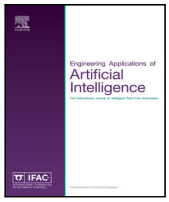
Engineering Applications of Artificial Intelligence 117 (2023) 105615

Contents lists available at ScienceDirect

Engineering Applications of Artificial Intelligence

journal homepage: www.elsevier.com/locate/engappai



Maximal margin hyper-sphere SVM for binary pattern classification

Ting Ke [a,*], Yangyang Liao [b], Mengyan Wu [a], Xuechun Ge [c], Xinyi Huang [a], Chuanlei Zhang [a], Jianrong Li [a]

[a] Department of Artificial Intelligence, College of Artificial Intelligence, Tianjin University of Science & Technology, Tianjin, 300457, China
[b] College of Science, Tianjin University of Science & Technology, Tianjin, 300457, China
[c] Signal and Communication Research Institute, China Academy of Railway Sciences Corporation Limited (Beijing Huatie Information Technology Corporation), Beijing, 100083, China

ARTICLE INFO

Keywords:
Optimization
SVMs
Pattern classification
Hyper-sphere
Structural risk minimization

ABSTRACT

In this paper, we propose a novel maximal margin hyper-sphere support vector machine (MMHS-SVM) for binary pattern classification. Our proposed MMHS-SVM aims to find two hyper-spheres simultaneously by solving a single quadratic programming problem and is consistent between its predicting and training processes. An essential difference that distinguishes it from other hyper-sphere SVMs is that the optimization model is constructed by maximizing the sum of the square distance between centers of two hyper-spheres, but not the sum of squares distances from the center of hyper-sphere to all examples of the opposite class. Such a principle of structural risk in our MMHS-SVM not only helps us grasp the critical samples and eliminate a large number of redundant samples, but also reduces the test cost due to the sparsity. In addition, an effective SMO-typed algorithm is designed to decrease the high time complexity and storage. Finally, a large number of experiments verify the above statements again. The experimental results on several artificial and publicly available benchmark datasets show the feasibility and effectiveness of the proposed method.

1. Introduction

As an excellent kernel-based tool, support vector machine (SVM) has been widely used in data classification analysis (Cortes and Vapnik, 1995). Its core idea is to search for an optimal separating hyper-plane by maximizing the margin between two parallel support planes, and then introduce the kernel trick to solve the nonlinear separable problems. Based on the theory of structural risk minimization, SVM has successfully solved the big-dimensional and local minimum problem. Compared with other machine learning methods, SVM has better generalization performance and achieves superior performance in a wide range of applications, such as text classification (Joachims et al., 1998), gene identification (Guyon et al., 2002), financial regression (Lin et al., 2006), image segmentation (Chen and Wang, 2005), time series analysis (Muller et al., 1999) and face detection (Tao et al., 2016), etc.

Subsequently, many extensions on SVM have been presented to increase the generalization, including parallel hyper-plane SVMs and nonparallel hyper-plane SVMs. Parallel SVMs contain least squares SVM (LSSVM) (Suykens and Vandewalle, 1999; Suykens, 2000; Suykens et al., 2002), proximal SVM (PSVM) (Fung and Mangasarian, 2001), norm-based SVM (Shao et al., 2018; Ke et al., 2021), distance metric SVM (Ke et al., 2020, 2018) and many forms of SVMs for overcoming noise sensitivity or other shortcomings (Hazarika and Gupta, 2020; Hazarika et al., 2021). And non-parallel hyper-plane SVMs include twin SVM (TWSVM) (Jayadeva et al., 2007), NHSVM (Shao et al., 2014) and their extended versions (Tanveer et al., 2022). It is worth mentioning that TWSVM seeks two non-parallel proximal hyper-planes by solving two SVM-typed quadratic programming problems (QPPs) such that each hyper-plane is close to one of two classes, and keeps away from the other class. TWSVM has a lower computational cost than classical SVM since each of the QPPs is only roughly of half size compared with that of the latter. For this reason, many related methods have been derived, such as twin support vector regressions (Xu and Wang, 2012; Gupta, 2017; Lpez and Maldonado, 2018), projection-based twin support vector machines (Chen et al., 2011; Shao et al., 2013; Chen et al., 2019), TWSVM with pinball loss (Xu et al., 2017), least square TWSVM to improve learning speed (Mir and Nasiri, 2018), density-weighted TWSVMs to deal with class imbalance problem (Hazarika and Gupta, 2022), twin parametric Margin SVMs (Rastogi et al., 2018), twin bound SVMs to handle the impacts of outliers (Gupta and Gupta, 2021; Borah and Gupta, 2021), fuzzy twin SVMs (Chen and Wu, 2018; Richhariya and Tanveer, 2018) and so on. However, in many TWSVM-based algorithms, they all need matrix inverse operation. Sometimes the matrix may not be reversible. Solving a matrix inverse is computationally expensive. Furthermore, both parallel and nonparallel hyper-planes just simply divide space into several parts and

* Corresponding author.
E-mail address: keting@tust.edu.cn (T. Ke).

https://doi.org/10.1016/j.engappai.2022.105615
Received 26 May 2022; Received in revised form 26 October 2022; Accepted 4 November 2022
Available online 23 November 2022

give little consideration to the distribution characteristics of each class. For example, it is impossible to use hyper-plane to reflect the examples coming from the distinct Gaussian distributions. To remedy this problem, spherical-structured SVMs have appeared (Hao et al., 2009; Wen et al., 2010; Tomar and Agarwal, 2015). Wu and Ye constructed a single model to find two homocentric spheres for the imbalanced data classification called SSLM (Wu and Ye, 2009). Existing SSLM solvers cannot deal with large data due to the expensive time cost. The twin hyper-sphere support vector machine (THSVM) (Peng and Xu, 2013) finds two irrelevant hyper-spheres by solving a pair of smaller QPPs. This strategy avoids the inversions of two matrices that appeared in the dual QPPs of the TWSVM successfully. However, its models in which the two hyper-spheres are constructed separately so that it cannot be consistent between its predicting and training processes. Moreover, the model does not implement the structural risk minimization (SRM) principle, leading to poor generalization ability. Xu (2016) came up with a maximum margin of twin hyper-spheres support vector machine (MMTSSVM) for the imbalanced data classification. It finds two homocentric hyper-spheres rather than two different hyper-spheres by solving one QPP and one linear programming problem (LPP), thus greatly increasing the computational speed. However, it is sensitive to outliers. To enhance the stability, ramp loss and pinball loss are participated in SSLM and MMTSSVM, obtain Pin-M^{3}HM (Xu et al., 2016), Ramp-MMTSSVM (Lu et al., 2018), Pin-MMTSM (Xu et al., 2018) and Ramp-SSLM (Wang and Xu, 2022). Subsequently, Cao et al. (2019) constructed a new safe screening rule for SSLM (MVE-SSR-SSLM) by integrating the v-property, KKT conditions, and variational inequalities. The inactive samples are removed before actually solving the problem to accelerate the solution procedure without any loss of safety. So, it has a lower time complexity. Yuan proposed a safe accelerative approach to reduce the computational cost of Pin-MMTSM, called SSR-DM-Pin-MMTSM (Yuan and Xu, 2021). Besides, KNN-M^{3}VHM (Xu et al., 2019) constructed two hyper-spheres with different centers and radiuses using the k-nearest neighbor (KNN) strategy to remove some redundant majority samples and balance two classes of samples. Nevertheless, The KNN algorithm in KNN-M^{3}VHM spends much training time and storage space. In summary, we find that all of SSLM, MMTSSVM, Ramp-MMTSSVM, Pin-MMTSM, MVE-SSR-SSLM, Ramp-SSLM and DM-Pin-MMTSM seek two homocentric spheres which will lead to consider only the data distribution of one class and that of another class is ignored. From another perspective, the above hyper-spheres, except for SSLM, MVE-SSR-SSLM and Ramp-SSLM, all lose sparsity because they construct two optimization models separately to keep all points away from the center of the opposite class. This will inevitably lead to the sensitiveness to outliers and an increase in the cost of the test (Wang et al., 2020).

To overcome the above drawbacks, while inheriting all advantages of hyper-plane SVMs and hyper-sphere SVMs, this paper proposes a general maximum margin hyper-sphere SVM classifier, called MMHS-SVM in short. First, we construct just one optimization model to seek two different hyper-spheres simultaneously in accordance with class distribution by maximizing the square of the difference between two class centers, rather than the sum of squares distances from the center of the hyper-sphere to the examples of the opposite class. It means that our hyper-spheres not only embody the structural risk principle, keeping one class far from another one, but also reflect the two classes' distribution information. Second, the dual problem is derived, and the kernel trick can be introduced to map the high-dimensional space to classify the nonlinear separable problems. Finally, an SMO-typed algorithm reduces the consuming time and storage space. In summary, we can conclude the following advantages compared to related methods:

1. Similar to the classical SVM, MMHS-SVM constructs the classifier based on the principle of structural risk minimization. Thus, MMHS-SVM possesses all merits of the classical SVM, such as global optimization, strong adaptability, strong generalization ability, sparse solution, and nonlinear separability with kernel trick.
2. Maximizing the square of the difference between two class centers in our optimization model, other than the sum of squares distances from the center of the hyper-sphere to all examples of the opposite class, such as in THSVM, MMTSSVM and their extensions, can avoid the sensitiveness to outliers or noise and has more strong robustness.
3. In the dual problem, only a small number of support vectors determine the final hyper-spheres, which can not only help us grasp the essential samples and eliminate a large number of redundant samples, but also reduce the test cost due to the sparsity.
4. Although a single optimization model may cause higher time complexity than that of two small-size models in theory, there is consistency between the training and prediction phase. It means that our model owns stronger generalization ability and robustness. Moreover, to reduce the high time complexity and storage of QPPs, we designed an effective SMO-typed algorithm inspired by Osuna's theorem (Osuna et al., 1997), which decomposes the general QPP into sub-problems of QPs and ensures convergence.
5. A large number of experiments again verify the above statements, which shows that our method has stronger generalization ability and robustness.
6. It is worth mentioning that the idea of MMHS-SVM is pioneering and its model is very similar to the classical SVM. Based on this model, many new methods will be derived, such as multi-class classification hyper-sphere SVMs, distance-based hyper-spheres, other loss hyper-spheres, etc. This is what we will do next.

The structure of this paper is arranged as follows. We will introduce some related work in Section 2. Section 3 presents our hyper-sphere optimal model. Then the SMO-typed algorithm is designed in Section 4. The connections between MMHS-SVM and other related methods are compared in Section 5. We implement the experiments on synthetic Gaussian datasets, UCI datasets, and digital handwritten datasets in Section 6. Finally, we conclude the overall work in Section 7.

2. Backgrounds

In this section, we summarize classical hyper-plane SVM and hyper-sphere SVMs, such as THSVM and MMTSSVM for binary classification problems. For convenience, the following notation is used throughout this paper. Consider a set of l labeled training samples: $D = \{(x_1, y_1), (x_2, y_2), \ldots, (x_l, y_l)\}$. $x_i \in R^n, i = 1, 2, \ldots, l$ is a n-dimensional feature vector, representing the ith training sample, and $y_i \in \{1, -1\}$, $i = 1, 2, \ldots, l$ is the class label of $x_i \in R^n$. $I_\pm$ denote the sets of indices such that $y_i = \pm 1$, respectively, and $I = I_+ \cup I_-$ is the set of all indices.

2.1. Support vector machine

Traditional SVM takes a separating hyper-plane to establish the classifier with a maximal margin. This separating hyper-plane can be described as the equation:

$$f(x) = \langle w, \varphi(x) \rangle + b = 0 \tag{1}$$

The function $\varphi(\cdot)$ maps the input space into a higher-dimensional space when the dataset is nonlinearly separable. $w \in R^n$ is the normal vector of Eq. (1) and b is a scalar. It is easy to find that the parameter pair (w, b) corresponding to the optimal hyperplane is the solution to the following convex QPP:

$$\begin{aligned} \min_{(w,b,\xi)} \quad & \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{l}\xi_i \\ s.t. \quad & y_i(\langle w, \varphi(x_i)\rangle + b) \geq 1 - \xi_i i = 1, 2, \ldots, l \\ & \xi_i \geq 0 i = 1, 2, \ldots, l \end{aligned} \tag{2}$$

where $\xi = (\xi_1, \xi_2, \ldots, \xi_l)^T$ is penalizing variable vector of training samples. The parameter $C > 0$ balances the importance between

maximization of the margin width and minimization of the training error. The dual problem of (2) with variables $\alpha_i, i = 1, \ldots, l$ is obtained by introducing the Lagrangian function and the class label y of sample x is determined by the sign of the following decision function:

$$w^T \varphi(x) + b = \sum_{i=1}^{l} \alpha_i y_i K\left(x_i, x\right) + b \tag{3}$$

$K\left(x_i, x_j\right)$ is a kernel function that computes the inner product of vectors $\varphi(x_i)$ and $\varphi(x_j)$.

2.2. Twin hyper-sphere SVM

In the spirit of TWSVM, Peng and Xu (2013) proposed a THSVM. THSVM uses a pair of hyper-spheres instead of a pair of hyper-planes to classify two classes of samples. Specifically, samples in one class are covered by the corresponding hyper-sphere, whereas samples in other class are as far as possible from this hyper-sphere. The optimization problems are constructed as follows:

$$\begin{aligned} \min_{a_1, R_1^2, \xi} \quad & -\sum_{j \in I_-} \left\| \varphi\left(x_j\right) - a_1 \right\|^2 + C_1 R_1^2 + C_2 \sum_{i \in I_+} \xi_i \\ s.t. \quad & \left\| \varphi\left(x_i\right) - a_1 \right\|^2 \le R_1^2 + \xi_i, i \in I_+ \\ & \xi_i \ge 0, i \in I_+, R_1^2 \ge 0 \end{aligned} \tag{4}$$

$$\begin{aligned} \min_{a_2, R_2^2, \eta} \quad & -\sum_{i \in I_+} \left\| \varphi\left(x_i\right) - a_2 \right\|^2 + C_3 R_2^2 + C_4 \sum_{j \in I_-} \eta_j \\ s.t. \quad & \left\| \varphi\left(x_j\right) - a_2 \right\|^2 \le R_2^2 + \eta_j, j \in I_- \\ & \eta_j \ge 0, j \in I_-, R_2^2 \ge 0 \end{aligned} \tag{5}$$

where $C_1, C_2, C_3, C_4 > 0$ are the penalty factors, and a_1, a_2 and R_1^2, R_2^2 are the centers and radiuses of the corresponding hyper-spheres, respectively. In brief, the samples in one class are as many as possibly covered by the corresponding hyper-sphere, whereas the samples in the other class are as far as possible from this hyper-sphere. Each model of (4) and (5) find a hyper-sphere for each class, and classifies a new point according to which hyper-sphere a given point is relatively closest. Specifically, once the centers and square radiuses are calculated, the two hyper-spheres $\left\| \varphi(x) - a_1 \right\|^2 \le R_1^2, \left\| \varphi(x) - a_2 \right\|^2 \le R_2^2$ are obtained from the Eqs. (4) and (5). A new example is assigned to the class + or −, depending on which of the two hyper-spheres given by (6) it lies closest to, i.e.,

$$f(x) = \operatorname{argmin} \left\{ \frac{\left\| \varphi(x) - a_1 \right\|^2}{R_1^2}, \frac{\left\| \varphi(x) - a_2 \right\|^2}{R_2^2} \right\} \tag{6}$$

2.3. Maximum margin of twin spheres support vector machine

Xu (2016) proposed a maximum margin of twin spheres support vector machine (MMTSSVM). MMTSSVM constructs two homocentric hyper-spheres rather than two different hyper-spheres. Namely, the small sphere contains as many positive samples as possible, and most negative samples are pushed outside the large sphere. Its models are formulated as follows.

$$\begin{aligned} \min_{a_1, R^2, \xi} \quad & -\frac{\upsilon}{l_-} \sum_{j \in I_-} \left\| \varphi\left(x_j\right) - a \right\|^2 + R^2 + \frac{1}{l_+ C_1} \sum_{i \in I_+} \xi_i \\ s.t. \quad & \left\| \varphi\left(x_i\right) - a \right\|^2 \le R^2 + \xi_i, i \in I_+ \\ & \xi_i \ge 0, i \in I_+, R^2 \ge 0 \end{aligned} \tag{7}$$

$$\begin{aligned} \min_{\rho^2, \eta} \quad & R^2 - \rho^2 + \frac{1}{l_- C_2} \sum_{j \in I_-} \eta_j \\ s.t. \quad & \left\| \varphi\left(x_j\right) - a \right\|^2 \ge R^2 + \rho^2 - \eta_j, j \in I_- \\ & \eta_j \ge 0, j \in I_-, R^2 \ge 0 \end{aligned} \tag{8}$$

where $R^2 + \rho^2$ is the radius of the large hyper-sphere. A new testing sample is assigned to a class i ($i = +1, -1$) by comparing its distance from the center of the sphere with $R^2 + \left(R^2 + \rho^2\right)/2$ which is the mean value of the radiuses of the small and large spheres. There is a similar structure for SSLM and MMTSSVM. And they have a greater ability to address the imbalanced data classification.

3. Maximal margin hyper-sphere SVM

SVM uses a hyper-plane to divide the classification space into two parts, ignoring the distributional characteristics and structure of each class. Although THSVM embodies the class distribution, it does not implement SRM and cannot be consistence between the training and testing process. Additionally, the following properties also exist: (1) both THSVM and MMTSSVM consider that all samples are far away from each other's class center. Such over-detailed consideration will cause that the outliers or noise are participated in the learning process too much, resulting in lower robustness. (2) THSVM and MMTSSVM lose sparsity and increase the cost in the process of test. To overcome these shortcomings, we present a novel classifier called maximal margin hyper-sphere support vector machine (MMHS-SVM) in this section.

