Hindawi Publishing Corporation
The Scientific World Journal
Volume 2015, Article ID 497617, 12 pages
http://dx.doi.org/10.1155/2015/497617

Research Article

A Learning Framework of Nonparallel Hyperplanes Classifier

Zhi-Xia Yang,[1,2] Yuan-Hai Shao,[3] and Yao-Lin Jiang[1]

[1]*College of Mathematics and Systems Science, Xinjiang University, Urumqi 830046, China*
[2]*State Key Lab of Biochemical Engineering, Institute of Process Engineering, Chinese Academy of Sciences, Beijing 100190, China*
[3]*Zhijiang College, Zhejiang University of Technology, Hangzhou 310024, China*

Correspondence should be addressed to Yuan-Hai Shao; shaoyuanhai21@163.com

Received 21 June 2014; Revised 19 September 2014; Accepted 19 September 2014

Academic Editor: Qiankun Song

A novel learning framework of nonparallel hyperplanes support vector machines (NPSVMs) is proposed for binary classification and multiclass classification. This framework not only includes twin SVM (TWSVM) and its many deformation versions but also extends them into multiclass classification problem when different parameters or loss functions are chosen. Concretely, we discuss the linear and nonlinear cases of the framework, in which we select the hinge loss function as example. Moreover, we also give the primal problems of several extension versions of TWSVM's deformation versions. It is worth mentioning that, in the decision function, the Euclidean distance is replaced by the absolute value $|w^{\mathrm{T}}x + b|$, which keeps the consistency between the decision function and the optimization problem and reduces the computational cost particularly when the kernel function is introduced. The numerical experiments on several artificial and benchmark datasets indicate that our framework is not only fast but also shows good generalization.

1. Introduction

Classification problem is an important issue in machine learning and data mining, which is mainly comprised of binary and multiclass classification. Support vector machine (SVM), proposed by Burges [1] and Cortes and Vapnik [2], is an excellent tool for classification. In contrast with conventional artificial neural networks (ANNS) which aim at reducing empirical risk, SVM is principled and implements the structural risk minimization (SRM) that minimizes the upper bound of the generalization error [3–5]. Within a few years after its introduction, SVM has been successfully applied to pattern classification and regression estimation like face detection [6, 7], text categorization [8], time series prediction [9], bioinformatics [10], and so forth.

Recently, for binary classification, Mangasarian and Wild [11] proposed the generalized eigenvalue proximal support vector machine (GEPSVM) via two nonparallel hyperplanes. In their approach, the data points of each class are proximal to one of two nonparallel hyperplanes. The nonparallel hyperplanes are determined by eigenvectors corresponding to the smallest eigenvalues of two related generalized eigenvalue problems. Inspired by GEPSVM [11], Jayadeva et al. [12] developed twin SVM (TWSVM) with two nonparallel hyperplanes. However, the two hyperplanes are got by solving two quadratic programming (QP) problems, similar to the standard SVM. Furthermore, TWSVM differs from the standard SVM in fundamental way. In TWSVM, one solves a pair of smaller size QP problems rather than a single QP problem in the standard SVM. Therefore, TWSVM works faster than the standard SVM. Subsequently, there are many extensions for TWSVM including the improvements on TWSVM (TBSVM) [13], the least square TWSVM (LS-TWSVM) [14–17], nonparallel plane proximal classifier (NPPC) [18], smooth TWSVM [19], geometric algorithm [20], and twin support vector regression (TWSVR) [21]. TWSVM was also extended to deal with multiclassification TWSVM [22–24]. More precisely, in [22], TWSVM was extended straight from binary classification to multiclass classification, in which each primal problem covers all patterns except the patterns of the kth class in the constraints for the kth ($k = 1, 2, \ldots, K$) hyperplane. In [23], the authors extended TWSVM based on the idea of "one-versus-rest" (1-v-r) from binary classification to multiclass classification, in which there are two quadratic programming

(QP) problems for each reconstructing binary classification. However, they both have not kept the advantage of TWSVM which has lower computational complexity than that of the standard SVM. In [24], Yang et al. proposed multiple birth SVM (MBSVM) with much lower computational complexity than that of both [22, 23] by solving K smaller size of QP problems for K-class classification; only the empirical risk is considered like TWSVM. However, in TBSVM [13], the structural risk minimization principle is implemented by introducing the regularization term.

In this paper, we propose a novel learning framework of nonparallel hyperplanes support vector machines based on TWSVM and its extension versions, called NPSVMs, which not only provide a unified view for TWSVM and its many extension versions but also can deal with binary and multiclass classification problems. For binary classification, if the loss function is the hinge loss function, then the framework can become TWSVM [12] or TBSVM [13] with different parameters; if the loss function is the square loss function, then the framework is LS-TWSVM [14]; if the loss function is the convex combination of the linear and square loss functions, then the framework is NPPC [18]. Actually, we can also get smooth TWSVM [19] by replacing 2-norm with 1-norm in the framework. However, for multiclass classification, the framework does not directly extend, in which we switch the roles of the patterns of the k-th class and the rest class and replace "min" with "max" in the decision function. Moreover, we only use the absolute value $|w^{\mathrm{T}}x + b|$ rather than the Euclidean distance in the decision function due to the twofold reasons: reducing the computational cost particularly when the kernel function is introduced and making the consistency since it is the corresponding absolute value that appears in the primal problems. Concretely, we discuss the linear and nonlinear cases of the framework, in which we select the hinge loss function as example. Moreover, we also give the primal problems of extensions of LS-TWSVM, 1-norm LS-TWSVM, NPPC, and smooth TWSVM. Finally, the numerical experiments on several artificial and benchmark datasets indicate that our frameworks are not only fast but also show good generalization.

The paper is organized as follows. Section 2 introduces the brief reviews of SVMs. Section 3 proposes our frameworks, in which Section 3.1 discusses the linear framework, Section 3.2 extend into the nonlinear framework, Section 3.3 gives SOR algorithm for solving the hinge NPSVMs, and Section 3.4 discusses several other extension approaches. Finally, Section 4 deals with experimental results and Section 5 contains concluding remarks.

2. Brief Reviews of SVMs

2.1. Twin Support Vector Machine. Given the following training set for the binary classification:

$$\mathrm{T} = \{(x_1, y_1), \ldots, (x_l, y_l)\}, \tag{1}$$

where (x_i, y_i) is the ith data point, the input $x_i \in R^n$ is a pattern, the output $y_i \in \{1, 2\}$ is a class label, $i = 1, \ldots, l$, and l is the number of data points. In addition, let l_1 and l_2 be the number of data points in positive class and negative class, respectively, and $l = l_1 + l_2$. Furthermore, the matrices $A_1 \in R^{l_1 \times n}$ and $A_2 \in R^{l_2 \times n}$ consist of the l_1 inputs of Class 1 and the l_2 inputs of Class 2, respectively.

The gaol of TWSVM [12] is to find two nonparallel hyperplanes in n-dimensional input space:

$$x^{\mathrm{T}}w_1 + b_1 = 0, \tag{2}$$

$$x^{\mathrm{T}}w_2 + b_2 = 0, \tag{3}$$

such that one hyperplane is close to the patterns of one class and far away from the patterns of the other class to some extent. TWSVM is in spirit of GEPSVM [11]. But both of GEPSVM and TWSVM are different from the standard SVM. For TWSVM, each hyperplane is generated by solving a QP problem looking like the primal problem of the standard SVM. The primal problems of TWSVM can be presented as follows:

$$\begin{aligned} \min_{w_1, b_1, \xi_2} \quad & \frac{1}{2}\left\|A_1 w_1 + e_1 b_1\right\|_2^2 + C_1 e_2^{\mathrm{T}} \xi_2, \\ \text{s.t.} \quad & -\left(A_2 w_1 + e_2 b_1\right) + \xi_2 \geq e_2, \\ & \xi_2 \geq 0; \end{aligned} \tag{4}$$

$$\begin{aligned} \min_{w_2, b_2, \xi_1} \quad & \frac{1}{2}\left\|A_2 w_2 + e_2 b_2\right\|_2^2 + C_2 e_1^{\mathrm{T}} \xi_1, \\ \text{s.t.} \quad & \left(A_1 w_2 + e_1 b_2\right) + \xi_1 \geq e_1, \\ & \xi_1 \geq 0, \end{aligned} \tag{5}$$

where C_1 and C_2 are nonnegative parameters and e_1 and e_2 are vectors of ones of appropriate dimensions. In the QP problem (4), the objective function tends to keep hyperplane (2) close to the patterns of Class 1 and the constraints require the hyperplane (2) to be at a distance of at least 1 from the patterns of Class 2. The QP problem (5) has similar property. Moreover, we note that the constraints do not contain all patterns in the training set (1) but are determined by only the patterns of one class in both classes. Therefore, in [12], the authors claimed that TWSVM is approximately four times faster than the standard SVM.

