Multi-label classification with weighted classifier selection and stacked ensemble

DOI:
10.1016/j.ins.2020.06.017

Document Version
Accepted author manuscript

Multi-Label Classification with Weighted Classifier Selection and Stacked Ensemble

Yuelong Xia[a], Ke Chen[b], Yun Yang[c,*]

a School of Information Science & Engineering, Yunnan University, Kunming, 650091, China
b School of Computer Science, The University of Manchester, Manchester M13 9PL, UK
c National Pilot School of Software, Yunnan University, Kunming, 650091, China

Abstract—Multi-label classification has attracted increasing attention for use in various application scenarios, such as medical diagnosis and semantic annotation. A large number of algorithms have been proposed for multi-label classification where many are ensemble-based. However, these ensemble-based methods usually employ bagging schemes for ensemble construction, with comparatively few stacked ensembles for multi-label classification. Existing research on stacked ensemble schemes remains active, but several issues remain such as (1) little has been done to learn the weights of classifiers for combined classifier selection; (2) pairwise label correlations is not investigated sufficiently to improve classification performance. To address these issues, we propose a novel approach that simultaneously exploits label correlations and the process of learning classifier weights to improve the existing stacked ensemble schemes. First, we introduce a weighted stacked ensemble for multi-label classification and use sparsity for regularization to facilitate classifier selection and ensemble construction. Second, we consider pairwise label correlations for assigning high similar weights to improve the classification performance. Finally, we develop an optimization algorithm based on the accelerated proximal gradient and the block coordinate descent techniques to find the optimal solution efficiently. Extensive experiments on publicly available datasets and real Cardiovascular and Cerebrovascular Disease datasets demonstrate that our proposed algorithm outperforms related state-of-the-art methods.

Keywords—multi-label classification, stacked ensemble, label correlation, base classifier selection, regularization via sparsity

1. Introduction

Multi-label learning has been widely applied in various research areas, such as text categorization [1], semantic annotation [2] and medical diagnosis [3], where each example can be associated with multiple class labels simultaneously. Different from single-label learners, multi-label methods can be affected by intrinsic latent label correlations. For example, a patient with high blood pressure is more likely to develop a heart disease than one with normal blood pressure but is less likely to develop a neuromuscular disease [4].

A significant number of algorithms have been proposed to solve multi-label classification problems [5,6]. These algorithms can be divided into two categories [7], namely problem transformation methods and algorithm adaptation methods. The first category transforms a multi-label classification problem into either several independent binary classification problems or one multi-class classification problem. Typical algorithms in this category include Binary Relevance (BR) [8], Classifier Chains [9] and Label Powerset (LP) [10,11]. The second category extends a specific learning algorithm to process multi-label data where MLKNN [12], ML-DT [13], and Rank-SVM [14] are well known algorithms in this category. Despite the success, the aforementioned algorithms are still subject to limitations such as correlations among labels, class imbalance, and high dimensionality. As a result, these algorithms may exhibit poor classification performance in real-world tasks.

Ensemble-based methods have drawn considerable attention by combining individual learners from heterogeneous or homogeneous models to obtain a joint learner that improves the performance and reduces overfitting problems [15,16]. Some of the benchmark work about ensemble multi-label classification was published in recent years [16,17,18,19,20]. In general, these methods usually employ bagging schemes for ensemble construction, where some of them utilize bagging to generate a diversity of classifiers and then combine the predictions of the base classifiers by majority voting. This ensemble construction predicts a new instance by averaging the confidence values of all the classifiers for each label, rather than capturing the optimal weights to different labels, and it neglects the effect of local pairwise label correlation.

To the best of our knowledge, few stacked ensembles for multi-label classification have been proposed to overcome the limitation in bagging-based ensembles. Multi-label Stacking (MLS) [21] can be viewed as a representative of the stacked ensemble technique, where it first trains independent binary classifiers for each label and uses these predictions as input into a meta-level learning model. Then, stacked combination schemes are generated by a function (a meta-level classifier) which outputs the final prediction of the ensemble. Although MLS considers the global correlations among labels at the meta-level, it neglects the effect of local pairwise label correlation. Moreover, existing stacked ensemble methods do not take into account classifier weights for combined classifier selection.

To alleviate the aforementioned problems, we simultaneously exploit and utilize the advantage of a weighted stacked ensemble and pairwise label correlations to overcome the limitations of existing stacked ensemble schemes, which leads to a novel algorithm named Multi-Label classification with Weighted classifier selection and Stacked Ensemble (MLWSE). In the MLWSE, different

* Corresponding author at: National Pilot School of Software, Yunnan University, Kunming, 650091, China
E-mail address: yangyun@ynu.edu.cn (Y. Yang)

weights are assigned to base classifiers for different class labels, and any two strongly correlated class labels can share high similar weights than two uncorrelated or weakly correlated ones. Unlike the existing stacked ensemble schemes, the MLWSE not only applies sparsity as regularization for classifier selection and ensemble construction, but also learns label meta-level specific features to address those well-known issues in multi-label classification.

In summary, our contributions are highlighted as follows:

1) We propose a novel weighted stacked ensemble scheme named MLWSE for multi-label classification via the sparsity regularization to facilitate classifier selection and ensemble construction, which can use any multi-label classifier as its base classifiers.

2) We simultaneously exploit the classifier weights and pairwise label correlations to select label meta-level specific features in MLWSE, which can be considered a label meta-level specific features selection method.

3) The proposed approach was applied to a real-world Cardiovascular and Cerebrovascular Disease dataset, and we demonstrated that our approach would be able to effectively assist clinicians with disease diagnosis.

The remainder of the paper is organized as follows. Section 2 reviews related work about ensembles of multi-label classification. Section 3 presents our MLWSE approach. Section 4 reports the experimental results and analysis. Section 5 discusses the issues related to our approach. Finally, the conclusions are drawn in Section 6.

2. Related Work

In this section, we review state-of-the-art ensemble methods for multi-label classification, and provide an overview of weighted ensembles of multi-label classification.

2.1 Ensemble of Multi-label Classification

Ensembles of multi-label classification are developed on top of the problem transformation or algorithm adaptation methods [5]. The method aims to overcome the drawbacks of single multi-label classifiers by constructing performance and diverse base classifiers. In this section, we mainly focus on ensemble construction in multi-label classification in conjunction with classifier combination strategies.

2.1.1 Bagging Combination Scheme

Ensemble of Binary Relevance classifiers (EBR) [22] and Ensemble of Label Powerset classifiers (ELP) [23] are two representative algorithms under the bagging framework.

In EBR, each BR classifier is obtained from a random sub-sample of the training dataset, and can effectively improve the performance of BR owing to the diversity among these classifiers. However, EBR does not consider the correlation information among the different labels, which is critical for many applications in which the semantics conveyed by different labels are correlated. For mining label correlation, Read et al. proposed in [22] the Ensemble of Classifier Chains (ECC) by using multiple CC as base classifier, where each CC is learned using a random subset of the training instances. ECC takes into account correlations among labels by augmenting the feature space of each classifier with the label predictions of previous classifiers. Although the diversity in ECC is generated by using different chains and by selecting random subsets of instances, it risks selecting sub-optimal chain ordering, which could adversely affect the prediction performance.

Similar to the EBR models, ELP is proposed by combining several LP classifiers, which uses bagging to generate diverse classifiers and then combines the predictions of the base classifiers by majority voting. Because the LP method generates a single-label dataset with a different class for each different combination of labels, this means that ELP label sets are usually associated with only a few examples, which may lead to an imbalanced dataset and complicate the learning process. Other approaches to address the class-imbalance problem involved ensemble methods based on LP; for example, the Ensemble of Pruned Sets (EPS) method [23] was proposed to overcome class-imbalance problems by pruning infrequently occurring label sets. The RAndom k-labEL (RAkEL) method [24] selects n random k-labels and learns n LP classifiers to reduce the sensitivity of the model to class-imbalance, where the base classifiers include a much more balanced distribution of classes. The Chi-Dep Ensemble (CDE) method [25] addresses the class-imbalance problem by selecting the n distinct top-scored partitions, in which each partition is computed based on the χ^2 score for all label pairs.