3.1. MMHS-SVM classifier

Considering the binary class classification problem, which is described in Section 2, we construct a single optimization model to find two hypers-spheres $\left\| \varphi(x) - a_1 \right\|^2 \le R_1^2, \left\| \varphi(x) - a_2 \right\|^2 \le R_2^2$ simultaneously so that the center distance of the two spheres is as far as possible. The radius of each hyper-sphere is as tiny as possible. Meantime, each hyper-sphere contains as many sample points in its corresponding sphere as possible. Thus, the optimization model is established by

$$\begin{aligned} \min_{a_1, a_2, R_1^2, R_2^2, \xi} \quad & -\left\| a_1 - a_2 \right\|^2 + C_1 \left(R_1^2 + R_2^2\right) + C_2 \left(\sum_{i \in I_+} \xi_i + \sum_{j \in I_-} \xi_j \right) \\ s.t. \quad & \left\| \varphi\left(x_i\right) - a_1 \right\|^2 \le R_1^2 + \xi_i, i \in I_+ \\ & \left\| \varphi\left(x_j\right) - a_2 \right\|^2 \le R_2^2 + \xi_j, j \in I_- \\ & \xi_i \ge 0, i \in I = I_+ \cup I_- \\ & R_1^2 > 0, R_2^2 > 0 \end{aligned} \tag{9}$$

where a_1, a_2 are centers and R_1, R_2 are the radiuses of two hyper-spheres for positive and negative samples, and R_1^2, R_2^2 are the squares of radiuses. The model (9) employs the term $-\left\| a_1 - a_2 \right\|^2$ instead of term $\sum_{j \in I_-} \left\| \varphi\left(x_j\right) - a_1 \right\|^2$ to make the distance between two hyper-spheres as large as possible, which implies that our model follows the maximum margin principle. In addition, it costs less time to compute term $-\left\| a_1 - a_2 \right\|^2$ than $\sum_{j \in I_-} \left\| \varphi\left(x_j\right) - a_1 \right\|^2$. The second term $\left(R_1^2 + R_2^2\right)$ controls the compactness of each hyper-sphere. Minimizing the relaxation variables with equality constraints means that the samples outside the hyper-sphere make as few mistakes as possible. C_1, C_2 are the weights to trade off the importance of the terms in objective function in (9). The function $\varphi(\cdot)$ maps the input space into a higher-dimensional space when the dataset is nonlinearly separable. Additionally, only one optimization problem can be consistency between the training and testing process. we compute the centers and square radiuses, a new test data point x is assigned to the class + if it lies closest to the first hyper-sphere, otherwise, it is classified as class −.

$$f(x) = \operatorname{argmin} \left\{ \frac{\left\| \varphi(x) - a_1 \right\|^2}{R_1^2}, \frac{\left\| \varphi(x) - a_2 \right\|^2}{R_2^2} \right\} \tag{10}$$

We now take into account optimizing Eq. (10). Introducing the Lagrange multiplier vectors $\alpha, \beta, \gamma, s_1, s_2$ for optimization problem (9),

we obtain the corresponding Lagrange function,

$$\begin{aligned} &L\left(a_1, a_2, R_1^2, R_2^2, \xi, \alpha, \beta, \gamma, s_1, s_2\right) \\ &= -\left\|a_1 - a_2\right\|^2 + C_1 R_1^2 + C_1 R_2^2 + C_2 \sum_{i \in I} \xi_i \\ &+ \sum_{i \in I_+} \alpha_i \left(\left\|\varphi\left(x_i\right) - a_1\right\|^2 - R_1^2 - \xi_i\right) \\ &+ \sum_{j \in I_-} \beta_j \left(\left\|\varphi\left(x_j\right) - a_2\right\|^2 - R_2^2 - \xi_j\right) - \sum_{i \in I} \gamma_i \xi_i - s_1 R_1^2 - s_2 R_2^2 \end{aligned} \tag{11}$$

Taking the partial derivatives of (11) concerning the variables $a_1, a_2, R_1^2, R_2^2, \xi, \alpha, \beta, \gamma, s_1, s_2$ and equating them to zero, the following optimal conditions are obtained:

$$\frac{\partial L}{\partial a_1} = -2\left(a_1 - a_2\right) - 2\sum_{i \in I_+} \alpha_i \left(\varphi\left(x_i\right) - a_1\right) = 0 \tag{12}$$

$$\frac{\partial L}{\partial a_2} = 2\left(a_1 - a_2\right) - 2\sum_{j \in I_-} \beta_j \left(\varphi\left(x_j\right) - a_2\right) = 0 \tag{13}$$

$$\frac{\partial L}{\partial R_1^2} = C_1 - \sum_{i \in I_+} \alpha_i - s_1 = 0 \tag{14}$$

$$\frac{\partial L}{\partial R_2^2} = C_1 - \sum_{j \in I_-} \beta_j - s_2 = 0 \tag{15}$$

$$\frac{\partial L}{\partial \xi_i} = C_2 - \alpha_i - \gamma_i = 0, i \in I_+ \tag{16}$$

$$\frac{\partial L}{\partial \xi_j} = C_2 - \beta_j - \gamma_j = 0, j \in I_- \tag{17}$$

$$\left\|\varphi\left(x_i\right) - a_1\right\|^2 \le R_1^2 + \xi_i, i \in I_+ \tag{18}$$

$$\left\|\varphi\left(x_j\right) - a_2\right\|^2 \le R_2^2 + \xi_j, j \in I_- \tag{19}$$

$$\alpha_i \left(\left\|\varphi\left(x_i\right) - a_1\right\|^2 - R_1^2 - \xi_i\right) = 0, \alpha_i \ge 0, i \in I_+ \tag{20}$$

$$\beta_j \left(\left\|\varphi\left(x_j\right) - a_2\right\|^2 - R_2^2 - \xi_j\right) = 0, \beta_j \ge 0, i \in I_- \tag{21}$$

$$\gamma_i \xi_i = 0, \gamma_i \ge 0, \xi_i \ge 0, i \in I \tag{22}$$

$$s_1 R_1^2 = 0, s_2 R_2^2 = 0 \tag{23}$$

Note that $R_1^2 > 0, R_2^2 > 0$, according to (23), we have $s_1 = 0, s_2 = 0$. Then we receive

$$\sum_{i \in I_+} \alpha_i = C_1, \sum_{j \in I_-} \beta_j = C_1 \tag{24}$$

By deriving (12) and (13), the following equations are obtained.

$$\begin{aligned} &\left(1 - \sum_{i \in I_+} \alpha_i\right) a_1 - a_2 + \sum_{i \in I_+} \alpha_i \varphi\left(x_i\right) = 0 \\ &\left(1 - \sum_{j \in I_-} \beta_i\right) a_2 - a_1 + \sum_{j \in I_-} \beta_j \varphi\left(x_j\right) = 0 \end{aligned} \tag{25}$$

Combine (24), solve the above equations

$$\begin{aligned} a_1 &= \frac{1}{\left(2 - C_1\right) C_1} \left(\left(1 - C_1\right) \sum_{i \in I_+} \alpha_i \varphi\left(x_i\right) + \sum_{j \in I_-} \beta_j \varphi\left(x_j\right)\right) \\ a_2 &= \frac{1}{\left(2 - C_1\right) C_1} \left(\sum_{i \in I_+} \alpha_i \varphi\left(x_i\right) + \left(1 - C_1\right) \sum_{j \in I_-} \beta_j \varphi\left(x_j\right)\right) \end{aligned} \tag{26}$$

Substituting (26) into (11), we obtain the dual QPP of (9) as shown in Eq. (27):

$$\begin{aligned} &\max_{\alpha, \beta} -\left(\frac{1}{\left(2 - C_1\right)}\right)^2 \left\|\sum_{i \in I_+} \alpha_i \varphi\left(x_i\right) - \sum_{j \in I_-} \beta_j \varphi\left(x_j\right)\right\|^2 \\ &+ \sum_{i \in I_+} \alpha_i \left\|\varphi\left(x_i\right) - \frac{1}{\left(2 - C_1\right) C_1} \left(\left(1 - C_1\right) \sum_{i \in I_+} \alpha_i \varphi\left(x_i\right) + \sum_{j \in I_-} \beta_j \varphi\left(x_j\right)\right)\right\|^2 \\ &+ \sum_{j \in I_-} \beta_j \left\|\varphi\left(x_j\right) - \frac{1}{\left(2 - C_1\right) C_1}\right. \\ &\left. \times \left(\sum_{i \in I_+} \alpha_i \varphi\left(x_i\right) + \left(1 - C_1\right) \sum_{j \in I_-} \beta_j \varphi\left(x_j\right)\right)\right\|^2 \\ &s.t. \quad \sum_{i \in I_+} \alpha_i = C_1 \\ &\quad \sum_{i \in I_-} \beta_j = C_1 \\ &\quad 0 \le \alpha_i \le C_2, i \in I_+ \\ &\quad 0 \le \beta_j \le C_2, j \in I_- \end{aligned} \tag{27}$$

Naturally, the inner product $\langle \varphi\left(x_i\right), \varphi\left(x_j\right) \rangle$ is deemed as a kernel function, recorded as $K\left(x_i, x_j\right) = \langle \varphi\left(x_i\right), \varphi\left(x_j\right) \rangle = K_{ij}$. Thus, (27) is organized as:

$$\begin{aligned} &\min_{\alpha, \beta} \left(\frac{1 - C_1}{\left(2 - C_1\right) C_1}\right) \sum_{i_1 \in I_+} \sum_{i_2 \in I_+} \alpha_{i_1} \alpha_{i_2} K\left(x_{i_1}, x_{i_2}\right) \\ &+ \left(\frac{1 - C_1}{\left(2 - C_1\right) C_1}\right) \sum_{j_1 \in I_-} \sum_{j_2 \in I_-} \beta_{j_1} \beta_{j_2} K\left(x_{j_1}, x_{j_2}\right) \\ &+ \frac{2}{\left(2 - C_1\right) C_1} \sum_{i \in I_+} \sum_{j \in I_-} \alpha_i \beta_j K\left(x_i, x_j\right) - \sum_{i \in I_+} \alpha_i K\left(x_i, x_i\right) - \sum_{j \in I_-} \beta_j K\left(x_j, x_j\right) \\ &s.t. \quad \sum_{i \in I_+} \alpha_i = C_1 \\ &\quad \sum_{i \in I_-} \beta_j = C_1 \\ &\quad 0 \le \alpha_i \le C_2, i \in I_+ \\ &\quad 0 \le \beta_j \le C_2, j \in I_- \end{aligned} \tag{28}$$

Once α, β are solved, centers in (26) can be computed. Next, according to (16)–(17), (20)–(21), we notice that $\left\|\varphi\left(x_i\right) - a_1\right\|^2 = R_1^2$ if $0 < \alpha_i < C_2, i \in I_+$, and $\left\|\varphi\left(x_j\right) - a_2\right\|^2 = R_2^2$ if $0 < \beta_j < C_2, j \in I_-$. Thus, we compute the square radius R_1^2 and R_2^2 as follows.

$$\begin{aligned} R_1^2 &= \frac{1}{\left|I_{k+}\right|} \sum_{i \in I_{k+}} \left\|\varphi\left(x_i\right) - a_1\right\|^2 \\ R_2^2 &= \frac{1}{\left|I_{k-}\right|} \sum_{j \in I_{k-}} \left\|\varphi\left(x_j\right) - a_2\right\|^2 \end{aligned} \tag{29}$$

where $I_{k+} = \left\{i \left| 0 < \alpha_i < C_2, i \in I_+ \right.\right\} I_{k-} = \left\{j \left| 0 < \beta_i < C_2, i \in I_- \right.\right\}$. Like the classical SVM, we call these points on the hyper-sphere as support vectors. That is to say, these sample points determine the final hyper-sphere, so the classifier also has sparsity and results in lower computation time in the testing phase.

3.2. The determination of the domain of the parameter C_1

Let $\theta = \left(\alpha^T, \beta^T\right)^T$, $H_{l \times l} = \begin{pmatrix} \left(1 - C_1\right) K_+ & K_\pm \\ K_\pm^T & \left(1 - C_1\right) K_- \end{pmatrix}$ where $\left(K_+\right)_{i_1 i_2} = \langle \varphi\left(x_{i_1}\right), \varphi\left(x_{i_2}\right) \rangle, i_1, i_2 \in I_+, \left(K_-\right)_{j_1 j_2} = \langle \varphi\left(x_{j_1}\right), \varphi\left(x_{j_2}\right) \rangle$, $j_1, j_2 \in I_-$, $\left(K_\pm\right)_{ij} = \langle \varphi\left(x_i\right), \varphi\left(x_j\right) \rangle, i \in I_+, j \in I_-$. Suppose that $Q =$

$$\left(K_{11}, K_{22}, \ldots, K_{ll}\right)^T, e_{10} = \left(\underbrace{1, \ldots, 1}_{l_+}, \underbrace{0, \ldots, 0}_{l_-}\right)^T, \; e_{01} = \left(\underbrace{0, \ldots, 0}_{l+}, \underbrace{1, \ldots, 1}_{l_-}\right)^T$$

where $l_\pm$ is the number of the elements of $I_\pm$. The matrix form of Eq. (30) is equivalent to formula (29)

$$\begin{aligned} \min_{\theta} \quad & \frac{1}{\left(2-C_1\right)C_1}\theta^T H\theta - Q^T\theta \\ s.t. \quad & e_{10}^T\theta = C_1 \\ & e_{01}^T\theta = C_1 \\ & O \le \theta \le C_2 e \end{aligned} \tag{30}$$

We find that formula (30) is a quadratic programming problem, but it may not be convex. It means that the global minimal solution may not exist. Thus, we adjust the parameters C_1 to make the objective function a convex quadratic programming problem. In this case, the following theorem gives an optimistic conclusion.

Theorem 1. *When the range of the parameter C_1 is in $0 < C_1 < \min\left\{2, \left\{\min_x \frac{x^T K x}{x^T K_{diag} x}\right\}\right\}$ or $C_1 > \max\left\{2, \left\{\max_x \frac{x^T K x}{x^T K_{diag} x}\right\}\right\}$, then $\frac{1}{(2-C_1)C_1} H$ is a positive semi-definite matrix, and formula (30) is a convex quadratic programming problem, which has a globally optimal solution.*

Proof. According to the formula (30), $\frac{1}{(2-C_1)C_1} H$ can be rewritten as $\frac{1}{(2-C_1)C_1} H = \frac{1}{(2-C_1)C_1}\left(K - C_1 K_{diag}\right)$. K is a Gram matrix which is positive semi-definite, and $K_{diag} = \begin{pmatrix} K_+ & 0 \\ 0 & K_- \end{pmatrix}$ is a block matrix which is also positive definite.

$\frac{1}{(2-C_1)C_1} H$ is positive semi-definite matrix if and only if it satisfies:

$$\forall x \in R^l, \frac{1}{\left(2-C_1\right)C_1} x^T H x = \frac{1}{\left(2-C_1\right)C_1}\left(x^T K x - x^T C_1 K_{diag} x\right) \ge 0.$$

Thus, it can be deduced that

(1) $\frac{1}{(2-C_1)C_1} > 0, \left(x^T K x - x^T C_1 K_{diag} x\right) \ge 0 \Rightarrow 0 < C_1 < \min\left\{2, \left\{\min_x \frac{x^T K x}{x^T K_{diag} x}\right\}\right\}$;

(2) $\frac{1}{(2-C_1)C_1} < 0, \left(x^T K x - x^T C_1 K_{diag} x\right) \le 0 \Rightarrow C_1 > \max\left\{2, \left\{\max_x \frac{x^T K x}{x^T K_{diag} x}\right\}\right\}$.

To sum up, when $0 < C_1 < \min\left\{2, \left\{\min_x \frac{x^T K x}{x^T K_{diag} x}\right\}\right\}$ and $C_1 > \max\left\{2, \left\{\max_x \frac{x^T K x}{x^T K_{diag} x}\right\}\right\}$, $\frac{1}{(2-C_1)C_1} H$ is a positive semi-definite.

Combining the above two situations, the theorem has been proved. Theorem 1 shows that the parameter value in a particular range can guarantee the convex function of the objective function. Besides, the determination of the parameter range helps us reduce the training process cost. However, how do we compute the $\max_x \frac{x^T K x}{x^T K_{diag} x}$ and $\min_x \frac{x^T K x}{x^T K_{diag} x}$? Next, we provide Theorem 2.

Theorem 2. *As K and K_{diag} are kernel matrices, they are all real symmetric, positive-definite and invertible. Suppose $\lambda_1 \ge \lambda_2 \ge \cdots \ge \lambda_l$ are the eigenvalues of $K_{diag}^{-1} K$, then $\max_{x\neq 0} \frac{x^T K x}{x^T K_{diag} x} = \lambda_1$ and $\min_{x\neq 0} \frac{x^T K x}{x^T K_{diag} x} = \lambda_l$.*

Proof. Let $y = K_{diag}^{1/2} x, K_1 = K_{diag}^{-1/2} K K_{diag}^{-1/2}$, K_1 is a real symmetric matrix. Substituting them into the following formula, we obtain

$$\max_{x\neq 0} \frac{x^T K x}{x^T K_{diag} x} = \max_{y\neq O} \frac{y^T K_1 y}{y^T y}.$$

Since $\left(K_{diag}^{-1/2} K\right) K_{diag}^{-1/2}$ and $K_{diag}^{-1/2}\left(K_{diag}^{-1/2} K\right) = K_{diag}^{-1/2} K_{diag}^{-1/2} K = K_{diag}^{-1} K$ have the same eigenvalues,
Thus

$$\max_{x\neq O} \frac{x^T K x}{x^T K_{diag} x} = \max_{y\neq O} \frac{y^T K_{diag}^{-1} K y}{y^T y} = \lambda_1$$

Similar to the above derivation, we obtain

$$\min_{x\neq O} \frac{x^T K x}{x^T K_{diag} x} = \lambda_l.$$

4. SMO-typed algorithm

Similar to that of classical hyper-plane SVM, training a classifier in our MMHS-SVM model requires the solution of QPP with two inequality constraints which may lead to higher computational time and storage costs, especially when the amount of data is large. SMO decomposes the large QPP into a series of the smallest QPP and then implements iterative operation (Platt, 1999). Inspired by this thinking, an improved algorithm is designed to solve the optimization problem (30) in this section. Three-step techniques will be implemented.

4.1. Solving two Lagrange multipliers

To reduce computational time and storage costs, the smallest optimization problem, involving two Lagrange multipliers, is chosen. The radius multipliers are the original solutions received in the last iterative step. This is because the Lagrange multipliers must obey one of two linear equality constraints in (28). How to choose them will be described in detail in Section 4.2. Without loss of generality, if the first linear equality constraints in (28) are satisfied, these two multipliers are noted as α_1, α_2, otherwise, they are noted as β_1, β_2. At each iterative step, we choose two Lagrange multipliers to jointly optimize, find the optimal values for these multipliers.

