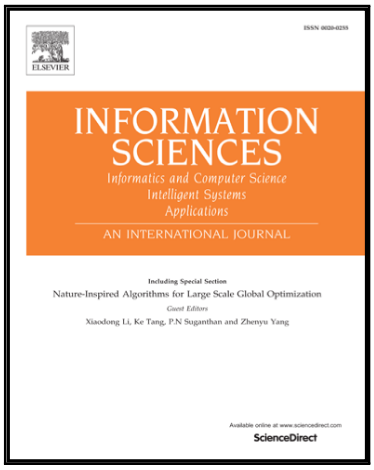
Accepted Manuscript



Insensitive Stochastic Gradient Twin Support Vector Machines for Large Scale Problems

Zhen Wang, Yuan-Hai Shao, Lan Bai, Chun-Na Li, Li-Ming Liu, Nai-Yang Deng

PII: S0020-0255(18)30450-X
DOI: 10.1016/j.ins.2018.06.007
Reference: INS 13700

To appear in: *Information Sciences*

Received date: 18 November 2017
Revised date: 30 May 2018
Accepted date: 3 June 2018

Please cite this article as: Zhen Wang, Yuan-Hai Shao, Lan Bai, Chun-Na Li, Li-Ming Liu, Nai-Yang Deng, Insensitive Stochastic Gradient Twin Support Vector Machines for Large Scale Problems, *Information Sciences* (2018), doi: 10.1016/j.ins.2018.06.007

Insensitive Stochastic Gradient Twin Support Vector Machines for Large Scale Problems

Zhen Wang[a], Yuan-Hai Shao[b,*], Lan Bai[a], Chun-Na Li[c], Li-Ming Liu[d], Nai-Yang Deng[e]

[a]*School of Mathematical Sciences, Inner Mongolia University, Hohhot, 010021, P.R.China*
[b]*School of Economics and Management, Hainan University, Haikou, 570228, P.R. China*
[c]*Zhijiang College, Zhejiang University of Technology, Hangzhou, 310024, P.R. China*
[d]*School of Statistics, Capital University of Economics and Business, Beijing, 100070, P.R.China*
[e]*College of Science, China Agricultural University, Beijing, 100083, P.R.China*

Abstract

Within the large scale classification problem, the stochastic gradient descent method called PEGASOS has been successfully applied to support vector machines (SVMs). In this paper, we propose a stochastic gradient twin support vector machine (SGTSVM) based on the twin support vector machine (TWSVM). Compared to PEGASOS, our method is insensitive to stochastic sampling. Furthermore, we prove the convergence of SGTSVM and the approximation between TWSVM and SGTSVM under uniform sampling, whereas PEGASOS is almost surely convergent and only has an opportunity to obtain an approximation to SVM. In addition, we extend SGTSVM to nonlinear classification problems via a kernel trick. Experiments on artificial and publicly available datasets show that our method has stable performance and can handle large scale problems easily.

Keywords: Classification, support vector machine, twin support vector machine, stochastic gradient descent, large scale problem.

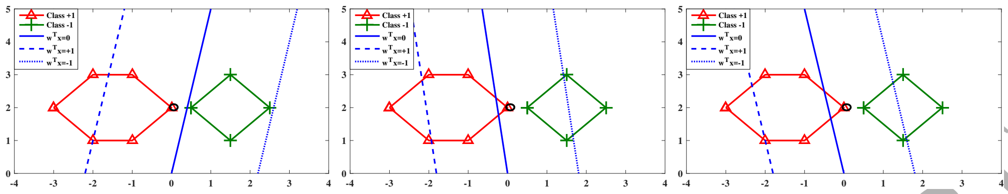
*Corresponding author. Tel./Fax:(+86)0571-87313551.
Email address: `shaoyuanhai21@163.com` (Yuan-Hai Shao)

1. Introduction

As a powerful classification tool, support vector machines (SVMs) [4, 42] have been widely used in various practical problems [19, 14, 9]. SVM searches parallel hyperplanes with the maximum margin between them to achieve classification. By dropping the parallelism condition, the twin support vector machine (TWSVM) [10, 33], which uses a pair of nonparallel hyperplanes, has been proposed. Benefiting from the nonparallel hyperplanes, TWSVM classifies some different types of heterogeneous data better than SVM. Therefore, TWSVM has been deeply studied and enhanced, resulting in the development of, e.g., the twin bounded support vector machine (TBSVM) [33], twin parametric margin support vector machine (TPMSVM) [22] and weighted Lagrangian twin support vector machine (WLTSVM) [31]. These classifiers have been widely applied in many practical problems [32, 39, 17, 38, 3, 30, 26, 25, 24].

Due to both SVM and TWSVM needing to solve quadratic programming problems (QPPs), it is difficult for these techniques to handle large scale problems [21, 36]. To accelerate the training of SVM, many improvements have been proposed. On the one hand, sequential minimal optimization (SMO) [23, 2], successive over-relaxation (SOR) [18] and the dual coordinate descent method (DCD) [6] were proposed to solve the dual problem of SVM. Correspondingly, these methods were also generalized to solve the dual problems of TWSVM [33, 35, 32]. However, the dual solutions of TWSVM cannot effectively address large scale problems because computation of the inverse of a large matrix is needed for all such solutions. On the other hand, the smooth Newton method [15] and the stochastic gradient descent algorithm (SGD) [43, 29, 41] were proposed to solve the primal problem of SVM, and the smooth Newton method has also been generalized to solve the primal problems of TWSVM [13, 39]. Although the smooth Newton method has a second-order convergence rate, it needs to calculate and store a large Hessian matrix or its approximation and hence is also difficult to apply to solving large scale problems.

In contrast, the SGD solver that partitions a large scale problem into a series of sub-problems by stochastic sampling has a surprisingly high learning speed with a very small memory requirement [8, 34, 37]. The SGD solver for SVM, called PEGASOS [29], stochastically selects only one sample at each iteration and merely needs a single vector multiplication without additional computations. PEGASOS has been successfully applied to large scale prob-

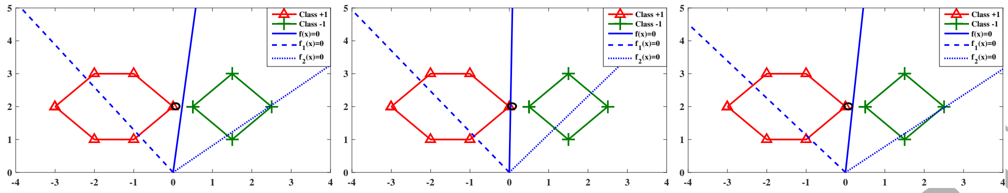


(a) (b) (c)

Figure 1: PEGASOS applied to 10 samples from two classes. (i) Training includes all 10 samples with 11 iterations, and the circle sample is used twice; (ii) training includes all 10 samples with 28 iterations, and the circle sample is used once; (iii) training includes 9 samples with 27 iterations, and the circle sample is excluded.

lems [34, 20, 27]. However, PEGASOS is defective in theory and practical application in the following sense: it has only been proven that PEGASOS is almost surely convergent and that it can find an approximation of SVM with a certain probability [1, 43, 29]. It is worth noting that PEGASOS does not contain the bias term b. The authors of PEGASOS proposed another model by adding a bias term to PEGASOS; however, this modification led to the problem of non-strong convexity and thus yielded a slow convergence rate [29]. Furthermore, it is well known that support vectors (SVs) are very important to SVM and that SVs directly determine the final classifier. However, stochastic sampling in PEGASOS may not adequately sample SVs, thus losing its generalization ability.

Therefore, this paper proposes an insensitive stochastic gradient twin support vector machine (SGTSVM) based on TWSVM. Our SGTSVM selects two samples at each iteration stochastically to construct a pair of nonparallel hyperplanes. Compared to SVM, TWSVM fits the entire set of training samples, i.e., TWSVM is robust to sampling, and the final classifier is not dependent on certain specific samples (such as SVs) [10, 33]. Thus, our SGTSVM is insensitive to sampling, and its generalization ability is more robust than that of PEGASOS. Moreover, we theoretically prove the convergence of our method and that under uniform sampling, our method is a good approximation to TWSVM. In addition, SGTSVM also inherits the advantages of TWSVM, such as the ability to handle a "cross planes" dataset [10]. Due to SGTSVM being very efficient in both calculation and storage, it is currently the fastest method among the TWSVM-type classifiers for large scale problems.



(a) (b) (c)

Figure 2: SGTSVM applied to 10 samples from two classes. (i) Training includes all 10 samples with 7 iterations, and the circle sample is used twice; (ii) training includes all 10 samples with 16 iterations, and the circle sample is used once; (iii) training includes 9 samples with 15 iterations, and the circle sample is excluded.

To show the influence of stochastic sampling on PEGASOS and SGTSVM, we perform an experiment on a toy example shown in Figs. 1 and 2. There are two classes in these figures, where the positive and negative classes contain 6 samples and 4 samples, respectively. The circle-enclosed sample is a potential SV. The blue solid lines are the final classification lines obtained by PEGASOS and SGTSVM. We use three methods to calculate the classification lines: (i) the potential SV is selected many times; (ii) the potential SV is only selected once; and (iii) the potential SV is not selected. The results shown in Fig. 1 show that the potential SV plays an important role in PEGASOS. If the potential SV is not selected or is infrequently selected in PEGASOS, the classification line deviates from the ideal classification position. On the other hand, Fig. 2 shows that even if the potential SV is not selected, this aspect has less influence on the classification line of SGTSVM. Therefore, SGTSVM is less sensitive to sampling than PEGASOS.

In summary, the main contributions of this paper include the following:
(i) An insensitive SGD-based TWSVM (SGTSVM) is proposed; this method can be easily extended to other TWSVM-type classifiers.
(ii) The convergence of SGTSVM is theoretically proven.
(iii) For uniform sampling, we prove that the optimal solution of SGTSVM is bounded by the optimal solution of TWSVM; therefore, our method is a good approximation of TWSVM.
(iv) SGTSVM is extended to the nonlinear case via a kernel trick.
(v) Experimental results show that our SGTSVM is more stable than PEGASOS and can handle large scale problems efficiently.

The rest of this paper is organized as follows. Section 2 briefly reviews

SVM, PEGASOS and TWSVM. Our linear and nonlinear SGTSVMs together with the theoretical analysis are elaborated in Section 3. Experiments are presented in Section 4. Section 5 concludes the paper.

2. Related Works

Consider a binary classification problem in the n-dimensional real space R^n. The set of training samples is represented by $X \in R^{n\times m}$, where $x \in R^n$ is the sample with the label $y \in \{+1, -1\}$. We further organize m_1 samples of Class $+1$ into a matrix $X_1 \in R^{n\times m_1}$ and m_2 samples of Class -1 into a matrix $X_2 \in R^{n\times m_2}$. Below, we give a brief outline of several related methods.