Another multi-label ensemble method, Random Forest of Predictive Clustering Trees (RF-PCT) [26], is also based on a bagging combination scheme. This scheme predicts a new instance by averaging the confidence values of all the classifiers for each label, rather than capturing the optimal weights to different labels. At the same time, it only partially considers global label correlations, rather than exploiting local pairwise label correlations.

2.1.2 Stacked Combination Scheme

Stacked ensemble techniques have become a well-established means for improving prediction accuracy. Model stacking [27] is an efficient ensemble method in which the predictions that are generated by using multiple base classifiers are used as inputs in a meta-level classifier. Similar to classical stacking, Multi-label stacking (MLS) [21] involves applying BR twice, and takes the predictions of several BR classifiers for each label that was trained in the first step to obtain a new meta-level BR classifier to make predictions for the corresponding label, thus considering the correlations among labels in the meta-level. Several MLS methods exist that depend on different meta-level data types, which are either discrete values (0/1) or continuous values (the confidence scores). In this study, we used the confidence scores to train the meta-level classifier.

To train the meta-level classifier, we need the confidence scores (meta-level data) of the training data; however, combining the training with the same data instances that are used to train the base classifiers would lead to overfitting. Inspired by cross validation, we partition the data into F disjoint parts, generating each base classifier F times, each using F-1 partitions for training and the remaining for gathering the predictions. In this way, the stacked ensemble is diversified by using a different feature space in each classifier.

However, MLS may introduce irrelevant information into the meta-level data. If a label is completely uncorrelated with a predicted value generated by the base classifiers, the meta-level classifier introduces uninteresting information and noise, which would cause the performance to deteriorate. In this regard, it is important to take into account the weights of the confidence scores of different base classifiers for different labels. At the same time, MLS only considers the global label correlations, rather than exploiting local pairwise label correlations.

2.2 Weighted Ensemble of Multi-label Classification

How to optimally combine the contributions of each classifier is still a question in multi-label classification. In [28], a weighted classifier ensemble is proposed, which is designed for MLKNN with a weight adjustment strategy that employs a confidence coefficient obtained by utilizing the distance in MLKNN. Improved BR (IBR) [29] employs the weighted majority voting strategy to achieve the classification of multi-label data streams. However, it is difficult to extend these methods to any other base classifier because these models involve algorithm-specific properties.

The AdaBoost.MH method [30,31] was extensively studied and used in multi-label classification, which not only maintains a set of weights over the instances, but also over the labels. Its weight adjustment strategy is the following: if training instances and the corresponding labels are difficult to predict, then incrementally increase the weights in the following classifiers, whereas if instances and labels are easy to classify then lower the weights. The AdaBoost.MH is based on BR and it is the same as applying AdaBoost to multiple binary classifiers, thus it does not consider correlations among labels.

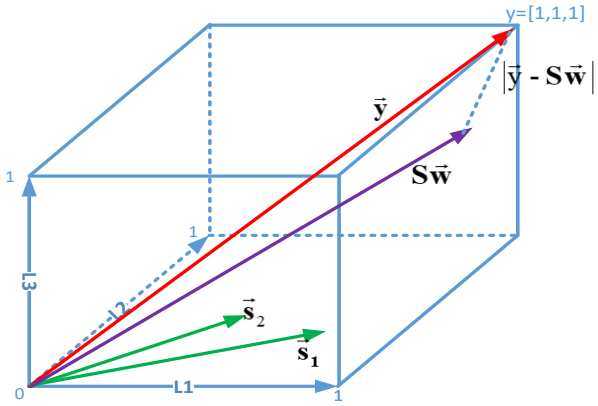


Fig. 1. Geometric explanation of a weighted stacked ensemble. Prediction scores of the combination: $\vec{\mathbf{s}}_1$ and $\vec{\mathbf{s}}_2$ (green). The optimal vector $\vec{\mathbf{y}}$ (red), generated from the ground truth. Weighted prediction of the stacked ensemble: $\mathbf{S}\vec{\mathbf{w}}$ (purple). The distance between $\vec{\mathbf{y}}$ and $\mathbf{S}\vec{\mathbf{w}}$ is minimized.

An intuitive geometric explanation is shown in Fig. 1 for a three-dimensional weighted stacked ensemble. Tai and Lin [32] utilized a similar geometrical setting, which they named Principal Label-Space Transformation, to reduce the high dimensionality of the multi-label data. Spyromitros-Xioufis et al. [33] proposed a multi-target regression method based on weighted stacking, which does not consider classifier selection. Sen and Erdogan [34] used the weighted sum rule (WS) and class-dependent weighted sum rule (CWS) for multi-class classification. Gunes et al. [35] proposed model selection methods based on the Lasso penalized for a stacking ensemble. However, these methods do not consider local pairwise label correlations.

As shown in Fig. 1, the weighted stacked ensemble minimizes the Euclidean distance between the combined prediction scores vector $\vec{\mathbf{S}}$ and the target vector that represents the ground truth $\vec{\mathbf{y}}$ in the label space, which can be seen as the following linear least-squares problem.

$$\min_{\vec{\mathbf{w}}} \left\| \vec{\mathbf{y}} - \mathbf{S}\vec{\mathbf{w}} \right\|_2^2 \tag{1}$$

where $\mathbf{S}$ is the prediction scores matrix, $\vec{\mathbf{w}}$ is the weight vector to be determined, and $\vec{\mathbf{y}}$ is the vector representing the ground truth for a given data point.

Notations For an $n \times d$ matrix $\mathbf{A}=[A_{i,j}]$, where $i \in \{1,2,...,n\}$, $j \in \{1,2,...,d\}$. $\mathbf{A}^{\mathbf{T}}$ denotes its transpose, $\mathrm{tr}(\mathbf{A}) = \sum_{i=1}^{n} A_{i,i}$ is the trace of $\mathbf{A}$, and $\|\mathbf{A}\|_F = \sqrt{\sum_{i=1}^{n}\sum_{j=1}^{d} A_{i,j}^2}$ is its Frobenius norm. For any vector $\mathbf{a}=[a_1,a_2,...a_n]^{\mathbf{T}}$, its

l_2-norm is defined as $\|\mathbf{a}\|_2 = \sqrt{\sum_{i=1}^{n} a_i^2}$ and its l_1-norm is $\|\mathbf{a}\|_1 = \sum_{i=1}^{n} |a_i|$.

3. Proposed Approach

In this section, we present our proposed approach, including the model, algorithm, and optimization method.

3.1 Preliminary

Suppose $\mathcal{X}=\mathbb{R}^d$ denotes the d-dimensional input space, and $\mathcal{Y}=\{y_1, y_2, ..., y_l\}$ denotes the label space with l possible class labels. $\mathcal{D}=\{(\mathbf{x}_i, \mathbf{y}_i) | 1 \le i \le n\}$ is the training dataset with n instances. For each multi-label example $(\mathbf{x}_i, \mathbf{y}_i)$, $\mathbf{x}_i \in \mathcal{X}$ is a d-dimensional feature vector $\mathbf{x}_i = [x_{i1}, x_{i2}, ..., x_{id}]$ and $\mathbf{y}_i = [y_{i1}, y_{i2}, ..., y_{id}]$ is the ground truth label of $\mathbf{x}_i$. Each element $y_{ij} = 1$ if the label y_j is associated with $\mathbf{x}_i$, otherwise $y_{ij} = 0$. The task of multi-label learning is to learn a function $\mathrm{h}: \mathcal{X} \to 2^{\mathcal{Y}}$ from training set $\mathcal{D}$. For any unseen example $\mathbf{x} \in \mathcal{X}$, the multi-label classifier h predicts $\mathrm{h}(\mathbf{x}) \subseteq \mathcal{Y}$ as the set of labels appropriate for $\mathbf{x}$. In this paper, we denote the input data as a matrix $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n]^{\mathbf{T}} \in \mathbb{R}^{n \times d}$, and denote the output label matrix as $\mathbf{Y} = [\mathbf{y}_1, \mathbf{y}_2, ..., \mathbf{y}_n]^{\mathbf{T}} \in \mathbb{R}^{n \times l}$.