Firstly, if α_1, α_2 are chosen from the first equality constraints in (28), we obtain the following QPP:

$$\begin{aligned} & \min_{\alpha_1,\alpha_2} \left(\frac{1-C_1}{\left(2-C_1\right)C_1}\right)\left(K_{11}\alpha_1^2 + K_{22}\alpha_2^2 + 2K_{12}\alpha_1\alpha_2 + 2v_1\alpha_1 + 2v_2\alpha_2\right) \\ & + \frac{2}{\left(2-C_1\right)C_1}\left(u_1\alpha_1 + u_2\alpha_2\right) - K_{11}\alpha_1 - K_{22}\alpha_2 + \text{Cons} \\ & s.t. \quad \alpha_1 + \alpha_2 = C_1 - \sum_{i=3}^{l_+} \alpha_i \\ & \quad \sum_{i\in I_-} \beta_j = C_1 \\ & \quad 0 \le \alpha_i \le C_2, i \in I_+ \\ & \quad 0 \le \beta_j \le C_2, j \in I_- \end{aligned} \tag{31}$$

where $v_i = \sum_{j=3}^{l_+} \alpha_i K_{ij}, i = 1, 2, u_i = \sum_{j=1}^{l_-} \beta_j K_{ij}, i = 1, 2$. Record $K\left(x_i, x_j\right)$ as K_{ij}. And *Cons* is the rest Lagrange multipliers, deemed as constants. Due to

$$\alpha_1 + \alpha_2 = C_1 - \sum_{i=3}^{l_+} \alpha_i \Rightarrow \alpha_1 = C_1 - \sum_{i=3}^{l_+} \alpha_i - \alpha_2$$

Substituting the above formula into (31),

$$\begin{aligned} \min_{\alpha_1,\alpha_2} \phi\left(\alpha_2\right) = & \left(\frac{1-C_1}{\left(2-C_1\right)C_1}\right)\begin{pmatrix} K_{11}\left(C_1 - \sum_{i=3}^{l_+}\alpha_i - \alpha_2\right)^2 \\ + K_{22}\alpha_2^2 + 2K_{12}\left(C_1 - \sum_{i=3}^{l_+}\alpha_i - \alpha_2\right)\alpha_2 \\ + 2v_1\left(C_1 - \sum_{i=3}^{l_+}\alpha_i - \alpha_2\right) + 2v_2\alpha_2 \end{pmatrix} \\ & + \frac{2}{\left(2-C_1\right)C_1}\left(u_1\left(C_1 - \sum_{i=3}^{l_+}\alpha_i - \alpha_2\right) + u_2\alpha_2\right) - K_{11}\left(C_1 - \sum_{i=3}^{l_+}\alpha_i - \alpha_2\right) \\ & - K_{22}\alpha_2 + \text{Cons} \end{aligned} \tag{32}$$

Computing the partial derivatives of (32) concerning α_2 and equating it to zero, optimal conditions are

$$\begin{aligned}\frac{\partial\phi(\alpha_2)}{\partial\alpha_2} &= \left(\frac{1-C_1}{(2-C_1)C_1}\right)\left(-2K_{11}\left(C_1-\sum_{i=3}^{l_+}\alpha_i-\alpha_2\right)+2K_{22}\alpha_2\right)\\ &+\left(\frac{1-C_1}{(2-C_1)C_1}\right)\left(2K_{12}\left(C_1-\sum_{i=3}^{l_+}\alpha_i\right)-4K_{12}\alpha_2-2v_1+2v_2\right)\\ &+\frac{2}{(2-C_1)C_1}(-u_1+u_2)+K_{11}-K_{22}=0\end{aligned} \tag{33}$$

As $C_1-\sum_{i=3}^{l_+}\alpha_i=\alpha_1^{old}+\alpha_2^{old}$, where $\alpha_1^{old},\alpha_2^{old}$ are original solutions calculated at the last iterative step, (33) is rewritten as

$$\begin{aligned}&\frac{2(1-C_1)}{(2-C_1)C_1}(K_{11}+K_{22}-2K_{12})\alpha_2=\\ &\frac{2(1-C_1)}{(2-C_1)C_1}\left((K_{11}-K_{12})\alpha_2^{old}+(K_{11}-K_{12})\alpha_1^{old}+v_1-v_2\right)\\ &+\frac{4}{(2-C_1)C_1}(u_1-u_2)-K_{11}+K_{22}\end{aligned} \tag{34}$$

In terms of Eq. (26), the square of the distance between the samples x_1,x_2 corresponding to α_1,α_2 and the center of the positive hypersphere is

$$E_2=\left\|\varphi(x_2)-a_1\right\|^2, E_1=\left\|\varphi(x_1)-a_1\right\|^2 \tag{35}$$

Then the difference between them is

$$\begin{aligned}E_2-E_1&=\left\|\varphi(x_2)-a_1\right\|^2-\left\|\varphi(x_1)-a_1\right\|^2\\ &=\frac{2(1-C_1)}{(2-C_1)C_1}\left((K_{12}-K_{22})\alpha_2^{old}+(K_{11}-K_{12})\alpha_1^{old}+v_1-v_2\right)\\ &+\frac{4}{(2-C_1)C_1}(u_1-u_2)-K_{11}+K_{22}\end{aligned} \tag{36}$$

Substituting (36) into (35), we obtain

$$\begin{aligned}&\frac{2(1-C_1)}{(2-C_1)C_1}(K_{11}+K_{22}-2K_{12})\alpha_2\\ &=\frac{2(1-C_1)}{(2-C_1)C_1}(K_{11}+K_{22}-2K_{12})\alpha_2^{old}+E_2-E_1\end{aligned} \tag{37}$$

Thus, the optimization solution of objective function without inequality constraint conditions in (31) is

$$\begin{aligned}\alpha_2^{new,unc}&=\alpha_2^{old}+\frac{E_2-E_1}{\eta}\\ \alpha_1^{new,unc}&=\alpha_1^{old}+\alpha_2^{old}-\alpha_2^{new,unc}\end{aligned} \tag{38}$$

where $\eta=\frac{2(1-C_1)}{(2-C_1)C_1}(K_{11}+K_{22}-2K_{12})$.

It is worth mentioning that $C_2\geq\alpha_i\geq 0, i=1,\ldots,l_+$ are not considered in (31). Taking into account the constraint conditions

$$\begin{aligned}&\begin{cases}0\leq\alpha_2^{new,unc}\leq C_2\\ 0\leq\alpha_1^{new,unc}\leq C_2\Leftrightarrow 0\leq\alpha_1^{old}+\alpha_2^{old}-\alpha_2^{new,unc}\leq C_2\end{cases}\\ \Rightarrow&\begin{cases}0\leq\alpha_2^{new,unc}\leq C_2\\ \alpha_1^{old}+\alpha_2^{old}-C_2\leq\alpha_2^{new,unc}\leq\alpha_1^{old}+\alpha_2^{old}\end{cases}\end{aligned} \tag{39}$$

Let

$$\begin{aligned}L&=\max\left\{0,\alpha_1^{old}+\alpha_2^{old}-C_2\right\}\\ H&=\min\left\{C_2,\alpha_1^{old}+\alpha_2^{old}\right\}\end{aligned} \tag{40}$$

The final solution in (31) is

$$\begin{cases}\alpha_2^{new}=\begin{cases}L & \alpha_2^{new,unc}<L\\ \alpha_2^{new,unc} & L\leq\alpha_2^{new,unc}\leq H\\ H & \alpha_2^{new,unc}>H\end{cases}\\ \alpha_1^{new}=\alpha_1^{old}+\alpha_2^{old}-\alpha_2^{new}\end{cases} \tag{41}$$

Secondly, if β_1,β_2 are chosen from the second equality constraints in (28), the same operation will be performed. In this case, the new iterative solutions are $\beta_1^{new},\beta_2^{new}$.

4.2. Choosing multipliers

At least one of two Lagrange multipliers are altered at every step. We select the Lagrange multipliers which violate the KKT conditions before the step. Each step will decrease the objective function according to Osuna's theorem (Yuan and Xu, 2021). Thus, convergence can be guaranteed. To ensure the convergence, we adopt the heuristics to choose which two Lagrange multipliers to jointly optimize.

Theorem 3. *As for hyper-sphere SVM, $\forall i\in I_+$ the KKT condition satisfies*

(1) If $0<\alpha_i<C_2$, then $\left\|\varphi(x_i)-a_1\right\|^2=R_1^2$.

(2) If $\alpha_i=0$, then $\left\|\varphi(x_i)-a_1\right\|^2\leq R_1^2$.

(3) If $\alpha_i=C_2$, then $\left\|\varphi(x_i)-a_1\right\|^2\geq R_1^2$.

In the same way, $\forall j\in I_-$, the KKT condition satisfies

(4) If $0<\beta_j<C_2$, then $\left\|\varphi(x_j)-a_2\right\|^2=R_2^2$.

(5) If $\beta_j=0$, then $\left\|\varphi(x_j)-a_2\right\|^2\leq R_2^2$.

(6) If $\beta_j=C_2$, then $\left\|\varphi(x_j)-a_2\right\|^2\geq R_2^2$.

The loop iterates over the entire training data set, determining whether each example violates the following KKT condition. If there is an example that violates the KKT conditions, Lagrange multiplier corresponding to the largest deviation is selected, which is may be the one in negative class (noted as β_1), also may be the one in positive class (noted as α_1).

Once a first Lagrange multiplier is chosen, SMO chooses the second Lagrange multiplier to maximize the size of the step taken in the same class during joint optimization. Evaluating the kernel function K is time-consuming, so the step size is approximated by the absolute value of the numerator in Eq. (38): $|E_2-E_1|$ when α_1 had been selected as the first multiplier. In this case, the second multiplier is recorded as α_2. Otherwise, it is recorded as β_2.

4.3. Updating a_1 (or a_2), E_1, E_2

a_1 (or a_2), E_1, E_2 are recomputed by formula (26) and (35) after each step so that the KKT conditions are fulfilled for both example points. Taking α_1,α_2 that are selected in positive class as an example, Table 1 shows the iterative process of our effective algorithm.

For our algorithm, we analyze its time complexity in the following. When searching for the first multiplier, we need to search the points on the boundary and within the boundary respectively. Then the time complexity of searching for the first multiplier is $f_1=O\left(\frac{1}{2}l^2\right)$. Searching for the second multiplier needs to traverse all same class samples, so its time complexity is $f_2=O\left(\frac{1}{4}l^2\right)$, and $f_1+f_2=O\left(\frac{3}{4}l^2\right)$ is the time complexity of a pair of multiplier optimization. If there are t iterations in the optimization process and the algorithm converges, the total time complexity is $f=O\left(\frac{3}{4}l^2t\right)$.

5. Relationship with other related approaches

There have already been some related classification learning algorithms. In this section, we will discuss the connection and difference between our proposed method and other existing related methods.

5.1. Relationship with classical SVM

As we have mentioned in Section 2, the classical is SVM classifier hyper-plane, whereas our MMHS-SVM classifier are hyper-spheres. The common points of them are that (1) The basic paradigm of SVM and

Table 1
The iterative process of our SMO-typed algorithm.

Input: Training dataset X, class labels of training data Y; A group of parameters C_1, C_2, σ, a threshold ε, the number of iterative times N_1; The number of positive samples l_+, the number of negative samples l_-.

Initialization: $\alpha_1^{old} = \alpha_2^{old} = \cdots = \alpha_{l_+}^{old} = \frac{C_2}{l_+}, \beta_1^{old} = \beta_2^{old} = \cdots = \beta_{l_-}^{old} = \frac{C_2}{l_-}, t = 1$;

Step1 For all $i \in I_+$,

If α_i^{old} violates KKT condition in Theorem 3.

$\alpha_1 = \alpha_i^{old}$;

Else

$\alpha_1 = \alpha_i^{old}$ that $0 < \alpha_i^{old} < C_2$;

End

For all $t \in I_+$,

$E_j^{old} = \arg\max_{t \in I_+} \left| E_t^{old} - E_i^{old} \right|$;

$\alpha_2 = \alpha_j^{old}$;

Step 2 Compute α_1, α_2 by equation (38);

Step 3 If $\left|\alpha_1 - \alpha_i^{old}\right| + \left|\alpha_2 - \alpha_j^{old}\right| > \varepsilon$ or $t \le N_1$

Update $a_1, E_i, i = 1, 2$ by equations (24) and (35);

$\alpha_i^{old} = \alpha_1, \alpha_j^{old} = \alpha_2$;

$t = t + 1$;

Go to **Step 1**;

Else

Exit.

End

Output: the solution $\alpha^* = \left(\alpha_1^{old}, \alpha_2^{old}, \cdots, \alpha_{l_+}^{old}\right)^T$, $\beta^* = \left(\beta_1^{old}, \beta_2^{old}, \cdots, \beta_{l_-}^{old}\right)^T$

MMHS-SVM is "structural risk+ empirical risk". Namely, they all have the idea of maximum margin; (2) they can use the kernel trick to solve nonlinear separable problems; (3) only the support vector determines the final classifier, which is sparse. Namely, our hyper-sphere inherits all the advantages of hyper-plane SVM. Besides, another merit of our MMHS-SVM is that two hyper-spheres capture the orientation information and the distribution of samples. MMHS-SVM combines the characteristics of data structure to learn two hyper-spheres, instead of simply finding a hyper-plane to separate the space into two parts. It is reasonable for some practical problems. For example, it is necessary to consider the data structure of independent and identically distributed (*i.i.d.*) data structure to improve the classification performance. Noting that many SVM learning algorithms can be easily extended to our MMHS-SVM. For example, our MMHS-SVM can be easily extended to multi-class classification problem.

5.2. Relationship with THSVM

Both THSVM and MMHS-SVM need to find two hyper-spheres classifier. However, THSVM maximizes the sum of squares of distances between the center of hyper-sphere and examples of the opposite class, but our MMHS-SVM only maximizes the square of the difference between two centers. In other words, our model uses the term $-\|a_1 - a_2\|^2$ instead of the term $\sum_{j \in I_-} \left\|\varphi(x_j) - a_1\right\|^2$ in THSVM to make the distance between two hyper-spheres as large as possible, which implies that our model follows the maximum margin principle. In addition, it costs less time to compute term $-\|a_1 - a_2\|^2$ than $\sum_{j \in I_-} \left\|\varphi(x_j) - a_1\right\|^2$. It means that our MMHS-SVM pays more attention to the overall margin which can avoid the occurrence of over fitting. On the other hand, the test cost of THSVM is larger than the classical SVM and our MMHS-SVM since the decision function of THSVM loses sparsity.

We should point out that THSVM gives rise to two smaller-sized optimization problems, each one is of roughly half size ($l/2$), where l denotes the total size of training data. Hence, the ratio of run times is approximately $O\left(2(l/2)^3\right) = O\left(l^3/4\right)$ under the assumption that the patterns in two classes are approximately equal, 4 times faster than the classical SVM and MMHS-SVM. However, if the SMO algorithm is performed in SVM and the SMO-typed algorithm is performed in our MMHS-SVM, the time complexity of SVM and MMHS-SVM are $O\left(l^3\right)$ and $O\left(\frac{3}{4}l^2 t\right)$ respectively. Besides, due to the lack of sparsity, it consumes more time than our MMHS-SVM in the testing phase.

5.3. Relationship with SSLM, MVE-SSR-SSLM and Ramp-SSLM

Although all of MMHS-SVM, SSLM, MVE-SSR-SSLM and Ramp-SSLM establish an optimization model based on the SRM principle and own sparsity, the distributions of two classes can been characterized by our MMHS-SVM respectively and the structure information of only one class is described by other approaches. That is to say, our MMHS-SVM obtains two different hyper-spheres, but two homocentric spheres are generated in SSLM, MVE-SSR-SSLM and Ramp-SSLM. The margin of them is maximized and the volume of the small sphere is minimized. But our MMHS-SVM only maximizes the square of the difference between two centers. It means that our MMHS-SVM pays more attention to the overall margin which can avoid the occurrence of over fitting.

In comparison, MMHS-SVM and SSLM only need to compute a quadratic programming problem. The time complexity of solving the

Table 2
Theoretical analysis of SVM-based approaches with their properties.

	Classifiers characteristic	SRM	# Optimization model	Time complexity	Sparsity	Class distribution information	Anti-noise ability
MMHS-SVM	Two different hyper-spheres	√	1	$O\left(\frac{3}{4}l^2t\right)$	√	√	√
SVM (1995)	Parallel hyper-plane	√	1	$O\left(l^3\right)$	√	×	×
OCSVM (2001)	A hyper-sphere	×	1	$O\left(l^3\right)$	×	One class	×
TWSVM (2007)	Non-parallel hyper-planes	×	2	$O\left(\frac{1}{4}l^3\right)$	×	×	×
THSVM (2013)	Two different hyper-spheres	×	2	$O\left(\frac{1}{4}l^3\right)$	×	√	×
SSLM (2009)	Homocentric spheres	√	1	$O\left(l^3\right)$	√	One class	×
MVE-SSR-SSLM (2019)	Homocentric spheres	√	1	$O\left(l^2+l_s^3\right)$	√	One class	×
Ramp-SSLM (2022)	Homocentric spheres	√	1	$O(kl^4)$	√	One class	√
MMTSSVM (2016)	Homocentric spheres	√	2	$O\left(\frac{1}{2}l\log\left(\frac{l}{2}\right)+\frac{1}{8}l^3\right)$	×	One class	×
Pin-MMTSM (2018)	Homocentric spheres	√	2	$O\left(\frac{1}{2}l\log\left(\frac{l}{2}\right)+\frac{1}{8}l^3\right)$	×	One class	√
SSR-DM-Pin-MMTSM (2021)	Homocentric spheres	√	2	$O\left(ln+\frac{1}{2}ql_sn+\frac{1}{2}l_s\log\left(\frac{l_s}{2}\right)\right)$	×	One class	√
Pin-M^{3}HM (2016)	Two different hyper-spheres	√	2	$O\left(\frac{1}{4}l^3\right)$	×	One class	√
KNN-M^{3}VHM (2019)	Two different hyper-spheres	√	2	$O\left(l^2\log(l)+\frac{l^3}{4}\right)$	×	One class	√

dual QPP in SSLM is $O(l^3)$. To reduce the expensive time cost, a safe screening rule is constructed in SSLM (MVE-SSR-SSLM) by integrating the ν-property, KKT conditions and variational inequalities. Before solving the dual QPP, the MVE-SSR step is conducted to reduce the scale of problem. The distance terms of all the training samples need to be computed. The time complexity of distance computation of each sample is $O(l)$. Since the distance computation is conducted l times, the computational complexity of MVE-SSR step is $O(l^2)$. Suppose that there is l_s remained training samples after applying MVE-SSR. Then the computational complexity of each MVE-SSR-SSLM is $O\left(l^2+l_s^3\right)$. SSLM is sensible to noises. Ramp-SSLM is more robust. But the computation time of Ramp-SSLM is much longer than that of MMHS-SVM, SSLM and MVE-SSR-SSLM. That is because of the CCCP procedure, which is used to solve Ramp-SSLM. It involves a sequence of convex problems. So, its time complexity is $O(kl^4)$, where k is the iterations of gradient descent. As mentioned above, the time complexity of MMHS-SVM is $O\left(\frac{3}{4}l^2t\right)$.

