IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received September 2, 2020, accepted September 13, 2020, date of publication September 18, 2020,
date of current version September 29, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3024745

An Efficient Multi-Label SVM Classification Algorithm by Combining Approximate Extreme Points Method and Divide-and-Conquer Strategy

ZHONGWEI SUN[1], XIUYAN LIU[1], KEYONG HU[1], (Member, IEEE), ZHUANG LI[1], AND JING LIU[2]

[1]The School of Information and Control Engineering, Qingdao University of Technology, Qingdao 266520, China
[2]The School of Science and Information, Qingdao Agriculture University, Qingdao 266109, China

Corresponding author: Keyong Hu (hukeyongouc@163.com)

This work was supported by the National Natural Science Foundation of China under Grant 61902205.

ABSTRACT Excessive time complexity has severely restricted the application of support vector machine (SVM) in large-scale multi-label classification. Thus, this paper proposes an efficient multi-label SVM classification algorithm by combining approximate extreme points method and divide-and-conquer strategy (AEDC-MLSVM). The AEDC-MLSVM classification algorithm firstly uses the approximate extreme points method to obtain the representative set from the multi-label training data set. While persisting almost all the useful information of multi-label training set, representative set can effectively reduce the scale of multi-label training set. Secondly, to acquire an efficient multi-label SVM classification model, the SVM based on the improved divide-and-conquer strategy is trained on the representative set, which will further improve the training speed and classification performance. The improvement is reflected in two aspects. (1) The improved divide-and-conquer strategy is applied to divide the representative set into subsets and this can ensure that each representative subset contains a certain number of positive and negative instances. This will avoid singular problems and overcome computation load imbalance problem. (2) The different error cost (DEC) method is applied to overcome the label imbalance problem. Effective experiments have proved that the training and testing speed of AEDC-MLSVM classification algorithm can be accelerated substantially while ensuring the classification performance.

INDEX TERMS Multi-label classification, approximate extreme points method, divide-and-conquer strategy, support vector machine, label imbalance.

I. INTRODUCTION

Compared with traditional binary or multi-class classification, multi-label classification is different in that each instance can have multiple labels and thus these labels are no longer mutually exclusive [1]. Many methods have been proposed to solve the multi-label classification problem, including SVM method, decision tree method, neural network method, K-nearest neighbor method, etc [2]. These methods have been widely recognized and successfully solved many real-world practical problems, such as image video semantic annotation [3], [4], text categorization [5], music emotion classification [6], bioinformatics prediction [7] and so on. With the arrival of big data era, many real-world applications need to be implemented in large-scale multi-label data sets. However, many existing multi-label classification methods cannot be applied to large-scale multi-label data sets effectively. The main reason for this problem is that these methods are severely restricted by the excessive time complexity. This phenomenon is especially evident in SVM. In this paper, we will focus on the research of efficient multi-label SVM classification methods.

SVM [8] is an extraordinary well-known machine learning method, which has been applied successfully in face detection, handwritten recognition, text categorization, etc [9]. Traditional SVM only can solve the single-instance single-label classification problem, but the improved SVM algorithms, such as Rank-SVM [10] algorithm, can be applied in

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

multi-label classification. However, many real-world multi-label data sets are non-linear. Hence, to obtain competitive performance, SVM needs non-linear kernel to train these multi-label data sets, which further limits the use of multi-label SVM classification algorithm in large-scale data sets. In addition, the multi-label SVM classification algorithm cannot avoid the problem that the vast majority of multi-label data sets are suffering from a serious label imbalance problem [11], which will seriously affect the classification performance.

The main contributions of this paper are as follows:

(1) The proposed AEDC-MLSVM classification algorithm can solve the problem that the application of multi-label SVM classification algorithm in large-scale data sets is seriously restricted by the excessive time complexity.

(2) The principle of the proposed SVM by combining approximate extreme points method and divide-and-conquer strategy (AEDC-SVM) is shown in FIGURE 2. The proposed AEDC-SVM can not only ensure the classification performance, but also reduce greatly the size of the training set and the negative impact of label imbalance problem, solve the computation load imbalance problem and prevent singular problems. This further improves the applicability of AEDC-MLSVM classification algorithm in large-scale data sets.

(3) The experimental results in three public real-world data sets show that the training and testing time of AEDC-MLSVM algorithm is the shortest compared with that of the existing multi-label classification algorithms such as ML-LIBSVM [12], ML-CVM [13] and ML-BVM [14]. And the performance of AEDC-MLSVM algorithm on the five evaluation indexes is pretty close to that of ML-LIBSVM and better than that of ML-CVM and ML-BVM.

The rest of this paper is organized as follows. Chapter 2 will introduce some related works. The new AEDC-MLSVM classification algorithm is proposed in chapter 3. After that, the analysis of the experiment results is presented in chapter 4. Chapter 5 is the summary of this paper.

II. RELATED WORK

From the first, multi-label classification has been widely concerned by experts in machine learning, pattern recognition, statistics and so on. In view of different practical problems, various kinds of multi-label classification methods have been proposed and achieved good effect. These multi-label classification methods can be summarized as the following two main strategies: problem transformation strategy and algorithm adaptation strategy [2]. Moreover, many methods have been proposed to solve the problem of label imbalance in multi-label classification. This chapter will firstly introduce the existing multi-label classification methods according to the two strategies, and then introduce current methods of processing the label imbalance problem.

The problem transformation strategy is mainly to transform a multi-label classification problem into several single-label classification problems. As a result, this type of multi-label classification methods is mainly achieved by the combination of problem transformation skill and existing single-label classification methods. Problem transformation skills mainly contain binary relevance (BR), one-by-one (OBO), one-versus-one (OVO) and label powerset (LP) [2], etc. Frequently-used single-label classification methods contain SVM, decision tree, neural network, nearest neighbor and so on [2].

In [15], three main defects of BR problem transformation strategy are described. First of all, since it assumes that the labels are independent, dependencies among labels will not be exploited. Secondly, it will likely cause the label imbalance problem. Finally, as the number of labels increases, the label imbalance problem will aggravate with the increase of classifiers. Despite of above problems, the BR problem transformation strategy is still considered to be simple and practical, and the data set can be reconstructed. In [16], the author highlights its superiority. Firstly, any single-label classifier can be used as the base classifier to accomplish the multi-label classification. Secondly, its complexity is lower than other methods, and its complexity is linear with the number of labels. Thirdly, because of the independence among labels, it can be easily parallelized. Finally, the advantage that needs to be emphasized is that it can optimize multiple loss functions. Thus, this paper will use the famous BR problem transformation strategy to accomplish multi-label classification.

