMO) [36], the geometric algorithms [37], the dual coordinate descent method (DCDM) [38], and so on. The recently proposed DCDM algorithm can directly solve the dual QPP of SVM by updating one variable sub-problem during each iteration. The updated variable will derive the largest decrease on the objective value. This DCDM method shows a fast learning speed and great efficiency in experiments. In this paper, we introduce it into our RKNN-STSVM. Comparisons of the RKNN-STSVM with some other algorithms in terms of classification accuracy and learning time have been made on several UCI datasets and Caltech image datasets, indicating the superiority of our RKNN-STSVM.

The rest of the paper is organized as follows. Section 2 outlines the TSVM, S-TSVM, and KNN-STSVM. Details of RKNN-STSVM are given in Section 3, both the linear and nonlinear cases are included. Section 4 discusses the experimental results on twenty-seven benchmark datasets and two popular image recognition datasets to investigate the effectiveness of our proposed RKNN-STSVM. In Section 5, the DCDM algorithm is introduced into RKNN-STSVM to accelerate the learning speed. The last section contains conclusions.

2 Related work

In this section, we give a brief description of TSVM, S-TSVM and KNN-STSVM. The training samples are denoted by a set $T = \{(x_1, y_1), \ldots, (x_l, y_l)\}$, where $x_i \in R^n$, $y_i \in \{1, -1\}, i = 1, 2, \ldots, l$. For convenience, matrix $A \in R^{l_1 \times n}$ represents the positive samples and matrix $B \in R^{l_2 \times n}$ represents the negative samples.

2.1 Twin support vector machine

TSVM generates two nonparallel hyper-planes instead of a single one as in the conventional SVMs. The two nonparallel hyper-planes are obtained by solving two smaller sized QPPs as opposed to a single large one in the standard SVMs. The linear TSVM aims to find two nonparallel hyper-planes in n-dimensional input space,

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

such that each hyper-plane is closer to one class and as far as possible from the other. A new sample is assigned to class $+1$ or -1 depending upon its proximity to the two nonparallel hyper-planes.

The formulations of linear TSVM can be written as follows:

$$\begin{aligned} \min_{w_1, b_1, \xi} \quad & \frac{1}{2}(Aw_1 + e_1 b_1)^T (Aw_1 + e_1 b_1) + c_1 e_2^T \xi \\ \text{s.t.} \quad & -(Bw_1 + e_2 b_1) + \xi \geqslant e_2, \xi \geqslant 0, \end{aligned} \tag{2}$$

and

$$\begin{aligned} \min_{w_2, b_2, \eta} \quad & \frac{1}{2}(Bw_2 + e_2 b_2)^T (Bw_2 + e_2 b_2) + c_2 e_1^T \eta \\ \text{s.t.} \quad & (Aw_1 + e_1 b_2) + \eta \geqslant e_2, \eta \geqslant 0, \end{aligned} \tag{3}$$

where $c_1, c_2 \geqslant 0$ are pre-specified penalty factors, e_1 and e_2 are vectors of ones of appropriate dimensions. By introducing the lagrangian multipliers, the Wolfe dual of QPPs (2) and (3) can be represented as follows:

$$\begin{aligned} \max_{\alpha} \quad & e_2^T \alpha - \frac{1}{2}\alpha^T G (H^T H)^{-1} G^T \alpha \\ \text{s.t.} \quad & 0 \leqslant \alpha \leqslant c_1, \end{aligned} \tag{4}$$

where $G = [B \;\; e]$ and $H = [A \;\; e]$, and

$$\begin{aligned} \max_{\gamma} \quad & e_1^T \gamma - \frac{1}{2}\gamma^T P (Q^T Q)^{-1} P^T \gamma \\ \text{s.t.} \quad & 0 \leqslant \gamma \leqslant c_2, \end{aligned} \tag{5}$$

where $P = [A \;\; e]$ and $Q = [B \;\; e]$.

After resolving the QPPs (4) and (5), we can obtain

$$\begin{bmatrix} w^{(1)} \\ b^{(1)} \end{bmatrix} = -(H^T H)^{-1} G^T \alpha, \tag{6}$$

and

$$\begin{bmatrix} w^{(2)} \\ b^{(2)} \end{bmatrix} = (Q^T Q)^{-1} P^T \beta. \tag{7}$$

A new testing sample $x \in R^n$ is assigned to a class $i = (+1, -1)$ by comparing the following perpendicular distance which measures the distance of the sample from the two hyper-planes (1), i.e.,

$$Class \;\; i = \arg \min_{i=1,2} \frac{|x^T w^{(i)} + b^{(i)}|}{\|w^{(i)}\|}. \tag{8}$$

In TSVM, if the number of samples in two classes is approximately equal to $l/2$, the computational complexity of TSVM is $O(2 \times (l/2)^3)$. Thus the ratio of run-times between SVM and TSVM is $l^3/(2 \times (l/2)^3) = 4$, which implies that TSVM works approximately four times faster than SVM [5].

2.2 Structural twin support vector machine

S-TSVM extracts the structural information of each class by some clustering methods. Then it applies the data distributions of the clusters in different classes into the objective functions of TBSVM, which makes S-TSVM fully exploit the structural information of samples into the model. Suppose there are C_P clusters in the positive class P and C_N clusters in the negative class N, respectively, ie., $P = P_1 \cup \ldots P_i \cup \ldots P_{C_P}, N = N_1 \cup \ldots N_j \cup \ldots N_{C_N}$. Then the subsequent optimization problem in the S-TSVM model can be formulated as follows [23]:

$$\begin{aligned} \min_{w_+, b_+, \xi} \quad & \frac{1}{2}\|Aw_+ + e_1 b_+\|_2^2 + c_1 e_2^T \xi + \frac{1}{2} c_2 (\|w_+\|_2^2 + b_+^2) \\ & + \frac{1}{2} c_3 w_+^T \Sigma_+ w_+ \\ \text{s.t.} \quad & -(Bw_+ + e_2 b_+) + \xi \geqslant e_2, \\ & \xi \geqslant 0, \end{aligned} \tag{9}$$

and

$$\begin{aligned} \min_{w_-, b_-, \eta} \quad & \frac{1}{2}\|Bw_- + e_2 b_-\|_2^2 + c_4 e_1^T \eta + \frac{1}{2} c_5 (\|w_-\|_2^2 + b_-^2) \\ & + \frac{1}{2} c_6 w_-^T \Sigma_- w_- \\ \text{s.t.} \quad & (Aw_- + e_1 b_-) + \eta \geqslant e_1, \\ & \eta \geqslant 0, \end{aligned} \tag{10}$$

where $c_i \geqslant 0 (i = 1, \ldots, 6)$ are the pre-specified penalty factors, e_1 and e_2 are vectors of ones of appropriate dimensions, both ξ and η are slack variables. $\Sigma_+ = \Sigma_{P_1} + \ldots + \Sigma_{P_{C_P}}$, $\Sigma_- = \Sigma_{N_1} + \ldots + \Sigma_{N_{C_N}}$, Σ_{P_i} and Σ_{N_j} are respectively the covariance matrices of clusters in the two

classes. The corresponding dual problems of S-TSVM are given as follows:

$$\begin{aligned} \max_{\alpha} \quad & e_2^T\alpha - \frac{1}{2}\alpha^T G(H^T H + c_2 I + c_3 J)^{-1} G^T \alpha \\ \text{s.t.} \quad & 0 \leqslant \alpha \leqslant c_1 e_2, \end{aligned} \tag{11}$$

and

$$\begin{aligned} \max_{\beta} \quad & e_1^T\beta - \frac{1}{2}\beta^T P(Q^T Q + c_5 I + c_6 F)^{-1} P^T \beta \\ \text{s.t.} \quad & 0 \leqslant \beta \leqslant c_4 e_1, \end{aligned} \tag{12}$$

where

$$G = [B\ e_2],\ H = [A\ e_1],\ J = \begin{bmatrix} \Sigma_+ & 0 \\ 0 & 0 \end{bmatrix},\ P = [A\ e_2],$$

$$Q = [B\ e_1]\ F = \begin{bmatrix} \Sigma_- & 0 \\ 0 & 0 \end{bmatrix}.$$

A new data point $x \in R^n$ is classified as the positive class or negative class depending on which of the two hyper-planes it lies closer to. The decision function of S-TSVM is

$$f(x) = \arg \min_{\pm} \left\{ \frac{|\, x^T w_{\pm} + b_{\pm}\,|}{\|\, w_{\pm}\,\|} \right\}. \tag{13}$$

2.3 K-nearest neighbor structural twin support vector machine

To further improve the computational precision and speed of a classifier, KNN-STSVM is proposed in the spirit of S-TSVM. The formulation of KNN-STSVM is similar to TSVM, as it also tries to obtain two nonparallel hyper-planes for two classes by solving a pair of small-sized QPPs as follows [26]:

$$\begin{aligned} \min_{w_+,b_+,\xi} \quad & \frac{1}{2}\sum_{i=1}^{l_1}\sum_{j=1}^{l_1} W_{s,ij}^{(1)}(w_+^T x_j^{(1)} + b_+)^2 + c_1 e_-^T \xi + \frac{1}{2} c_2 w_+^T \Sigma_+ w_+ \\ \text{s.t.} \quad & -f_j^{(2)}(w_+^T x_j^{(2)} + b_+) + \xi_j \geqslant f_j^{(2)}, \\ & \xi_j \geqslant 0,\ j = 1, \ldots, l_2, \end{aligned} \tag{14}$$

and

$$\begin{aligned} \min_{w_-,b_-,\eta} \quad & \frac{1}{2}\sum_{i=1}^{l_2}\sum_{j=1}^{l_2} W_{s,ij}^{(2)}(w_-^T x_j^{(2)} + b_-)^2 + c_3 e_+^T \eta + \frac{1}{2} c_4 w_-^T \Sigma_- w_- \\ \text{s.t.} \quad & f_j^{(1)}(w_-^T x_j^{(1)} + b_-) + \eta_j \geqslant f_j^{(1)}, \\ & \eta_j \geqslant 0,\ j = 1, \ldots, l_1, \end{aligned} \tag{15}$$

where the parameters $c_1, c_2, c_3, c_4 \geqslant 0$ determine the penalty weights, and ξ, η are slack variables. Both e_+ and e_- are vectors of ones with appropriate dimensions, $W_{s,ij}$ is the weight matrix and f_j is the vector with only 0 or 1. $\Sigma_+ = \Sigma_{P_1} + \ldots + \Sigma_{P_{C_P}}$, $\Sigma_- = \Sigma_{N_1} + \ldots + \Sigma_{N_{C_N}}$, Σ_{P_i} and Σ_{N_j} are respectively the covariance matrixes corresponding to the ith and the jth clusters in two classes. The corresponding dual problems of KNN-STSVM are given as follows:

$$\begin{aligned} \max_{\alpha} \quad & e_-^T F\alpha - \frac{1}{2}\alpha^T (F^T G)(H^T D H + c_2 J_+)^{-1}(G^T F)\alpha \\ \text{s.t.} \quad & 0 \leqslant \alpha \leqslant c_1 e_-, \end{aligned} \tag{16}$$

and

$$\begin{aligned} \max_{\gamma} \quad & e_+^T P\gamma - \frac{1}{2}\gamma^T (P^T H)(G^T Q G + c_4 J_-)^{-1}(H^T P)\gamma \\ \text{s.t.} \quad & 0 \leqslant \gamma \leqslant c_3 e_+. \end{aligned} \tag{17}$$

A new testing sample $x \in R^n$ is assigned to a class $i = (+1, -1)$ by comparing the following perpendicular distance which measures the distance of the sample from the two hyper-planes, i.e.,

$$Class\ \ i = \arg \min_{i=1,2} \frac{|x^T w^{(i)} + b^{(i)}|}{\|w^{(i)}\|}. \tag{18}$$

3 An efficient regularized K-nearest neighbor structural twin support vector machine

Motivated by [23, 25, 26, 28], we propose a reasonable and efficient variant of KNN-STSVM called regularized KNN-based weighted structural twin support vector machine, RKNN-STSVM for short. RKNN-STSVM has three steps: clustering, applying KNN tricks, and learning. RKNN-STSVM adopts some clustering techniques to capture the data distribution within classes, and KNN tricks are also applied to our model to improve the generalization performance.

