IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received November 9, 2020, accepted November 19, 2020, date of publication November 24, 2020,
date of current version December 9, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3040199

An Effective Semi-Supervised Multi-Label Least Squares Twin Support Vector Machine

QING AI [1,2], YUDE KANG [1], ANNA WANG [2], XIANGNA LI [3], AND FEI LI [1]
[1]School of Computer Science and Software Engineering, University of Science and Technology Liaoning, Anshan 114051, China
[2]College of Information Science and Engineering, Northeastern University, Shenyang 110819, China
[3]Beijing FibrLink Communications Company Ltd., State Grid Information and Telecommunication Group Company Ltd., Beijing 100053, China

Corresponding author: Qing Ai (lyaiqing@126.com)

This work was supported in part by the Natural Science Foundation of Liaoning province in China under Grant 2020-MS-281, Grant 20180551048, and Grant 20170520248; and in part by the Talent Cultivation Project of University of Science and Technology Liaoning in China under Grant 2018RC05.

ABSTRACT Multi-label twin support vector machine (MLTSVM), being an effective multi-label classifier based on twin support vector machine (TSVM), has been widely studied and applied due to its excellent classification performance. However, there are some disadvantages in classical MLTSVM: (a) MLTSVM needs to solve a series of quadratic programming problems (QPPs), which makes its learning speed lower. (b) For multi-label learning problems, it is very difficult to obtain all labels of all samples. In fact, the datasets that we can obtain only contain a small amount of labeled samples and a large amount of partially labeled and unlabeled samples. However, MLTSVM can only use expensive labeled samples and ignore cheap unlabeled and partially labeled samples. For the drawbacks, we propose a novel semi-supervised multi-label least squares twin support vector machine, called SS-MLLSTSVM. Firstly, to speed up solving, SS-MLLSTSVM introduces the least squares idea into each sub-classifier of MLTSVM, which makes each sub-classifier only need to solve a system of linear equations, instead of one QPP. Secondly, SS-MLLSTSVM can make full use of the geometric information in unlabeled and partially labeled samples by introducing manifold regularization term into each sub-classifier. The experimental results on the benchmark datasets show that, compared with the existing multi-label classification algorithms, our SS-MLLSTSVM has better classification performance.

INDEX TERMS Twin support vector machine, least squares, multi-label learning, semi-supervised learning.

I. INTRODUCTION

TSVM [1], proposed by Jayadeva *et al.* in 2007, can be used to solve the binary classification problem. Because of its high learning speed and good generalization performance, it has been widely studied and applied. Many improvements of TSVM have been proposed, such as [2]–[18].

The above improvements can only solve the single-label learning problems, not solve the multi-label learning problems in which each sample may simultaneously belong to multiple labels. The multi-label learning problem is common, such as [19]–[22]. Up till now, there are two types of methods to solve the multi-label learning problem: problem transformation and algorithm adaptation. The problem transformation method solves the multi-label learning problem by transforming it into one or more single-label problems, such as label powerset (LP) [23], binary relevance (BR) [24], random k-labelsets (RAKEL) [25], classifier chains (CC) [26], calibrated label ranking (CLR) [27], and so on. The algorithm adaptation method extends the existing single-label learning algorithm to handle multi-label learning problem, such as ranking support vector machine (Rank-SVM) [28], collective multi-label classifier (CML) [29], multi-label k-nearest neighbor (ML-KNN) [30], multi-label decision tree (ML-DT) [31], backpropagation for multilabel learning (BPMLL) [32], and so on.

In order to extend TSVM to handle the multi-label learning problem, in 2016, Chen *et al.* proposed a multi-label twin support vector machine (MLTSVM) [33]. Compared with other traditional multi-label classification algorithms, MLTSVM has better generalization performance. Thereafter, many improvements of MLTSVM have been presented, such as KNN-based multi-label twin support vector machine with priority of labels (PKNN-MLTSVM) [34], structural least

The associate editor coordinating the review of this manuscript and approving it for publication was Wentao Fan.

square twin support vector machine for multi-label learning (ML-SLSTSVM) [35], *et al.*

However, there are some disadvantages in classical MLTSVM and its improvements: (a) MLTSVM needs to solve a series of quadratic programming problems (QPPs), which makes its learning speed lower. (b) For multi-label learning problems, it is very difficult to obtain all labels of all samples. In fact, the datasets we can obtain only contain a small amount of labeled samples and a large amount of partially labeled and unlabeled samples. However, MLTSVM can only use expensive labeled samples and ignore cheap unlabeled and partially labeled samples. For the drawbacks, we propose a novel semi-supervised muti-label least squares twin support vector machine, called SS-MLLSTSVM. Firstly, to speed up solving, SS-MLLSTSVM introduces the least squares idea into each sub-classifier of MLTSVM, which makes each sub-classifier only need to solve a system of linear equations, instead of one QPP. Secondly, SS-MLLSTSVM can make full use of the geometric information in unlabeled and partially labeled samples by introducing manifold regularization term into each sub-classifier. The experimental results on the benchmark datasets show that, compared with the existing multi-label classification algorithms, our SS-MLLSTSVM has better generalization performance.

The structure of this article is as follows: Section 2 introduces some related works, such as TSVM, least squares twin support vector machine (LSTSVM), MLTSVM, etc. In Section 3, the SS-MLLSTSVM is proposed, including linear case, nonlinear case and decision function. The fourth section presents the experimental results and analysis of our proposed algorithm on the benchmark datasets. The fifth section is the conclusions.

II. RELATED WORKS

For the binary classification problem, the training set is marked as $T = \{(x_i, y_i) \,|i = 1, \ldots, m\}$, where $x_i \in R^n$ is the training sample and $y_i \in \{+1, -1\}$ is the label corresponding to the training sample x_i. For convenience, we denote positive training samples as $A \in R^{m_1 \times n}$ and negative training samples as $B \in R^{m_2 \times n}$, where $m = m_1 + m_2$ is the total number of the training samples.

A. TSVM

The goal of TSVM is to seek the following two nonparallel hyperplanes:

$$f_+(x) : xw_+ + b_+ = 0 \;\; and \;\; f_-(x) : xw_- + b_- = 0. \quad (1)$$

The original problem of TSVM is as follows:

$$\begin{aligned} &\min_{w_+,b_+,\xi_-} \quad \frac{1}{2}\|Aw_+ + e_+b_+\|^2 + c_+e_-^T\xi_-, \\ &s.t. \; -(Bw_+ + e_-b_+) + \xi_- \geq e_-, \quad \xi_- \geq 0, \quad (2) \\ &\min_{w_-,b_-,\xi_+} \quad \frac{1}{2}\|Bw_- + e_-b_-\|^2 + c_-e_+^T\xi_+, \\ &s.t. \; Aw_- + e_+b_- + \xi_+ \geq e_+, \quad \xi_+ \geq 0, \quad (3) \end{aligned}$$

where $c_\pm$ are the penalty parameters, $\xi_\pm$ are the relaxation variables, and $e_\pm$ are the vector of all 1 of the proper dimension.

The dual problems of (2) and (3) are as follows:

$$\begin{aligned} &\max_{\alpha} \quad e_-^T\alpha - \frac{1}{2}\alpha^T E\left(F^TF\right)^{-1}E^T\alpha, \\ &s.t. \; 0 \leq \alpha \leq c_+e_-, \quad (4) \\ &\max_{\gamma} \quad e_+^T\gamma - \frac{1}{2}\gamma^T F\left(E^TE\right)^{-1}F^T\gamma, \\ &s.t. \; 0 \leq \gamma \leq c_-e_+, \quad (5) \end{aligned}$$

where α and γ are the Lagrange multipliers, $E = \begin{bmatrix} B & e_- \end{bmatrix}$ and $F = \begin{bmatrix} A & e_+ \end{bmatrix}$.

The two nonparallel hyperplanes can be obtained by solving the dual problems (4) and (5) as follows:

$$v_1 = \begin{bmatrix} w_+^T & b_+ \end{bmatrix}^T = -\left(F^TF\right)^{-1}E^T\alpha, \quad (6)$$

$$v_2 = \begin{bmatrix} w_-^T & b_- \end{bmatrix}^T = \left(E^TE\right)^{-1}F^T\gamma. \quad (7)$$

B. LSTSVM

Similar to TSVM, LSTSVM also seeks two nonparallel hyperplanes. However LSTSVM solves the following two quadratic programming problems (QPPs) to obtain the two nonparallel hyperplanes:

$$\begin{aligned} &\min_{w_+,b_+,\xi_-} \quad \frac{1}{2}\|Aw_+ + e_+b_+\|^2 + \frac{1}{2}c_+\xi_-^T\xi_-, \\ &s.t. \; -(Bw_+ + e_-b_+) + \xi_- = e_-, \quad (8) \\ &\min_{w_-,b_-,\xi_+} \quad \frac{1}{2}\|Bw_- + e_-b_-\|^2 + \frac{1}{2}c_-\xi_+^T\xi_+, \\ &s.t. \; Aw_- + e_+b_- + \xi_+ = e_+. \quad (9) \end{aligned}$$

Different from the primal problems (2) and (3) of TSVM, LSTSVM replaces the inequality constraints with equality constraints and 1-norm of slack variables $\xi_\pm$ with the square of 2-norm.