Algorithm adaptation strategy accomplishes multi-label classification by improving single-label classification algorithms. By improving the information entropy formula and setting the leaf nodes as a label set, C4.5 type multi-label classification algorithm is proposed in [17]. This multi-label classification algorithm is suitable for small-scale data sets. Rank-SVM [10] algorithm accomplishes multi-label classification by minimizing the ranking loss of multi-class SVM, which will cause an extremely complex quadratic programming problem. To overcome the high time complexity problem of Rank-SVM algorithm, Rank-CVM [18] and Rank-CVMz [19] algorithms are proposed by adopting core vector machine (CVM) and zero label, which can improve the training speed to a certain extent, but reduce the classification effect. By combining the advantages of Ranking support vector machine and Binary Relevance with robust Low-rank learning, RBRL [32] algorithm is proposed. This algorithm can solve the optimization problem efficiently by adopting two accelerated proximal gradient (APG) methods. The ML-SLSTSVM [33] algorithm improves the MLTSVM algorithm by introducing structural information of training instances and adopting least square method. Although RBRL and ML-SLSTSVM algorithms can improve the classification performance, it can only be applied to small-scale data sets. ML-KNN [20] algorithm which is based on the k nearest neighbor (KNN) can achieve label prior probability and conditional probability by independently using the discrete binary bayes rule for each label. The BP-MLL classification algorithm [21] can express the multi-label features by constructing a new empirical loss function. These algorithms are difficult to be applied to large-scale data sets.

In [22], the influence of label imbalance problem on various classification algorithms is introduced in detail. In order to overcome this problem, many countermeasures have been proposed and have achieved good effect. These countermeasures can be summarized as the following three mainstream methods: resampling method [22], instance-based method [23] and cost sensitive method [24]. The DEC method adopted in this paper is a specific implementation of cost sensitive method.

To sum up, the use of existing multi-label classification algorithms in large-scale data sets is seriously limited by heavy time complexity and this phenomenon is severer for these algorithms based on SVM. The AEDC-MLSVM classification algorithm proposed in this paper will be a good solution to this problem. It not only can shorten the time consumption of training and testing greatly, but also achieve the classification performance close to that of ML-LIBSVM classification algorithm, and better than that of ML-CVM and ML-BVM classification algorithms. In addition, the DEC method is adopted to reduce the impact of label imbalance problem.

III. AN EFFICIENT MULTI-LABEL SVM CLASSIFICATION ALGORITHM BY COMBINING APPROXIMATE EXTREME POINTS METHOD AND DIVIDE-AND-CONQUER STRATEGY

In this chapter, we will firstly introduce the BR problem transformation strategy. Secondly, we will explain the principle of approximate extreme points method. Thirdly, the AEDC-SVM is elaborated in detail. Fourthly, we will design and implement the AEDC-MLSVM classification algorithm. Finally, we will analyze the time and space complexity of the AEDC-MLSVM classification algorithm.

A. BINARY RELEVANCE PROBLEM TRANSFORMATION STRATEGY

Suppose that $D = \{(\boldsymbol{x}_i, Y_i) | \boldsymbol{x}_i \in \mathbb{R}^d, Y_i \subseteq Q, i = 1, 2, \cdots, N\}$ represents a multi-label training data set, $\boldsymbol{x}_i$ represents a data vector of d feature values, $Q = \{q_1, \cdots, q_k\}$ represents the label set. First of all, the BR problem transformation strategy is to transform the multi-label training data set D into k independent binary training subsets, i.e. $D_{q_j} = \{(\boldsymbol{x}_i, y_i) | \boldsymbol{x}_i \in \mathbb{R}^d, y_i \in \{-1, 1\}, i = 1, 2, \cdots, N \text{ and } j = 1, 2, \cdots, k\}$. The principle is as follows: for each multi-label training instance $(\boldsymbol{x}_i, Y_i)$, if $q_j \in Y_i, \boldsymbol{x}_i$ is regarded as a positive training instance of D_{q_j}, i.e. $y_i = 1$ and a negative training instance otherwise, i.e. $y_i = -1$. After that, by training on each binary training subset D_{q_j}, corresponding binary classifier $h_{q_j}(\boldsymbol{x})$ is constructed. Finally, the BR problem transformation strategy integrates the results of the k binary classifiers to realize multi-label classification, and its formula is as below.

$$\begin{aligned} \boldsymbol{y} &= \boldsymbol{h}(\boldsymbol{x}) \\ [q_1, \cdots, q_k] &= h_{q_1}(\boldsymbol{x}), \cdots, h_{q_k}(\boldsymbol{x}) \end{aligned} \tag{1}$$

In order to utilize the BR problem transformation strategy to realize multi-label classification effectively, the following decision function is used to integrate all binary classification results. The decision function is as below.

$$Y = \{q_j, s.t. h_{q_j}(\boldsymbol{x}) \geq 0, \forall q_j \in Q, j = 1, \cdots, k\} \tag{2}$$

Meanwhile, the following rule is used to avoid obtaining an empty relevant label set. The equation is as below.

$$Y = \{q_j, s.t. \max h_{q_j}(\boldsymbol{x}), \forall h_{q_j}(\boldsymbol{x}) < 0, q_j \in Q\} \tag{3}$$

From chapter 2, we know that the BR problem transformation strategy is effective and practical, and it can be effectively applied to large-scale multi-label classification. Consequently, the BR problem transformation strategy will be used to realize multi-label classification.

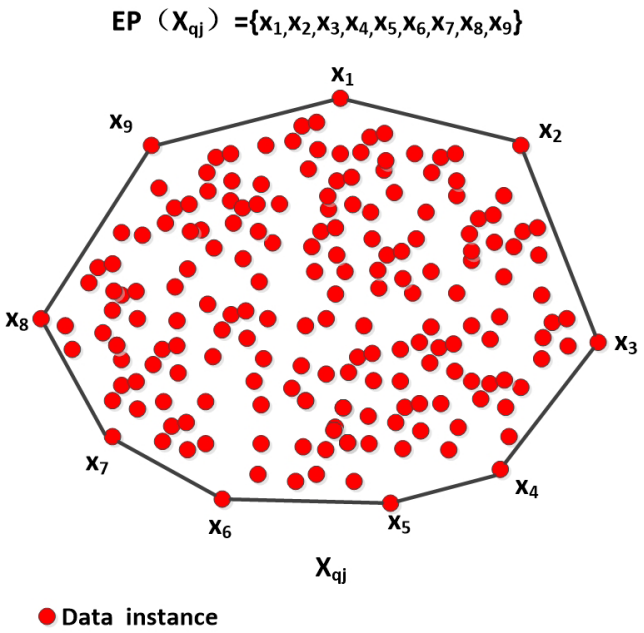

FIGURE 1. the principle of extreme points.