2.1. SVM

A support vector machine (SVM) [4] seeks a separating hyperplane

$$w^\top x + b = 0, \tag{1}$$

where $w \in R^n$ and $b \in R$. The separating hyperplane is determined by a pair of parallel supporting hyperplanes $w^\top x + b = \pm 1$ by considering the following QPP:

$$\begin{array}{ll} \min\limits_{w,b,\xi} & \frac{1}{2}||w||^2 + \frac{c}{m}e^\top \xi \\ \text{s.t.} & D(X^\top w + b) \geq e - \xi, \quad \xi \geq 0, \end{array} \tag{2}$$

where $||\cdot||$ denotes the L_2 norm, $c > 0$ is a parameter with certain quantitative meanings [4], e is a vector of ones with an appropriate dimension, $\xi \in R^m$ is the slack vector, and $D = \text{diag}(y_1, \ldots, y_m)$. Note that the minimization of the regularization term $\|w\|^2$ is equivalent to maximizing the margin of the pair of parallel supporting hyperplanes $w^\top x + b = \pm 1$. Additionally, the structural risk minimization principle is implemented in this problem [4].

Once the solution to (2) has been obtained, a new sample x can be predicted by

$$y = \text{sign}(w^\top x + b). \tag{3}$$

2.2. PEGASOS

PEGASOS [29] considers a strongly convex problem by modifying (2) as

$$\begin{array}{ll} \min\limits_{w,\xi} & \frac{1}{2}||w||^2 + \frac{c}{m}e^\top \xi \\ s.t. & DX^\top w \geq e - \xi, \xi \geq 0 \end{array} \tag{4}$$

and recasts the above problem to

$$\min_{w} \ \tfrac{1}{2}||w||^2 + \tfrac{c}{m}e^\top(e - DX^\top w)_+, \tag{5}$$

where $(\cdot)_+$ replaces the negative components of a vector with zeros.

PEGASOS solves the above problem iteratively. In the t-th iteration ($t \geq 1$), PEGASOS constructs a temporary function defined by a random sample $x_t \in X$ as

$$g_t(w) = \tfrac{1}{2}||w||^2 + c(1 - y_t w^\top x_t)_+. \tag{6}$$

Then, starting with an initial w_1, PEGASOS iteratively updates $w_{t+1} = w_t - \eta_t \nabla_{w_t} g_t(w)$ for $t \geq 1$, where $\eta_t = 1/t$ is the step size, $\nabla_{w_t} g_t(w)$ is the sub-gradient of $g_t(w)$ at w_t, and

$$\nabla_{w_t} g_t(w) = w_t - c y_t x_t \text{sign}(1 - y_t w_t^\top x_t)_+. \tag{7}$$

When certain termination conditions are satisfied, the last w_t is output as w. Additionally, a new sample x is predicted by

$$y = \text{sign}(w^\top x). \tag{8}$$

It has been proven that the average solution $\bar{w} = \frac{1}{T}\sum_{t=1}^{T} w_t$ is bounded by the optimal solution w^* to (5) with $o(1)$, and thus, PEGASOS has a probability of at least $1/2$ to find a good approximation of w^* [29]. The authors of [29] also noted that w_T is often used instead of $\bar{w}$ in practice. The sample x_t that is selected randomly can be replaced with a small subset belonging to the whole dataset, and the subset only including a sample is often used in practice [43, 29, 41]. To extend the generalization ability of PEGASOS, the bias term b in SVM can be appended to PEGASOS by replacing $g(w_t)$ of (6) with

$$g(w_t, b) = \tfrac{1}{2}||w_t||^2 + C(1 - y_t(w_t^\top x_t + b))_+. \tag{9}$$

However, this modification leads to the function not being strongly convex, thus yielding a slow convergence rate [29].

2.3. TWSVM

TWSVM [10, 33] seeks a pair of nonparallel hyperplanes in R^n, which can be expressed as

$$w_1^\top x + b_1 = 0 \ \text{ and } \ w_2^\top x + b_2 = 0, \tag{10}$$

such that each hyperplane is close to the samples of one class and has a certain distance from the other class. To find the pair of nonparallel hyperplanes, it is necessary to obtain solutions to the primal problems

$$\begin{array}{ll}\min\limits_{w_1,b_1,\xi_1} & \frac{1}{2}(||w_1||^2+b_1^2)+\frac{c_1}{2m_1}\|X_1^\top w_1+b_1\|^2+\frac{c_2}{m_2}e^\top\xi_1\\ \text{s.t.} & X_2^\top w_1+b_1-\xi_1\leq -e,\quad \xi_1\geq 0\end{array} \tag{11}$$

and

$$\begin{array}{ll}\min\limits_{w_2,b_2,\xi_2} & \frac{1}{2}(||w_2||^2+b_2^2)+\frac{c_3}{2m_2}\|X_2^\top w_2+b_2\|^2+\frac{c_4}{m_1}e^\top\xi_2\\ \text{s.t.} & X_1^\top w_2+b_2+\xi_2\geq e,\quad \xi_2\geq 0,\end{array} \tag{12}$$

where c_1, c_2, c_3, and c_4 are positive parameters, and $\xi_1 \in R^{m_2}$ and $\xi_2 \in R^{m_1}$ are slack vectors. Their geometric meanings are clear. For instance, the objective function of (11) makes the samples of Class +1 proximal to the hyperplane $w_1^\top x + b_1 = 0$ together with the regularization term, while the constraints make each sample of Class -1 have a distance of greater than $1/||w_1||$ from the hyperplane $w_1^\top x + b_1 = -1$.

Once solutions (w_1, b_1) and (w_2, b_2) to problems (11) and (12), respectively, have been obtained, a new sample x is assigned to a class depending on the distances to the hyperplanes of (10), i.e.,

$$y = \arg\min_i \frac{|w_i^\top x+b_i|}{\|w_i\|}, \tag{13}$$

where $|\cdot|$ denotes obtaining the absolute value.

3. SGTSVM

In this section, we describe our SGTSVM and provide its theoretical analysis.

3.1. Linear Formulation

Our SGTSVM aims at solving the QPPs (11) and (12) in TWSVM. Note that these QPPs are equivalent to the unconstrained problems

$$\min_{w_1,b_1} \frac{1}{2}(||w_1||^2+b_1^2)+\frac{c_1}{2m_1}||X_1^\top w_1+b_1||^2+\frac{c_2}{m_2}e^\top(e+X_2^\top w_1+b_1)_+ \tag{14}$$

and

$$\min_{w_2,b_2} \frac{1}{2}(||w_2||^2+b_2^2)+\frac{c_3}{2m_2}||X_2^\top w_2+b_2||^2+\frac{c_4}{m_1}e^\top(e-X_1^\top w_2-b_2)_+, \tag{15}$$

respectively.

To solve the above two problems, we construct a series of strictly convex functions $f_{1,t}(w_1, b_1)$ and $f_{2,t}(w_2, b_2)$ with $t \geq 1$ as follows:

$$f_{1,t} = \frac{1}{2}(||w_1||^2 + b_1^2) + \frac{c_1}{2}||w_1^\top x_t + b_1||^2 + c_2(1 + w_1^\top \hat{x}_t + b_1)_+, \tag{16}$$

and

$$f_{2,t} = \frac{1}{2}(||w_2||^2 + b_2^2) + \frac{c_3}{2}||w_2^\top \hat{x}_t + b_2||^2 + c_4(1 - w_2^\top x_t - b_2)_+, \tag{17}$$

where x_t and $\hat{x}_t$ are selected randomly from X_1 and X_2, respectively. The subgradients of the above functions at $(w_{1,t}, b_{1,t})$ and $(w_{2,t}, b_{2,t})$ can be obtained by

$$\begin{array}{l} \nabla_{w_{1,t}} f_{1,t} = w_{1,t} + c_1(w_{1,t}^\top x_t + b_{1,t})x_t + c_2\hat{x}_t \text{sign}(1 + w_{1,t}^\top \hat{x}_t + b_{1,t})_+, \\ \nabla_{b_{1,t}} f_{1,t} = b_{1,t} + c_1(w_{1,t}^\top x_t + b_{1,t}) + c_2\text{sign}(1 + w_{1,t}^\top \hat{x}_t + b_{1,t})_+ \end{array} \tag{18}$$

and

$$\begin{array}{l} \nabla_{w_{2,t}} f_{2,t} = w_{2,t} + c_3(w_{2,t}^\top \hat{x}_t + b_{2,t})\hat{x}_t - c_4 x_t \text{sign}(1 - w_{2,t}^\top x_t - b_{2,t})_+, \\ \nabla_{b_{2,t}} f_{2,t} = b_{2,t} + c_3(w_{2,t}^\top \hat{x}_t + b_{2,t}) - c_4\text{sign}(1 - w_{2,t}^\top x_t - b_{1,t})_+, \end{array} \tag{19}$$

respectively.

Our SGTSVM starts from the initial $(w_{1,1}, b_{1,1})$ and $(w_{2,1}, b_{2,1})$. Then, for $t \geq 1$, the updates are given by

$$\begin{array}{l} w_{1,t+1} = w_{1,t} - \eta_t \nabla_{w_{1,t}} f_{1,t}, \\ b_{1,t+1} = b_{1,t} - \eta_t \nabla_{b_{1,t}} f_{1,t}, \\ w_{2,t+1} = w_{2,t} - \eta_t \nabla_{w_{2,t}} f_{2,t}, \\ b_{2,t+1} = b_{2,t} - \eta_t \nabla_{b_{2,t}} f_{2,t}, \end{array} \tag{20}$$

where η_t is the step size, set typically at $1/t$. If certain termination conditions are satisfied, $(w_{1,t}, b_{1,t})$ is assigned to (w_1, b_1), and $(w_{2,t}, b_{2,t})$ is assigned to (w_2, b_2). Then, a new sample $x \in R^n$ can be predicted by (13).

The above steps are summarized in Algorithm 1.