5.4. Relationship with MMTSSVM, Pin-MMTSM and SSR-DM- Pin-MMTSM

Both MMHS-SVM and MMTSSVM-based approaches (Pin-MMTSM and SSR-DM- Pin-MMTSM) follow the SRM principle. But their measurements of structural risk are different. Minimization of $-\rho^2$ in MMTSSVM-based approaches implies that it maximizes the margin of two homocentric spheres. In our MMHS-SVM, the term $-\|a_1-a_2\|^2$ makes the distance between two hyper-spheres as large as possible. On the other hand, MMTSSVM-based approaches construct two optimization problems to obtain two homocentric with one center. Specially, the small sphere contains as many majority samples (positive class) as possible; on the other hand, most minority samples (negative class) are pushed outside the large sphere. But our MMHS-SVM builds one optimization model to derive two different hyper-spheres according to the data distribution of each class. This means that our MMHS-SVM embodies' all classes distribution and structure information, whereas MMTSSVM merely reflects one class (positive class) distribution which reduces generalization ability. Additionally, our MMHS-SVM can be consistent between the training and testing phase, but MMTSSVM-based approaches cannot. This will lead to poor testing performance.

It should be pointed out that MMTSSVM and Pin-MMTSM solve a smaller-sized QPP and an LPP. So, the time complexity of solving the QPP is $O\left(\frac{l^3}{8}\right)$ and the computational complexity of solving the LPP is $O\left(\frac{l}{2}\log\left(\frac{l}{2}\right)\right)$. The total computational complexity of MMTSSVM is $O\left(\frac{l}{2}\log\left(\frac{l}{2}\right)+\frac{l^3}{8}\right)$ under the assumption that the patterns in two classes are approximately equal. SSR-DM- Pin-MMTSM introduces a safe screening idea into Pin- MMTSM to reduce calculation time. It constructs the region of the center C by variational inequalities and estimates the upper and lower bounds of R by utilizing ν-property. For convenience, we assume that the screening ratios of positive and negative samples are the same, and there are ls remaining training samples. Then the computational complexity of SSR-DM-Pin-MMTSM is $O\left(ln+\frac{1}{2}ql_sn+\frac{1}{2}l_s\log\left(\frac{l_s}{2}\right)\right)$, q denotes the size of the working set which is separated from the remaining training set. As mentioned above, the time complexity of MMHS-SVM is $O\left(\frac{3}{4}l^2t\right)$.

5.5. Relationship with Pin-M^{3}HM and KNN-M^{3}HM

All MMHS-SVM, Pin-M^{3}HM and KNN-M^{3}HM find two hyper-spheres to classify two classes and they all follow the SRM principle by maximizing the distance of two centers. Such an SRM reduces the computational complexity. The difference between them is that KNN-M^{3}HM constructs two models where the first model keeps a hyper-sphere covering as many samples in the first class and requires the other class of samples far away from its center after removing some redundant samples in the first class using the KNN-based strategy, and the second model only calculates the distance of two centers. The pinball loss function is introduced into THSVM (Pin-M^{3}HM) to avoid the noise disturbance. But our MMHS-SVM builds only one optimization to obtain two hyper-spheres simultaneously. Only one optimization ensures that two classes' distribution can be embodied, training and testing phase can be consistent. More important, the sparsity of our MMHS-SVM can reduce the test cost. Besides, as for KNN-M^{3}VHM, the computational

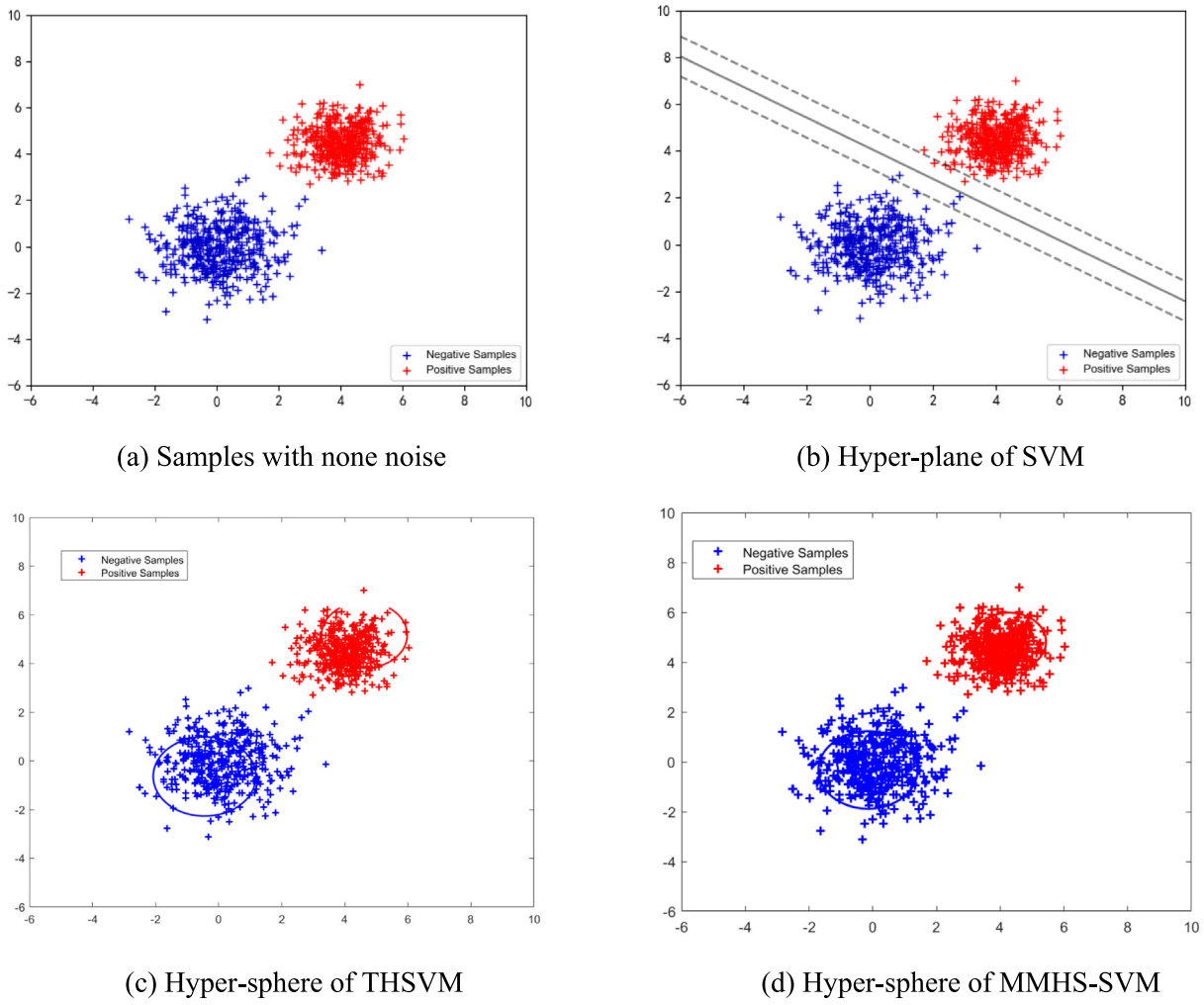


Fig. 1. Classification results of SVM, THSVM and MMHS-SVM on "Gaussian data" with no noise.

complexity for the KNN step is $O\left(l^2\log(l)\right)$. Suppose that the ratio of positive and negative samples (IR) is $r = 1$, then the computational complexity of KNN-M^{3}VHM is $O\left(l^2\log(l) + 2\left(\frac{l}{2}\right)^3\right)$, which is equal to $O\left(l^2\log(l) + \frac{l^3}{4}\right)$. Both THSVM and Pin-M3HM find two hyper-spheres by solving a pair of small-sized QPPs. Thus, the time complexity is $O\left(\frac{1}{4}l^3\right)$.

5.6. Relationship with classical machine learning algorithms

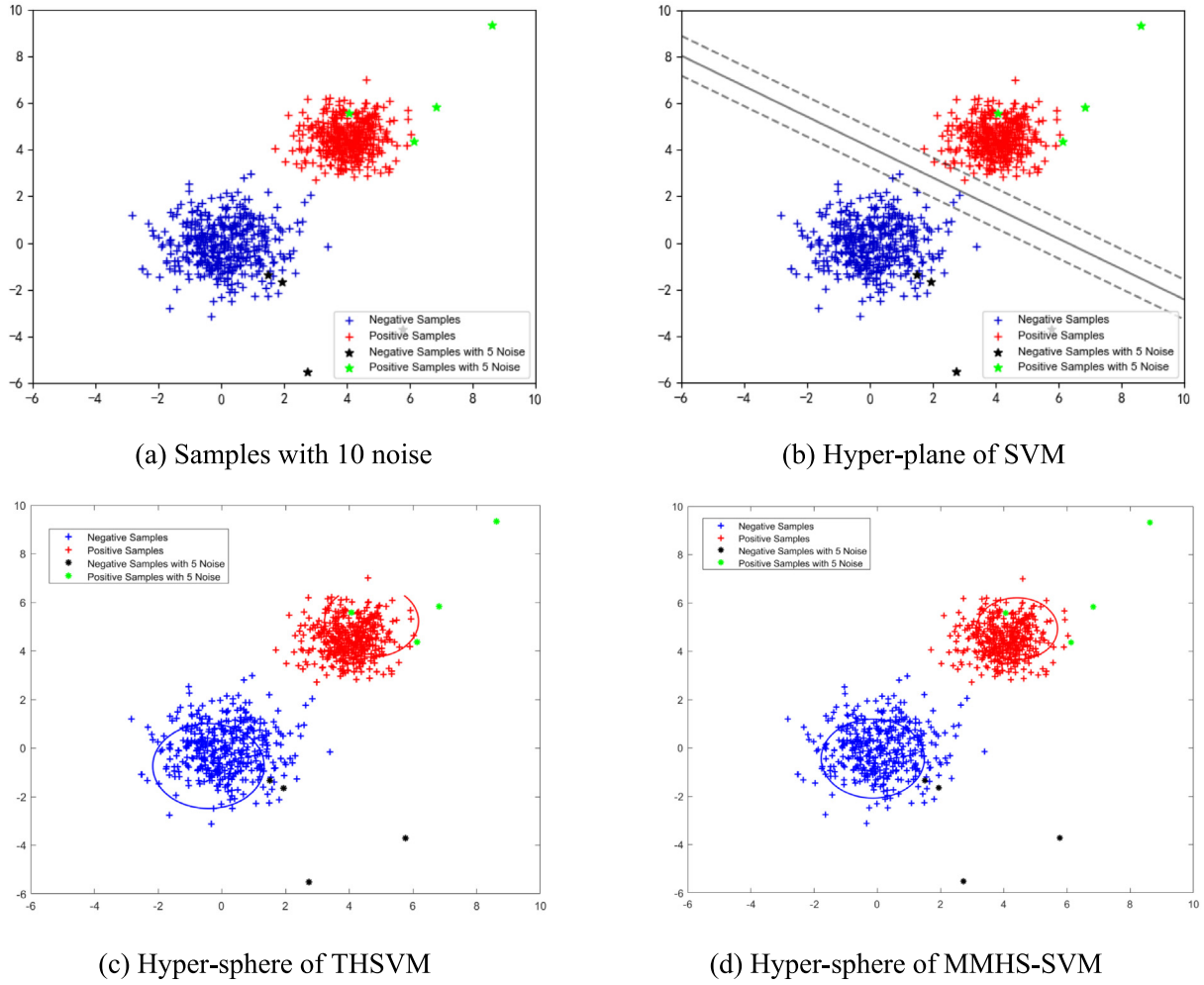
LDA is one of the popular discriminant analysis methods, which is widely applied in many fields. Fisher first proposed LDA to solve the binary classification problem (Fisher, 1936) and then Rao extended it to deal with the multi-class classification problem (Rao, 1948). The objective of LDA is to maximize the ratio of the between-class scatter measure to the within-class one and obtain a subspace by solving the trace ratio problem. Note that, both LDA and our MMHS-SVM attempt to simultaneously guarantee class separation and within-class compactness. However, LDA is a supervised dimensionality reduction method, whereas our MMHS-SVM is a classification method. Besides, LDA may suffer from some difficulties in the process of solving. For examples. the within-class scatter matrix of LDA may be singular and the small sample size problem (SSS) may occur when the dimensionality of data is far larger than the number of training samples. Thus, LDA fails to obtain satisfactory results when the dimensionality of data is high.

Generally, MMHS-SVM is similar to the one-class support vector machine (OCSVM) (Schölkopf et al., 2001), support vector data description (SVDD) (Tax and Duin, 2004) and their expansions (Zhong et al., 2022; Y.P. et al., 2021; Guo et al., 2021) by using a hypersphere to estimate the data distribution. However, such methods do not reflect the maximal margin principle, so they are limited to classification. For all that, they are the pioneers of the idea of hyper-sphere, which is of great significance.

Due to limited space, it is not possible to compare MMHS-SVM with all methods, so Table 2 gives the theoretical analysis of some related methods with some important properties. Table 2 contains 12 related SVM-based methods and our MMHS-SVM with 7 properties, which are classification characteristic, SRM, the number of optimization models, time complexity, class distribution information and noise sensitivity respectively. From Table 2, we find that our MMHS-SVM owns SRM, sparsity, class distribution information and anti-noise ability by constructing only one optimization. Its time complexity is also ok after designing the SMO-typed algorithm. In other words, our approach has more prominent advantages compared with other methods.

6. Experiments

In this section, we run a series of experiments systematically on binary classification datasets, including synthetic datasets and some publicly available datasets. In Section 6.1, four groups of synthetic datasets are shown to intuitively compare MMHS-SVM with THSVM



(a) Samples with 10 noise

(b) Hyper-plane of SVM

(c) Hyper-sphere of THSVM

(d) Hyper-sphere of MMHS-SVM

Fig. 2. Classification results in "Gaussian data" with 5 noise points for each class on SVM, THSVM and MMHS-SVM.

and SVM which are the most relevant methods. In Section 6.2, we demonstrate the outstanding performance of our proposed MMHS-SVM on 15 benchmark datasets.[1] And seven other algorithms are compared. Large datasets are used to verify the effectiveness of our MMHS-SVM for training time, testing time, sparsity, and classification performance in Section 6.3.

During the experiments, all experiments are implemented on a PC with Intel Core (TM) core i5 CPU (2.4 GHz) with 2 GB RAMS. If there are no other special instructions, we use 70% of the datasets to train and 30% to test. The average accuracy (ACC) and standard deviation (STD) are calculated to measure classification performance. For the nonlinear case, these problems are tested using RBF kernel as it is often employed and yields great generalization performance. The parameters in classifiers are obtained using the grid search method. In seven algorithms, the kernel parameter σ is selected from the set $\{2^i \mid i = -10, -9, \ldots, 9, 10\}$. For the experiment on each dataset, we use 5-fold cross validation to evaluate the performance of eight algorithms. That is to say, each dataset is split randomly into five subsets, and one of those sets is reserved as a test set whereas the remaining data are considered for training; This process is repeated five times. Due to experimental parameters that may influence the identification performance, to obtain the best generalization performance, C_2 is selected in the range of $\{2^{-3}, 2^{-2}, \ldots, 2^{10}\}$, and C_1 is in the range of $\{2^{-3}, 2^{-2}, \ldots, 2^{0}, 2^{1}, \ldots, 2^{9}\}$ and must satisfy Theorems 1 and 2. We use the LIBSVM (Lin, 2016) tool is used to build a classifier for SVM-based algorithm on both classification datasets.

Table 3
Accuracy (%), SVs and testing CPU time on four "Gaussian" datasets.

Datasets	Results	SVM	THSVM	MMHS-SVM
None noise	Accuracy (%)	100	100	**100**
	SVs	2	800	12
	Testing time	0.1201	0.1095	0.1224
#No. 10-noise	Accuracy (%)	100	100	**100**
	SVs	2	810	125
	Testing time	0.1072	0.1153	0.1032
#No. 20-noise	Accuracy (%)	99.95	100	**100**
	SVs	5	820	11
	Testing time	0.1183	0.1113	0.1057
#No. 40-noise	Accuracy (%)	99.9	99.95	**100**
	SVs	3	840	10
	Testing time	0.1074	0.1199	0.1062

[1] https://archive.ics.uci.edu/ml/index.php.

