Large-Scale Least Squares Twin SVMs

M. TANVEER and S. SHARMA, Department of Mathematics, Indian Institute of Technology Indore, India
K. MUHAMMAD, Department of Software, Sejong University, Republic of Korea

In the last decade, twin support vector machine (TWSVM) classifiers have achieved considerable emphasis on pattern classification tasks. However, the TWSVM formulation still suffers from the following two shortcomings: (1) TWSVM deals with the inverse matrix calculation in the Wolfe-dual problems, which is intractable for large-scale datasets with numerous features and samples, and (2) TWSVM minimizes the empirical risk instead of the structural risk in its formulation. With the advent of huge amounts of data today, these disadvantages render TWSVM an ineffective choice for pattern classification tasks. In this article, we propose an efficient large-scale least squares twin support vector machine (LS-LSTSVM) for pattern classification that rectifies all the aforementioned shortcomings. The proposed LS-LSTSVM introduces different Lagrangian functions to eliminate the need for calculating inverse matrices. The proposed LS-LSTSVM also does not employ kernel-generated surfaces for the non-linear case, and thus uses the kernel trick directly. This ensures that the proposed LS-LSTSVM model is superior to the original TWSVM and LSTSVM. Lastly, the structural risk is minimized in LS-LSTSVM. This exhibits the essence of statistical learning theory, and consequently, classification accuracy on datasets can be improved due to this change. The proposed LS-LSTSVM is solved using the sequential minimal optimization (SMO) technique, making it more suitable for large-scale problems. We further proved the convergence of the proposed LS-LSTSVM. Exhaustive experiments on several real-world benchmarks and NDC-based large-scale datasets demonstrate that the proposed LS-LSTSVM is feasible for large datasets and, in most cases, performed better than existing algorithms.

CCS Concepts: • **Computer systems organization** → **Embedded systems**; *Redundancy*; Robotics; • **Networks** → Network reliability;

Additional Key Words and Phrases: Machine learning, support vector machines (SVMs), large scale SVMs, least squares twin SVM

ACM Reference format:
M. Tanveer, S. Sharma, and K. Muhammad. 2021. Large-Scale Least Squares Twin SVMs. *ACM Trans. Internet Technol.* 21, 2, Article 29 (May 2021), 19 pages.
https://doi.org/10.1145/3398379

This work is supported by the Science and Engineering Research Board (SERB), India (Grant Nos. SB/S2/RJN-001/2016 and ECR/2017/000053) under the Ramanujan Fellowship and Early Career Research Award Schemes. It is also supported by the Council of Scientific & Industrial Research (CSIR), New Delhi, India (Grant No. 22(0751)/17/EMR-II) under the Extra Mural Research (EMR) Scheme.
Authors' addresses: M. Tanveer (corresponding author) and S. Sharma, Department of Mathematics, Indian Institute of Technology Indore, Simrol, Indore, 453552, India; emails: {mtanveer, msc1703141010}@iiti.ac.in. K. Muhammad (corresponding author), Department of Software, Sejong University, Seoul, 143747, Republic of Korea; email: khan.muhammad@ieee.org.
Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
© 2021 Association for Computing Machinery.
1533-5399/2021/05-ART29 $15.00
https://doi.org/10.1145/3398379

1 INTRODUCTION

The foundation of support vector machines (SVMs) was laid by Vapnik and co-workers [6, 52] in the early 1990s and has attained great popularity due to many of its interesting features. After few years of its foundation, SVM was successfully applied to several important domains such as face detection [26], web mining [2], remote sensing [27], electroencephalogram (EEG) signal classification [29], cancer diagnosis [5], Alzheimer's disease diagnosis [32, 46], and feature extraction [20]. SVM searches for separating hyperplanes by formulating a convex quadratic programming problem (QPP), which seeks to maximize the margin between two classes. It minimizes an upper bound of generalization error due to incorporating the structural risk minimization (SRM) principle [33] in its optimization problem.

Solving an objective function in SVM necessitates minimization of one large QPP, subject to linear inequality constraints, which results in escalation of computational complexity and makes SVM inappropriate for large-scale data analysis. Afterwards, several SVM-based algorithms were developed for large-scale datasets such as [4, 14–16, 34, 39, 55]. In order to reduce the complexity, Mangasarian and Wild [22] developed an efficient algorithm named generalized eigen-value proximal support vector machine (GEPSVM) for binary classification, which is based on the idea of generating non-parallel hyperplanes. Later on, Jayadeva et al. [13] introduced an efficient twin support vector machine (TWSVM) for binary classification problems. TWSVM generates two non-parallel hyperplanes such that each hyperplane is proximal to the samples of one of two classes and is at least one distance from the samples of the other class. TWSVM solves two small QPPs instead of solving one large QPP, which makes TWSVM approximately four times faster than the standard SVM. While training TWSVM, calculation of an inverse matrix is an unavoidable task and demands matrix to be non-singular. So, to avoid ill-conditioning, an extra small positive quantity is added in the matrix. TWSVM implements the empirical risk minimization (ERM) principle. Subsequently, Shao et al. [37] proposed twin-bounded SVM (TBSVM), which is an improved version of TWSVM. TBSVM also searches for two non-parallel hyperplanes and solves two small QPPs. Primal formulation of TBSVM involves an additional regularization term, which reflects the structural risk minimization principle. The main advantage of TBSVM over TWSVM is its dual formulation and that a solution can be derived without any extra assumptions. In recent years, many variants of TWSVM have been developed based on the idea of generating a pair of non-parallel hyperplanes for small as well as large datasets such as least squares TWSVM [18, 19, 48], robust and sparse TWSVM [36, 40, 42], improved TWSVM [41, 49, 51], stochastic gradient TWSVM [53], angle-based TWSVM [17, 30], pinball TWSVMs [38, 45, 47, 50, 54], large-scale non-parallel SVM [21, 35, 53], and universum-based TWSVM [28].

Further, robust energy-based least squares twin SVM (RELS-TWSVM) is proposed by Tanveer et al. [44], inspired by energy-based least squares TWSVM (ELS-TSVM) [25]. RELS-TWSVM maximizes the margin with a positive definite matrix formulation to overcome the positive semi-definite matrices formulation in case of TWSVM and ELS-TSVM. Moreover, RELS-TWSVM does not need any special optimizer but uses energy parameters to reduce the effect of outliers. Recently, Tanveer et al. [43] provided an exhaustive analysis of 8 variants of twin SVM-based classifiers along with 179 classifiers of 17 familiesm and we see that RELS-TWSVM [44] emerged as the best variant of TWSVMs. To handle the imbalanced datasets, recently, a novel reduced universum twin SVM for class imbalance learning (RUTSVM-CIL) [31] was developed. The experimental results show that RUTSVM-CIL [31] is an effective model for imbalanced datasets.

Besides their advantages, there are several drawbacks of TWSVM, TBSVM, and RELS-TWSVM, which make them unable to cope with real-life data, numerous samples, and features:

- Computation of inverse matrices may lead to suboptimal solutions in some cases and make them ill-suited for large scale data analysis.
- These models need to solve linear and nonlinear cases separately since the nonlinear case with a linear kernel in the original TWSVM algorithm is not equivalent to the linear case.

Motivated by [51], we propose a novel large-scale least squares twin SVM (LS-LSTSVM), which can efficiently handle the large-scale data with multitudinous features and samples. LS-LSTSVM generates a pair of non-parallel hyperplanes, inheriting the essence of TWSVM. LS-LSTSVM solves two objective functions subject to linear equality constraints. The proposed LS-LSTSVM has some agreeable advantages:

- Unlike TWSVM, TBSVM, RELS-TWSVM, and RUTSVM-CIL, the proposed LS-LSTSVM introduces different Lagrangian functions to eliminate the need for calculating the inverse matrices.
- The proposed LS-LSTSVM is solved using a fast sequential minimal optimization (SMO) solver [14, 15, 34], making it more suitable for large-scale problems.
- The proposed LS-LSTSVM does not employ kernel-generated surfaces for the non-linear case, and thus uses the kernel trick directly.

Furthermore, numerous experiments on several benchmarks and NDC-based large-scale datasets demonstrate that the proposed LS-LSTSVM is an efficient algorithm for the large-scale datasets.

The rest of the article is organized as follows: Section 2 presents the linear and nonlinear TBSVM formulation. In Section 3, an efficient LS-LSTSVM is proposed for both linear and nonlinear cases. Section 4 gives the brief formulation and proof of convergence for the SMO solver. Section 5 shows numerical experiments and compares results of TWSVM, TBSVM, RELS-TWSVM, and the proposed LS-LSTSVM. In Section 6, we conduct statistical analysis of the proposed LS-LSTSVM, and finally we conclude our work with future directions in Section 7.