Algorithm 1 Linear SGTSVM

Input: Positive class $X_1 \in R^{n\times m_1}$, negative class $X_2 \in R^{n\times m_2}$, positive parameters c_1, c_2, c_3, c_4 and a small tolerance tol; typically, $tol = 10^{-3}$.
Output: w_1, b_1, w_2 and b_2.
1. Set $w_{1,1}$, $b_{1,1}$, $w_{2,1}$ and $b_{2,1}$ to be zero;
2. **For** $t = 1, \dots,$
(a) choose a pair of samples x_t and $\hat{x}_t$ at random from X_1 and X_2, respectively;
(b) compute the gradients using (18) to update $(w_{1,t+1}, b_{1,t+1})$ and/or (19) to update $(w_{2,t+1}, b_{2,t+1})$ by (20);
(c) if $||w_{1,t+1} - w_{1,t}|| + |b_{1,t+1} - b_{1,t}| < tol$, stop updating $w_{1,t+1}$ and $b_{1,t+1}$;
(d) if $||w_{2,t+1} - w_{2,t}|| + |b_{2,t+1} - b_{2,t}| < tol$, stop updating $w_{2,t+1}$ and $b_{2,t+1}$;
(e) if all $w_{1,t+1}$, $b_{1,t+1}$, $w_{2,t+1}$ and $b_{2,t+1}$ are no longer being updated, end this loop and go to step 3;
3. Set $w_1 = w_{1,t+1}$, $b_1 = b_{1,t+1}$, $w_2 = w_{2,t+1}$ and $b_2 = b_{2,t+1}$.

3.2. Nonlinear Formulation

Now, we extend our SGTSVM to the nonlinear case via a kernel trick [10, 33, 12, 16]. Suppose that $K(\cdot,\cdot)$ is the predefined kernel function; then, the nonparallel hyperplanes in the kernel-generated space can be expressed as

$$K(x, X)^\top w_1 + b_1 = 0 \quad \text{and} \quad K(x, X)^\top w_2 + b_2 = 0. \tag{21}$$

The counterparts of (14) and (15) can be formulated as

$$\min_{w_1,b_1} \ \tfrac{1}{2}(||w_1||^2 + b_1^2) + \tfrac{c_1}{2m_1}||K(X_1, X)^\top w_1 + b_1||^2 + \tfrac{c_2}{m_2} e^\top (e + K(X_2, X)^\top w_1 + b_1)_+ \tag{22}$$

and

$$\min_{w_2,b_2} \ \tfrac{1}{2}(||w_2||^2 + b_2^2) + \tfrac{c_3}{2m_2}||K(X_2, X)^\top w_2 + b_2||^2 + \tfrac{c_4}{m_1} e^\top (e - K(X_1, X)^\top w_2 - b_2)_+. \tag{23}$$

Let $K_t = K(x_t, X)$ and $\hat{K}_t = K(\hat{x}_t, X)$. Then, we construct a series of functions with $t \geq 1$ as follows:

$$h_{1,t} = \tfrac{1}{2}(||w_1||^2 + b_1^2) + \tfrac{c_1}{2}||K_t^\top w_1 + b_1||^2 + c_2(1 + \hat{K}_t^\top w_1 + b_1)_+, \tag{24}$$

and

$$h_{2,t} = \tfrac{1}{2}(||w_2||^2 + b_2^2) + \tfrac{c_3}{2}||\hat{K}_t^\top w_2 + b_2||^2 + c_4(1 - K_t^\top w_2 - b_2)_+. \tag{25}$$

Similar to (18), (19) and (20), the sub-gradients and updates are as follows:

$$\begin{array}{l}\nabla_{w_{1,t}} h_{1,t} = w_{1,t} + c_1(K_t^\top w_{1,t} + b_{1,t})K_t + c_2\hat{K}_t \text{sign}(1 + \hat{K}_t^\top w_{1,t} + b_{1,t})_+,\\ \nabla_{b_{1,t}} h_{1,t} = b_{1,t} + c_1(K_t^\top w_{1,t} + b_{1,t}) + c_2 \text{sign}(1 + \hat{K}_t^\top w_{1,t} + b_{1,t})_+,\end{array} \tag{26}$$

$$\begin{array}{l}\nabla_{w_{2,t}} h_{2,t} = w_{2,t} + c_3(\hat{K}_t^\top w_{2,t} + b_{2,t})\hat{K}_t - c_4 K_t \text{sign}(1 - K_t^\top w_{2,t} - b_{2,t})_+,\\ \nabla_{b_{2,t}} h_{2,t} = b_{2,t} + c_3(\hat{K}_t^\top w_{2,t} + b_{2,t}) - c_4 \text{sign}(1 - K_t^\top w_{2,t} - b_{1,t})_+,\end{array} \tag{27}$$

and

$$\begin{array}{l} w_{1,t+1} = w_{1,t} - \nabla_{w_{1,t}} h_{1,t}/t,\\ b_{1,t+1} = b_{1,t} - \nabla_{b_{1,t}} h_{1,t}/t,\\ w_{2,t+1} = w_{2,t} - \nabla_{w_{2,t}} h_{2,t}/t,\\ b_{2,t+1} = b_{2,t} - \nabla_{b_{2,t}} h_{2,t}/t.\end{array} \tag{28}$$

A new sample $x \in R^n$ is predicted by

$$y = \arg\min_i \frac{|K(x,X)^\top w_i + b_i|}{\|w_i\|}. \tag{29}$$

The nonlinear SGTSVM is summarized in Algorithm 2.

Algorithm 2 Nonlinear SGTSVM

Input: Positive class $X_1 \in R^{n\times m_1}$, negative class $X_2 \in R^{n\times m_2}$, positive parameters c_1, c_2, c_3, c_4, kernel function $K(\cdot,\cdot)$ and a small tolerance tol; typically, $tol = 10^{-3}$.
Output: w_1, b_1, w_2 and b_2.
1. Set $w_{1,1}$, $b_{1,1}$, $w_{2,1}$ and $b_{2,1}$ to be zero;
2. **For** $t = 1, \ldots,$
(a) choose a pair of samples x_t and $\hat{x}_t$ at random from X_1 and X_2, respectively, and compute $K_t = K(x_t, X)$ and $\hat{K}_t = K(\hat{x}_t, X)$;
(b) compute the tth gradients using (26) to update $(w_{1,t+1}, b_{1,t+1})$ and/or (27) to update $(w_{2,t+1}, b_{2,t+1})$ by (28);
(c) if $||w_{1,t+1} - w_{1,t}|| + |b_{1,t+1} - b_{1,t}| < tol$, stop updating $w_{1,t+1}$ and $b_{1,t+1}$;
(d) if $||w_{2,t+1} - w_{2,t}|| + |b_{2,t+1} - b_{2,t}| < tol$, stop updating $w_{2,t+1}$ and $b_{2,t+1}$;
(e) if all $w_{1,t+1}$, $b_{1,t+1}$, $w_{2,t+1}$ and $b_{2,t+1}$ are no longer being updated, end this loop and go to step 3;
3. Set $w_1 = w_{1,t+1}$, $b_1 = b_{1,t+1}$, $w_2 = w_{2,t+1}$ and $b_2 = b_{2,t+1}$.

For large scale problems, it is time consuming to calculate the kernel $K(\cdot, X)$. However, the reduced kernel strategy, which has been successfully applied to SVM and TWSVM [16, 40, 39], can also be applied to our SGTSVM. This strategy replaces $K(\cdot, X)$ with $K(\cdot, \tilde{X})$, where $\tilde{X}$ is a randomly sampled subset of X. In practice, $\tilde{X}$ needs only $0.01\% \sim 1\%$ of samples from X to obtain a good performance, reducing the learning time without loss of generalization [40].

3.3. Analysis

In this subsection, we discuss two issues: (i) the convergence of Algorithm 1 and (ii) the relationship between the solution in SGTSVM and the optimal one in TWSVM. For convenience, we only consider the first QPP (14) of the linear TWSVM together with the SGD formulation of the linear SGTSVM. The conclusions for another QPP (15) and the nonlinear algorithm can be obtained similarly.

Let $u = (w_1^\top, b_1)^\top$, $Z_1 = (X_1^\top, e)^\top$, $Z_2 = (X_2^\top, e)^\top$ and $z = (x^\top, 1)^\top$; the notations with the subscripts in SGTSVM also comply with these definitions. Then, the first QPP (14) is reformulated as

$$\min_{u} \quad f(u) = \tfrac{1}{2}||u||^2 + \tfrac{c_1}{2m_1}||Z_1 u||^2 + \tfrac{c_2}{m_2} e^\top (e + Z_2 u)_+. \tag{30}$$

Next, we reformulate the t-th ($t \geq 1$) function in SGTSVM as

$$f_t(u) = \tfrac{1}{2}||u||^2 + \tfrac{c_1}{2}||u^\top z_t||^2 + c_2(1 + u^\top \hat{z}_t)_+, \tag{31}$$

where z_t and $\hat{z}_t$ are the samples selected randomly from Z_1 and Z_2, respectively, for the t-th iteration. The sub-gradient of $f_t(u)$ at u_t is denoted by

$$\nabla_t = u_t + c_1(u_t^\top z_t) z_t + c_2 \hat{z}_t \mathrm{sign}(1 + u_t^\top \hat{z}_t)_+. \tag{32}$$

Given u_1 and the step size $\eta_t = 1/t$, u_{t+1} for $t \geq 1$ is updated by

$$u_{t+1} = u_t - \eta_t \nabla_t, \tag{33}$$

i.e.,

$$u_{t+1} = (1 - \tfrac{1}{t}) u_t - \tfrac{c_1}{t} z_t z_t^\top u_t - \tfrac{c_2}{t} \hat{z}_t \mathrm{sign}(1 + u_t^\top \hat{z}_t)_+. \tag{34}$$

To prove the convergence of our SGTSVM, we consider the boundedness of $||u_t||$ first. Intuitively, if $||u_t||$ does not have an upper bound, this immediately results in the non-convergence of SGTSVM. In fact, we have the following lemma.

Lemma 3.1. The sequences $\{||\nabla_t|||t = 1, 2, \ldots\}$ and $\{||u_t|||t = 1, 2, \ldots\}$ have upper bounds.