3.2 Generating the Confidence Score Matrix

In the classifier combination problem with confidence score outputs, the combining process accepts as its input the prediction scores belonging to the different labels obtained from the base classifiers. Let s_j^k be the prediction score of label j obtained from classifier k for any data instances. Let $\mathbf{s}^k = [s_1^k, s_2^k, ..., s_l^k]^{\mathbf{T}}$ be the prediction score of all labels obtained from classifier k, then the input to the combiner is $\mathbf{s} = [\mathbf{s}^1 | \mathbf{s}^2 | .. | \mathbf{s}^m]$, where m is the number of classifiers. Let $\mathbf{s}_i$ contain the scores for training data point i obtained from the base classifiers, then the final confidence score matrix is $\mathbf{S}=[s_{ij}^k]$ indicated as

$$\mathbf{S}=\begin{bmatrix} \overbrace{s_{11}^1 \quad s_{12}^1 \quad \cdots \quad s_{1l}^1}^{\mathbf{s}^1} \cdots \overbrace{s_{11}^k \quad s_{12}^k \quad \cdots \quad s_{1l}^k}^{\mathbf{s}^k} \cdots \\ s_{21}^1 \quad s_{22}^1 \quad \cdots \quad s_{2l}^1 \cdots s_{21}^k \quad s_{22}^k \quad \cdots \quad s_{2l}^k \cdots \\ \vdots \quad \vdots \quad \cdots \quad \vdots \cdots \vdots \quad \vdots \quad \cdots \quad \vdots \cdots \\ s_{n1}^1 \quad s_{n2}^1 \quad \cdots \quad s_{nl}^1 \cdots s_{n1}^k \quad s_{n2}^k \quad \cdots \quad s_{nl}^k \cdots \end{bmatrix}$$

3.3 Weighted Classifier Selection and Stacked Ensemble

In the stacked ensemble framework, the combiner is defined as a function $\mathrm{g}: \mathbb{R}^{m \times l} \to \mathbb{R}^l$, hence our aim is to learn the g function using data $\left\{\left\{\left(\mathbf{s}_i, y_{ij}\right)\right\}_{j=1}^{l}\right\}_{i=1}^{n}$. With the loss function given in (1), our objective function to be minimized is the following:

$$\mathrm{g}(\mathbf{w}^1, \mathbf{w}^2, ..., \mathbf{w}^m) = \sum_{i=1}^{n} \sum_{j=1}^{l} \left(\sum_{k=1}^{m} (s_{ij}^k w_j^k - y_{ij}) \right)^2 \tag{2}$$

where w_j^k denotes the weight of classifier k for label j, and $\mathbf{w}^k = [w_1^k, w_2^k, ..., w_l^k]$ is the weight vector of classifier k.

Let $\mathbf{W}_j = [\mathbf{w}_j^1 | \mathbf{w}_j^2 | .. | \mathbf{w}_j^m]^{\mathbf{T}}$ represent the combined weight vector of all classifiers for the j-th label, and $\mathbf{Y}_j = [y_{1j}, y_{2j}, ..., y_{nj}]^{\mathbf{T}}$ represent the j-th column of $\mathbf{Y}$, $1 \le j \le l$. Based on generating the confidence score matrix, the objective function given in (2) can be further derived as

$$\min_{\mathbf{w}_j} \frac{1}{2} \|\mathbf{S}\mathbf{W}_j - \mathbf{Y}_j\|_2^2 \tag{3}$$

3.3.1 Sparsity Regularization for Classifier Selection

As mentioned above, generating the confidence score matrix might contain irrelevant information that is not helpful for label prediction. To combine the selection of classifiers, we add a regularization term to ensure that the weights are sparse to prevent the stacked ensemble from combining all the base classifiers. One main advantage of using sparsity regularization for classifier selection is that classifiers are selected automatically, and the number of selected classifiers is not specified beforehand. In this regard, l_1-norm regularization (Lasso) [36,37] can be considered the most successful method for inducing sparsity. We introduce the l_1-norm regularization to the model for each weight vector $\mathbf{w}_j$. Combined with the least-squares loss given in (3), the objective function for classifier selection is

$$\min_{\mathbf{w}_j} \frac{1}{2}\left\|\mathbf{S}\mathbf{W}_j - \mathbf{Y}_j\right\|_2^2 + \alpha\left\|\mathbf{W}_j\right\|_1 \tag{4}$$

where α is a regularized parameter shared by all labels for balancing the loss and regularization term, and which can be adjusted to determine the number of selected classifiers. The details are discussed in the experimental section (Section 4).

Considering all the binary classifiers simultaneously, equation (4) can be rewritten as

$$\min_{\mathbf{W}} \frac{1}{2}\left\|\mathbf{S}\mathbf{W} - \mathbf{Y}\right\|_F^2 + \alpha\left\|\mathbf{W}\right\|_1 \tag{5}$$

If $\mathbf{w}_j^k = 0$, it indicates that the k-th classifier will be eliminated and will have no effect on the prediction of the j-th label, thereby accelerating the testing instance. However, the problem with l_1-norm regularization for MLWSE is that not all $\mathbf{w}_j^k$ are zero, which means that all information from a selected classifier would not be used effectively. Motivated by group sparsity regularization [38], we propose to use Group Sparsity Lasso to solve this problem. The difference between Lasso and Group Sparsity Lasso is illustrated by the example in Fig. 2.

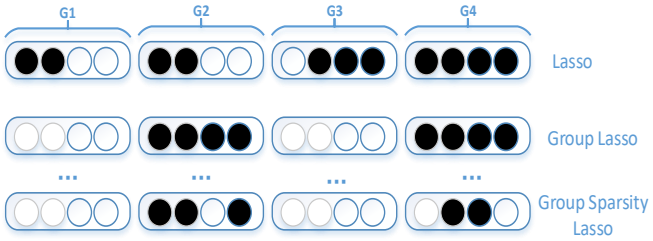


Fig. 2. Illustrative comparison of Lasso, Group Lasso, and Group Sparsity Lasso. The confidence score matrix can be divided into four groups by using the four base classifiers, G_1, G_2, G_3, G_4. The solid circles denote selected prediction scores whereas the open circles denote unselected prediction scores.

As shown in Fig. 2, Lasso forces many confidence scores to become useless, and the corresponding classifier weight is zero. Group Lasso only selects two groups G_2 and G_4, whereas the other two groups G_1 and G_3 are not selected. However, in many cases, not all information in the selected group would be useful. Group Sparsity Lasso takes advantage of both Lasso and Group Lasso in that it first selects groups, before making another selection from the selected group; i.e., it simultaneously considers intra-classifier and inter-classifier sparsity. We obtain the MLWSE of Group Sparsity Lasso in combination with the regularization term of Lasso and Group Lasso:

$$\min_{\mathbf{W}} \frac{1}{2}\left\|\mathbf{S}\mathbf{W} - \mathbf{Y}\right\|_F^2 + \alpha\lambda\left\|\mathbf{W}\right\|_1 + (1-\alpha)\lambda\sum_{k=1}^{m} c_k \left\|\mathbf{W}_{G_k}\right\|_2 \tag{6}$$

where $\alpha \in [0,1]$ is a convex combination of the Lasso and Group Lasso penalties, and the sparsity is determined by the magnitude of the tuning parameter λ. Further, c_k is a weight for the k-th group $\mathbf{W}_{G_k}$, which can be formulated as a prior to generating the contribution of the k-th group in the classifier selection process. In our experiments, we set $c_k = \sqrt{l}$.