By substituting equality constraints into the objective functions in (8) and (9), we can obtain

$$\begin{aligned} &\begin{bmatrix} A & e_+ \end{bmatrix}^T\begin{bmatrix} A & e_+ \end{bmatrix}\begin{bmatrix} w_+^T & b_+ \end{bmatrix}^T \\ &\quad + c_+\begin{bmatrix} B & e_- \end{bmatrix}^T\begin{bmatrix} B & e_- \end{bmatrix}\begin{bmatrix} w_+^T & b_+ \end{bmatrix}^T \\ &\quad + c_+\begin{bmatrix} B & e_- \end{bmatrix}^Te_- = 0, \quad (10) \\ &\begin{bmatrix} B & e_- \end{bmatrix}^T\begin{bmatrix} B & e_- \end{bmatrix}\begin{bmatrix} w_-^T & b_- \end{bmatrix}^T \\ &\quad + c_-\begin{bmatrix} A & e_+ \end{bmatrix}^T\begin{bmatrix} A & e_+ \end{bmatrix}\begin{bmatrix} w_-^T & b_- \end{bmatrix}^T \\ &\quad - c_-\begin{bmatrix} A & e_+ \end{bmatrix}^Te_+ = 0. \quad (11) \end{aligned}$$

Supposing $G = \begin{bmatrix} B & e_- \end{bmatrix}$, $H = \begin{bmatrix} A & e_+ \end{bmatrix}$, we can obtain

$$\begin{bmatrix} w_+^T & b_+ \end{bmatrix}^T = -\left(G^TG + c_+H^TH\right)^{-1}G^Te_-, \quad (12)$$

$$\begin{bmatrix} w_-^T & b_- \end{bmatrix}^T = \left(H^TH + c_-G^TG\right)^{-1}H^Te_+. \quad (13)$$

C. MLTSVM

For the multi-label learning problems, we set the training set as $T = \{(x_i, y_i) \,|i = 1, \ldots, m\}$, where $x_i \in R^n$ is the training sample, $y_i = \{y_{i1}, \ldots, y_{ik}, \ldots, y_{iK}\}$ is the label sequence of the sample x_i,

$$y_{ik} = \begin{cases} +1, & \textit{if } x_i \textit{ belongs to the kth class}, \\ -1, & \textit{otherwise}, \end{cases} \tag{14}$$

$1 \le k \le K$, m is the total number of training samples and K is the total number of labels.

The MLTSVM seeks the following K hyperplanes:

$$f_k(x) : xw_k + b_k = 0, \quad k = 1, 2, \ldots, K. \tag{15}$$

Denote the samples belonging to the kth class by A_k and the samples not belonging to the kth class by B_k. To obtain the kth hyperplane, the original problem of MLTSVM is as follows:

$$\begin{aligned} \min_{w_k, b_k, \xi_{B_k}} \; & \frac{1}{2}\left\|A_k w_k + e_{A_k} b_k\right\|^2 + c_k e_{B_k}^T \xi_{B_k} \\ & + \frac{1}{2}\lambda_k(\|w_k\|^2 + b_k^2), \\ s.t. \; & -(B_k w_k + e_{B_k} b_k) + \xi_{B_k} \ge e_{B_k}, \quad \xi_{B_k} \ge 0, \end{aligned} \tag{16}$$

where c_k is the penalty parameter, ξ_{B_k} is the slack variable, λ_k is the regularization parameter, and $e_{A_k(B_k)}$ are the vector of all 1 of the proper dimension.

By introducing Lagrange function and using Karush-Kuhn-Tucker (KKT) optimization theory, the dual problem of (16) can be obtained as follows:

$$\begin{aligned} \max_{\alpha_{B_k}} \; & e_{B_k}^T \alpha_{B_k} - \frac{1}{2}\alpha_{B_k}^T G(H^T H + \lambda_k I_k)^{-1} G^T \alpha_{B_k}, \\ s.t. \; & 0 \le \alpha_{B_k} \le c_k, \end{aligned} \tag{17}$$

where $H = \left[A_k \; e_{A_k}\right]$, $G = \left[B_k \; e_{B_k}\right]$, I_k is the identity matrix of proper dimensions, and α_{B_k} is the Lagrange multiplier.

By solving the dual problem, we can obtain

$$u_k = \left[w_k^T \quad b_k^T\right]^T = -(H^T H + \lambda_k I_k)^{-1} G^T \alpha_{B_k}. \tag{18}$$

III. SS-MLLSTSVM

Consider the semi-supervised multi-label learning problem with training set $T = \{(x_i, y_i) \,|i = 1, \ldots, u\}$, where $x_i \in R^n$ is the training sample, and $y_i = \{y_{i1}, \ldots, y_{ik}, \ldots, y_{iK}\}$ is the label sequence of the sample x_i.

$$y_{ik} = \begin{cases} +1, & \textit{if } x_i \textit{ belongs to the kth class}, \\ -1, & \textit{if } x_i \textit{ does not belongs to the kth class}, \\ 0, & \textit{uncertain}, \end{cases} \tag{19}$$

$1 \le k \le K$, u is the total number of all training samples, including labeled, unlabeled and partially labeled samples and K is the total number of labels.

A. SEMI-SUPERVISED LEARNING FRAMEWORK

To solve the semi-supervised learning problems, Belkin *et al.* proposed a manifold regularization framework. The objective function of the manifold regularization framework is expressed as follows:

$$f^* = \underset{f \in H_k}{\operatorname{argmin}} \sum_{i=1}^{l} V(x_i, y_i, f) + \gamma_H \|f\|_H^2 + \gamma_M \|f\|_M^2, \tag{20}$$

where f is the classification function to be solved, H_k is the reproducing kernel Hilbert space (RKHS), the first part V is the loss function of labeled samples, the second part $\|f\|_H^2$ is a regularization term used to control the complexity of the classifier, and the third part $\|f\|_M^2$ is a manifold regularization term, which reflects the internal manifold structure of data distribution.

B. MODEL

1) LINEAR CASE

For the label k, SS-MLLSTSVM seeks a hyperplane

$$f_k(x) : xw_k + b_k = 0, \quad k = 1, 2, \ldots, K. \tag{21}$$

The second part $\|f\|_H^2$ of (20) can be expressed as:

$$\|f_k\|_H^2 = \frac{1}{2}\left(\|w_k\|_2^2 + b_k^2\right). \tag{22}$$

The third part $\|f\|_M^2$ of (20) can be expressed as:

$$\|f_k\|_M^2 = \frac{1}{u^2}\sum_{i,j=1}^{u} W_{i,j}\left(f_k(x_i) - f_k\left(x_j\right)\right)^2 = f_k^T L f_k, \tag{23}$$

where $f_k = [f_k(x_1), \ldots, f_k(x_u)]^T = Tw_k + eb_k$, $L = D - W$ is the Laplace matrix of the whole samples, W is defined as follows:

$$W_{i,j} = \begin{cases} \exp\left(-\left\|x_i - x_j\right\|_2^2 / 2\sigma^2\right), & \textit{if } x_i \textit{ and } x_j \textit{ are} \\ 0, & k \textit{ nearest neighbor}^1 \\ & \textit{otherwise}, \end{cases} \tag{24}$$

and D is defined as follows:

$$D_{i,i} = \sum_{j=1}^{l+u} W_{i,j}. \tag{25}$$

For the kth label, we suppose $A_k = \{x_i | y_{ik} = +1\}$, $B_k = \{x_i | y_{ik} = -1\}$, $U_k = \{x_i | y_{ik} = 0\}$, $T = A_k \cup B_k \cup U_k$. The original optimization problem of the linear SS-MLLSTSVM is:

$$\begin{aligned} \min \; & \frac{1}{2}\left(A_k w_k + e_{A_k} b_k\right)^2 + \frac{1}{2} c_{k2}\left(\|w_k\|^2 + b_k^2\right) \\ & + \frac{1}{2} c_{k1} \xi_{B_k}^T \xi_{B_k} + \frac{1}{2} c_{k3} (Tw_k + eb_k)^T L (Tw_k + eb_k), \\ s.t. \; & -\left(B_k w_k + e_{B_k} b_k\right) + \xi_{B_k} = e_{B_k}, \end{aligned} \tag{26}$$

[1] K nearest neighbor algorithm (KNN) [36] is used to judge whether samples x_i and x_j are neighbors.

where $c_{ki}(i = 1, 2, 3)$ are the penalty parameters, and L is the Laplace matrix of the whole samples. It can be observed from the optimization problem (26) that (a) similar to LSTSVM, SS-MLLSTSVM replaces the inequality constraints of the MLTSVM with the equality constraints and 1-norm of slack variables ξ_{B_k} of MLTSVM with the square of 2-norm; (b) unlike MLTSVM and LSTSVM, SS-MLLSTSVM adds manifold regularization term in order to make full use of the information of partially labeled and unlabeled samples. Therefore, SS-MLLSTSVM can effectively solve the semi-supervised multi-label problem.

The Lagrange function of (26) is as follows:

$$\begin{aligned} L(w_k, b_k) = &\frac{1}{2}\left(A_k w_k + e_{A_k} b_k\right)^2 + \frac{1}{2}c_{k2}\left(\|w_k\|^2 + b_k^2\right) \\ &+ \frac{1}{2}c_{k1}\left(e_{B_k} + B_k w_k + e_{B_k} b_k\right)^2 \\ &+ \frac{1}{2}c_{k3}(Tw_k + eb_k)^T L\left(Tw_k + eb_k\right). \end{aligned} \tag{27}$$

Using KKT condition, we can obtain:

$$\begin{aligned} \frac{\partial L}{\partial w_k} = &A_k^T\left(A_k w_k + e_{A_k} b_k\right) + c_{k3}T^T L\left(Tw_k + eb_k\right) \\ &+ c_{k1}B_k^T\left(e_{B_k} + B_k w_k + e_{B_k} b_k\right) + c_{k2}w_k = 0, \end{aligned} \tag{28}$$

$$\begin{aligned} \frac{\partial L}{\partial b_k} = &e_{A_k}^T\left(A_k w_k + e_{A_k} b_k\right) + c_{k3}e^T L\left(Tw_k + eb_k\right) \\ &+ c_{k1}e_{B_k}^T\left(e_{B_k} + B_k w_k + e_{B_k} b_k\right) + c_{k2}b_k = 0. \end{aligned} \tag{29}$$

Combining (28) and (29), we can obtain:

$$\begin{aligned} &\begin{bmatrix} A_k^T \\ e_{A_k}^T \end{bmatrix}\begin{bmatrix} A_k & e_{A_k} \end{bmatrix}\begin{bmatrix} w_k \\ b_k \end{bmatrix} \\ &\quad + c_{k1}\begin{bmatrix} B_k^T \\ e_{B_k}^T \end{bmatrix}\begin{bmatrix} B_k & e_{B_k} \end{bmatrix}\begin{bmatrix} w_k \\ b_k \end{bmatrix} \\ &\quad + c_{k1}\begin{bmatrix} B_k^T \\ e_{B_k}^T \end{bmatrix} e_{B_k} + c_{k2}\begin{bmatrix} w_k \\ b_k \end{bmatrix} \\ &\quad + c_{k3}\begin{bmatrix} T^T \\ e^T \end{bmatrix} L\begin{bmatrix} T & e \end{bmatrix}\begin{bmatrix} w_k \\ b_k \end{bmatrix} = 0. \end{aligned} \tag{30}$$

Denoting $E = \begin{bmatrix} A_k & e_{A_k} \end{bmatrix}$, $F = \begin{bmatrix} B_k & e_{B_k} \end{bmatrix}$, $J = \begin{bmatrix} T & e \end{bmatrix}$ and $u_k = \begin{bmatrix} w_k^T & b_k \end{bmatrix}^T$, we can obtain

$$u_k = -c_{k1}[E^T E + c_{k2}I_k + c_{k1}F^T F + c_{k3}J^T LJ]^{-1}F^T e_{B_k}. \tag{31}$$

2) NONLINEAR CASE

In this section, we extend the linear SS-MLLSTSVM to the nonlinear case using the approximate kernel generating surface. For the nonlinear case, SS-MLLSTSVM constructs K approximate kernel generating surface

$$f_k(x) : K(x, T^T)w_k + b_k = 0, \quad k = 1, 2, \ldots, K, \tag{32}$$

where $K(\cdot, \cdot)$ is a suitable kernel function.