Proof. The formulation (34) can be rewritten as

$$u_{t+1} = A_t u_t + \tfrac{1}{t} v_t, \tag{35}$$

where $A_t = \frac{1}{t}((t-1)I - c_1 z_t z_t^\top)$, I is the identity matrix, and $v_t = -c_2 \hat{z}_t \text{sign}(1 + u_t^\top \hat{z}_t)_+$. Note that for a sufficiently large t, there is a positive integer N such that for $t > N$, A_t is positive definite, and the largest eigenvalue λ_t of A_t is smaller than or equal to $\frac{t-1}{t}$. Based on (35), we have

$$u_{t+1} = \prod_{i=N+1}^{t} A_{t+N+1-i} u_{N+1} + \sum_{i=N+1}^{t} \tfrac{1}{i} (\prod_{j=i+1}^{t} A_{t+i+1-j}) v_i. \tag{36}$$

For $i \geq N+1$, $||A_{t+N+1-i} u_{N+1}|| \leq \lambda_i ||u_{N+1}|| \leq \frac{i-1}{i} ||u_{N+1}||$ [7]. Therefore,

$$|| \prod_{i=N+1}^{t} A_{t+N+1-i} u_{N+1}|| \leq \tfrac{N}{t} ||u_{N+1}||, \tag{37}$$

and

$$|| \tfrac{1}{i} (\prod_{j=i+1}^{t} A_{t+i+1-j}) v_i|| \leq \tfrac{1}{t} \max_{i \leq t} ||v_i||. \tag{38}$$

Thus, we have

$$\begin{aligned} ||u_{t+1}|| &\leq \tfrac{N}{t} ||u_{N+1}|| + \tfrac{t-N}{t} \max_{i \leq t} ||v_i|| \\ &\leq ||u_{N+1}|| + c_2 \max_{z \in Z_2} ||z||. \end{aligned} \tag{39}$$

Let M be the largest norm of the samples in the dataset and

$$G_1 = \max\{\max\{||u_1||, \ldots, ||u_N||\}, ||u_{N+1}|| + c_2 M\}. \tag{40}$$

This leads to G_1 being an upper bound of $||u_t||$ and $G_2 = G_1 + c_1 G_1 M^2 + c_2 M$ being an upper bound of $||\nabla_t||$. □

Now, we can establish convergence of our SGTSVM.

Theorem 3.1. The iterative formulation (34) is convergent.

Proof. On the one hand, from (37) in the proof of Lemma 3.1, we have

$$\lim_{t\to\infty} \|\prod_{i=N+1}^{t} A_{t+N+1-i} u_{N+1}\| = 0, \tag{41}$$

which indicates that

$$\lim_{t\to\infty} \prod_{i=N+1}^{t} A_{t+N+1-i} u_{N+1} = 0. \tag{42}$$

On the other hand, from (38), we have

$$\sum_{i=N+1}^{t} \|\tfrac{1}{i}(\prod_{j=i+1}^{t} A_{t+i+1-j}) v_i\| \leq M, \tag{43}$$

which indicates that

$$\lim_{t\to\infty} \sum_{i=N+1}^{t} \|\tfrac{1}{i}(\prod_{j=i+1}^{t} A_{t+i+1-j}) v_i\| < \infty. \tag{44}$$

Note that an infinite series of vectors is convergent if its norm series is convergent [28]. Therefore, the following limit exists:

$$\lim_{t\to\infty} \sum_{i=N+1}^{t} \tfrac{1}{i}(\prod_{j=i+1}^{t} A_{t+i+1-j}) v_i < \infty. \tag{45}$$

Combining (42) with (45), we conclude that the series of u_{t+1} is convergent if $t \to \infty$. □

The above theorem states that the first of two iterative problems in Algorithm 1 is convergent. The same conclusion can be obtained easily for the other problem for the nonlinear case. Thus, we immediately have the following:

Theorem 3.2. Algorithms 1 and 2 are convergent.

Theorem 3.1 shows that the termination conditions of Algorithms 1 and 2 are reasonable. Moreover, the initialization $u_1 = 0$ in these algorithms is shown to be reasonable by noting that

$$u_{t+1} = \prod_{i=1}^{t} A_{t+1-i} u_1 + \sum_{i=1}^{t} \tfrac{1}{i}(\prod_{j=i+1}^{t} A_{t+i+1-j}) v_i, \tag{46}$$

as it speeds up convergence of these algorithms.

Before analyzing the relationship between the solution u_t in SGTSVM and the optimal solution $u^* = (w^{*\top}, b^*)^\top$ in TWSVM, we give a generalized conclusion for the iterative formulation used in SGTSVM.

Lemma 3.2. Let $f_1, \ldots, f_T$ be a sequence of convex functions and $u_1, \ldots, u_{T+1} \in R^n$ be a sequence of vectors. For $t \geq 1$, $u_{t+1} = u_t - \eta_t \nabla_t$, where ∇_t belongs to the sub-gradient set of f_t at u_t, and $\eta_t = 1/t$. Suppose that $||u_t||$ and $||\nabla_t||$ have upper bounds G_1 and G_2, respectively. Then, for all $\theta \in R^n$, we have

(i) $\frac{1}{T}\sum\limits_{t=1}^{T} f_t(u_t) \leq \frac{1}{T}\sum\limits_{t=1}^{T} f_t(\theta) + G_2(G_1 + ||\theta||) + \frac{1}{2T}G_2^2(1 + \ln T)$;

(ii) given any $\varepsilon > 0$, for a sufficiently large T, $\frac{1}{T}\sum\limits_{t=1}^{T} f_t(u_t) \leq \frac{1}{T}\sum\limits_{t=1}^{T} f_t(\theta) + \varepsilon$.

Proof. As f_t is convex and ∇_t is the sub-gradient of f_t at u_t, we have

$$f_t(u_t) - f_t(\theta) \leq (u_t - \theta)^\top \nabla_t. \tag{47}$$

Note that

$$(u_t - \theta)^\top \nabla_t = \frac{1}{2\eta_t}(||u_t - \theta||^2 - ||u_{t+1} - \theta||^2) + \frac{\eta_t}{2}||\nabla_t||^2. \tag{48}$$

Combining (47) and (48), we have

$$\begin{array}{ll} & \sum\limits_{t=1}^{T}(f_t(u_t) - f_t(\theta)) \\ \leq & \frac{1}{2}\sum\limits_{t=1}^{T}\frac{1}{\eta_t}(||u_t - \theta||^2 - ||u_{t+1} - \theta||^2) + \frac{1}{2}\sum\limits_{t=1}^{T}(\eta_t||\nabla_t||^2) \\ = & \frac{1}{2}(\sum_{t=1}^{T}||u_t - \theta||^2 - T||u_{T+1} - \theta||^2) + \frac{1}{2}\sum_{t=1}^{T}(\eta_t||\nabla_t||^2) \\ \leq & (G_1 + ||\theta||)\sum\limits_{t=1}^{T}||u_{T+1} - u_t|| + \frac{1}{2}G_2^2(1 + \ln T) \\ = & (G_1 + ||\theta||)\sum\limits_{t=1}^{T}||\sum\limits_{i=t}^{T}\frac{1}{i}\nabla_i|| + \frac{1}{2}G_2^2(1 + \ln T) \\ \leq & TG_2(G_1 + ||\theta||) + \frac{1}{2}G_2^2(1 + \ln T). \end{array} \tag{49}$$

Multiplying (49) by $1/T$ leads to conclusion (i).

Furthermore, assuming that $\lim\limits_{T\to\infty} u_T = \tilde{u}$, we have $\lim\limits_{T\to\infty} ||u_T|| = ||\tilde{u}||$. Then, $\lim\limits_{T\to\infty}\frac{1}{T}\sum\limits_{t=1}^{T}||u_t - \theta|| = \lim\limits_{T\to\infty}||u_T - \theta|| = ||\tilde{u} - \theta||$. Note that $\lim\limits_{T\to\infty}\frac{G_2^2(1+lnT)}{T} =$

0. Given any $\varepsilon > 0$, for a sufficiently large T,

$$\begin{aligned} & \frac{1}{T}\sum_{t=1}^{T}(f_t(u_t) - f_t(\theta)) \\ \leq\ & \frac{1}{2}(\frac{1}{T}\sum_{t=1}^{T}||u_t - \theta||^2 - ||u_{T+1} - \theta||^2) + \frac{1}{2T}G_2^2(1 + lnT) \\ \leq\ & \frac{1}{2}\varepsilon + \frac{1}{2}\varepsilon = \varepsilon. \end{aligned} \tag{50}$$

□

The above lemma shows that the average convex functions' value w.r.t. an arbitrary sequence of variables is bounded by the corresponding average value w.r.t. an arbitrary constant. Because SGTSVM satisfies the conditions of this lemma, we straightforwardly obtain the same boundedness for SGTSVM as follows.

Theorem 3.3. For f_t $(t = 1, \dots, T)$ defined by (31) in SGTSVM, u_t $(t = 1, \dots, T)$ is constructed by (34), and u^* is the optimal solution to (30). Then, (i) there are two constants G_1 and G_2 (in fact, they are the upper bounds of $||u_t||$ and $||\nabla_t||$, respectively) such that $\frac{1}{T}\sum\limits_{t=1}^{T} f_t(u_t) \leq \frac{1}{T}\sum\limits_{t=1}^{T} f_t(u^*) + G_2(G_1 + ||u^*||) + \frac{1}{2T}G_2^2(1 + \ln T)$;
(ii) given any $\varepsilon > 0$, for a sufficiently large T, $\frac{1}{T}\sum\limits_{t=1}^{T} f_t(u_t) \leq \frac{1}{T}\sum\limits_{t=1}^{T} f_t(u^*) + \varepsilon$.

Recall that the average instantaneous objective of SGTSVM correlates with the objective of TWSVM. We may estimate the relation between the solutions of SGTSVM and TWSVM under certain special conditions. For instance, for uniform sampling, we have the following desirable conclusion.

Corollary 3.1. Assume that the conditions stated in Theorem 3.1 are satisfied and $m_1 = m_2$, where m_1 and m_2 are the sample numbers of X_1 and X_2, respectively. Suppose that $T = km_1$, where $k > 0$ is an integer, and each sample is selected k times at random. Then,
(i) $f(u_T) \leq f(u^*) + G_2(G_1 + ||u^*|| + G_2) + \frac{1}{2T}G_1^2(1 + \ln T)$;
(ii) given any $\varepsilon > 0$, for a sufficiently large T, $f(u_T) \leq f(u^*) + G_2^2 + \varepsilon$.

Proof. First, we prove that for all $i, j = 1, 2, \dots, T$,

$$|f_t(u_i) - f_t(u_j)| \leq G_2||u_i - u_j||, \quad t = 1, 2, \dots, T. \tag{51}$$

From the formulation of $f_t(u)$, we have

$$\begin{array}{rl} |f_t(u_i) - f_t(u_j)| & \leq \frac{1}{2}|||u_i||^2 - ||u_j||^2| \\ & +\frac{c_1}{2}|(u_i^\top z_t)^2 - (u_j^\top z_t)^2| \\ & +c_2|(1 + u_i^\top \hat{z}_t)_+ - (1 + u_j^\top \hat{z}_t)_+|. \end{array} \tag{52}$$

As G_1 is the upper bound of $||u_t||$ $(t \geq 1)$ and M is the largest norm of samples in the dataset, the first, second and third parts on the right-hand side of (52) are

$$\frac{1}{2}|||u_i||^2 - ||u_j||^2| \leq G_1||u_i - u_j||, \tag{53}$$

$$\begin{array}{rl} & \frac{c_1}{2}|(u_i^\top z_t)^2 - (u_j^\top z_t)^2| \\ = & \frac{c_1}{2}|(u_i + u_j)^\top z_t(u_i - u_j)^\top z_t| \\ \leq & c_1 G_1 M^2||u_i - u_j||, \end{array} \tag{54}$$

and

$$\begin{array}{rl} & c_2|(1 + u_i^\top \hat{z}_t)_+ - (1 + u_j^\top \hat{z}_t)_+| \\ = & c_2|(u_i - u_j)^\top \hat{z}_t| \\ \leq & c_2 M||u_i - u_j||, \end{array} \tag{55}$$

respectively. Therefore, there is a constant $G_2 = G_1 + c_1 G_1 M^2 + c_2 M$ satisfying (51).