3.3.2 Modeling Label Correlations

Exploiting label correlations generally plays a key role in multi-label classification. This is motivated by work on multi-task learning [39,40,41], which shares correlated information between tasks or modalities by considering their correlation. We assume

that if label y_j and label y_k are strong correlated the classifier discriminative to y_j may also be discriminative to y_k with a higher probability. In other words, if two labels y_j and y_k are strongly correlated, the weight vector pair $(\mathbf{W}_j, \mathbf{W}_k)$ should have a high similarity; otherwise, they would have low similarity. We construct a graph $<V, E>$ in the label space, where V denotes the vertex/label set, and E is the set of edges containing the edges between each label pair. Given the label correlation matrix $\mathbf{R}$ on E, the target can be formulated as minimizing the following equation

$$\frac{1}{2}\sum_{j=1}^{l}\sum_{k=1}^{l}\left\|\mathbf{W}_j-\mathbf{W}_k\right\|^2 R_{jk}=\mathrm{tr}(\mathbf{W}(\mathbf{D}-\mathbf{R})\mathbf{W}^{\mathbf{T}})=\mathrm{tr}\left(\mathbf{WHW}^{\mathbf{T}}\right) \tag{7}$$

where $\mathbf{H}=\mathbf{D}-\mathbf{R}$ is the graph Laplacian matrix and $\mathbf{D}$ is a diagonal matrix with $D_{ii}=\sum_{j=1}^{n}R_{ij}$. R_{jk} denotes the similarity between label y_j and label y_k. In this study, we take the cosine similarity to calculate the label correlation matrix.

Combining equations (5) and (7), we obtain the final objective function based on Lasso, MLWSE-L1:

$$\min_{\mathbf{W}}\frac{1}{2}\left\|\mathbf{SW}-\mathbf{Y}\right\|_F^2+\alpha\left\|\mathbf{W}\right\|_1+\frac{\beta}{2}\mathrm{tr}(\mathbf{WHW}^{\mathbf{T}}) \tag{8}$$

Combining equations (6) and (7), we obtain the final objective function based on Group Sparsity Lasso, MLWSE-L21:

$$\min_{\mathbf{W}}\frac{1}{2}\left\|\mathbf{SW}-\mathbf{Y}\right\|_F^2+\alpha\lambda\left\|\mathbf{W}\right\|_1+(1-\alpha)\lambda\sum_{k=1}^{m}c_k\left\|\mathbf{W}_{G_k}\right\|_2+\frac{\beta}{2}\mathrm{tr}(\mathbf{WHW}^{\mathbf{T}}) \tag{9}$$

In MLWSE-L1 and MLWSE-L21, α and β are tradeoff parameters with non-negative values. In MLWSE-L1, parameter α controls the sparsity of the model while parameter β balances the contribution of label correlations and weight learning. In MLWSE-L21, parameter α is the tradeoff parameter with the Lasso and Group Lasso, and parameter λ controls the sparsity of the model whereas parameter β balances the contribution of label correlations and weight learning.

3.3.3 Multi-label Prediction

After training MLWSE-L1 and MLWSE-L21, we obtain the classifier weight matrix $\mathbf{W}^*$. Given test data represented by matrix $\mathbf{X}^*$, we first generate the confidence score matrix $\mathbf{S}^*$ by the different base classifiers, after which we determine the predict labels by a thresholding function $\mathrm{sign}:\mathcal{Y}\to\mathbb{R}$

$$\mathrm{sign}\left(\mathbf{S}^*\mathbf{W}^*,\tau\right)=\begin{cases}1, \text{if } \mathbf{S}^*\mathbf{W}^*\geq\tau\\ 0, \text{otherwise}\end{cases} \tag{10}$$

where τ is a threshold, and in our experiments, τ is set to 0.5.

3.4 Optimization Method

Although the minimization of (8) and (9) are two convex optimization problems, the objective functions are non-smooth due to the non-smoothness of the l_1-norm regularization terms. In this section, we use the accelerated proximal gradient and block coordinate descent to optimize MLWSE-L1 and MLWSE-L21, respectively.

3.4.1 Optimization of MLWSE-L1

A general accelerated proximal gradient method can be formulated as the following convex optimization problem [42,43]:

$$\min_{\mathbf{W}\in\mathbb{H}}\left\{\mathrm{F}(\mathbf{W})=f(\mathbf{W})+g(\mathbf{W})\right\} \tag{11}$$

where $\mathbb{H}$ is a real Hilbert space, $f(\mathbf{W})$ is convex and smooth, $g(\mathbf{W})$ is convex and can be non-smooth. If $f(\mathbf{W})$ has a Lipschitz continuous gradient with Lipschitz constant L, i.e., $\left\|\nabla f(\mathbf{W_1})-\nabla f(\mathbf{W}_2)\right\|\leq L\left\|\mathbf{W_1}-\mathbf{W}_2\right\|$, instead of directly minimizing $\mathrm{F}(\mathbf{W})$, the proximal gradient algorithms can minimize its composite quadratic approximation

$$Q_L(\mathbf{W},\mathbf{W}^{(t)}) = f(\mathbf{W}^{(t)}) + \left\langle \nabla f(\mathbf{W}^{(t)}), \mathbf{W}-\mathbf{W}^{(t)} \right\rangle + \frac{L}{2}\left\|\mathbf{W}-\mathbf{W}^{(t)}\right\|_F^2 + g(\mathbf{W}) \tag{12}$$

According to (8) and (11), $f(\mathbf{W})$ and $g(\mathbf{W})$ can be defined as follows

$$f(\mathbf{W}) = \frac{1}{2}\|\mathbf{SW}-\mathbf{Y}\|_F^2 + \frac{\beta}{2}\mathrm{tr}(\mathbf{WHW^T}) \tag{13}$$

$$g(\mathbf{W}) = \alpha\|\mathbf{W}\|_1 \tag{14}$$

According to (13), we can calculate $\nabla f(\mathbf{W})$ as

$$\nabla f(\mathbf{W}) = \mathbf{S^T}(\mathbf{SW}-\mathbf{Y}) + \beta\mathbf{WH} \tag{15}$$

Inspired by the work of [44] and [45], for MLWSE-L1, given $\mathbf{W}_1$ and $\mathbf{W}_2$, we obtain the Lipschitz constant as

$$L=\sqrt{2\left\|\mathbf{S^TS}\right\|_2^2 + 2\left\|\beta\mathbf{H}\right\|_2^2} \tag{16}$$

According to (12), (14), and (16), the weight matrix $\mathbf{W}$ can be optimized by

$$\begin{aligned}\mathbf{W}^* &= \arg\min_{\mathbf{W}} Q_L(\mathbf{W},\mathbf{W}^{(t)}) = \arg\min_{\mathbf{W}} \frac{1}{2}\left\|\mathbf{W}-\mathbf{Z}^{(t)}\right\|_F^2 + g(\mathbf{W}) \\ &= \arg\min_{\mathbf{W}} \frac{1}{2}\left\|\mathbf{W}-\mathbf{Z}^{(t)}\right\|_F^2 + \frac{\alpha}{L}\|\mathbf{W}\|_1\end{aligned} \tag{17}$$

where $\mathbf{Z}^{(t)} = \mathbf{W}^{(t)} - \frac{1}{L}\nabla f(\mathbf{W}^{(t)})$.