Define $G = [A_2 \ e_2]$ and $H = [A_1 \ e_1]$. It has been shown that when both $G^{\mathrm{T}}G$ and $H^{\mathrm{T}}H$ are positive definites, the Wolfe duals of (4) and (5) are written as follows:

$$\begin{aligned} \max_{\alpha_2} \quad & e_2^{\mathrm{T}}\alpha_2 - \frac{1}{2}\alpha_2^{\mathrm{T}} G \left(H^{\mathrm{T}}H\right)^{-1} G^{\mathrm{T}}\alpha_2, \\ \text{s.t.} \quad & 0 \leq \alpha_2 \leq C_1, \end{aligned} \tag{6}$$

$$\begin{aligned} \max_{\alpha_1} \quad & e_1^{\mathrm{T}}\alpha_1 - \frac{1}{2}\alpha_1^{\mathrm{T}} H \left(G^{\mathrm{T}}G\right)^{-1} H^{\mathrm{T}}\alpha_1, \\ \text{s.t.} \quad & 0 \leq \alpha_1 \leq C_2, \end{aligned} \tag{7}$$

respectively, where α_2 and α_1 are Lagrangian multipliers.

In order to avoid the possible ill-conditioning of $H^{\mathrm{T}}H$ and $G^{\mathrm{T}}G$, TWSVM introduces a term ϵI ($\epsilon > 0$), where I

is an identity matrix of appropriate dimensions. Thus, the nonparallel hyperplanes (2) and (3) can be obtained from the solutions α_1 and α_2 of the QP problems (6) and (7). Consider

$$z_1 = -\left(H^{\mathrm{T}}H + \epsilon I\right)^{-1} G^{\mathrm{T}}\alpha_2, \qquad z_2 = \left(G^{\mathrm{T}}G + \epsilon I\right)^{-1} H^{\mathrm{T}}\alpha_1, \tag{8}$$

where $z_k = \left[w_k^{\mathrm{T}} \;\; b_k\right]^{\mathrm{T}}$, $k = 1, 2$. Moreover, a new pattern $x \in R^n$ is assigned to Class k ($k = 1, 2$), depending on which of the two nonparallel hyperplanes given by (2) and (3) lies closer to; that is,

$$f(x) = \arg\min_{k=1,2} \frac{\left|w_k^{\mathrm{T}}x + b_k\right|}{\left\|w_k\right\|_2}. \tag{9}$$

2.2. Multiple Birth Support Vector Machine. Given the training set

$$\mathrm{T} = \left\{\left(x_1, y_1\right), \ldots, \left(x_l, y_l\right)\right\}, \tag{10}$$

where the input $x_i \in R^n$, $i = 1, \ldots, l$, is the pattern and the output $y_i \in \{1, \ldots, K\}$ is the class label. The task is to seek K hyperplanes,

$$w_k^{\mathrm{T}}x + b_k = 0, \quad k = 1, \ldots, K, \tag{11}$$

and assign the class label according to which hyperplane a new pattern is farthest from.

For convenience, denote the number of data points of the kth class in the training set (10) as l_k and define the following matrixes: the patterns belonging to the kth class are represented by the matrix $A_k \in R^{l_k \times n}$, $k = 1, \ldots, K$. In addition, define the matrix

$$B_k = \left[A_1^{\mathrm{T}}, \ldots, A_{k-1}^{\mathrm{T}}, A_{k+1}^{\mathrm{T}}, \ldots, A_K^{\mathrm{T}}\right]^{\mathrm{T}}; \tag{12}$$

that is, $B_k \in R^{(l-l_k)\times n}$ consists of the patterns belonging to all classes except the kth class, $k = 1, \ldots, K$. The primal problems of MPSVM [24] are comprised of the following K QP problem:

$$\begin{aligned} \min_{w_k, b_k, \xi_k} \quad & \frac{1}{2}\left\|B_k w_k + e_{k1} b_k\right\|^2 + C_k e_{k2}^{\mathrm{T}} \xi_k, \\ \text{s.t.} \quad & \left(A_k w_k + e_{k2} b_k\right) + \xi_k \geq e_{k2}, \quad \xi_k \geq 0, \end{aligned} \tag{13}$$

where $e_{k1} \in R^{(l-l_k)}$ and $e_{k2} \in R^{l_k}$ are the vectors of ones, ξ_k is the slack variable, and $C_k > 0$ is the penalty parameter, $k = 1, \ldots, K$. The dual problem of QP problem (13) is formulated as follows:

$$\begin{aligned} \max_{\alpha_k} \quad & e_{k2}^{\mathrm{T}}\alpha_k - \frac{1}{2}\alpha_k G_k \left(H_k^{\mathrm{T}}H_k\right)^{-1} G_k^{\mathrm{T}}\alpha_k, \\ \text{s.t.} \quad & 0 \leq \alpha_k \leq C_k, \end{aligned} \tag{14}$$

where the penalty parameter $C_k > 0$, $H_k = \left[B_k \;\; e_{k1}\right]$, and $G_k = \left[A_k \;\; e_{k2}\right]$, $k = 1, 2, \ldots, K$. Similarly, in order to avoid the possibility of the ill-conditioning of the matrix $H_k^{\mathrm{T}}H_k$ in some situations, one introduces a regularization term ϵI, where $\epsilon > 0$ is a fixed small scalar and I is the identity matrix with appropriate size.

After getting the solution $\left[w_k^{\mathrm{T}} \;\; b_k\right]^{\mathrm{T}} = -(H_k^{\mathrm{T}}H_k + \epsilon I)^{-1}G_k^{\mathrm{T}}\alpha_k$ to the above QP problem (13) with $k = 1, \ldots, K$, a new pattern $x \in R^n$ is assigned to class k ($k \in \{1, \ldots, K\}$), depending on which of the K hyperplanes given by (11) lies farthest from; that is, the decision function is represented as

$$f(x) = \arg\max_{k=1,\ldots,K} \frac{\left|w_k^{\mathrm{T}}x + b_k\right|}{\left\|w_k\right\|_2}, \tag{15}$$

where $|\cdot|$ is the absolute value.

3. The Framework of Nonparallel Hyperplanes Classifiers

In this section, we propose a learning framework of nonparallel hyperplanes classifier, which gives a unified form for TWSVM and its many extension versions and extend them into multiclass classification problem. We first develop the linear framework and then extend it to nonlinear framework.

3.1. Linear Framework. Given the training set (10), the task is to find K nonparallel hyperplanes:

$$w_k^{\mathrm{T}}x + b_k = 0, \quad k = 1, 2, \ldots, K, \tag{16}$$

one for each class. For obtaining the K unknown hyperplanes, we construct the following standard framework for each unknown hyperplane:

$$\begin{aligned} \min_{w_k, b_k} \; & \frac{1}{2}\left\|B_k w_k + e_{k2} b_k\right\|_2^2 + \frac{1}{2}C_k^*\left(\left\|w_k\right\|_2^2 + b_k^2\right) \\ & + C_k e_{k1}^{\mathrm{T}} L\left(e_{k1}, A_k w_k + e_{k1} b_k\right), \end{aligned} \tag{17}$$

where the matrix A_k is comprised of the patterns in the kth class, the matrix B_k is defined (12), $C_k^* \geq 0$ and $C_k > 0$ are the parameters, e_{k1} and e_{k2} are vectors of ones of appropriate dimensions, $k = 1, 2, \ldots, K$, and $L(\cdot, \cdot)$ is the loss function (e.g., square loss, hinge loss, etc.). In the optimization problem (17), the first term approximatively minimizes the sum of the squared Euclidean distances from the patterns except for the kth class to hyperplanes; the second term is the Tikhonov regularization term [25] and can implement the structural risk minimization principle like TBSVM [13]; the third term constitutes the loss function which is defined different loss functions corresponding to different models.

For a new pattern $x \in R^n$, we assign to class k ($k = 1, 2, \ldots, K$) according to the following decision function:

$$f(x) = \arg\max_{k=1,2,\ldots,K} \left|w_k^{\mathrm{T}}x + b_k\right|, \tag{18}$$

where $|\cdot|$ is the absolute value. Note that we only use the absolute value $|w^{\mathrm{T}}x + b|$ in the decision function. There are two main reasons: one is that the first term of the optimization problem (17) just minimizes the sum of the square rather

than the sum of square Euclidean distance from the patterns to hyperplanes, so it should keep consistency between the optimization problem and the decision function; another is that it reduces the computational cost particularly when the kernel function is introduced afterwards.