Similar to the linear case, the second part and the third part in (20) can be respectively expressed as follows:

$$\|f_k\|_H^2 = \frac{1}{2}\left(w_k^T K\left(T, T^T\right)w_k + b_k^2\right), \tag{33}$$

$$\begin{aligned} \|f_k\|_M^2 &= f_k^T L f_k \\ &= \left(K\left(T, T^T\right)w_k + eb_k\right)^T L\left(K\left(T, T^T\right)w_k + eb_k\right). \end{aligned} \tag{34}$$

The original optimization problem of nonlinear SS-MLLSTSVM is as follows:

$$\begin{aligned} \min\ &\frac{1}{2}\left(K\left(A_k, T^T\right)w_k + e_{A_k}b_k\right)^2 + \frac{1}{2}c_{k1}\xi_{B_k}^T\xi_{B_k} \\ &+ \frac{1}{2}c_{k3}\left(K\left(T, T^T\right)w_k + eb_k\right)^T \\ &\times L\left(K\left(T, T^T\right)w_k + eb_k\right) \\ &+ \frac{1}{2}c_{k2}\left(\|w_k\|^2 + b_k^2\right), \\ s.t.\ &-\left(K\left(B_k, T^T\right)w_k + e_{B_k}b_k\right) + \xi_{B_k} = e_{B_k}. \end{aligned} \tag{35}$$

The Lagrange function of (35) can be constructed as follows:

$$\begin{aligned} L(w_k, b_k) = &\frac{1}{2}\left(K\left(A_k, T^T\right)w_k + e_{A_k}b_k\right)^2 \\ &+ \frac{1}{2}c_{k3}\left(K\left(T, T^T\right)w_k + eb_k\right)^T \\ &\times L\left(K\left(T, T^T\right)w_k + eb_k\right) \\ &+ \frac{1}{2}c_{k1}\left(K\left(B_k, T^T\right)w_k + e_{B_k}b_k + e_{B_k}\right)^2 \\ &+ \frac{1}{2}c_{k2}\left(\|w_k\|^2 + b_k^2\right). \end{aligned} \tag{36}$$

Using KKT condition, we can obtain:

$$\begin{aligned} \frac{\partial L}{\partial w_k} = &K\left(A_k, T^T\right)^T\left(K\left(A_k, T^T\right)w_k + e_{A_k}b_k\right) + c_{k2}w_k \\ &+ c_{k1}K\left(B_k, T^T\right)^T\left(K\left(B_k, T^T\right)w_k + e_{B_k}b_k + e_{B_k}\right) \\ &+ c_{k3}K\left(T, T^T\right)^T L\left(K\left(T, T^T\right)w_k + eb_k\right) = 0, \end{aligned} \tag{37}$$

$$\begin{aligned} \frac{\partial L}{\partial b_k} = &e_{A_k}^T\left(K\left(A_k, T^T\right)w_k + e_{A_k}b_k\right) + c_{k2}b_k \\ &+ c_{k1}e_{B_k}^T\left(K\left(B_k, T^T\right)w_k + e_{B_k}b_k + e_{B_k}\right) \\ &+ c_{k3}e^T L\left(K\left(T, T^T\right)w_k + eb_k\right) = 0. \end{aligned} \tag{38}$$

Combining (37) and (38), we can obtain:

$$\begin{aligned} &\begin{bmatrix} K\left(A_k, T^T\right)^T \\ e_{A_k}^T \end{bmatrix}\begin{bmatrix} K\left(A_k, T^T\right) & e_{A_k} \end{bmatrix}\begin{bmatrix} w_k \\ b_k \end{bmatrix} \\ &\quad + c_{k1}\begin{bmatrix} K\left(B_k, T^T\right)^T \\ e_{B_k}^T \end{bmatrix}\begin{bmatrix} K\left(B_k, T^T\right) & e_{B_k} \end{bmatrix}\begin{bmatrix} w_k \\ b_k \end{bmatrix} \\ &\quad + c_{k1}\begin{bmatrix} K\left(B_k, T^T\right)^T \\ e_{B_k}^T \end{bmatrix} e_{B_k} + c_{k2}\begin{bmatrix} w_k \\ b_k \end{bmatrix} \end{aligned}$$

$$+c_{k3}\begin{bmatrix} K\left(T,T^T\right)^T \\ e^T \end{bmatrix} L \left[K\left(T,T^T\right)\ e \right] \begin{bmatrix} w_k \\ b_k \end{bmatrix} = 0. \tag{39}$$

Denoting $E = \left[K\left(A_k, T^T\right)\ e_{A_k}\right]$, $F = \left[K\left(B_k, T^T\right)\ e_{B_k}\right]$, $J = \left[K\left(T, T^T\right)\ e\right]$ and $u_k = \left[w_k^T\ b_k\right]^T$, we can obtain

$$u_k = -c_{k1}[E^T E + c_{k2} I_k + c_{k1} F^T F + c_{k3} J^T L J]^{-1} F^T e_{B_k}. \tag{40}$$

C. DECISION FUNCTION

In this subsection, by fusing the K sub-classifiers [37], we present the decision function of our SS-MLLSTSVM. For a new sample x, as mentioned above, if the sample x is close enough to a hyperplane, it should be marked as the corresponding label. In other words, if the distance $d_k(x)$ between x and the kth hyperplane

$$d_k(x) = \frac{\left|w_k^T x + b_k\right|}{\|w_k\|}, \quad k = 1, \ldots, K, \tag{41}$$

is less than or equal to the given value $\Delta_k, k = 1, \ldots, K$, the sample x is assigned to the kth label.

To choose the proper Δ_k, we apply the strategy in the MLTSVM, which is a simple and effective method, i.e. we set $\Delta_k = \Delta = min_{p=1,\ldots,K}\left(\frac{1}{\|w_p\|}\right), k = 1, \ldots, K$.

D. COMPLEXITY ANALYSIS

In this subsection, we analyze the computational complexity of SS-MLLSTSVM. The computational complexity of SS-MLLSTSVM mainly includes two parts: solving the Laplace matrix of the whole samples and solving the linear equations. The main computational burden of solving the Laplace matrix of the whole samples is to seek the K nearest neighbor of each sample, and the computational complexity of seeking the K nearest neighbor of all samples is $O\left(u^2 \log(u)\right)$. For linear case, the computational complexity of solving linear equations is $O\left(d^3\right)$, where d is the dimension of training samples. For nonlinear case, the computational complexity of solving linear equations is $O\left(u^3\right)$, where u is the total number of all training samples. Therefore, the computational complexities of linear and nonlinear SS-MLLSTSVM are respectively $O\left(u^2 \log(u) + Kd^3\right)$ and $O\left(u^2 \log(u) + Ku^3\right)$.

IV. EXPERIMENTS

In this section, we present the classification results of our proposed SS-MLLSTSVM on multiple datasets. We compare our SS-MLLSTSVM with BPMLL [32], Rank-SVM [28] and MLTSVM on the multi-label benchmark datasets. All the algorithms are implemented in MATLAB (R2017b), and the experimental environment is Intel Core i3 processor with 4G RAM.

A. BENCHMARK DATASETS

In the experiments, we used five common multi-label datasets, including flags, birds, emotions, yeast and scene. The datasets cover multiple fields, including image, audio, music, biology, and so on. The details of the datasets are listed in Table 1. In addition, in order to investigate the classification ability of our proposed algorithm, we choose 50% of the datasets as labeled samples and the remaining samples as unlabeled samples.

B. EVALUATION CRITERIA

In the experiments, in order to evaluate the performance of the algorithm, we use 7 common evaluation metrics, including Hamming loss, average precision, coverage, one error, ranking loss, balanced accuracy and Kappa. Next, we will introduce the 7 evaluation metrics in detail.

Let m be the total number of samples and K be the total number of labels. Y_i and $\overline{Y_i}$ respectively represent the relevant label set and irrelevant label set of sample x_i. The function $f(x, y)$ returns the confidence of y being the right label of sample x, and the function $rank(x, y, f)$ returns a descending rank of $f(x, y)$ for any $y \in \{y_1, \ldots, y_K\}$. For the kth label, TP_k represents the number of samples that belong to the kth label and are predicted correctly; TN_k represents the number of samples that do not belong to the kth label and are not predicted to be the kth label; FP_k represents the number of samples that do not belong to the kth label and are predicted to be the kth label; FN_k represents the number of samples that belong to the kth label and are not predicted to be the kth label.

1) HAMMING LOSS

Hamming loss is used to measure the proportion of labels which are misclassified

$$Hamming\ loss = \frac{1}{m}\sum_{i=1}^{m}\frac{1}{K}\left|h(x_i)\,\Delta Y_i\right|, \tag{42}$$

where $h(x_i)$ is the predicted labels of sample x_i.