Second, from $u_{t+1} = u_t - \frac{1}{t}\nabla_t$, it is easy to obtain

$$u_{t+1} = u_1 - \sum_{i=1}^{t} \frac{1}{i}\nabla_t, \quad t = 1, 2, \ldots, T. \tag{56}$$

Thus, for $1 \leq i < j \leq T$,

$$||u_i - u_j|| = ||\sum_{t=i}^{j-1} \frac{1}{t}\nabla_t|| \leq \sum_{t=i}^{j-1} \frac{1}{t} G_2. \tag{57}$$

As $T = km_1 = km_2$, for all $u \in R^n$, $\frac{1}{T}\sum_{t=1}^{T} f_t(u) = f(u)$. Note that $f(u)$ is the objective of TWSVM. Based on (51) and (57), we have

$$\begin{array}{rl} & f(u_T) - \frac{1}{T}\sum_{t=1}^{T} f_t(u_t) \\ = & \frac{1}{T}\sum_{t=1}^{T}(f_t(u_T) - f_t(u_t)) \\ \leq & \frac{1}{T}\sum_{t=1}^{T} G_2||u_T - u_t|| \\ \leq & \frac{G_2^2(T-1)}{T} \\ \leq & G_2^2. \end{array} \tag{58}$$

Finally, by using Theorem 3.3, we reach the conclusion immediately. □

If $m_1 \neq m_2$, we can modify the sampling rule to obtain the same result as that in Corollary 3.1.

Corollary 3.2. Assume that the conditions stated in Corollary 3.1 are satisfied, but $m_1 \neq m_2$. Suppose that $T = kd(m_1, m_2)$, where $k > 0$ is an integer, and d is the least common multiple of m_1 and m_2. The sample in X_1 is selected kd/m_1 times at random, and that in X_2 is selected kd/m_2 times at random. Then,
(i) $f(u_T) \leq f(u^*) + G_2(G_1 + ||u^*|| + G_2) + \frac{1}{2T}G_1^2(1 + \ln T)$;
(ii) given any $\varepsilon > 0$, for a sufficiently large T, $f(u_T) \leq f(u^*) + G_2^2 + \varepsilon$.

Note that for all $u \in R^n$, $\frac{1}{T}\sum_{t=1}^{T} f_t(u) = f(u)$. The proof of the above corollary is similar to that of Corollary 3.1.

As the inequality $f(u^*) \leq f(u_T)$ always holds, the above two corollaries provide the approximations of u^* by u_T. If the sampling rule is not as stated in these corollaries, these upper bounds no longer hold. However, Kakade and Tewari [11] have shown a way to obtain similar bounds with a high probability.

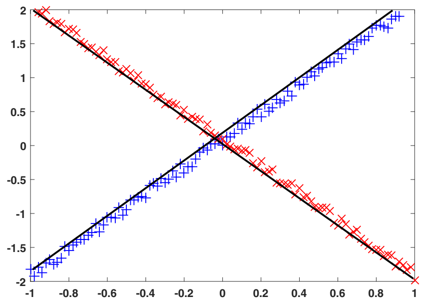
4. Experiments

In the experiments, we compared our SGTSVM to SVM [4], PEGASOS [29], and TWSVM [10, 33] applied to several artificial and publicly available datasets. All methods were implemented on a PC with an Intel Core Duo processor (3.4 GHz) with 4 GB of RAM.

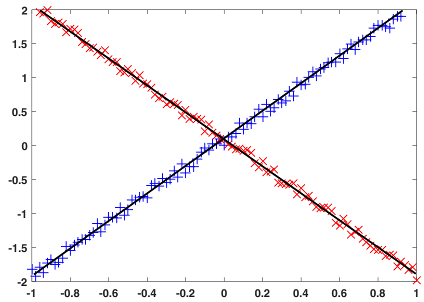
4.1. Benchmark datasets

For application to the benchmark datasets, SVM, PEGASOS, TWSVM and our SGTSVM were implemented in Matlab. The corresponding SGTSVM Matlab source code is available at `http://www.optimal-group.org/Resources/Code/SGTSVM.html`.

First, we consider the similarity between TWSVM and SGTSVM. These two methods were implemented on the "cross planes" dataset, where TWSVM was superior [10]. Fig. 3 shows the proximal lines on the dataset. It is clear that the two proximal lines obtained by SGTSVM are similar to those obtained by TWSVM; hence, TWSVM and SGTSVM can precisely capture the



(a) TWSVM



(b) SGTSVM

Figure 3: Results of TWSVM and SGTSVM on the "cross planes", where the black solid lines are $w_1^\top x + b_1 = 0$ and $w_2^\top x + b_2 = 0$.

Table 1: The mean accuracy (%) and standard deviation of TWSVM and SGTSVM attained by 10-fold cross validation.

Dataset	TWSVM†	SGTSVM†	TWSVM$^\sharp$	SGTSVM$^\sharp$
Cross Planes	96.05±0.70	97.71±0.41	99.01±2.24	98.51±2.15
Australia	86.87±0.38	87.34±0.13	87.10±0.43	85.21±0.16
Creadit	85.78±0.32	85.72±0.23	86.71±0.33	85.21±0.45
Hypothyroid	98.21±0.09	97.28±0.01	98.08±0.09	98.07±0.03

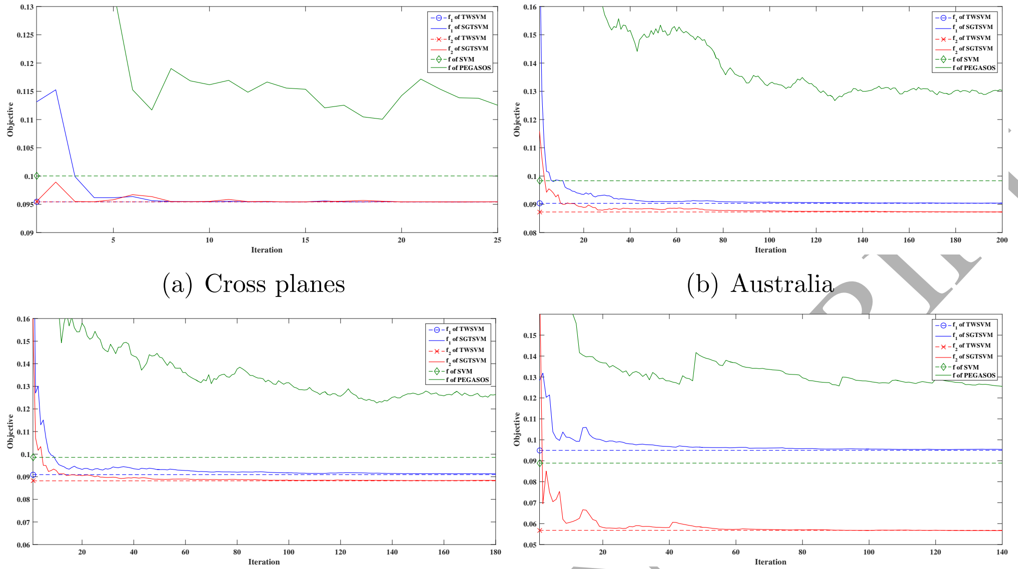
†*linear case*;$^\sharp$*nonlinear case.*

data distribution, and thus, both of them obtain good classifiers. To measure the similarity quantitatively, 10-fold cross validation [5] was used on the "cross planes" and several UCI datasets (`http://archive.ics.uci.edu/ml/index.php`, e.g., the Australia dataset that includes 690 samples with 14 features, the Creadit dataset that includes 690 samples with 15 features, and the Hypothyroid dataset that includes $3,163$ samples with 25 features). The linear TWSVM, SGTSVM, and their nonlinear versions were implemented, with the Gaussian kernel $K(x,y) = \exp\{-\mu||x-y||^2\}$ being used for nonlinear versions. We ran TWSVM and SGTSVM 10 times and report the mean accuracy and standard deviation in Table 1. The differences in the mean accuracy values are at most 2% between the two methods, implying that the classifiers obtained by TWSVM and SGTSVM do not have significant differences.

The following test compares the optimums between TWSVM and SGTSVM together with SVM and PEGASOS. The optimums f_1 of (11) and f_2 of (12) in TWSVM and f of (4) were calculated and compared to those of each iteration in SGTSVM and PEGASOS run on these datasets. Parameters c_1, c_2, c_3, c_4 and μ were fixed at 0.1. Fig. 4 shows results from the linear classifiers, while Fig. 5 corresponds to the nonlinear case. In Figs. 4 and 5, the horizontal axis denotes the iteration of SGTSVM and PEGASOS, while the vertical axis denotes the objectives of these methods. Due to the objectives of TWSVM and SVM being constant, they are denoted by the horizontal dashed lines, while the objectives of SGTSVM and PEGASOS for each iteration are denoted by the solid lines in these figures. It can be observed that the number of iterations needed for our SGTSVM to converge to TWSVM varies with the dataset. For instance, the linear SGTSVM converges to TWSVM after 20 iterations in Fig. 4 (a), while convergence appears in Fig. 4 (b) after 180 iterations. Generally, SGTSVM converges to TWSVM after 150 iterations on these datasets for both linear and nonlinear cases. However, PEGASOS does not converge to SVM within 200 iterations, indicating that our SGTSVM converges much faster than PEGASOS. Moreover, the objectives of PEGASOS fluctuate within 200 iterations; hence, PEGASOS needs to run many more iterations to obtain a stable solution, while the same does not apply to SGTSVM.