B. THE PRINCIPLE OF APPROXIMATE EXTREME POINTS METHOD

Before explaining the principle of approximate extreme points method, we convert D_{q_j} (defined in the previous subchapter) into $\boldsymbol{X}_{q_j} = \{\boldsymbol{x}_i | \boldsymbol{x}_i \in \mathbb{R}^d, i = 1, 2, \cdots, N\}$ and $Y_{q_j} = \{y_i : y_i \in \{-1, 1\}, i = 1, 2, \cdots, N\}$. Since the approximate extreme points method is based on the extreme points principle, we will firstly introduce the extreme points principle [25], [26]. It can be seen from FIGURE 1 that any vector $\boldsymbol{x}_i$ in $\boldsymbol{X}_{q_j}$ can be represented by a convex combination vector set $EP(\boldsymbol{X}_{q_j})$, and its formula is as below.

$$\begin{aligned} &\boldsymbol{x}_i = \sum_{\boldsymbol{x}_t \in EP(\boldsymbol{X}_{q_j})} \beta_{i,t} \boldsymbol{x}_t, \\ &where\ 0 \leq \beta_{i,t} \leq 1,\ and \sum_{\boldsymbol{x}_t \in EP(\boldsymbol{X}_{q_j})} \beta_{i,t} = 1. \end{aligned} \tag{4}$$

It can be seen from formula 4 that any vector $\boldsymbol{x}_i$ in $\boldsymbol{X}_{q_j}$ can be obtained only by using $EP(\boldsymbol{X}_{q_j})$ and convex combination weight parameter set$\{\beta_{i,t}\}$. Therefore, we define $EP(\boldsymbol{X}_{q_j})$ as the extreme points set of $\boldsymbol{X}_{q_j}$. $EP(\boldsymbol{X}_{q_j})$ not only contains almost all the important information of $\boldsymbol{X}_{q_j}$, but also its quantity is far less than that of $\boldsymbol{X}_{q_j}$. Thus training SVM on $EP(\boldsymbol{X}_{q_j})$ can greatly improve the training and testing speed on the basis of ensuring the classification performance. However, when facing large-scale training data set, the solution complexity of extreme points method is high. For this reason, the approximate extreme points method is proposed. Before that, we assume that the kernel space transformation set of $\boldsymbol{X}_{q_j}$ is $\boldsymbol{A}$, i.e., $\boldsymbol{A} = \{\boldsymbol{a}_i | \boldsymbol{a}_i = \Phi(\boldsymbol{x}_i), \forall \boldsymbol{x}_i \in \boldsymbol{X}_{q_j}, \Phi : \mathbb{R}^d \longrightarrow H\}$, here $\Phi(\boldsymbol{x}_i)$ represents the explicit representation of $\boldsymbol{x}_i$ in kernel space. $\boldsymbol{A}$ can be decomposed into l disjoint subsets $\boldsymbol{A}_l$, $l = \{1, 2, \cdots, \frac{N}{|\boldsymbol{A}_l|}\}$, i.e., $\boldsymbol{A} = \bigcup_l \boldsymbol{A}_l$, for simplicity, we assume that N can be divided by $|\boldsymbol{A}_l|$. When $i \neq j$, for $\forall \boldsymbol{a}_i, \boldsymbol{a}_j \in \boldsymbol{A}_l$, we can obtain $y_i = y_j$. Here, $|\boldsymbol{A}_l|$ represents the number of training instances in $\boldsymbol{A}_l$. $\boldsymbol{A}_{lg}$ denotes an arbitrary subset of $\boldsymbol{A}_l$, i.e., $\boldsymbol{A}_{lg} \subseteq \boldsymbol{A}_l$. For $\forall \boldsymbol{a}_i \in \boldsymbol{A}_l$, the following formula is obtained according to the extreme points principle.

$$h(\boldsymbol{a}_i, \boldsymbol{A}_{lg}) = \min_{\bar{\rho}_i} \|\boldsymbol{a}_i - \sum_{\boldsymbol{a}_t \in \boldsymbol{A}_{lg}} \rho_{i,t} \boldsymbol{a}_t\|^2, \quad s.t.\ 0 \leq \rho_{i,t} \leq 1,\ and \sum_{\boldsymbol{a}_t \in \boldsymbol{A}_{lg}} \rho_{i,t} = 1. \tag{5}$$

$\boldsymbol{A}_l^*$ can be defined as an approximate extreme points set of $\boldsymbol{A}_l$,if it satisfies the following formula.

$$\max_{\boldsymbol{a}_i \in \boldsymbol{A}_l} h(\boldsymbol{a}_i, \boldsymbol{A}_l^*) \leq \varepsilon. \tag{6}$$

Therefore, the representative set $\boldsymbol{A}^*$ of $\boldsymbol{A}$ can be obtained as follows.

$$\boldsymbol{A}^* = \bigcup_{l=1}^{\frac{N}{|\boldsymbol{A}_l|}} \boldsymbol{A}_l^* \tag{7}$$

We can get the representative set of $\boldsymbol{X}_{q_j}$ as $\boldsymbol{X}_{q_j}^* := \{\boldsymbol{x}_t | \boldsymbol{a}_t \in \boldsymbol{A}^*, \boldsymbol{x}_t \in \mathbb{R}^d, t = 1, 2, \cdots, M\}$ and its label set as $Y_{q_j}^* = \{y_t | y_t \in \{-1, 1\}, t = 1, 2, \cdots, M\}$. The data size of $\boldsymbol{X}_{q_j}^*$ is much smaller than that of $\boldsymbol{X}_{q_j}$, and it contains almost all important information of $\boldsymbol{X}_{q_j}$. The time complexity of representative set solution is linear with the size of $\boldsymbol{X}_{q_j}$.

C. SVM BY COMBINING APPROXIMATE EXTREME POINTS METHOD AND DIVIDE-AND-CONQUER STRATEGY

The obtained $\boldsymbol{X}_{q_j}^*$ and its corresponding label set $Y_{q_j}^*$ are applied to train SVM. The primal optimization problem of SVM can be transformed into the following quadratic optimization problem, the equation is as below.

$$\min_{\boldsymbol{\alpha}} h(\boldsymbol{\alpha}) = \frac{1}{2}\boldsymbol{\alpha}^T \kappa \boldsymbol{\alpha} - \boldsymbol{e}^T \boldsymbol{\alpha}, \quad s.t.\ 0 \leq \alpha_t \leq C \tag{8}$$

Equation 8 is a standard C-SVM model. The parameter C is used to balance the model complexity and the sum of losses of training data set, $\boldsymbol{\alpha} \in \mathbb{R}^M$ is the vector of dual variables. $\boldsymbol{e} \in 1^M$ is a vector of all ones, α_t is the t-th dual variable. κ is a $M \times M$ matrix, $\kappa_{tf} = y_t y_f K(\boldsymbol{x}_t, \boldsymbol{x}_f)$, where $K(\boldsymbol{x}_t, \boldsymbol{x}_f)$ is the kernel function. By solving equation 8, we can get the optimal solution $\boldsymbol{\alpha}^*$. Then we search for the α_t^* of $\boldsymbol{\alpha}^*$ in section $(0, C)$ to calculate b^*, the equation is as below.

$$b^* = y_t - \sum_{t=1}^{M} y_t \alpha_t^* K(\boldsymbol{x}_t, \boldsymbol{x}_f) \tag{9}$$

Finally, we construct decision function $h_{q_j}(\boldsymbol{x})$ to realize classification and the equation is as below.

$$h_{q_j}(\boldsymbol{x}) = sgn(\sum_{t=1}^{M} y_t \alpha_t^* K(\boldsymbol{x}_t, \boldsymbol{x}) + b^*) \tag{10}$$

Although SVM using approximate extreme points method has good classification performance, its use in large-scale data sets will be still restricted by excessive computational complexity. In [27], the author proposed the DC-SVM algorithm, in which the divide-and-conquer strategy is used and the training speed is improved greatly. But it has the following problems: firstly, the whole problem is partitioned by the unsupervised kernel kmeans clustering method, which will easily lead to the singular problems; secondly, it is difficult to balance the computation among subproblems. Therefore, we will propose an algorithm in which the approximate extreme points method and divide-and-conquer strategy are combined to improve SVM, namely AEDC-SVM. FIGURE 2 shows the improvement of AEDC-SVM. The steps are as follows.