2 BACKGROUND

In this section, we briefly describe the formulation of TBSVM. Interested readers are referred to [37] for more details.

2.1 Twin-Bounded Support Vector Machine (TBSVM)

TBSVM, proposed by Shao et al. [37], is a regularized version of the original TWSVM [13] formulation. TBSVM is a binary classifier that seeks a pair of non-parallel hyperplanes for pattern classification. The pair of hyperplanes can be expressed as

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

and are obtained such that each hyperplane is proximal to the samples of one class and farthest from the samples of the other class. That is, the patterns of one class create constraints to the patterns of the other class. An unknown sample point is allocated to class +1 or −1 depending upon the proximity of the hyperplane to an unknown sample point.

2.1.1 Linear TBSVM. Consider the binary class problem with the training set

$$T = \{(x_1, +1), (x_2, +1), \ldots, (x_p, +1), (x_{p+1}, -1), \ldots, (x_{p+q}, -1)\}, \tag{2}$$

where $x_i \in \mathbb{R}^n,\ i = 1, \ldots, p+q,\ p+q = l$.

The linear TBSVM formulation can be expressed as follows:

$$\begin{aligned} \min_{w_1, b_1, \zeta_2} \quad & \frac{c_3}{2}(||w_1||^2 + b_1^2) + \frac{1}{2}||Aw_1 + e_1 b_1||^2 + c_1 e_2^T \zeta_2 \\ \text{s.t.} \quad & -(Bw_1 + e_2 b_1) + \zeta_2 \geq e_2,\ \zeta_2 \geq 0 \end{aligned} \tag{3}$$

and

$$\begin{aligned} \min_{w_2, b_2, \zeta_1} \quad & \frac{c_4}{2}(||w_2||^2 + b_2^2) + \frac{1}{2}||Bw_2 + e_2 b_2||^2 + c_2 e_1^T \zeta_1 \\ \text{s.t.} \quad & (Aw_2 + e_1 b_2) + \zeta_1 \geq e_1,\ \zeta_1 \geq 0, \end{aligned} \tag{4}$$

where $c_i, i = 1, 2, 3, 4$ are penalty parameters, ζ_1 and ζ_2 are the slack variables, e_1, e_2 are vectors of ones of appropriate dimensions, and A and B are matrices of the positive and negative class, respectively.

By introducing the Lagrangian function with Lagrange multipliers $\alpha \geq 0$ and $\gamma \geq 0$ and using the Karush-Kuhn-Tucker (KKT) [23] optimality conditions, the duals of Equations (3) and (4) are obtained as follows:

$$\begin{aligned} \max_{\alpha} \quad & -\frac{1}{2}\alpha^T G(H^T H + c_3 I)^{-1} G^T \alpha + e_2^T \alpha \\ \text{s.t.} \quad & 0 \leq \alpha \leq c_1 \end{aligned} \tag{5}$$

and

$$\begin{aligned} \max_{\gamma} \quad & \frac{1}{2} - \gamma^T H(G^T G + c_4 I)^{-1} H^T \gamma + e_1^T \gamma \\ \text{s.t.} \quad & 0 \leq \gamma \leq c_2, \end{aligned} \tag{6}$$

where $H = [A\ e_1]$ and $G = [B\ e_2]$ are augmented matrices and I is an identity matrix of an appropriate dimension.

Finally, solutions are obtained as follows:

$$v_1 = -(H^T H + c_3 I)^{-1} G^T \alpha, \quad v_1 = [w_1, b_1]^T \tag{7}$$

and

$$v_2 = (G^T G + c_4 I)^{-1} H^T \gamma, \quad v_2 = [w_2, b_2]^T. \tag{8}$$

Here, c_3 and c_4 are weighting factors that determine the trade-off between the regularization term and the empirical risk. Therefore, optimal values of weighting factors reflect the structural risk minimization principle.

Every new data point, $x \in \mathbb{R}^n$, is categorized according to the decision function given by

$$class = argmin_{l=1,2}(|x^T w_l + b_l|), \tag{9}$$

where $|.|$ denotes a perpendicular distance of a point to the plane.

2.1.2 Nonlinear TBSVM. The nonlinear TBSVM considers the following kernel-generated surfaces:

$$K(x^T, C^T)w_1 + b_1 = 0 \text{ and } K(x^T, C^T)w_2 + b_2 = 0, \tag{10}$$

where $C = [A; B]$, $w_1,\ w_2 \in \mathbb{R}^n$, and K is a Gaussian kernel function. The nonlinear TBSVM formulation can be expressed as follows:

$$\begin{aligned} \min_{w_1, b_1, \zeta_2} \quad & \frac{1}{2}c_3(||w_1||^2 + b_1^2) + \frac{1}{2}||K(A, C^T)w_1 + e_1 b_1||^2 + c_1 e_2^T \zeta_2 \\ \text{s.t.} \quad & -(K(B, C^T)w_1 + e_2 b_1) + \zeta_2 \geq e_2,\ \zeta_2 \geq 0 \end{aligned} \tag{11}$$

and

$$\begin{aligned} \min_{w_2, b_2, \zeta_1} \quad & \frac{1}{2} c_4 (||w_2||^2 + b_2^2) + \frac{1}{2} ||K(B, C^T) w_2 + e_2 b_2||^2 + c_2 e_1^T \zeta_1 \\ \text{s.t.} \quad & (K(A, C^T) w_2 + e_1 b_2) + \zeta_1 \geq e_1, \ \zeta_1 \geq 0. \end{aligned} \tag{12}$$

The corresponding duals are

$$\begin{aligned} \max_{\alpha} \quad & -\frac{1}{2} \alpha^T R (S^T S + c_3 I)^{-1} R^T \alpha + e_2^T \alpha \\ \text{s.t.} \quad & 0 \leq \alpha \leq c_1 \end{aligned} \tag{13}$$

and

$$\begin{aligned} \max_{\gamma} \quad & -\frac{1}{2} \gamma^T S (R^T R + c_4 I)^{-1} S^T \gamma + e_1^T \gamma \\ \text{s.t.} \quad & 0 \leq \gamma \leq c_2, \end{aligned} \tag{14}$$

where $R = [K(B, C^T) \ e_2]$ and $S = [K(A, C^T) \ e_2]$.

One can get the augmented vectors as follows:

$$Z_1 = -(S^T S + c_3 I)^{-1} R^T \alpha, \quad Z_1 = [w_1, b_1]^T \tag{15}$$

and

$$Z_2 = (R^T R + c_4 I)^{-1} S^T \gamma, \quad Z_2 = [w_2, b_2]^T. \tag{16}$$

Categorization of new data point $x \in \mathbb{R}^n$ is done by using the decision function as follows:

$$class = argmin_{l=1,2} (|K(x^T, C^T) w_l + b_l|). \tag{17}$$

3 PROPOSED LARGE-SCALE LEAST SQUARES TWIN SVM (LS-LSTSVM)

Most of the TWSVM-based algorithms including TBSVM, RELS-TWSVM, and RUTSVM-CIL suffer mainly from two shortcomings: computing inverse matrices, which may lead to suboptimal solutions in some cases and make them ill-suited for large-scale analysis, and employing kernel-generated surfaces for the nonlinear case.

The proposed LS-LSTSVM is a least squares version of TBSVM that works for the large-scale datasets. LS-LSTSVM searches for a pair of non-parallel hyperplanes for classifying the samples. By introducing different Lagrangian functions, we obtained a pair of unconstrained dual optimization problems.

3.1 Linear LS-LSTSVM

The formulation of linear LS-LSTSVM can be expressed as follows:

$$\begin{aligned} \min_{w_1, b_1, \zeta_1} \quad & \frac{c_3}{2} (||w_1||^2 + b_1^2) + \frac{1}{2} \eta_1^T \eta_1 + \frac{c_1}{2} ||\zeta_1||^2 \\ s.t. \quad & A w_1 + e_2 b_1 = \eta_1 \\ & -(B w_1 + e_1 b_1) + \zeta_1 = e_1 \end{aligned} \tag{18}$$

and

$$\begin{aligned} \min_{w_2, b_2, \zeta_2} \quad & \frac{c_4}{2} (||w_2||^2 + b_2^2) + \frac{1}{2} \eta_2^T \eta_2 + \frac{c_2}{2} ||\zeta_2||^2 \\ s.t. \quad & B w_2 + e_1 b_2 = \eta_2 \\ & A w_2 + e_2 b_2) + \zeta_2 = e_2, \end{aligned} \tag{19}$$

where $c_i, i = 1, 2, 3, 4$ are penalty parameters, ζ_1 and ζ_2 are slack variables, and e_1 and e_2 are vectors of ones of appropriate dimensions.