In fact, if $K = 2$, the parameter C_k^* is equal to 0, and the loss function is hinge loss function, that is, $L(1, g(x)) = \max(0, 1-g(x))$, then the optimization problem (17) becomes TWSVM [12]. Moreover, if the parameter $C_k^* > 0$ is alterable, then it is TBSVM [13]. And if the loss function is the square loss function, that is, $L(1, g(x)) = (1 - g(x))^2$, it is LS-TWSVM [14]. And if the loss function is a convex combination of linear and square loss, that is, $L(1, g(x)) = \delta(1 - g(x)) + (1 - \delta)(1 - g(x))^2$, where $\delta \in (0, 1)$, then it is NPPC [18]. Other extension versions of TWSVM also can be contained in the optimization problem (17), for instance, smooth TWSVM, 1-norm LS-TWSVM [17], and so forth, in which we just need to select proper norm or loss function.

More importantly, our framework can solve multiclass classification problem, which is extension of TWSVM, TBSVM, LS-TWSVM, NPPC, and so forth. It should be pointed out that our framework is not straight extension of TWSVM and its deformation versions. Concretely, from the optimization problem (17), we can see that the first term contains the patterns except for those of the kth class and the third term just involves the patterns of the kth class. This strategy cannot lead to significant increase of the complexity of the optimization when the number K of classes increases. We will dwell on in specific algorithm afterwards.

Now, we give the detailed algorithm to the hinge loss function as an example, called hinge NPSVM (HNPSVM). And then the optimization problem (17) is the following formulation with the hinge loss function:

$$\min_{w_k, b_k} \frac{1}{2} \left\| B_k w_k + e_{k2} b_k \right\|_2^2 + \frac{1}{2} C_k^* \left(\left\| w_k \right\|_2^2 + b_k^2 \right) + C_k e_{k1}^{\mathrm{T}} \max\left(0, e_{k1} - \left(A_k w_k + e_{k1} b_k\right)\right), \tag{19}$$

where the matrix A_k is comprised of the patterns in the kth class, the matrix B_k is defined (12), $C_k^* \geq 0$ and $C_k > 0$ are the parameters, e_{k1} and e_{k2} are vectors of ones of appropriate dimensions, and $k = 1, 2, \ldots, K$. Actually, the problem is equivalent to the following quadratic programming:

$$\begin{aligned} \min_{w_k, b_k, \xi_k} \quad & \left\| B_k w_k + e_{k2} b_k \right\|_2^2 + \frac{1}{2} C_k^* \left(\left\| w_k \right\|_2^2 + b_k^2 \right) + C_k e_{k1}^{\mathrm{T}} \xi_k, \\ \text{s.t.} \quad & \left(A_k w_k + e_{k1} b_k\right) + \xi_k \geq e_{k1}, \quad \xi_k \geq 0, \end{aligned} \tag{20}$$

where the matrix A_k is comprised of the patterns in the kth class, the matrix B_k is defined (12), $C_k^* \geq 0$ and $C_k > 0$ are the parameters, e_{k1} and e_{k2} are vectors of ones of appropriate dimensions, and $k = 1, 2, \ldots, K$.

In fact, for $k = 1, 2, \ldots, K$, we have K QP problems like (20). In particular, when K is equal to 2, that is, $k = 1, 2$, the QP problems (4) and (5) can be obtained as a special case of (20) with $C_k^* = 0$. For simplicity, assume that the number of each class points is almost balanced; namely, the number of the kth class is $l_k = l/K$. Then, note that the constraints just involve the patterns of the kth class, so the complexity of the the problem (20) is no more than $(l_k)^3 = (l/K)^3$. However, if TWSVM is directly extended to multiclass classification case like [22], we will get a different optimization problem, in which the roles of patterns of the kth class and the rest class are switched. Thus, the complexity of the optimization problem will increase significantly and is determined by the patterns except for the patterns of the kth class in the training set (10), which is no more than $((K-1)(l/K))^3$. Obviously, our approach is approximately $(K-1)^3$ times faster than the model in [22]. On the other hand, when the number of each class points is unbalanced, our apprach still is faster than the model in [22] because the complexity of our optimization problem just is decided by the number of the patterns of the kth class rather than the patterns of the rest classes. Therefore, our HNPSVM keeps the computation complexity low.

It is well known that the solution of primal problem (20) is obtained from the solutions of their dual problems. So we now derive their dual problems. The Lagrangian function of the problem (20) is given by

$$\begin{aligned} & L(w_k, b_k, \xi_k, \alpha_k, \eta_k) \\ & = \frac{1}{2} \left\| B_k w_k + e_{k2} b_k \right\|_2^2 + \frac{1}{2} C_k^* \left(\left\| w_k \right\|_2^2 + b_k^2 \right) + C_k e_{k1} \xi_k \\ & \quad - \alpha_k^{\mathrm{T}} \left(\left(A_k w_k + e_{k1} b_k\right) + \xi_k - e_{k1} \right) - \eta_k^{\mathrm{T}} \xi_k, \end{aligned} \tag{21}$$

where α_k, η_k are nonnegative Lagrange multiplier vectors. The Karush-Kuhn-Tucker (KKT) necessary and sufficient optimality conditions [26] for the QP problem (20) are given by

$$\nabla_{w_k} L = B_k^{\mathrm{T}} \left(B_k w_k + e_{k2} b_k\right) + C_k^* w_k - A_k^{\mathrm{T}} \alpha_k = 0, \tag{22}$$

$$\nabla_{b_k} L = e_{k2}^{\mathrm{T}} \left(B_k w_k + e_{k2} b_k\right) + C_k^* b_k - e_{k1}^{\mathrm{T}} \alpha_k = 0, \tag{23}$$

$$\nabla_{\xi_k} L = C_k e_{k1} - \alpha_k - \eta_k = 0, \tag{24}$$

$$\left(A_k w_k + e_{k1} b_k\right) + \xi_k \geq e_{k1}, \quad \xi_k \geq 0, \tag{25}$$

$$-\alpha_k^{\mathrm{T}} \left(\left(A_k w_k + e_{k1} b_k\right) + \xi_k - e_{k1} \right) = 0, \qquad \eta_k^{\mathrm{T}} \xi_k = 0, \tag{26}$$

$$\alpha_k \geq 0, \qquad \eta_k \geq 0. \tag{27}$$

Since $\eta_k \geq 0$, according to (24), we have

$$0 \leq \alpha_k \leq C_k. \tag{28}$$

Next, from (22) and (23), we can obtain

$$\left(\begin{bmatrix} B_k^{\mathrm{T}} & e_k^{\mathrm{T}} \end{bmatrix} \begin{bmatrix} B_K & e_k \end{bmatrix} + C_k^* I \right) \begin{bmatrix} w_k^{\mathrm{T}} & b_k \end{bmatrix}^{\mathrm{T}} - \begin{bmatrix} A_k^{\mathrm{T}} & e_k^{\mathrm{T}} \end{bmatrix} \alpha_k = 0, \tag{29}$$

where I is an identity matrix of appropriate dimensions. Let $v_k = \begin{bmatrix} w_k^{\mathrm{T}} & b_k \end{bmatrix}^{\mathrm{T}}$; (29) can be written as

$$\begin{aligned} & \left(H_k^{\mathrm{T}} H_k + C_k I\right) v_k - G_k^{\mathrm{T}} \alpha_k = 0, \\ & \text{or } v_k = \left(H_k^{\mathrm{T}} H_k + C_k I\right)^{-1} G_k^{\mathrm{T}} \alpha_k, \end{aligned} \tag{30}$$

where $H_k = [B_k \ \ e_{k2}]$ and $G_k = [A_k \ \ e_{k1}]$. And then putting (30) into the Lagrangian function (21) and using (22)–(28), we can get the dual problem of the primal problem (20):

$$\begin{aligned} \max_{\alpha_k} \quad & e_{k1}^{\mathrm{T}}\alpha_k - \frac{1}{2}\alpha_k^{\mathrm{T}} H_k \left(G_k^{\mathrm{T}} G_k + C_k^* I\right)^{-1} H_k^{\mathrm{T}}\alpha_k, \\ \text{s.t.} \quad & 0 \le \alpha_k \le C_k, \end{aligned} \tag{31}$$

where $C_k^* > 0$ and $C_k > 0$ are parameters and $k = 1, 2, \ldots, K$. Obviously, if we have the solution of the QP problem (31), then we obtain the K nonparallel hyperplanes (16) by (30).