3.1 Clustering

Structural support vector machine usually considers the data distribution through clustering methods. Here, we use the same clustering methods (the Ward's linkage clustering, which is one of the hierarchical clustering method) as S-TSVM to study the clusters in each individual class. The main advantage of Ward's linkage clustering (WIL) [39] is that clusters derived from this clustering method are compact and spherical, which provides a reliable basis to calculate the covariance matrices. Concretely, if S and T are two clusters, μ_S and μ_T are means of the clusters, the Ward's linkage $W(S, T)$ between clusters S and T can be computed as [21]

$$W(S, T) = \frac{|S| \cdot |T| \cdot \|\mu_S - \mu_T\|}{|S| + |T|}. \tag{19}$$

Initially, each individual sample is considered as a cluster. The Ward's linkage of two samples x_i and x_j is $W(x_i, x_j) = \|x_i - x_j\|^2/2$. When two clusters are being merged to a new cluster A', the linkage $W(A', C)$ can be represented by $W(A, C)$, $W(B, C)$ and $W(A, B)$ as [21]

$$W(A', C) = \frac{(|A|+|C|)(W, A, C)+(|B|+|C|)W(B, C)-(|C|)W(A, B)}{|A|+|B|+|C|}. \tag{20}$$

The Ward's linkage between clusters to be merged increases as the number of clusters decreases during the hierarchical clustering. A relation curve between the merge distance and the number of clusters can be determined by finding the knee point [40]. Furthermore, the WIL can be also applicable to the nonlinear case.

3.2 K-nearest neighbor method

Weighted TSVM [25] mines as much underlying similarity information within samples as possible. Weighted TSVM firstly constructs two neighbor graphs in the input space, i.e. intra-class graph G_s and inter-class graph G_d, to search the weights of samples from one class and to extract the possible SVs residing in the other class. For each sample x_i in class +1, it defines two nearest sets: $Nea_s(x_i)$ and $Nea_d(x_i)$. $Nea_s(x_i)$ contains its neighbors in class +1, while $Nea_d(x_i)$ contains its neighbors in class -1. Specifically,

$$\begin{aligned} Nea_s(x_i) = \{x_i^j | \text{ if } x_i^j \text{ and } x_i \text{ belong to the same class,} \\ 0 \le j \le k_1\}, \end{aligned} \tag{21}$$

and

$$\begin{aligned} Nea_d(x_i) = \{x_i^j | \text{ if } x_i^j \text{ and } x_i \text{ belong to the different classes,} \\ 0 \le j \le k_2\}. \end{aligned} \tag{22}$$

Clearly, Nea_s denotes a set of its k_1-nearest neighbors in class $+1$, and Nea_d stands for a set of its k_2-nearest neighbors in class -1. Two adjacent (i.e. similarity) matrices of the plane of class $+1$ corresponding to G_s and G_d can be defined [41], respectively, as follows:

$$W_{s,ij} = \begin{cases} 1, & \text{if } x_j \in Nea_s(x_i) \text{ or } x_i \in Nea_s(x_j), \\ 0, & \text{otherwise,} \end{cases} \tag{23}$$

and

$$W_{d,ij} = \begin{cases} 1, & \text{if } x_j \in Nea_d(x_i) \text{ or } x_i \in Nea_d(x_j). \\ 0, & \text{otherwise.} \end{cases} \tag{24}$$

When $W_{s,ij} = 1$ or $W_{d,ij} = 1$, an undirected edge between x_i and x_j is added to the corresponding graph. To find the possible SVs from samples in class -1, here we redefine the weight matrix $W_{d,ij}$ of G_d as follows:

$$f_j = \begin{cases} 1, & \text{if } \exists\ j, \quad W_{d,ij} \neq 0, \\ 0, & \text{otherwise.} \end{cases} \tag{25}$$

3.3 Linear case

By introducing the extra regularization terms $(\|w_+\|_2^2 + b_+^2)$ and $(\|w_-\|_2^2 + b_-^2)$ into primal problems (14) and (15), respectively, our RKNN-STSVM can be expressed as follows:

$$\begin{aligned} \min_{w_+, b_+, \xi} \quad & \frac{1}{2}\sum_{i=1}^{l_1}\sum_{j=1}^{l_1} W_{s,ij}^{(1)}(w_+^T x_j^{(1)} + b_+)^2 + c_1 e_-^T \xi \\ & + \frac{1}{2}c_2 w_+^T \Sigma_+ w_+ + \frac{1}{2}c_3(\|w_+\|_2^2 + b_+^2) \\ \text{s.t.} \quad & -f_j^{(2)}(w_+^T x_j^{(2)} + b_+) + \xi_j \geqslant f_j^{(2)}, \\ & \xi_j \geqslant 0,\ j = 1, \ldots, l_2, \end{aligned} \tag{26}$$

and

$$\begin{aligned} \min_{w_-, b_-, \eta} \quad & \frac{1}{2}\sum_{i=1}^{l_2}\sum_{j=1}^{l_2} W_{s,ij}^{(2)}(w_-^T x_j^{(2)} + b_-)^2 + c_4 e_+^T \eta \\ & + \frac{1}{2}c_5 w_-^T \Sigma_- w_- + \frac{1}{2}c_6(\|w_-\|_2^2 + b_-^2) \\ \text{s.t.} \quad & f_j^{(1)}(w_-^T x_j^{(1)} + b_-) + \eta_j \geqslant f_j^{(1)}, \\ & \eta_j \geqslant 0,\ j = 1, \ldots, l_1, \end{aligned} \tag{27}$$

where $c_1, c_2, \ldots, c_6 \geqslant 0$ are positive parameters given in advance, which are used to denote the tradeoff among each term in the objective function, respectively. ξ and η are slack variables, both e_+ and e_- are vectors of ones of appropriate dimensions, $W_{s,ij}$ and f_j are defined as $\Sigma_+ = \Sigma_{P_1} + \ldots + \Sigma_{P_{C_P}}$, $\Sigma_- = \Sigma_{N_1} + \ldots + \Sigma_{N_{C_N}}$, Σ_{P_i} and Σ_{N_j} are respectively the covariance matrixes corresponding to the ith and the jth clusters in the two classes.

To solve (26), the lagrangian function is defined as:

$$\begin{aligned} L_1 = \ & \frac{1}{2}\sum_{j=1}^{l_1} d_j^{(1)}(w_+^T x_j^{(1)} + b_+)^2 + c_1 e_-^T \xi + \frac{1}{2}c_2 w_+^T \Sigma_+ w_+ \\ & + \frac{1}{2}c_3(\|w_+\|_2^2 + b_+^2) \\ & - \sum_{j=1}^{l_2} \alpha_j(-f_j^{(2)}(w_+^T x_j^{(2)} + b_+) + \xi_j - f_j^{(2)}) - \beta^T \xi, \end{aligned} \tag{28}$$

where $d_j^{(1)} = \sum_{i=1}^{l_1} W_{s,ij}^{(1)}$, α, β are lagrangian multipliers. Differentiating the lagrangian function L_1 with respect to

variables w_+, b_+ and ξ yields the following Karush-Kuhn-Tucker (KKT) conditions:

$$\frac{\partial L_1}{\partial w_+} = \sum_{j=1}^{l_1} d_j^{(1)} x_j^{(1)} (w_+^T x_j^{(1)} + b_+) + c_2 \Sigma_+ w_+ + \sum_{j=1}^{l_2} \alpha_j f_j^{(2)} x_j^{(2)} + c_3 w_+ = 0, \quad (29)$$

$$\frac{\partial L_1}{\partial b_+} = \sum_{j=1}^{l_1} d_j^{(1)} (w_+^T x_j^{(1)} + b_+) \sum_{j=1}^{l_2} \alpha_j f_j^{(2)} + c_3 b_+ = 0, \quad (30)$$

$$\frac{\partial L_1}{\partial \xi} = c_1 e_- - \alpha - \beta = 0. \quad (31)$$

Rewrite (29) and (30) in their matrix forms, we get the following equations:

$$A^T D(Aw_+ + e_+ b_+) + B^T F\alpha + c_2 \Sigma_+ w_+ + c_3 w_+ = 0, \quad (32)$$

$$e_+^T D(Aw_+ + e_+ b_+) + e_-^T F\alpha + c_3 b_+ = 0, \quad (33)$$

where $D = diag(d_1^{(1)}, d_2^{(1)}, \ldots, d_{l_1}^{(1)})$ and $F = diag(f_1^{(2)}, f_2^{(2)}, \ldots, f_{l_2}^{(2)})$ are diagonal matrices, and $f_j^{(2)} (j = 1, 2, \ldots, l_2)$ is either 0 or 1. The following equation is obtained by combining (32) and (33).

$$H^T DHU + G^T F\alpha + (c_2 J_+ + c_3 I)U = 0, \quad (34)$$

i.e.,

$$\begin{bmatrix} w_+ \\ b_+ \end{bmatrix} = -(H^T DH + c_2 J_+ + c_3 I)^{-1} G^T F\alpha, \quad (35)$$

where

$G = [B\ e_2]$, $H = [A\ e_1]$, $J_+ = \begin{bmatrix} \Sigma_+ & 0 \\ 0 & 0 \end{bmatrix}$ and $U = [w_+\ b_+]^T$.