Notice that instead of 1-norm, we have applied 2-norm on slack variables with weights $\frac{c_1}{2}$ and $\frac{c_2}{2}$, respectively, which makes constraints $\zeta_1 \geq 0$ and $\zeta_2 \geq 0$ redundant. We introduce the Lagrangian in the primal problem of Equation (18) as follows:

$$L(w_1, b_1, \eta_1, \zeta_1, \alpha, \beta) = \frac{1}{2}\eta_1^T\eta_1 + \frac{c_1}{2}||\zeta_1||^2 + \frac{c_3}{2}(||w_1||^2 + b_1^2) + \alpha^T(\eta_1 - Aw_1 - e_2b_1) + \beta^T(\zeta_1 - Bw_1 - e_1b_1 - e_1),$$

where $\alpha = (\alpha_1, \ldots, \alpha_p)^T$, $\beta = (\beta_1, \ldots, \beta_q)^T$ are vectors of Lagrangian multipliers. The KKT necessary and sufficient conditions are given by

$$c_3w_1 - A^T\alpha - B^T\beta = 0, \tag{20}$$

$$c_3b_1 - e_2^T\alpha - e_1^T\beta = 0, \tag{21}$$

$$\eta_1 + \alpha = 0, \tag{22}$$

$$c_1\zeta_1 + \beta = 0, \tag{23}$$

$$Aw_1 + e_2b_1 = \eta_1, \tag{24}$$

$$-(Bw_1 + e_1b_1) + \zeta_1 = e_1. \tag{25}$$

From Equations (20) and (21), we have

$$\begin{pmatrix} w_1 \\ b_1 \end{pmatrix} = \frac{1}{c_3}\begin{pmatrix} A^T & B^T \\ e_2^T & e_1^T \end{pmatrix}\begin{pmatrix} \alpha \\ \beta \end{pmatrix}. \tag{26}$$

Using Equations (26), (22), and (23), we get an unconstrained dual formulation of Equation (18) as follows:

$$\max_{\alpha,\beta} -\frac{1}{2}(\alpha^T \;\; \beta^T)\widehat{Q}(\alpha^T \;\; \beta^T)^T - c_3\beta^Te_1, \tag{27}$$

where

$$\widehat{Q} = \begin{pmatrix} AA^T + c_3I_p & AB^T \\ BA^T & BB^T + \frac{c_3}{c_1}I_q \end{pmatrix} + E,$$

I_p and I_q are $p \times p$ and $q \times q$ identity matrices, respectively, and E is an $l \times l$ matrix of ones.

Similarly, the dual formulation for Equation (19) is given by

$$\max_{\lambda,\gamma} -\frac{1}{2}(\lambda^T \;\; \gamma^T)\tilde{Q}(\lambda^T \;\; \gamma^T)^T - c_4\gamma^Te_2, \tag{28}$$

where

$$\tilde{Q} = \begin{pmatrix} BB^T + c_4I_q & BA^T \\ AB^T & AA^T + \frac{c_4}{c_2}I_q \end{pmatrix} + E,$$

and w_2 and b_2 can be given by

$$\begin{pmatrix} w_2 \\ b_2 \end{pmatrix} = -\frac{1}{c_4}\begin{pmatrix} B^T & A^T \\ e_1^T & e_2^T \end{pmatrix}\begin{pmatrix} \lambda \\ \gamma \end{pmatrix}. \tag{29}$$

The class of an unknown input can be predicted with the help of the following decision function:

$$class = argmin_{l=1,2}(|x^Tw_l + b_l|), \tag{30}$$

where $|.|$ is the perpendicular distance of x from the plane.

Due to the least squares formulation and an extra variable in the Lagrangian function, the dual formulation of LS-LSTSVM has nothing to do with the inverse of the matrix. Moreover, the dual of LS-LSTSVM is an unconstrained optimization problem that consumes less effort to get the solution.

3.2 Nonlinear LS-LSTSVM

The nonlinear case of LS-LSTSVM is different from the nonlinear case of TBSVM; i.e., instead of considering the kernel-generated surfaces, we can directly introduce the kernel function in Equations (27) and (28) as we do in the standard SVM.

We will take the transformation of the feature vector as follows:

$$\mathbf{x} = \phi(x), \tag{31}$$

where $\mathbf{x} \in \mathcal{H}$, $\mathcal{H}$ is the Hilbert space. The training set will become

$$\mathbf{T} = \{(\mathbf{x}_1, +1), (\mathbf{x}_2, +1), \ldots, (\mathbf{x}_p, +1), (\mathbf{x}_{p+1}, -1), \ldots, (\mathbf{x}_{p+q}, -1)\}. \tag{32}$$

Now, the primal problems for the nonlinear case are as follows:

$$\begin{aligned} \min_{w_1, b_1, \zeta_1} \quad & \frac{c_3}{2}(||w_1||^2 + b_1^2) + \frac{1}{2}\eta_1^T\eta_1 + \frac{c_1}{2}||\zeta_1||^2 \\ s.t. \quad & \phi(A)w_1 + e_2 b_1 = \eta_1, \\ & -(\phi(B)w_1 + e_1 b_1) + \zeta_1 = e_1 \end{aligned} \tag{33}$$

and

$$\begin{aligned} \min_{w_2, b_2, \zeta_2} \quad & \frac{c_4}{2}(||w_2||^2 + b_2^2) + \frac{1}{2}\eta_2^T\eta_2 + \frac{c_2}{2}||\zeta_2||^2 \\ s.t. \quad & \phi(B)w_2 + e_1 b_2 = \eta_2, \\ & (\phi(A)w_2 + e_2 b_2) + \zeta_2 = e_2. \end{aligned} \tag{34}$$

Similar to the linear case, we can obtained their duals:

$$\max_{\alpha, \beta} -\frac{1}{2}(\alpha^T \ \beta^T)\widehat{Q}(\alpha^T \ \beta^T)^T - c_3\beta^T e_1, \tag{35}$$

where

$$\widehat{Q} = \begin{pmatrix} K(A, A^T) + c_3 I_p & K(A, B^T) \\ K(B, A^T) & K(B, B^T) + \frac{c_3}{c_1} I_q \end{pmatrix} + E,$$

and

$$\max_{\lambda, \gamma} -\frac{1}{2}(\lambda^T \ \gamma^T)\tilde{Q}(\lambda^T \ \gamma^T)^T - c_4\gamma^T e_2, \tag{36}$$

where

$$\tilde{Q} = \begin{pmatrix} K(B, B^T) + c_4 I_q & K(B, A^T) \\ K(A, B^T) & K(A, A^T) + \frac{c_4}{c_2} I_q \end{pmatrix} + E.$$

After getting an optimal solution, the pair of non-parallel planes in the Hilbert space $\mathcal{H}$ are defined as

$$f_1(x) := \frac{1}{c_3}(K(x, A^T)\alpha + K(x, B^T)\beta) + b_1, \tag{37}$$

where

$$b_1 = \frac{1}{c_3}(e_2^T\alpha + e_1^T\beta),$$

and

$$f_2(x) := \frac{1}{c_4}(K(x, B^T)\lambda + K(x, A^T)\gamma) + b_2, \tag{38}$$

where

$$b_2 = -\frac{1}{c_4}(e_1^T\lambda + e_2^T\gamma).$$

The decision function is given by

$$class = argmin_{k=1,2}|f_k(x)|, \tag{39}$$

where |.| is the perpendicular distance.

Here, we do not need to calculate the inverse of the matrices. Moreover, Equations (35) and (36) can easily be degenerated to linear LS-LSTSVM by taking the linear kernel. Similar to linear LS-LSTSVM, Equations (35) and (36) are unconstrained optimization problems.

4 FAST SOLVER FOR LS-LSTSVM

To deal with the large-scale problems, we use the iterative method to solve the dual formulation. In the literature, several approaches are used such as CVX solver [10, 11]; however, we employ the SMO solver for solving the dual formulations of Equations (35) and (36) [14, 34].