It is worth mentioning that the parameter C_k^* replaces ϵ as in (8), so C_k^* is no longer a fixed small scalar but a weighting factor which determines the trade-off between the regularization term and the empirical risk in the problem (20). Therefore, the high and low of the value of C_k^* reflects the structure of minimization principle and our HNPSVM includes MBSVM.

3.2. Nonlinear Framework. Similarly, we also extend the linear framework of NPSVMs to nonlinear case. For a K-class classification (10), our goal is to find K kernel-generated hyperplanes:

$$K\left(x, A^{\mathrm{T}}\right) u_k + b_k = 0, \qquad k = 1, \ldots, K, \tag{32}$$

where $A = [A_1, \ldots, A_k]$ and $K(x, A^{\mathrm{T}})$ is an appropriately chosen kernel function.

In order to obtain the K hyperlanes (32), we construct the following framework formulation:

$$\begin{aligned} \min_{u_k, b_k} \ & \frac{1}{2}\left\|K(B_k^{\mathrm{T}}, A^{\mathrm{T}})u_k + e_{k2}b_k\right\|_2^2 + \frac{1}{2}C_k^*\left(\|u_k\|_2^2 + b_k^2\right) \\ & + C_k e_{k1}^{\mathrm{T}} L\left(e_{k1}, K\left(A_k^{\mathrm{T}}, A^{\mathrm{T}}\right)u_k + e_{k1}b_k\right), \end{aligned} \tag{33}$$

where the matrix A_k is comprised of the patterns in the kth class, the matrix B_k is defined (12), $C_k^* \ge 0$ and $C_k > 0$ are the parameters, e_{k1} and e_{k2} are vectors of ones of appropriate dimensions, $k = 1, 2, \ldots, K$, and $L(\cdot, \cdot)$ is the loss function (e.g., square loss or hinge loss, etc.). Similarly, as discussed in the last subsection, the problem (33) can be reduced to the nonlinear formulations of the difference approaches (e.g., TWSVM, TBSVM, LS-TWSVM, NPPC, etc.) when the difference loss functions or parameters are selected for $K = 2$.

A new pattern $x \in R^n$ is assigned to the kth class by the following decision functions:

$$f(x) = \arg \max_{k=1,\ldots,K} \left|K\left(x, A^{\mathrm{T}}\right)u_k + b_k\right|, \tag{34}$$

where $|\cdot|$ is the absolute value. Note that, in this decision function (34), we just compute the absolute value rather than Euclidean distance from the pattern x to the hyperplanes. This strategy reduces the complexity of computation because Euclidean distance should be $|K(x, A^{\mathrm{T}})u_k + b_k| / \sqrt{u_k^{\mathrm{T}} K(A^{\mathrm{T}}, A^{\mathrm{T}}) u_k}$ from the pattern x to the kth hyperplanes. Thus, the decision function (34) not only saves the computation quantity but also keeps the consistency with the first term of the problem (33).

Now, we still select the hinge loss function as example. Then, the problem (33) can be formulated as follows:

$$\begin{aligned} \min_{u_k, b_k, \xi_k} \quad & \frac{1}{2}\left\|K\left(B_k^{\mathrm{T}}, A^{\mathrm{T}}\right)u_k + e_{k2}b_k\right\|_2^2 \\ & + \frac{1}{2}C_k^*\left(\|u_k\|_2^2 + b_k^2\right) + C_k e_{k1}^{\mathrm{T}}\xi_k, \\ \text{s.t.} \quad & \left(K\left(A_k^{\mathrm{T}}, A^{\mathrm{T}}\right)u_k + e_{k1}b_k\right) + \xi_k \ge e_{k1}, \qquad \xi_k \ge 0, \end{aligned} \tag{35}$$

where the matrix A_k is comprised of the patterns in the kth class, the matrix B_k is defined by (12), $C_k^* > 0$ and $C_k > 0$ are parameters, e_{k1} and e_{k2} are vectors of ones of appropriate dimensions, and $k = 1, 2, \ldots, K$. Similarly, derived process with the linear case, its dual problem is formulated as:

$$\begin{aligned} \max_{\alpha_k} \quad & e_{k1}^{\mathrm{T}}\alpha_k - \frac{1}{2}\alpha_k^{\mathrm{T}} R_k \left(S_k^{\mathrm{T}} S_k + C_k^* I\right) R_k^{\mathrm{T}}\alpha_k, \\ \text{s.t.} \quad & 0 \le \alpha_k \le C_k, \end{aligned} \tag{36}$$

where $C_k^* > 0$ and $C_k > 0$ are parameters, $R_k = \left[K(A_k^{\mathrm{T}}, A^{\mathrm{T}}) \ \ e_{k1}\right]$, $S_k = \left[K(B_k^{\mathrm{T}}, A^{\mathrm{T}}) \ \ e_{k2}\right]$, and $k = 1, 2, \ldots, K$. And the augmented vector $z_k = \left[u_k \ \ b_k\right]^{\mathrm{T}}$ is given by $z_k = \left(S_k^{\mathrm{T}} S_k + C_k^* I\right)^{-1} R_k^{\mathrm{T}}\alpha_k$.

3.3. SOR Algorithm. In our HNPSVMs, the QP problems (31) and (36) can be rewritten as the following unified forms:

$$\begin{aligned} \min_{\alpha} \quad & \frac{1}{2}\alpha^{\mathrm{T}} Q\alpha - e^{\mathrm{T}}\alpha, \\ \text{s.t.} \quad & 0 \le \alpha \le C, \end{aligned} \tag{37}$$

where $Q \in R^{m \times m}$ is positive definite. For example, the above problem becomes the problem (36), when $Q = R_k(S_k^{\mathrm{T}} S_k + C_k^* I)^{-1} R_k^{\mathrm{T}}$, $C = C_k$.

The above problem (37) can be solved efficiently by the following successive overrelaxation (SOR) algorithm; see [27].

Algorithm 1. SOR for the QP problem (36) is as follows.

(1) Select the parameter $t_k \in (0, 2)$ and the initial value $\alpha_k^0 \in R^{m_k}$.

(2) Suppose that α_k^r is obtained by the r times iterate; compute α_k^{r+1} according to the following iterate formula:

$$\alpha_k^{r+1} = \left(\alpha_k^r - t_k D_k^{-1}\left(Q_k\alpha_k^r - e_{k2} + L_k\left(\alpha_k^{r+1} - \alpha_k^r\right)\right)\right)_\sharp, \tag{38}$$

where $Q = R_k\left(S_k^{\mathrm{T}} S_k + C_k^* I\right)^{-1} R_k^{\mathrm{T}}$. And define $L_k + D_k + L_k^{\mathrm{T}} = Q_k$, where $L_k \in R^{m_k \times m_k}$ and $D_k \in R^{m_k \times m_k}$ are the strictly lower triangular matrix and the diagonal matrix, respectively.

(3) Stop if $\|\alpha_k^{r+1} - \alpha_k^r\|$ is less than some desired tolerance. Else, replace α_k^r by α_k^{r+1} and r by $r+1$ and go to 2.

SOR is an excellent TWSVM solver, because it can process efficiently very large datasets that need not reside in memory. Furthermore, it has been proved that this algorithm converges linearly to a solution in [27, 28]. It should be pointed out that we employ the Sherman-Morrison-Woodbury formula [29] for the inversion of matrix $(S_k^{\mathrm{T}}S_k + C_k^* I)$ and, hence, need only to invert matrix with a lower order l_k, instead of the order l. Further, in practise, if the number of patterns in the kth classe is large, then the rectangular kernel technique [30, 31] can be applied to reduce the dimensionality of our nonlinear classifiers.

3.4. Several Others Approaches. In this section, we briefly give several extension versions based on our framework by selecting different loss function or replacing 2-norm.

First, if the square loss function is chosen, that is, $L(1, g(x)) = (1 - g(x))^2$, then we can get the following formulation from the framework (17):

$$\begin{aligned} \min_{w_k,b_k} \quad & \frac{1}{2}\left\|B_k w_k + e_{k2} b_k\right\|_2^2 + \frac{1}{2} C_k^* \left(\|w_k\|_2^2 + b_k^2\right) + C_k \xi_k^{\mathrm{T}} \xi_k, \\ \text{s.t.} \quad & \left(A_k w_k + e_{k1} b_k\right) + \xi_k = e_{k1}, \end{aligned} \tag{39}$$

where the matrix A_k is comprised of the patterns in the kth class and the matrix B_k is defined by (12), $C_k^* > 0$ and $C_k > 0$ are parameters, e_{k1} and e_{k2} are vectors of ones of appropriate dimensions, and $k = 1, 2, \ldots, K$. This is extension version of LS-TWSVM [14].

