IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received March 19, 2020, accepted April 19, 2020, date of publication April 27, 2020, date of current version May 20, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2990611

Twin Hyper-Ellipsoidal Support Vector Machine for Binary Classification

ZEINAB EBRAHIMPOUR[1,2], WANGGEN WAN[1,2], (Senior Member, IEEE), ARASH SIOOFY KHOOJINE[3], AND LI HOU[4]

[1]School of Communication and Information Engineering, Shanghai University, Shanghai 200444, China
[2]Institute of Smart City, Shanghai University, Shanghai 200444, China
[3]School of Mathematical Sciences, Shanghai Jiao Tong University, Shanghai 200400, China
[4]School of Information Engineering, Huangshan University, Huangshan 245041, China

Corresponding author: Zeinab Ebrahimpour (z_ebrahimpour@shu.edu.cn)

This work was supported in part by the Science and Technology Commission of Shanghai Municipality under Grant 18510760300, in part by the Anhui Natural Science Foundation under Grant 1908085MF178, and in part by the Anhui Excellent Young Talents Support Program Project under Grant gxyqZD2019069.

ABSTRACT In this paper, a twin hyper-ellipsoidal support vector machine (TESVM) for binary classification of data is presented. Similar to twin support SVM(TWSVM) and twin hypersphere SVM (THSVM), as in the literature, our proposed method finds two hyper-ellipsoidals by solving two related SVM-type quadratic programming problem (QPPs), each of which is smaller than that of the classical SVM, causing it to achieve higher speed. The main idea of this paper is to employ Mahalanobis distance-based kernels for two classes of data in the THSVM algorithm to improve its generalization performance. Since the kernel used in SVM, TWSVM, and THSVM is based on Euclidean distance, it is assumed that the data points have been distributed in a hyper-spherical region, while the data points of two classes have been distributed in two different hyper-ellipsoidal regions. As mentioned in the literature, to work with hyper-ellipsoidal areas, Mahalanobis distance is a better choice than Euclidean distance. The effect of computational results of SVM, TWSVM, THSVM, and TESVM in terms of generalization performance and central processing unit (CPU) learning time on several benchmarks as well as synthetic and image datasets indicates, TESVM achieves fast learning speed along with higher generalization.

INDEX TERMS Hyper-ellipsoidal region, Mahalanobis distance, orientation information, support vector machine (SVM), twin hypersphere SVM (THSVM), twin Mahalanobis distance-based SVM (TMSVM), twin support vector machine (TWSVM).

I. INTRODUCTION

{S}upport vector machines (SVMs) has become a popular method in classification and regression data, introduced by Vapnik and colleagues [1], [2]. The use of kernel techniques in SVM is one of the reasons to improve and propose several extensions for this popular method for application in many fields [3]–[5].

Recently, many extensions of SVM have been presented to increase the generalization performance and improve the learning speed [6]–[8]. One of these methods, twin support vector machine (TWSVM), was proposed by Jayadeva [9] for binary classification. TWSVM is aimed at generating two nonparallel hyperplanes for two classes, each of which is as close to the points of the corresponding class as possible and at a distance of at least one from the other. Each hyperplane is derived by solving an SVM-type quadratic programming problem (QPP). TWSVM classifies new test points according to the distance from each of two nonparallel hyperplanes such that the points belong to the corresponding class of the nearest hyperplane. TWSVM has recently received more attention due to its lower computational complexity, mainly because each of its QPPs is only half as large as the full-sized QPP in the classical SVM. This is why many extensions of TWSVM have been proposed, such as recursive projection twin SVM [10], non-parallel plane proximal classifier [11], least squares twin SVM [12], least squares twin support vector hypersphere [13], sparse twin SVM [14], twin SVM [15], twin parametric margin SVM [16], twin support vector regression (SVR) [17], twin-parametric insensitive

The associate editor coordinating the review of this manuscript and approving it for publication was Geng-Ming Jiang.

SVR [18], etc. TWSVM, despite the faster learning speed and higher generalization performance than the classical SVM, cannot effectively depict the characteristics of two classes [19]. In particular, for the classification of two classes from different Gaussian distributions, we cannot use two non-parallel hyperplanes. Therefore, in [19], a twin-hypersphere support vector machine (THSVM) for binary classification of data was introduced. THSVM determines two hyperspheres, with each one covering as many data points in one class as possible while keeping far away from the opposite class. To find these hyperspheres, THSVM solves two smaller QPPs instead of one big QPP as in the classical SVM. THSVM benefits from higher learning speed than TWSVM, because it avoids the-inversion matrix. It uses two hyperspheres instead of two nonparallel hyperplanes to show the characteristics of two classes, which is more practical for databases produced by Gaussian distributions. So THSVM is also superior to TWSVM in terms of its generalization performance.

As mentioned earlier, one reason for the success of SVM and its extensions, such as TWSVM and THSVM, is the employment of the kernel technique. Since the kernels used in SVM, TWSVM, and THSVM are based on Euclidean distance, these methods assume that the data points have been distributed within a hyper-spherical region, while the data points of two classes are distributed within two different hyper-ellipsoidal regions. Therefore, two hyperspheres of THSVM cannot effectively extract the data information in two classes. By considering the structural nature of a hypersphere, THSVM assumes that the data points in one class have been raised in all directions with the same scale simultaneously. In other words, using Euclidean distance in two hyperspheres of THSVM leads to ignoring the covariance matrices of two classes of data.

Mahalanobis distance not only saves the correlations between data points but also it is scaled invariant, and therefore it is a better choice to work with hyper-ellipsoidal areas [20]. Mahalanobis distance is a more general case of Euclidean distance such that if any data point is considered as a unique region, then Mahalanobis distance degenerates to Euclidean distance.

In this paper, we propose a twin hyper-ellipsoidal support vector machine (TESVM) for the binary classification of data. First of all, a pair of Mahalanobis distance-based kernels is introduced. Then, according to these kernels, which are obtained by covariance matrices of two classes of data, TESVM generates two hyper-ellipsoidals via two smaller QPPs, in such a way that each hyper-ellipsoidal covers as many data points in one class as possible and keeps as far away from the other class as possible. Due to the use of Mahalanobis distance instead of Euclidean distance, TESVM is a more general case of THSVM. If the covariance matrices of the two classes reduce to identity matrices, TESVM performs similarly to THSVM.

The main idea of this paper is to use Mahalanobis distance-based kernels for two classes of data in the THSVM algorithm, which leads to improved generalization performance. TESVM, besides successfully inheriting the merits of THSVM and TWSVM, such as fast learning speed, effectively takes the advantages of covariance matrix information of two classes, into the prediction phase.

Computational comparisons of SVM, TWSVM, THSVM, and TESVM in terms of generalization performance and learning CPU time, on several benchmarks and synthetic and image datasets indicate that TESVM not only obtains fast learning speed but also demonstrates comparable generalization performance.

This paper is organized as follows: The next section represents a brief review of two related algorithms and presents the related formulas. In Section III our proposed TESVM algorithm is explained in detail, and its computational complexity and connections with other algorithms are introduced. The experimental results on the toy, benchmark, and image datasets are shown in Section IV, and the analysis experiments are introduced in this section. Finally, the last section concludes this paper.

II. BACKGROUND

Suppose the training points for two classes are as follows:

$$X^{(i)} = \left[x_1^{(i)}, x_2^{(i)}, \ldots, x_{N_i}^{(i)}\right], i = 1, 2 \tag{1}$$

where N_i is the samples with n dimension in class i, such as matrix with $N_1 \times n$ samples (A_i as ith sample) in class $+1$ and matrix B with $N_2 \times n$ samples in class -1, where $N_1 + N_2 = N$.

A. TWIN SUPPORT VECTOR MACHINE

As an extension of the classical SVM to improve its learning speed and generalization performance, Jayadeva [9] proposed a new twin support vector machine (TWSVM). There is a significant difference between SVM and TWSVM, in that TWSVM solves classification problems by generating two non-parallel hyperplanes, each of which passes through as many samples in one class as possible and gets far away, a distance of at least one from the other class. Each TWSVM hyperplane is constructed by solving a small half-sized QPP, compared with only one large-QPP in SVM. This leads to an approximately four times increased in learning speed.

For the linear case of TWSVM, it solves the following two QPPs:

$$\begin{aligned} &\min \frac{1}{2} \sum_{i \in \mathrm{I}^{(1)}} \left(w_{(1)}^T x_i + b_{(1)}\right)^2 + \frac{c_1}{l^{(2)}} \sum_{j \in \mathrm{I}^{(2)}} \xi_j \\ &\text{s.t.} - \left(w_{(1)}^T x_j + b_{(1)}\right) \geq 1 - \xi_j, \\ &\quad \xi_j \geq 0, \quad j \in \mathrm{I}^{(2)} \end{aligned} \tag{2}$$

$$\begin{aligned} &\min \frac{1}{2} \sum_{j \in \mathrm{I}^{(2)}} \left(w_{(2)}^T x_j + b_{(2)}\right)^2 + \frac{c_2}{l^{(1)}} \sum_{i \in \mathrm{I}^{(1)}} \xi_i \\ &\text{s.t.} \; w_{(2)}^T x_i + b_{(2)} \geq 1 - \xi_i, \\ &\quad \xi_i \geq 0, \quad i \in \mathrm{I}^{(1)} \end{aligned} \tag{3}$$

where $c_1 > 0$ and $c_2 > 0$ represent the pre-specified penalty factors, and ξ_i and ξ_j are the slack variables.

A test point X is assigned to a class according to its distance from the two nonparallel hyperplanes. More details and the nonlinear form of TWSVM can be found in [9].