Finally, we can derive the dual formulation of (26) as follows:

$$\max_{\alpha} \quad e_-^T F\alpha - \frac{1}{2}\alpha^T (F^T G)(H^T DH + c_2 J_+ + c_3 I)^{-1} (G^T F)\alpha$$
$$\text{s.t.} \quad 0 \leqslant \alpha \leqslant c_1 e_-. \quad (36)$$

Similarly, the lagrangian function for (27) is defined as

$$L_2 = \frac{1}{2}\sum_{j=1}^{l_2} d_j^{(2)} (w_-^T x_j^{(2)} + b_-)^2 + c_4 e_+^T \eta + \frac{1}{2} c_5 w_-^T \Sigma_- w_- + \frac{1}{2} c_6 (\|w_-\|_2^2 + b_-^2) - \sum_{j=1}^{l_1} \gamma_j (-f_j^{(1)} (w_-^T x_j^{(1)} + b_-) + \eta_j - f_j^{(1)}) - \nu^T \eta, \quad (37)$$

where $d_j^{(2)} = \sum_{i=1}^{l_2} W_{s,ij}^{(2)}$, γ, ν are lagrangian multipliers. After differentiating the lagrangian function (37) with respect to variables w_-, b_- and η, we can obtain the simplified dual QPP of (27), which is,

$$\max_{\gamma} \quad e_+^T P\gamma - \frac{1}{2}\gamma^T (P^T H)(G^T QG + c_5 J_- + c_6 I)^{-1} (H^T P)\gamma$$
$$\text{s.t.} \quad 0 \leqslant \gamma \leqslant c_4 e_+, \quad (38)$$

where $Q = diag(d_1^{(2)}, d_2^{(2)}, \cdots, d_{l_2}^{(2)})$ and $P = diag(f_1^{(1)}, f_2^{(1)}, \cdots, f_{l_1}^{(1)})$ are diagonal matrices, $f_j^{(1)}$ is either 0 or 1, $J_- = \begin{bmatrix} \Sigma_- & 0 \\ 0 & 0 \end{bmatrix}$.

Once the solution γ is calculated, we will get the following augmented vector,

$$\begin{bmatrix} w_- \\ b_- \end{bmatrix} = (G^T QG + c_5 J_- + c_6 I)^{-1} H^T P\gamma. \quad (39)$$

3.4 Nonlinear case

For the nonlinear case, the two nonparallel hyper-planes are modified as the following kernel-generated expressions:

$$K(x)\mu_+ + b_+ = 0, \quad K(x)\mu_- + b_- = 0, \quad (40)$$

where

$$K(x) = [K(x_1, x), K(x_2, x), ..., K(x_l, x)], \quad (41)$$

and $K(\cdot)$ stands for a chosen kernel function. The primal QPPs of nonlinear RKNN-STSVM corresponding to the hyper-planes (40) are respectively given as follows:

$$\min_{\mu_+, b_+, \xi} \quad \frac{1}{2}\sum_{i=1}^{l_1}\sum_{j=1}^{l_1} W_{s,ij}^{(1)} (\mu_+^T K(x_j^{(1)}) + b_+)^2 + c_1 e_-^T \xi + \frac{1}{2} c_2 \mu_+^T \phi(M) \Sigma_+^{\phi} \phi(M) \mu_+ + \frac{1}{2} c_3 (\|\mu_+\|_2^2 + b_+^2)$$
$$\text{s.t.} \quad -f_j^{(2)} (\mu_+^T K(x_j^{(2)}) + b_+) + \xi_j \geqslant f_j^{(2)},$$
$$\xi_j \geqslant 0, j = 1, \ldots, l_2, \quad (42)$$

and

$$\min_{\mu_-, b_-, \eta} \quad \frac{1}{2}\sum_{i=1}^{l_2}\sum_{j=1}^{l_2} W_{s,ij}^{(2)} (\mu_-^T K(x_j^{(2)}) + b_-)^2 + c_4 e_+^T \eta + \frac{1}{2} c_5 \mu_-^T \phi(M) \Sigma_-^{\phi} \phi(M) \mu_- + \frac{1}{2} c_6 (\|\mu_-\|_2^2 + b_-^2)$$
$$\text{s.t.} \quad f_j^{(1)} (\mu_-^T K(x_j^{(1)}) + b_-) + \eta_j \geqslant f_j^{(2)},$$
$$\eta_j \geqslant 0, j = 1, \ldots, l_1, \quad (43)$$

where $c_1, c_2, c_3, c_4 \geqslant 0$ are predefined parameters, and e is the vector of appropriate dimensions, $W_{s,ij}$ and f_j are defined as in the linear case. Σ_+^{ϕ} and Σ_-^{ϕ} are respectively the

covariance matrices in the two classes by the kernel Ward's linkage clustering.

The Wolfe dual of problem (42) is formulated as follows:

$$\begin{aligned} \max_{\alpha} \quad & e_-^T F\alpha - \frac{1}{2}\alpha^T (F^T G_\phi)(H_\phi^T D H_\phi + c_2 J_+^\phi + c_3 I)^{-1}(G_\phi^T F)\alpha \\ \text{s.t.} \quad & 0 \leqslant \alpha \leqslant c_1 e_-, \end{aligned} \tag{44}$$

where $H_\phi = [K(A)\ e_+]$, $G_\phi = [K(B)\ e_-]$, $J_+^\phi = \begin{bmatrix} \Sigma_+^\phi & 0 \\ 0 & 0 \end{bmatrix}$, then we can further get

$$\begin{bmatrix} w_+ \\ b_+ \end{bmatrix} = -(H_\phi^T D H_\phi + c_2 J_+^\phi + c_3 I)^{-1} G_\phi^T F\alpha. \tag{45}$$

In a similar manner, the dual formulation of (43) is

$$\begin{aligned} \max_{\gamma} \quad & e_+^T P\gamma - \frac{1}{2}\gamma^T (P^T H_\phi)(G_\phi^T Q G_\phi + c_5 J_-^\phi + c_6 I)^{-1}(H_\phi^T P)\gamma \\ \text{s.t.} \quad & 0 \leqslant \gamma \leqslant c_4 e_+. \end{aligned} \tag{46}$$

Once the solution γ is found, we get the following result,

$$\begin{bmatrix} w_- \\ b_- \end{bmatrix} = (G_\phi^T Q G_\phi + c_5 J_-^\phi + c_6 I)^{-1} H_\phi^T P\gamma. \tag{47}$$

Here, specifications of the matrices D and Q are analogous to the linear case.

A new point $x \in R^n$ is classified as the positive class or negative class depending on which of the two hyper-planes given by (45) and (47) it lies closer to. The decision function of RKNN-STSVM is

$$f(x) = \arg \min_{\pm} \left\{ \frac{| x^T w_\pm + b_\pm |}{\| w_\pm \|} \right\}. \tag{48}$$

3.5 Analysis of algorithm

We first discuss the differences between TSVM, S-TSVM, KNN-STSVM, and our RKNN-STSVM.

(1) Optimization of traditional TSVM costs around $O(l^3/4)$ if we assume that the patterns in two classes are approximately equal. As for S-TSVM, the computational complexity of the clustering step is $O((l_1^2 + l_2^2)n)$. So the computational complexity of S-TSVM is $O((l_1^2 + l_2^2)n + l^3/4)$. Thus, the overall computational complexity of KNN-STSVM and RKNN-STSVM is around $O(l^2 log(l) + (l_1^2 + l_2^2)n + l^3/4)$. Certainly, this conclusion is under the assumption of no constraints deleted. In fact, some components of f are set to be 0, which implies that the constraints are redundant, so the computational complexity of KNN-STSVM and RKNN-STSVM is less than $O((l_1^2 + l_2^2)n + l^3/4)$.

(2) Unlike TSVM and KNN-STSVM, our RKNN-STSVM implements the structural risk minimization principle by introducing extra regularization terms in each objective function so that the problem becomes well-posed. It can not only help to alleviate over-fitting issue and improve the generation performance as in conventional SVM but also introduce invertibility in the dual formulation.

(3) Two parameters c_3 and c_6 introduced in our RKNN-STSVM are the weights between the regularization term and the empirical risk, so that they can be chosen to improve the RKNN-STSVM performance.

4 Numerical experiments

To validate the superiority of our algorithm, in this section, we compare our proposed RKNN-STSVM with TSVM, Iν-TBSVM, S-TSVM, KNN-STSVM, KNN-LSTSVM on twenty-seven benchmark datasets from UCI machine learning repository[1] or the LIBSVM Database [42]. They are Australian (A), BCI-1a (B), Breast Cancer1 (B1), Breast Cancer2 (B2), Breast Cancer3 (B3), Balance Scale (BS), BUPA (BU), DBworld e-mail (D), Echocardiogram (E), Fertility (F), Hepatitis (H), Haberman (HA), Heart (HE), Ionosphere (I), IRIS (IR), LSVT (L), Liver Disorder (LD), Monks (M), Pima (P), Parkinson (PA), Planning Relax (PR), Sonar (S), Spect Heart (SH), Spambase (SP), Vertebral (V), WPBC (W), Waveform (WA). For convenience, we will use the abbreviations of them in the following paper. Table 1 lists the characteristics of these datasets. To further evaluate our algorithm, we also conduct experiments and make comparisons on popular Caltech image datasets.

To make the results more convincing, we use the 5-fold cross validation method to get the classification accuracy. More specifically, we spilt each dataset into five subsets randomly, and one of those sets is reserved as a test set whereas remaining sets are considered for training. This process is repeated five times until all subsets have been a test one once. All experiments are carried out in Matlab R2017a on Windows 10.

4.1 Parameters selection

The performance of six algorithms depends heavily on the choice of kernel functions and their parameters. In

[1] http://archive.ics.uci.edu/ml/datasets.html

Table 1 The statistics of twenty-seven benchmark datasets

Datasets	#Samples	#Positive	#Negative	#Features
A	690	307	383	14
B	268	133	135	5376
B1	194	46	148	33
B2	569	357	212	30
B3	683	239	444	9
BS	576	288	288	4
BU	345	145	200	6
D	64	29	35	4702
E	131	43	88	10
F	100	88	12	9
H	80	13	67	19
HA	306	225	81	3
HE	270	150	120	13
I	351	126	225	34
IR	150	50	100	4
L	126	42	84	310
LD	345	145	200	6
M	432	216	216	6
P	768	268	500	8
PA	195	147	48	22
PR	182	130	52	12
S	208	97	111	60
SH	267	212	55	44
SP	4601	1813	2788	57
V	310	210	100	6
W	198	47	151	34
WA	3304	1647	1657	21

this paper, we only consider Gaussian kernel function $k(x_i, x_j) = exp(-\|x_i - x_j\|^2/\gamma^2)$ for these datasets. We choose optimal values of the parameters by the grid search method. The Gaussian kernel parameter γ is selected from the set $\{2^i | i = -1, 0, \ldots, 8\}$. For brevity's sake, we let $c_1 = c_4, c_2 = c_5, c_3 = c_6$ in S-TSVM, RKNN-STSVM, $c_1 = c_3, c_2 = c_4$ in KNN-STSVM, $r_1 = r_2$, $\nu_1 = \nu_2$ in Iν-TBSVM, and $c_1 = c_2$ in TSVM, KNN-LSTSVM. The parameters c_1, c_2 are selected from the set $\{2^i | i = -1, 0, \ldots, 8\}$ and r_1, r_2 are selected from the set $\{10^i | i = -6, -5, \ldots, 0\}$. In order to save calculation time and maintain calculation accuracy, we calculate the accuracy of the Breast Cancer 1 and Echocardiogram datasets along with the curve of parameter c_3 in Fig. 1, we found that the optimal parameters are chosen from 10^{-6} to 1, so c_3 is selected from the set $\{10^i | i = -6, -5, \ldots, 0\}$. We also calculate the accuracy of the Australian and the Breast Cancer 1 datasets along with the curve of parameter k in Fig. 2, the optimal value for k in KNN-STSVM, KNN-LSTSVM and RKNN-STSVM is chosen from the set $\{3, 4, 5, 6, 10, 20\}$. The parameter ν_1 is searched from the set $\{0.1, 0.2, \ldots, 0.7\}$ in Iν-TBSVM. For large-scale datasets, the range of parameters will be narrowed uniformly due to the long running time.