4.1 SMO for LS-LSTSVM

For applying the SMO solver in the proposed LS-LSTSVM, we modify Equations (35) and (36) as follows:

$$\max_{\alpha,\beta} -\frac{1}{2}(\alpha^T \ \beta^T)\widehat{Q}(\alpha^T \ \beta^T)^T - (0_p^T \ e_1^T)(\alpha^T \ \beta^T)^T, \tag{40}$$

where

$$\widehat{Q} = \begin{pmatrix} K(A,A^T) + c_3 I_p & K(A,B^T) \\ K(B,A^T) & K(B,B^T) + \frac{c_3}{c_1} I_q \end{pmatrix} + E,$$

and

$$\max_{\lambda,\gamma} -\frac{1}{2}(\lambda^T \ \gamma^T)\tilde{Q}(\lambda^T \ \gamma^T)^T - (0_q^T \ e_2^T)(\lambda^T \ \gamma^T)^T, \tag{41}$$

where

$$\tilde{Q} = \begin{pmatrix} K(B,B^T) + c_4 I_q & K(B,A^T) \\ K(A,B^T) & K(A,A^T) + \frac{c_4}{c_2} I_q \end{pmatrix} + E,$$

0_p and 0_q are the vectors of zeros of dimensions $p \times 1$ and $q \times 1$, respectively.

The dual formulations of Equations (40) and (41) can be re-written in the following way:

$$f(x) = \max_x -\frac{1}{2}x^T Q x - v^T x, \tag{42}$$

where x is the $l \times 1$ variable vector, Q is the $l \times l$ positive definite matrix, and $v \geq 0$ is a vector of appropriate dimension. We see that Equation (42) is an unconstrained optimization problem that can be efficiently solved using SMO.

In order to obtain an optimal solution, iterative technique is used for an unconstrained optimization problem. Roughly speaking, we need to obtain the derivative of Equation (42) and then check an optimal condition for every updated vector x. The vector x for which the optimal condition becomes less than the given tolerance is chosen as an optimal solution. We have the following algorithm:

Proposition: *Sequence of* $\{x^n\}$ *converges to optimal global solution of Equation (42)*

Proof: From the definition of $||F(x^n)||^2$, where $F(x^n) = (F_1(x^n), \ldots, F_l(x^n))$, we get

$$||F(x^{n+1})||^2 - ||F(x^n)||^2 = -t^2 \sum_{j=1}^{l} Q(j,j), \tag{43}$$

where$t$ is an optimized step size. The above equality is always negative since the positive definite kernel function gives $Q(j,j) \geq 0$, $\forall j$. Furthermore, $||x^{n+1} - x^n||^2 = t^2$, and substituting it in

ALGORITHM 1: Sequential minimal optimization (SMO)

1: Set $n = 0$ and initialize x as $x^n = x^0$, where $x^n = (x_1^n, \ldots, x_l^n)$.
2: Compute $F_i(x^0) = \frac{\partial f}{\partial x_i^0} \quad \forall i$.
  while x is not optimal **do**
  compute $i = argmax_j \frac{F_j^2}{2Q(j,j)}$;
  Update x_i^n as x_i^{n+1} and $x^n \rightarrow x^{n+1}$;
  Compute $F_i(x^{n+1}) = \frac{\partial f}{\partial x_i^{n+1}}, \quad \forall i$;
  end
3: Optimal x is obtained.

Equation (43), we get

$$||F(x^{n+1})||^2 - ||F(x^n)||^2 = -||x^{n+1} - x^n||^2 \sum_{j=1}^{l} Q(j,j). \tag{44}$$

Equation (44) submitted that $\{||F(x^n)||^2\}$ is a decreasing sequence in a Euclidean space with lower-bound zero since $||F(x^n)||^2 \geq 0$. Thus, $\{||F(x^n)||^2\}$ is monotonically decreasing and bounded below sequence; hence, it converges to 0. From Equation (44), we can conclude that $\{x^n\}$ is a Cauchy sequence.

Since F_j^2 is a positive definite quadratic form, $||F(x^n)||^2$ is also a positive definite quadratic form. Hence, the set $S = \{x \mid ||F(x)||^2 \leq ||F(x^0)||^2\}$ is a compact set. Also, $\{x^n\}$ is an infinite bounded subset of set S and hence it has a convergent subsequence in S. Let $\{x^{n_k}\}, k \in \mathbb{N}$ be the convergent subsequence of $\{x^n\}$ and $\tilde{x}$ be the limit point of the subsequence. Now, $\lim_{k\rightarrow\infty} x^{n_k} = \tilde{x}$ implies $\lim_{k\rightarrow\infty} F_j^2(x^{n_k}) = F_j^2(\tilde{x})$ for all j. Since $\{||F(x^n)||^2\}$ is a decreasing sequence, we get $\lim_{k\rightarrow\infty} ||F(x^{n_k})||^2 = 0$. We know that $0 \leq F_j(x^{n_k}) \leq F(x^{n_k})$ for all j; therefore, $\lim_{k\rightarrow\infty} F_j^2(x^{n_k}) = F_j^2(\tilde{x}) = 0$. So, $F_j(\tilde{x}) = 0$, $\forall j$. KKT conditions deduced that $\tilde{x}$ is an optimal solution of Equation (42). As $f(x)$ is a strictly convex function, Equation (42) has a unique global solution. Denote the global optimal solution as x^*.

Now, we need to prove that the subsequence $\{x^{n_k}\}, k \in \mathbb{N}$ converges to x^*. On the contrary, assume that it does not converge to x^*. Then there exists ϵ neighborhood of x^*, satisfying $||x^{n_k} - x^*|| \gneq \epsilon$ for all but finitely many terms. This means $||\tilde{x} - x^*|| \gneq \epsilon$, which contradicts the fact that x^* is a unique global optimal solution of Equation (42) since $\tilde{x}$ is proved to be the global optimal solution of Equation (42). Hence, the proof is completed.

5 EXPERIMENTAL RESULTS

In this section, we present the comparison of the proposed LS-LSTSVM with the TBSVM [37], TWSVM [13], and RELS-TWSVM [44] algorithms. Numerical experiments for both linear and nonlinear cases are performed using MATLAB R2017a on a high-performance computer with 2x Intel Xeon Processor, 128 GB RAM, with 4 TB of secondary storage. For the nonlinear case, we used the Gaussian kernel function $K(x, x') = e^{-\mu||x-x'||^2}$, where μ is a kernel parameter. To obtain the optimal parameters, we used standard 10-fold cross-validation [8] for all the algorithms. The accuracy is calculated with the formula $Accuracy = (TP + TN)/(TN + TP + FP + FN)$, where TP, TN, FP, and FN are numbers of true positives, true negatives, false positives, and false negatives, respectively.

5.1 Parameter Selection

The tolerance parameter for the SMO algorithm, applied in LS-LSTSVM, is fixed to be $\epsilon = 0.01$. We fixed the iteration in the SMO algorithm to get the fastest convergence. TBSVM and TWSVM are implemented using the SOR algorithm. Here, ϵ for TBSVM and TWSVM is selected according to [13, 37]. For the best classification accuracy, we tuned the penalty parameters $c_i, i = 1, 2, 3, 4$ and kernel parameter σ by the well-known grid search method [12]. We consider penalty parameters in all 4 algorithms as $c_1 = c_2$ and $c_3 = c_4$ for both linear and nonlinear cases. Further, the optimal values of the parameters $c_i, i = 1, 2, 3, 4$ and Gaussian kernel σ are selected from the range sets $\{2^i | i = -9, -7, -5, \ldots, 5, 7, 9\}$ and $\{2^i | i = -5, -4, \ldots, 4, 5\}$, respectively. Once the optimal parameters are obtained, the tuning set is returned to learn the classifier.

Further, we have compared the results on various UCI benchmark and NDC-based large-scale datasets. In order to avoid complexity for large-scale classification, we have compared nonlinear classifiers by fixing the penalty parameters of all the algorithms to be one ($c_1 = c_2 = c_3 = c_4 = 1$).

5.2 Results and Discussion

In this subsection, we present the comparison of the proposed LS-LSTSVM with TBSVM [37], TWSVM [13], and RELS-TWSVM [44] algorithms on small- as well as large-scale datasets. The numerical experiments are conducted for both linear and nonlinear cases. Moreover, an analysis of computation time is shown in Tables 3 and 4 to show the applicability of the proposed LS-LSTSVM on large-scale datasets.

5.2.1 Comparison on Small-Scale Datasets. Here, we compare the performance of the linear and nonlinear LS-LSTSVM with the baseline algorithms on several small-scale UCI benchmark datasets [1, 9]. The experimental results are summarized in Tables 1 and 2. The best accuracy is shown by bold figures.