(1) AEDC-SVM uses the approximate extreme points method to obtain the representative set $\boldsymbol{X}_{q_j}^*$ and its corresponding label set $Y_{q_j}^*$.

(2) Divide $\boldsymbol{X}_{q_j}^*$ into $\boldsymbol{X}_{q_j}^{*+}$ and $\boldsymbol{X}_{q_j}^{*-}$ according to the positive and negative labels.

(3) The kernel kmeans algorithm is used to obtain w cluster centers, i.e. $\{\boldsymbol{c}_1^+, \cdots, \boldsymbol{c}_w^+\}$, $\{\boldsymbol{c}_1^-, \cdots, \boldsymbol{c}_w^-\}$, and w cluster subsets i.e. $\{\boldsymbol{V}_1^+, \cdots, \boldsymbol{V}_w^+\}$, $\{\boldsymbol{V}_1^-, \cdots, \boldsymbol{V}_w^-\}$ of $\boldsymbol{X}_{q_j}^{*+}$ and $\boldsymbol{X}_{q_j}^{*-}$ respectively.

(4) According to the distance between positive and negative clustering centers from near to far, the combined w representative subsets are obtained, i.e., $\{V_1, \cdots, V_w\}$.

(5) Each combined representative subset V_v can be trained on SVM efficiently and independently with the following equation.

$$\min_{\boldsymbol{\alpha}_{(v)}} h(\boldsymbol{\alpha}_{(v)}) = \frac{1}{2}\boldsymbol{\alpha}_{(v)}^T \kappa_{(v,v)} \boldsymbol{\alpha}_{(v)} - \boldsymbol{e}_{(v)}^T \boldsymbol{\alpha}_{(v)}, \quad s.t.\ 0 \leq \alpha_{(v)t} \leq C \tag{11}$$

where $v = \{1, \cdots, w\}$, $\boldsymbol{\alpha}_{(v)}$ denotes the sub-vector composed by V_v, $\{\alpha_{(v)t} | \alpha_{(v)t} \in V_v, t = 1, \cdots, |V_v|\}$. $\kappa_{(v,v)}$ is a $|V_v| \times |V_v|$ sub matrix. $\alpha_{(v)t}$ is the t-th dual variable of $\boldsymbol{\alpha}_{(v)}$. $\boldsymbol{e}_{(v)} \in 1^{|V_v|}$ is the vector of all ones.

(6) All subproblem solutions are integrated to initialize an approximate whole solution $\bar{\boldsymbol{\alpha}} = \left[\bar{\boldsymbol{\alpha}}_{(1)}, \cdots, \bar{\boldsymbol{\alpha}}_{(w)}\right]$. $\bar{\boldsymbol{\alpha}}_{(v)}$ is the optimal solution for the v-th subproblem. Above method

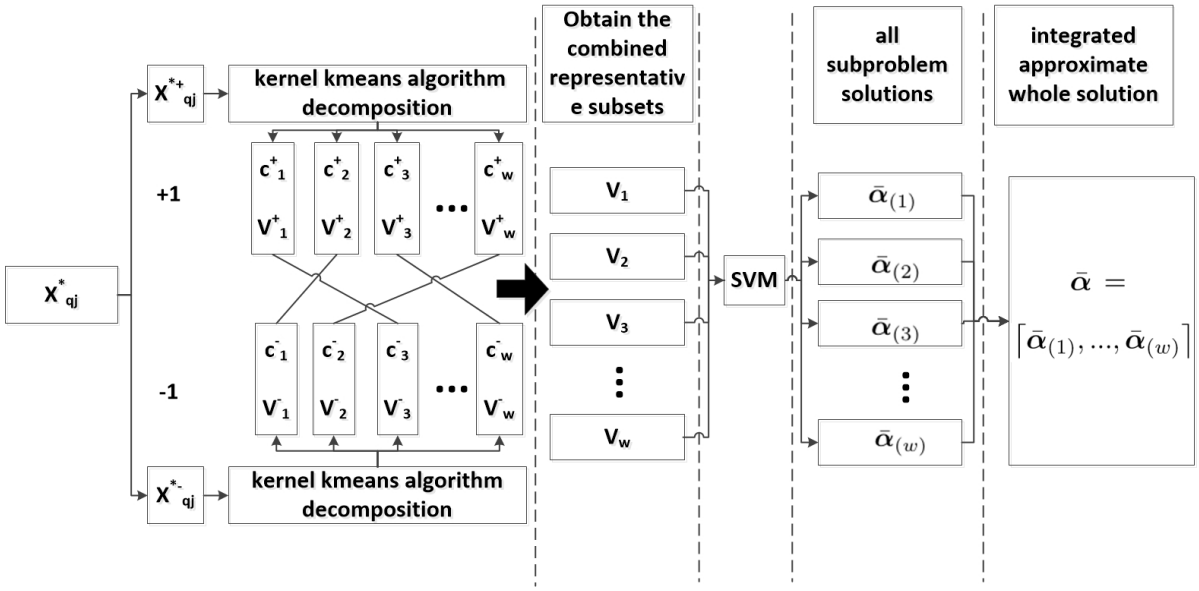

FIGURE 2. The principle of AEDC-SVM.

can overcome the computation load imbalance problem and avoid singular problems, and improve the classification performance effectively.

Although the proposed AEDC-SVM can reduce the computational complexity and has good performance when facing large-scale data sets, it cannot solve the multi-label classification problem and label imbalance issue. When facing the label imbalance issue, AEDC-SVM tends to treat each instance as negative instance. Therefore, the result has skewness. Setting different punishment parameters to two kinds of instances can solve the label imbalance issue. A larger value is set to C when facing few positive instances, which means more attention is paid to the positive instance and the misclassification of positive instance will be punished rigorously. This is the idea of DEC method. Based on DEC method, we improve the AEDC-SVM to solve the label imbalance issue. We improve the original AEDC-SVM optimization problem according to the following equation.

$$\begin{aligned} \min_{\boldsymbol{\alpha}_{(v)}} h(\boldsymbol{\alpha}_{(v)}) &= \frac{1}{2}\boldsymbol{\alpha}_{(v)}^T K_{(v,v)}\boldsymbol{\alpha}_{(v)} - \boldsymbol{e}_{(v)}^T\boldsymbol{\alpha}_{(v)}, \\ s.t.\ 0 &\le \alpha_{(v)t} \le C^+, y_t = +1, \\ 0 &\le \alpha_{(v)t} \le C^-, y_t = -1, \end{aligned} \tag{12}$$

where C^+ and C^- denote different punishment parameters. It can be seen from formula 12 that by selecting different penalty parameters C^+ and C^- for two kinds of instances, the label imbalance problem can be effectively solved.