4.2 Result comparison and discussion

The experimental results on twenty-seven datasets are summarized in Table 2, where the "Accuracy" denotes the mean value of five times testing results, and plus or minus the standard deviation. "Time" denotes the mean value of the time taken by six methods, and each consists of training time and testing time. And the optimal parameters of six algorithms are shown in Table 3.

From the perspective of prediction accuracy in Table 2, we can learn that our proposed RKNN-STSVM outperforms other five algorithms, i.e., TSVM, S-TSVM, Iν-TBSVM, KNN-STSVM and KNN-LSTSVM, for most datasets. RKNN-STSVM follows, it produces better testing accuracy than TSVM, Iν-TBSVM, S-TSVM, and KNN-LSTSVM for most cases. The TSVM yields the lowest testing accuracy and KNN-LSTSVM yields the lowest computational cost. Two main reasons lead RKNN-STSVM to produce better accuracy. One is that different weights are proposed to the samples according to their numbers of KNNs, and the point is important if it has more KNNs. The other is that our RKNN-STSVM implements the structural risk minimization principle by introducing extra regularization terms in each objective function.

Compared with other five algorithms, our RKNN-STSVM yields the highest testing accuracy on high-dimensional datasets including both BCI-1a and Breast Cancer3. This implies that our RKNN-STSVM is also suitable for the high-dimensional dataset. However, it yields the second lowest testing accuracy on high dimensional dataset Dbworld e-mails. The main reason lies in that the number of samples in this dataset is too small, it only has 64 samples, but has 4702 features. In large datasets, we can see RKNNS-TSVM yields the highest testing accuracy and its computational time is also lower than S-TSVM and KNN-STSVM in Spambase (SP) and Waveform (WA) datasets. But too many parameters leading to much training time is the major drawback of our algorithm. In Section 5, we add the DCDM fast algorithm to accelerate the calculation speed of our RKNN-STSVM.

4.3 Friedman tests

From Table 2, one can easily observe that our proposed RKNN-STSVM does not outperform other five algorithms for all the datasets in terms of testing accuracy. To analyze the performance of six algorithms on multiple datasets statistically, as it was suggested in Demšar [43] and García and Fernández [44], we use Friedman test with the

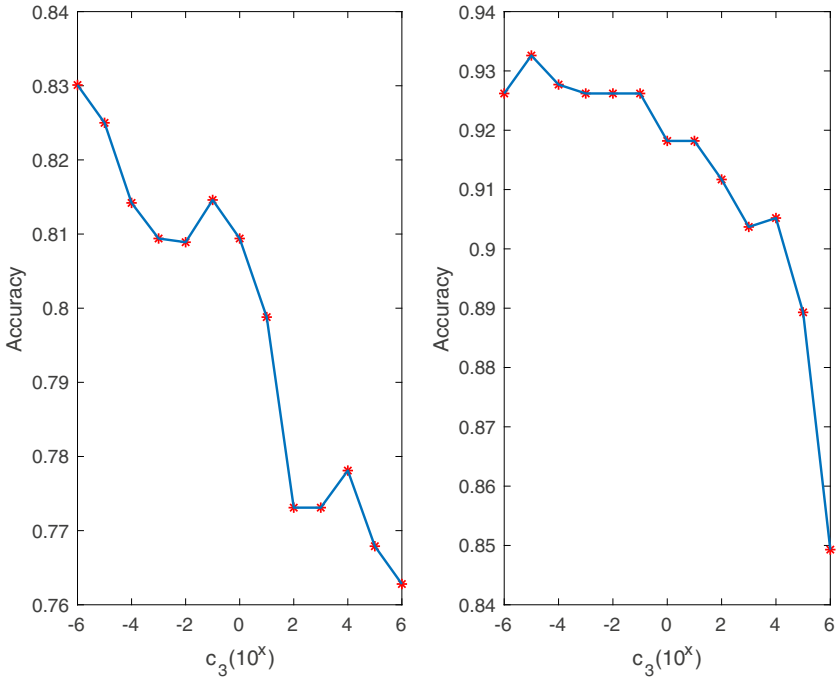

Fig. 1 Changes of accuracy with the growth of c_3 on the Breast Cance1 and Echocardiogram datasets

corresponding post hoc test which considered to be a simple, nonparametric yet safe test. For this, the average ranks of six algorithms on accuracy for all datasets are calculated and listed in Table 4. Under the null-hypothesis that all the algorithms are equivalent, one can compute the Friedman statistic [43] according to (49):

$$\chi_F^2 = \frac{12N}{k(k+1)}\left[\sum_j R_j^2 - \frac{k(k+1)^2}{4}\right], \tag{49}$$

where $R_j = \frac{1}{N}\sum_i r_i^j$, and r_i^j denotes the jth of k algorithms on the ith of N datasets. Friedman's χ_F^2 is undesirably conservative and derives a better statistic

$$F_F = \frac{(N-1)\chi_F^2}{N(k-1)-\chi_F^2}, \tag{50}$$

which is distributed according to the F-distribution with $k-1$ and $(k-1)(N-1)$ degrees of freedom.

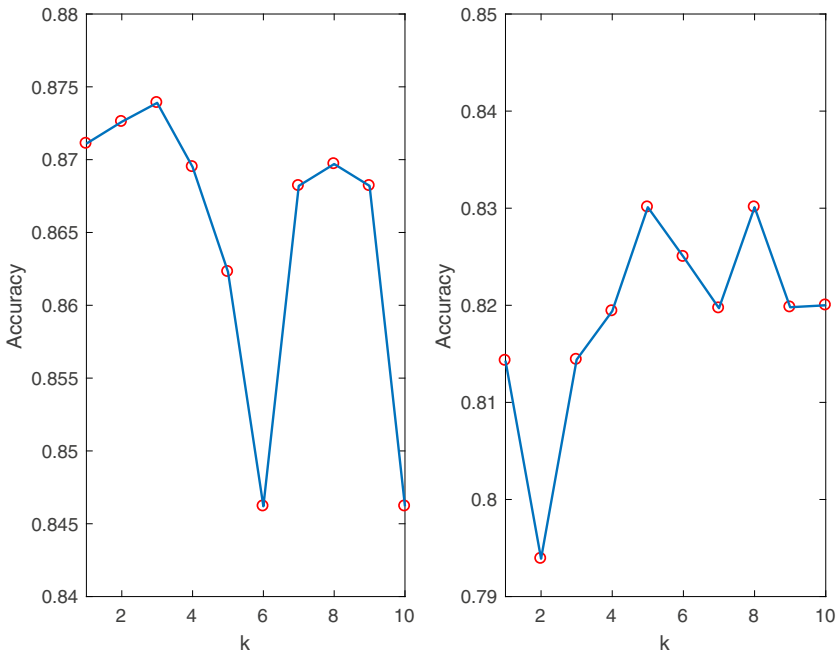

Fig. 2 Changes of accuracy with the growth of k on the Australian and Breast Cance1 datasets

Table 2 Performance comparisons of six algorithms on twenty-seven datasets. Bold type shows the best result for each dataset