Second, if we replace 2-norm with 1-norm in the problem (39), then we can get the extension of 1-norm LS-TWSVM [17] as follows:

$$\begin{aligned} \min_{w_k,b_k} \quad & \left\|B_k w_k + e_{k2} b_k\right\|_1 + C_k^* \left(\|w_k\|_1 + |b_k|\right) + C_k \left\|\xi_k\right\|_1, \\ \text{s.t.} \quad & \left(A_k w_k + e_{k1} b_k\right) + \xi_k = e_{k1}, \end{aligned} \tag{40}$$

where the matrix A_k is comprised of the patterns in the kth class, the matrix B_k is defined by (12), $C_k^* > 0$ and $C_k > 0$ are parameters, e_{k1} and e_{k2} are vectors of ones of appropriate dimensions, and $k = 1, 2, \ldots, K$.

Third, if the loss function is a convex combination of linear and square loss, that is, $L(1, g(x)) = \delta(1 - g(x)) + (1 - \delta)\left(1 - g(x)\right)^2$, where $\delta \in (0, 1)$, then we can obtain extension version of NPPC [18] as follows:

$$\begin{aligned} \min_{w_k,b_k} \quad & \frac{1}{2}\left\|B_k w_k + e_{k2} b_k\right\|_2^2 + \frac{1}{2} C_k^* \left(\|w_k\|_2^2 + b_k^2\right) \\ & + C_k \left(\delta e_{k1}^{\mathrm{T}} \xi_k + (1 - \delta)\, \xi_k^{\mathrm{T}} \xi_k\right), \\ \text{s.t.} \quad & \left(A_k w_k + e_{k1} b_k\right) + \xi_k = e_{k1}, \end{aligned} \tag{41}$$

where the matrix A_k is comprised of the patterns in the kth class, the matrix B_k is defined by (12), $C_k^* > 0$ and $C_k > 0$ are parameters, e_{k1} and e_{k2} are vectors of ones of appropriate dimensions, and $k = 1, 2, \ldots, K$.

Forth, if the square hinge loss function is selected, that is, $L(1, g(x)) = \left(\max\left(0, 1 - g(x)\right)\right)^2$, then we can get the extension version of smooth TWSVM as follows:

$$\begin{aligned} \min_{w_k,b_k,\xi_k} \quad & \left\|B_k w_k + e_{k2} b_k\right\|_2^2 + \frac{1}{2} C_k^* \left(\|w_k\|_2^2 + b_k^2\right) + C_k \xi_k^{\mathrm{T}} \xi_k, \\ \text{s.t.} \quad & \left(A_k w_k + e_{k1} b_k\right) + \xi_k \geq e_{k1}, \qquad \xi_k \geq 0, \end{aligned} \tag{42}$$

where the matrix A_k is comprised of the patterns in the kth class, the matrix B_k is defined by (12), $C_k^* > 0$ and $C_k > 0$ are parameters, e_{k1} and e_{k2} are vectors of ones of appropriate dimensions, and $k = 1, 2, \ldots, K$.

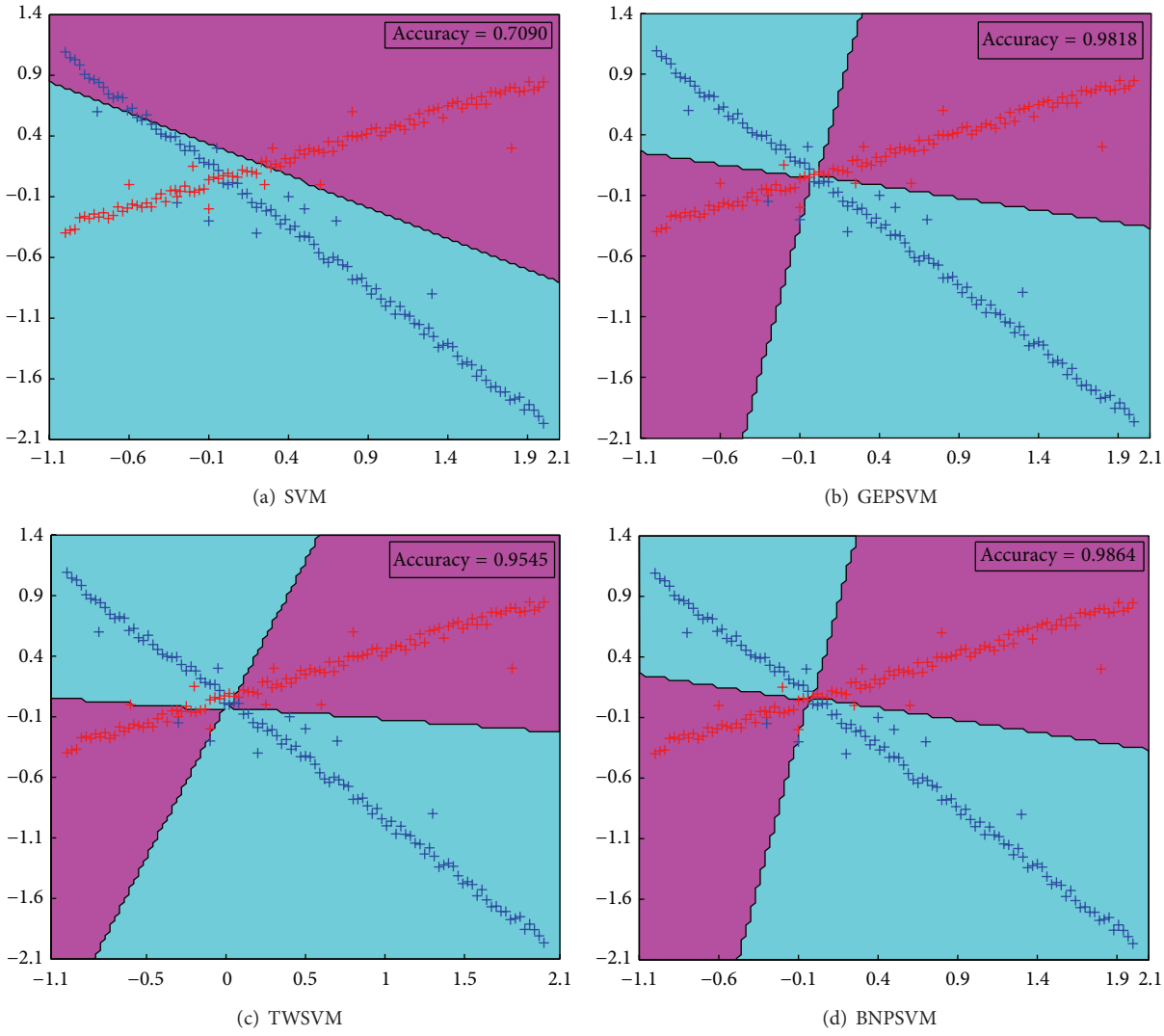
These approaches have the same decision function (18) and can be extended into nonlinear case. And their solving methods can construct based on their binary algorithms.

4. Numerical Experiments

In this section, we present experimental results of our binary HNPSVM (BHNPSVM) and multiclass HNPSVM (MHNPSVM) on both artificial and benchmark datasets. In experiments, we focus on the comparison between our methods and some state-of-the-art classification methods, including SVM, GEPSVM, TWSVM, "1-v-1," "1-v-r," and MBSVM. All the classification methods are implemented in MATLAB 7.0 [32] environment on a PC with Intel P4 processor (2.9 GHz) with 1 GB RAM. In order to give the fastest training speed, we employ Libsvm [33] to implement the SVM, "1-v-1," and "1-v-r". Our BHNPSVM and MHNPSVM and TWSVM and MBSVM are implemented using SOR technique; GEPSVM is implemented by simple MATLAB functions like "eig," respectively. As for the problem of selecting parameters, we employ standard 10-fold cross-validation technique [34]. Furthermore, the parameters for all methods are selected from the set $\{2^{-8}, \ldots, 2^8\}$.

4.1. Toy Examples. Firstly, we consider a simple two-dimensional "Cross Planes" dataset as Example 1, which was tested in [11, 13] to indicate that nonparallel hyperplanes classifiers can handle the cross planes dataset much better compared with parallel ones. Now, we show that our BHNPSVM also can handle cross-planes type data well due to use of our decision function. The "Cross Planes" dataset is generated by perturbing points lying on two intersecting lines. Figures 1(a)–1(d) show the dataset and the linear classifiers obtained by SVM, GEPSVM, TWSVM, and our BNPSVM. It is easy to see that the result of our BNPSVM is more reasonable than that of SVM, and better than that of GEPSVM and TWSVM. In addition, we list the accuracy and CPU time for these four classifiers in Table 1. From Table 1, we can see that our BNPSVM obtains the best accuracy while not the slowest computing time.