B. TWIN-HYPERSPHERE SUPPORT VECTOR MACHINE

As mentioned in [9], the experimental results show that the TWSVM algorithm not only derives faster-learning speed but also obtains better generalization performance compared to the classical SVM. However, there remain some shortcomings in TWSVM. First, its optimization QPPs need to compute inversion of matrices with size $(m+1)\times(m+1)$ or $(l+1)\times(l+1)$ where m denotes the dimensions of training points and l the number of training points, which causes an-increase in central processing unit (CPU) learning time. Second, the algorithm cannot effectively depict the properties of data in two classes, since each hyperplane of TWSVM tries to be closer to as many samples in the corresponding class as possible and simultaneously get far away, to a distance of at least one from the other class. So, to overcome the problem of binary classification with two classes, coming from two different Gaussian distributions, we cannot use two nonparallel hyperplanes. To overcome the shortcomings, a new twin hypersphere support vector machine (THSVM) was proposed in [19]. THSVM aims at generating two hyperspheres, each of which covers as many samples in one class as possible and is as far away from the other class as possible. In the learning phase of the THSVM classifier the following pair of QPPs must be solved:

$$\begin{aligned}
&\min\ R_{(1)}^2 - \frac{v_1}{l^{(2)}}\sum_{j\in \mathrm{I}^{(2)}}\left\|\varphi\left(x_j\right)-c_{(1)}\right\|^2 + \frac{c_1}{l^{(1)}}\sum_{i\in \mathrm{I}^{(1)}}\xi_i \\
&\text{s.t.}\ \left\|\varphi\left(x_i\right)-c_{(1)}\right\|^2 \le R_{(1)}^2+\xi_i \\
&\quad R_{(1)}^2\ge 0,\quad \xi_i\ge 0,\ i\in \mathrm{I}^{(1)}
\end{aligned} \tag{4}$$

$$\begin{aligned}
&\min\ R_{(2)}^2 - \frac{v_2}{l^{(1)}}\sum_{i\in \mathrm{I}^{(1)}}\left\|\varphi\left(x_i\right)-c_{(2)}\right\|^2 + \frac{c_2}{l^{(2)}}\sum_{j\in \mathrm{I}^{(2)}}\xi_j \\
&\text{s.t.}\ \left\|\varphi\left(x_j\right)-c_{(2)}\right\|^2 \le R_{(2)}^2+\xi_j \\
&\quad R_{(2)}^2\ge 0,\quad \xi_j\ge 0,\ j\in \mathrm{I}^{(2)}
\end{aligned} \tag{5}$$

where $c_1 > 0$ and $c_2 > 0$ represent the pre-specified penalty factors, and ξ_i and ξ_j are the slack variables.

After optimizing Equation (4) and considering suitable parameters and, the center of the positive hypersphere can be achieved as follows:

$$c_{(1)} = \frac{1}{1-v_1}\left(\sum_{i\in \mathrm{I}^{(1)}}\alpha_i\varphi\left(x_i\right) - \frac{v_1}{l^{(2)}}\sum_{j\in \mathrm{I}^{(2)}}\varphi\left(x_j\right)\right) \tag{6}$$

and its squared radius $R_{(1)}^2$ value is obtained as:

$$R_{(1)}^2 = \frac{1}{\left|\mathrm{I}_R^{(1)}\right|}\sum_{i\in \mathrm{I}_R^{(1)}}\left\|\varphi\left(x_i\right)-c_{(1)}\right\|^2 \tag{7}$$

where the index set $\mathrm{I}_R^{(1)}$ is calculated as:

$$\mathrm{I}_R^{(1)} = \left\{i|0<\alpha_i<\frac{c_1}{l^{(1)}}, i\in \mathrm{I}^{(1)}\right\} \tag{8}$$

Similar equations for the negative hypersphere are obtained as in [19]. When the two hyperspheres are calculated:

$$\left\|\varphi(x)-c_{(1)}\right\|^2 \le R_{(1)}^2 \text{and } \left\|\varphi(x)-c_{(2)}\right\|^2 \le R_{(2)}^2 \tag{9}$$

a new test point belongs to the positive or negative class according to the following minimization:

$$f(x) = \underset{(1),(2)}{\arg\min}\left\{\frac{\left\|\varphi(x)-c_{(1)}\right\|^2}{R_{(1)}^2}, \frac{\left\|\varphi(x)-c_{(2)}\right\|^2}{R_{(2)}^2}\right\} \tag{10}$$

III. TWIN HYPER-ELLIPSOIDAL SUPPORT VECTOR MACHINE (TESVM)

As previously mentioned, a pair of hyperspheres can better depict the data characteristics of classes than two nonparallel hyperplanes, especially when the two classes are of different Gaussian distributions [19]. However, the THSVM is blind to the orientation information of samples in two classes. Note that for many real-world classification problems, orientation information in two classes is often different, which makes their covariance matrices different. In the spirit of the covariance matrices of two classes, we introduce Mahalanobis distance-based kernels to THSVM and propose a novel THSVM classifier, which call the twin hyper-ellipsoidal support vector machine (TESVM) classifier.

A. TESVM CLASSIFIE

The TESVM classifier uses two SVM-type QPPs in its optimization problems, similar to THSVM and TWSVM. However, compared with the Euclidean distance-based kernels used in the two QPPs of THSVM and TWSVM, our proposed method employs Mahalanobis distance instead. In brief, this model finds a pair of hyper-ellipsoidals such that each one not only covers as many samples in one class as possible, and keeps as far away from the other class as possible, but also captures the orientation information of the corresponding class samples. A new sample is assigned to a positive or negative class according to which hyper-ellipsoidal it lies closest to.

The aim is to fit one hyper-ellipsoidal for each class in the feature space with minimum effective radius $R(R>0)$ that covers a majority of samples and is far away from the other class.

Fact 1: Two TESVM hyper-ellipsoidals with Mahalanobis distance-based kernels in the feature space are obtained by solving the following pair of optimization problems:

$$\begin{aligned}
\min\ & R_+^2 - \frac{v_1}{l^-}\sum_{j\in \mathrm{I}^-}\left(\varphi\left(x_j\right)-c_+\right)\Sigma_+^{-1}\left(\varphi\left(x_j\right)-c_+\right) \\
&+\frac{c_1}{l^+}\sum_{i\in \mathrm{I}^+}\xi_i
\end{aligned}$$

$$\begin{aligned}
&\text{s.t. } (\varphi(x_i) - c_+)\,\Sigma_+^{-1}\,(\varphi(x_i) - c_+) \le R_+^2 + \xi_i \\
&\quad R_+^2 \ge 0, \quad \xi_i \ge 0,\ i \in \mathrm{I}^+ \qquad (11)\\
&\min\ R_-^2 - \frac{v_2}{l^+}\sum_{i\in \mathrm{I}^+}(\varphi(x_i) - c_-)\Sigma_-^{-1}(\varphi(x_i) - c_-) \\
&+ \frac{c_2}{l^-}\sum_{j\in \mathrm{I}^-}\xi_j \\
&\text{s.t. } \left(\varphi\left(x_j\right) - c_-\right)\Sigma_-^{-1}\left(\varphi\left(x_j\right) - c_-\right) \le R_-^2 + \xi_j \\
&\quad R_-^2 \ge 0, \quad \xi_j \ge 0,\ j \in \mathrm{I}^- \qquad (12)
\end{aligned}$$

where the penalty factors $c_1, c_2 > 0$ and $v_1, v_2 > 0$ are pre-specified by the user, and c_+ and R_+ denote as the center and radius of the corresponding hyper-ellipsoidal, respectively. (proof in Appendix).

We now discuss the optimization problems, Equations (11) and (12), more precisely. The first term in the objective function of these two Equations minimizes the square of the effective radius of the corresponding hyper-ellipsoidal. This leads to construction of as compact a hyper-ellipsoidal as possible. The second term in the objective functions maximizes the sum of squared Mahalanobis distances from the center of the corresponding hyper-ellipsoidal to the points of the opposite class, which causes the center of this hyper-ellipsoidal to be as far from the samples of the opposite class as possible, by taking into account the orientation information of the corresponding class, embedded in the covariance matrices and of Equations (11) and (12), respectively. The constraints make the corresponding hyper-ellipsoidal cover samples of the corresponding class with regard to the orientation information of its samples. On the other hand, this information can change the shape of the corresponding hyper-ellipsoidal to better cover its class samples and be farther from the opposite class samples.

Besides, the slack variables ξ_i, $i \in \mathrm{I}^+$ and ξ_j, $j \in \mathrm{I}^-$ for error measurement are added. As the last term in the objective functions of Equations (11) and (12), the sum of error variables is minimized, which leads to reduced misclassification due to the points belonging to the opposite class.

Now we optimize the optimization problems, Equations (11) and (12):

Fact 2: The dual QPP form of Equation (11) is as follows:

$$\begin{aligned}
&max\ (1 - \sum_{i\in \mathrm{I}^+}\alpha_i - s)R_+^2 + \sum_{i\in \mathrm{I}^+}\left(\frac{c_1}{l^+} - \alpha_i - r_i\right)\xi_i \\
&- \frac{v_1}{l^-}\sum_{j\in \mathrm{I}^-}\left(\left(\varphi\left(x_j\right) - c_+\right)\Sigma_+^{-1}\left(\varphi\left(x_j\right) - c_+\right)\right) \\
&+ \sum_{i\in \mathrm{I}^+}\alpha_i\left(\left(\varphi(x_i) - c_+\right)\Sigma_+^{-1}\left(\varphi(x_i) - c_+\right)\right) \\
&\text{s.t.} \sum_{i\in \mathrm{I}^+}\alpha_i = 1, \quad 0 \le \alpha_i \le \frac{c_1}{l^+},\ i \in \mathrm{I}^+ \qquad (13)
\end{aligned}$$

(proof in Appendix.)