One can observe from Tables 1 and 2 that the accuracy of the proposed LS-LSTSVM is better than TBSVM, TWSVM, and RELS-TWSVM in both linear and nonlinear cases on most of the datasets. Specifically, linear LS-LSTSVM outperforms on 15 of 23 datasets, whereas linear RELS-TWSVM outperforms on 3 of 23 datasets. This indicates that solving an unconstrained dual using the SMO technique is more reasonable than solving the system of linear equations. Tables 1 and 2 also depict the training time of TBSVM, TWSVM, RELS-TWSVM, and the proposed LS-LSTSVM. It is evident that the proposed LS-LSTSVM performs better on most of the datasets. From Table 1, we can see that the proposed LS-LSTSVM requires the least time for classifying WPBC, Oocytes-trisopterus-nucleus-2f, Breast-cancer-wisc, and many other datasets.

Furthermore, Figure 1 shows the sensitivity of the penalty parameters on accuracies for six different UCI benchmark datasets in the proposed linear LS-LSTSVM. The figure shows the 3D representation of accuracy in relation to $c_1 = c_2$ and $c_3 = c_4$. In Figure 1(a), we can see an escalation in accuracy up to approximately 100% with a variation in c_1 and c_3. Similarly, we can observe the effect of parameters in subfigures (b), (c), (d), (e), and (f). Therefore, selection of parameter range is an important issue when analyzing the performance of the proposed LS-LSTSVM.

5.2.2 Comparison on Large-Scale UCI Datasets. In this subsection, we see the performance of the proposed LS-LSTSVM on large-scale datasets. Table 3 shows the comparisons of TBSVM, TWSVM, RELS-TWSVM, and the proposed LS-LSTSVM algorithms with respect to classification accuracy and computational time. From Table 3, one can observe that the TBSVM, TWSVM, and RELS-TWSVM algorithms suffer from high time complexity when the size of the datasets increases. We only report the numerical results of five large datasets as TBSVM, TWSVM, and RELS-TWSVM run out of memory. From Table 3, it is clearly visible that the best performance is achieved by

Table 1. Performance Comparison of the Proposed LS-LSTSVM with TBSVM, TWSVM, and RELS-TWSVM Using Linear Kernel

Datasets (Train size, Test size)	**TBSVM** *Accuracy (%)* *Time (s)*	**TWSVM** *Accuracy (%)* *Time (s)*	**RELS-TWSVM** *Accuracy (%)* *Time (s)*	**LS-LSTSVM** *Accuracy (%)* *Time (s)*
Ionosphere	98.0952	98.0952	98.0952	**99.0476**
$(246 \times 34, 105 \times 34)$	0.0049	0.0048	0.0057	0.0044
WPBC	60.3448	74.1379	62.069	**81.0345**
$(136 \times 28, 58 \times 28)$	0.0179	0.0186	0.0265	0.0050
Oocytes-trisopterus-nucleus-2f	60.219	70.073	**70.438**	67.8832
$(638 \times 25, 274 \times 25)$	0.0984	0.0756	0.0076	0.0074
Blood-transfusion	80.8036	79.7291	80.8036	**81.6964**
$(524 \times 4, 224 \times 4)$	0.1125	0.3311	0.0032	0.0027
Breast-cancer-wisc-diag	**98.2456**	95.3216	**98.2456**	**98.2456**
$(398x \times 30, 171 \times 30)$	0.0092	0.0055	0.0013	0.0024
Breast-cancer-wisc	98.5714	98.0952	98.5714	**99.5238**
$(490 \times 9, 209 \times 9)$	0.0070	0.0051	0.0071	0.0012
Haberman-survival	73.913	61.9565	76	**76.087**
$(214 \times 3, 92 \times 3)$	0.0261	0.0287	0.0050	0.0093
Acute-inflammation	99.01	98.945	**100**	**100**
$(84 \times 6, 37 \times 6)$	0.0059	0.0066	0.0061	0.0030
Echo-cardiogram	92	91.9201	90.3	**92.3077**
$(92 \times 10, 39 \times 10)$	0.0060	0.0026	0.0013	0.0012
Cylinder-bands	60.7532	60.9481	60.5584	**63.6364**
$(358 \times 35, 54 \times 35)$	0.0113	0.0103	0.0039	0.0021
Acute-nephritis	98.89	99.91	**100**	**100**
$(84 \times 6, 37 \times 6)$	0.01003	0.0035	0.0036	0.0026
Parkinsons	67.7966	61.0169	**69.4915**	67.7966
$(136 \times 22, 59 \times 22)$	0.0118	0.0247	0.0048	0.0020
Heart-switzerland	51.3514	48.6486	67.5676	**72.973**
$(86 \times 12, 37 \times 12)$	0.0051	0.0049	0.0015	0.0011
Hepatitis	74.4681	63.8298	72.3404	**76.5957**
$(108 \times 19, 47 \times 19)$	0.0074	0.0049	0.0036	0.0031
Planning	63.6364	56.3636	63.6364	**65.4545**
$(127 \times 12, 55 \times 12)$	0.0075	0.0032	0.0024	0.0020
Musk-1	69.2308	51.049	67.8322	**72.028**
$(333 \times 166, 143 \times 166)$	0.0097	0.0087	0.0054	0.0051
Ilpd-indian-liver	70.8571	70	**73.1429**	72
$(408 \times 9, 75 \times 9)$	0.387	0.406	0.364	0.678
Mammographic	79.5139	**80.5556**	79.8611	73.6111
$(673 \times 5, 288 \times 5)$	0.0145	0.0262	0.0086	0.0245
Pima	79.1304	79.1304	79.1304	**80.4348**
$(538 \times 8, 230 \times 8)$	0.0582	0.0511	0.0315	0.036
Statlog-australian-credit	68	66.6667	66.6667	**68.599**
$(483 \times 14, 207 \times 14)$	0.0280	0.0602	0.0427	0.0137
Statlog-german-credit	63.667	72	78	**78.6667**
$(700 \times 24, 300 \times 24)$	0.0875	0.0262	0.0112	0.0197
Statlog-heart	84	83.12	86.4198	**87.6543**
$(189 \times 13, 81 \times 13)$	0.0812	0.346	0.0779	0.0191
Credit-approval	83.0918	83.0918	**84.058**	**84.058**
$(189 \times 13, 81 \times 13)$	0.0089	0.0061	0.0066	0.0084
Average Ranks	2.80	3.30	2.32	1.43
overall *Win-Tie-Loss*	0-1-22	1-0-22	3-4-16	15-4-4

The best accuracy is shown by bold face.

Table 2. Performance Comparison of the Proposed LS-LSTSVM with TBSVM, TWSVM, and RELS-TWSVM Using Gaussian Kernel

Datasets (Train size, Test size)	**TBSVM** *Accuracy (%)* *Time (s)*	**TWSVM** *Accuracy (%)* *Time (s)*	**RELS-TWSVM** *Accuracy (%)* *Time (s)*	**LS-LSTSVM** *Accuracy (%)* *Time (s)*
Iono-sphere	91.75	92	96.11	**96.201**
$(246 \times 34, 105 \times 34)$	0.014385	0.0124	0.00991	0.00934
WPBC	**81.0345**	**81.0345**	62.069	**81.0345**
$(136 \times 28, 58 \times 28)$	0.0047	0.122	0.0049	0.0033
Oocytes-trisopterus-nucleus-2f	66.7883	66.0584	**68**	67.8832
$(638 \times 25, 274 \times 25)$	0.957	0.811	0.258	0.206
Blood-transfusion	81.19	71.4286	80.3571	**81.25**
$(524 \times 4, 224 \times 4)$	0.1125	0.3311	0.0032	0.0027
Breast-cancer-wisc-diag	94.7368	77.193	95.9064	**97.6608**
$(398 \times 30, 171 \times 30)$	0.081	0.0594	0.0522	0.0477
Breast-cancer-wisc	97.1429	78.5714	97	**97.619**
$(490 \times 9, 209 \times 9)$	0.0659	0.0656	0.0305	0.0290
Haberman-survival	73.913	71.7391	**77**	72.8261
$(214 \times 3, 92 \times 3)$	0.298	0.927	0.035	0.0324
Acute-inflammation	77.7778	73.8889	**100**	**100**
$(84 \times 6, 37 \times 6)$	0.00597	0.00206	0.00193	0.00247
Echo-cardiogram	79.4872	76.9231	84.6154	**89.7436**
$(92 \times 10, 39 \times 10)$	0.0179	0.018	0.00175	0.00108
Acute-nephritis	97.2222	97.2222	88.8889	**100**
$(84 \times 6, 37 \times 6)$	0.0054	0.087	0.248	0.0087
Planning	**63.6364**	**63.6364**	54.5455	**63.6364**
$(127 \times 12, 55 \times 12)$	0.0737	0.074	0.06628	0.06998
Musk-1	79.021	79.021	79.021	**86.7133**
$(333 \times 166, 143 \times 166)$	0.0377	0.0196	0.137	0.0178
Ilpd-indian-liver	68.5714	70.2857	71.4286	**73.1429**
$(408 \times 9, 75 \times 9)$	0.8257	0.7252	0.1775	0.108
Breast-cancer-wisc-prog	81.3559	79.661	**84.7458**	79.661
$(139 \times 33, 59 \times 33)$	0.059	0.0874	0.03151	0.0209
Mammographic	68.5714	68.0556	79.2083	**80.2083**
$(673 \times 5, 288 \times 5)$	0.1855	0.972	0.6642	0.0529
Pima	74.3478	76.48	80.68	**80.92**
$(538 \times 8, 230 \times 8)$	0.0989	0.3717	0.0826	0.0734
Statlog-german-credit	69	63	**78.3333**	78
$(700 \times 24, 300 \times 24)$	0.346	0.704	0.311	0.359
Statlog-heart	83.8	82	85.1852	**86.4198**
$(189 \times 13, 81 \times 13)$	0.5404	0.0777	0.0817	0.0747
Credit-approval	**84.5411**	56.5217	83.5749	75.3623
$(189 \times 13, 81 \times 13)$	0.563	0.114	0.0563	0.0547
Heart-switzerland	**59.4595**	**59.4595**	54.0541	59
$(86 \times 12, 37 \times 12)$	0.05534	0.0255	0.0109	0.0147
Average Ranks	2.65	3.37	2.32	1.65
overall *Win-Tie-Loss*	1-3-16	0-3-17	4-1-15	11-3-7