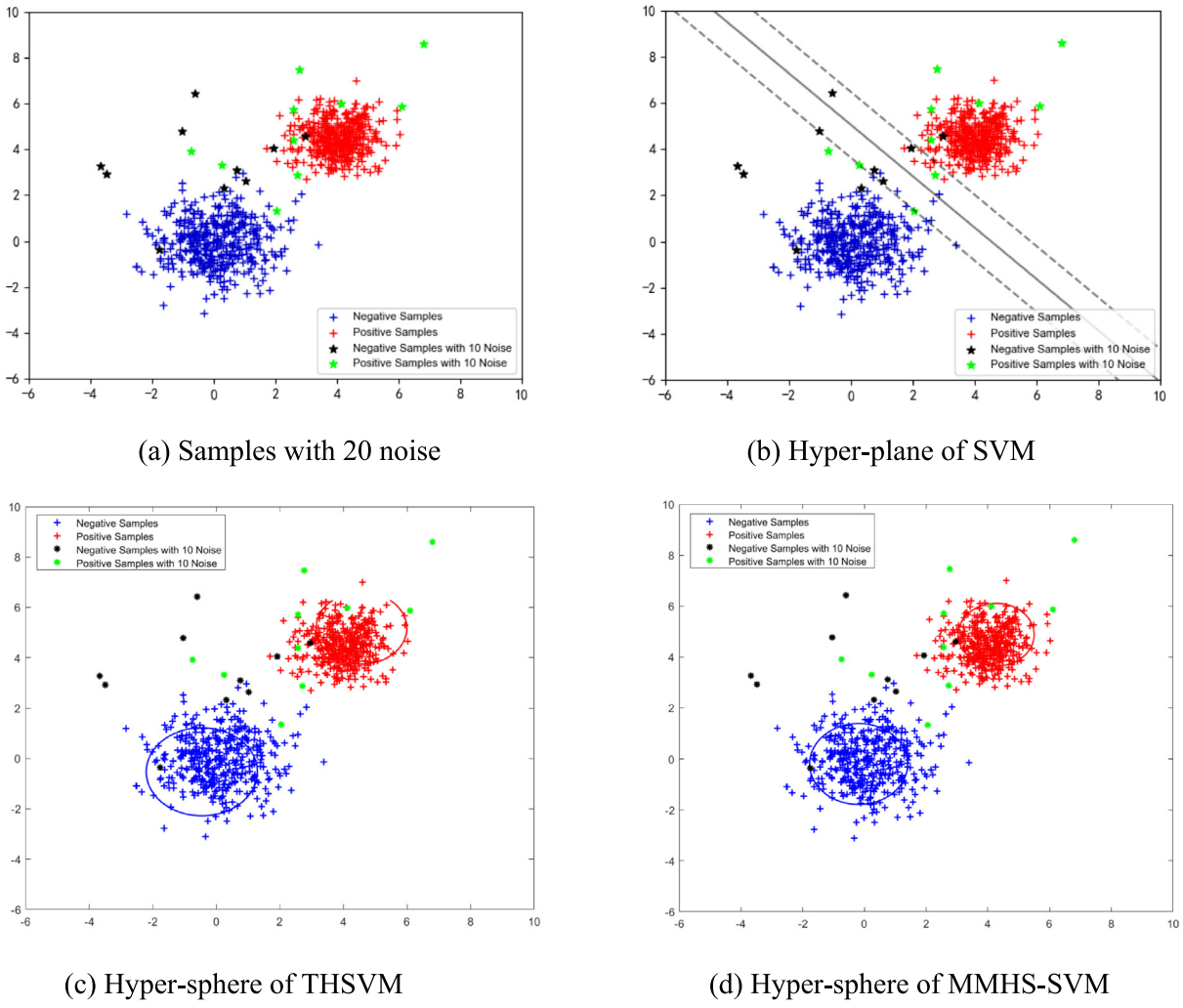
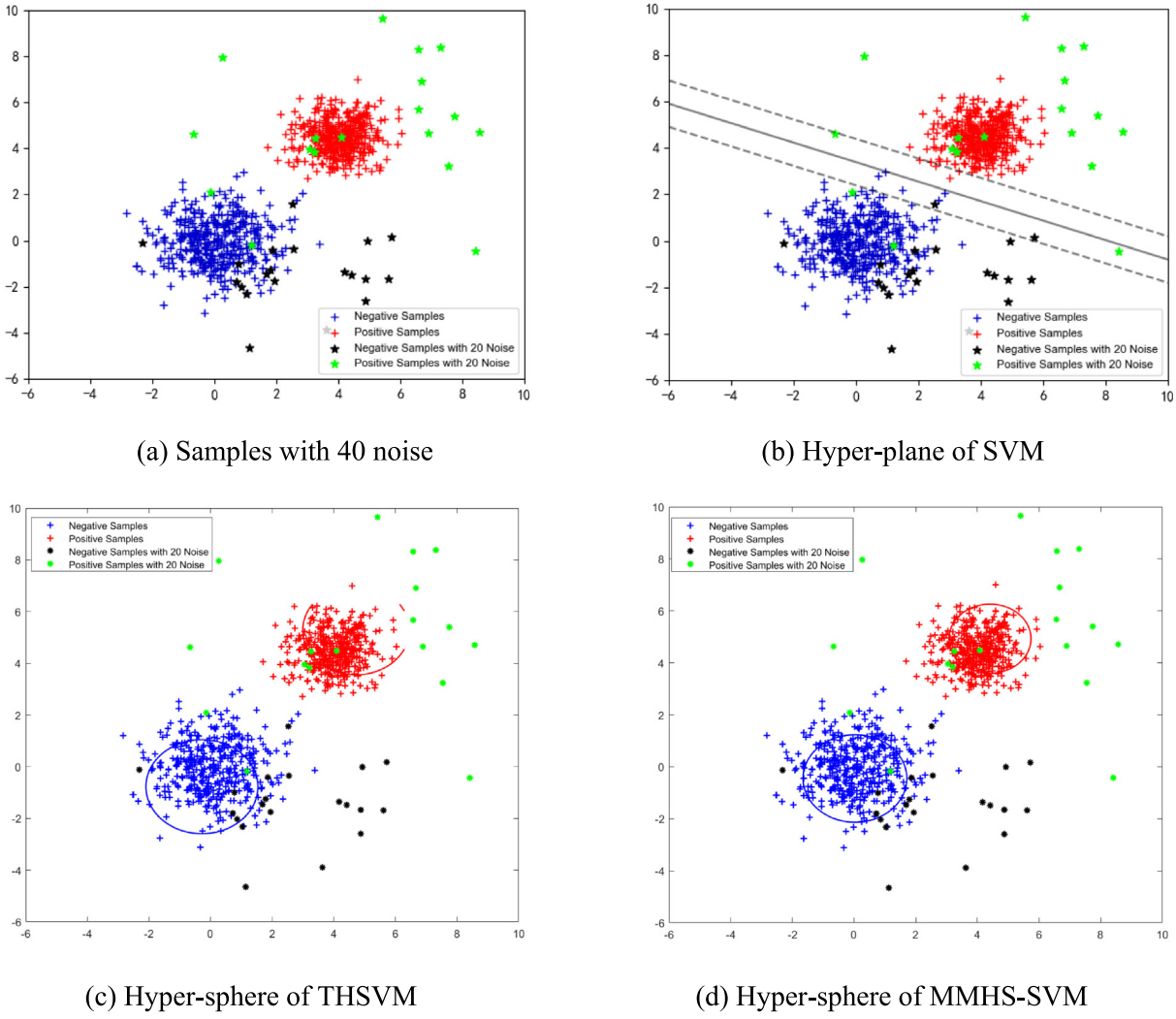


Fig. 3. Classification results in "Gaussian data" with 10 noise points for each class on SVM, THSVM and MMHS-SVM.

6.1. Artificial datasets

In this section, to illustrate graphically the effectiveness of our MMHS-SVM in considering data structure, the artificial datasets with two classes are generated randomly by two Gaussian distributions. The two classes of training samples consist of 400 positive points and 400 negative points, where "red +" and "blue +" represent positive and negative samples respectively. And the test dataset containing 1000 points for each class are generated by the Gaussian distributions randomly. To reflect the sensitivity of noise to classification results, we add 5, 10, and 20 noise samples for each class respectively where positive noise points are depicted as "green star" and negative noise points are "black star". Figs. 1–4(a) shows the generated two-dimensional data with no noise, 10 noise, 20 noise and 40 noise respectively. The classification hyper-plane and hyper-sphere are plotted in Figs. 1–4(b–d). From Fig. 1(b), we can see that the hyper-plane in SVM locates roughly the midway between the SVs of the two classes. Thus, it successfully classifies the training samples. However, this separating hyperplane ignores the data scattering, which leads it to be possibly inappropriate for this example. In contrast, the two circles of the linear THSVM and our MMHS-SVM capture the orientation information of training samples and effectively cover the positive and negative samples, which makes the separating hypersphere more natural than SVM. After adding the noise, the classification hyper-plane of SVM and hyper-sphere of THSVM have changes, whereas little has altered in our MMHS-SVM, as shown in Figs. 2–4(b–c). Besides, from Figs. 2–4, we find that with the increase of noise, the hyper-sphere of MMHS-SVM deviates from its own data distribution structure gradually. This trend shows MMHS-SVM is sensitive to noise and it has more strong robustness, which further supports the theory in Section 3.

Table 3 lists the test accuracy, testing CPU time and SVs of the three methods on raw Gaussian data and three data with noise. It can be found that the test accuracy of THSVM and SVM is lower than that of our MMHS-SVM when the noise points increase for this example. More concretely, the prediction precision of MMHS-SVM is 100% whatever the ratio of noise is. This indicates that our MMHS-SVM can not only depict the characteristics of samples, but also can resist the influence of noise. Besides, to further demonstrate the sparsity and lower test cost of our MMHS-SVM, Table 3 lists the testing time on four datasets. Due to the sparse solution of our MMHS-SVM and SVM, their test time is less than THSVM. Next, we investigate the samples that play a decisive role in classification in SVM, THSVM and MMHS-SVM. That is to say, we record the numbers of support vectors of SVM, THSVM and MMHS-SVM on four datasets in Table 3. It is not hard to find that THSVM always uses all the samples to learn the classifier. Whereas, only a small part of points determines the final classifier in SVM and MMHS-SVM, which proves once again that they only need to use the data with outstanding characteristics to learn a classifier and abandon the redundant data.



(a) Samples with 40 noise

(b) Hyper-plane of SVM

(c) Hyper-sphere of THSVM

(d) Hyper-sphere of MMHS-SVM

Fig. 4. Classification results in "Gaussian data" with 20 noise points for each class on SVM, THSVM and MMHS-SVM.

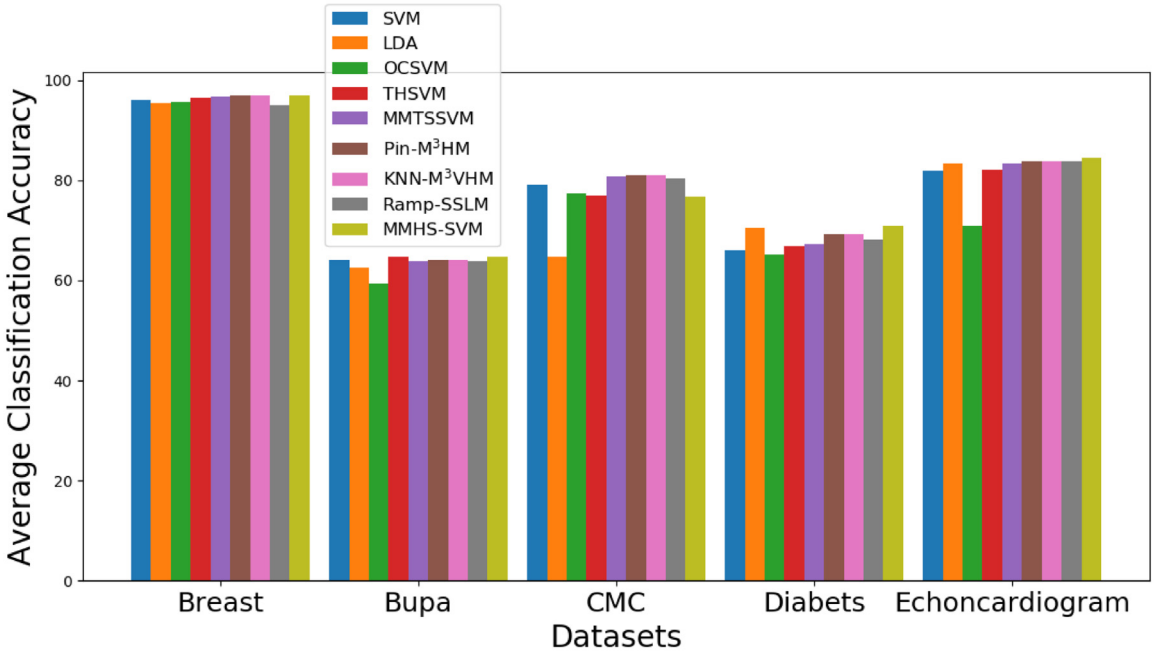
Table 4
The detailed descriptions of 15 benchmark datasets.

Datasets	#No. train samples	#No. test samples	# Dimension
Breast	479	204	9
Bupa	242	103	6
CMC	1032	441	9
Diabets	538	230	8
Echoncardiogram	93	38	10
Haberman	215	91	3
Heart_disease	207	87	14
Heart-statlog	189	81	13
Hepatitis	110	45	19
Ionosphere	200	151	34
PimaIndian	538	230	8
Seeds	147	63	7
Sonar	146	62	60
Spect	188	79	44
Wdbc	400	169	30

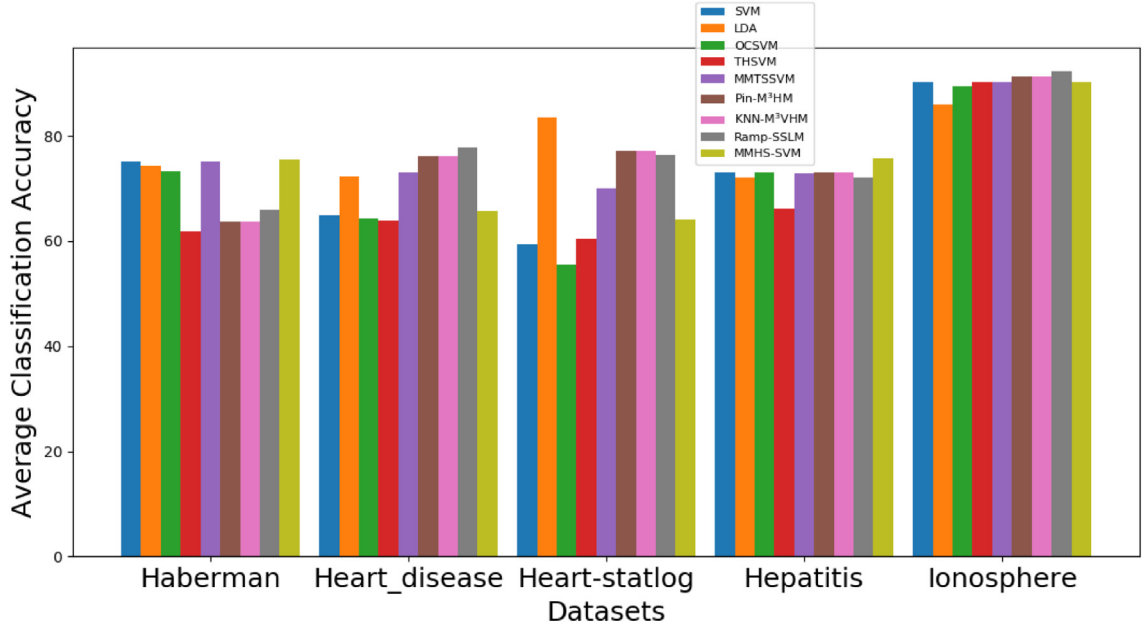
Table 4 (*continued*).

6.2. Benchmark datasets

In this subsection, we widely choose various SVM-based algorithms, including parallel hyper-plane SVM (SVM), non-parallel hyper-plane SVM (TWSVM), one class hyper-sphere (OCSVM), two different hyper-spheres SVM (THSVM, Pin-M^{3}HM, KNN-M^{3}VHM), representative homocentric spheres (MMTSSVM, Ramp-SSLM) to compare with our MMHS-SVM on 15 publicly available benchmark datasets from the UCI Repository. The statistics of them are shown in Table 4, including the number of train samples and test samples and their dimensions. The optimal values for c and ν are selected from the set $\{2^i \mid i = 0, 1, \ldots, 8\}$ in Pin-M^{3}HM and KNN-M^{3}VHM. For KNN-M^{3}VHM, the optimal values for k in KNN-based strategy are chosen from the set $\{3, 4, \ldots, 8\}$. τ is selected from set $\{0.1, 0.2, \ldots, 0.9\}$ in Pin-M^{3}HM. We set $k_1 = k_2 = k$ in



(a) Average accuracy on Breast, Bupa, CMC, Diabets and Echhoncardiogram datasets



(b) Average accuracy on Haberman, Heart_disease, Heart-statlog, Hepatitis and Ionosphere datasets

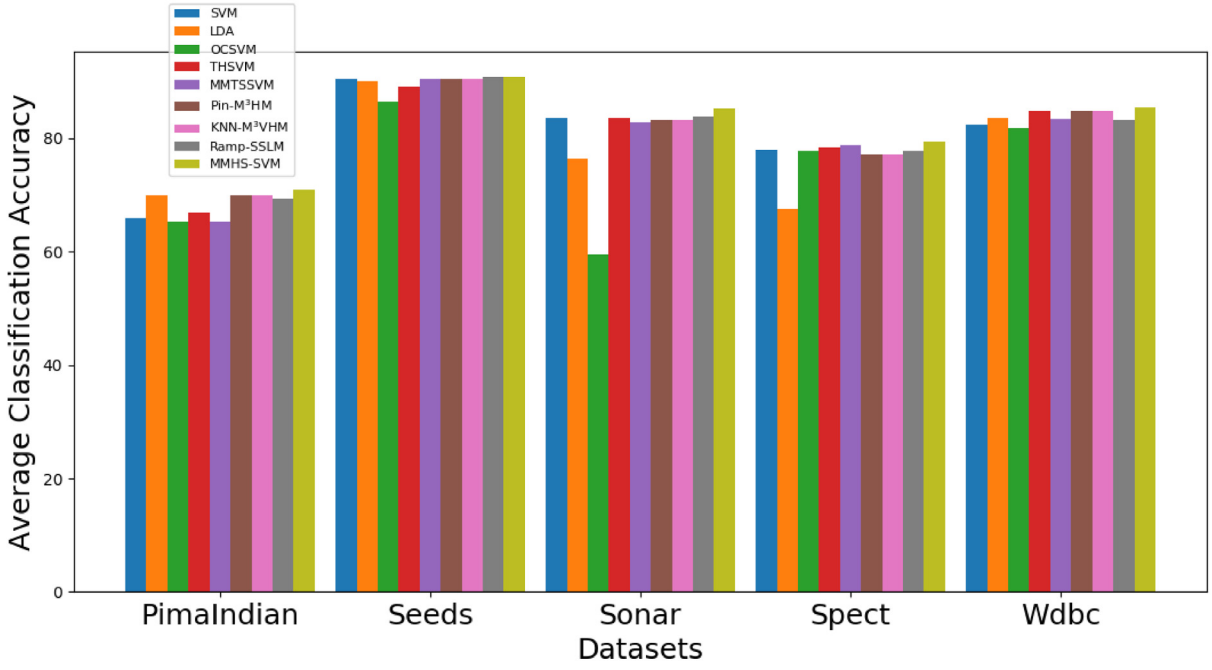
Fig. 5. Average accuracy (%) on 15 Benchmark datasets.