Fact 3: The simpler dual form of Equation (12) becomes:

$$\begin{aligned}
&\max \sum_{j_1,j_2\in \mathrm{I}^-}\beta_{j_1}\beta_{j_2}\left(\left(\frac{1}{1-v_2}\right)^2\left(1 - \left(\frac{v_2}{l^+}\right)\right)K_-(x_{j_1}, x_{j_2})\right) \\
&+ \sum_{j\in \mathrm{I}^-}\beta_j\left(\left(\frac{v_2}{l^+}\right)\left(\frac{1}{1-v_2}\right)\sum_{i\in \mathrm{I}^+}K_-(x_i, x_j)\right. \\
&\left. + K_-(x_j, x_i) - \left(\frac{2}{1-v_2}\right)K_-(x_j, x_j)\right) \\
&\text{s.t.} \sum_{j\in \mathrm{I}^-}\beta_j = 1, 0 \le \beta_j \le \frac{c_2}{l^-}, j \in \mathrm{I}^- \qquad (14)
\end{aligned}$$

where $\beta_i, j \in \mathrm{I}^-$ are the Lagrangian multipliers, and the center c_- is computed as:

$$c_- = \frac{1}{1-v_2}\left(\sum_{j\in \mathrm{I}^-}\beta_j\varphi\left(x_j\right) - \frac{v_2}{l^+}\sum_{i\in \mathrm{I}^+}\varphi(x_i)\right) \qquad (15)$$

also, we have:

$$R_-^2 = \frac{1}{|\mathrm{I}_R^-|}\sum_{j\in \mathrm{I}_{R^-}}\left(\varphi\left(x_j\right) - c_-\right)\Sigma_-^{-1}\left(\varphi\left(x_j\right) - c_-\right) \qquad (16)$$

where $\mathrm{I}_R^- = \left\{j | 0 < \beta_j < \frac{c_2}{l^-}, j \in \mathrm{I}^-\right\}$.

(proof in Appendix.)

When the positive and negative centers c_+ and their squared radiuses R_+^2 are obtained, we can acquire the following two hyper-ellipsoidals for positive and negative classes, respectively:

$$\begin{aligned}
&(\varphi(x) - c_+)\,\Sigma_+^{-1}\,(\varphi(x) - c_+) \le R_+^2, \\
&(\varphi(x) - c_-)\,\Sigma_-^{-1}\,(\varphi(x) - c_-) \le R_-^2 \qquad (17)
\end{aligned}$$

Based on the distance of a test point x from the two hyper-ellipsoidals (Equation (12)), we can determine its class label, such that test point belongs to a class whose hyper-ellipsoidal is located closer to x, i.e:

$$\begin{aligned}
f(x) = \arg\min_{+,-}\left\{\frac{(\varphi(x) - c_+)\Sigma_+^{-1}(\varphi(x) - c_+)}{R_+^2},\right. \\
\left.\frac{(\varphi(x) - c_-)\Sigma_-^{-1}(\varphi(x) - c_-)}{R_-^2}\right\} \qquad (18)
\end{aligned}$$

B. COMPUTATIONAL COMPLEXIT

Our proposed method comprises two hyper-ellipsoidals to model two classes of data, each of which is constructed concerning the points of only one class.

Therefore, any QPPs of the TESVM are almost of the size of the classical SVM, making TESVM almost four times faster.

In contrast to TWSVM and similar to TMSVM, our proposed algorithm has higher computational complexity. The calculation of two Mahalanobis distance-based kernels

$K_+(.,.)$ and $K_-(.,.)$ is the main cost in TESVM, which needs to reverse

$$\left(\sigma \mathrm{I} + J_+^1 K_{v(1)} J_+\right)$$

and

$$\left(\sigma \mathrm{I} + J_-^1 K_{v(2)} J_-\right)$$

The time order of matrix inversion is $O(l^{2.3})$ [21]. However, by caching some necessary matrices, we can avoid the extra communication cost. For instance, caching

$$J_+ \left(\sigma \mathrm{I} + J_+^1 K_{v(1)} J_+\right)^{-1} J_+^1$$

and

$$J_- \left(\sigma \mathrm{I} + J_-^1 K_{v(2)} J_-\right)^{-1} J_-^1,$$

avoids the extra computation cost in the test section but consumes more RAM capacity.

In summary, the computational complexity of TESVM is higher than TWSVM and THSVM algorithms but, as shown in the experiments, the learning speed of TESVM is still faster than that of the classical SVM.

C. CONNECTION TO OTHER METHODS

In this section, we compare our corresponding TESVM to other related classification algorithms.

1) CONNECTION TO TWSVM

As we saw earlier, the TESVM algorithm finds a pair of hyper-ellipsoidals to classify data points, while the TWSVM algorithm does it with a pair of hyperplanes. Each of the TESVM's hyper-ellipsoidals concerning the trend of data distribution in different directions tries to cover as many samples in the corresponding class as possible and keeps away from the other class. For TWSVM, each hyperplane passes through as many samples in one class as possible and resides at a distance of at least one from the other class. As another difference, for TWSVM we have two matrix inversions with size $(l+1)^*(l+1)$ or $(m+1)^*(m+1)$ at a cost of $O(l^2)$, which is calculated in its optimization problems. TESVM needs to calculate $K_+(.,.)$ and $K_-(.,.)$ with inversion of $\left(\sigma \mathrm{I} + J_+^T K_{v(1)} J_+\right)$ and $\left(\sigma \mathrm{I} + J_-^T K_{v(2)} J_-\right)$ at a cost of $O(l^{2.3})$ [21], which makes the computational complexity larger for THSVM than TWSVM.

2) CONNECTION TO THSVM

Similar to THSVM and TWSVM, our proposed algorithm comprises a pair of QPPs. In the constraints of each QPP, only the samples of one class participate. In other words, each QPP is roughly half the size of all samples compared with the full-sized QPP in the classical SVM. THSVM aims to find two hyperspheres such that each one covers as many samples in one class as possible and keeps away from the other class. Note that a hypersphere is a set of points at a constant distance from the center of the hypersphere. Also, it implicitly assumes that the growth rate of points in different directions is the same. As seen in the THSVM models, they use this assumption, which does not usually occur in real-world problems.

On the other hand, TESVM does not consider this assumption and can effectively change its shape due to the different growth rates of points in different directions. Therefore, TESVM can capture the trends of points in one class in different directions while trying to effectively cover as many samples in the corresponding class as possible and be as far away from the other class. TESVM exploits this orientation information of samples in one class by employing Mahalanobis distance-based kernels in its optimization problems. Compared with THSVM, our TESVM considers that the samples of two classes are distributed in two different hyper-ellipsoidal regions. This makes TESVM surpass THSVM in terms of classification accuracy and generalization performance, as can be observed in the experiments.

However, the THSVM algorithm needs to solve two SVM-type optimization problems with no matrix inversion in the objective functions of its dual QPPs, while TESVM has to calculate matrix inversions at the cost of $O(l^{2.3})$, leading to higher computational complexity.

IV. EXPERIMENTS

To evaluate our proposed algorithm with THSVM, TWSVM, and SVM, we show the execution results on several benchmark datasets in terms of classification accuracy and CPU learning time. These experiments use MATLAB software on a system with a 2.26 GHZ CPU and 4 GB of RAM. For simplicity of all algorithms, we set $c_1 = c_2 = c$ and $v_1 = v_2 = v$; they are selected from the set of values $\{2^i | i = -9, -8, \cdots, 0, \cdots, 9, 10\}$ through the random data including 30% of the training data as a tuning set. The v value in THSVM is chosen from the set $\{0.1, 0.2, \ldots, 0.9\}$ Once the parameters are determined, the tuning set is returned to the training set to obtain the final classifier.

A. TOY DATASET

In this section, to show the performance of our TESVM, we compare it with the other algorithms on the two Gaussian problem dataset. Similar to [19], this dataset comprises 400 samples such that each positive and negative class has 200 samples, with a Gaussian distribution $\mathcal{N}\left((3,3)^T, diag\{4, 0.5\}\right)$ for the positive class (blue samples) and $\mathcal{N}\left((0,0)^T, diag\{0.5, 4\}\right)$ for the negative class (red samples). For testing, we produced 2000 samples of each positive and negative class with the same Gaussian distribution. As shown in Figure 1, the positive class (blue) has a roughly horizontal orientation and the negative class has a roughly vertical orientation, such that a structural conflict between the two classes appears.

The results of SVM, TWSVM, THSVM, and TESVM with the linear kernel are shown in Figure 1. As seen in this figure, linear SVM, THSVM, and TESVM obtain better classification accuracy than linear TWSVM. This is for

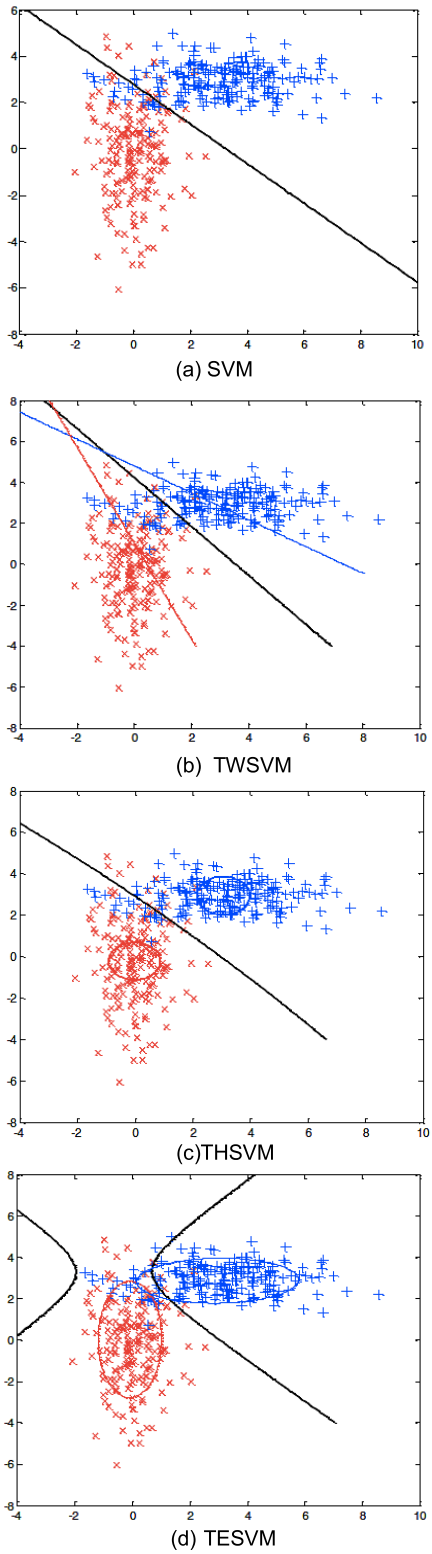

FIGURE 1. Classification results for the (a)SVM, (b)TWSVM, (c)THSVM, and (d)TESVM on the "Two-Gaussian" dataset with a linear kernel. The separating hyperplanes are shown as solid curves, and the nonparallel hyperplanes or hyperspheres are shown as dashed curves.