The accelerated proximal gradient has shown that $\mathbf{W}^{(t)} = \mathbf{W}_t + \frac{b_{t-1}-1}{b_t}(\mathbf{W}_t - \mathbf{W}_{t-1})$ for a sequence b_t satisfying $b_t^2 - b_t \le b_{t-1}^2$ can improve the convergence rate to $O(1/t^2)$, where $\mathbf{W}_t$ is the result of $\mathbf{W}$ at the t-th iteration [43]. The proximal operator associated with the $g(\mathbf{W})$ of (17) is the soft-thresholding operator; then, in each iterative step, $\mathbf{W}^*$ can be obtained by the following optimization problem:

$$\mathbf{W}^{(t+1)} = prox_{\varepsilon}[\mathbf{Z}^{(t)}] = \arg\min_{\mathbf{W}} \frac{1}{2}\left\|\mathbf{W}-\mathbf{Z}^{(t)}\right\|_F^2 + \varepsilon\|\mathbf{W}\|_1 \tag{18}$$

where $prox_{\varepsilon}[\bullet]$ is a soft-thresholding operation is defined as

$$prox_{\varepsilon}[w_{ij}] = \begin{cases} w_{ij}-\varepsilon, \text{if } w_{ij} > \varepsilon \\ w_{ij}+\varepsilon, \text{if } w_{ij} < \varepsilon \\ 0, \text{otherwise} \end{cases} \tag{19}$$

According to (17) and (19), $\mathbf{W}$ can be obtained by the following soft-thresholding operation,

$$\mathbf{W}^{(t+1)}=prox_{\frac{\alpha}{L}}[\mathbf{Z}^{(t)}] \tag{20}$$

The details of the accelerate proximal gradient for MLWSE-L1 are summarized in Algorithm 1. In our experiments, we employ APG [43] to learn the weight matrix $\mathbf{W}^{*}$.

3.4.2 Optimization of MLWSE-L21

Following the spirit of the work reported by [47] and [48], we use block coordinate descent to optimize MLWSE-L21. It essentially has two components: the outer loop over the different feature groups and the inner loop that solves each of the block subproblems. The confidence score matrix $\mathbf{S}$ given in (9) can be broken down into m groups, $\mathbf{S}^1,\mathbf{S}^2,...,\mathbf{S}^m$, with each $\mathbf{S}^k \in \mathbb{R}^{n\times l}$. Let $\mathbf{S}^{-k}$ denote the remaining groups when $\mathbf{S}$ is associated to group k, and $\mathbf{W}^{-k}$ is the components of $\mathbf{W}$ over the other groups.

When a group k is selected (cyclically or otherwise), the other groups of the current $\mathbf{W}$ are fixed and the objective function is minimized only over $\mathbf{W}^k$. Then at each block we have to minimize

$$\frac{1}{2}\left\|r_{-k}-\mathbf{S}^{(k)}\mathbf{W}^{(k)}\right\|_2^2+(1-\alpha)\lambda c_k\left\|\mathbf{W}^{(k)}\right\|_2+\alpha\lambda\left\|\mathbf{W}^{(k)}\right\|_1+\frac{\beta}{2}\mathbf{tr}(\mathbf{W}^{(k)}\mathbf{H}\mathbf{W}^{(k)\mathbf{T}}) \tag{21}$$

where r_{-k} the partial residual of $\mathbf{Y}$, subtracting all group fits other than group k

$$r_{-k}=\left\|\mathbf{Y}-\sum_{j\neq k}\mathbf{S}^{(j)}\mathbf{W}^{(j)}\right\| \tag{22}$$

Let $\ell(r_{-k},\mathbf{W}^{(k)})=\frac{1}{2}\left\|r_{-k}-\mathbf{S}^{(k)}\mathbf{W}^{(k)}\right\|_2^2$ denote the least-squares loss function, and let $\nabla\ell(r_{-k},\mathbf{W}^{(k)})$ denote its gradient, then our goal is to find $\mathbf{W}_*^{(k)}$ to minimize (21). Minimizing (21) is equivalent to minimizing the following equation, centered at a point $\mathbf{W}_0^{(k)}$ by

$$\begin{aligned}\frac{1}{2t}\left\|\mathbf{W}^{(k)}-(\mathbf{W}_0^{(k)}-t\nabla\ell(r_{-k},\mathbf{W}_0^{(k)}))\right\|_2^2+(1-\alpha)\lambda c_k\left\|\mathbf{W}^{(k)}\right\|_2\\ +\alpha\lambda\left\|\mathbf{W}^{(k)}\right\|_1+\frac{\beta}{2}\mathbf{tr}(\mathbf{W}^{(k)}\mathbf{H}\mathbf{W}^{(k)\mathbf{T}})\end{aligned} \tag{23}$$

where t denotes our usual gradient step.

It was previously shown [47] that $\mathbf{W}_*^{(k)}=0$ if

$$\left\|\zeta(\mathbf{W}_0^{(k)}-t\nabla\ell(r_{-k},\mathbf{W}_0^{(k)}),t\alpha\lambda)\right\|_2\leq t(1-\alpha)\lambda c_k \tag{24}$$

and otherwise $\mathbf{W}_*^{(k)}$ satisfies

$$\left(1-\frac{t(1-\alpha)\lambda c_k}{\left\|\zeta(\mathbf{W}_0^{(k)}-t\nabla\ell(r_{-k},\mathbf{W}_0^{(k)}),t\alpha\lambda)\right\|_2}\right)_+\zeta(\mathbf{W}_0^{(k)}-t\nabla\ell(r_{-k},\mathbf{W}_0^{(k)}),t\alpha\lambda) \tag{25}$$

where $\zeta(\bullet)$ denotes the soft-thresholding operator

$$(\zeta(z,t\alpha\lambda))_i=sign(z_i)(|z_i|-t\alpha\lambda)_+ \tag{26}$$

Catalina et al. [48] showed that the inner loop can be accelerated by using proximal gradient, thus we set $t=\frac{1}{L}$ with the Lipschitz constant L given in (16). The details of the block coordinate descent for MLWSE-L21 are summarized in Algorithm 2.

Algorithm 1: Optimization of MLWSE-L1

Input: Training data matrix $\mathbf{X}\in\mathbb{R}^{n\times d}$, label matrix $\mathbf{Y}\in\mathbb{R}^{n\times l}$, base classifier $\{C_i\}_{i=1}^{m}$ and parameter α, β, η.

Output: The weight matrix $\mathbf{W}^{*}\in\mathbb{R}^{ml\times l}$.

1 Generate the confidence score matrix $\mathbf{S}\in\mathbb{R}^{n\times ml}$ by the base classifier $\{C_i\}_{i=1}^{m}$

2 Initialization:

$b_0, b_1 \leftarrow 1;\ t\leftarrow 1;\ \mathbf{W}_0, \mathbf{W}_1 \leftarrow (\mathbf{S}^{\mathbf{T}}\mathbf{S}+\eta\mathbf{I})^{-1}\mathbf{S}^{\mathbf{T}}\mathbf{Y}$

3 Compute the graph Laplacian matrix $\mathbf{H}$ on $\mathbf{Y}$

4 Compute L according to Eq. (16)

5 While not converged do

$\mathbf{W}^{(t)} \leftarrow \mathbf{W}_t + \frac{b_{t-1}-1}{b_t}(\mathbf{W}_t - \mathbf{W}_{t-1})$;

Compute $\nabla f(\mathbf{W}^{(t)})$ according to Eq. (15);

$\mathbf{Z}^{(t)} \leftarrow \mathbf{W}^{(t)} - \frac{1}{L}\nabla f(\mathbf{W}^{(t)})$;

$\mathbf{W}^{(t+1)} \leftarrow prox_{\frac{\alpha}{L}}[\mathbf{Z}^{(t)}]$;

$b_{t+1} \leftarrow \frac{1+\sqrt{4b_t^2+1}}{2}$;

$t \leftarrow t+1$;

Output: $\mathbf{W}^{*} \leftarrow \mathbf{W}^{(t+1)}$

Algorithm 2: Optimization of MLWSE-L21

Input: Training data matrix $\mathbf{X}\in\mathbb{R}^{n\times d}$, label matrix $\mathbf{Y}\in\mathbb{R}^{n\times l}$, base classifier $\{C_i\}_{i=1}^{m}$ and parameter α, β, λ, η.