2) COVERAGE

Coverage is used to measure that, to cover all possible labels of samples, how far we need to go down the ranked

TABLE 1. Detailed description of the datasets.

Datasets	Domain	Labeled Instance	Unlabeled Instance	Label	Feature
Flags	image	100	94	7	19
Emotions	music	300	293	6	72
Birds	audio	322	323	19	260
Scene	image	1204	1203	6	294
Yeast	biology	1236	1235	14	103

label list.

$$coverage = \frac{1}{m}\sum_{i=1}^{m}\max_{y\in Y_i} rank\ (x_i, y, f) - 1. \tag{43}$$

3) ONE ERROR

One error is used to measure the proportion of samples whose label with the highest prediction probability isn't in the true label set.

$$one\ error = \frac{1}{m}\sum_{i=1}^{m} H\ (x_i), \tag{44}$$

where

$$H\ (x_i) = \begin{cases} 0, & if\ \ \arg\max f\ (x_i, y) \in Y_i \\ 1, & otherwise. \end{cases} \tag{45}$$

4) RANKING LOSS

Ranking loss is used to measure the proportion of label pairs that are reversely ordered.

$$\begin{aligned} &ranking\ loss \\ &= \frac{1}{m}\sum_{i=1}^{m}\left(\frac{1}{|Y_i|\left|\overline{Y_i}\right|}\left|\left\{\left(y'y''\right)|f\left(x_i, y'\right) \le f\left(x_i, y''\right),\right.\right.\right. \\ &\qquad \left.\left.\left. y' \in Y_i, y'' \in \overline{Y_i}\right\}\right|\right). \end{aligned} \tag{46}$$

5) AVERAGE PRECISION

Average precision is used to measure the proportion of labels ranked above a particular label $y \in Y_i$.

$$\begin{aligned} &average\ precision \\ &= \frac{1}{m}\sum_{i=1}^{m} \\ &\times\left(\frac{1}{|Y_i|}\sum_{y\in Y_i}\frac{\left|\left\{\left(y' \in Y_i\right)|rank\left(x_i, y', f\right) \le rank\ (x_i, y, f)\right\}\right|}{rank\ (x_i, y, f)}\right). \end{aligned} \tag{47}$$

6) BALANCED ACCURACY

Balanced accuracy is used to measure the classification performance of the classifier for unbalanced dataset.

$$balanced\ accuracy = \frac{1}{K}\sum_{k=1}^{K}\frac{TPR_k + TNR_k}{2}, \tag{48}$$

where $TPR_k = \frac{TP_k}{TP_k+FN_k}$, $TNR_k = \frac{TN_k}{TN_k+FP_k}$.

7) KAPPA

Kappa coefficient is used to test the consistency between the predicted results of the classifier and the actual results.

$$kappa = \frac{1}{K}\sum_{k=1}^{K}\frac{p_{ok} - p_{ek}}{1 - p_{ek}}, \tag{49}$$

where

$$p_{ok} = \frac{TP_k + TN_k}{m}, \tag{50}$$

$$p_{ek} = \frac{TP_k \times (TP_k + FN_k) + TN_k \times (TN_k + FP_k)}{m^2}. \tag{51}$$

C. PARAMETER SETTING

The parameters of classifiers have an important impact on the classification performance. We use 5-fold cross validation to select optimal parameters. The parameters of each algorithm are set as follows: For the BPMLL, the number of hidden neurons is set to 20% of the input dimension, and the number of training epochs is 100. For the Rank-SVM, the kernel function parameter and penalty parameter c are selected from $\left\{2^{-6}, \ldots, 2^{0}, \ldots, 2^{6}\right\}$. For the MLTSVM, the penalty parameters c_k and regularization parameter λ_k are selected from $\left\{2^{-6}, \ldots, 2^{0}, \ldots, 2^{6}\right\}$. For the SS-MLLSTSVM, the penalty parameters c_{k1} and regularization parameters c_{k2}, c_{k3} are selected from $\left\{2^{-6}, \ldots, 2^{0}, \ldots, 2^{6}\right\}$.

D. RESULTS

The classification results of BPMLL, Rank-SVM, MLTSVM and our SS-MLLSTSVM on benchmark datasets are presented in this subsection. In the experiments, we use 5-fold cross validation to evaluate these algorithms. The mean and standard deviation of 20 rounds 5-fold cross validation for each metrics are respectively listed in Tables 2 to 8.

From Table 2 and 3, we can observe that our SS-MLLSTSVM is superior to all other multi-label classifiers for average precision and balanced accuracy. However, from Table 4 to 8, we can observe that no algorithm is superior to any other algorithms on all datasets for coverage, Hamming loss, one error, ranking loss and kappa. Further, we use Friedman test to evaluate each algorithm statistically. The Friedman statistics are as follows:

$$\chi_F^2 = \frac{12N}{k\ (k+1)}\left[\sum_j R_j^2 - \frac{k(k+1)^2}{4}\right], \tag{52}$$

where $R_j = \frac{1}{N}\sum_i r_i^j$, r_i^j represents the rank of the jth algorithm on the ith dataset, k is the number of classifiers, and N is the number of datasets. Because χ_F^2 is undesirably conservative, we apply the better statistic

$$F_F = \frac{(N-1)\ \chi_F^2}{N\ (k-1) - \chi_F^2} \sim F\ (k-1, (k-1)\ (N-1)). \tag{53}$$

For coverage, Hamming loss, one error, ranking loss and kappa, we list the rank of different multi-label classifiers in Table 9 to 13. We can obtain $\chi_F^2\ (coverage) = 12.24$, $\chi_F^2\ (Hamming\ loss) = 6.36$, $\chi_F^2\ (one\ error) = 9.72$, $\chi_F^2\ (ranking\ loss) = 8.76$, $\chi_F^2\ (kappa) = 10.92$ and $F_F\ (coverage) = 17.74$, $F_F\ (Hamming\ loss) =$

TABLE 2. Average precision of 4 algorithms on the benchmark datasets.

	BPMLL	Rank-SVM	MLTSVM	SS-MLLSTSVM
Flags	0.775475±0.058365	0.788136±0.003669	0.773697±0.011447	**0.791300±0.007308**
Emotions	0.583526±0.032627	0.718366±0.005625	0.754178±0.010533	**0.812264±0.007244**
Birds	0.363741±0.014535	0.410044±0.003245	0.333250±0.016647	**0.604462±0.019554**
Scene	0.459482±0.022864	0.827531±0.002350	0.844705±0.005490	**0.848723±0.004143**
Yeast	0.723663±0.047532	0.730048±0.001207	0.743770±0.001668	**0.747245±0.004030**

TABLE 3. Balanced accuracy of 4 algorithms on the benchmark datasets.

	BPMLL	Rank-SVM	MLTSVM	SS-MLLSTSVM
Flags	0.468681±0.054886	0.499285±0.002277	0.518505±0.012119	**0.563401±0.011952**
Emotions	0.497986±0.005075	0.531717±0.006086	0.670206±0.006409	**0.690014±0.003425**
Birds	0.498701±0.004107	0.499473±0.001109	0.593721±0.009356	**0.637222±0.006717**
Scene	0.504651±0.006685	0.818622±0.002091	0.807825±0.005088	**0.826202±0.004330**
Yeast	0.511163±0.007799	0.552296±0.000083	0.554540±0.002277	**0.562105±0.002662**

TABLE 4. Coverage of 4 algorithms on the benchmark datasets.

	BPMLL	Rank-SVM	MLTSVM	SS-MLLSTSVM
Flags	3.884626±0.035263	**3.803867±0.026583**	4.050580±0.070484	3.845263±0.075064
Emotions	2.643704±0.094634	2.224868±0.031156	2.028147±0.033584	**1.860791±0.023553**
Birds	3.918503±0.064267	3.899046±0.030467	3.966183±0.114917	**2.516635±0.124385**
Scene	2.137436±0.373585	0.557480±0.008570	0.535686±0.016421	**0.523748±0.025564**
Yeast	6.345796±0.342050	**6.240031±0.010076**	6.368937±0.023531	6.345201±0.014414

TABLE 5. Hamming loss of 4 algorithms on the benchmark datasets.

	BPMLL	Rank-SVM	MLTSVM	SS-MLLSTSVM
Flags	0.352689±0.023536	**0.315087±0.009075**	0.328975±0.013833	0.537068±0.000217
Emotions	0.352689±0.015893	0.269650±0.003902	0.233917±0.005581	**0.221290±0.004073**
Birds	0.121837±0.002642	0.105592±0.002625	0.101371±0.001201	**0.071802±0.001995**
Scene	0.256802±0.062719	0.099796±0.000977	0.122428±0.003183	**0.099763±0.001164**
Yeast	0.232863±0.001693	0.225726±0.002796	0.235961±0.000862	**0.212568±0.066639**

TABLE 6. One error of 4 algorithms on the benchmark datasets.

	BPMLL	Rank-SVM	MLTSVM	SS-MLLSTSVM
Flags	0.254057±0.027592	**0.197643±0.023565**	0.257734±0.029228	0.235789±0.011217
Emotions	0.624975±0.035892	0.383431±0.011459	0.312904±0.011281	**0.232938±0.022256**
Birds	0.760375±0.025837	0.644792±0.004402	0.837401±0.023330	**0.393114±0.030723**
Scene	0.819435±0.025464	0.277306±0.004466	0.260361±0.008587	**0.256075±0.003396**
Yeast	0.251554±0.058432	0.242496±0.002402	0.248029±0.005746	**0.241300±0.010886**

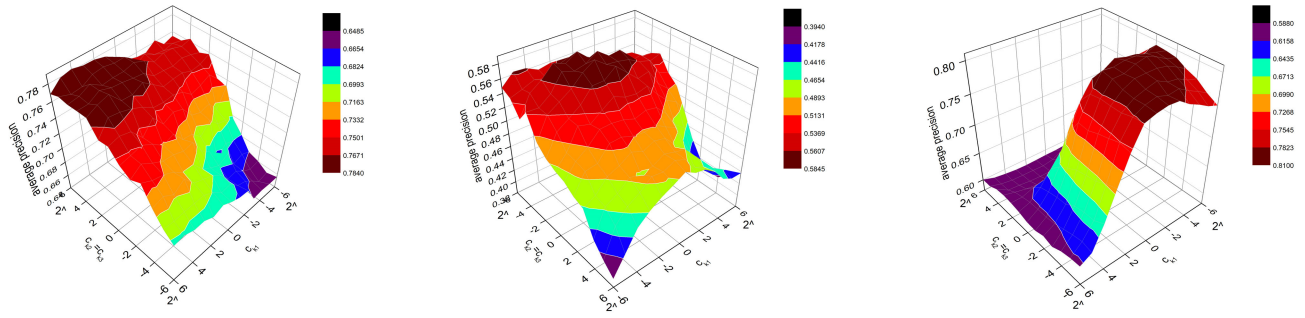
TABLE 7. Ranking loss of 4 algorithms on the benchmark datasets.