two non-parallel hyperplanes of TWSVM that could not effectively extract the characteristics of the samples in two classes. Therefore, the separating hyperplane of TWSVM cannot classify the training samples correctly. For SVM, since its attention is focused on the support vectors, the separating hyperplane resides almost in the middle of the support vectors of the two classes and can classify most of the training samples. On the other hand, one of the assumptions of SVM on data orientation or data shape is $\Sigma_+ = \Sigma_- = \Sigma$, which means two positive and negative classes have the same covariance matrices. This assumption makes SVM ignore the orientation information of samples in two classes and is not appropriate for this example. THSVM, uses two hyperspheres to model two classes, so the data scattering magnitude can take into account but cannot insert the orientation information of two classes into its models. Thus positive and negative classes would not be effectively covered. On the other hand, two hyper-ellipsoidals of our TESVM, due to the merits of the hyper-ellipsoidal shape, exploit the orientation information of two classes embedded in the corresponding covariance matrices. Therefore, they can better follow the data trend of two classes than the other two algorithms and cover the corresponding class samples more precisely. Thus TESVM's separating hyperplane can classify the samples more accurately than that of the other algorithms.

The results of the TESVM and other algorithms on the two Gaussian problem is shown in Table 1. As can be found, THSVM and TESVM algorithms achieve better classification accuracy than the other algorithms. Besides, TESVM surpasses THSVM and TWSVM in terms of classification accuracy. This indicates that TESVM can more effectively extract the characteristics of samples in two classes. In terms of the learning CPU time, although TESVM spends more time in the learning phase compared with THSVM and TWSVM, it is still faster than the traditional SVM on the artificial two Gaussian example.

TABLE 1. Performance of the SVM, TWSVM, THSVM, and TESVM on the "Two-Gaussian" and checkerboard datasets.

Dataset	Performance	SVM	TWSVM	THSVM	TESVM
Two-Gaussian	Accuracy	92.05	89.60	92. 13	92.98
	Time (s)	1.30	0.19	0.15	0.42
Checker-board	Accuracy	94.90	95.63	96.87	98.10
	Time (s)	30.21	8.32	7.64	8.63

For another experiment, we have the checkerboard example. This example contains 16 squares of uniformly distributed samples taken from two classes of data. Similar to \ cite{b19}, in the checkerboard experiment we independently produced 10 groups of datasets, each one containing 1000 training samples (500 for each class) and 1000 testing samples (500 for each class). The classification results of SVM, TWSVM, THSVM, and TESVM on the checkerboard dataset with the Gaussian kernel are shown in Figure 2. Observing the performance of the hyperplanes, hyperspheres, and hyper-ellipsoidals of TWSVM, THSVM and TESVM, in Figure 2, similar to [19], the hyperspheres of THSVM effectively covered the samples of the corresponding classes

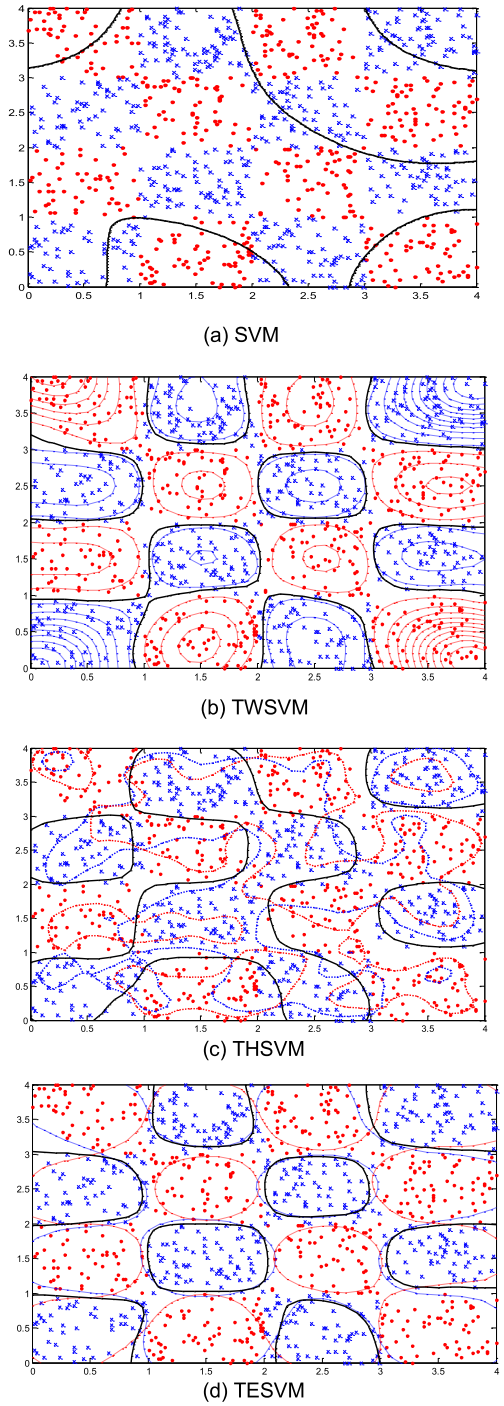

FIGURE 2. The result of SVM, TWSVM, THSVM, and TESVM on the checkerboard dataset with Gaussian kernel. The dashed curves denote for hyperellipsoids, hyperspheres, and nonparallel hyperplanes and for the separating hyperplanes we used solid curves.

and obtained a better result than TWSVM and SVM. This is because the embedded information in the samples of two classes could not be exploited in SVM and TWSVM. However, in THSVM, two hyperspheres can partially depict the characteristics of the two classes, since the orientation information still could not be derived in the models. Considering the covariance matrices, we can effectively introduce the orientation information into TESVM's optimization problems.

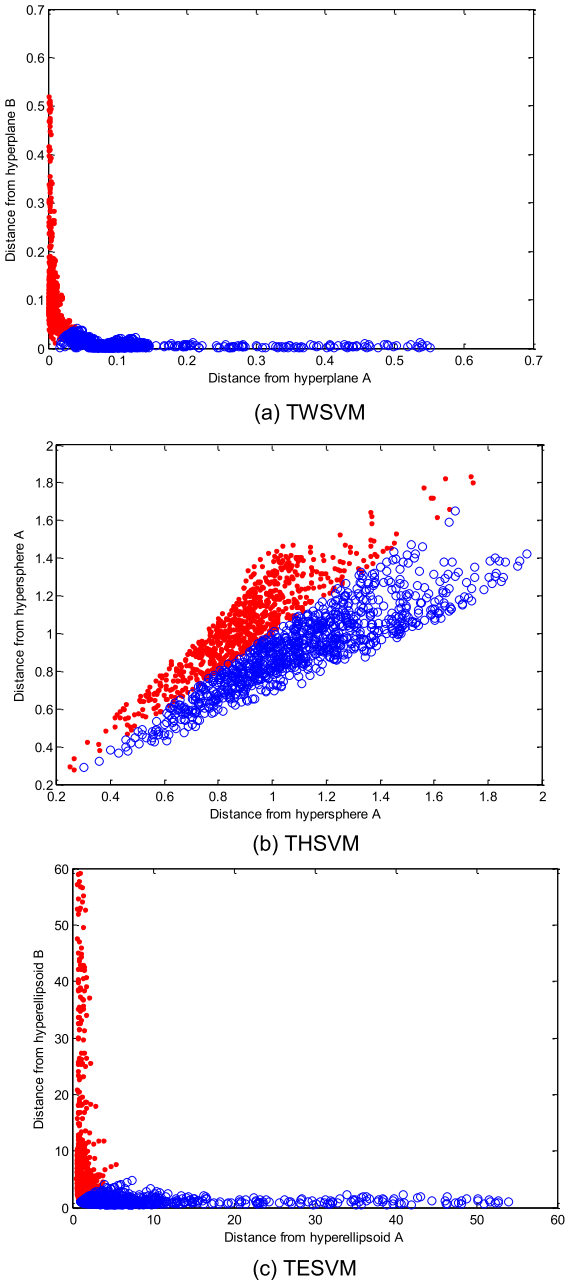

FIGURE 3. Two-dimensional projections for the checkerboard dataset by (a) TWSVM, (b) THSVM and (c) TESVM.

This makes it better cover the samples of each class, leading to a better separating hyperplane compared to the others.

The classification accuracy of SVM, TWSVM, THSVM, and TESVM on the checkerboard dataset is also shown in Table 1. As can be seen, TESVM's classification accuracy is much better than other algorithms and THSVM achieves higher classification accuracy. For the CPU learning time, TESVM needs more time compared with THSVM and TWSVM, but, it is faster than the classical SVM. To further illustrate the difference between TESVM and THSVM, we show two-dimensional scatter plots for the 1000 samples (500 for each class) of this checkerboard dataset obtained by THSVM and TESVM in Figure 3.

In Figure 3-a, the distances of each test point x_i from two hyperspheres of THSVM are calculated by using $d^i_{TH^+} = \frac{\|\varphi(x_i)-c_+\|}{R_+}$ for positive hypersphere and $d^i_{TH^-} = \frac{\|\varphi(x_i)-c_-\|}{R_-}$ for negative hypersphere. In Figure 3-b, the distances of x_i from two hyper-ellipsoidals of TESVM are obtained by using $d^i_{TE^+} = \frac{(\varphi(x_i)-c_+)\Sigma_+^{-1}(\varphi(x_i)-c_+)}{R_+}$ and $d^i_{TE^-} = \frac{(\varphi(x_i)-c_-)\Sigma_-^{-1}(\varphi(x_i)-c_-)}{R_-}$ and for positive and negative hyper-ellipsoidals, respectively. If $d^i_{TH^+} \leq d^i_{TH^-}$ or $d^i_{TE^+} \leq d^i_{TE^-}$ test point x_i belongs to the positive class and if $d^i_{TH^+} > d^i_{TH^-}$ or $d^i_{TE^+} > d^i_{TE^-}$, it belongs to the negative class.