Output: The weight matrix $\mathbf{W}^{*}\in\mathbb{R}^{ml\times l}$.

1 Generate the confidence score matrix $\mathbf{S}\in\mathbb{R}^{n\times ml}$ by the base classifier $\{C_i\}_{i=1}^{m}$

2 Compute the graph Laplacian matrix $\mathbf{H}$ on $\mathbf{Y}$

3 Compute L according to Eq. (16)

4 Compute r_{-k} according to Eq. (22)

5 Cyclically iterate through the groups; at each group (k) execute step 6

6 Initialization: $t \leftarrow 1/L$, $\mathbf{W}^{(k)} \leftarrow (\mathbf{S}^{\mathbf{T}}\mathbf{S}+\eta\mathbf{I})^{-1}\mathbf{S}^{\mathbf{T}}\mathbf{Y}$

7 Check if $\mathbf{W}^{(k)}=0$ according to Eq. (24), otherwise, within the group apply step 8

8 While not converged do

Update gradient $\nabla\ell(r_{-k}, \mathbf{W}^{(k)})$

Update $\mathbf{W}^{(k+1)}$ according to Eq. (25)

Output: $\mathbf{W}^{*} \leftarrow \mathbf{W}^{(k+1)}$

4. Experiments

In this section, we conducted our experiments on three types of datasets, including 2-D synthetic datasets, multi-label benchmark

datasets, and a real-world application dataset to evaluate the effectiveness of our proposed approach from different perspectives. Section 4.1 describes the experimental settings and datasets. Section 4.2 presents experimental results and analysis. Section 4.3 reports Friedman statistics analysis. Section 4.4 discusses parameter sensitively analysis, and finally, the convergence analysis is drawn in Section 4.5.

4.1 Experimental Settings and Datasets

Experiments are carried out on three types of datasets, including 2-D synthetic datasets, multi-label benchmark datasets, and the real-world dataset. To evaluate the performance of different algorithms for multi-label classification, we use six common evaluation metrics to verify the performance.

4.1.1 Datasets

The first dataset of our experiments is the 2-D synthetic datasets, which consists of four different distribution scenarios. The results of the four different simulations are presented in Fig. 3. All four simulations involve a univariate X drawn from a uniform distribution in [-4, +4]. The outcome follows the function described below:

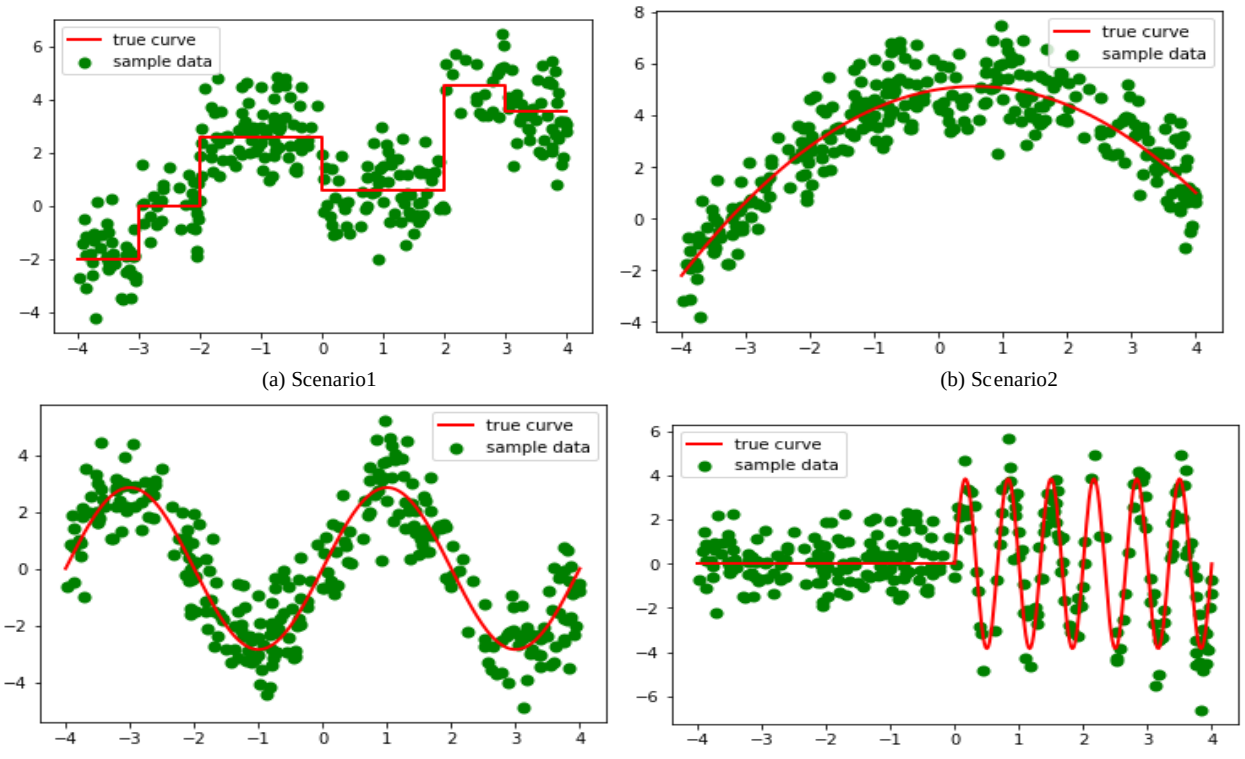
Scenario1: $Y = -2 \times \mathrm{I}(X < -3) + 2.55 \times \mathrm{I}(X > -2) - 2 \times \mathrm{I}(X > 0) + 4 \times \mathrm{I}(X > 2) - 1 \times \mathrm{I}(X > 3) + N(0,1)$

Scenario2: $Y = 5 + 0.4X - 0.36X^2 + 0.005X^3 + N(0,1)$

Scenario3: $Y = 2.85 \times \sin(\frac{\pi}{2} \times X) + N(0,1)$

Scenario4: $Y = 3.85 \times \sin(3\pi \times X) \times \mathrm{I}(X > 0) + N(0,1)$

where $\mathrm{I}(\cdot)$ is the usual indicator function and $N(0,1)$ is an independent standard normal distribution for each scenario. These scenarios were chosen because they represent a diverse set of true models. Fig. 3 contains a scatterplot of the 300 samples from each of the four simulations and the true curve for each scenario is represented by the red line.



(a) Scenario1 (b) Scenario2

(c) Scenario3 (d) Scenario4

Fig. 3. Scatterplots of the four scenarios. The red line represents the true relationship. The number of simulation samples is 300.

In particular, a multi-label classification problem can be transformed into several independent single-label classification problems [7]. For simple understanding, we consider only the case of the single-label classification. With these four synthetic datasets, we can evaluate the weighted classifier selection ability of our approach.

Further, we examine our approach for multi-label classification tasks by using a collection of 13 multi-label benchmark datasets, the details of which are summarized in Table 1. $LC = \frac{1}{N}\sum_{i=1}^{N}|Y_i|$ denotes the label cardinality, which is the average number of

labels associated with each instance. All datasets can be downloaded from the websites of Mulan[1], KDIS[2], and Meka[3]. With these benchmarking multi-label datasets, we initially compare our approach with seven state-of-the-art ensemble multi-label classification methods including EBR [22], ECC [22], EPS [23], RAkEL [24], CDE [25], AdaBoost.MH [30] and MLS [21]. All of these seven methods were implemented using the Mulan [49] and Meka [50] frameworks, which provide an API to use their functionalities in Java code[4]. These methods used $n=10$ classifiers in the ensemble, a threshold value of τ =0.5 and the C4.5 decision tree as a single-label base classifier.