	BPMLL	Rank-SVM	MLTSVM	SS-MLLSTSVM
Flags	0.297491±0.024654	**0.212556±0.006700**	0.265753±0.010780	0.250772±0.007843
Emotions	0.392776±0.038706	0.252968±0.006085	0.216210±0.004670	**0.160475±0.003648**
Birds	0.314854±0.004965	0.302967±0.003972	0.338282±0.014664	**0.196370±0.009960**
Scene	0.438322±0.037855	0.102484±0.001378	0.087549±0.003061	**0.084058±0.004990**
Yeast	0.185285±0.002865	**0.175893±0.000286**	0.187133±0.000864	0.180623±0.001368

TABLE 8. Kappa of 4 algorithms on the benchmark datasets.

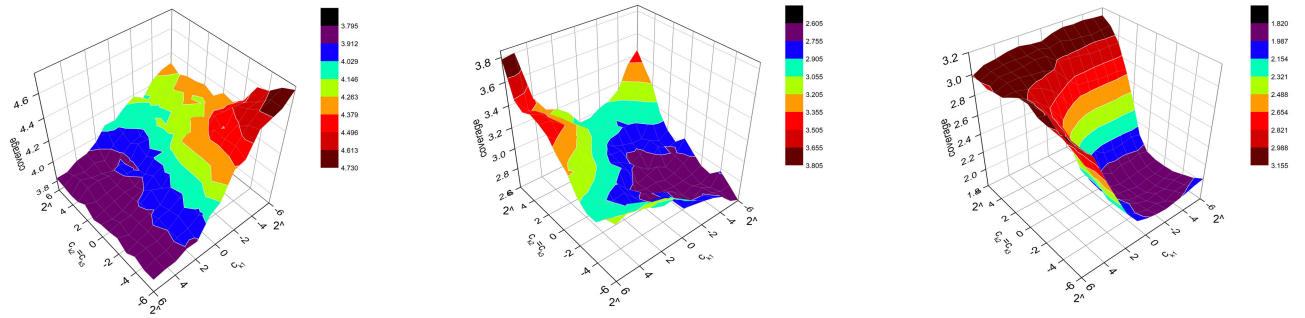
	BPMLL	Rank-SVM	MLTSVM	SS-MLLSTSVM
Flags	0.371319±0.079612	**0.412994±0.009232**	0.376198±0.009312	0.386901±0.001113
Emotions	0.398527±0.013574	0.415419±0.008018	0.463762±0.008136	**0.560522±0.003611**
Birds	0.534942±0.012876	0.540938±0.010764	0.409151±0.018452	**0.566587±0.020559**
Scene	0.391362±0.017265	0.713512±0.002670	0.651246±0.005800	**0.727719±0.006207**
Yeast	0.433074±0.017331	**0.491263±0.002749**	0.461694±0.003187	0.474906±0.004322

3.09, F_F (*one error*) = 7.36, F_F (*ranking loss*) = 5.62, F_F (*kappa*) = 8.60. For the significance level $\alpha = 0.10$, the critical values $F(3, 12) = 2.61$. Because F_F (*coverage*), F_F (*Hamming loss*), F_F (*one error*), F_F (*ranking loss*) and F_F (*kappa*) are larger than the critical values, 4 algorithms have significant differences for the 5 metrics.



(a) Flags (b) Birds (c) Emotions

FIGURE 1. The influence of the parameters c_{k1} and $c_{k2} = c_{k3}$ on average precision.



(a) Flags (b) Birds (c) Emotions

FIGURE 2. The influence of the parameters c_{k1} and $c_{k2} = c_{k3}$ on coverage.

TABLE 9. Ranks of 4 algorithms for coverage.

	BPMLL	Rank-SVM	MLTSVM	SS-MLLSTSVM
Flags	3	**1**	4	2
Emotions	4	3	2	**1**
Birds	3	2	4	**1**
Scene	4	3	2	**1**
Yeast	3	**1**	4	2
Average	3.4	2	3.2	**1.4**

TABLE 10. Ranks of 4 algorithms for Hamming loss.

	BPMLL	Rank-SVM	MLTSVM	SS-MLLSTSVM
Flags	3	**1**	2	4
Emotions	4	3	2	**1**
Birds	4	2	3	**1**
Scene	3	2	4	**1**
Yeast	3	2	4	**1**
Average	3.4	2	3	**1.6**

TABLE 11. Ranks of 4 algorithms for one error.

	BPMLL	Rank-SVM	MLTSVM	SS-MLLSTSVM
Flags	3	**1**	4	2
Emotions	4	3	2	**1**
Birds	3	2	4	**1**
Scene	4	3	2	**1**
Yeast	4	2	3	**1**
Average	3.6	2.2	3	**1.2**

From Table 9 to 13, we can see that the average rank of our SS-MLLSTSVM is lower than other algorithms, in other words, our SS-MLLSTSVM has better classification performance for the 5 metrics.

TABLE 12. Ranks of 4 algorithms for ranking loss.

	BPMLL	Rank-SVM	MLTSVM	SS-MLLSTSVM
Flags	4	**1**	3	2
Emotions	4	3	2	**1**
Birds	3	2	4	**1**
Scene	4	3	2	**1**
Yeast	3	**1**	4	2
Average	3.6	2	3	**1.4**

TABLE 13. Ranks of 4 algorithms for kappa.

	BPMLL	Rank-SVM	MLTSVM	SS-MLLSTSVM
Flags	4	**1**	3	2
Emotions	4	3	2	**1**
Birds	3	2	4	**1**
Scene	4	2	3	**1**
Yeast	4	**1**	3	2
Average	3.8	1.8	3	**1.4**

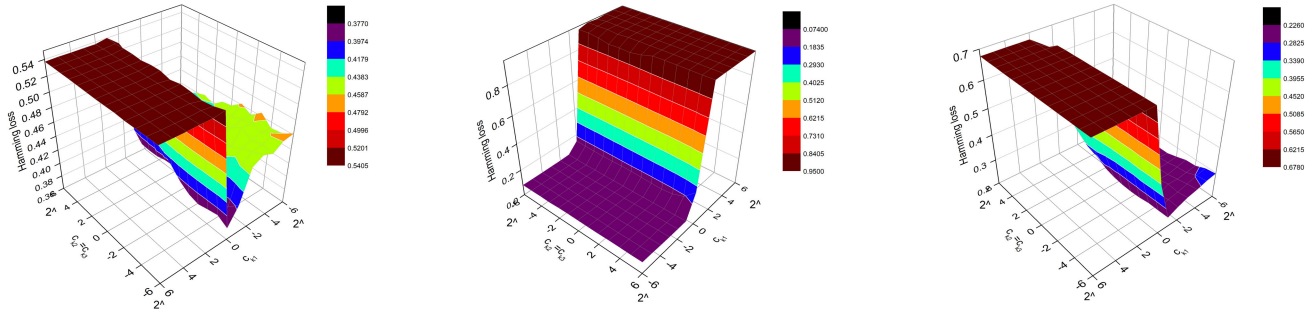
We present the training time of each algorithm in Table 14. From Table 14, we can observe that, compared with other algorithms, although SS-MLLSTSVM needs to calculate the Laplace matrix of the whole samples, our SS-MLLSTSVM still has higher leaning speed.

E. PARAMETERS ANALYSIS

In this subsection, we investigate the influence of parameters c_{k1}, c_{k2} and c_{k3} on the classification performance of the SS-MLLSTSVM. The results are shown in Figure 1 to 7.

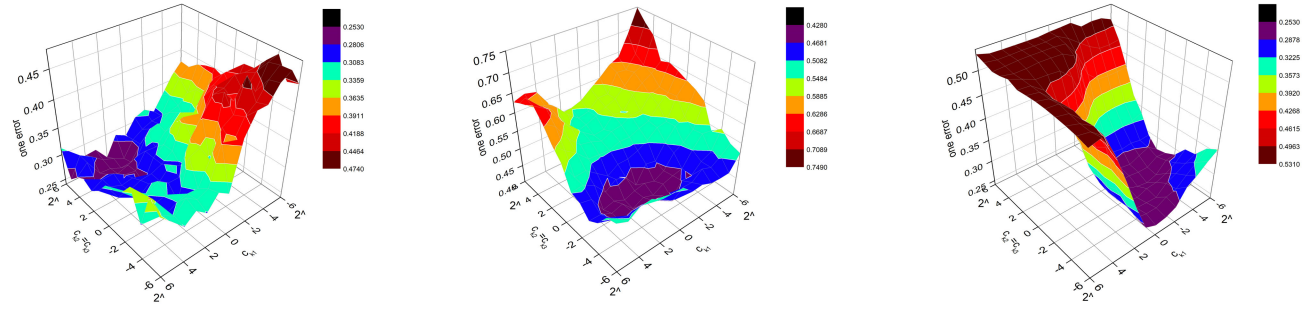
TABLE 14. Learning time of 4 algorithms on the benchmark datasets.