The two-dimensional projection plots represent the ability of each algorithm to separate between two classes. The dots marked in red denote negative class and those in blue denote positive class. As can be seen in Figure 3-a, each hyperplane of TWSVM tries to pass through as many points in one class as possible. So the distance between the points of each class from the corresponding hyperplane will be near zero, which causes an over-fitting problem. On the other hand, the distance between this hyperplane and the points of the opposite class also approaches zero, which reduces the generalization ability.

In THSVM, as observed in Figure 3-b, the points of each class are almost covered by a hypersphere such that their distances to the corresponding hypersphere are almost less than one. This indicates that THSVM often succeeds in extracting the structure of each class and has less over-fitting than TWSVM. However, as derived from Figure 3-b, the distance of points in one class to the opposite hypersphere is still small, which makes THSVM's generalization ability remain low.

On the other hand, each hyper-ellipsoidal of the TESVM algorithm, not only tries to cover as many points in one class as possible and stay far away from the points of the opposite class, but also can effectively extract the characteristics and trends of data points in one class. As seen in Figure 3-c, the distances of data points in one class from the corresponding hyper-ellipsoidal become approximately less than one. Therefore, TESVM, similar to THSVM, can successfully exploit the structure of each class, leading to decreased over-fitting. On the other hand, the distance of data points in one class from the opposite hyper-ellipsoidal greatly increases improving the separability of the algorithm and achieving higher generalization performance compared with THSVM. Thus the TESVM algorithm can not only better extract the characteristics of each class, more effectively covering it, but also better discriminate between the two classes compared with THSVM.

B. BENCHMARK DATASET

We compared TESVM with the other related algorithms in terms of classification accuracy and CPU learning time on the common benchmark classification databases. A description of each database is shown in Table 2.

For a graphical comparison of TESVM with other algorithms, the banana dataset, which is two-dimensional, was used. As a result of the four algorithms on the banana dataset, as shown in Figure 4, the separating hyperplane of TESVM is better than that of the others. This indicates that TESVM has better performance than the other algorithms. This is because each hypersphere and hyper-ellipsoidal in THSVM and TESVM try to cover as many samples in one class as possible, and be as far away from the other class as possible, while the hyperplanes in TWSVM, try to pass through as many samples in one class as possible and keep away from the other class.

As shown in Figure 4, THSVM and TESVM can better extract the structure of two classes than TWSVM. On the other hand, for a class, since THSVM assumes that the growth rate of its samples in different directions is the same, it uses a hypersphere to cover its data points. The two hyper-ellipsoidals of TESVM also inherit the characteristics of the THSVM's hyperspheres and try to cover all samples of the corresponding classes.

However, each hyper-ellipsoidal considers the different growth of samples in one class in different directions by employing covariance matrices. So TESVM can better model each class and the separating hyperplane of TESVM is more accurate than that of THSVM.

The results of 10 independent executions of the TESVM, THSVM, TWSVM, and SVM nonlinear algorithms on the benchmark datasets and the comparison in terms of classification accuracy and CPU learning time are given in Table 3. According to the results in this table, it is observed that the accuracy of TESVM is better than the others, consequently, it has higher generalization performance. In addition to the classification accuracy in this table, the CPU learning times are also given for each algorithm on every dataset. As a reminder, the relevant matrices during the learning process of these algorithms are stored to reduce the CPU learning time, and no time for calculating duplicate matrices is needed.

TABLE 2. The UCI dataset information.

Dataset	Number of training points	Number of testing points	Dimension
Banana	400	4900	2
Breast Cancer	200	77	9
Diabetes	468	300	8
Flare solar	666	400	9
German	700	300	20
Heart	170	100	13
Image	1300	1010	18
Splice	1000	2175	60
Thyroid	140	75	5
Titanic	150	2051	3
Ionosphere	200	152	34
Planning relax	91	91	12
Fertility	50	50	9
Sonar	146	62	60
Liver disorder	172	173	6
SPECTF	80	187	44
Haberman	153	153	3
Image	500	1000	18
Ringnorm	400	3000	20

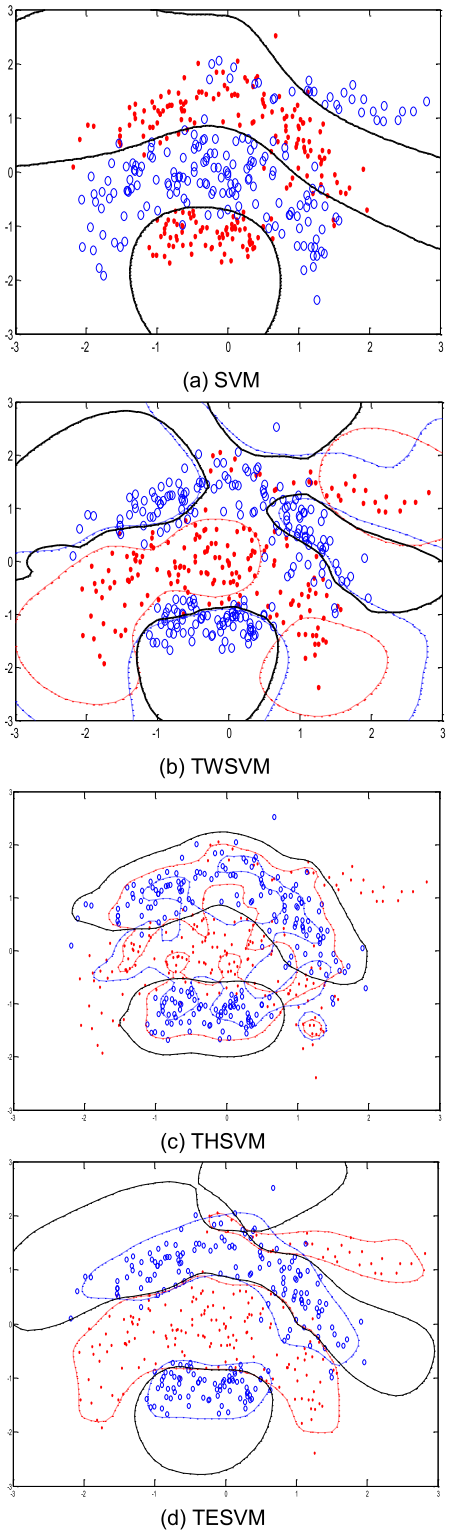

FIGURE 4. The results of execution SVM, TWSVM, THSVM, and TESVM on the Banana dataset. The dashed curves denote for nonparallel hyperplanes, hyperspheres, and hyper-ellipsoids and the separating hyperplane is shown using solid curves.

As can be seen, the TESVM algorithm needs more time in its learning process than THSVM and TWSVM. However, we need to calculate the covariance matrices of two classes to extract the orientation information of their samples. This information indicates the growth rate of samples in different directions. However, the learning speed of our proposed algorithm is still faster than the classical SVM.

TABLE 3. The results of the SVM, TWSVM, THSVM and TESVM on the UCI benchmark datasets.

Dataset	SVM Acc. Time(s)	TWSVM Acc. Time(s)	THSVM Acc. Time(s)	TESVM Acc. Time (s)
Banana	88.13±0.153 1.3301	88.07±0.014 0.3056	88.56±0.054 0.2025	88.98±0.177 1.2307
Breast Cancer	65.71±0.046 2.5316	70.91±0.055 0.8566	72.47±0.053 0.5577	73.77±0.065 1.7157
Diabetes	75.47±0.028 1.3456	75.53±0.017 0.2517	75.78±0.023 0.0822	76.07±0.064 0.5436
German	71.86±2.563 12.9509	72.13±1.040 10.2972	74.33±2.163 0.4462	74.35±2.041 12.023
Heart	77.92±2.025 0.480	77.20±1.106 0.183	78.20±1.329 0.092	79.40±1.371 0.4233
Image	92.60±0.018 1.5478	89.12±0.015 0.8443	90.00±0.044 0.7245	91.92±0.008 1.2015
Ringnorm	84.91±0.380 0.4007	95.34±0.453 0.1437	98.49±0.128 0.1424	97.95±0.117 0.3412
Splice	85.67±0.008 1.61	86.71±0.013 0.52	87.29±0.051 0.43	88.81±0.019 0.64
Thyroid	96.10±0.023 1.04	95.20±0.033 0.56	96.27±0.018 0.04	96.53±0.015 0.85
Titanic	77.03±0.023 2.563	74.16±0.210 0.245	77.13±0.050 0.175	77.33±0.083 0.4181
Ionosphere	85.03±0.022 0.453	92.45±0.012 0.202	92.58±0.076 0.119	94.04±0.070 0.322
Planning relax	65.94±0.037 0. 6350	65.64±0.051 0.3347	65.93±0.089 0.3254	67.03±0.031 0.5245
Fertility	78.00±1.450 0.1921	87.00±2.320 0. 1719	87.00±2.430 0.1036	90.00±2.010 0. 1825
Sonar	68.97±0.074 1.1457	71.34±0.145 0.6723	76.77±0.069 0.6548	83.55±0.071 0.7134
Liver disorder	64.97±0.034 1.143	65.20±0.019 0.8713	67.28±0.026 0.8034	68.09±0.038 0.9319
SPECTF	87.70±0.021 0.3542	90.91±0.023 0.0546	90.91±0.014 0.0162	92.19±0.002 0.1704
Haberman	66.14±0.037 1.2257	71.63±0.040 0.6332	71.76±0.048 0.4599	73.59±0.032 0.8088

C. IMAGE RECOGNITION

In this section, for further comparison of the algorithms, we apply them to typical image recognition problems such as object recognition (COIL-20) [23] and handwriting recognition (USPS) [24]. COIL-20 contains 20 objects. Images of each object were taken 5° apart as the object was rotated on a turntable, and there are 72 images of each object. The size

of each image is 32 × 32 pixels, with 256 gray levels per pixel. Thus, each image is represented by a 1024-dimensional vector.