The best accuracy is shown by bold face.

Table 3. Performance Comparison of the Proposed LS-LSTSVM with TBSVM, TWSVM, and RELS-TWSVM Using Gaussian Kernel on Large-Scale Datasets

Datasets (Data size)	**TBSVM** *Accuracy (%)* *Time (s)*	**TWSVM** *Accuracy (%)* *Time (s)*	**RELS-TWSVM** *Accuracy (%)* *Time (s)*	**LS-LSTSVM** *Accuracy (%)* *Time (s)*
Bank	87.6755	88.4218	87.823	**88.5693**
(4521×16)	7.25	6.90	3.52	**2.34**
Spam-base	85.2899	99.7826	81.7391	**100**
(4601×58)	9.89	10.62	8.29	**2.81**
Magic	99.5619	99.5619	99.5619	**100**
(19020×10)	267.738	260.584	284.684	**15.2722**
Connect-4			**70.0301**	70
(67557×42)	*	*	23196.1	**280.046**
Miniboone				**100**
(130064×50)	*	*	*	**1852.01**

Boldface shows best result.
*Out of memory.

the proposed LS-LSTSVM. In terms of accuracy and time consumption, the proposed LS-LSTSVM gives better accuracies in much less time. As we can see, for classification of Connect-4, Magic, and Miniboone, the proposed LS-LSTSVM gives better or similar accuracies with less time. It is simply visible that the proposed LS-LSTSVM classifies faster than TBSVM, TWSVM, and RELS-TWSVM on all the datasets.

5.2.3 Comparison on NDC Datasets. From comparisons in the previous subsections, we reached the conclusion that the proposed LS-LSTSVM performs better on most of the UCI benchmark small and large datasets. Further, to show the advantage of the proposed LS-LSTSVM in the training speed, we conducted numerical experiments on NDC datasets, as an example of large-scale datasets. The David Musicants NDC Data Generator [24] is used to generate datasets with the size increased from 10^3 to 10^5, while the number of features is fixed to be 32. Table 4 reports the results on NDC datasets detailing accuracies and training time. The best accuracy and the least computing time are highlighted in the table. According to the table, when the size of samples increases, TBSVM and TWSVM tend to run out of memory and the proposed LS-LSTSVM classifies much faster. For example, NDC-50K, TBSVM, and TWSVM ran out of memory; RELS-TWSVM took a very long time; and the proposed LS-LSTSVM classified in 253.232s.

6 STATISTICAL ANALYSIS

In this section, we perform statistical tests viz. the Friedman test [7] and Nemenyi post hoc test in the linear and non-linear cases to verify the statistical significance of the proposed LS-LSTSVM in comparison to TBSVM, TWSVM, and RELS-TWSVM. We report the average ranks of TBSVM, TWSVM, RELS-TWSVM, and the proposed LS-LSTSVM on accuracies with linear and non-linear kernels in the last rows of Tables 1 and 2. We applied the Friedman test to check whether the measured ranks are significantly different from the mean rank $R_i = 2.5$. Friedman statistics is distributed under the null hypothesis according to χ_F^2 with $k-1$ degrees of freedom

$$\chi_F^2 = \frac{12N}{k(k+1)}\left[\sum_i R_i^2 - \frac{k(k+1)^2}{4}\right],$$

Table 4. Performance Comparison of the Proposed LS-LSTSVM with TBSVM, TWSVM, and RELS-TWSVM Using Gaussian Kernel on NDC Datasets

Datasets (Data size)	**TBSVM** *Accuracy (%)* *Time (s)*	**TWSVM** *Accuracy (%)* *Time (s)*	**RELS-TWSVM** *Accuracy (%)* *Time (s)*	**LS-LSTSVM** *Accuracy (%)* *Time (s)*
NDC-10K	76.36 40.0175	76.36 29.537	76 45.333	76 **8.8475**
NDC-15K	70 103.965	70 77.5899	70.667 133.805	70.667 **18.9641**
NDC-20K	79 208.167	78 156.529	78.5 293.807	78.01 **33.2295**
NDC-25K	75.33 365.024	75.6 274.811	73 557.494	75.6 **53.228**
NDC-30K	80.667 554.244	80 431.864	80.667 801.541	80.667 **79.37**
NDC-40K	73.01 1144.55	73 908.85	72 2209.97	73.01 **154.019**
NDC-45K	69.6067 1462.48	70.667 1220.32	70.667 3220.01	69.333 **191.215**
NDC-50K	*	*	71.2 4612.54	71 **253.232**
NDC-75K	*	*	*	70.04 **574.25**
NDC-100K	*	*	*	69 **2896.61**

The least computational time is shown by boldface.
*Out of memory.

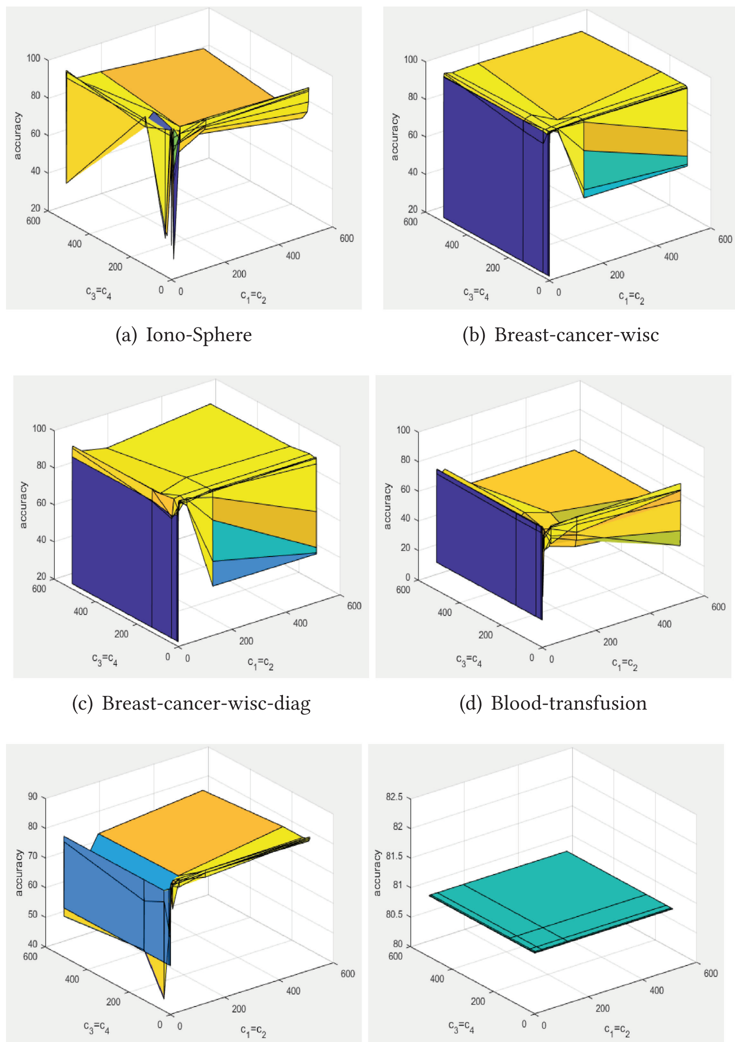
where k and N represent the number of algorithms compared and number of datasets, respectively.