Table 1. Description of benchmark datasets

Dataset	Domain	Instance	Features	Labels	LC
Emotions	Music	593	72	6	1.868
Flags	Image	194	19	7	3.392
Scene	Image	2407	294	6	1.074
Yeast	Biology	2417	103	14	4.237
Birds	Audio	645	260	19	1.014
GpositiveGO	Biology	519	912	4	1.008
CHD-49	Medicine	555	49	6	2.580
Enron	Text	1702	1001	53	3.378
Langlog	Text	1460	1004	75	1.180
Medical	Text	978	1449	45	1.245
VirusGo	Biology	207	749	6	1.217
Water-qy	Chemistry	1060	16	14	5.073
3s-bbc1000	Text	352	1000	6	1.125

To explore the potential application of our proposed method, finally, we apply our approach to a real Cardiovascular and Cerebrovascular Disease (CCD) dataset [54,55] to demonstrate its potential for practical applications in medical diagnosis, and we take CCD dataset as another benchmarking dataset to run the experiments. The dataset is collected from cardiovascular and cerebrovascular patients in a hospital in Yunnan Province, China. It contains 3,823 samples, 59 features, and 9 labels, where the nine labels are *cerebral ischemic stroke* (CIS), *cerebral hemorrhage* (CH), *subarachnoid hemorrhage* (SAH), *cerebral venous thrombosis* (CVT), *intracranial aneurysm* (IA), *cerebrovascular malformation* (CVM), *heart disease* (HD), *diabetes mellitus* (DM), and *hypertension* (HT). The number of examples corresponding to each label is listed in Table 2. In addition, we discuss the effectiveness of label correlations on our algorithms.

Table 2. Correspondence of the example size to different labels of the CCD dataset

Label	Examples	Label Frequency
CIS	3380	0.884
CH	140	0.036
SAH	134	0.035
CVT	8	0.002
IA	23	0.006
CVM	20	0.005
HD	1133	0.296
DM	920	0.240
HT	2513	0.657

In our experiments, the confidence score matrix $\mathbf{S}$ is generated by using BR, CC, and LP, where SVM is used as the single-label base classifier, and the other parameters are set as default parameters in the scikit-multilearn[5] library [51]. Hence, the number of groups m is set to 3. For MLWSE-L1, parameters α , β are searched in $\{10^{-5},10^{-4},...,10^{3},10^{4}\}$, and η is searched in {0.1, 1}. For MLWSE-L21, parameter α is searched in {0.01, 0.05, 0.1, 0.15, 0.2}, β is searched in $\{10^{-4},10^{-3},\ldots,10^{1},10^{2}\}$, λ is searched in $\{10^{-5},10^{-4},\ldots,10^{1},10^{2}\}$, and η is searched in {0.1, 1}. The software implementation of the proposed algorithms has been available at *https://github.com/AiXia520/MLWSE.*

4.1.2 Evaluation Metrics

To evaluate the performance of different algorithms for multi-label classification, we use six common evaluation metrics to

[1] http://mulan.sourceforge.net/
[2] http://www.uco.es/kdis/mllresources/
[3] http://waikato.github.io/meka/datasets/
[4] Code of Mulan is available at: https://github.com/kdis-lab/ExecuteMulan
[5] http://scikit.ml/api/skmultilearn.html

verify the performance. In general terms, they can be categorized into two groups [5], i.e., example-based metrics (Hamming loss, Accuracy, Ranking loss and F1) and label-based metrics (Macro B(h) and Micro B(h)). For each evaluation metric, the testing dataset is defined as $\mathcal{D}_t=\{(\mathbf{x}_i,\mathbf{y}_i)|1\le i\le n\}$, where $\mathbf{y}_i\in\{0,1\}^l$ is the ground truth labels of the i-th test example, and $\hat{\mathbf{y}}_\mathbf{i}=\mathrm{h}(\mathbf{x}_\mathbf{i})$ is its predicted labels.

1) Hamming loss: it evaluates the fraction of misclassified example-label pairs. The smaller the Hamming loss, the more accurate the performance of the classifier is.

$$\text{Hamming loss}=\frac{1}{n}\sum_{i=1}^{n}\frac{1}{l}\sum_{j=1}^{l}I(y_{ij}\neq\hat{y}_{ij})$$

where $I(\bullet)$ is an indication function that returns 1 if $I(\bullet)$ holds and 0, otherwise.

2) Accuracy: it evaluates the Jaccard similarity between the ground truth labels and predicted labels.

$$\text{Accuracy}=\frac{1}{n}\sum_{i=1}^{n}\frac{|\mathbf{y}_\mathbf{i}\cap\hat{\mathbf{y}}_\mathbf{i}|}{|\mathbf{y}_\mathbf{i}\cup\hat{\mathbf{y}}_\mathbf{i}|}$$

3) Ranking loss: it evaluates the fraction of reversely ordered label pairs, i.e., when an irrelevant label is ranked higher than a relevant label.

$$\text{Ranking loss}=\frac{1}{n}\sum_{i=1}^{n}\frac{1}{|\mathbf{y}_\mathbf{i}||\bar{\mathbf{y}}_\mathbf{i}|}|\{(y',y'')\mid f(x_i,y')\le f(x_i,y''),(y',y'')\in\mathbf{y}_\mathbf{i}\times\bar{\mathbf{y}}_\mathbf{i}\}$$

where $f(x,y)$ can be regarded as the confidence score of $y\in Y$ being the proper label of x.

4) F1: it is the harmonic mean of recall and precision, where p_i and q_i are the recall and precision for the i-th example.

$$\text{F1}=\frac{1}{n}\sum_{i=1}^{n}\frac{2p_iq_i}{p_i+q_i}$$

5) Label-based classification metrics can be obtained in either of the following modes [49]:

$$\text{Macro B(h)}=\frac{1}{l}\sum_{j=1}^{l}B(TP_j,FP_j,TN_j,FN_j)$$

$$\text{Micro B(h)}=B(\sum_{j=1}^{l}TP_j,\sum_{j=1}^{l}FP_j,\sum_{j=1}^{l}TN_j,\sum_{j=1}^{l}FN_j)$$

where TP_j, FP_j, TN_j, and FN_j represent the number of true positive, false positive, true negative, and false negative test examples with respect to label y_j. $B(TP_j,FP_j,TN_j,FN_j)$ indicate some specific binary classification metric (e.g., F1). Macro B(h) and Micro B(h) assume "equal weights" for labels and examples, respectively.

4.2 Experimental Results and Analysis

In this section, we present experimental results on three type of datasets and give a detailed experimental analysis for each dataset.

4.2.1 2-D Synthetic Datasets Analysis

With 2-D synthetic datasets, we evaluate the weighted classifier selection ability of our approach by gradually adding different technical components, including the weighted setting given in equation (3) as baseline, the Lasso selection given in equation (4), and the Group Sparsity Lasso selection given in equation (6). We randomly divided each dataset into a training set (35%), a validation set (35%), and a testing set (30%). The experimental results for all four scenarios are presented in Table 3.