The USPS database involves gray-scale handwritten images of the digits 0 to 9 such that there are 1100 images for each digit with a size of pixels in 256 gray levels. For this dataset, five-pairwise digits for odd vs. even digit classification were selected. We randomly partitioned these images into two groups with the same sizes and repeated this process 10 times. One group was used for training and the other for testing. Also, we used Gaussian kernel in the learning phase of all algorithms. The experimental results of algorithms over COIL-20 and USPS datasets are listed in Table 4.

TABLE 4. The results of the classification accuracy of SVM, TWSVM, THSVM and TESVM on the COIL-20 and USPS datasets.

Datasets	SVM	TWSVM	THSVM	TESVM
COIL-20	99.16±0.346	99.25±0.32	99.31±0.21	99.92±0.241
USPS 1 vs. 7	98.12±0.34	97.92±0.42	98.18±0.31	98.45±0.31
2 vs. 3	99.32±0.42	99.36±0.32	99.76±0.34	99.82±0.34
2 vs. 7	97.99±0.62	97.42±0.58	97.45±0.58	98.00±0.42
3 vs. 8	98.51±0.53	98.64±0.53	99.24±0.49	99.27±0.51
4 vs. 7	98.77±0.81	98.82±0.63	98.87±0.77	98.91±0.75

As shown in Table 4, for almost all cases, our TESVM algorithm achieved better classification accuracy than the others.

D. ANALYSIS EXPERIMENTS

1) AUC TEST

To evaluate the performance of classifiers in this paper, we used the area under the receiver operating curve (AUC) as a criterion. AUC is defined as follows:

$$AUC = \int_0^1 \frac{TP}{(TP+FN)} d\frac{FP}{(FP+TN)} = \int_0^1 \frac{TP}{P} d\frac{FP}{N} \quad (19)$$

where TP stands for true positive, FN for false negative, for false positive and TN for true negative. If the output of an algorithms does not surpass the random manner, the AUC value becomes 0.5, and for refer to the best algorithm outputs the AUC value becomes 1. Similar to [25], all AUC values are medians of five twofold cross-validations.

According to [26], in this paper, we performed five twofold cross-validations ($5 \times 2\,f\ cv$) as a simple method for model selection. In this method, initially, all data are divided into two equal parts, one is used for learning the algorithm and the other to evaluate it, and the process is repeated five times. As can be seen in Table 5, the results of median AUC are shown as $5 \times 2\,f\ cv$. Since, all median AUC values are bigger than 0.5, classifier predictions are better than the random manner.

As can be observed, TESVM is better than the others in terms of AUC and THSVM has a higher value than TWSVM and SVM. As is clear from Table 6, since the interquartile range (IQR) of TESVM is smaller than the others, its variations are lower and it has better stability than the other algorithms.

TABLE 5. Median AUC of the folds for algorithms.

Algorithms	SVM	TWSVM	THSVM	TESVM
Median AUC	0.7209	0.8051	0.8060	0.8126

TABLE 6. The interquartile range of the ten folds for AUC.

Algorithms	SVM	TWSVM	THSVM	TESVM
IQR	0.1216	0.0385	0.0412	0. 0325

For better statistical comparison between algorithms, as proposed in [26], [27], we used the Friedman test with corresponding post hoc tests. For this study, we used the average (mean) ranks of four algorithms on AUC, shown in Table 7.

TABLE 7. Mean ranking of the folds for AUC.

Algorithms	SVM	TWSVM	THSVM	TESVM
Mean Ranking	3.9	2.65	2.2	1.5

Regardless of the null hypothesis that the algorithms are all the same, the Friedman test formula is calculated as follows [10]:

$$\chi_F^2 = \frac{12N}{k(k+1)} \left[\sum_j R_j^2 - \frac{k(k+1)^2}{4} \right] \quad (20)$$

where N represents the fold number, K is the number of algorithms, R_j indicates the jth algorithm (of K algorithms) in the ith fold (from N folds), and $R_j = \frac{1}{N} \sum_i r_i^j$, where r_i^j represents AUC values for the ith fold and jth algorithm. By using χ_F^2, a better statistic can be achieved. We can calculate the F statistic as follows:

$$F_F = \frac{(N-1)\chi_F^2}{N(k-1) - \chi_F^2} \quad (21)$$

The F statistic, or f value, is a random variable with f distribution and $k-1$ and $(k-1)(N-1)$ degrees of freedom. According to Equations (59) and (60), we have $\chi_F^2 = 25.9350$ and $F_F = 57.4207$, where F_F has F distribution with (3,27) degrees of freedom. For the degree of importance $\alpha = 0.05$, the critical value $F(3, 27)$ is 2.96, for $\alpha = 0.025$ it is 3.64 and for $\alpha = 0.01$ it becomes 4.60. Since the value of F_F is much larger than the critical value, there are significant differences between the four algorithms. Recall that according to Table 7, the average rank of the TESVM algorithm is much lower than other algorithms. This indicates that our TESVM algorithm is more valid than the other three algorithms.

V. CONCLUSIONS

In this paper, an improved THSVM algorithm for binary classification of data is presented. In our algorithm, Mahalanobis distance-based kernels are made by the covariance matrices of two classes of data and then TESVM finds two hyper-ellipsoidals by these obtained kernels, such that each one covers as many data points in one class as possible and stays as far away from the other class as possible. This improvement allows the TESVM to take advantage of the orientation information of the two classes embedded in their covariance matrices. Note that for many real-world problems, two classes often have different covariance matrices.

The experiments on benchmark, synthetic and image datasets in Section IV indicate that TESVM also has better generalization performance compared with the other algorithms and faster learning speed than the classical SVM. Finally, increasing the learning speed of TESVM can be investigated in the future works.

Further research could look into comparing the developed model with classification methods such as Bayesian networks and random forests. However, another improved model could be developed for high-dimensional datasets where the number of variables is more than the number of observations.

CONFLICT OF INTEREST

The authors declare no conflict of interest.

APPENDIX

PROOF OF FACT 1

First, we have to calculate the covariance matrices, so according to [24] we can use the following equations:

$$\Sigma_+ = \varphi\left(X^{(1)}\right) J_+ J_+^T \varphi\left(X^{(1)}\right)^T,$$
$$\Sigma_- = \varphi\left(X^{(2)}\right) J_- J_-^T \varphi\left(X^{(2)}\right)^T \tag{22}$$

where $J_+J_+^T = \frac{1}{l^{(1)}}\left(\mathrm{I} - \frac{1}{l^{(1)}} ee^T\right)$, $J_-J_-^T = \frac{1}{l^{(2)}}\left(\mathrm{I} - \frac{1}{l^{(2)}} ee^T\right)$, $l^{(1)}$ and $l^{(2)}$ denote the number of samples in positive and negative classes, respectively. I and e are a vector and a matrix, respectively, with appropriate dimensions. In addition, $\varphi\left(X^{(\mathrm{I})}\right) = \left[\varphi\left(X_1^{(1)}\right) \ldots \varphi\left(X_{N_1}^{(1)}\right)\right], X_k^{(1)} \in \mathrm{I}^{(1)}, k = 1, \ldots, l^{(1)}$ and similarly, $\varphi\left(X^{(2)}\right) = \left[\varphi\left(X_1^{(2)}\right) \ldots \varphi\left(X_{N_2}^{(2)}\right)\right]$, $X_k^{(2)} \in \mathrm{I}^{(2)}, k = 1, \ldots, l^{(2)}$.

To calculate the Mahalanobis distance-based inner product $< \varphi(\cdot), \varphi(\cdot) >$ (i.e., kernels) in the feature space, we have to calculate the inverses of Since $\sum_{\pm}$ are often ill-conditioned, to improve the robustness, we add a small positive value σ to the diagonal elements of $\sum_{\pm}$. Thus, instead of $\sum_{\pm}^{-} 1$ we have to compute $(\sigma \mathrm{I} + \sum_{\pm})^{-1}$.

By using the Woodbury matrix identity [27]:

$$\left(U + VV^T\right)^{-1} = U^{-1} - U^T V\left(\mathrm{I} + V^T U^{-1} V\right)^{-1} V^T U^{-1}$$

we set $U = \sigma \mathrm{I}$ and $V = \varphi\left(X^{(1)}\right) J_+$, so we have:

$$\begin{aligned}
&(\sigma \mathrm{I} + \textstyle\sum_+)^{-1} \\
&= \left[\sigma \mathrm{I} + \varphi\left(X^{(1)}\right) J_+ J_+^T \varphi\left(X^{(1)}\right)^T\right]^{-1} \\
&= \sigma^{-1}\mathrm{I} - \sigma^{-1}\varphi\left(X^{(1)}\right) J_+ \left(\sigma \mathrm{I} + J_+^T K_{X^{(1)}} J_+\right)^{-1} J_+^T \varphi\left(X^{(1)}\right)^T
\end{aligned} \tag{23}$$

where $K_{X^{(1)}} = \varphi\left(X^{(1)}\right)^T \varphi\left(X^{(1)}\right) = k\left(X^{(1)}, X^{(1)}\right)$ ..

Similarly, we obtain:

$$\begin{aligned}
&(\sigma \mathrm{I} + \Sigma_-)^{-1} \\
&= \left[\sigma \mathrm{I} + \varphi\left(X^{(2)}\right) J_- J_-^T \varphi\left(X^{(2)}\right)^T\right]^{-1} \\
&= \sigma^{-1}\mathrm{I} - \sigma^{-1}\varphi\left(X^{(2)}\right) J_- \left(\sigma \mathrm{I} + J_-^T K_{X^{(2)}} J_-\right)^{-1} J_-^T \varphi\left(X^{(2)}\right)^T
\end{aligned} \tag{24}$$

where $K_{X^{(2)}} = \varphi\left(X^{(2)}\right)^T \varphi\left(X^{(2)}\right) = k\left(X^{(2)}, X^{(2)}\right)$.