Datasets	TSVM Accuracy(%) Time(s)	Iν-TBSVM Accuracy(%) Time(s)	S-TSVM Accuracy(%) Time(s)	KNN-STSVM Accuracy(%) Time(s)	KNN-LSTSVM Accuracy(%) Time(s)	RKNN-STSVM Accuracy(%) Time(s)
A	86.39 ± 2.35 0.0926	86.67 ± 2.10 0.0920	86.64 ± 3.83 0.1286	87.25 ± 2.00 0.1750	**87.39 ± 2.21** 0.0557	**87.39 ± 3.31** 0.1467
B	86.99 ± 4.31 0.0828	89.60 ± 4.43 0.0687	89.60 ± 5.01 0.2193	89.60 ± 5.53 0.2519	88.48 ± 3.52 0.1049	**90.34 ± 4.08** 0.2011
B1	77.83 ± 4.79 0.0298	80.97 ± 4.93 0.0256	80.38 ± 3.72 0.0374	81.44 ± 1.00 0.0229	80.42 ± 1.15 0.0048	**83.01 ± 3.77** 0.0251
B2	97.72 ± 1.19 0.0709	98.25 ± 0.78 0.0670	98.25 ± 0.55 0.0945	98.25 ± 0.78 0.1011	98.25 ± 0.55 0.0350	**98.42 ± 0.66** 0.0946
B3	97.22 ± 1.49 0.0922	97.51 ± 1.88 0.1393	97.52 ± 1.43 0.1258	97.52 ± 1.50 0.1322	97.51 ± 1.50 0.0527	**97.66 ± 1.62** 0.1324
BS	98.10 ± 1.50 0.0646	97.25 ± 2.36 0.0716	98.45 ± 1.27 0.0859	98.81 ± 1.25 0.1050	95.33 ± 4.92 0.0331	**98.96 ± 0.85** 0.0331
BU	73.04 ± 2.17 0.0391	72.75 ± 2.49 0.0358	73.91 ± 4.00 0.0459	74.49 ± 2.98 0.0379	73.91 ± 3.55 0.0124	**74.78 ± 3.85** 0.0401
D	79.62 ± 11.64 0.0310	84.23 ± 11.17 0.0243	**87.31 ± 9.70** 0.1438	85.90 ± 5.85 0.0437	79.62 ± 11.64 0.0138	82.69 ± 10.46 0.0387
E	90.37 ± 5.46 0.0169	91.17 ± 6.43 0.0214	92.26 ± 5.40 0.0142	92.62 ± 4.86 0.0195	92.62 ± 4.86 0.0030	**93.26 ± 4.26** 0.0211
F	89.30 ± 4.91 0.0216	89.08 ± 4.91 0.0119	89.30 ± 4.91 0.0269	90.13 ± 5.30 0.0184	89.30 ± 3.61 0.0023	**90.35 ± 4.26** 0.0276
H	88.72 ± 4.70 0.0222	88.79 ± 4.40 0.0185	**89.97 ± 3.12** 0.0253	87.62 ± 3.51 0.0162	88.64 ± 2.80 0.0018	88.79 ± 4.40 0.0317
HA	74.84 ± 2.58 0.0455	76.81 ± 1.78 0.0359	76.48 ± 2.30 0.0455	77.13 ± 1.64 0.0372	77.13 ± 1.38 0.0104	**77.79 ± 2.07** 0.0501
HE	85.56 ± 1.81 0.0419	85.19 ± 2.62 0.0336	85.93 ± 2.51 0.0466	85.93 ± 2.77 0.0354	**86.30 ± 2.22** 0.0080	85.93 ± 3.01 0.0377
I	92.87 ± 3.26 0.0378	94.86 ± 2.33 0.0469	94.01 ± 3.32 0.0588	93.44 ± 3.08 0.0475	93.45 ± 4.10 0.0127	**96.30 ± 2.14** 0.0446
IR	**100.00 ± 0.00** 0.0224	**100.00 ± 0.00** 0.0261	**100.00 ± 0.00** 0.0261	**100.00 ± 0.00** 0.0218	**100.00 ± .00** 0.0039	**100.00 ± 0.00** 0.0266
L	87.00 ± 8.31 0.0217	87.83 ± 8.92 0.0236	87.17 ± 8.19 0.0190	**89.17 ± 5.00** 0.0239	88.33 ± 6.12 0.0039	89.00 ± 5.71 0.0223
LD	72.17 ± 2.66 0.0396	73.04 ± 2.17 0.0460	74.20 ± 3.48 0.0551	74.78 ± 3.25 0.0477	72.75 ± 2.32 0.0130	**75.07 ± 2.96** 0.0541
M	81.52 ± 13.89 0.0461	**96.98 ± 4.13** 0.0534	81.52 ± 13.89 0.0540	89.92 ± 13.04 0.0614	82.95 ± 14.42 0.0196	91.75 ± 10.94 0.0587
P	77.08 ± 3.00 0.1231	77.82 ± 3.16 0.1136	78.22 ± 2.48 0.1724	**78.62 ± 3.14** 0.2036	**78.62 ± 2.81** 0.0705	78.49 ± 3.03 0.1897
PA	83.37 ± 9.68 0.0223	84.94 ± 8.71 0.0320	85.17 ± 7.20 0.0249	86.99 ± 9.36 0.0270	85.53 ± 10.71 0.0052	**87.33 ± 8.41** 0.0232
PR	**71.46 ± 1.52** 0.0246	**71.46 ± 1.52** 0.0290	**71.46 ± 1.52** 0.0229	**71.46 ± 1.52** 0.0289	**71.46 ± 1.52** 0.0046	**71.46 ± 1.52** 0.0268
S	70.42 ± 12.63 0.0279	71.50 ± 15.86 0.0320	72.25 ± 9.75 0.0256	71.25 ± 10.84 0.0289	70.00 ± 13.78 0.0056	**72.75 ± 10.74** 0.0384
SH	79.79 ± 2.91 0.0401	83.17 ± 4.01 0.0399	82.05 ± 3.30 0.0334	82.04 ± 2.42 0.0380	82.02 ± 4.99 0.0099	**83.56 ± 4.48** 0.0384

Table 2 (continued)

Datasets	TSVM Accuracy(%) Time(s)	Iν-TBSVM Accuracy(%) Time(s)	S-TSVM Accuracy(%) Time(s)	KNN-STSVM Accuracy(%) Time(s)	KNN-LSTSVM Accuracy(%) Time(s)	RKNN-STSVM Accuracy(%) Time(s)
SP	92.38 ± 4.49 10.4367	92.23 ± 4.68 11.9103	92.43 ± 4.25 14.3491	92.56 ± 4.25 15.1454	92.25 ± 4.71 6.8930	**92.58 ± 4.32** 13.9915
V	82.90 ± 10.97 0.0394	82.90 ± 8.32 0.0446	84.19 ± 8.86 0.0384	84.52 ± 8.07 0.0453	84.19 ± 5.90 0.0102	**84.84 ± 11.66** 0.0365
W	78.46 ± 6.20 0.0231	79.74 ± 3.84 0.0301	80.92 ± 4.34 0.0281	81.90 ± 4.47 0.0280	80.44 ± 5.40 0.0046	**83.92 ± 2.85** 0.0244
WA	92.01 ± 0.50 4.7165	**92.46 ± 0.61** 3.7437	92.28 ± 0.87 6.0072	**92.46 ± 0.99** 6.6676	92.34 ± 0.67 3.5239	**92.46 ± 0.88** 5.7681

We can obtain $\chi_F^2 = 59.7356$ and $F_F = 20.6356$ according to (49) and (50). Where F_F is distributed according to F-distribution with $(5, 130)$ degrees of freedom. The critical value of $F(5, 130)$ is 2.2839 for the level of significance $\alpha = 0.05$, and similarly it is 2.6656 for $\alpha = 0.025$ and 3.1612 for $\alpha = 0.01$.

Table 3 The optimal parameters of six algorithms used in the experiments

Dataset	TSVM	Iν-TBSVM	S-TSVM	KNN-STSVM	KNN-LSTSVM	RKNN-STSVM
	(c_1, γ)	(r_1, ν_1, γ)	(c_1, c_2, c_3, γ)	(c_1, c_2, γ, k)	(c_1, γ, k)	$(c_1, c_2, c_3, \gamma, k)$
A	(2,4)	(10^{-6},0.2,4)	(2,32,0.1,32)	(8,4,4,5)	(4,256,4)	(0.5,128,10^{-6},64,3)
B	(1,256)	(10^{-4},0.6,256)	(0.5,16,10^{-6},256)	(2,256,256,4)	(0.5,256,6)	(32,0.5,0.01,256,3)
B1	(8,32)	(10^{-5},0.3,32)	(8,1,10^{-6},32)	(16,1,32,3)	(8,32,10)	(8,256,10^{-6},64,5)
B2	(1,2)	(10^{-6},0.2,4)	(0.5,4,10^{-6},2)	(4,2,4,20)	(1,2,6)	(1,256,10^{-5},16,6)
B3	(4,256)	(1,0.3,2)	(8,32,10^{-6},16)	(64,1,32,4)	(16,32,5)	(32,128,0.1,16,6)
BS	(8,4)	(0.1,0.1,1)	(8,16,10^{-6},2)	(256,256,2,3)	(4,4,3)	(32,128,10^{-6},8,4)
BU	(1,2)	(10^{-4},0.7,2)	(0.5,4,10^{-6},2)	(1,4,2,3)	(8,4,6)	(4,2,0.1,1,4)
D	(1,64)	(10^{-6},0.6,256)	(1,0.5,10^{-5},64)	(128,4,256,20)	(0.5,128,10)	(0.5,0.5,0.1,128,20)
E	(8,64)	(10^{-3},0.1,4)	(8,32,10^{-6},4)	(0.5,1,32,6)	(0.5,32,10)	(64,8,10^{-5},16,5)
F	(1,0.25)	(10^{-6},0.1,0.5)	(0.5,0.5,10^{-6},2)	(0.5,4,4,3)	(1,8,4)	(0.5,16,10^{-3},4,4)
H	(1,2)	(0.01,0.7,8)	(1,16,1,8)	(0.5,0.5,4,3)	(0.5,1,3)	(0.5,32,10^{-4},32,5)
HA	(1,4)	(0.1,0.6,4)	(0.5,128,10^{-6},4)	(8,32,8,6)	(4,16,5)	(0.5,128,10^{-3},4,3)
HE	(1,16)	(10^{-6},0.4,8)	(0.5,8,10^{-6},32)	(8,4,64,6)	(64,128,5)	(2,128,1,2,4)
I	(4,4)	(10^{-3},0.4,2)	(4,256,0.1,2)	(1,256,2,20)	(1,4,20)	(4,16,1,1,20)
IR	(1,0.125)	(10^{-6},0.1,0.5)	(1,1,1,0.5)	(1,1,0.5,3)	(0.5,0.5,4)	(1,1,1,0.5,3)
L	(1,16)	(10^{-6},0.3,64)	(1,4,1,8)	(0.5,4,32,3)	(0.5,64,3)	(0.5,4,0.1,4,3)
LD	(1,2)	(10^{-6},0.6,2)	(0.5,8,1,2)	(8,128,2,10)	(4,4,20)	(4,16,10^{-3},2,10)
M	(4,16)	(10^{-6},0.6,4)	(4,1,10^{-6},16)	(0.5,64,2,20)	(0.5,4,10)	(0.5,2,10^{-6},4,20)
P	(1,32)	(1,0.2,1)	(1,32,10^{-6},2)	(32,256,2,10)	(8,4,3)	(32,128,1,1,20)
PA	(1,32)	(10^{-5},0.4,1)	(1,32,1,1)	(4,16,0.5,3)	(32,64,6)	(4,256,10^{-3},2,3)
PR	(1,0.0625)	(10^{-3},0.1,64)	(0.5,128,10^{-6},2)	(0.5,0.5,128,4)	(0.5,128,4)	(0.5,256,0.1,2,5)
S	(16,64)	(10^{-6},0.6,128)	(16,128,1,16)	(128,2,64,3)	(0.5,64,3)	(64,0.5,0.1,8,4)
SH	(8,16)	(0.01,0.3,0.5)	(0.5,64,1,1)	(256,64,1,6)	(32,64,20)	(4,128,0.1,0.5,5)
SP	(4,2)	(10^{-5},0.2,1)	(2,2,10^{-6},2)	(16,0.5,2,10)	(2,2,10)	(4,128,0.1,0.5,5)
V	(8,16)	(10^{-6},0.1,2)	(4,8,10^{-6},1)	(256,1,2,20)	(2,1,3)	(2,32,0.01,1,3)
W	(4,4)	(10^{-5},0.3,8)	(16,4,10^{-6},4)	(256,1,8,10)	(64,8,20)	(0.5,64,10^{-6},4,6)
WA	(16,256)	(10^{-5},0.2,64)	(4,256,10^{-6},8)	(16,256,16,5)	(0.5,256,4)	(16,256,10^{-5},16,5)

Table 4 Average rank on classification accuracy of six algorithms

Datasets	TSVM	Iν-TBSVM	S-TSVM	KNN-STSVM	KNN-LSTSVM	RKNN-STSVM
A	6	4	5	3	1.5	1.5
B	6	3	3	3	5	1
B1	6	3	5	2	4	1
B2	6	3.5	3.5	3.5	3.5	1
B3	6	4.5	2.5	2.5	4.5	1
BS	4	5	3	2	6	1
BU	5	6	3.5	2	3.5	1
D	5.5	3	1	2	5.5	4
E	6	5	3	3	3	1
F	4	6	4	2	4	1
H	4	2.5	1	6	5	2.5
HA	6	4	5	2.5	2.5	1
HE	5	6	3	3	1	3
I	6	2	3	5	4	1
IR	3.5	3.5	3.5	3.5	3.5	3.5
L	6	4	5	1	3	2
LD	6	4	3	2	5	1
M	5.5	1	5.5	3	4	2
P	6	4.5	4.5	1.5	1.5	3
PA	6	5	4	2	3	1
PR	3.5	3.5	3.5	3.5	3.5	3.5
S	5	3	2	4	6	1
SH	6	2	3	4	5	1
SP	4	6	3	2	5	1
V	5.5	5.5	3.5	2	3.5	1
W	6	5	3	2	4	1
WA	6	2	5	2	4	2
Average rank	5.35	3.94	3.48	2.74	3.85	1.63

Since the value of F_F is much larger than the critical value, there is a significant difference between the six algorithms. Note that, the average rank of RKNN-STSVM is far lower than the remaining algorithms. It means that our RKNN-STSVM is more valid than the other five algorithms.