	BPMLL(s)	Rank-SVM(s)	MLTSVM(s)	SS-MLLSTSVM(s)
Flags	4.240823	0.379595	0.333238	**0.036964**
Emotions	18.376880	1.532958	0.421554	**0.271069**
Birds	17.648423	4.754735	**1.000000**	1.049683
Scene	163.495303	5.258294	**4.900547**	10.291504
Yeast	71.363476	49.277432	12.100704	**5.615573**
Average	55.024980	12.240602	3.751208	**3.452958**



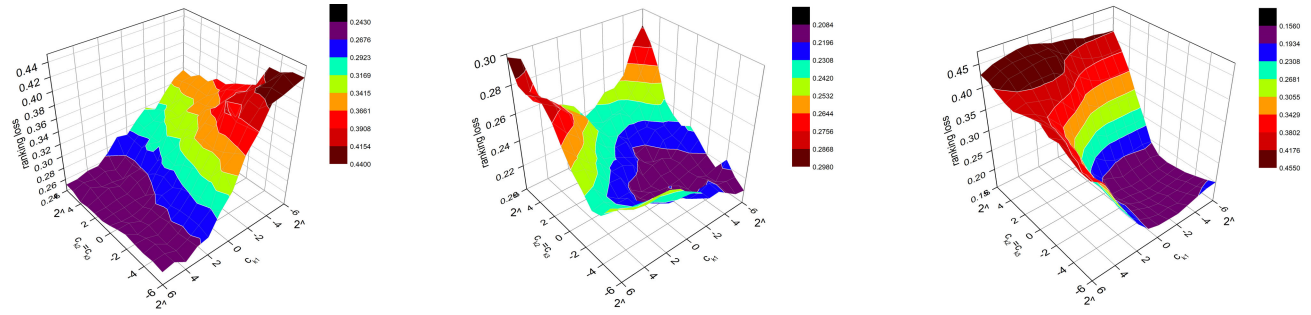
(a) Flags (b) Birds (c) Emotions

FIGURE 3. The influence of the parameters c_{k1} and $c_{k2} = c_{k3}$ on Hamming loss.



(a) Flags (b) Birds (c) Emotions

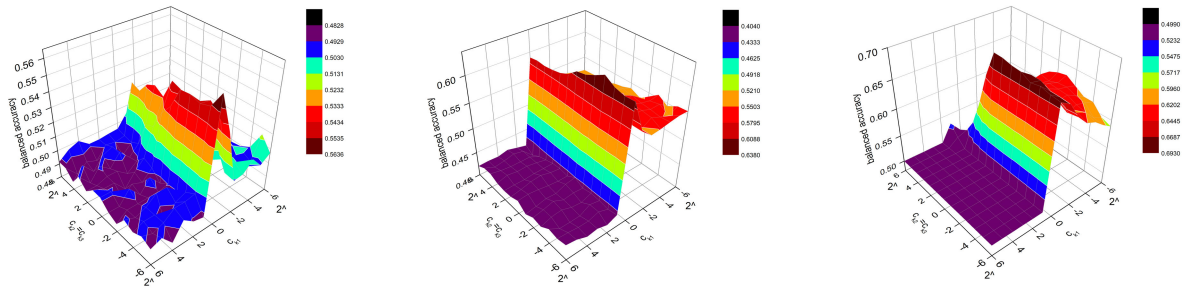
FIGURE 4. The influence of the parameters c_{k1} and $c_{k2} = c_{k3}$ on one error.



(a) Flags (b) Birds (c) Emotions

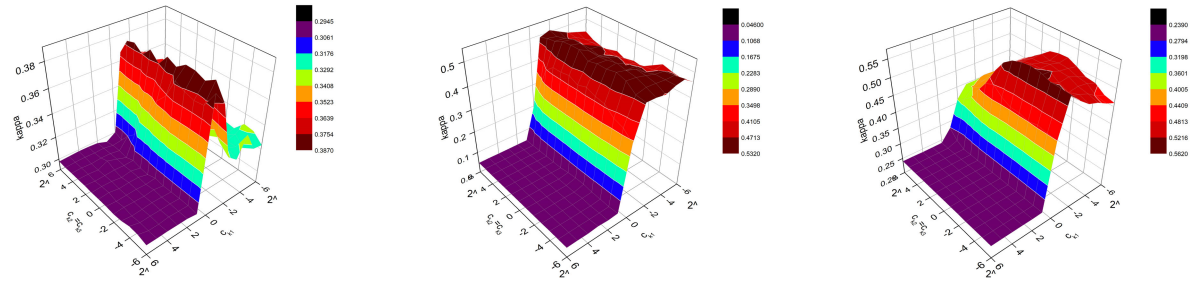
FIGURE 5. The influence of the parameters c_{k1} and $c_{k2} = c_{k3}$ on ranking loss.

From Figure 1 to 7, we can observe that (a) the classification metrics of the SS-MLLSTSVM vary dramatically with the change of the parameters, which means that the parameters have great influence on the classification metrics of the SS-MLLSTSVM; (b) for Hamming loss, balanced accuracy and kappa, the parameter c_{k1} has strong influence and the parameters c_{k2} and c_{k3} have weak influence, while there is no obvious difference among c_{k1}, c_{k2} and c_{k3} for other metrics.



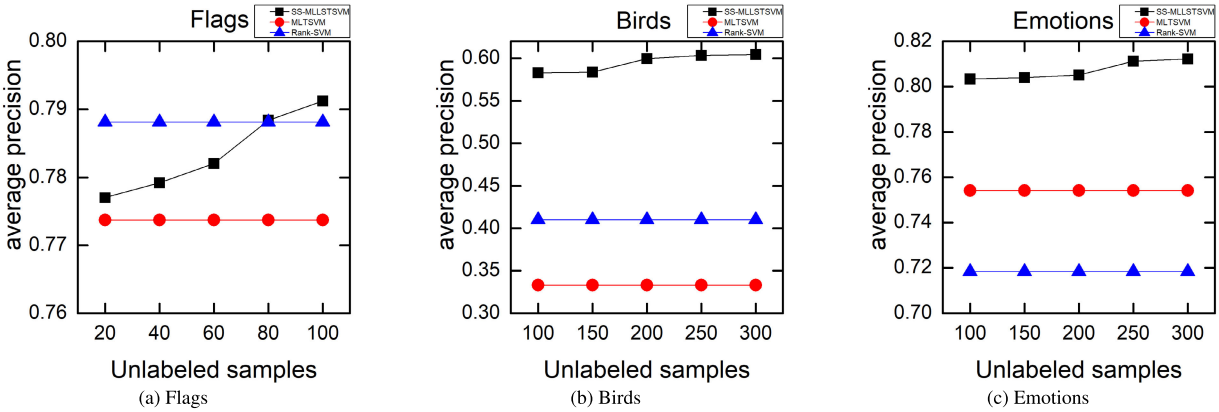
(a) Flags (b) Birds (c) Emotions

FIGURE 6. The influence of the parameters c_{k1} and $c_{k2} = c_{k3}$ on balanced accuracy.



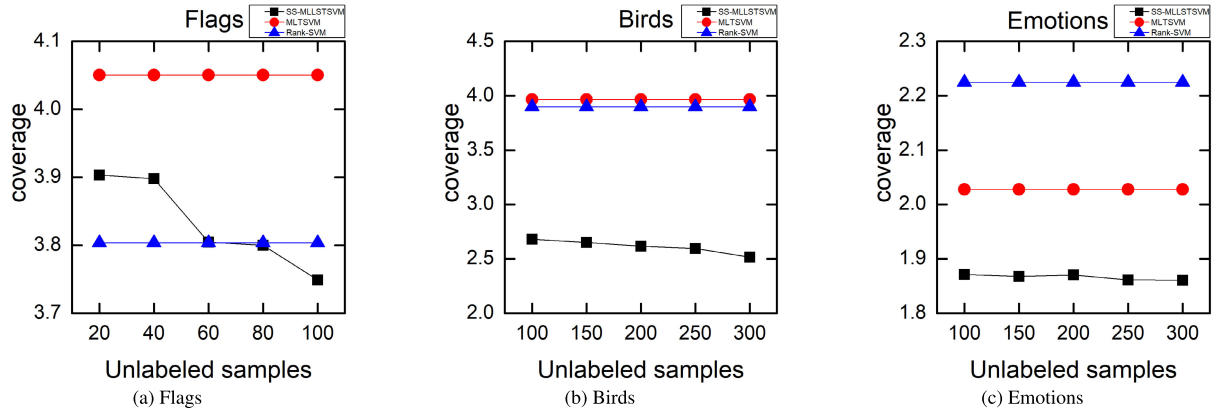
(a) Flags (b) Birds (c) Emotions

FIGURE 7. The influence of the parameters c_{k1} and $c_{k2} = c_{k3}$ on kappa.



(a) Flags (b) Birds (c) Emotions

FIGURE 8. Average precision of SS-MLLSTSVM, MLTSVM and Rank-SVM for different number of unlabeled samples.



(a) Flags (b) Birds (c) Emotions

FIGURE 9. Coverage of SS-MLLSTSVM, MLTSVM and Rank-SVM for different number of unlabeled samples.

F. SENSITIVITY ANALYSIS

In this subsection, we investigate the effect of size of unlabeled samples on classification performance. Figure 8 to 14 shows the classification results of MLTSVM, Rank-SVM and SS-MLLSTSVM on flags, birds and emotions for the different number of unlabeled samples.