Ramp-SSLM to reduce the computational complexity of the parameter selection, which ranges between {0.1; 0.2; ……; 1}.

6.2.1. Classification performance analysis

Table 5 lists the average accuracy and the standard deviation with 10 independent runs of the nonlinear SVM, LDA, OCSVM, THSVM, MMTSSVM, Pin-M^{3}HM, KNN-M^{3}VHM, Ramp-SSLM and MMHS-SVM with Gaussian kernels. The best result on each dataset is bold. It can be seen that the MMHS-SVM derives the best classification performance on 11 datasets, such as Breast, Bupa, Echoncardiogram Haberman and so on. The datasets of best results account for more than 70% of the total datasets. The reason is that our approach depicts the data structure, owns sparsity and generalization capacity because of establishing that novel "maximal margin" model. On the other hand, the performance of KNN-M^{3}VHM and Ramp-SSLM are the best on Ionosphere datasets. It means that they possess almost the same effect.

To more intuitively compare the performance of various algorithms on fifteen datasets, Fig. 5(a–c) shows three histograms to compare their average accuracies. In Fig. 5, the yellow bar represents the average accuracy of our approach. From Fig. 5, we can observe that the yellow bars are always the highest compared with other colors in most datasets. This result is consistent with Table 5.



(c) Average accuracy on PimaIndian, Seeds, Sonar, Spect and Wdbc datasets

Fig. 5. (*continued*).

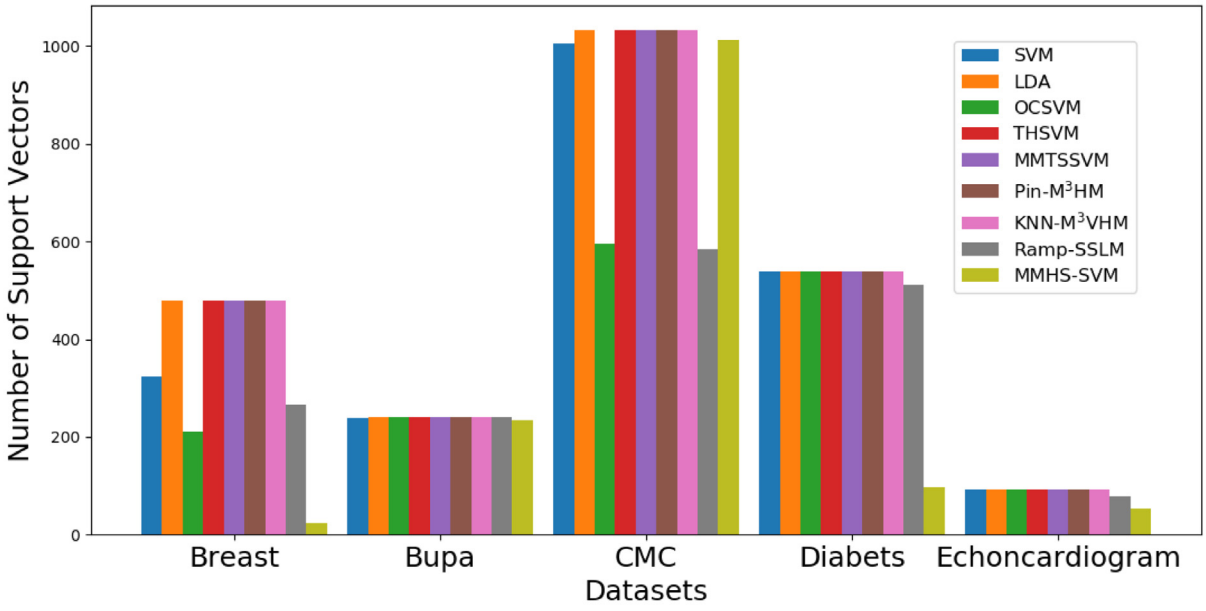
Table 5
Average accuracy (%) and standard deviation on UCI datasets.

Datasets	SVM	LDA	OCSVM	THSVM	MMTSSVM	Pin-M^{3}HM	KNN-M^{3}VHM	Ramp-SSLM	MMHS-SVM
Breast	96.0784 ± 2.0826	95.3431 ± 0.9872	95.6863 ± 0.7234	96.5686 ± 1.2444	96.6483 ± 1.2039	96.8451 ± 1.4060	96.3595 ± 2.6372	95.06 ± 3.05	**96.8627 ± 0.8396**
Bupa	64.0485 ± 3.3302	62.6214 ± 2.9749	59.4175 ± 1.1022	**64.6602 ± 3.1443**	63.9548 ± 2.8902	64.0794 ± 2.8043	63.6310 ± 2.7935	63.8910 ± 3.1084	**64.6602 ± 3.0428**
CMC	79.1837 ± 0.5104	64.6712 ± 1.3649	77.3016 ± 0.9881	76.9615 ± 0.8362	80.8305 ± 0.5972	**81.0639 ± 0.5308**	80.0820 ± 0.7343	80.4922 ± 0.3892	76.8481 ± 0.5065
Diabets	65.913 ± 0.5500	70.6087 ± 1.5716	65.2174 ± 0	66.8667 ± 1.642	67.3010 ± 0.6307	69.3076 ± 1.4809	68.1806 ± 1.0942	68.0724 ± 2.3013	**70.8667 ± 2.4905**
Echoncardiogram	81.8421 ± 3.6061	83.3684 ± 5.6577	71.0526 ± 2.4811	82.1053 ± 6.8847	83.3672 ± 4.9742	83.9396 ± 6.0740	82.4989 ± 5.3978	83.9396 ± 5.2076	**84.4737 ± 5.1821**
Haberman	75.1143 ± 1.0928	74.3956 ± 4.3357	73.2967 ± 1.4698	61.8681 ± 12.2264	75.1048 ± 1.2075	63.7593 ± 3.9648	64.1738 ± 3.5732	65.8301 ± 4.0472	**75.6044 ± 2.5797**
Heart_disease	64.8276 ± 0.6525	72.2989 ± 3.1687	64.3678 ± 0	63.908 ± 4.2734	73.0384 ± 0.3972	76.1093 ± 2.0281	**79.8401 ± 5.4062**	77.7340 ± 4.2018	65.6322 ± 4.2475
Heart-statlog	59.3827 ± 1.4780	**83.4568 ± 4.0822**	55.5556 ± 0	60.3704 ± 4.5991	69.9135 ± 1.5018	77.1753 ± 3.9702	78.6584 ± 4.0694	76.3924 ± 4.8204	64.0741 ± 4.9537
Hepatitis	73 ± 0	72 ± 3.0452	73 ± 0	66.2222 ± 4.8911	72.9042 ± 0.4892	73 ± 0	73 ± 0	72 ± 3.0452	**75.7778 ± 3.2203**
Ionosphere	90.3062 ± 1.5570	85.9615 ± 2.1357	89.4038 ± 3.1223	90.1987 ± 1.3828	90.2084 ± 1.7047	91.2363 ± 3.9426	**92.2356 ± 3.0678**	**92.2356 ± 3.0678**	90.3311 ± 1.3637
PimaIndian	65.9565 ± 0.7404	69.9130 ± 3.2732	65.2174 ± 0	66.8261 ± 5.6319	65.3907 ± 3.8720	69.9635 ± 3.9307	70.8589 ± 5.1263	69.3021 ± 5.8205	**70.9130 ± 6.1229**
Seeds	90.4444 ± 2.7997	90.0159 ± 2.9173	86.5079 ± 2.9221	89.0476 ± 2.8444	90.4803 ± 4.0842	90.5340 ± 3.7076	90.6083 ± 4.0938	90.7937 ± 4.2081	**90.7937 ± 2.8786**
Sonar	83.5484 ± 2.8903	76.4516 ± 4.7604	59.5161 ± 3.6026	83.7097 ± 5.6105	82.7392 ± 3.8579	83.2346 ± 4.3393	84.9518 ± 4.8862	83.7302 ± 4.7201	**85.3226 ± 4.7756**
Spect	78.0468 ± 0	67.5949 ± 5.3438	77.7468 ± 0	78.481 ± 3.4279	78.8403 ± 0.4822	77.1957 ± 1.4802	76.7593 ± 4.5703	77.8205 ± 3.8027	**79.3671 ± 2.8649**
Wdbc	82.3682 ± 1.8642	83.5794 ± 3.6744	81.7783 ± 1.7942	84.7337 ± 1.2636	83.4928 ± 1.0738	84.7409 ± 1.4072	84.9749 ± 1.0793	83.2902 ± 1.0730	**85.5030 ± 1.5028**

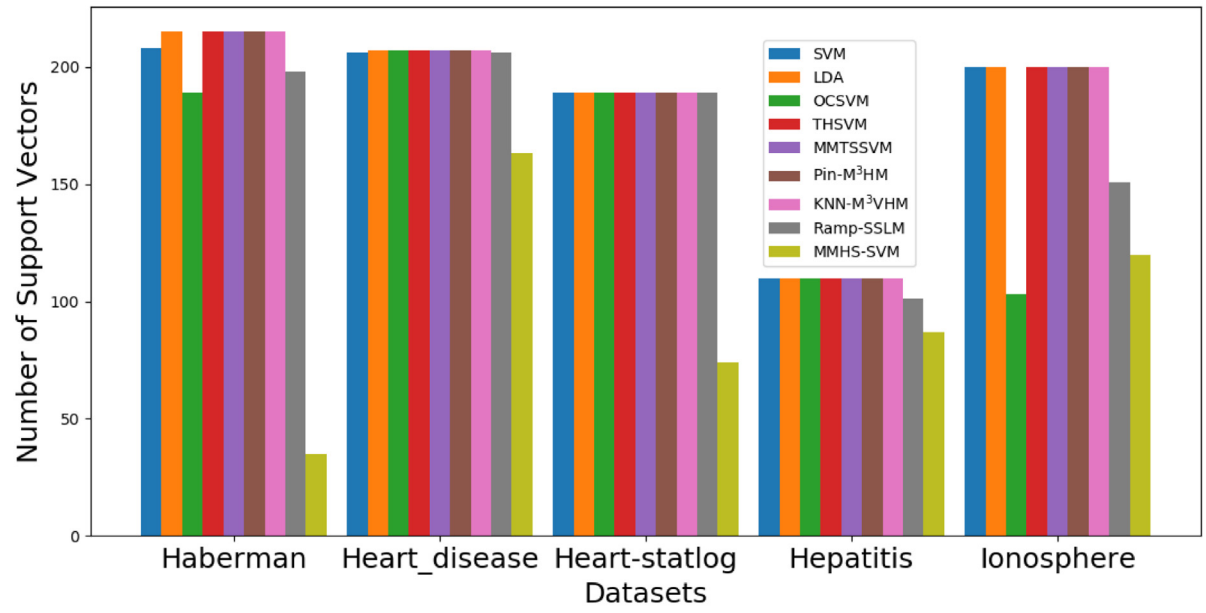
In addition, to reflect the sparse of our method, we give the number of support vectors in Table 6. From Table 6, we find that all of THSVM, MMTSSVM, Pin-M^{3}HM and KNN-M^{3}VHM always uses all training samples to obtain the final classifier, whereas our MMHS-SVM just keeps a small number of critical samples to learn the final classifier, abandoning most redundant data. For example, our MMHS-SVM owns 23 SVs, but THSVM has up to 479. Fewer SVs can reduce the impact of redundant data on classification, and be sparser and faster. Compared with SVM, the SVs of our MMHS-SVM are much less than those of SVM. It should be noted that Ramp-SSLM has the least support vectors on CMC and Wdbc datasets and thus owns better scaling properties. This is consistent with the theory. i.e., ramp loss theoretically guarantees that it is sparser. More intuitively, we also show the histogram to reflect the SVs of nine approaches in Fig. 6. Besides, Table 7 lists the optimal parameters obtained from training on 15 datasets.

6.2.2. Comparison of CPU testing time

Regarding the testing efficiency of these algorithms, Table 8 also shows the testing time of these algorithms on these benchmark datasets. It can be observed that the MMHS-SVM needs the least CPU time on Echoncardiogram, Heart-statlog, Ionosphere, PimaIndian, Seeds, Sonar, Spect and Wdbc datasets (near to zero). When processing large-scale datasets, the speed advantage of our MMHS-SVM is prominent. Two main reasons for MMHS-SVM producing faster testing time are: it operates a fast SMO-typed iterative procedure; it only calculates the distance of two centers, so it is simple and reduces the computational complexity. Further, we find that the LDA is much faster than the rest six algorithms as LDA uses prior knowledge to predict directly Specially, it is worth mentioning that THSVM, MMTSSVM, Pin-M^{3}HM and



(a) The SVs on Breast, Bupa, CMC, Diabets and Echhoncardiogram datasets



(b) The SVs on Haberman, Heart_disease, Heart-statlog, Hepatitis and Ionosphere datasets

Fig. 6. The numbers of support vectors on 15 Benchmark datasets.

KNN-M^{3}VHM need more CPU time because they are not sparse. And KNN-M^{3}VHM need calculates k nearest neighbors. The nonconvexity of Ramp-SSLM means that it is also time-consuming to calculate and store the kernel matrix multiple times. As a result, a longer run-time may be a drawback of Ramp-SSLM. To further show the time effectiveness of our proposed method, we draw a histogram to display the CPU time for these methods in Fig. 7(a–c).

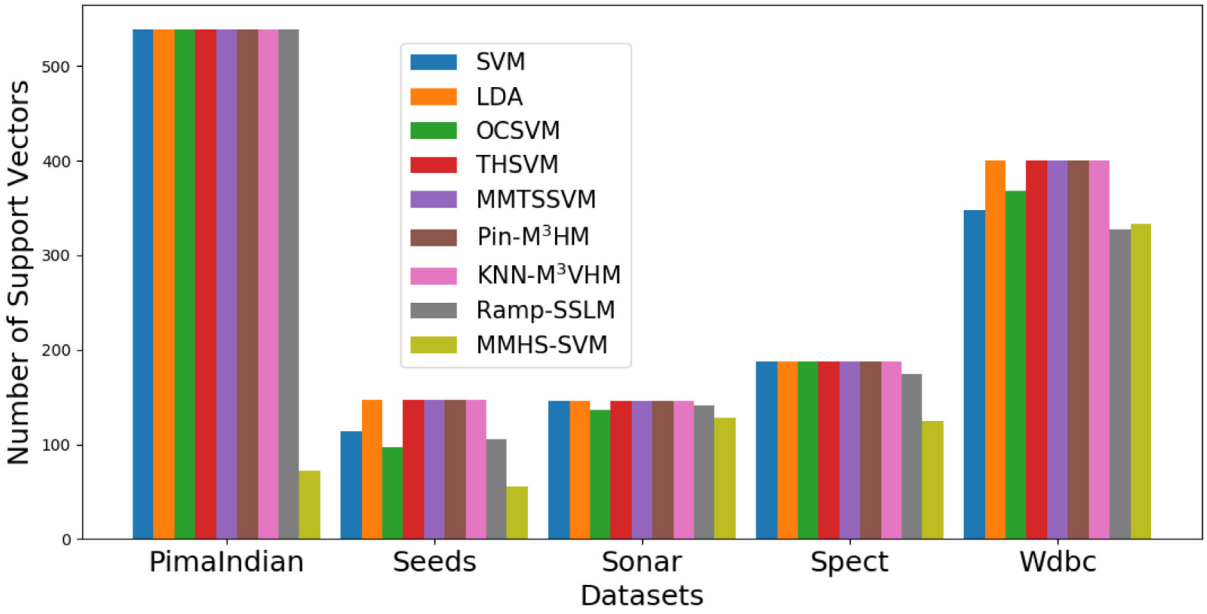
6.2.3. Statistical tests

In this subsection, the classification accuracies of the 9 classifiers are examined by introducing the Friedman test (Friedman, 1997). First, we list the average accuracy ranks on 15 UCI datasets in Table 9. Then, we compute the following statistics of chi square distribution:

$$\chi_F^2 = \frac{12n}{m(m+1)}\left(\sum_j^m R_j^2 - \frac{m(m+1)^2}{4}\right) \tag{42}$$

where $R_j = \frac{1}{n}\sum_{i=1}^n r_i^j, j = 1, 2, \ldots, m$ and r_i^j denotes the average rank of the jth classifier (m classifiers) on the ith dataset (n datasets). Finally, the following Friedman statistic which follows an F-distribution with $m - 1$ and $(m - 1)(n - 1)$ degrees of freedom is obtained:

$$F_F = \frac{(n-1)\chi_F^2}{n(m-1) - \chi_F^2} \tag{43}$$



(c) The SVs on PimaIndian, Seeds, Sonar, Spect and Wdbc datasets

Fig. 6. (*continued*).

Table 6
The number of support vectors on the UCI datasets.