D. DESIGN AND IMPLEMENTATION OF AEDC-MLSVM CLASSIFICATION ALGORITHM

The proposed AEDC-MLSVM classification algorithm adopts the BR problem transformation strategy to implement multi-label classification.

Firstly, it transforms the multi-label training data set into k binary training data sets based on the number of labels. Each binary training data set is composed of positive instances and negative instances, and the number of instances of each binary training data set is the same as that in the multi-label training data set.

Secondly, for each binary training data set, we adopt AEDC-SVM method to get its classifier $h_{q_j}(\boldsymbol{x})$. The main steps are as follows.

Step 1: The representative set is obtained by using the approximate extreme points method.

Step 2: The representative set is divided into positive representative set and negative representative set according to positive and negative labels.

Step 3: m training instances are chosen randomly from the positive representative set. Then the kernel kmeans algorithm is run on the m training instances and w positive cluster centers are constructed in the kernel space. After that, the w cluster centers are used to separate the positive representative set into w positive subsets.

Step 4: m training instances are chosen randomly from the negative representative set. Then the kernel kmeans algorithm is run on the m training instances and w negative cluster centers are constructed in the kernel space. After that, the w cluster centers are used to separate the negative representative set into w negative subsets.

Step 5: According to the distance between positive and negative clustering centers from near to far, w representative subsets are obtained by combining the positive and negative subsets. Each representative subset contains positive and negative instances.

Step 6: Each representative subset can be trained on the improved LIBSVM algorithm and the vector of dual variables are obtained, i.e., $\bar{\boldsymbol{\alpha}}_{(v)}$ represents the optimal solution for the v-th representative subset. The improved LIBSVM is used because it is the combination of DEC method and SMO algorithm.

Step 7: The vector of dual variables of representative set is obtained by integrating each representative subset $\bar{\boldsymbol{\alpha}}_{(v)}$, i.e., $\bar{\boldsymbol{\alpha}} = \left[\bar{\boldsymbol{\alpha}}_{(1)}, \cdots, \bar{\boldsymbol{\alpha}}_{(w)}\right]$. Then the classifier $h_{q_j}(\boldsymbol{x})$ is obtained.

Finally, we integrate the results of each classifier $h_{q_j}(\boldsymbol{x})$ by formulas 2 and 3, and efficient multi-label classification is achieved. The pseudocode of AEDC-MLSVM classification algorithm is shown in Algorithm 1.

Algorithm 1 AEDC-MLSVM Classification Algorithm

Input :
D *training data set*$\{(\boldsymbol{x}_i, Y_i)\,|i = 1, \cdots, N\}$;
Q *the set of all labels* $\{q_1, q_2, \cdots, q_k\}$;
$\boldsymbol{x}$ *testing data* $\{\boldsymbol{x} \in \mathbb{R}^d\}$; k *total number of labels*;
P *maximum size of subsets after first level partition*;
V *maximum size of subsets after second level partition*;
ε *minimal positive real constant*;
β *positive real constant*; w *number of cluster centers*;
Output:
Y the prediction label set of $\boldsymbol{x}$
begin
1) Transform multi-label training data set D into k binary training data sets with the BR problem transformation strategy, i.e., $D_{q_1}, D_{q_2}, \cdots, D_{q_k}$.
2) **for** *each binary training data set obtained* $D_{q_j}(q_j \in Q, j = 1, 2, \cdots, k)$ **do**
  (a) Get the representative set $D_{q_j}^*$ of D_{q_j}:
  $[D_{q_j}^*, \rho_{q_j}] = ImpAEPoints(D_{q_j}, P, V, \varepsilon)$, according to formulas 5, 6 and 7.
  (b) $D_{q_j}^*$ is divided into $\boldsymbol{D}_{q_j}^{*+}$ and $\boldsymbol{D}_{q_j}^{*-}$ according to the positive and negative labels.
  (c) For $\boldsymbol{D}_{q_j}^{*+}$ and $\boldsymbol{D}_{q_j}^{*-}$
    [1] Randomly select m instances from $\boldsymbol{D}_{q_j}^{*+}$ and $\boldsymbol{D}_{q_j}^{*-}$ respectively;
    [2] Run the kernel kmeans algorithm on m instances to construct w cluster centers,i.e., $\{\boldsymbol{c}_1^+, \cdots, \boldsymbol{c}_w^+\}$. Use the w cluster centers to separate $\boldsymbol{D}_{q_j}^{*+}$ into w subsets, i.e. $\{\boldsymbol{V}_1^+, \cdots, \boldsymbol{V}_w^+\}$;
    [3] Run the kernel kmeans algorithm on m instances to construct w cluster centers,i.e., $\{\boldsymbol{c}_1^-, \cdots, \boldsymbol{c}_w^-\}$. Use the w cluster centers to separate $\boldsymbol{D}_{q_j}^{*-}$ into w subsets, i.e. $\{\boldsymbol{V}_1^-, \cdots, \boldsymbol{V}_w^-\}$;
  (d) Calculate the distances of $\{(\boldsymbol{c}_1^+, \boldsymbol{c}_1^-), \cdots, (\boldsymbol{c}_1^+, \boldsymbol{c}_w^-), (\boldsymbol{c}_2^+, \boldsymbol{c}_1^-), \cdots, (\boldsymbol{c}_w^+, \boldsymbol{c}_w^-)\}$ respectively. According to the distance from near to far, obtain w representative subsets of mutual exclusion, i.e. $\{\boldsymbol{V}_1, \cdots, \boldsymbol{V}_w\}$;
  (e) **for** *each representative subset* $\boldsymbol{V}_v(v = 1, \cdots, w)$ **do**
    Use LIBSVM ($\boldsymbol{V}_v$,β) to obtain the optimal solution, i.e., $\bar{\boldsymbol{\alpha}}_{(v)}$, with the formulas of 11 and 12.
  end
  (f) Get $h_{q_j}(\boldsymbol{x})$ of $D_{q_j}^*$ according to $\bar{\boldsymbol{\alpha}} = \left[\bar{\boldsymbol{\alpha}}_{(1)}, \ldots, \bar{\boldsymbol{\alpha}}_{(w)}\right]$ and the formulas of 9 and 10;
end
3)Get the prediction label set Y of $\boldsymbol{x}$.
 if *all* $h_{q_j}(\boldsymbol{x}) < 0(j = 1, 2, \cdots, k)$ **then**
  $Y = \{j, s.t. \max h_{q_j}\}$, based on formula 3.
else
  $Y = \{j, s.t. h_{q_j} \geqslant 0\}$, based on formula 2.
end
4) **return** Y.