$$\chi_F^2 = \frac{12 \times 23}{4(4+1)}\left[(2.80)^2 + (3.30)^2 + (2.32)^2 + (1.43)^2 - \frac{4(4+1)^2}{4}\right] = 15.9707.$$

$$F_F = \frac{(N-1)\chi_F^2}{N(k-1) - \chi_F^2} = \frac{(23-1)\chi_F^2}{23(3) - \chi_F^2} = 6.6257.$$

F_F is distributed according to the $F-$ distribution with $(3, 3 \times 22) = (3, 66)$ degrees of freedom on four algorithms and 23 datasets. The critical value of $F(3, 66)$ for $\alpha = 0.05$ level of significance is 2.74. We reject the null hypothesis since the value of $F_F \geq F(3, 66)$. We concluded that there is a significant difference between the algorithms. Now we use the Nemenyi test for further pairwise comparison. At $p = 0.10$ the critical difference (CD) $= q_\alpha\sqrt{\frac{k(k+1)}{6N}} = 2.291\sqrt{\frac{4\times5}{6\times23}} = 0.8721$. We can easily see that the difference between RELS-TWSVM and LS-LSTSVM is larger than CD ($2.32 - 1.43 = 0.89 > 0.8721$), which concludes that the generalization performance of our LS-LSTSVM is superior to RELS-TWSVM. In a similar manner, we can conclude that the proposed LS-LSTSVM is significantly better than TBSVM and TWSVM.

Further, we see the performance of TBSVM, TWSVM, and RELS-TWSVM and the proposed LS-LSTSVM statistically on accuracies for the nonlinear case. Under the null hypothesis, the Friedman



(a) Iono-Sphere

(b) Breast-cancer-wisc

(c) Breast-cancer-wisc-diag

(d) Blood-transfusion

(e) Credit-approval

(f) WPBC

Fig. 1. Effects of $c_1 = c_2$ and $c_3 = c_4$ on accuracy for the proposed linear LS-LSTSVM on six datasets.

statistic is given by

$$\chi_F^2 = \frac{12 \times 20}{4(4+1)}\left[(2.65)^2 + (3.37)^2 + (2.32)^2 + (1.65)^2 - \frac{4(4+1)^2}{4}\right] = 17.8116,$$

$$F_F = \frac{(N-1)\chi_F^2}{N(k-1) - \chi_F^2} = \frac{(20-1)\chi_F^2}{20(3) - \chi_F^2} = 8.0216.$$

With four algorithms, F_F is distributed according to F distribution with $(k-1)$ and $(k-1)(N-1) = (3, 57)$ degrees of freedom. The critical value for $\alpha = 0.05$ is $F(3, 57) = 2.77$. Since the value of F_F is greater than $F(3, 57)$, we reject the null hypothesis. Therefore, we used the Nemenyi test for further analysis. At $p = 0.10$, CD $= 2.291\sqrt{\frac{4\times5}{6\times20}} = 0.9353$. We can easily see that the difference between LS-LSTSVM and TBSVM is larger than CD; hence, the performance of the proposed

Table 5. Statistical Difference for TBSVM, TWSVM, and RELS-TWSVM and the Proposed LS-LSTSVM Algorithms Based on Friedman Ranking Test

	TBSVM	**TWSVM**	**RELS-TWSVM**
TBSVM			
TWSVM			
RELS-TWSVM		✓	
LS-LSTSVM	✓	✓	✓

Linear kernel was employed.
The rows with ✓ entries show that the two methods are statistically different and the method in the row is better than the method in the column. Blank entries show that no statistical difference exists among the methods given in the column and row.

Table 6. Statistical Difference for TBSVM, TWSVM, and RELS-TWSVM and the Proposed LS-LSTSVM Algorithms Based on Friedman Ranking Test

	TBSVM	**TWSVM**
TBSVM		
TWSVM		
RELS-TWSVM		✓
LS-LSTSVM	✓	✓

Gaussian kernel was employed.
The rows with ✓ entries show that the two methods are statistically different and the method in the row is better than the method in the column. Blank entries show that no statistical difference exists among the methods given in the column and row.

LS-LSTSVM is superior to TBSVM. Similarly, we can conclude that the performance of the proposed LS-LSTSVM is superior to TWSVM. Further, the difference between RELS-TWSVM and LS-LSTSVM is smaller than CD and thus we conclude that the post hoc test is not powerful enough to detect any significant difference between the algorithms.

Tables 5 and 6 summarize the statistical test for LS-LSTSVM. In these tables, the tick mark indicates the existence of a significant difference between algorithms among the row and column method. The row method is significantly better than the column method.

7 CONCLUSIONS AND FUTURE DIRECTIONS

In this article, we proposed a novel LS-LSTSVM for binary classification. We obtained an unconstrained dual optimization problem that can efficiently be solved using a fast iterative scheme named SMO, which made it more suitable for large-scale analysis. The proposed LS-LSTSVM does not need to compute the inverse of the large matrix that is predestined for the TBSVM, TWSVM, and RELS-TWSVM algorithms. Moreover, we can directly apply the kernel trick for the nonlinear LS-LSTSVM. The supremacy of the proposed LS-LSTSVM is justified on various real-world benchmark datasets on both small- and large-scale datasets. The proposed LS-LSTSVM can be applied on biomedical applications. Due to its lesser computation cost, the proposed LS-LSTSVM can be very effective for multiclass classification. Our LS-LSTSVM MATLAB codes are available on the author's Github page: https://github.com/mtanveer1/.

ACKNOWLEDGMENTS

We gratefully acknowledge the Indian Institute of Technology Indore for providing facilities and support.

REFERENCES

[1] C. Blake and C. J. Merz. 1998. UCI Repository of Machine Learning Databases, Dept. of Information and Computer Science, Univ. of California, Irvine.
[2] Danushka Bollegala, Yutaka Matsuo, and Mitsuru Ishizuka. 2011. A web search engine-based approach to measure semantic similarity between words. *IEEE Transactions on Knowledge and Data Engineering* 23, 7 (2011), 977–990.
[3] Christopher J. C. Burges. 1998. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery* 2, 2 (1998), 121–167.
[4] Chih-Chung Chang and Chih-Jen Lin. 2011. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology* 2, 3, Article 27 (May 2011), 27 pages. DOI: https://doi.org/10.1145/1961189.1961199
[5] Feng Chu, Guosheng Jin, and Lipo Wang. 2005. Cancer diagnosis and protein secondary structure prediction using support vector machines. In *Support Vector Machines: Theory and Applications*. Springer, 343–363.
[6] C. Cortes and V. Vapnik. 1995. Support vector networks. *Machine Learning* 20 (1995), 273–297.
[7] Janez Demšar. 2006. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research* 7, (Jan.2006), 1–30.
[8] Richard O. Duda, Peter E. Hart, and David G. Stork. 2012. *Pattern Classification*. John Wiley & Sons.
[9] Manuel Fernández-Delgado, Eva Cernadas, Senén Barro, and Dinani Amorim. 2014. Do we need hundreds of classifiers to solve real world classification problems? *Journal of Machine Learning Research* 15, 1 (Jan. 2014), 3133–3181. Retrieved from http://dl.acm.org/citation.cfm?id=2627435.2697065.
[10] Michael Grant and Stephen Boyd. 2014. CVX: Matlab software for disciplined convex programming, version 2.1. http://cvxr.com/cvx/citing/.
[11] Michael Grant, Stephen Boyd, and Yinyu Ye. 2009. cvx users' guide. *Technical Report, Technical Report Build 711, Citeseer.*
[12] Chih-Wei Hsu and Chih-Jen Lin. 2002. A comparison of methods for multiclass support vector machines. *IEEE Transactions on Neural Networks* 13, 2 (2002), 415–425.
[13] Jayadeva, R. Khemchandani, and S. Chandra. 2007. Twin support vector machines for pattern classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 29, 5 (2007), 905–910.
[14] S. S. Keerthi and S. K. Shevade. 2003. SMO algorithm for least squares SVM. In *Proceedings of the International Joint Conference on Neural Networks, 2003*, Vol. 3. IEEE, 2088–2093.
[15] S. Sathiya Keerthi and Elmer G. Gilbert. 2002. Convergence of a generalized SMO algorithm for SVM classifier design. *Machine Learning* 46, 1–3 (2002), 351–360.
[16] S. Sathiya Keerthi, Shirish Krishnaj Shevade, Chiranjib Bhattacharyya, and Karuturi Radha Krishna Murthy. 2001. Improvements to Platt's SMO algorithm for SVM classifier design. *Neural Computation* 13, 3 (2001), 637–649.
[17] Reshma Khemchandani, Pooja Saigal, and Suresh Chandra. 2018. Angle-based twin support vector machine. *Annals of Operations Research* 269, 1–2 (2018), 387–417.
[18] Reshma Khemchandani and Sweta Sharma. 2016. Robust least squares twin support vector machine for human activity recognition. *Applied Soft Computing* 47 (2016), 33–46.
[19] M. A. Kumar and M. Gopal. 2009. Least squares twin support vector machines for pattern classification. *Expert Systems with Applications* 36 (2009), 7535–7543.
[20] Yuanqing Li and Cuntai Guan. 2008. Joint feature re-extraction and classification using an iterative semi-supervised support vector machine algorithm. *Machine Learning* 71, 1 (2008), 33–53.
[21] Dalian Liu, Dewei Li, Yong Shi, and Yingjie Tian. 2018. Large-scale linear nonparallel SVMs. *Soft Computing* 22, 6 (2018), 1945–1957.
[22] O. L. Mangasarian and E. W. Wild. 2006. Multisurface proximal support vector classification via generalized eigenvalues. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 28, 1 (2006), 69–74.
[23] Olvi L. Mangasarian. 1994. *Nonlinear Programming*. SIAM.
[24] D. R. Musicant. 1998. NDC: Normally Distributed Clustered Datasets. Retrieved from http://www.cs.wisc.edu/dmi/svm/ndc/.
[25] Jalal A. Nasiri, Nasrollah Moghadam Charkari, and Kourosh Mozafari. 2014. Energy-based model of least squares twin support vector machines for human action recognition. *Signal Processing* 104 (2014), 248–257.
[26] Edgar Osuna, Robert Freund, and Federico Girosit. 1997. Training support vector machines: An application to face detection. In *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. IEEE, 130–136.