4.2. Artificial datasets

Second, we test the stability of SGTSVM compared to PEGASOS on several artificial datasets. One hundred datasets were generated randomly,

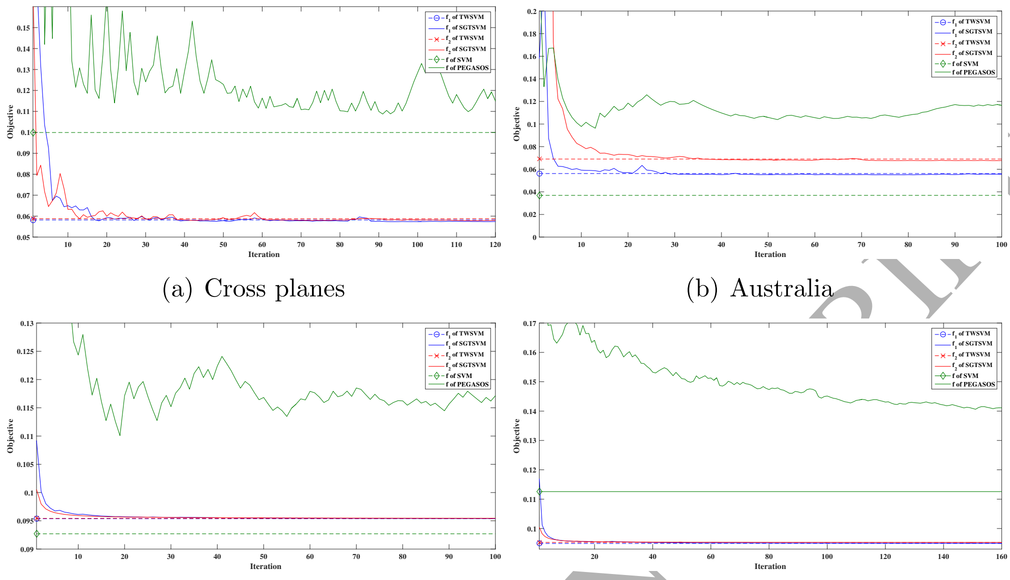


(a) Cross planes (b) Australia

(c) Creadit (d) Hypothyroid

Figure 4: Results of linear TWSVM and SGTSVM applied to the four datasets, where the vertical axis denotes the objectives of f_1 and f_2.

with each containing $10,000$ samples in R, where $5,000$ negative samples were from a normal distribution $N(-2,1)$ and $5,000$ positive ones were from $N(2,1)$. The best classification point is at zero. We applied PEGASOS and SGTSVM to the 100 datasets and obtained 100 classifiers, as shown in Fig. 6, where the numbers in the upper right corner represent the mean of the classifiers and their standard deviation (parameters c in PEGASOS and c_1, c_2, c_3 and c_4 in SGTSVM were fixed at 0.1). It is clear that our SGTSVM obtains a much more compact set of classification lines than does PEGASOS. The mean line of SGTSVM is at -0.0016, which is closer to zero and has a smaller standard deviation than that for PEGASOS. To investigate the effect of sampling, PEGASOS and SGTSVM were applied to the above 100 datasets with restricted sampling (i.e., some possible SVs from the negative samples in SVM and the samples close to these SVs were made invisible to sampling). Fig. 7 shows the results of PEGASOS and SGTSVM, where the dashed line denotes that the samples in the corresponding range are invisible to sampling. Fig. 7 shows that the classification lines obtained

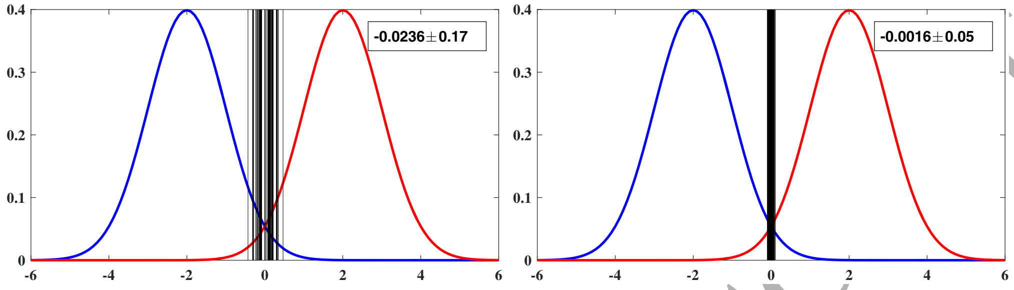


(a) Cross planes (b) Australia

(c) Creadit (d) Hypothyroid

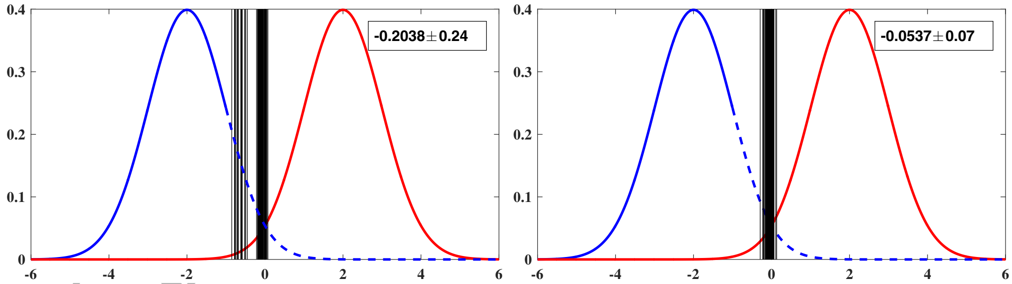
Figure 5: Results of nonlinear TWSVM and SGTSVM applied to the four datasets, where the vertical axis is the same as that in Fig. 4.

by PEGASOS belong to two regions, while SGTSVM obtains a compact region. Thus, this result indicates that the possible SVs significantly influence PEGASOS, while SGTSVM is comparatively reliant on the data distribution. According to Figs. 6 and 7, PEGASOS always results in a mean classification line further from zero and with a larger standard deviation than SGTSVM. Therefore, SGTSVM is more stable than PEGASOS on these datasets with or without the restricted sampling. To further show the classifiers' stability, we recorded the classification accuracies (%) of PEGASOS and SGTSVM on one of the 100 datasets. PEGASOS and SGTSVM were applied 100 times to this dataset, with parameters set as before, and the two methods were iterated 200 times. The accuracies of these methods are reported in Fig. 8. According to Fig. 8, the accuracies of SGTSVM are in the range of $[99.0, 99.5]$, while the values for PEGASOS are within $[96.5, 99.5]$, indicating that SGTSVM is more stable than PEGASOS from the perspective of the classification result. Although PEGASOS obtains the highest accuracy in this test, SGTSVM obtains a higher accuracy than PEGASOS in most cases.



(a) PEGASOS (b) SGTSVM

Figure 6: Results of PEGASOS and SGTSVM applied to 100 artificial datasets, where the 100 vertical black solid lines are the final classifiers.



(a) PEGASOS (b) SGTSVM

Figure 7: Results of PEGASOS and SGTSVM applied to 100 artificial datasets, where the 100 vertical black solid lines are the final classifiers, and the samples along the dashed line are invisible to sampling.

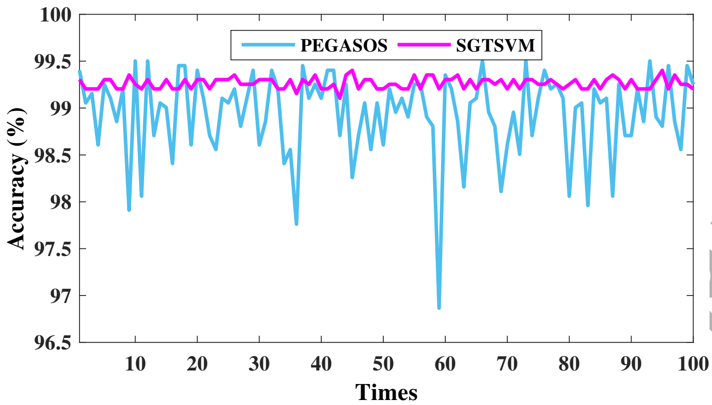


Figure 8: Accuracies of PEGASOS and SGTSVM applied to a normally distributed dataset, where each method was implemented 100 times.

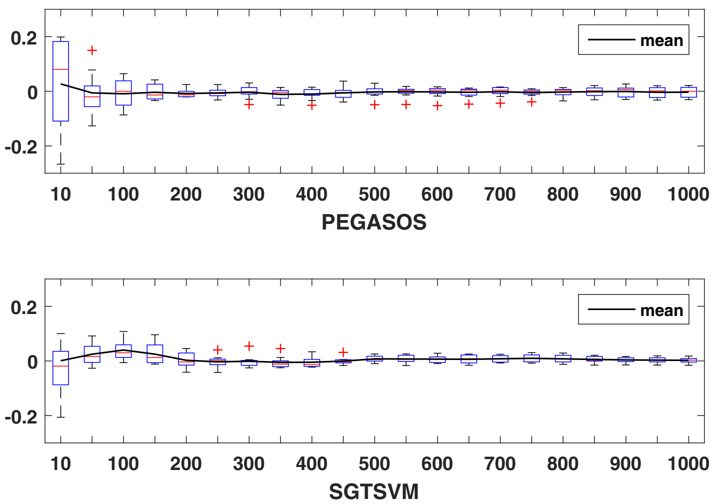


Figure 9: Results of PEGASOS and SGTSVM applied to a normally distributed dataset, where each method was implemented 10 times. The horizontal axis shows the iteration count, while the vertical axis represents the classification location.

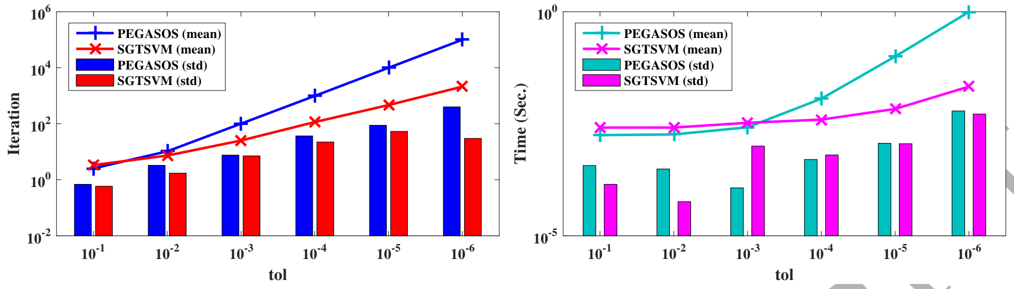


Figure 10: The number of iterations and running time of PEGASOS and SGTSVM on a normally distributed dataset, where each method was implemented 100 times.