Through the introduction of AEDC-SVM in the previous subchapter, we can expect that the AEDC-MLSVM classification algorithm with non-linear kernel can have good performance in large-scale data sets. It will shorten the training and testing time. Meanwhile, the performance ofAEDC-MLSVM classification algorithm is similar to that of ML-LIBSVM.

E. TIME AND SPACE COMPLEXITY ANALYSIS OF AEDC-MLSVM CLASSIFICATION ALGORITHM

We know that the training time complexity of standard SVM classification algorithm is $O(N^3)$, and its space complexity is $O(N^2)$, where N represents the size of the training data set. The time complexity of obtaining representative set with approximate extreme points method is $O(kN)$. Because there are M/w dual variables in formulas 11 and 12, the time complexity of AEDC-MLSVM classification algorithm is at least $O(kM^2/w)$, and its space complexity is $O(kM^2/w^2)$, where k represents the number of labels, w represents the number of clustering centers, and M represents the size of representative set and it is far less than N. Therefore, the time-space complexity of AEDC-MLSVM classification algorithm will be greatly reduced, and it can be well applied to large-scale multi-label data sets.

IV. EXPERIMENTS

A. DESCRIPTION OF THREE PUBLIC REAL-WORLD DATA SETS

To confirm the effectiveness of the proposed AEDC-MLSVM classification algorithm, we will conduct experiments in three public real-world data sets. TMC2007-500 is a text data set, in which each instance represents an aviation safety report, and each label represents a type of safety issues described in the aviation safety report. In this data set, each aviation safety report may contain multiple types of safety issues, that is, multiple labels. Mediamill(exp1) is a video data set in which each instance represents a video and each label represents an annotation concept. In this data set, each video can contain multiple annotation concepts, that is, multiple labels. EukaryoteGO is a bioinformatics data set, in which each instance represents an protein sequence, and each label represents a type of sub-cellular location. In this data set, each protein sequence may contain multiple types of sub-cellular location, that is, multiple labels. These data sets can be obtained from public websites [28], and the detailed descriptions are shown in TABLE 1.

TABLE 1. Three public real-world data sets.

Data set	Training data set	Testing data set	Features	Labels
TMC2007-500	21519	7077	500	22
Mediamill(exp1)	30993	12914	120	101
EukaryoteGO	4658	3108	440	22

B. THREE COMPARABLE MULTI-LABEL CLASSIFICATION ALGORITHMS

To verify the advantages of AEDC-MLSVM classification algorithm, we select three comparable multi-label classification algorithms, i.e., ML-LIBSVM [12], ML-CVM [13]

and ML-BVM [14]. These algorithms are implemented by combining the BR problem transformation skills and existing single-label algorithms, i.e., LIBSVM, CVM and BVM. As the benchmark algorithm of experiments, ML-LIBSVM algorithm can achieve good classification performance, but its training and testing time complexity is too high. ML-CVM and ML-BVM algorithms are commonly used to improve the training efficiency of multi-label classification. These algorithms have been applied to many practical problems and achieved good results.

C. FIVE COMMON EVALUATION INDEXES

Because of the characteristics of multi-label classification, its evaluation index is more complex than that of single-label classification. At present, many multi-label classification evaluation indexes have been used [29]–[31]. Five common evaluation indexes are chosen to evaluate the experimental results. They are coverage, ranking loss, hamming loss, one-error and average-precision.

(1) Coverage: it is applied to evaluate how many steps are needed, on average, to move along the ranked label list in order to get all the relevant labels of an instance. This evaluation index is computed as follows.

$$Coverage = \frac{1}{N}\sum_{i=1}^{N} max_{l_j \in L}\ r_i\left(l_j\right) - 1 \tag{13}$$

here, $r_i\left(l_j\right)$ represents the rank position of label l_j in the label set L.

(2) Ranking loss: it is applied to evaluate the average of pairs of labels that are misordered for the instance. This evaluation index is computed as follows.

$$Ranking\ loss = \frac{1}{N}\sum_{i=1}^{N} \frac{1}{|Y_i||\bar{Y}_i|}\left|\left\{\left(l_j, l_a\right):\ r_i\left(l_j\right) > r_i\left(l_a\right), \left(l_j, l_a\right) \in Y_i \times \bar{Y}_i\right\}\right| \tag{14}$$

here, $\bar{Y}_i$ represents the irrelevant label set of $\boldsymbol{x}_i$, $|\bar{Y}_i|$ represents the number of the irrelevant labels for $\boldsymbol{x}_i$.

(3) Hamming loss: it is applied to evaluate how many times, on average, an instance-label pair is misclassified, i.e., an irrelevant is predicted or a relevant label is not in the prediction result. This evaluation index is computed as follows.

$$Hamming\ loss = \frac{1}{N}\sum_{i=1}^{N} \frac{Dif\ (Y_i, Z_i)}{k} \tag{15}$$

here, $Dif\ (Y_i, Z_i)$ represents the symmetric difference for Y_i and Z_i.

(4) One-error: it is applied to evaluate how many times the top-ranked label is not in the possible label set. This evaluation index is computed as follows.

$$One - error = \frac{1}{N}\sum_{i=1}^{N} \delta\left(arg\ min_{l_j \in L}\ r_i\left(l_j\right)\right) \tag{16}$$

here, $arg\ min_{l_j \in L}\ r_i\left(l_j\right)$ represents the top-ranked label of $\boldsymbol{x}_i$. If $arg\ min_{l_j \in L}\ r_i\left(l_j\right) \notin Y_i$, then $\delta\left(arg\ min_{l_j \in L}\ r_i\left(l_j\right)\right) = 1$, otherwise 0.

(5) Average-precision: it is applied to evaluate the average fraction of relevant labels ranked higher than a particular label. This evaluation index is computed as follows.

$$Average - precision = \frac{1}{N}\sum_{i=1}^{N} \frac{1}{|Y_i|}\sum_{l_j \in Y_i} \frac{\left|l_j' \in Y_i : r_i\left(l_j'\right) \le r_i\left(l_j\right)\right|}{r_i\left(l_j\right)} \tag{17}$$

The characteristics of these evaluation indexes are shown in TABLE 2. In the value indication column, ↓ represents that the smaller the value, the better the multi-label classification performance. ↑ represents that the larger the value, the better the multi-label classification performance.

TABLE 2. The characteristics of five common evaluation indexes.

Evaluation index	Value range	Value indication
Coverage	[0,k-1]	↓
Ranking loss	[0,1]	↓
Hamming loss	[0,1]	↓
One-error	[0,1]	↓
Average-precision	[0,1]	↑

D. EXPERIMENTAL SETUP AND RESULT ANALYSIS

In this experiment, the radial basis function, i.e., $K(x, y) = exp(-\gamma\|x - y\|_2^2)$ is used in the proposed AEDC-MLSVM classification algorithm and the other three comparable multi-label classification algorithms. Symbol γ indicates the scale factor of kernel and symbol $\|\cdot\|_2$ indicates the Euclidean distance. In order to obtain the optimal representative set, parameters P, V and ε need to be set in AEDC-MLSVM classification algorithm. And to obtain the optimal solution of the divide-and-conquer strategy, parameters w and β also need to be set. The meaning of the five parameters refers to Algorithm 1. In addition, two parameters, i.e., judgment basis of allowing termination e and loss function punishment parameter C need to be set in the four multi-label classification algorithms. For different data sets, above parameters are obtained through cross validation. The experiment is run on a computer with Intel i7-8565U CPU and 8GB RAM.