Secondly, we consider a two-dimensional three-class dataset as Example 2 to show the operating mechanism of our MNPSVM and other multiple-class classifiers. The three-class dataset is generated by perturbing points lying on three



(a) SVM (b) GEPSVM

(c) TWSVM (d) BNPSVM

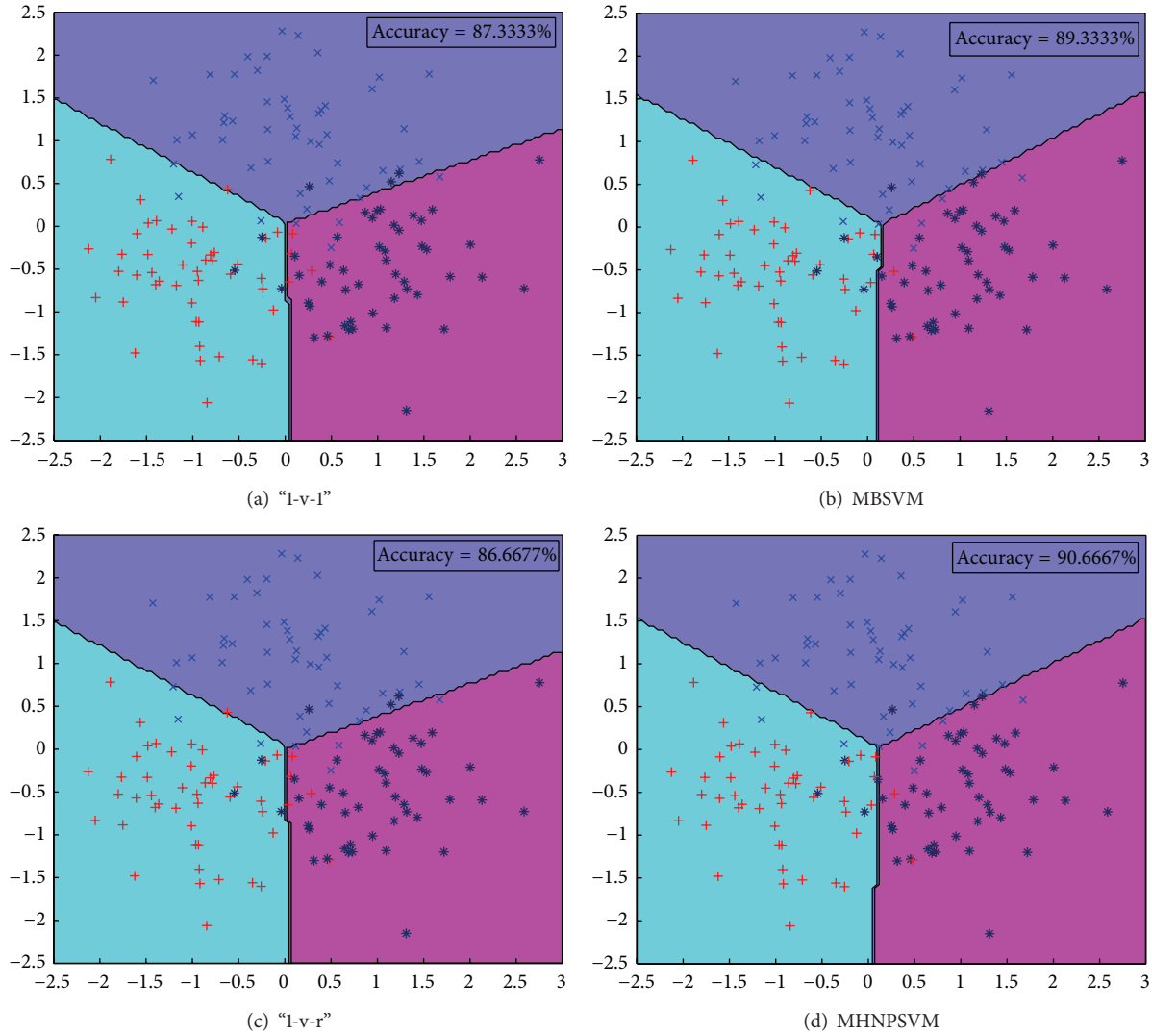
Figure 1: Results of linear SVM, GEPSVM, TWSVM, and BNPSVM on Example 1 dataset.

Table 1: Tenfold testing percentage test set accuracy (%) on example data sets.

Data set	SVM Accuracy % Time (s)	GEPSVM Accuracy % Time (s)	TWSVM Accuracy % Time (s)	BHNPSVM Accuracy % Time (s)
Example 1	70.90	95.45	98.18	**98.64**
(202 × 2)	0.122	0.0005	0.0064	0.0052
Data set	"1-v-1" Accuracy % Time (s)	"1-v-r" Accuracy % Time (s)	MBSVM Accuracy % Time (s)	MHNPSVM Accuracy % Time (s)
Example 2	87.33	86.67	89.33	**90.67**
(330 × 2)	0.098	0.0006	0.0079	0.0095

intersecting lines. Figures 2(a)–2(d) show the dataset and the linear classifiers obtained by "1-v-1," "1-v-r," MBSVM, and MHNPSVM. It is easy to see that the result of MBSVM and MHNPSVM is more reasonable than that of "1-v-1" and "1-v-r." We also list the accuracy and CPU time of Example 2 for these four classifiers in Table 1. From Table 1, we can see that our MHNPSVM obtains the best accuracy in all these two examples, indicating that our MHNPSVM is suitable for both "Cross Planes" and multiclass problems.

4.2. Benchmark Datasets. In order to further compare our methods with others, we examine nine binary-class datasets



(a) "1-v-1" (b) MBSVM

(c) "1-v-r" (d) MHNPSVM

Figure 2: Results of linear "1-v-1," "1-v-r," MBSVM, and MHNPSVM on Example 2 dataset.

and nine multiclass datasets used by [12, 35], from the UCI Repository of machine learning database [36]. Table 2 gives the details of these eighteen datasets.

In order to compare the behavior of our linear BHNPSVM with SVM, GEPSVM, and TWSVM, the numerical experimental results for binary-class UCI datasets are summarized in Table 3. In Table 3, the classification accuracy and computation time are listed. In Table 3, the best accuracy is shown by bold figures. It is easy to see that most of the accuracies of our linear BHNPSVM are better than linear SVM, GEPSVM, and TWSVM on these datasets. It can also be seen that our BHNPSVM is a little faster than TWSVM and is competitive with SVM (implements by Libsvm). We also list the mean accuracy and mean time for these four classifiers. Our BHNPSVM gains the the highest mean accuracy while faster training speed than TWSVM.

Table 4 is concerned with our kernel BHNPSVM, SVM, GEPSVM, and TWSVM on binary-class UCI datasets. The Gaussian kernel $K(x, x') = e^{-\mu\|x-x'\|^2}$ is used. The kernel parameter μ is also obtained through searching from the range from 2^{-8} to 2^{8}. The training CPU times for these four classifiers are also listed. The results in Table 4 are similar to those appearing in Table 3 and therefore confirm the above conclusion further.

In order to compare the behavior of our MHNPSVM with other multiple-class classifiers, we compare our MHNPSVM with "1-v-1," "1-v-r," and MBSVM, the linear results of numerical experiments on multiclass UCI datasets are summarized in Table 5. In Table 5, the classification accuracy and computation time are listed.

From Table 5, we can see that the accuracy of linear MHNPSVM is significantly better than linear MBSVM on all 9 UCI datasets. We also obtain that MHNPSVM and MBSVM are almost same fast because they both solve two SOR algorithms instead of two QP problems with the same size. In contrast, classification accuracy of "1-v-1" and "1-v-r" is no statistical difference with MHNPSVM for all cases except for vowel dataset, and "1-v-1" and "1-v-r" are a bit lower than

Table 2: The detailed characteristics of the datasets.

Data	#Ins	#Fea	#class	Data	#Ins	#Fea	#class
Hepatitis	155	19	2	Votes	435	16	2
WBPC	198	34	2	Sonar	208	60	2
Heart-statlog	270	13	2	BUPA	345	6	2
Pima-Indian	768	8	2	CMC	1473	9	2
Australian	690	14	2	Iris	150	3	4
Wine	178	3	13	Ecoli	336	8	8
Vowel	528	11	10	Glass	214	6	13
Vehicle	846	4	18	Car	1728	6	4
Segment	2310	7	19	Satimage	4435	6	36

#Ins is the number of the training points; #attributes is the number of attributes; #class is the number of class.

Table 3: Tenfold testing percentage test set accuracy (%) on binary-class UCI data sets for linear classifiers.