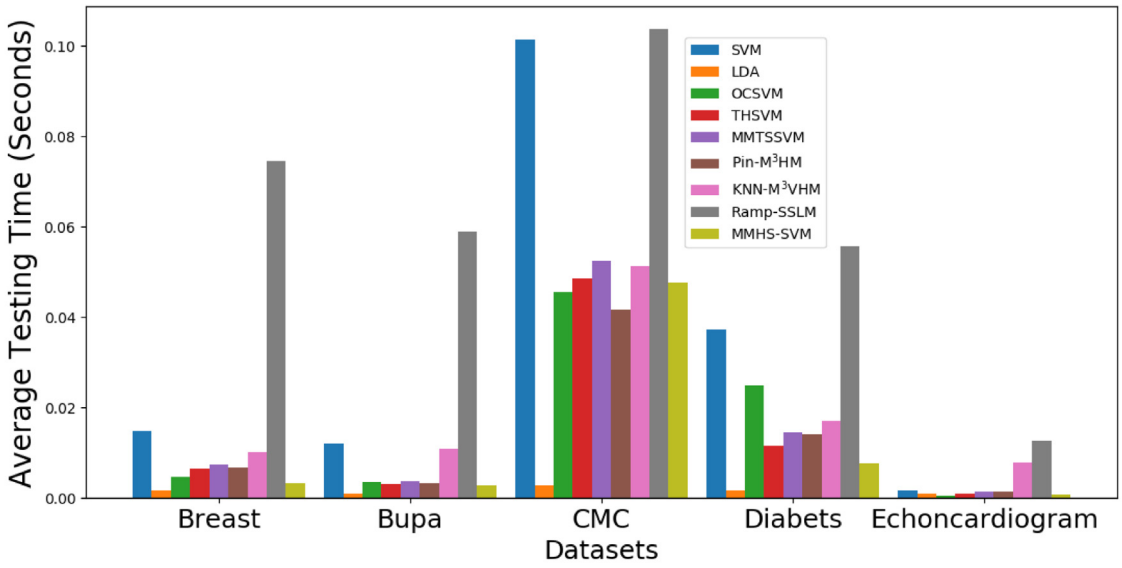
Datasets	SVM	LDA	OSVC	THSVM	Pin-M^{3}HM	KNN-M3VHM	Ramp-SSLM	MMHS-SVM
Breast	323	479	212	479	479	479	267	**23**
Bupa	239	242	241	242	242	242	240	**234**
CMC	1005	1032	595	1032	1032	1032	**584**	1012
Diabets	538	538	538	538	538	538	512	**98**
Echoncardiogram	93	93	93	93	93	93	78	**54**
Haberman	208	215	189	215	215	215	198	**35**
Heart_disease	206	207	207	207	207	207	206	**163**
Heart-statlog	189	189	189	189	189	189	189	**74**
Hepatitis	110	110	110	110	110	110	101	**87**
Ionosphere	200	200	103	200	200	200	151	**120**
PimaIndian	538	538	538	538	538	538	538	**72**
Seeds	114	147	97	147	147	147	105	**55**
Sonar	146	146	137	146	146	146	141	**128**
Spect	188	188	188	188	188	188	175	**124**
Wdbc	347	400	368	400	400	400	**327**	333

According to (42) and (43), we obtain $\chi_F^2 = 42.977$ and $F_F = 7.81$ with (8, 112) degrees of freedom for the F-distribution. From the table of critical values for F-distribution, we obtain that the critical values of F (8, 112) are approximately 2.3073, 2.0221, 13 and 1.7257 for the significance levels of $\alpha = 0.025, \alpha = 0.05, \alpha = 0.1$, respectively. This result illustrates that there is a significant difference between the 9 algorithms since the real value of F_F is much larger than the critical values. Because the average rank of MMH-SVM is the lowest, our proposed algorithm is shown to be the most effective.

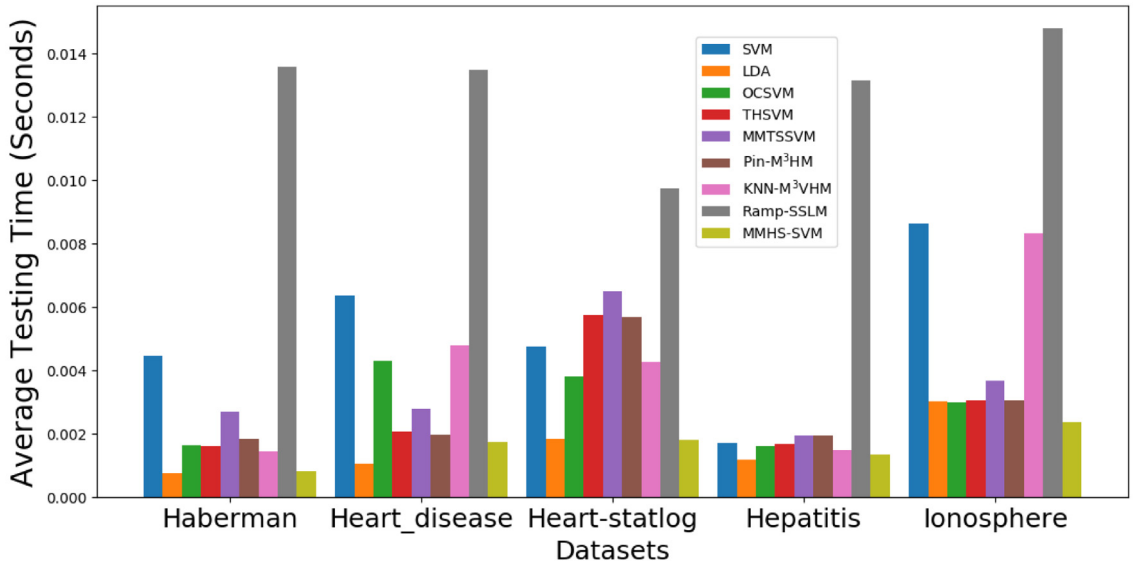
6.3. Large datasets

In this subsection, we select large scale data, digital handwriting recognition datasets, to embody the low operation time and storage space of our SMO-typed algorithm. The USPS database[2] is a raster scan of the 16∗16 grey level pixel intensities handwritten digit images, consisting of 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, as shown in Fig. 8. There are 1100 images for each digit. Three pairs of similar digits, such as digit 1 versus

[2] https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/.



(a) Average testing time on Breast, Bupa, CMC, Diabets and Echoncardiogram datasets



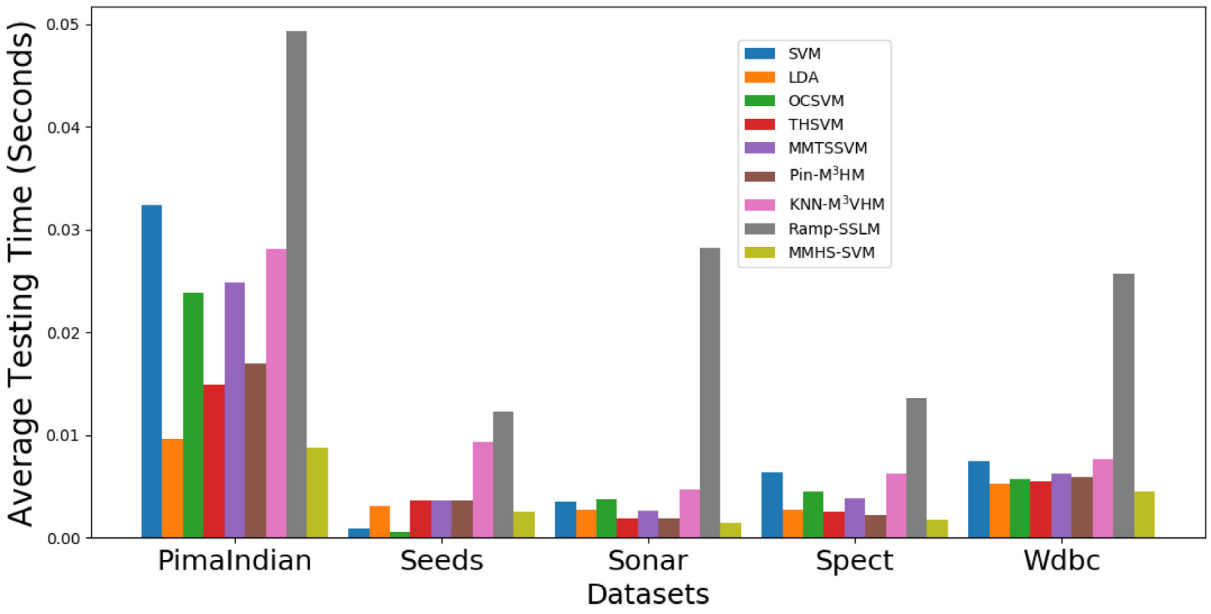
(b) Average testing time on Haberman, Heart_disease, Heart-statlog, Hepatitis and Ionosphere datasets

Fig. 7. The average testing CPU time on 15 Benchmark datasets.

7, 2 versus 5 and 4 versus 6, are carried on the classification. In this subsection, we compare MMHS-SVM with the approaches SVM, LDA, OCSVM, THSVM, MMTSSVM, Pin-M3HM, KNN-M^{3}VHM and Ramp-SSLM. The linear kernel is chosen. Table 10 and Fig. 9(a) illustrate the comparisons of results of digit 1 versus 7, 2 versus 5 and 4 versus 6. We find that the accuracy of our MMHS-SVM is the highest on 1 versus 7 and 4 versus 6 datasets, and is lower than KNN-M^{3}VHM on 2 vs. 5 dataset. In a word, hyper-sphere SVMs surpass other classifiers on large scale data. Besides, we also record the number of support vectors training time and testing time in Fig. (b–d). Fig. 9(c) and (d) list the average central processing unit (CPU) testing time and training time (the time units are seconds). From Fig. 9, MMHS-SVM always consumes the least testing time on handwriting recognition datasets. As the amount of data increases, Ramp-SVM spends more and more testing time. What we have to say is that the average training time of MMTSSVM is prominent and OCSVM follows. It is consistent with the theory analysis, and MMTSSVM only solves the linear equations for one class. Moreover, the sparsity of our method is verified again on large scale data. For 2 vs. 5, # SV of our MMHS-SVM is 30. It means only 30 samples decide the final classifier which can grasp the key samples to learn the decision function.

7. Conclusions

In this paper, we have put forward maximal margin hyper-spheres SVM (MMHS-SVM). Our proposed MMHS-SVM simultaneously determines a pair of hyper-spheres by a single optimization, in which each one contains as many samples as possible and the centers keep away from each other to some extent. The formulation of MMHS-SVM is in the spirit of SVM, thus it follows the SRM, i.e., maximizing the distance between two centers. Such an operation can keep the sparsity, reduce test cost, and improve anti-noise ability. In addition, the application of



(c) Average testing time on PimaIndian, Seeds, Sonar, Spect and Wdbc datasets

Fig. 7. (*continued*).

Fig. 8. Digital Handwritten images on USPS.

Table 7
The optimal parameters on 15 datasets.

Datasets	C_1	C_2	Sigma
Breast	2^5	2^1	2^4
Bupa	2^2	2^{-3}	2^2
CMC	2^{-1}	2^{-2}	2^2
Diabets	2^6	2^{-3}	2^2
Echoncardiogram	2^3	2^0	2^2
Haberman	2^6	2^{-2}	2^2
Heart_disease	2^4	2^4	2^7
Heart-statlog	2^7	2^{-3}	2^2
Hepatitis	2^4	2^3	2^3
Ionosphere	2^0	2^1	2^2
PimaIndian	2^{10}	2^0	2^5
Seeds	2^{-2}	2^3	2^2
Sonar	2^{-1}	2^{-3}	2^2
Spect	2^5	2^{-2}	2^2
Wdbc	2^9	2^1	2^7

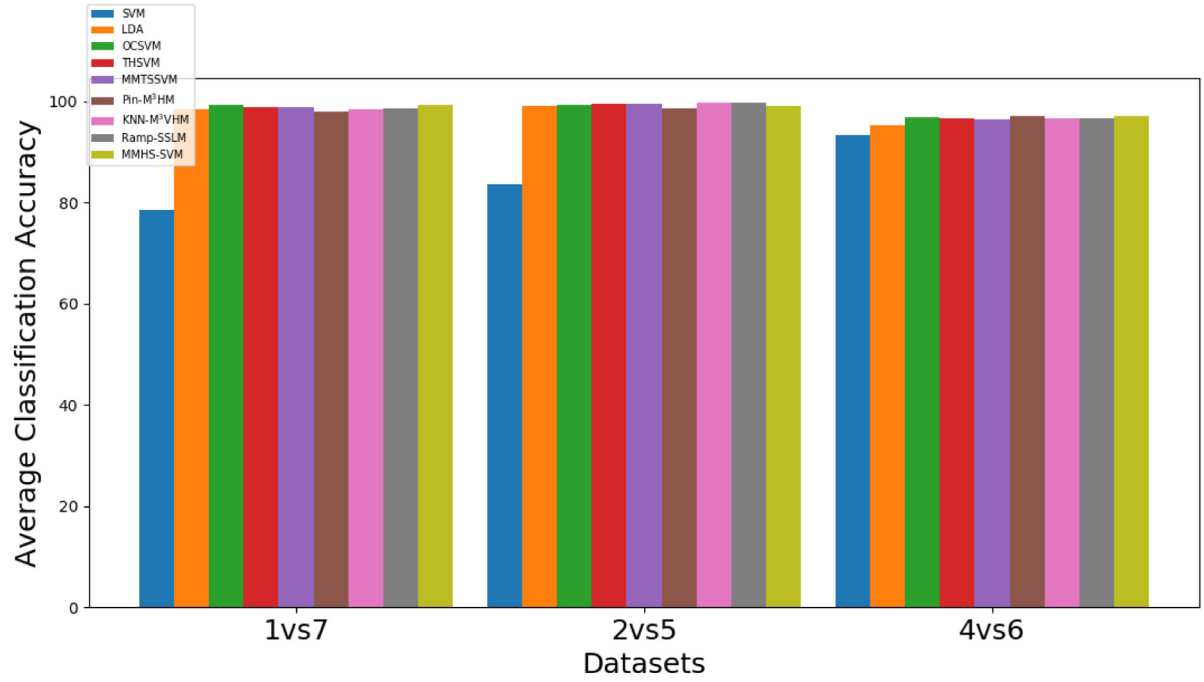
Table 7 (*continued*).

the SMO-typed technique overcomes the disadvantage of slow training speed. The results of experiments conducted on three artificial and several publicly available benchmark datasets verify that our proposed MMHS-SVM is feasible and effective.

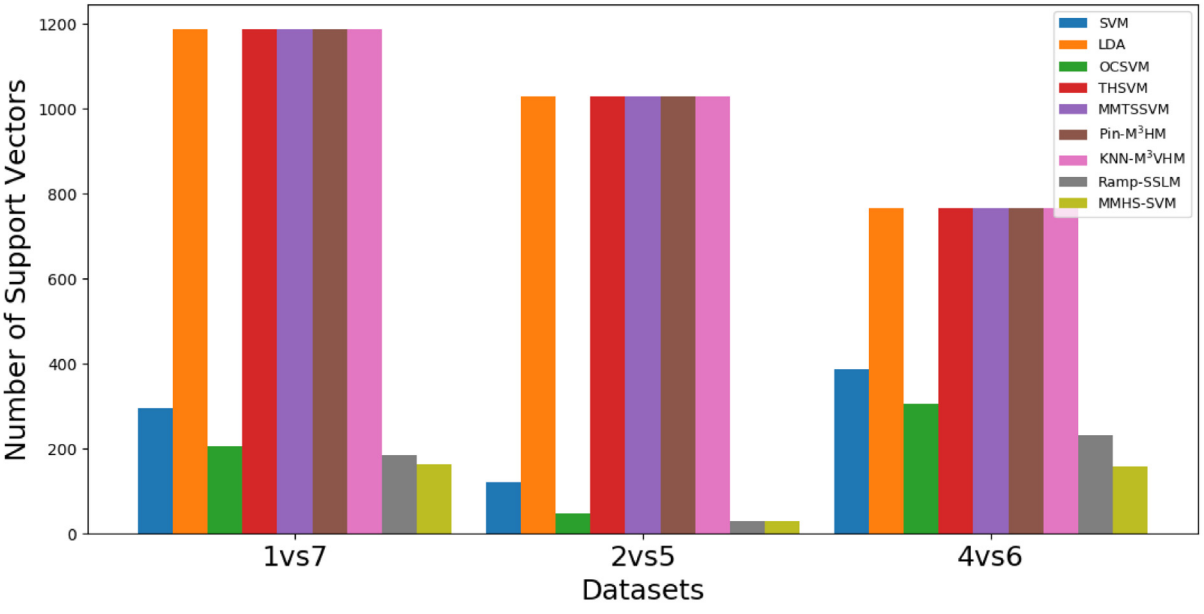
MMHS-SVM is a kind of novel hyper-sphere SVM. Due to the advantages mentioned above, we intend to conduct in-depth research in the following aspects: (1) The model of MMHS-SVM is extended to multi-class classification problems. (2) The Pinball loss function is added to the model to overcome the influence of outliers (under study). (3) The hyper-ellipsoids SVM model will be addressed to avoid the correlation between features by utilizing the Mahalanobis distance in the future.

CRediT authorship contribution statement

Ting Ke: Conceptualization, Methodology, Software, Investigation, Writing - original draft, Validation. **Yangyang Liao:** Validation, Software, Formal analysis. **Mengyan Wu:** Data curation. **Xuechun Ge:**



(a) Average classification accuracy



(b) Number of support vectors

Fig. 9. Average classification accuracy (a), #SV (b), Average training time (c) and Average testing time (d) on digital handwriting recognition.

Software, Supervision, Validation, Formal analysis. **Xinyi Huang:** Validation. **Chuanlei Zhang:** Writing – review & editing, Supervision. **Jianrong Li:** Resources.