Through cross validation, the parameter settings of data set TMC2007-500 are as follows, $P = 30$, $V = 15$, $\varepsilon = 1.7$, $w = 2$, $\beta = 3$, $e = 1.95e^{-5}$ and $C = 4$. TABLE 3 and 4 show the experimental results of four different multi-label classification algorithms in this data set.

TABLE 3. Experimental results of data set TMC2007-500.

Evaluation index	AEDC-MLSVM	ML-LIBSVM	ML-CVM	ML-BVM
Coverage	**1.3674**	**1.2470**	1.9237	1.9158
Ranking loss	**0.0040**	**0.0014**	0.0221	0.0239
Hamming loss	**0.0109**	**0.0051**	0.0501	0.0479
One-error	**0.0189**	**0.0106**	0.1888	0.2421
Average-precision	**0.9843**	**0.9918**	0.8341	0.8313

From TABLE 3 and 4, it can be seen that the performance of AEDC-MLSVM on five common evaluation

TABLE 4. Time cost of data set TMC2007-500.

Time(second)	AEDC-MLSVM	ML-LIBSVM	ML-CVM	ML-BVM
Training time	**896.1**	3202.4	2604.1	915.2
Testing time	**72.8**	310.3	586.1	290.3

indexes is pretty close to that of ML-LIBSVM, but its training and testing time only accounts for 27.9% and 23.4% of ML-LIBSVM's training and testing time respectively. At the same time, the performance of AEDC-MLSVM on five common evaluation indexes is much better than that of ML-CVM and ML-BVM, especially on average-precision, the value rises at least 15.3%, and its training and testing time is less than that of ML-CVM and ML-BVM.

Through cross validation, the parameter settings of data set Mediamill(exp1) are as follows, $P = 50$, $V = 30$, $\varepsilon = 0.2$, $w = 2$, $\beta = 3.3$, $e = 1.95e^{-5}$ and $C = 8$. TABLE 5 and 6 show the experimental results of four different multi-label classification algorithms in this data set.

TABLE 5. Experimental results of data set Mediamill(exp1).

Evaluation index	AEDC-MLSVM	ML-LIBSVM	ML-CVM	ML-BVM
Coverage	27.5657	36.4066	**22.7116**	**23.5098**
Ranking loss	**0.0894**	0.1220	**0.0841**	0.0909
Hamming loss	**0.0518**	**0.0315**	0.0656	0.0819
One-error	**0.2449**	**0.1442**	0.5291	0.7168
Average-precision	**0.6346**	**0.6783**	0.4229	0.3832

TABLE 6. Time cost of data set Mediamill(exp1).

Time(second)	AEDC-MLSVM	ML-LIBSVM	ML-CVM	ML-BVM
Training time	**716.7**	6931.1	1364.5	917.1
Testing time	**146.0**	1307.5	874.0	638.2

It can be seen from TABLE 5 and 6 that the performance of AEDC-MLSVM on five common evaluation indexes is similar to that of ML-LIBSVM, but its training and testing time only accounts for 10.3% and 11.2% of ML-LIBSVM's training and testing time respectively. At the same time, the performance of AEDC-MLSVM on the indexes of Hamming loss, One-error and Average-precision is much better than that of ML-CVM and ML-BVM, especially on average-precision, the value rises at least 21.1%, and its training and testing time is less than that of ML-CVM and ML-BVM.

Through cross validation, the parameter settings of data set EukaryoteGO are as follows, $P = 30$, $V = 14$, $\varepsilon = 3.34$, $w = 2$, $\beta = 12$, $e = 1e^{-5}$ and $C = 1/4$. TABLE 7 and 8 show the experimental results of four different multi-label classification algorithms in this data set.

TABLE 7. Experimental results of data set EukaryoteGO.

Evaluation index	AEDC-MLSVM	ML-LIBSVM	ML-CVM	ML-BVM
Coverage	**2.4698**	**2.5972**	4.2168	4.2297
Ranking loss	**0.1070**	**0.1096**	0.1878	0.1884
Hamming loss	**0.0904**	**0.0682**	0.0971	0.0973
One-error	**0.5766**	**0.5376**	0.9801	0.9981
Average-precision	**0.6019**	**0.6244**	0.3156	0.3068

From TABLE 7 and 8, it can be seen that the performance of AEDC-MLSVM on five common evaluation indexes is pretty close to that of ML-LIBSVM, but its training and testing time only accounts for 23.5% and 9.5% of ML-LIBSVM's training and testing time respectively. At the same time, the performance of AEDC-MLSVM on five common evaluation indexes is much better than that of ML-CVM and ML-BVM, especially on average-precision, the value rises at least 28.6%, and its training and testing time is less than that of ML-CVM and ML-BVM.

TABLE 8. Time cost of data set EukaryoteGO.

Time(second)	AEDC-MLSVM	ML-LIBSVM	ML-CVM	ML-BVM
Training time	**161.5**	686.8	532.5	197.0
Testing time	**51.4**	543.3	829.3	239.2

V. CONCLUSION

In this paper, to solve the problem that the application of multi-label SVM classification algorithm in large-scale data sets is seriously restricted by heavy time complexity, AEDC-MLSVM classification algorithm is proposed. This algorithm improves the traditional multi-label SVM classification algorithm by combining approximate extreme points method and divide-and-conquer strategy, and the DEC method is used to deal with the label imbalance problem. All of these can greatly improve the applicability of this algorithm in large-scale multi-label data sets. The experimental results in three public real-world data sets show that the performance of AEDC-MLSVM algorithm is pretty close to that of ML-LIBSVM on the five commonly-used evaluation indexes, and superior to that of ML-CVM and ML-BVM. Its training and testing time is reduced greatly. We will further improve the classification performance of this algorithm by using the correlation information among labels in the future.