4.4 Influence of parameter k

To investigate the influence of parameter k, we conduct experiments on twenty datasets from former twenty-seven datasets. The value of parameter k is chosen from the set $\{3, 4, 5, 6, 10, 20\}$.

Table 5 shows that the accuracy of our proposed RKNN-STSVM is closely related to the parameter of k, this phenomenon is more pronounced in dataset DBworld e-mail. It can be noticed that when k is gradually increased, the accuracy of DBworld e-mail is also increased. And a too large or too small value of k will both reduce the prediction accuracy. Therefore, an appropriate parameter k is very important for our model.

4.5 Image recognition datasets

We conduct experiments on Caltech image datasets, including the Caltech 101 [45] and the Caltech 256 datasets [46]. Caltech 101 dataset contains 102 categories, about 40 to 800 images per category. The size of each image is about 300×200 pixels. And the Caltech 256 dataset has 256 categories of images and at least 80 images per category. In addition, the Caltech 256 dataset has a cluttered class and the clutter category can be seen as noises or backgrounds. In our experiments, we choose Emu, Pigeon in Caltech 101 and Bonsai-101, Cactus in Caltech 256. We show some image samples in Fig. 3. Every row of image samples come from the same class.

Table 5 Performance comparisons of RKNN-STSVM with different parameter k on twenty datasets. Bold type shows the best result for each dataset

k	3	4	5	6	10	20
Datasets	Accuracy(%)	Accuracy(%)	Accuracy(%)	Accuracy(%)	Accuracy(%)	Accuracy(%)
	Time(s)	Time(s)	Time(s)	Time(s)	Time(s)	Time(s)
Australian	**87.39 ± 3.31**	86.95 ± 2.95	86.23 ± 3.10	86.23 ± 3.61	84.62 ± 5.04	81.57 ± 6.40
	0.1627	0.1661	0.1578	0.1580	0.1557	0.1583
BCI-1a	**90.34 ± 4.08**	88.48 ± 3.89	88.11 ± 4.26	87.36 ± 4.53	87.74 ± 4.71	88.11 ± 4.42
	0.2861	0.1894	0.1980	0.1803	0.1828	0.1953
Breast Cancer2	96.14 ± 2.26	95.79 ± 2.44	**96.49 ± 2.00**	96.14 ± 2.33	95.61 ± 2.93	94.91 ± 3.25
	0.1797	0.0936	0.0964	0.0971	0.0953	0.0968
Breast Cancer3	96.49 ± 3.04	96.49 ± 2.71	96.79 ± 2.59	**97.66 ± 1.62**	96.93 ± 1.81	95.03 ± 2.93
	0.1421	0.1356	0.1343	0.1331	0.1334	0.1326
Balance Scale	98.80 ± 1.02	**98.96 ± 0.85**	98.10 ± 1.60	97.92 ± 1.80	97.92 ± 1.80	96.90 ± 2.69
	0.1089	0.1160	0.1121	0.1055	0.1050	0.1095
BUPA	73.04 ± 2.84	**74.78 ± 3.85**	73.91 ± 4.40	73.62 ± 3.48	72.75 ± 3.48	71.88 ± 3.62
	0.0495	0.0540	0.0549	0.0560	0.0502	0.0558
DBworld e-mail	67.18 ± 7.50	70.38 ± 5.38	70.38 ± 5.38	75.00 ± 10.18	78.08 ± 9.07	**82.69 ± 10.46**
	0.0396	0.0400	0.0399	0.0376	0.0407	0.0379
Echocardiogram	86.22 ± 4.02	87.82 ± 4.40	**93.26 ± 4.26**	89.26 ± 5.32	89.26 ± 3.93	88.62 ± 5.35
	0.0258	0.0246	0.0266	0.0257	0.0256	0.0236
Haberman	**77.79 ± 2.08**	75.50 ± 3.66	74.52 ± 2.79	74.52 ± 2.79	75.82 ± 2.77	75.82 ± 2.35
	0.0445	0.0502	0.0474	0.0475	0.0480	0.0485
Heart	83.70 ± 3.95	**85.93 ± 3.01**	85.19 ± 2.62	84.44 ± 2.22	83.33 ± 2.62	83.70 ± 2.46
	0.0388	0.0433	0.0440	0.0378	0.0395	0.0400
Ionosphere	95.73 ± 2.39	95.73 ± 2.39	95.73 ± 2.39	95.44 ± 1.90	94.87 ± 2.14	**96.30 ± 2.14**
	0.0500	0.0537	0.0545	0.0561	0.0549	0.0540
LSVT	**89.00 ± 5.71**	87.50 ± 7.45	88.17 ± 7.20	85.83 ± 8.58	86.67 ± 8.90	85.67 ± 11.94
	0.0247	0.0274	0.0275	0.0278	0.0267	0.0267
Monks	88.51 ± 14.08	88.51 ± 14.08	88.96 ± 13.52	88.96 ± 13.52	90.36 ± 11.98	**91.75 ± 10.94**
	0.0681	0.0675	0.0760	0.0706	0.0699	0.0693
Pima	65.12 ± 5.78	65.12 ± 5.78	65.12 ± 5.78	65.12 ± 5.78	69.47 ± 6.23	**78.49 ± 3.03**
	0.2110	0.2381	0.2386	0.2081	0.2117	0.2202
Parkinson	**87.33 ± 8.41**	84.86 ± 9.46	86.28 ± 8.24	85.75 ± 7.57	83.06 ± 6.34	81.54 ± 8.49
	0.0287	0.0276	0.0302	0.0289	0.0314	0.0306
Planning Relax	69.80 ± 3.30	70.35 ± 2.36	**71.46 ± 1.52**	70.91 ± 1.64	69.82 ± 2.37	68.71 ± 2.88
	0.0290	0.0310	0.0312	0.0367	0.0308	0.0309
Sonar	69.92 ± 11.11	**72.75 ± 10.74**	70.83 ± 8.68	70.25 ± 10.20	70.67 ± 10.88	65.67 ± 9.00
	0.0302	0.0381	0.0349	0.0363	0.0293	0.0321
Spect Heart	81.30 ± 3.20	82.43 ± 3.28	**83.56 ± 4.48**	82.07 ± 3.79	80.56 ± 3.47	80.56 ± 2.79
	0.0438	0.0499	0.0445	0.0409	0.0399	0.0446
Vertebral	**84.84 ± 11.66**	83.87 ± 12.20	84.19 ± 12.88	83.87 ± 13.49	82.26 ± 14.21	82.26 ± 13.80
	0.0483	0.0502	0.0490	0.0483	0.0504	0.0524
WPBC	77.95 ± 6.61	82.93 ± 3.74	81.94 ± 4.49	**83.92 ± 2.85**	81.83 ± 4.00	81.83 ± 1.68
	0.0323	0.0323	0.0307	0.0300	0.0303	0.0339

Due to the lack of samples in the majority class in the Caltech 101 dataset, we used no more than 50 samples from each category. The Caltech 256 dataset provides more categories and images and we used no more than 80 samples from each category. As for the image feature extraction, the popular dense-sift algorithm is used to extract features form those images. Then, we quantize those features into 1000 visual-words with bag-of-words models. We also reduce

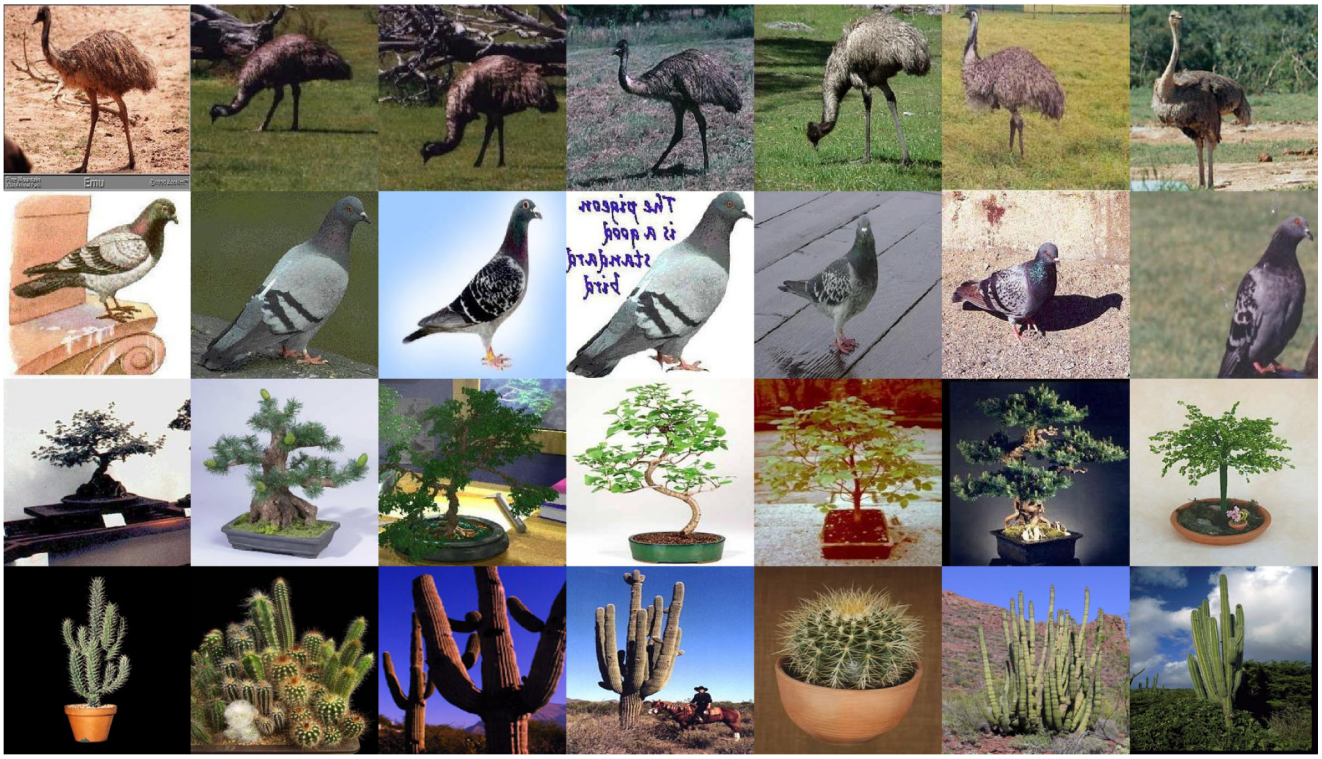

Fig. 3 Image samples in Caltech datasets

the feature vectors to appropriate dimensions with PCA, to retain 97% of the variance. Thus, we can use the kernel trick to improve classification accuracy.