So the Mahalanobis distance-based kernels $K_+(x_i, x_j)$ and $K_-(x_i, x_j)$ are obtained as follows:

$$\begin{aligned}
&K_+\left(x_i, x_j\right) \\
&= \left\langle \varphi\left(x_i\right) \cdot \varphi\left(x_j\right)\right\rangle_+ \\
&= \varphi\left(x_i\right) \Sigma_+^{-1} \varphi\left(x_j\right) \approx \varphi\left(x_i\right)^T \left(\sigma \mathrm{I} + \Sigma_+\right)^{-1} \varphi\left(x_j\right) \\
&= \varphi\left(x_i\right)^T \left[\sigma^{-1}\mathrm{I} - \sigma^{-1}\varphi\left(X^{(1)}\right) J_+ \left(\sigma \mathrm{I} + J_+^T K_{X^{(1)}} J_+\right)^{-1}\right. \\
&\quad \left. \times J_+^T \varphi\left(X^{(1)}\right)^T\right] \varphi\left(x_j\right) \\
&= \sigma^{-1} K\left(x_i, x_j\right) - \sigma^{-1} K\left(x_i, X^{(1)}\right) J_+ \left(\sigma \mathrm{I} + J_+^T K_{x^{(1)}} J_+\right)^{-1} \\
&\quad \times J_+^T K\left(X^{(1)}, x_j\right)
\end{aligned} \tag{25}$$

$$\begin{aligned}
&K_-\left(x_i, x_j\right) \\
&= \left\langle \varphi\left(x_i\right), \varphi\left(x_j\right)\right\rangle_- = \varphi\left(x_i\right) \Sigma_-^{-1} \varphi\left(x_j\right) \\
&\approx \varphi\left(x_i\right)^T \left(\sigma \mathrm{I} + \Sigma_-\right)^{-1} \varphi\left(x_j\right) \\
&= \varphi\left(x_i\right)^T \left[\sigma^{-1}\mathrm{I} - \sigma^{-1}\varphi\left(X^{(2)}\right) J_- \left(\sigma \mathrm{I} + J_-^T K_{X^{(2)}} J_-\right)^{-1}\right. \\
&\quad \left. \times J_-^T \varphi\left(X^{(2)}\right)^T\right] \varphi\left(x_j\right) \\
&= \sigma^{-1} K\left(x_i, x_j\right) - \sigma^{-1} K\left(x_i, X^{(2)}\right) J_- \left(\sigma \mathrm{I} + J_-^T K_{X^{(2)}} J_-\right)^{-1} \\
&\quad \times J_-^T K\left(X^{(2)}, x_j\right).
\end{aligned} \tag{26}$$

PROOF OF FACT 2

The Lagrangian function of Equation (11) is:

$$\begin{aligned}
&L\left(c_+, R_+, \xi, \alpha, r, s\right) \\
&= R_+^2 - \frac{v_1}{l^-} \sum_{j \in \mathrm{I}^-} \left(\varphi\left(x_j\right) - c_+\right) \Sigma_+^{-1} \left(\varphi\left(x_j\right) - c_+\right) \\
&\quad + \frac{c_1}{l^+} \sum_{i \in \mathrm{I}^+} \xi_i + \sum_{i \in \mathrm{I}^+} \alpha_i \left(\left(\varphi\left(x_i\right) - c_+\right) \Sigma_+^{-1} \left(\varphi\left(x_i\right) - c_+\right)\right. \\
&\quad \left. - R_+^2 - \xi_i\right) - \sum_{i \in \mathrm{I}^+} r_i \xi_i - s R_+^2
\end{aligned} \tag{27}$$

where $s \geq 0, \alpha_i \geq 0, r_i \geq 0, i \in \mathrm{I}^+$ denote the Lagrangian multipliers. Differentiating the Lagrangian function (Equation (27)) concerning c_+, R_+^2 and $\xi_i, i \in \mathrm{I}^+$ yields the following necessary and sufficient Karush-Kuhn-Tucker (KKT) optimality conditions:

$$\frac{\partial L}{\partial R_+^2} = 1 - \sum_{i\in \mathrm{I}^+} \alpha_i - s = 0 => \sum_{i\in \mathrm{I}^+} \alpha_i \leq 1 \tag{28}$$

$$\frac{\partial L}{\partial \xi_i} = \frac{c_1}{l^+} - \alpha_i - r_i = 0 => 0 \leq \alpha_i \leq \frac{c_1}{l^+}, i \in \mathrm{I}^+ \tag{29}$$

$$\begin{aligned}\frac{\partial L}{\partial c_+} &= \frac{2v_1}{l^-} \sum_{j\in \mathrm{I}^-} \left(\varphi\left(x_j\right) - c_+\right)\Sigma_+^{-1} \\ &\quad - 2\sum_{i\in \mathrm{I}^+} \alpha_i \left(\varphi\left(x_i\right) - c_+\right)\Sigma_+^{-1} = 0 \\ &\frac{2v_1}{l^-}\sum_{j\in \mathrm{I}^-}\varphi\left(x_j\right) - \frac{2v_1}{l^-}c_+ - 2\sum_{i\in \mathrm{I}^+}\alpha_i\varphi\left(x_i\right) \\ &+ 2\sum_{i\in \mathrm{I}^+}\alpha_i c_+ = 0 c_+ \left(\frac{-2v_1}{1^-} + 2\sum_{i=\mathrm{I}^+}\alpha_i\right) \\ &= \frac{-2v_1}{l^-}\sum_{j\in \mathrm{I}^-}\varphi\left(x_j\right) + 2\sum_{i\in \mathrm{I}^+}\alpha_i\varphi\left(x_i\right)\end{aligned} \tag{30}$$

$$c_+ = \frac{1}{\sum_{i\in \mathrm{I}^+}\alpha_i - \frac{v_1}{l^-}}\left(\sum_{i\in \mathrm{I}^+}\alpha_i\varphi\left(x_i\right) - \frac{v_1}{l^-}\sum_{j\in \mathrm{I}^-}\varphi\left(x_j\right)\right)$$

$$\left(\varphi\left(x_i\right) - c_+\right)\Sigma_+^{-1}\left(\varphi\left(x_i\right) - c_+\right) \leq R_+^2 + \xi_i, \quad i \in \mathrm{I}^+ \tag{31}$$

$$\alpha_i\left(\left(\varphi\left(x_i\right) - c_+\right)\Sigma_+^{-1}\left(\varphi\left(x_i\right) - c_+\right) - R_+^2 - \xi_i\right) = 0, \quad \alpha_i \geq 0, i \in \mathrm{I}^+ \tag{32}$$

$$r_i\xi_i = 0, \quad \xi_i \geq 0, \ r_i \geq 0, \ i \in \mathrm{I}^+ \tag{33}$$

$$sR_+^2 = 0, \quad R_+^2 \geq 0, \ s \geq 0. \tag{34}$$

By selecting suitable parameters v_1 and c_1 in optimizing Equation (11), the condition will hold, and according to the KKT conditions of Equations (28) and (34), we have:

$$\sum_{i\in \mathrm{I}^+}\alpha_i = 1 \tag{35}$$

and

$$c_+ = \frac{1}{1 - v_1}\left(\sum_{i\in \mathrm{I}^+}\alpha_i\varphi\left(x_i\right) - \frac{v_1}{l^-}\sum_{j\in \mathrm{I}^-}\varphi\left(x_j\right)\right) \tag{36}$$

which also derives an implicit value for v so that $0 < v_1 < 1$. By substituting Equations (28), (29), and (36) into Equation (27), the dual QPP form of Equation (11) is as follows:

$$\begin{aligned}\max &\left(1 - \sum_{i\in \mathrm{I}^+}\alpha_i - s\right)R_+^2 + \sum_{i\in \mathrm{I}^+}\left(\frac{c_1}{l^+} - \alpha_i - r_i\right)\xi_i \\ &- \frac{v_1}{l^-}\sum_{j\in \mathrm{I}^-}\left(\left(\varphi\left(x_j\right) - c_+\right)\Sigma_+^{-1}\left(\varphi\left(x_j\right) - c_+\right)\right) \\ &+ \sum_{i\in \mathrm{I}^+}\alpha_i\left(\left(\varphi\left(x_i\right) - c_+\right)\Sigma_+^{-1}\left(\varphi\left(x_i\right) - c_+\right)\right)\end{aligned}$$

$$\text{s.t.} \sum_{i\in \mathrm{I}^+}\alpha_i = 1, 0 \leq \alpha_i \leq \frac{c_1}{l^+}, i \in \mathrm{I}^+ \tag{37}$$

PROOF OF FACT 3

To solve Equation (37) we have to find the term $\sum_{i\in \mathrm{I}^+}\left(\varphi\left(x_i\right) - c_+\right)\Sigma_+^{-1}\left(\varphi\left(x_i\right) - c_+\right)$ as follows:

$$\begin{aligned}&\sum_{i\in \mathrm{I}^+}\left(\left(\varphi\left(x_i\right) - c_+\right)\Sigma_+^{-1}\left(\varphi\left(x_i\right) - c_+\right)\right) \\ &= \sum_{i\in \mathrm{I}^+}\alpha_i\left(\frac{-2}{1 - v_1}K_+\left(x_i, x_i\right) - \left(\frac{1}{1 - v_1}\right)^2\left(\frac{v_1}{l^-}\right)\right. \\ &\left.\sum_{j\in \mathrm{I}^-}\left(K_+\left(x_i, x_j\right) + K_+\left(x_j, x_i\right)\right)\right) \\ &+ \left(\frac{1}{1 - v_1}\right)^2\sum_{i_1, i_2\in \mathrm{I}^+}\alpha_{i_1}\alpha_{i_2}K_+\left(x_{i_1}, x_{i_2}\right)\end{aligned} \tag{38}$$