[27] Mahesh Pal and P. M. Mather. 2005. Support vector machines for classification in remote sensing. *International Journal of Remote Sensing* 26, 5 (2005), 1007–1011.
[28] B. Richhariya, A. Sharma, and M. Tanveer. 2018. Improved universum twin support vector machine. In *2018 IEEE Symposium Series on Computational Intelligence (SSCI'18)*. IEEE, 2045–2052.
[29] B. Richhariya and M. Tanveer. 2018. EEG signal classification using universum support vector machine. *Expert Systems with Applications* 106 (2018), 169–182.
[30] B. Richhariya and M. Tanveer. 2021. An efficient angle based universum least squares twin support vector machine for pattern classification. *ACM Transactions on Internet Technology (TOIT)* (In press) (2021). https://doi.org/10.1145/3387131
[31] B. Richhariya and M. Tanveer. 2020. A reduced universum twin support vector machine for class imbalance learning. *Pattern Recognition* 102 (2020), 107150.
[32] B. Richhariya, M. Tanveer, A. H. Rashid, and Alzheimer's Disease Neuroimaging Initiative. 2020. Diagnosis of Alzheimer's disease using universum support vector machine based recursive feature elimination (USVM-RFE). *Biomedical Signal Processing and Control* 59 (2020), 101903.
[33] Bernhard Schölkopf, Alexander J. Smola, Francis Bach, et al. 2002. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond.* MIT Press.
[34] Xigao Shao, Kun Wu, and Bifeng Liao. 2013. Single directional SMO algorithm for least squares support vector machines. *Computational Intelligence and Neuroscience* 2013, Article 968438 (2013).
[35] Yuan-Hai Shao, Wei-Jie Chen, Zhen Wang, Chun-Na Li, and Nai-Yang Deng. 2015. Weighted linear loss twin support vector machine for large-scale classification. *Knowledge-Based Systems* 73 (2015), 276–288.
[36] Yuan-Hai Shao, Chun-Na Li, Ming-Zeng Liu, Zhen Wang, and Nai-Yang Deng. 2018. Sparse Lq-norm least squares support vector machine with feature selection. *Pattern Recognition* 78 (2018), 167–181.
[37] Yuan-Hai Shao, Chun-Hua Zhang, Xiao-Bo Wang, and Nai-Yang Deng. 2011. Improvements on twin support vector machines. *IEEE Transactions on Neural Networks* 22, 6 (2011), 962–968.
[38] S. Sharma, R. Rastogi, and S. Chandra. 2021. Large-scale twin parametric support vector machine using Pinball loss function. *IEEE Transactions on Systems, Man, and Cybernetics: Systems* 51, 2 (2021), 987–1003.
[39] Shirish K. Shevade, S. Sathiya Keerthi, Chiranjib Bhattacharyya, and Karaturi Radha Krishna Murthy. 2000. Improvements to the SMO algorithm for SVM regression. *IEEE Transactions on Neural Networks* 11, 5 (2000), 1188–1193.
[40] M. Tanveer. 2015. Application of smoothing techniques for linear programming twin support vector machines. *Knowledge and Information Systems* 45, 1 (2015), 191–214.
[41] M. Tanveer. 2015. Newton method for implicit Lagrangian twin support vector machines. *International Journal of Machine Learning and Cybernetics* 6, 6 (2015), 1029–1040.
[42] M. Tanveer. 2015. Robust and sparse linear programming twin support vector machines. *Cognitive Computation* 7, 1 (2015), 137–149.
[43] M. Tanveer, C. Gautam, and Ponnuthurai N. Suganthan. 2019. Comprehensive evaluation of twin SVM based classifiers on UCI datasets. *Applied Soft Computing* 83 (2019), 105617.
[44] M. Tanveer, M. A. Khan, and S.-S. Ho. 2016. Robust energy-based least squares twin support vector machines. *Applied Intelligence.* 45, 1 (2016), 174–186.
[45] M. Tanveer, T. Rajani, and M. A. Ganaie. 2019. Improved sparse pinball twin SVM. In *2019 IEEE International Conference on Systems, Man and Cybernetics (SMC'19)*. IEEE, 3287–3291.
[46] M. Tanveer, B. Richhariya, R. U. Khan, A. H. Rashid, P. Khanna, M. Prasad, and C. T. Lin. 2020. Machine learning techniques for the diagnosis of Alzheimer's disease: A review. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)* 16, 1s (2020), 1–35.
[47] M. Tanveer, A Sharma, and Ponnuthurai N. Suganthan. 2019. General twin support vector machine with pinball loss function. *Information Sciences* 494 (2019), 311–327.
[48] M. Tanveer, A. Sharma, and Ponnuthurai N. Suganthan. 2020. Least squares KNN-based weighted multiclass twin SVM. *Neurocomputing* (In Press) (2020). DOI: https://doi.org/10.1016/j.neucom.2020.02.132
[49] M. Tanveer and K. Shubham. 2017. Smooth twin support vector machines via unconstrained convex minimization. *Filomat* 31, 8 (2017), 2195–2210.
[50] M. Tanveer, Aruna Tiwari, Rahul Choudhary, and Sanchit Jalan. 2019. Sparse pinball twin support vector machines. *Applied Soft Computing* 78 (2019), 164–175.
[51] Yingjie Tian, Xuchan Ju, Zhiquan Qi, and Yong Shi. 2014. Improved twin support vector machine. *Science China Mathematics* 57, 2 (2014), 417–432.
[52] Vladimir Vapnik. 2013. *The Nature of Statistical Learning Theory.* Springer Science & Business Media.
[53] Zhen Wang, Yuan-Hai Shao, Lan Bai, Chun-Na Li, Li-Ming Liu, and Nai-Yang Deng. 2018. Insensitive stochastic gradient twin support vector machines for large scale problems. *Information Sciences* 462 (2018), 114–131.

[54] Yitian Xu, Zhiji Yang, and Xianli Pan. 2016. A novel twin support-vector machine with pinball loss. *IEEE Transactions on Neural Networks and Learning Systems* 28, 2 (2016), 359–370.

[55] Zhi-Qiang Zeng, Hong-Bin Yu, Hua-Rong Xu, Yan-Qi Xie, and Ji Gao. 2008. Fast training support vector machines using parallel sequential minimal optimization. In *2008 3rd International Conference on Intelligent System and Knowledge Engineering*, Vol. 1. IEEE, 997–1001.

Received March 2020; revised April 2020; accepted May 2020