The experimental results on the Caltech datasets are shown in Table 6. The experimental process and parameter selection are the same as the previous benchmark experiment. From the perspesctive of prediction accuracy in Table 6, compared with other five algorithms, we can learn that our RKNN-STSVM yields the highest testing accuracy of most datasets. In addition, our proposed RKNN-STSVM improves the classification accuracy of KNN-STSVM and ensures that the computational time is not slower than KNN-STSVM. It can be noticed that the TSVM yields the lowest testing accuracy and KNN-LSTSVM yields the lowest

Table 6 Performance comparisons of six algorithms on four image datasets. Bold type shows the best result for each dataset

Datasets	TSVM Accuracy(%) Time(s)	Iν-TBSVM Accuracy(%) Time(s)	S-TSVM Accuracy(%) Time(s)	KNN-STSVM Accuracy(%) Time(s)	KNN-LSTSVM Accuracy(%) Time(s)	RKNN-STSVM Accuracy(%) Time(s)
Emu	72.00 ± 9.79	73.00 ± 9.80	**83.00 ± 9.27**	79.00 ± 7.35	72.00 ± 7.48	82.00 ± 6.78
(100 × 363)	0.0140	0.0190	0.0411	0.0312	0.0034	0.0271
Pigeon	62.11 ± 3.94	65.26 ± 5.37	**69.47 ± 7.74**	66.32 ± 9.18	64.21 ± 9.06	**69.47 ± 7.74**
(95 × 363)	0.0161	0.0166	0.0432	0.0288	0.0030	0.0270
Bonsai-101	78.75 ± 5.38	80.00 ± 5.08	82.50 ± 3.19	83.75 ± 4.59	73.75 ± 8.29	**84.38 ± 3.42**
(160 × 392)	0.0228	0.0303	0.0707	0.0447	0.0076	0.0425
Cactus	80.00 ± 3.19	80.63 ± 5.38	80.63 ± 5.00	**81.88 ± 3.06**	80.00 ± 2.50	**81.88 ± 3.64**
(160 × 392)	0.0338	0.0166	0.0696	0.0592	0.0076	0.0442

Table 7 The optimal parameters of six algorithms used in the experiments

Dataset	TSVM (c_1, γ)	Iν-TBSVM (r_1, ν_1, γ)	S-TSVM (c_1, c_2, c_3, γ)	KNN-STSVM (c_1, c_2, γ, k)	KNN-LSTSVM (c_1, γ, k)	RKNN-STSVM $(c_1, c_2, c_3, \gamma, k)$
Emu	(1,16)	(10^{-6},0.1,16)	(1,32,10^{-4},16)	(1,0.5,4,10)	(0.5,2,4)	(0.5,256,10^{-3},16,10)
Pigeon	(1,8)	(10^{-6},0.2,128)	(0.5,0.5,0.1,4)	(256,4,16,20)	(0.5,4,3)	(0.5,128,0.1,4,10)
Bonsai-101	(0.5,2)	(10^{-2},0.7,2)	(0.5,4,10^{-3},8)	(0.5,32,2,20)	(0.5,256,20)	(0.5,128,1,2,20)
Cactus	(0.5,4)	(10^{-4},0.4,16)	(0.5,16,10^{-6},2)	(16,8,32,10)	(1,32,20)	(1,4,0.1,4,20)

computational cost. At last, our proposed RKNN-STSVM performs better than TSVM, Iν-TBSVM, KNN-LSTSVM on four image datasets. The optimal parameters of six algorithms used in the experiments are shown in Table 7.

5 The DCDM for accelerating RKNN-STSVM

5.1 The DCDM algorithm

Although Matlab's built-in algorithm "Quadprog" can give a generally precise solution, the consumption of time is expensive for large-scale problems. In this section, to further improve the solving efficiency, a novel fast algorithm dual coordinate descent method (DCDM) [38] is embedded into the learning process of our RKNN-STSVM to accelerate the learning speed. DCDM is a kind of coordinate descent method since it aims to decompose an optimization problem into a series of single-variable sub-problems. That is, in each iteration, DCDM only updates one variable. By solving these sub-problems efficiently, we can realize the optimization process in less time. The procedure to embed DCDM into our RKNN-STSVM is shown below.

One of the dual QPP of the RKNN-STSVM is as follows [38]:

$$\begin{aligned} \min_{\alpha} \quad & f(\alpha) = \frac{1}{2}\alpha^T Q\alpha - e^T\alpha \\ \text{s.t.} \quad & 0 \leqslant \alpha \leqslant C. \end{aligned} \tag{51}$$

The optimization process begins with an initial point $\alpha^0 \in R^n$. The process from α^k to α^{k+1} is referred as an outer iteration. In each outer iteration, we have n inner iterations, so that sequentially $\alpha_1, \alpha_2, \ldots, \alpha_n$ are updated. Each outer iteration generates vectors $\alpha^{k,i} \in R^n, i = 1, 2, \ldots, n+1$, such that $\alpha^{k,1} = \alpha^k$, $\alpha^{k,n+1} = \alpha^{k+1}$, and

$$\alpha^{k,i} = [\alpha_1^{k+1}, \ldots, \alpha_{i-1}^{k+1}, \alpha_i^k, \ldots, \alpha_l^k]^T, \forall i = 2, \ldots, n.$$

For updating $\alpha^{k,i}$ to $\alpha^{k,i+1}$, we solve the following one-variable sub-problem:

$$\begin{aligned} \min_{d} \quad & f(\alpha^{k,i} + de_i) \\ \text{s.t.} \quad & 0 \leqslant \alpha_i^k \leqslant C_i, \end{aligned} \tag{52}$$

where $e_i = [0, \ldots, 0, 1, 0, \ldots, 0]^T$. The objective function of (51) is a simple quadratic function of d:

$$f(\alpha^{k,i} + de_i) = \frac{1}{2}Q_{ii}d^2 + \nabla_i f(\alpha^{k,i})d + constant, \tag{53}$$

where $\nabla_i f$ is the ith component of the gradient ∇f. So (51) has an optimum at $d = 0$ (i.e., no need to update α_i) if and only if

$$\nabla_i^P f(\alpha^{k,i}) = 0, \tag{54}$$

where $\nabla^P f(\alpha)$ means the projected gradient

$$\nabla_i^P f(\alpha) = \begin{cases} \nabla_i f(\alpha), & if \quad 0 < \alpha_i < C_i, \\ min(0, \nabla_i f(\alpha)), & if \quad \alpha_i = 0, \\ max(0, \nabla_i f(\alpha)), & if \quad \alpha_i = C_i. \end{cases} \tag{55}$$

If (54) holds, we move to the index $i+1$ without updating $\alpha_i^{k,i}$. Otherwise, we must find the optimal solution of (51). If $Q_{ii} > 0$, the solution is:

$$\alpha_i^{k,i+1} = min(max(\alpha_i^{k,i} - \frac{\nabla_i f(\alpha^{k,i})}{Q_{ii}}, 0), C_i).$$

The stopping criterion we set for DCDM is $\|\alpha^{k+1} - \alpha^k\| < \epsilon$ (ϵ is a sufficient small real number) or maximum iterations reaching 1000. A flowchart of DCDM is given in Algorithm 1.

Algorithm 1 DCDM for nonlinear RKNN-STSVM.

1. Given an initial α^0
2. While α is not optimal (Outer iteration)
3. For $i = 1, 2, \ldots, n$ (Inner iteration)
   (a) $G = \nabla_i f(\alpha^{k,i}) = \sum_{j=1}^n Q_{ij}\alpha_j - 1$;
   (b)
   $$PG = \begin{cases} G, & if\ 0 < \alpha_i < C_i, \\ min(0, G), & if\ \alpha_i = 0, \\ max(0, G), & if\ \alpha_i = C_i. \end{cases}$$
   (c) If $|PG| \neq 0$,
   $$\alpha_i \leftarrow min(max(\alpha_i - \frac{PG}{Q_{ii}}, 0), C_i)$$
4. End for
5. End While

Table 8 Performance comparisons of RKNN-STSVM and DCDM+RKNN-STSVM on twenty-five datasets

Datasets	RKNN-STSVM		DCDM+RKNN-STSVM		Speedup
	Accuracy(%)	Time(s)	Accuracy(%)	Time(s)	
Australian	87.39 ± 3.31	0.1467	87.39 ± 3.31	0.1110	1.32
BCI-1a	90.34 ± 4.08	0.2011	90.34 ± 4.08	0.1708	1.18
Breast Cancer1	83.01 ± 3.77	0.0251	81.96 ± 3.25	0.0153	1.64
Breast Cancer2	98.42 ± 0.66	0.0946	96.14 ± 2.33	0.0674	1.40
Breast Cancer3	97.66 ± 1.62	0.1324	97.66 ± 1.62	0.1198	1.11
Balance Scale	98.96 ± 0.85	0.0893	98.96 ± 0.85	0.0875	1.02
BUPA	74.78 ± 3.85	0.0401	74.78 ± 3.85	0.0274	1.46
DBworld e-mail	82.69 ± 10.46	0.0387	82.69 ± 10.46	0.0209	1.85
Echocardiogram	93.26 ± 4.26	0.0211	93.26 ± 4.26	0.0114	1.85
Fertility	90.35 ± 4.26	0.0276	90.35 ± 4.26	0.0061	4.52
Hepatitis	88.79 ± 4.40	0.0317	88.79 ± 4.40	0.0055	5.76
Haberman	77.79 ± 2.07	0.0501	77.79 ± 2.07	0.0258	1.94
Heart	85.93 ± 3.01	0.0377	85.93 ± 3.01	0.0209	1.80
Ionosphere	96.30 ± 2.14	0.0446	96.30 ± 2.14	0.0289	1.54
IRIS	100.00 ± 0.00	0.0266	100.00 ± 0.00	0.0081	3.28
LSVT	89.00 ± 5.71	0.0223	89.00 ± 5.71	0.0084	2.65
Liver Disorder	75.07 ± 2.96	0.0541	75.07 ± 2.96	0.0247	2.19
Monks	91.75 ± 10.94	0.0587	100.00 ± 0.00	0.0563	1.04
Pima	78.49 ± 3.03	0.1897	78.49 ± 3.03	0.1714	1.11
Parkinson	87.33 ± 8.41	0.0232	87.33 ± 8.41	0.0131	1.77
Planning Relax	71.46 ± 1.52	0.0268	71.46 ± 1.52	0.0111	2.41
Sonar	72.75 ± 10.74	0.0384	72.75 ± 10.74	0.0186	2.06
Spect Heart	83.56 ± 4.48	0.0384	83.56 ± 4.48	0.0195	1.97
Vertebra	84.84 ± 11.66	0.0365	84.84 ± 11.66	0.0258	1.41
WPBC	83.92 ± 2.85	0.0244	82.93 ± 3.37	0.0123	1.98

5.2 Experiments of the DCDM algorithm for RKNN-STSVM

Now we introduce this DCDM algorithm into the dual QPPs of our RKNN-STSVM, which have been given by (44) and (46). Similarly, compare models (44) and (51), the matrix Q in (51) is substituted by $G_\phi(H_\phi^T D H_\phi + c_2 J_+^\phi + c_3 I)^{-1} G_\phi^T$. Notably, we have known that the $F = diag(f_1^{(2)}, f_2^{(2)}, \ldots, f_{l_2}^{(2)})$ in our QPP is a diagonal matrix and $f_j^{(2)}$ is either 0 or 1. That is to say, the component of α corresponding to 0 in F has no contribution to the optimization problem (44), so it can be ignored during the iteration of DCDM, which will further improve the computing speed.