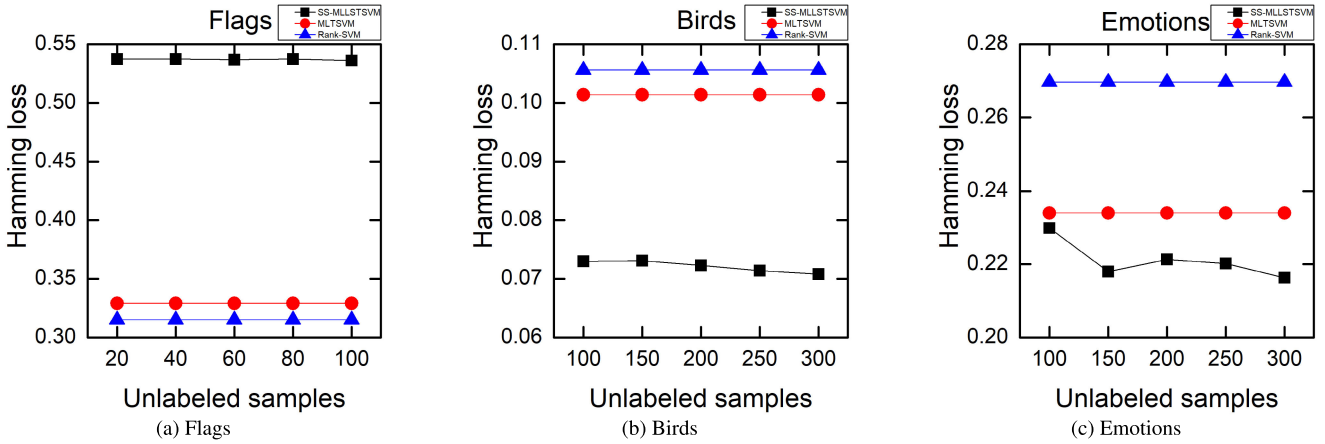


FIGURE 10. Hamming loss of SS-MLLSTSVM, MLTSVM and Rank-SVM for different number of unlabeled samples.

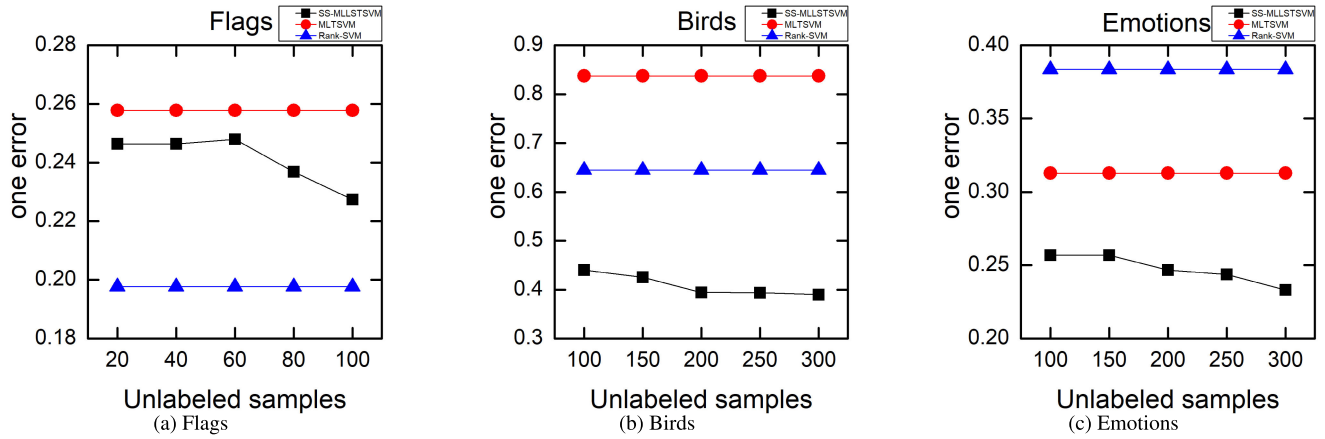


FIGURE 11. One error of SS-MLLSTSVM, MLTSVM and Rank-SVM for different number of unlabeled samples.

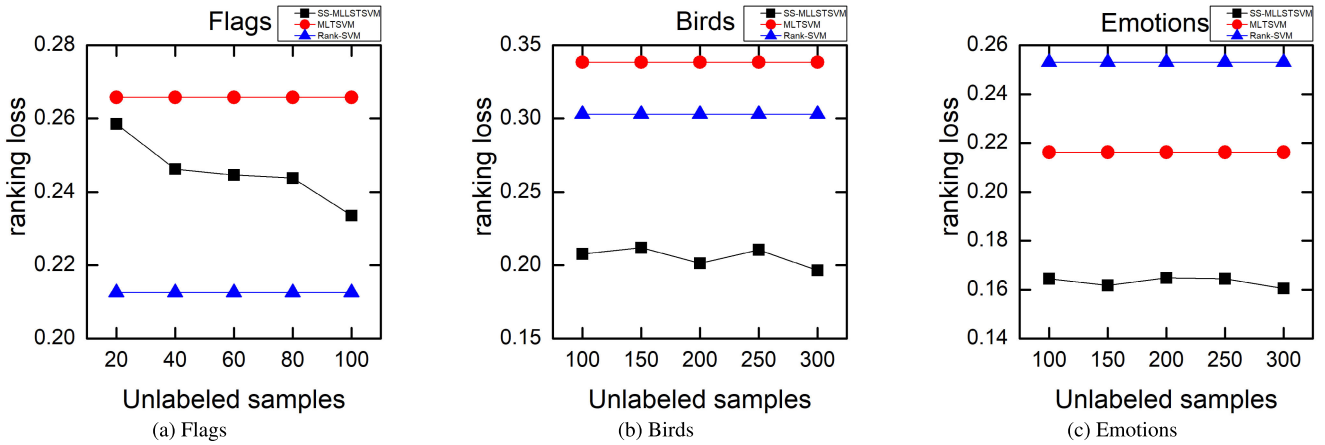


FIGURE 12. Ranking loss of SS-MLLSTSVM, MLTSVM and Rank-SVM for different number of unlabeled samples.

From Figure 8 to 14, we can observe that, with the increase of unlabeled samples, the classification metrics of the MLTSVM and Rank-SVM remain constant, that is mainly because MLTSVM and Rank-SVM can only use labeled samples, not unlabeled samples, while the classification metrics of SS-MLLSTSVM become better and better, that's mainly because, SS-MLLSTSVM can make full use of the valuable samples to construct a

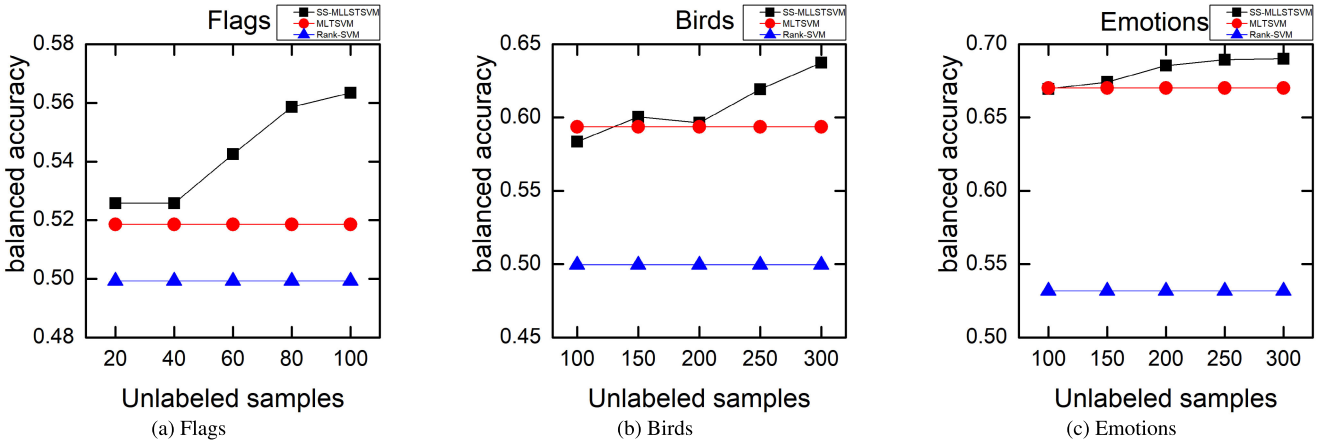


FIGURE 13. Balanced accuracy of SS-MLLSTSVM, MLTSVM and Rank-SVM for different number of unlabeled samples.

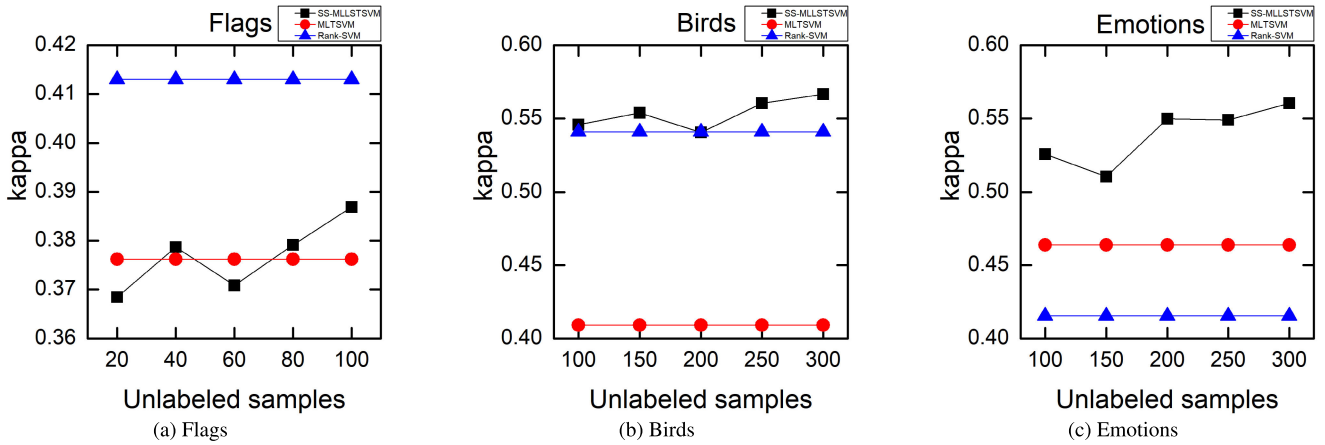


FIGURE 14. Kappa of SS-MLLSTSVM, MLTSVM and Rank-SVM for different number of unlabeled samples.

more reasonable classifier and improve the classification performance.

V. CONCLUSION

In this article, we propose an semi-supervised multi-label learning algorithm, named SS-MLLSTSVM. SS-MLLSTSVM introduces the least squares idea into each sub-classifier of MLTSVM to improve learning speed and make full use of the geometric information in unlabeled and partially labeled samples to improve generalization performance. The experimental results on the benchmark datasets indicate that, compared with popular multi-label classifiers, our SS-MLLSTSVM has better classification performance, especially for the dataset that contains a large number of partially labeled and unlabeled samples. The high-dimensional data have great effects on the classification performance. Therefore, feature reduction for multi-label learning will be the focus of our future research.