Data sets	TWSVM Accuracy % Time (s)	SVM Accuracy % Time (s)	GEPSVM Accuracy % Time (s)	BHNPSVM Accuracy % Time (s)
Hepatitis	82.89 ± 6.30*	84.13 ± 5.58	80.07 ± 5.43	**85.47 ± 1.36***
	0.012	0.012	0.0006	0.0304
BUPA liver	66.40 ± 7.74*	67.78 ± 5.51	61.33 ± 6.26	**69.97 ± 0.56***
	0.840	0.0549	0.0012	0.2143
Heart-statlog	**84.44 ± 6.80**	83.12 ± 5.41	75.37 ± 7.02	**84.44 ± 0.56**
	0.023	0.0281	0.0022	0.1092
Votes	95.85 ± 2.75	95.80 ± 2.65	91.93 ± 3.18	**95.58 ± 2.75**
	0.797	1.1446	0.0039	0.1027
WPBC	**83.68 ± 5.73***	83.30 ± 4.53	76.76 ± 6.67	81.32 ± 1.36*
	0.012	0.0432	0.0002	0.0465
Sonar	77.00 ± 6.10	**80.13 ± 5.43**	73.16 ± 8.33	74.15 ± 1.73
	0.007	0.0946	0.0225	0.007
Australian	85.94 ± 5.84	**88.51 ± 4.85**	80.00 ± 3.99	85.27 ± 3.26
	0.3460	0.2350	0.0029	0.4250
Pima-Indian	73.80 ± 4.97*	**77.34 ± 4.37**	75.47 ± 4.64	77.05 ± 0.48*
	0.121	0.261	0.0016	0.4793
CMC	68.28 ± 2.21*	67.82 ± 2.63	66.76 ± 2.98	**77.86 ± 0.22***
	1.247	0.597	0.0050	1.197
Mean accuracy	79.81	80.88	75.65	**81.23**
Mean time	0.38	0.27	0.004	0.29

* A greater difference between BHNPSVM and TWSVM.

MHNPSVM and MBSVM in average training time. Thus, with the proposed formulation of MHNPSVM allows the classifier to learn better by reducing the generalization errors. However, this improved performance is obtained at the cost of more tuning effort involved. This is because MHNPSVM requires tuning of more parameters than MBSVM.

Table 6 shows the nonlinear MHNPSVM with "1-v-1," "1-v-r," and MBSVM, the results of numerical experiments. In Table 6, the classification accuracy and computation time are listed. The results in Table 6 are similar to those appearing in Table 5; MHNPSVM has better classification accuracy than MBSVM in eight datasets, while MBSVM is better than MHNPSVM in one dataset, and MHNPSVM and MBSVM are much faster than "1-v-1" and "1-v-r", especially when the amount of data increases.

5. Conclusions

In this paper, a general framework of nonparallel hyperplanes support vector machines, termed NPSVMs, are proposed for binary classification and multiclass classification. For binary classification, this framework includes TWSVM and its many deformation versions, for instance, TWSVM, TBSVM, LS-TWSVM, NPPC, and so forth, when different loss functions and parameters are selected. For multiclass classification, we do not directly extend TWSVM and its deformation versions to get the framework, in which we switch the roles of the patterns of the kth class and the rest classes. This strategy does not lead to significant increase of the computation complexity when the number of classes is increasing. Moreover, in the decision function, "min" and Euclidean distance in TWSVM

Table 4: Tenfold testing percentage test set accuracy (%) on binary-class UCI datasets for nonlinear classifiers.

Datasets	TWSVM Accuracy % Time (s)	SVM Accuracy % Time (s)	GEPSVM Accuracy % Time (s)	BHNPSVM Accuracy % Time (s)
Hepatitis	83.39 ± 7.31 0.016	**84.13 ± 6.25** 0.0142	80.00 ± 5.2 0.0035	83.40 ± 3.58 0.0697
BUPA liver	67.83 ± 6.49* 0.033	68.32 ± 7.20 0.0129	63.01 ± 7.46 1.305	**74.24 ± 0.64*** 0.1522
Heart-statlog	82.96 ± 4.67* 0.029	83.33 ± 9.11 0.0250	**86.52 ± 7.36** 0.438	84.04 ± 4.56* 0.1120
Votes	94.91 ± 4.37 0.072	**95.64 ± 7.23** 0.0495	94.5 ± 3.37 0.087	95.21 ± 5.18 0.0152
WPBC	**81.28 ± 5.92** 0.029	80.18 ± 6.90 0.0148	80.07 ± 5.97 0.0043	80.89 ± 1.17 0.0468
Sonar	**89.64 ± 6.11** 0.014	88.93 ± 10.43 0.0781	81.93 ± 4.41 0.020	88.05 ± 1.79 0.2896
Australian	75.8 ± 4.91* 0.420	**85.51 ± 4.85** 0.0425	69.55 ± 5.37 0.334	77.58 ± 2.53* 0.497
Pima-Indian	73.74 ± 5.2* 0.427	76.09 ± 3.58 0.442	74.66 ± 5.00 15.892	**77.70 ± 0.39*** 0.381
CMC	73.95 ± 3.48* 1.708	68.98 ± 3.44 1.755	68.67 ± 3.84 1.042	**78.43 ± 0.13*** 1.920
Mean accuracy	80.39	81.23	77.66	**82.17**
Mean time	0.3053	0.27	2.1251	0.3871

* A greater difference between BHNPSVM and TWSVM.

Table 5: Tenfold testing percentage test set accuracy (%) on multiclass UCI datasets for linear classifiers.

Dataset	1-v-1 Accuracy (%) Time (s)	1-v-r Accuracy (%) Time (s)	MBSVM Accuracy (%) Time (s)	MHNPSVM Accuracy (%) Time (s)
Iris	96.83 ± 1.75 0.025	95.73 ± 3.78 0.014	95.00 ± 4.95 0.009	**96.96 ± 1.12** 0.010
Wine	96.59 ± 1.48 0.058	**97.72 ± 0.74** 0.021	94.77 ± 4.07 0.028	95.88 ± 2.21 0.023
Ecoli	**87.63 ± 0.81** 0.863	86.77 ± 0.87 0.522	85.72 ± 1.02 0.097	86.78 ± 0.75 0.089
Vowel	54.21 ± 2.24 1.459	57.44 ± 3.26 0.580	59.42 ± 4.96* 0.160	**64.60 ± 3.06*** 0.172
Glass	94.16 ± 1.84 1.037	94.42 ± 4.06 0.405	92.80 ± 9.80* 0.183	**95.83 ± 1.04*** 0.105
Vehicle	77.79 ± 2.21 28.11	**78.22 ± 2.10** 10.05	77.59 ± 2.16 2.96	77.13 ± 1.87 2.58
Car	86.78 ± 0.50 16.042	86.72 ± 0.31 13.79	84.09 ± 0.33* 5.92	**87.79 ± 0.91*** 6.05
Segment	91.60 ± 2.428 28.078	**92.54 ± 2.03** 15.26	92.68 ± 1.87 17.04	93.04 ± 2.01 17.55
Satimage	91.80 ± 0.81 60.50	90.20 ± 1.13 32.29	**92.40 ± 2.08** 47.45	91.40 ± 1.49 45.27
Mean accuracy	86.38	86.64	86.05	**87.71**
Mean time	15.13	8.10	8.21	7.98

* A greater difference between MHNPSVM and MBSVM.

Table 6: Tenfold testing percentage test set accuracy (%) on multiclass UCI datasets for nonlinear classifiers.