In this experiment, twenty-five datasets are selected to validate the significance of applying this DCDM algorithm to speed up the operation. This initial value of α is a random vector between 0 and c_1, and the stopping condition in the DCDM algorithm is chosen as [38]

$\|\alpha^{k+1} - \alpha^k\| < \epsilon$, or maximum iterations reaching 1000.

We set $\epsilon = 10^{-2}$ in the experiments. Averages and standard deviations of test accuracies (in %) and computation time derived by RKNN-STSVM and RKNN-STSVM solved with DCDM are shown in Table 8. The last column shows the speedup of DCDM algorithm. These results indicate that DCDM algorithm gives similar performance on testing accuracy with the Matlab built-in algorithm, while it makes our RKNN-STSVM obtain a faster learning speed, the largest speedup almost reaches to 5.76 times.

6 Conclusion

In this paper, we present an improved version of KNN-STSVM. We reformulate the primal problems of RKNN-STSVM by adding regularization terms in the objective

functions to utilize the structural risk minimization principle and to remove singularity in the solution. Therefore, our RKNN-STSVM avoids the over-fitting problem to a certain degree and yields higher testing accuracy. Numerical experiments on twenty-seven benchmark datasets and image recognition datasets demonstrate the feasibility and validity of our RKNN-STSVM. The DCDM fast algorithm is further employed to speed up our RKNN-STSVM. The selection of an appropriate parameter k can greatly improve the testing accuracy of our algorithms. It should be pointed out that there are several parameters in our RKNN-STSVM, so it is meaningful to address the parameter selection in future work.

Acknowledgments The authors gratefully acknowledge the helpful comments of the reviewers, which have improved the presentation. This work was supported in part by National Natural Science Foundation of China (No.11671010) and Beijing Natural Science Foundation (No. 4172035).

References

1. Vapnik V (1995) The nature of statistical learning theory. Springer, New York
2. Xu Y, Wang L (2005) Fault diagnosis system based on rough set theory and support vector machine. Lect Notes Comput Sci 3614:980–988
3. Zhang W, Yoshida T, Tang X (2008) Text classification based on multi-word with support vector machine. Knowl-Based Syst 21(8):879–886
4. Zhang C, Bi J, Xu S, Ramentol E, Fan G, Qiao B, Fujita H (2019) Multi-imbalance: an open-source software for multi-class imbalance learning. Knowl-Based Syst 174:137–143
5. Jayadeva KhemchandaniR, Chandra S (2007) Twin support vector machines for pattern classification. IEEE Trans Pattern Anal Mach Intell 29(5):905–910
6. Mangasarian O, Wild E (2005) Multisurface proximal support vector machine classification via generalized eigenvalues. IEEE Trans Pattern Anal Mach Intell 28(1):69–74
7. Kumar M, Gopal M (2008) Application of smoothing technique on twin support vector machines. Pattern Recognit Lett 29(13):1842–1848
8. Peng X (2010) TSVR: an efficient twin support vector machine for regression. Neural Netw: the Official Journal of the International Neural Network Society 23(3):365–372
9. Xu Y, Wang L (2012) A weighted twin support vector regression. Knowl-Based Syst 33:92–101
10. Xu Y, Wang L (2014) K-nearest neighbor-based weighted twin support vector regression. Appl Intell 41(1):299–309
11. Zhao J, Xu Y, Fujita H (2019) An improved non-parallel Universum support vector machine and its safe sample screening rule. Knowl-Based Syst 170:79–88
12. Shao Y, Zhang C, Wang X, Deng N (2011) Improvements on twin support vector machines. IEEE Trans Neural Netw 22(6):962–968
13. Tian Y, Ju X, Qi Z, Shi Y (2014) Improved twin support vector machine. Sci China Mater 57(2):417–432
14. Wang H, Zhou Z, Xu Y (2018) An improved ν-twin bounded support vector machine. Appl Intell 48(4):1041–1053
15. Yang Z, Wu H, Li C, Shao Y (2016) Least squares recursive projection twin support vector machine for multi-class classification. Int J Mach Learn Cybern 7(3):411–426
16. Tanveer M, Khan M, Ho S (2016) Robust energy-based least squares twin support vector machines. Appl Intell 45(1):174–186
17. Khemchandani R, Saigal P, Chandra S (2016) Improvements on ν-twin support vector machine. Neural Netw 79:97–107
18. Tanveer M (2015) Application of smoothing techniques for linear programming twin support vector machines. Knowl Inf Syst 45(1):191–214
19. Shao Y, Wang Z, Chen W, Deng N (2013) Least squares twin parametric-margin support vector machine for classification. Appl Intell 39(3):451–464
20. Shao Y, Deng N, Yang Z (2012) Least squares recursive projection twin support vector machine for classification. Pattern Recognit 45(6):2299–2307
21. Yeung D, Wang D, Ng W, Tsang E, Wang X (2007) Structured large margin machines: sensitive to data distributions. Mach Learn 68(2):171–200
22. Xue H, Chen S, Yang Q (2011) Structural regularized support vector machine: a framework for structural large margin classifier. IEEE Trans Neural Netw 22(4):573–587
23. Qi Z, Tian Y, Shi Y (2013) Structural twin support vector machine for classification. Knowl-Based Syst 43:74–81
24. Xu Y, Pan X, Zhou Z, Yang Z, Zhang Y (2015) Structural least square twin support vector machine for classification. Appl Intell 42(3):527–536
25. Ye Q, Zhao C, Gao S, Zheng H (2012) Weighted twin support vector machines with local information and its application. Neural Netw 35:31–39
26. Pan X, Luo Y, Xu Y (2015) K-nearest neighbor based structural twin support vector machine. Knowl-Based Syst 88:34–44
27. Mir A, Nasiri J (2018) KNN-based least squares twin support vector machine for pattern classification. Appl Intell 48(12):4551–4564
28. Tanveer M, Shubham K, Aldhaifallah M, Ho S (2016) An efficient regularized k-nearest neighbor-based weighted twin support vector regression. Knowl-Based Syst 94:70–87
29. Thongkam J, Xu G, Zhang Y, Huang F (2008) Support vector machine for outlier detection in breast cancer survivability prediction. Adv Web Network Technol Appl 4977:99–109
30. Cui W, Yan X (2009) Adaptive weighted least square support vector machine regression integrated with outlier detection and its application in QSAR. Chemom Intell Lab Syst 98(2):130–135
31. Xu Y, Liu C (2013) A rough margin-based one class support vector machine. Neural Comput Appl 22(6):1077–1084
32. Shao Y, Wang Z, Chen W, Deng N (2013) A regularization for the projection twin support vector machine. Knowl-Based Syst 37:203–210
33. Shao Y, Zhang C, Yang Z, Jing L, Deng N (2013) An ϵ-twin support vector machine for regression. Neural Comput Appl 23(1):175–185
34. Tanveer M (2015) Robust and sparse linear programming twin support vector machines. Cogn Comput 7(1):137–149
35. Osuna E, Freund R, Girosi F (1997) An improved training algorithm for support vector machines. Neural Netw Signal Process VII(17):276–285
36. Platt J (1999) Fast training of support vector machines using sequential minimal optimization. Advances in kernel methods. MIT Press, Cambridge
37. Mavroforakis M, Theodoridis S (2006) A geometric approach to support vector machine (svm) classification. IEEE Trans Neural Netw 17(3):671–682

38. Hsieh C, Chang K, Lin C, Keerthi S, Sundararajan S (2008) A dual coordinate descent method for large-scale linear SVM. In: Proceedings of the international conference on machine learning, vol 9. ACM, pp 408–415
39. Ward J (1963) Hierarchical grouping to optimize an objective function. Publ Am Stat Assoc 58(301):236–244
40. Salvador S, Chan P (2004) Determining the number of clusters/segments in hierarchical clustering/segmentation algorithms. In: 16th IEEE international conference on tools with artificial intelligence. Proceedings, pp 576–584
41. Xue H, Chen S, Yang Q (2009) Discriminatively regularized least-squares classification. Pattern Recognit 42(1):93–104
42. Chang C, Lin C (2011) LIBSVM: a library for support vector machines. ACM
43. Ar J (2006) Statistical comparisons of classifiers over multiple data sets. J Mach Learn Res 7:1–30
44. Salvador G, Alberto F, Julián L, Herrera F (2010) Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: experimental analysis of power. Inf Sci 180(10):2044–2064
45. Feifei L, Fergus R, Perona P (2006) One-shot learning of object categories. IEEE Trans Pattern Anal Mach Intell 28(4):594–611
46. Griffin G, Holub A, Perona P (2006) The Caltech 256, Caltech Technical Report

Fan Xie was born in 1994 in China. He received the BS degree in College of Science from China Agricultural University in 2018. Now he is a graduate student in College of Science from China Agricultural University. His research interests include machine learning and data mining.

Yitian Xu received the Ph.D. degree from the College of Science, China Agricultural University, Beijing, China, in 2007.

He was a Visiting Scholar with the Department of Computer Science and Engineering, Arizona State University, Tempe, AZ, USA, from 2013 to 2014. He is currently a Professor at the College of Science, China Agricultural University. He has authored about 50 papers. His current research interests include machine learning and data mining.

Prof. Xu's research has appeared in IEEE Transactions on Neural Networks and Learning Systems, IEEE Transactions on Cybernetics, Information Science, Pattern Recognition, Knowledge-Based Systems, Neurocomputing, and so on.