Declaration of competing interest

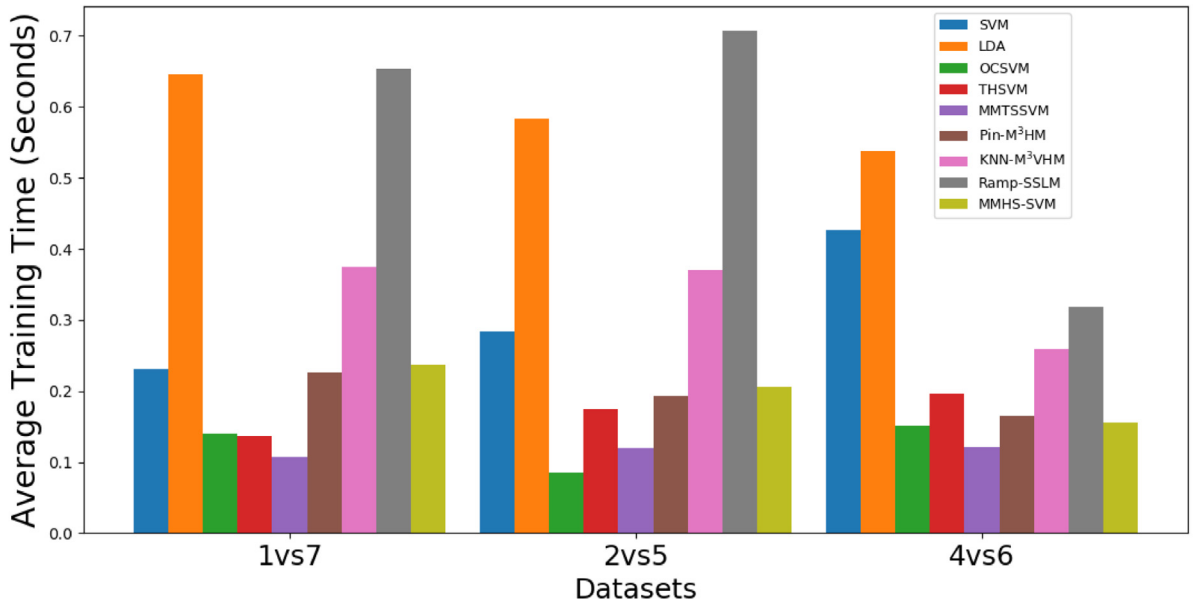
The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

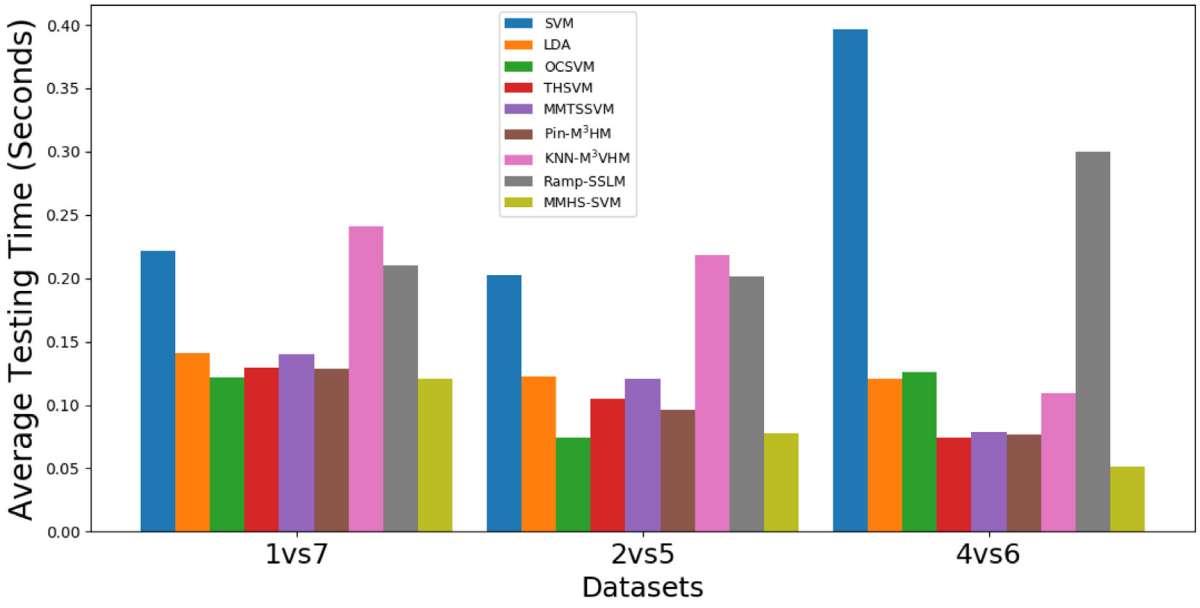
Data will be made available on request.

Acknowledgments

The authors appreciate the helpful comments and suggestions of the reviewers, which have improved the presentation. This research is supported by the Basic Research Project of Laboratory of National Railway



(c) Average training time



(d) Average testing time

Fig. 9. (*continued*).

Table 8
The average testing CPU time on UCI datasets.

Datasets	SVM	LDA	OSVC	THSVM	MMTSSVM	Pin-M^{3}HM	KNN-M3VHM	Ramp-SSLM	MMHS-SVM
Breast	0.01482	**0.00173**	0.00461	0.00658	0.00737	0.00672	0.01018	0.07447	0.00327
Bupa	0.01208	**0.00101**	0.00345	0.00307	0.00379	0.00338	0.01084	0.0589	0.0028
CMC	0.10128	**0.0028**	0.04558	0.04861	0.05246	0.04173	0.05128	0.1036	0.04756
Diabets	0.03723	**0.00177**	0.02487	0.01148	0.01461	0.01418	0.01704	0.0557	0.00777

(*continued on next page*)

Table 8 (*continued*).

Datasets	SVM	LDA	OSVC	THSVM	MMTSSVM	Pin-M^{3}HM	KNN-M3VHM	Ramp-SSLM	MMHS-SVM
Echoncardiogram	0.00179	0.00104	0.00059	0.00107	0.00137	0.00149	0.00791	0.01273	**0.00081**
Haberman	0.00446	**0.00078**	0.00164	0.00162	0.00269	0.00185	0.00146	0.01358	0.00082
Heart_disease	0.00637	**0.00107**	0.00429	0.00208	0.00281	0.00198	0.00478	0.01348	0.00174
Heart-statlog	0.00477	0.00184	0.00382	0.00576	0.00649	0.00569	0.00427	0.00974	**0.00183**
Hepatitis	0.00172	**0.0012**	0.00161	0.00168	0.00194	0.00194	0.00149	0.01314	0.00136
Ionosphere	0.00863	0.00303	0.00299	0.00307	0.00369	0.00307	0.00833	0.01478	**0.00236**
PimaIndian	0.03241	0.00965	0.02382	0.01486	0.02483	0.01695	0.02815	0.04927	**0.00879**
Seeds	**0.00094**	0.00311	0.0006	0.00361	0.00365	0.00366	0.00928	0.01228	**0.00251**
Sonar	0.00352	0.00276	0.00377	0.00185	0.00271	0.00195	0.00473	0.02826	**0.00144**
Spect	0.00641	0.00278	0.00452	0.00251	0.00382	0.00225	0.00624	0.01354	**0.00181**
Wdbc	0.00748	0.00532	0.00568	0.00547	0.00629	0.00592	0.00764	0.02576	**0.00454**

Table 9
Average rank.

Datasets	SVM	LDA	OCSVM	THSVM	MMTSSVM	Pin-M^{3}HM	KNN-M^{3}VHM	Ramp-SSLM	MMHS-SVM
Breast	6	9	8	4	3	2	5	7	**1**
Bupa	4	8	9	2	5	3	7	6	**1**
CMC	5	9	6	7	2	**1**	4	3	8
Diabets	8	2	9	7	6	3	4	5	**1**
Echoncardiogram	8	4	9	7	5	3	6	2	**1**
Haberman	2	4	5	9	3	8	7	6	**1**
Heart_disease	7	5	8	9	4	3	**1**	**2**	6
Heart-statlog	8	**1**	9	7	5	3	2	4	6
Hepatitis	3.5	7.5	3.5	9	6	3.5	3.5	7.5	**1**
Ionosphere	5	9	8	7	6	3	**1.5**	**1.5**	4
PimaIndian	7	4	9	6	8	3	2	5	**1**
Seeds	6	7	9	8	5	4	3	2	**1**
Sonar	5	8	9	4	7	6	2	3	**1**
Spect	4	9	6	3	2	7	8	5	**1**
Wdbc	8	5	9	4	6	3	2	7	1
Average	5.77	6.10	7.77	6.20	4.87	3.70	3.87	4.40	**2.33**

Table 10
Accuracy (%), the number of support vectors (# SV), training time and testing time on digital handwriting recognition.

Results	Datasets	SVM	LDA	OCSVM	THSVM	MMTSSVM	Pin-M^{3}HM	KNN-M3VHM	Ramp-SSLM	MMHS-SVM
Accuracy	1 vs. 7	78.5683	98.3333	99.25	98.8333	98.8901	98.0157	98.5382	98.7115	**99.3333**
	2vs5	83.5204	99.0826	99.2883	99.5413	99.6392	98.6106	**99.6843**	99.6675	99.0826
	4vs6	93.4589	95.3428	96.9628	96.6365	96.3949	**97.0246**	96.7641	96.5795	**97.0246**
# SV	1 vs. 7	295	1187	207	1187	1187	1187	1187	185	**164**
	2vs5	123	1031	49	1031	1031	1031	1031	**30**	**30**
	4vs6	387	767	307	767	767	767	767	232	**158**

(*continued on next page*)

Table 10 (*continued*).

Results	Datasets	SVM	LDA	OCSVM	THSVM	MMTSSVM	Pin-M^{3}HM	KNN-M3VHM	Ramp-SSLM	MMHS-SVM
Training time	1 vs. 7	0.2315	0.6461	0.1402	0.1375	**0.1063**	0.2267	0.3748	0.6545	0.2371
	2vs5	0.2836	0.5832	**0.0849**	0.1748	0.1194	0.1935	0.3701	0.7065	0.2064
	4vs6	0.4272	0.5381	0.1504	0.1963	**0.1206**	0.1648	0.2595	0.3188	0.1563
Testing time	1 vs. 7	0.2213	0.1405	0.1217	0.1293	0.1403	0.1286	0.2406	0.2106	**0.1203**
	2vs5	0.2021	0.1223	0.0743	0.1046	0.1209	0.0963	0.2184	0.2014	**0.0777**
	4vs6	0.3962	0.1204	0.1255	0.0744	0.0787	0.0769	0.1095	0.2995	**0.0511**

Intelligent Transportation System Engineering and Technology, China (L2022G004).

References

Borah, P., Gupta, D., 2021. Robust twin bounded support vector machines for outliers and imbalanced data. Appl. Intell. (3).

Cao, Y., Xu, Y., Du, J., 2019. Multi-variable estimation-based safe screening rule for small sphere and large margin support vector machine. Knowl.-Based Syst. 191 (6).

Chen, S., Wang, M., 2005. Seeking multi-threshold directly from support vectors for image segmentation. Neurocomputing 67, 335–344.

Chen, S.G., Wu, X.J., 2018. A new fuzzy twin support vector machine for pattern classification. Int. J. Mach. Learn. Cybern. 9 (9), 1553–1564.

Chen, S., Wu, X., Yin, H., 2019. A novel projection twin support vector machine for binary classification. Soft Comput. 23 (2), 655–668.

Chen, X.B., Yang, J., Ye, Q.L., et al., 2011. Recursive projection twin support vector machine via within-class variance minimization. Pattern Recognit. 44 (10–11), 2643–2655.

Cortes, C., Vapnik, V.N., 1995. Support vector networks. Mach. Learn. 20 (3), 273–297.

Fisher, R.A., 1936. The use of multiple measurements in taxonomic problems. Ann. Eug. 7 (2), 179–188.

Friedman, M., 1997. The use of ranks to avoid the assumption of normality implicit in the analysis of variance. J. Amer. Statist. Assoc. 32 (200), 675–701.

Fung, G., Mangasarian, O.L., 2001. Proximal support vector machine classifiers. In: ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. pp. 77–86.

Guo, W., Wang, Z., Hong, S., Li, D.D., Yang, H., et al., 2021. Multi-kernel support vector data description with boundary information. Eng. Appl. Artif. Intell. 102, 104254. http://dx.doi.org/10.1016/j.engappai.

Gupta, D., 2017. Training primal K-nearest neighbor based weighted twin support vector regression via unconstrained convex minimization. Appl. Intell. Int. J. Artif. Intell. Neural Netw. Complex Probl. Solving Technol. 47 (3), 962–991.

Gupta, U., Gupta, D., 2021. Kernel-target alignment based fuzzy Lagrangian twin bounded support vector machine. Int. J. Uncertain. Fuzziness Knowl.-Based Syst. 29 (5), 677–707.

Guyon, I., Weston, J., Barnhill, S., et al., 2002. Gene selection for cancer classification using support vector machine. Mach. Learn. 46, 389–422.

Hao, P.Y., Chiang, J.H., Lin, Y.H., 2009. A new maximal-margin spherical-structured multi-class support vector machine. Appl. Intell. 30 (2), 98–111.

Hazarika, B.B., Gupta, D., 2020. Density-weighted support vector machines for binary class imbalance learning. Neural Comput. Appl. 2.

Hazarika, B.B., Gupta, D., 2022. Density weighted twin support vector machines for binary class imbalance learning. Neural Process. Lett. 54 (2), 1091–1130.

Hazarika, B.B., Gupta, D., Borah, P., 2021. An intuitionistic fuzzy kernel ridge regression classifier for binary classification. Appl. Soft Comput. 112 (4), 107816.

Jayadeva, R., Khemchandani, Chandra, S., 2007. Twin support vector machines for pattern classification. IEEE Trans. Pattern Anal. Mach. Intell. 29 (5), 905–910.

Joachims, T., Ndellec, C., Rouveriol, 1998. Text categorization with support vector machines: learning with many relevant features. In: European Conference on Machine Learning, Chemnitz, Vol. 10. Germany, pp. 137–142.

Ke, T., Lv, H., Sun, M.J., et al., 2018. A biased least square support vector machine based on mahalanobis distance for PU learning. Physica A 509, 422–438.

Ke, T., Zhang, L.D., Ge, X.C., et al., 2020. A robust least squares support vector machine based on L_∞-norm. Neural Process. Lett. 52, 2371–2397.

Ke, T., Zhang, L.D., Ge, X.C., et al., 2021. Construct a robust least squares support vector machine based on Lp-norm and L∞-norm. Eng. Appl. Artif. Intell. 99, 104134.

Lin, Z.R., 2016. LIBSVM. http://www.csie.ntuedu.tw/~cjlin/libsvm.

Lin, J.Y., Cheng, C.T., Chau, K.W., 2006. Using support vector machines for long-term discharge prediction. Hydrol. Sci. J. 51 (4), 599–612.

Lpez, J., Maldonado, S., 2018. Robust twin support vector regression via second-order cone programming. Knowl.-Based Syst. 152, 83–93.

Lu, S., et al., 2018. All-in-one multicategory ramp loss maximum margin of twin spheres support vector machine. Appl. Intell. 49, 2301–2314.

Mir, A., Nasiri, J.A., 2018. KNN-based least squares twin support vector machine for pattern classification. Appl. Intell. 48 (12), 4551–4564.

Muller, K.R., Smola, A.J., G., Ratsch., et al., 1999. Using Support Vector Machines for Time Series Prediction, Advances in Kernel Methods-Support Vector Learning. MIT Press, Cambridge, MA, pp. 243–254.

Osuna, E., Freund, R., Girosi, F., 1997. Improved training algorithm for support vector machines. In: Proc. IEEE NNSP.

Peng, X., Xu, D., 2013. A twin-hypersphere support vector machine classifier and the fast-learning algorithm. Inform. Sci. 221 (1), 12–27.

Platt, J.C., 1999. Fast Training of Support Vector Machines using Sequential Minimal Optimization. MIT Press.

Rao, C.R., 1948. The utilization of multiple measurements in problems of biological classification. J. R. Stat. Soc. Ser. B 10 (2), 159–203.

Rastogi, R., Saigal, P., Chandra, S., 2018. Angle-based twin parametric-margin support vector machine for pattern classification. Knowl.-Based Syst. 139, 64–77.

Richhariya, B., Tanveer, M., 2018. A robust fuzzy least squares twin support vector machine for class imbalance learning. Appl. Soft Comput. 71, 418–432.

Schölkopf, B., Platt, J., et al., 2001. Estimating the support of a high-dimensional distribution. Neural Comput. 13 (7), 1443–1471.

Shao, Y.H., Chen, W.J., Deng, N.Y., 2014. Nonparallel hyperplane support vector machine for binary classification problems. Inf. Sci. Int. J. 263 (3), 22–35.

Shao, Y.H., Li, C.N., Liu, M.Z., et al., 2018. Sparse lq-norm least squares support vector machine with feature selection. Pattern Recognit. 78, 167–181.

Shao, Y.H., Wang, Z., Chen, W.J., et al., 2013. A regularization for the projection twin support vector machine. Knowl. Based Syst. 37, 203–210.

Suykens, J.A.K., 2000. Least squares support vector machines for classification and nonlinear modeling. Neural Network World 10 (1–2), 29–47.

Suykens, J.A.K., Gestel, T.V., Brabanter, J.D., et al., 2002. Least Squares Support Vector Machines. World Scientific, Singapore.

Suykens, J.A.K., Vandewalle, J., 1999. Least squares support vector machine classifiers. Neural Process. Lett. 9 (3), 293–300.

Tanveer, M., Rajani, T., Rastogi, R., et al., 2022. Comprehensive review on twin support vector machines. Ann. Oper. Res. 1–46.

Tao, Q., Zhan, S., Li, X.H., et al., 2016. Robust face detection using local CNN and SVM based on kernel combination. Neurocomputing 211, 98–105.

Tax, D., Duin, R., 2004. Support vector data description. Mach. Learn. 54 (1), 45–66.

Tomar, D., Agarwal, S., 2015. Hybrid feature selection based weighted least squares twin support vector machine approach for diagnosing breast cancer, hepatitis, and diabetes. Adv. Artif. Neural Syst. 1.

Wang, K.N., Cao, J.D., Pei, H.M., 2020. Robust extreme learning machine in the presence of outliers by iterative reweighted algorithm. Appl. Math. Comput. 377, 125186.

Wang, Y., Xu, Y., 2022. A non-convex robust small sphere and large margin support vector machine for imbalanced data classification. Neural Comput. Appl. http://dx.doi.org/10.1007/s00521-022-07882-2.

Wen, C.J., Zhan, Y.Z., Chen, C.J., 2010. Maximal-margin minimal-volume hypersphere support vector machine. Control Decis. 25 (1), 79–83.

Wu, M., Ye, J., 2009. A small sphere and large margin approach for novelty detection using training data with outliers. IEEE Trans. Pattern Anal. Mach. Intell. 31 (11), 2088–2092.

Xu, Y.T., 2016. Maximum margin of twin spheres support vector machine for imbalanced data classification. IEEE Trans. Cybern. 47 (6), 1540–1550.

Xu, Y., Wang, L., 2012. A weighted twin support vector regression. Knowl. Based Syst. 33, 92–101.

Xu, Y.T., Wang, Q., et al., 2018. Maximum margin of twin spheres machine with pinball loss for imbalanced data classification. Appl. Intell. Int. J.f Artif. Intell. Neural Netw. Complex Probl. Solving Technol. 48 (1), 23–34.

Xu, Y., Yang, Z., Pan, X., 2017. A novel twin support vector machine with pinball loss. IEEE Trans. Neural Netw. Learn. Syst. 28 (2), 359–370.

Xu, Y., Zhang, Y., Zhao, J., et al., 2019. KNN-based maximum margin and minimum volume hyper-sphere machine for imbalanced data classification. Int. J. Mach. Learn. Cybern. 10 (2), 357–368.

Xu, Y., et al., 2016. A maximum margin and minimum volume hyper-spheres machine with pinball loss for imbalanced data classification. Knowl.-Based Syst. 95 (Mar.1), 75–85.

Y.P., Zhao, Xie, Y.L., Ye, Z.F., 2021. A new dynamic radius SVDD for fault detection of aircraft engine. Eng. Appl. Artif. Intell. 100, 104177.

Yuan, M., Xu, Y.T., 2021. Bound estimation-based safe acceleration for maximum margin of twin spheres machine with pinball loss. Pattern Recognit. 114, 107860.

Zhong, G., Xiao, Y., Liu, B., et al., 2022. Pinball loss support vector data description for outlier detection. Appl. Intell. http://dx.doi.org/10.1007/s10489-022-03237-5.