REFERENCES

[1] G. Tsoumakas, I. Katakis, and I. Vlahavas, "Mining multi-label data," in *Data Mining and Knowledge Discovery Handbook*. New York, NY, USA: Springer, 2009, pp. 667–685.
[2] E. Gibaja and S. Ventura, "A tutorial on multilabel learning," *ACM Comput. Surv.*, vol. 47, no. 3, pp. 1–38, 2015.
[3] F. Markatopoulou, V. Mezaris, and I. Patras, "Implicit and explicit concept relations in deep neural networks for multi-label video/image annotation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 29, no. 6, pp. 1631–1644, Jun. 2019.
[4] C. Li, C. Liu, L. Duan, P. Gao, and K. Zheng, "Reconstruction regularized deep metric learning for multi-label image classification," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 7, pp. 2294–2303, Jul. 2020.
[5] S. Burkhardt and S. Kramer, "Online multi-label dependency topic models for text classification," *Mach. Learn.*, vol. 107, no. 5, pp. 859–886, May 2018.
[6] J. Lee, W. Seo, J.-H. Park, and D.-W. Kim, "Compact feature subset-based multi-label music categorization for mobile devices," *Multimedia Tools Appl.*, vol. 78, no. 4, pp. 4869–4883, Feb. 2019.
[7] H. Chougrad, H. Zouaki, and O. Alheyane, "Multi-label transfer learning for the early diagnosis of breast cancer," *Neurocomputing*, vol. 392, pp. 168–180, Jun. 2020.
[8] V. N. Vladimir and V. Vapnik, *The Nature of Statistical Learning Theory*. New York, NY, USA: Springer-Verlag, 1995.
[9] H. Qian, Y. Mao, W. Xiang, and Z. Wang, "Recognition of human activities using SVM multi-class classifier," *Pattern Recognit. Lett.*, vol. 31, no. 2, pp. 100–111, Jan. 2010.
[10] A. Elisseeff and J. Weston, "A kernel method for multi-labelled classification," in *Proc. Adv. Neural Inf. Process. Syst.*, 2001, pp. 681–687.
[11] M. A. Tahir, J. Kittler, and A. Bouridane, "Multilabel classification using heterogeneous ensemble of multi-label classifiers," *Pattern Recognit. Lett.*, vol. 33, no. 5, pp. 513–523, Apr. 2012.
[12] C. C. Chang and C. J. Lin, "LIBSVM: A library for support vector machines," *ACM Trans. Intell. Syst. Technol.*, vol. 2, no. 3, p. 27, 2011.
[13] I. W. Tsang, J. T. Kwok, and P. M. Cheung, "Core vector machines: Fast SVM training on very large data sets," *J. Mach. Learn. Res.*, vol. 6, pp. 363–392, Apr. 2005.

[14] I. W. Tsang, A. Kocsor, and J. T. Kwok, "Simpler core vector machines with enclosing balls," in *Proc. 24th Int. Conf. Mach. Learn.*, 2007, pp. 911–918.
[15] Z. H. Zhou, M.-L. Zhang, S.-J. Huang, and Y.-F. Li, "Multi-instance multi-label learning," *Artif. Intell.*, vol. 176, no. 1, pp. 2291–2320, 2012.
[16] O. Luaces, J. Diez, J. Barranquero, J. J. del Coz, and A. Bahamonde, "Binary relevance efficacy for multi-label classification," *Prog. Artif. Intell.*, vol. 1, no. 4, pp. 303–313, 2012.
[17] A. Clare and R. D. King, "Knowledge discovery in multi-label phenotype data," in *Principles of Data Mining and Knowledge Discovery*. Berlin, Germany: Springer, 2001, pp. 42–53.
[18] J. Xu, "Fast multi-label core vector machine," *Pattern Recognit.*, vol. 46, no. 3, pp. 885–898, Mar. 2013.
[19] J. Xu, "Multi-label core vector machine with a zero label," *Pattern Recognit.*, vol. 47, no. 7, pp. 2542–2557, Jul. 2014.
[20] 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.
[21] 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.
[22] H. He and E. A. Garcia, "Learning from imbalanced data," *IEEE Trans. Knowl. Data Eng.*, vol. 21, no. 9, pp. 1263–1284, Sep. 2009.
[23] Y. Li and X. Zhang, "Improving K nearest neighbor with exemplar generalization for imbalanced classification," in *Proc. Pacific–Asia Conf. Knowl. Discovery Data Mining*. Berlin, Germany: Springer, 2011, pp. 321–332.
[24] Y. Sun, M. S. Kamel, A. K. C. Wong, and Y. Wang, "Cost-sensitive boosting for classification of imbalanced data," *Pattern Recognit.*, vol. 40, no. 12, pp. 3358–3378, Dec. 2007.
[25] R. T. Rockafellar, *Convex Analysis*. Princeton, NJ, USA: Princeton Univ. Press, 2015.
[26] M. Nandan, P. P. Khargonekar, and S. S. Talathi, "Fast SVM training using approximate extreme points," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 59–98, 2014.
[27] C. J. Hsieh, S. Si, and I. S. Dhillon, "A divide-and-conquer solver for kernel support vector machines," in *Proc. Int. Conf. Mach. Learn.*, 2013, pp. 566–574.
[28] *Multi-Label Data Sets*. [Online]. Available: http://computer.njnu.edu.cn/Lab/LABIC/LABIC-software.html and https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/
[29] M.-L. Zhang and Z.-H. Zhou, "A review on multi-label learning algorithms," *IEEE Trans. Knowl. Data Eng.*, vol. 26, no. 8, pp. 1819–1837, Aug. 2014.
[30] R. E. Schapire and Y. Singer, "BoosTexter: A boosting-based system for text categorization," *Mach. Learn.*, vol. 39, no. 2, pp. 135–168, 2000.
[31] Z. Sun, K. Hu, T. Hu, J. Liu, and K. Zhu, "Fast multi-label low-rank linearized SVM classification algorithm based on approximate extreme points," *IEEE Access*, vol. 6, pp. 42319–42326, 2018.
[32] G. Wu, R. Zheng, Y. Tian, and D. Liu, "Joint ranking SVM and binary relevance with robust low-rank learning for multi-label classification," *Neural Netw.*, vol. 122, pp. 24–39, Feb. 2020.
[33] 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.

ZHONGWEI SUN received the Ph.D. degree in computer science and technology from the Ocean University of China, Qingdao, China, in 2017.

He is currently an Assistant Professor with the School of Information and Control Engineering, Qingdao University of Technology, Qingdao. His main research interests are machine learning, multi-label classification, sensor networks, cloud computing, and big data.

XIUYAN LIU received the Ph.D. degree in computer application technology from the Ocean University of China, Qingdao, China, in 2017.

She is currently an Assistant Professor with the School of Information and Control Engineering, Qingdao University of Technology, Qingdao. Her current research interests include deep learning, mechanical fault diagnosis, and advanced signal processing.

KEYONG HU (Member, IEEE) received the Ph.D. degree in computer science and technology from the Ocean University of China, Qingdao, China, in 2014.

He is currently an Associate Professor with the School of Information and Control Engineering, Qingdao University of Technology, Qingdao. His main research interests are sensor networks, machine learning, and big data.

ZHUANG LI received the Ph.D. degree in computer science from Shanghai University, Shanghai, China, in 2016.

He is currently an Assistant Professor with the School of Information and Control Engineering, Qingdao University of Technology, Qingdao, China. His main research interests are machine learning and software testing.

JING LIU received the Ph.D. degree in computer science and technology from the Ocean University of China, Qingdao, China, in 2018.

She is currently an Assistant Professor with the School of Science and Information, Qingdao Agriculture University, Qingdao. Her main research interests include sensor networks and machine learning.