Also, the term $\sum_{j\in \mathrm{I}^-}\left(\left(\varphi\left(x_j\right) - c_+\right)\Sigma_+^{-1}\left(\varphi\left(x_j\right) - c_+\right)\right)$ can be computed similarly. Finally, by discarding the constant items from the optimization problem Equation (29), it becomes simpler:

$$\begin{aligned}\max &\sum_{i_1, j_2\in \mathrm{I}^+}\alpha_{i_1}\alpha_{i_2}\left(\left(\frac{1}{1 - v_1}\right)^2\left(1 - \left(\frac{v_1}{l^-}\right)\right)K_+\left(x_{i_1}, x_{i_2}\right)\right) \\ &+ \sum_{i\in \mathrm{I}^+}\alpha_i\left(\left(\frac{v_1}{l^-}\right)\left(\frac{1}{1 - v_1}\right)\sum_{j\in \mathrm{I}^-}\left(K_+\left(x_j, x_i\right) + K_+\left(x_i, x_j\right)\right)\right. \\ &\left. - \left(\frac{2}{1 - v_1}\right)K_+\left(x_i, x_i\right)\right) \\ &\text{s.t.} \sum_{i\in \mathrm{I}^+}\alpha_i = 1, 0 \leq \alpha_i \leq \frac{c_1}{l^+}, i \in \mathrm{I}^+\end{aligned} \tag{39}$$

Now the squared radius R_+^2 is obtained by the following formula:

$$R_+^2 = \frac{1}{\left|\mathrm{I}_R^+\right|}\sum_{i\in \mathrm{I}_{R^+}}\left(\varphi\left(x_i\right) - c_+\right)\Sigma_+^{-1}\left(\varphi\left(x_i\right) - c_+\right) \tag{40}$$

where

$$\mathrm{I}_R^+ = \left\{i | 0 < \alpha_i < \frac{c_1}{l^+}, i \in \mathrm{I}^+\right\}.$$

REFERENCES

[1] B. E. Boser, I. M. Guyon, and V. N. Vapnik, "A training algorithm for optimal margin classifiers," in *Proc. 5th Annu. Workshop Comput. Learn. Theory (COLT)*, Pittsburgh, PA, USA, 1992, pp. 144–152.
[2] V. N. Vapnik, *The Nature of Statistical Learning Theory*, 2nd ed. New York, NY, USA: Springer, 2013.
[3] T. Joachims, C. Ndellec, and C. Rouveriol, "Text categorization with support vector machines: Learning with many relevant features," in *Proc. Eur. Conf. Mach. Learn., Chemnitz*. Berlin, Germany: Springer, 1998, pp. 137–142.

[4] E. Osuna, R. Freund, and F. Girosit, "Training support vector machines: An application to face detection," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, San Juan, Puerto Rico, 1997, pp. 130–136.
[5] J.-Z. Xiao, H.-R. Wang, X.-C. Yang, and Z. Gao, "Multiple faults diagnosis in motion system based on SVM," *Int. J. Mach. Learn. Cybern.*, vol. 3, no. 1, pp. 77–82, Mar. 2012.
[6] Q. He and C. Wu, "Separating theorem of samples in banach space for support vector machine learning," *Int. J. Mach. Learn. Cybern.*, vol. 2, no. 1, pp. 49–54, Mar. 2011.
[7] Z. Liu, Q. Wu, Y. Zhang, and C. L. Philip Chen, "Adaptive least squares support vector machines filter for hand tremor canceling in microsurgery," *Int. J. Mach. Learn. Cybern.*, vol. 2, no. 1, pp. 37–47, Mar. 2011.
[8] X.-Z. Wang, S.-X. Lu, and J.-H. Zhai, "Fast fuzzy multi-category SVM based on support vector domain description," *Int. J. Pattern Recognit. Artif. Intell.*, vol. 22, no. 01, pp. 109–120, Feb. 2008.
[9] C. Jayadeva, R. Khemchandani, and S. Chandra, "Twin support vector machines for pattern classification," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 29, no. 5, pp. 905–910, May 2007.
[10] X. Chen, J. Yang, Q. Ye, and J. Liang, "Recursive projection twin support vector machine via within-class variance minimization," *Pattern Recognit.*, vol. 44, nos. 10–11, pp. 2643–2655, Oct. 2011.
[11] S. Ghorai, A. Mukherjee, and P. K. Dutta, "Nonparallel plane proximal classifier," *Signal Process.*, vol. 89, no. 4, pp. 510–522, Apr. 2009.
[12] M. Arun Kumar and M. Gopal, "Least squares twin support vector machines for pattern classification," *Expert Syst. Appl.*, vol. 36, no. 4, pp. 7535–7543, May 2009.
[13] X. Peng, "Least squares twin support vector hypersphere (LS-TSVH) for pattern recognition," *Expert Syst. Appl.*, vol. 37, no. 12, pp. 8371–8378, Dec. 2010.
[14] X. Peng, "Building sparse twin support vector machine classifiers in primal space," *Inf. Sci.*, vol. 181, no. 18, pp. 3967–3980, Sep. 2011.
[15] M. A. Kumar and M. Gopal, "Application of smoothing technique on twin support vector machines," *Pattern Recognit. Lett.*, vol. 29, no. 13, pp. 1842–1848, Oct. 2008.
[16] X. Peng, "TPMSVM: A novel twin parametric-margin support vector machine for pattern recognition," *Pattern Recognit.*, vol. 44, nos. 10–11, pp. 2678–2692, Oct. 2011.
[17] X. Peng, "TSVR: An efficient twin support vector machine for regression," *Neural Netw.*, vol. 23, no. 3, pp. 365–372, Apr. 2010.
[18] X. Peng, "Efficient twin parametric insensitive support vector regression model," *Neurocomputing*, vol. 79, pp. 26–38, Mar. 2012.
[19] X. Peng and D. Xu, "A twin-hypersphere support vector machine classifier and the fast learning algorithm," *Inf. Sci.*, vol. 221, pp. 12–27, Feb. 2013.
[20] X. Peng and D. Xu, "Twin mahalanobis distance-based support vector machines for pattern recognition," *Inf. Sci.*, vol. 200, pp. 22–37, Oct. 2012.
[21] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*. Cambridge, MA, USA: MIT Press, 2001, pp. 755–756.
[22] S. A. Nene, S. K. Nayar, and H. Murase, "Columbia object image library (COIL-20)," Dept. Comput. Sci., Columbia Univ., New York, NY, USA, Tech. Rep. CUCS-005-96, Feb. 1996.
[23] D. Ruppert, "The elements of statistical learning: Data mining, inference, and prediction," *J. Amer. Stat. Assoc.*, vol. 99, no. 466, p. 567, Jun. 2004.
[24] M. Ballings, D. Van den Poel, N. Hespeels, and R. Gryp, "Evaluating multiple classifiers for stock price direction prediction," *Expert Syst. Appl.*, vol. 42, no. 20, pp. 7046–7056, Nov. 2015.
[25] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," *J. Mach. Learn. Res.*, vol. 7, pp. 1–30, Jan. 2006.
[26] S. García, A. Fernández, J. Luengo, and F. Herrera, "Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: Experimental analysis of power," *Inf. Sci.*, vol. 180, no. 10, pp. 2044–2064, May 2010.
[27] X. Zhang, *Matrix Analysis and Applications*, 1st ed. Cambridge, U.K.: Cambridge Univ. Press, 2017.

ZEINAB EBRAHIMPOUR was born in Sary, Mazandaran, Iran, in 1991. She received the B.S. degree in software engineering and the M.S. degree in computer science from the Hadaf Higher Education Institute, Sary, in 2014 and 2016, respectively. She is currently pursuing the Ph.D. degree in communication and information engineering with Shanghai University, Shanghai, China. From September 2016, she joined the Institute of Smart City, Shanghai University, starting her research on data mining and big data analysis and cooperating with other research groups in Universidad de las Amé ricas Puebla, Mexico. Her research interests include big data, urban planning, human mobility and machine learning and algorithms in social media data analysis that have been involved in different projects funded by the Shanghai Science and Technology Commission. Her honors include the talented student in master's degree, the Co-Chair of the IEEE Conference ICSSC 2016 and ICALIP 2017, and receiving the CSC Scholarship (Chinese Scholarship Council).

WANGGEN WAN (Senior Member, IEEE) received the Ph.D. degree from Xidian University, China, in 1992. From 1991 to 1992, he was a Visiting Scholar with the Department of Engineering, Mink Radio Engineering Institute, USSR. He was a Postdoctoral Research Fellow with the Department of Information and Control Engineering, Xian Jiaotong University, from 1993 to 1995. He was a Visiting Scholar with the Department of Electrical and Electronic Engineering, The Hong Kong University of Science and Technology, from 1998 to 1999. He was a Visiting Professor and the Section Head of Multimedia Innovation Center, The Hong Kong Polytechnic University, from 2000 to 2004. He has been a Full Professor with the School of Communication and Information Engineering, Shanghai University, since 2004, where he is currently the Director of the Institute of Smart City. He has authored over 200 academic articles on international journals and conferences. He has been involved in 30 research projects as the Principal Investigator. His research interests include computer graphics, video and image processing, and data mining. He is the Vice-Chair of the IEEE CIS Shanghai Chapter and the Chair of the IET Shanghai Local Network. He is a fellow of the IET and an ACM Professional Member. He has been the Co-Chairman for many well-known international conferences, since 2008.

ARASH SIOOFY KHOOJINE is currently pursuing the Ph.D. degree in statistics with the School of Mathematical Sciences, Shanghai Jiao Tong University, Shanghai, China, in 2015. His main areas of research interests are statistical network analysis and applied statistics.

LI HOU received the B.S. degree in communication engineering and the M.S. degree in power electronics from the Liaoning University of Technology, in 2003 and 2006, respectively, and the Ph.D. degree in communication and information engineering from Shanghai University, in 2017. She joined the School of Information Engineering, Huangshan University, in 2006. Her current research interests include computer vision, machine learning algorithms, and video/image processing.