Finally, we test the convergence of PEGASOS and SGTSVM. A dataset containing $20,000$ samples in R was generated randomly, with $10,000$ negative samples being from a normal distribution $N(-2,1)$ and $10,000$ positive ones being from $N(2,1)$. PEGASOS and SGTSVM were implemented 10 times, and each method was iterated $1,000$ times. The current classification locations for various iterations are reported in Fig. 9, where the horizontal axis shows the iteration count, and the vertical axis represents the classification location. Fig. 9 shows that (i) the initially selected samples do not affect either PEGASOS or SGTSVM after iterating 150 times; (ii) after iterating 100 times, the classification locations of the two methods center around zero, and the error is less than 0.1; and (iii) PEGASOS obtains a higher error than SGTSVM after iterating 800 times, which is important, indicating that PEGASOS converges slower than SGTSVM. To explore convergence more precisely, PEGASOS and SGTSVM were implemented 100 times, and each method was terminated based on the solution error parameter tol (more details about tol can be found in Algorithms 3.1 and 3.2). Parameter tol was selected from $\{10^i|i=-1,-2,\ldots,-6\}$, and the corresponding number of iterations and the time cost are reported in Fig. 10. It is clear from Fig. 10 that our SGTSVM converges faster than PEGASOS if $tol \leq 10^{-3}$. Moreover, if one needs a smaller solution error, such as $tol = 10^{-4}$ or $tol = 10^{-5}$, PEGASOS would need approximately 10 times as many iterations as SGTSVM, and the ratio of required iterations would be 100 if $tol = 10^{-6}$ (thus, the learning times of PEGASOS and SGTSVM differ by more than a hundredfold). Therefore, SGTSVM converges much faster than PEGASOS.

Table 2: The details of large scale datasets.

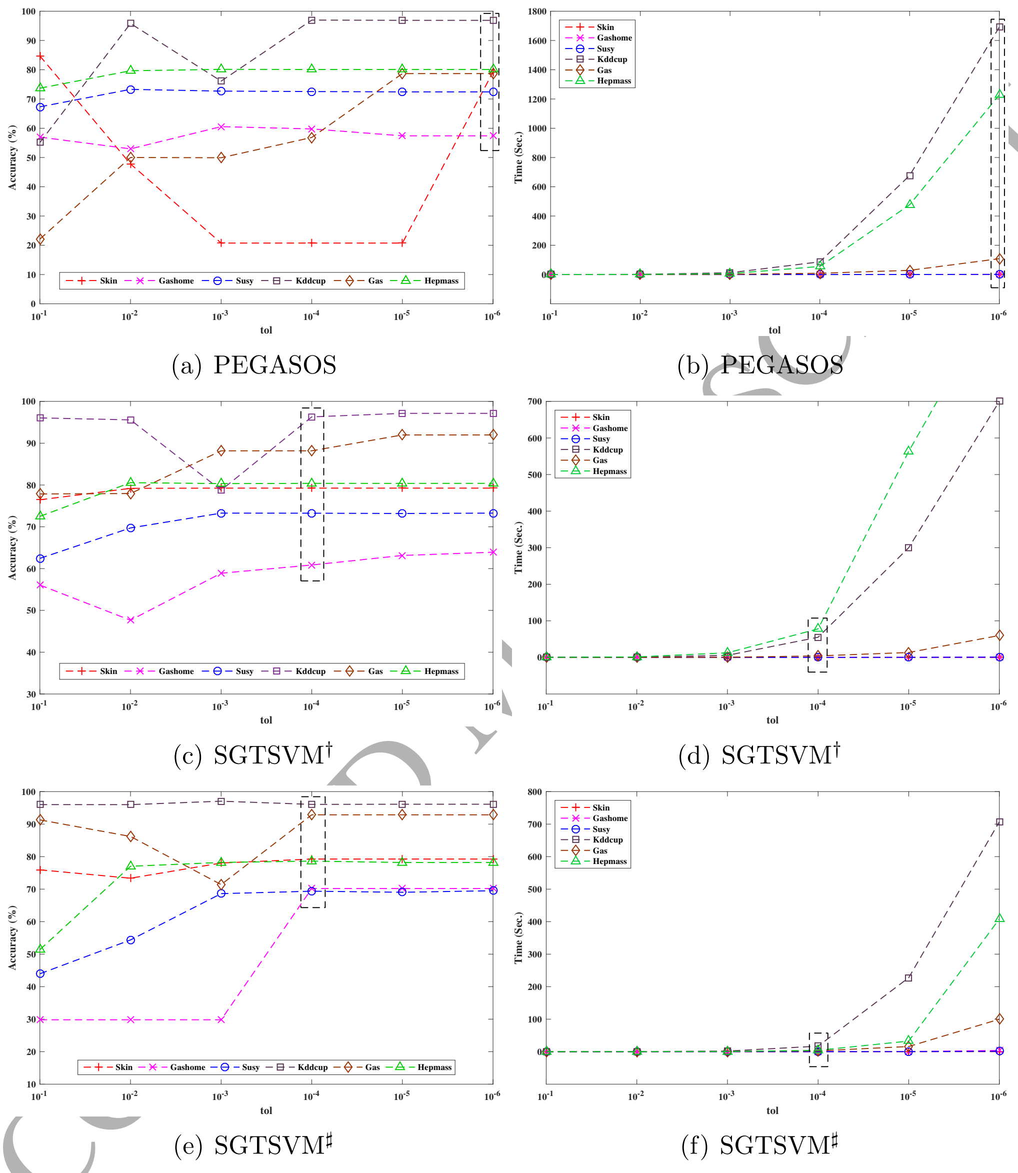
Dataset	Name	No. of samples	Dimension	Ratio
(a)	Skin	245,057	3	0.262
(b)	Gashome	928,990	10	0.578
(c)	Susy	5,000,000	18	0.844
(d)	Kddcup	4,898,432	41	0.248
(e)	Gas	8,386,764	16	0.077
(f)	Hepmass	10,500,000	28	1.000

4.3. Large scale datasets

To test the feasibility of these methods on large scale datasets, we ran SVM, PEGASOS, and SGTSVM on six large scale datasets (`http://archive.ics.uci.edu/ml/index.php`). Table 2 shows the details of the large scale datasets, where Ratio is the ratio of the number of samples in the positive class to that in the negative class. Each dataset is split into two subsets, with one (including 90% of samples) used for training and the other (including 10% of samples) for testing. SVM was implemented by Liblinear [6], while PEGASOS and SGTSVM were implemented by software programs written in the C language. The corresponding software programs can be downloaded from `http://www.optimal-group.org/Resources/Code/SGTSVM.html`. For the nonlinear SGTSVM, the reduced kernel [16] was used, and the kernel size was fixed at 100.

First, let us test the influence of parameter *tol* on PEGASOS and SGTSVM. These methods were implemented on large scale datasets, with *tol* selected from $\{10^i|i = -1, -2, \ldots, -6\}$ and other parameters fixed at 0.1. The testing accuracy and the learning time are reported in Fig. 11. A comparison of Fig. 11 (a), (c) and (e) shows that our SGTSVM (including the linear and nonlinear cases) is more stable than PEGASOS if $tol \leq 10^{-4}$. To select a high accuracy with an acceptable learning time from Fig. 11, *tol* is set to 10^{-6} for PEGASOS and to 10^{-4} for SGTSVM.

Then, we use these datasets to compare SVM and PEGASOS to our SGTSVM at fixed *tol*. The methods' accuracy values are shown in Table 3, where the validation accuracy is obtained by 5-fold cross validation on the training subset, and the testing accuracy is obtained for the testing subset. Parameters c in SVM and PEGASOS and c_1, c_2, c_3 and c_4 in SGTSVM were selected from $\{2^i|i = -8, -7, \ldots, 1\}$, and the Gaussian kernel parameter μ



(a) PEGASOS (b) PEGASOS

(c) SGTSVM† (d) SGTSVM†

(e) SGTSVM$^\sharp$ (f) SGTSVM$^\sharp$

Figure 11: The accuracy and learning time of PEGASOS, the linear SGTSVM (†), and the nonlinear SGTSVM ($^\sharp$) on six large scale datasets. The dashed box corresponds to the chosen parameter *tol*.

Table 3: The results for the large scale datasets.

Dataset		SVM	PEGASOS	SGTSVM†	SGTSVM$^\sharp$
Skin	validation(%)	78.87	82.46	**85.23**	84.70
245,057×3	testing(%)	84.28	85.39	**87.70**	85.34
Gashome	validation(%)	49.11	70.09	67.50	**74.49**
919,438×10	testing(%)	82.57	72.85	76.09	**89.13**
Susy	validation(%)	**78.41**	54.11	76.14	69.90
5,000,000×18	testing(%)	**78.52**	56.44	75.09	68.61
Kddcup	validation(%)	*	**96.39**	95.24	93.19
4,898,432×41	testing(%)	*	96.42	97.45	**99.20**
Gas	validation(%)	*	69.77	89.73	**92.60**
8,386,764×16	testing(%)	*	50.54	92.45	**92.86**
Hepmass	validation(%)	*	80.63	80.80	**82.18**
10,500,000×28	testing(%)	*	80.84	**81.10**	79.59

†*linear case*; $^\sharp$*nonlinear case*; **out of memory.*

Table 4: The optimal parameters of SVM, PEGASOS and SGTSVM.

Dataset		SVM	PEGASOS	SGTSVM†	SGTSVM$^\sharp$
		c	c	$c_1 = c_3, c_2 = c_4$	$c_1 = c_3, c_2 = c_4, \mu$
		2^i	2^i	$2^i, 2^j$	$2^i, 2^j, 2^k$
Skin	validation	-1	-6	0,-5	-6,-5,-3
	testing	-1	-4	1,-6	-1,0,-9
Gashome	validation	0	-6	-4,-5	-3,-5,-2
	testing	-1	-1	-8,-7	-8,-1,-2
Susy	validation	1	0	-2,-6	-3,-1,-4
	testing	0	-7	-1,-3	-3,-3,-3
Kddcup	validation	NA	-6	-8,-4	0,-3,-4
	testing	NA	-2	-8,-4	-6,-1,-8
Gas	validation	NA	-1	-4,0	-1,-1,-6
	testing	NA	1	-3,1	-4,-8,-6
Hepmass	validation	NA	0	-1,-2	-4,-1,-3
	testing	NA	0	0,-2	-4,-2,-3

†*linear case*; $^\sharp$*nonlinear case.*

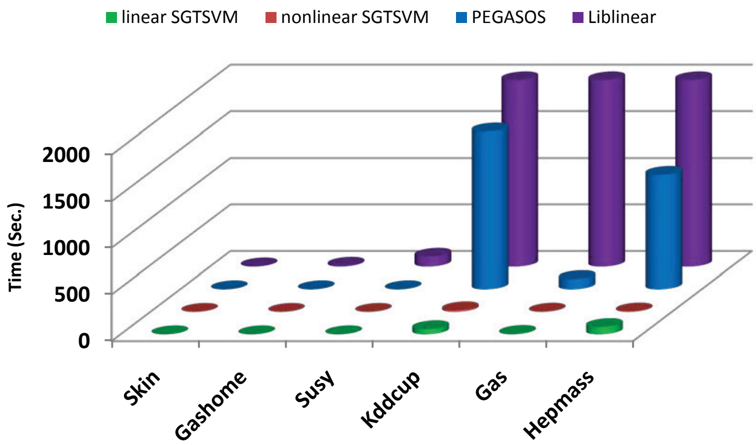


Figure 12: The learning time of SGTSVM, PEGASOS and Liblinear with the optimal parameters on large scale datasets.