REFERENCES

[1] Jayadeva, R. Khemchandani, and S. Chandra, "Twin support vector machines for pattern classification," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 29, no. 5, pp. 905–910, May 2007.

[2] M. A. Kumar and M. Gopal, "Least squares twin support vector machines for pattern classification," *Expert Syst. Appl.*, vol. 36, no. 4, pp. 7535–7543, May 2009.

[3] J. Chen and G. Ji, "Weighted least squares twin support vector machines for pattern classification," in *Proc. 2nd Int. Conf. Comput. Automat. Eng. (ICCAE)*, Feb. 2010, pp. 242–246.

[4] A. Mir and J. A. Nasiri, "KNN-based least squares twin support vector machine for pattern classification," *Int. J. Speech Technol.*, vol. 48, no. 12, pp. 4551–4564, Dec. 2018.

[5] J. S. Sartakhti, H. Afrabandpey, and N. Ghadiri, "Fuzzy least squares twin support vector machines," *Eng. Appl. Artif. Intell.*, vol. 85, pp. 402–409, Oct. 2019.

[6] X. Xie, "Improvement on projection twin support vector machine," *Neural Comput. Appl.*, vol. 30, no. 2, pp. 371–387, Jul. 2018.

[7] X. Hua, S. Xu, J. Gao, and S. Ding, "L1-norm loss-based projection twin support vector machine for binary classification," *Soft Comput.*, vol. 23, no. 21, pp. 10649–10659, Nov. 2019.

[8] F. Xie and Y. Xu, "An efficient regularized K-nearest neighbor structural twin support vector machine," *Int. J. Speech Technol.*, vol. 49, no. 12, pp. 4258–4275, Dec. 2019.

[9] Q. Ai, A. Wang, Y. Wang, and H. Sun, "Improvements on twin-hypersphere support vector machine using local density information," *Prog. Artif. Intell.*, vol. 7, no. 3, pp. 167–175, Sep. 2018.

[10] Q. Ai, A. Wang, Y. Wang, and H. Sun, "An improved twin-KSVC with its applications," *Neural Comput. Appl.*, vol. 31, no. 10, pp. 6615–6624, Oct. 2019.

[11] Q. Ai, A. Wang, A. Zhang, Y. Wang, and H. Sun, "A multi-class classification weighted least squares twin support vector hypersphere using local density information," *IEEE Access*, vol. 6, pp. 17284–17291, 2018.

[12] M. Chu, L. Liu, Y. Yang, and R. Gong, "Twin support vector machine with local structural information for pattern classification," *IEEE Access*, vol. 6, pp. 64237–64249, 2018.
[13] H. Wang, Y. Xu, and Z. Zhou, "Twin-parametric margin support vector machine with truncated pinball loss," *Neural Comput. Appl.*, early access, Aug. 2020, doi: 10.1007/s00521-020-05225-7.
[14] J. A. Nasiri and A. M. Mir, "An enhanced KNN-based twin support vector machine with stable learning rules," *Neural Comput. Appl.*, vol. 32, no. 16, pp. 12949–12969, Aug. 2020.
[15] S. Rezvani, X. Wang, and F. Pourpanah, "Intuitionistic fuzzy twin support vector machines," *IEEE Trans. Fuzzy Syst.*, vol. 27, no. 11, pp. 2140–2151, Nov. 2019.
[16] H. Wang, Z. Zhou, and Y. Xu, "An improved nu-twin bounded support vector machine," *Appl. Intell.*, vol. 48, no. 4, pp. 1041–1053, 2018.
[17] L. Liu, M. Chu, Y. Yang, and R. Gong, "Twin support vector machine based on adjustable large margin distribution for pattern classification," *Int. J. Mach. Learn. Cybern.*, vol. 11, no. 10, pp. 2371–2389, Oct. 2020.
[18] Q. Hou, J. Zhang, L. Liu, Y. Wang, and L. Jing, "Discriminative information-based nonparallel support vector machine," *Signal Process.*, vol. 162, pp. 169–179, Sep. 2019.
[19] H. Peng, J. Li, S. Wang, L. Wang, Q. Gong, R. Yang, B. Li, P. Yu, and L. He, "Hierarchical taxonomy-aware and attentional graph capsule RCNNs for large-scale multi-label text classification," *IEEE Trans. Knowl. Data Eng.*, early access, Dec. 16, 2019, doi: 10.1109/TKDE.2019.2959991.
[20] T. Wang, L. Liu, N. Liu, H. Zhang, L. Zhang, and S. Feng, "A multi-label text classification method via dynamic semantic representation model and deep neural network," *Int. J. Speech Technol.*, vol. 50, no. 8, pp. 2339–2351, Aug. 2020.
[21] Y. Guo, F.-L. Chung, G. Li, and L. Zhang, "Multi-label bioinformatics data classification with ensemble embedded feature selection," *IEEE Access*, vol. 7, pp. 103863–103875, 2019.
[22] Y. Liu, K. Wen, Q. Gao, X. Gao, and F. Nie, "SVM based multi-label learning with missing labels for image annotation," *Pattern Recognit.*, vol. 78, pp. 307–317, Jun. 2018.
[23] E. A. Cherman, M. C. Monard, and J. Metz, "Multi-label problem transformation methods: A case study," *CLEI Electron. J.*, vol. 14, no. 1, pp. 1–10, Apr. 2011.
[24] M. R. Boutell, J. Luo, X. Shen, and C. M. Brown, "Learning multi-label scene classification," *Pattern Recognit.*, vol. 37, no. 9, pp. 1757–1771, Sep. 2004.
[25] G. Tsoumakas and I. Vlahavas, "Random k-labelsets: An ensemble method for multilabel classification," in *Proc. 18th Eur. Conf. Mach. Learn.* Berlin, Germany: Springer, 2007, pp. 406–417.
[26] J. Read, B. Pfahringer, G. Holmes, and E. Frank, "Classifier chains for multi-label classification," *Mach. Learn.*, vol. 85, no. 3, pp. 333–359, Dec. 2011.
[27] J. Fürnkranz, E. Hüllermeier, E. L. Mencía, and K. Brinker, "Multilabel classification via calibrated label ranking," *Mach. Learn.*, vol. 73, no. 2, pp. 133–153, Nov. 2008.
[28] A. Elisseeff and J. Weston, "A kernel method for multi-labelled classification," in *Proc. Adv. Neural Inf. Process. Syst.*, 2001, pp. 681–687.
[29] N. Ghamrawi and A. McCallum, "Collective multi-label classification," in *Proc. 14th ACM Int. Conf. Inf. Knowl. Manage. (CIKM)*, 2005, pp. 195–200.
[30] M.-L. Zhang and Z.-H. Zhou, "ML-KNN: A lazy learning approach to multi-label learning," *Pattern Recognit.*, vol. 40, no. 7, pp. 2038–2048, Jul. 2007.
[31] A. Clare and R. D. King, "Knowledge discovery in multi-label phenotype data," in *Proc. Eur. Conf. Princ. Data Mining Knowl. Discovery (PKDD)*, 2001, pp. 42–53.
[32] M.-L. Zhang and Z.-H. Zhou, "Multilabel neural networks with applications to functional genomics and text categorization," *IEEE Trans. Knowl. Data Eng.*, vol. 18, no. 10, pp. 1338–1351, Oct. 2006.
[33] W.-J. Chen, Y.-H. Shao, C.-N. Li, and N.-Y. Deng, "MLTSVM: A novel twin support vector machine to multi-label learning," *Pattern Recognit.*, vol. 52, pp. 61–74, Apr. 2016.
[34] Z. Hanifelou, P. Adibi, S. A. Monadjemi, and H. Karshenas, "KNN-based multi-label twin support vector machine with priority of labels," *Neurocomputing*, vol. 322, pp. 177–186, Dec. 2018.
[35] M. Azad-Manjiri, A. Amiri, and A. S. Sedghpour, "ML-SLSTSVM: A new structural least square twin support vector machine for multi-label learning," *Pattern Anal. Appl.*, vol. 23, no. 1, pp. 295–308, Feb. 2020.
[36] J. M. Keller, M. R. Gray, and J. A. Givens, "A fuzzy K-nearest neighbor algorithm," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-15, no. 4, pp. 580–585, Aug. 1985.
[37] G. Safont, A. Salazar, and L. Vergara, "Vector score alpha integration for classifier late fusion," *Pattern Recognit. Lett.*, vol. 136, pp. 48–55, Aug. 2020.

QING AI received the B.S. and M.S. degrees in 2003 and 2007, respectively. He is currently pursuing the Ph.D. degree in pattern recognition and intelligent system with Northeastern University, China. He is currently an Associate Professor with the University of Science and Technology, Liaoning, China. His research interests include pattern recognition, support vector machines, optimization theory and applications, and fault diagnosis.

YUDE KANG received the B.S. degree in 2018. He is currently pursuing the M.S. degree with the School of Computer Science and Software Engineering, University of Science and Technology, Liaoning. His research interests are machine learning and pattern recognition.

ANNA WANG received the Ph.D. degree from Northeastern University, China. She is currently a Professor with the College of Information Science and Engineering, Northeastern University. Her research interests include electrical signal processing, fault diagnosis, and pattern recognition.

XIANGNA LI received the B.S. and M.S. degrees in 2005 and 2008, respectively. She is currently with State Grid Information and Telecommunication Group Company Ltd., Beijing FibrLink Communications Company Ltd. She has been engaged in the information system construction and operation and maintenance of State Grid Information and Telecommunication Group Company Ltd., for a long time, where she became a Deputy Senior Engineer in 2016.

FEI LI received the B.S. degree from the Anhui University of Science and Technology in 2020. He is currently pursuing the M.S. degree with the School of Computer Science and Software Engineering, University of Science and Technology, Liaoning. His research interests are machine learning and pattern recognition.