Dataset	1-v-1 Accuracy (%) Time (s)	1-v-r Accuracy (%) Time (s)	MBSVM Accuracy (%) Time (s)	MHNPSVM Accuracy (%) Time (s)
Iris	**98.93 ± 1.11** 0.0054	97.63 ± 5.46 0.0264	98.12 ± 2.08 0.037	98.74 ± 1.92 0.030
Wine	97.08 ± 3.32 7.294	**97.72 ± 0.86** 4.6504	96.45 ± 1.29 0.592	97.28 ± 0.96 0.523
Ecoli	92.27 ± 1.03 0.382	90.35 ± 0.47 0.0843	91.06 ± 1.45* 0.154	**92.95 ± 0.89*** 0.182
Glass	98.09 ± 1.04 0.692	99.14 ± 0.97 0.1085	98.76 ± 1.22 0.089	**99.24 ± 0.93** 0.092
Vowel	91.37 ± 0.86 1.482	**94.32 ± 0.18** 0.3844	80.42 ± 4.37* 0.623	85.86 ± 4.72* 0.593
Vehicle	81.03 ± 5.73 19.562	82.49 ± 4.26 11.456	82.01 ± 1.33 2.81	**83.57 ± 1.79** 2.50
Car	**88.37 ± 0.55** 3.6571	87.36 ± 0.68 0.9405	85.74 ± 0.33 1.832	86.57 ± 0.46 1.944
Segment	95.15 ± 6.02 128.42	94.65 ± 4.38 91.69	**95.96 ± 4.08** 53.27	95.90 ± 3.29 49.58
Satimage	93.80 ± 1.46 190.27	93.05 ± 1.46 132.47	94.03 ± 1.93 89.05	**94.47 ± 1.58** 88.36
Mean accuracy	92.90	**92.97**	91.39	92.73
Mean time	39.08	26.87	16.50	15.98

* A greater difference between MHNPSVM and MBSVM.

are replaced by "max" and the absolute value $|w^{\mathrm{T}}x+b|$, respectively. The absolute value $|w^{\mathrm{T}}x + b|$ is not only simpler but also more consistent with the primal problems. In particular, we discuss the linear and nonlinear case of the framework with the hinge loss function as example. Moreover, we also give the primal problems of several extensions of TWSVM's deformation versions. The numerical experiments on several artificial and benchmark datasets indicate that our NPSVMs yield comparable generalization performance compared with SVM, GEPSVM, TWSVM, MBSVM, "1-v-1," and "1-v-r". In short, the proposed framework not only includes TWSVM and its many deformation versions but also extends them into multiclass classification under keeping the merit of TWSVM (learning speed).

In the future, we will develop the idea of nonparallel hyperplanes classifiers to other problems such as ordinal regression, multi-instance, and multilabel classification.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

This work is supported by the National Natural Science Foundation of China (no. 11161045); the Project is sponsored by SRF for ROCS, SEM, and the Open Funding Project of the National Key Laboratory of Biochemical Engineering (no. 2013KF-01) and the Zhejiang Provincial Natural Science Foundation of China (no. Q12A010077).

References

[1] C. J. C. Burges, "A tutorial on support vector machines for pattern recognition," *Data Mining and Knowledge Discovery*, vol. 2, no. 2, pp. 121–167, 1998.

[2] C. Cortes and V. N. Vapnik, "Support-vector networks," *Machine Learning*, vol. 20, no. 3, pp. 273–297, 1995.

[3] C. Nello and S. T. John, *An Introduction to Support Vector Machines and Other Kernel-Based Learning Methods*, Cambridge University Press, Cambridge, Mass, USA, 1st edition, 2000.

[4] N. Deng, Y. Tian, and C. Zhang, *Support Vector Machines: Theory, Algorithms and Extensions*, Chapman & Hall/CRC Data Mining and Knowledge Discovery Series, CRC press, 2013.

[5] V. N. Vapnik, *The Nature of Statistical Learning Theory*, Springer, New York, NY, USA, 1995.

[6] E. Osuna, R. Freund, and F. Girosi, "Training support vector machines: an application to face detection," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 130–136, IEEE, San Juan, Puerto Rico, June 1997.

[7] H. Cevikalp, B. Triggs, and V. Franc, "Face and landmark detection by using cascade of classifiers," in *Proceedings of the 10th IEEE International Conference and Workshops on Automatic Face and Gesture Recognition (FG '13)*, pp. 1–7, Shanghai, China, April 2013.

[8] T. Joachims, "Text categorization with support vector machines: learning with many relevant features," in *Machine Learning: ECML-98*, pp. 137–142, 1998.

[9] L. J. Cao, "Support vector machines experts for time series forecasting," *Neurocomputing*, vol. 51, no. 1–4, pp. 321–339, 2003.

[10] Y.-C. Wang, Z.-X. Yang, and N.-Y. Deng, "Support vector machine prediction of enzyme function with conjoint triad feature and hierarchical context," *BMC Systems Biology*, vol. 5, no. 1, article S6, 2011.

[11] O. L. Mangasarian and E. W. Wild, "Multisurface proximal support vector machine classification via generalized eigenvalues," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 1, pp. 69–74, 2006.

[12] Jayadeva, R. Khemchandani, and S. Chandra, "Twin support vector machines for pattern classification," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 5, pp. 905–910, 2007.

[13] Y.-H. Shao, C.-H. Zhang, X.-B. Wang, and N.-Y. Deng, "Improvements on twin support vector machines," *IEEE Transactions on Neural Networks*, vol. 22, no. 6, pp. 962–968, 2011.

[14] M. Arun Kumar and M. Gopal, "Least squares twin support vector machines for pattern classification," *Expert Systems with Applications*, vol. 36, no. 4, pp. 7535–7543, 2009.

[15] M. Arun Kumar, R. Khemchandani, M. Gopal, and S. Chandra, "Knowledge based least squares twin support vector machines," *Information Sciences*, vol. 180, no. 23, pp. 4606–4618, 2010.

[16] Y.-H. Shao, N.-Y. Deng, and Z.-M. Yang, "Least squares recursive projection twin support vector machine for classification," *Pattern Recognition*, vol. 45, no. 6, pp. 2299–2307, 2012.

[17] S. Gao, Q. Ye, and N. Ye, "1-Norm least squares twin support vector machines," *Neurocomputing*, vol. 74, no. 17, pp. 3590–3597, 2011.

[18] S. Ghorai, A. Mukherjee, and P. K. Dutta, "Nonparallel plane proximal classifier," *Signal Processing*, vol. 89, no. 4, pp. 510–522, 2009.

[19] M. A. Kumar and M. Gopal, "Application of smoothing technique on twin support vector machines," *Pattern Recognition Letters*, vol. 29, no. 13, pp. 1842–1848, 2008.

[20] X. Peng, "A v-twin support vector machine (v-TSVM) classifier and its geometric algorithms," *Information Sciences*, vol. 180, no. 20, pp. 3863–3875, 2010.

[21] X. Peng, "TSVR: an efficient twin support vector machine for regression," *Neural Networks*, vol. 23, no. 3, pp. 365–372, 2010.

[22] Z. Wang, J. Chen, and M. Qin, "Non-parallel planes support vector machine for multi-class classification," in *Proceedings of the International Conference on Logistics Systems and Intelligent Management (ICLSIM '10)*, vol. 1, pp. 581–585, January 2010.

[23] H. Cong, C. Yang, and X. Pu, "Efficient speaker recognition based on multi-class Twin Support Vector Machines and GMMs," in *Proceedings of the IEEE International Conference on Robotics, Automation and Mechatronics (RAM '08)*, pp. 348–352, September 2008.

[24] Z.-X. Yang, Y.-H. Shao, and X.-S. Zhang, "Multiple birth support vector machine for multi-class classification," *Neural Computing and Applications*, vol. 22, no. 1, pp. 153–161, 2013.

[25] A. N. Tikhonov and V. Y. Arsenin, *Solutions of Ill-Posed Problems*, John Wiley & Sons, New York, NY, USA, 1977.

[26] O. L. Mangasarian, *Non Linear Programming*, SIAM, Philadelphia, Pa, USA, 1994.

[27] O. L. Mangasarian and D. R. Musicant, "Successive overrelaxation for support vector machines," *IEEE Transactions on Neural Networks*, vol. 10, no. 5, pp. 1032–1037, 1999.

[28] I. W.-H. Tsang, A. Kocsor, and J. T.-Y. Kwok, "Large-scale maximum margin discriminant analysis using core vector machines," *IEEE Transactions on Neural Networks*, vol. 19, no. 4, pp. 610–624, 2008.

[29] G. H. Golub and C. F. van Loan, *Matrix Computations*, The John Hopkins University Press, 1996.

[30] Y. J. Lee and O. L. Mangasarian, *Rsvm: Reduced Support Vector Machines*, 2001.

[31] Y.-J. Lee and S.-Y. Huang, "Reduced support vector machines: a statistical theory," *IEEE Transactions on Neural Networks*, vol. 18, no. 1, pp. 1–13, 2007.

[32] *MATLAB*, The MathWorks, 2007, http://www.mathworks.com.

[33] C. C. Chang and C. J. Lin, "LIBSVM: a library for support vector machines," 2001, http://www.csie.ntu.edu.tw/~cjlin/.

[34] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification*, John Wiley & Sons, New York, NY, USA, 2nd edition, 2001.

[35] C.-W. Hsu and C.-J. Lin, "A comparison of methods for multiclass support vector machines," *IEEE Transactions on Neural Networks*, vol. 13, no. 2, pp. 415–425, 2002.

[36] C. L. Blake and C. J. Merz, "UCI Repository for Machine Learning Databases," 1998, http://www.ics.uci.edu/~mlearn/MLRepository.html.