in the nonlinear SGTSVM was selected from $\{2^i|i = -10, -9, \ldots, -1\}$. For simplicity, we also set $c_1 = c_3$ and $c_2 = c_4$ in SGTSVM. The optimal parameters are shown in Table 4. Table 3 clearly shows that our SGTSVM obtains the highest accuracy on 9 groups of comparisons and performs as well as SVM and PEGASOS on the other 3 groups. However, SVM performs much worse than SGTSVM on the Gashome dataset and cannot be applied to three much larger datasets. Though PEGASOS can be applied to these datasets, it performs much worse than SGTSVM on the Susy and Gas datasets. To further compare the learning time of these methods, we report the time for a single run in Fig. 12 with the optimal parameters. It is clear that SGTSVM (including the linear and nonlinear cases) is much faster than the others. Thus, our SGTSVM is comparable to SVM and PEGASOS on these large scale datasets. In addition, the software implementations of SGTSVM and PEGASOS need much less RAM than does Liblinear (the software implementation of SVM). In particular, Liblinear needs to store the entire training set in RAM, while PEGASOS and SGTSVM only store a subset related to the iteration. Due to the required memory of Liblinear increasing with the size of the dataset, the method tends to run out of memory with the increasing data size, while PEGASOS or SGTSVM does not.

5. Conclusion

An insensitive stochastic gradient twin support vector machine (SGTSVM) has been proposed. This method is less sensitive to sampling than PEGASOS while having better convergence and approximation. The experimental results have shown that our method has a better performance and a higher training speed than PEGASOS and LIBLINEAR. For practical convenience, the corresponding SGTSVM source code (including programs in Matlab and the C language) have been uploaded to `http://www.optimal-group.org/Resources/Code/SGTSVM.html`. The possibilities for future research include designing a special sampling for SGTSVM to obtain a better performance and applying SGTSVM to big data problems.

Acknowledgment

This work is supported by the National Natural Science Foundation of China (Nos. 11501310 and 61703370), Natural Science Foundation of Hainan Province (No. 118QN181), Natural Science Foundation of Inner Mongolia Autonomous Region of China (No. 2015BS0606), Inner Mongolia Autonomous Region University Scientific Research Project (No. NJZC17006) and Zhejiang Provincial Natural Science Foundation of China (Nos. LY15F030013 and LQ17F030003).

References

[1] A. Bennar and J.M. Monnez. Almost sure convergence of a stochastic approximation process in a convex set. *International Journal of Applied Mathematics*, 20(5):713–722, 2007.

[2] C.C. Chang and C.J. Lin. Libsvm : a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2(27):1–27, 2011.

[3] W.J. Chen, Y.H. Shao, C.N. Li, and N.Y. Deng. Mltsvm: A novel twin support vector machine to multi-label learning. *Pattern Recognition*, 52:61–74, 2015.

[4] C. Cortes and V.N. Vapnik. Support vector networks. *Machine Learning*, 20:273–297, 1995.

[5] R.O. Duda, P.E. Hart, and D.G. Stork. *Pattern Classification, 2nd Edition*. John Wiley and Sons, 2001.

[6] R.E. Fan, K.W. Chang, C.J. Hsieh, X.R. Wang, and C.J. Lin. LIBLINEAR: a library for large linear classification. *Journal of Machine Learning Research*, 9:1871–1874, 2008.

[7] G.H. Golub and L.C.F. Van. *Matrix Computations*. The John Hopkins University Press, 1996.

[8] P. Goyal, P. Dollár, R. Girshick, and et al. Accurate, large minibatch sgd: training imagenet in 1 hour. In *Data Mining Workshop (ICDMW), 2014 IEEE International Conference on*, volume arXiv:1706.02677. arXiv preprint, 2017.

[9] H. Ince and T.B. Trafalis. Support vector machine for regression and applications to financial forecasting. In *International Joint Conference on Neural Networks*, pages 6348–6354, Italy, 2002.

[10] Jayadeva, R. Khemchandani, and S. Chandra. Twin support vector machines for pattern classification. *IEEE Trans.PatternAnal. Machine Intell*, 29(5):905–910, 2007.

[11] S.M. Kakade and A. Tewari. On the generalization ability of online strongly convex programming algorithms. In *Advances in Neural Information Processing Systems*, pages 801–808, 2009.

[12] R. Khemchandani, Jayadeva, and S. Chandra. Optimal kernel selection in twin support vector machines. *Optimization Letters*, 3:77–88, 2009.

[13] M.A. Kumar and M. Gopal. Application of smoothing technique on twin support vector machines. *Pattern Recognition Letters*, 29(13):1842–1848, 2008.

[14] T.N. Lal, M. Schröder, T. Hinterberger, J. Weston, M. Bogdan, N. Birbaumer, and B. Schölkopf. Support vector channel selection in BCI. *Data Mining and Knowledge Discovery*, 51(6):1003–1010, 2004.

[15] Y.J. Lee and O.L. Mangasarian. Ssvm: A smooth support vector machine for classification. *Computational optimization and Applications*, 20(1):5–22, 2001.

[16] Y.J. Lee and O.L. Mangasarian. RSVM: Reduced support vector machines. In *First SIAM International Conference on Data Mining*, pages 5–7, Chicago, IL, USA, 2001.

[17] D.W. Li, Y.J. Tian, and H.G. Xu. Deep twin support vector machine. In *Data Mining Workshop (ICDMW), 2014 IEEE International Conference on*, pages 65–73. IEEE, 2014.

[18] O.L. Mangasarian and D.R. Musicant. Successive overrelaxation for support vector machines. *IEEE Transactions on Neural Networks*, 10(5):1032–1037, 1999.

[19] W.S. Noble. Support vector machine applications in computational biology. In *Kernel Methods in Computational Biology*, Cambridge, 2004.

[20] de J.F. Oliveira and M.S. Alencar. Online learning early skip decision method for the hevc inter process using the svm-based pegasos algorithm. *Electronics Letters*, 52(14):1227–1229, 2016.

[21] M. N. Omidvar, X. Li, and K. Tang. Designing benchmark problems for large-scale continuous optimization. *Information Sciences*, 316:419–436, 2015.

[22] X.J. Peng. TPMSVM: A novel twin parametric-margin support vector machine for pattern recognition. *Pattern Recognition*, 44(10-11):2678–2692, 2011.

[23] J. Platt. Fast training of support vector machines using sequential minimal optimization. In *Advances in kernel methods-support vector learning*, pages 185–208, Cambridge, MA: MIT Press, 1999.

[24] Z. Qi, Y. Tian, and Y. Shi. Twin support vector machine with universum data. *Neural Networks*, 36:112–119, 2012.

[25] Z. Qi, Y. Tian, and Y. Shi. Robust twin support vector machine for pattern classification. *Pattern Recognition*, 46(1):305–316, 2013.

[26] Z. Qi, Y. Tian, and Y. Shi. Successive overrelaxation for laplacian support vector machine. *IEEE transactions on neural networks and learning systems*, 26(4):674–683, 2015.

[27] J.L. Reyes-Ortiz, L. Oneto, and D. Anguita. Big data analytics in the cloud: Spark on hadoop vs mpi/openmp on beowulf. volume 53, pages 121–130, 2015.

[28] W. Rudin. *Principles of mathematical analysis*, volume 3. McGraw-Hill New York, 1964.

[29] S.S. Shai, Y. Singer, N. Srebro, and A. Cotter. Pegasos: Primal estimated sub-gradient solver for svm. *Mathematical programming*, 127(1):3–30, 2011.

[30] Y.H. Shao, W.J. Chen, and N.Y. Deng. Nonparallel hyperplane support vector machine for binary classification problems. *Information Sciences*, 263:22–35, 2014.

[31] Y.H. Shao, W.L. Chen, J.J. Zhang, Z. Wang, and N.Y. Deng. An efficient weighted lagrangian twin support vector machine for imbalanced data classification. *Pattern Recognition*, 47(9):3158–3167, 2014.

[32] Y.H. Shao and N.Y. Deng. A coordinate descent margin based-twin support vector machine for classification. *Neural Networks*, 25:114–121, 2012.

[33] Y.H. Shao, C.H. Zhang, X.B. Wang, and N.Y. Deng. Improvements on twin support vector machines. *IEEE Transactions on Neural Networks*, 22(6):962 – 968, 2011.

[34] K. Sopyla and P. Drozda. Stochastic gradient descent with barzilaicborwein update step for svm. *Information Sciences*, 316:218–233, 2015.

[35] Y.J. Tian and Y. Ping. Large-scale linear nonparallel support vector machine solver. *Neural Networks*, 50:166–174, 2014.

[36] R. Ñanculef, E. Frandi, C. Sartori, and et al. A novel frankcwolfe algorithm. analysis and applications to large-scale svm training. *Information Sciences*, 285:66–99, 2014.

[37] D. Valiente, A. Gil, L. Fernndez, and et al. A modified stochastic gradient descent algorithm for view-based slam using omnidirectional images. *Information Sciences*, 279:326–337, 2014.

[38] Z. Wang, Y.H. Shao, L. Bai, and N.Y. Deng. Twin support vector machine for clustering. *IEEE Transactions on Neural Networks and Learning Systems*, 26(10):2583–2588, 2015.

[39] Z. Wang, Y.H. Shao, and T.R. Wu. A ga-based model selection for smooth twin parametric-margin support vector machine. *Pattern Recognition*, 46(8):2267–2277, 2013.

[40] Z. Wang, Y.H. Shao, and T.R. Wu. Proximal parametric-margin support vector classifier and its applications. *Neural Computing and Applications*, 24(3-4):755–764, 2014.

[41] W. Xu. Towards optimal one pass large scale learning with averaged stochastic gradient descent. *arXiv preprint arXiv:1107.2490*, 2011.

[42] C.H. Zhang, Y.J. Tian, and N.Y. Deng. The new interpretation of support vector machines on statistical learning theory. *Science China*, 53(1):151–164, 2010.

[43] T. Zhang. Solving large scale linear prediction problems using stochastic gradient descent algorithms. In *Proceedings of the twenty-first international conference on Machine learning*, page 116. ACM, 2004.