Table 3. Experimental results on the synthetic datasets

Algorithm		Scenario1		Scenario2		Scenario3		Scenario4	
		Accuracy	Weight Vector	Accuracy	Weight Vector	Accuracy	Weight Vector	Accuracy	Weight Vector
Base classifier	SVM (linear kernel)	0.5222	-	0.4444	-	0.5333	-	0.7667	-
	SVM (poly kernel)	0.5333	-	0.4667	-	0.5333	-	0.7667	-
	Random Forest	0.5222	-	0.4667	-	0.8333	-	0.7111	-
Baseline	SVM (linear kernel)	0.4889	-0.6785	0.5000	-0.9274	0.8000	-59.1183	0.7667	-30.1166
	SVM (poly kernel)		1.6200		2.1858		56.9153		30.0606
	Random Forest		0.0364		-0.1974		0.9372		0.1643
Lasso selection	SVM (linear kernel)	0.5222	0.3391	0.5000	-0.8521	0.8333	-0.0184	0.7667	0.0715
	SVM (poly kernel)		0.5894		2.1046		0.1053		0.6603
	Random Forest		0.0521		-0.1970		0.9197		0.2771
Group sparsity lasso selection	SVM (linear kernel)	0.5333	0	0.51111	0	0.8333	0	0.7667	0
	SVM (poly kernel)		0.9326		1.1716		0.0797		0.7317
	Random Forest		0.0445		-0.1811		0.9226		0.2737

In the first scenario, all three base classifiers presented in our experiment perform well, but the SVM-based poly kernel performs

the best. The accuracy of the Lasso selection and Group Sparsity Lasso selection methods is 0.5222 and 0.5333, respectively, and the corresponding weight vectors are $\langle 0.3391, 0.5894, 0.0521 \rangle$ and $\langle 0, 0.9326, 0.0445 \rangle$, which are higher than the baseline method. The results show that Lasso selection and Group Sparsity Lasso selection learn optimal weights compared to the baseline method, which assigns a higher weight to a more effective base classifier. In the second scenario, the SVM-based poly kernel is a more optimal base classifier. Similarly, Lasso selection and Group Sparsity Lasso selection are able to adapt to the underlying structure to achieve superior accuracy. The same trend is exhibited in scenarios 3 and 4, in which the weighted classifier selection methods do nearly as well as the individual best algorithm, and even outperform the individual best algorithm. The individual best algorithm is not known, especially in a multi-label setting in which the performance of the base classifier might improve on some labels and decline on others. However, our algorithms can adaptively learn the optimal weights to select the base classifier and adapt to real changeable scenarios. Here, we verify the ability of our approach to select a weighted classifier, and the label correlations are discussed in Section 4.2.3.

4.2.2 Benchmark Datasets Analysis

Table 4 and Table 5 compare the analysis of the results of the proposed method MLWSE-L1 and MLWSE-L21 against state-of-the-art algorithms on 13 datasets. For each dataset, the parameters α, β, and η of MLWSE-L1 are set to 10^{-4}, 10^{-3}, and 0.1, respectively, and the parameters α, λ, β, and η of MLWSE-L21 are set to 0.05, 10^{-3}, 10^{-2}, and 0.1, respectively. We conducted a five-fold validation and recorded the mean and standard deviation for each evaluation metric. In the largest datasets, some algorithms cannot complete corresponding to the available resources, and these cases are marked as "DNF" in the result tables. The best results among all the algorithms being compared are highlighted in boldface. According to these results, we have the following points.

1) Comparing with bagging combination scheme (i.e., EBR, ECC, EPS, RAkEL and CDE), in more cases, MLWSE outperforms bagging combination methods. The reason is that MLWSE can comprehensively capture the optimal weights to different labels and take local pairwise label correlations into account.

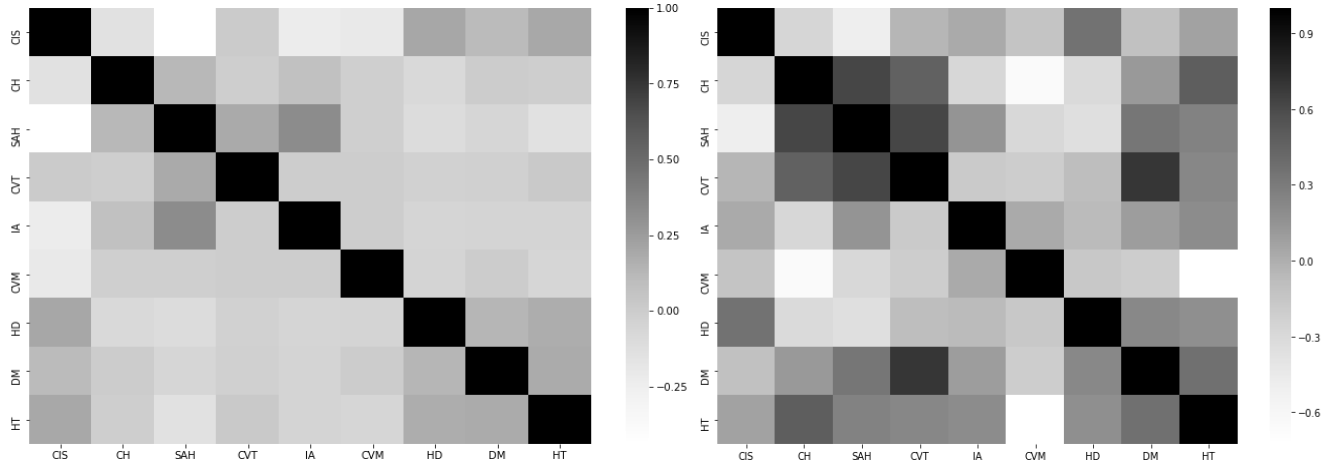
2) Comparing with stacked combination scheme (i.e., MLS), in most cases, MLWSE significantly improves the performance (i.e., Accuracy and F1). This is because our approaches can exploit the weights of the confidence scores of different base classifiers for different labels and consider local pairwise label correlations.

3) Comparing with weighted ensemble method (i.e., AdaBoost.MH), in most cases, MLWSE outperforms AdaBoost.MH. The reason is that we simultaneously exploit the classifier weights and pairwise label to address those well-known issues in multi-label classification.

4.2.3 Real-world Application Analysis

Table 6 reports the experimental results that obtained from the different multi-label ensemble algorithms. The results demonstrate that our approach achieves statistically superior performance compared to the other approaches.

To verify that if any two labels are strongly correlated, the pairs of classifier weight vectors should have high similarity. Fig. 4 shows the affinity matrices of the label matrix and the learned weight matrix for CCD, where a stronger greyscale represents stronger label correlations. The results in Fig. 8 indicate that the affinity matrix of the label matrix and the affinity of the learned weight matrix are surprisingly consistent, which means that, if two labels y_j and y_k are strongly correlated, the weight vector pair $(\mathbf{W}_j, \mathbf{W}_k)$ should be highly similar. This experimental result verifies the correctness of our proposed hypothesis.



(a) Affinity matrix of the label matrix (b) Affinity matrix of the learned weight matrix

Fig. 4. Label and weight affinity matrices of the CCD dataset

Table 4. Comparison of the experimental results obtained with each algorithm (mean ± std) in terms of Accuracy, Hamming loss, and Ranking loss

Dataset	Accuracy↑								
	EBR	ECC	EPS	RAkEL	CDE	AdaBoost.MH	MLS	MLWSE-L1	MLWSE-L21
Emotions	0.517±0.034	0.532±0.039	0.533±0.021	0.422±0.028	0.524±0.035	0.028±0.016	0.422±0.028	0.806±0.007	**0.807±0.007**
Flags	0.598±0.067	0.630±0.067	0.590±0.063	0.607±0.051	0.609±0.077	0.514±0.064	0.607±0.051	0.727±0.014	**0.743±0.014**
Scene	0.605±0.008	0.659±0.013	0.642±0.007	0.534±0.017	0.538±0.004	0.000±0.000	0.534±0.017	**0.917±0.001**	0.915±0.003
Yeast	0.489±0.014	0.505±0.008	0.491±0.015	0.434±0.012	0.478±0.008	0.335±0.015	0.434±0.012	**0.804±0.002**	0.801±0.002
Birds	0.593±0.021	0.602±0.018	0.589±0.015	0.568±0.036	0.588±0.039	0.456±0.015	0.568±0.036	0.949±0.003	**0.955±0.002**
GpositiveGO	0.933±0.011	0.929±0.016	0.937±0.008	0.930±0.017	0.928±0.018	0.000±0.000	0.930±0.017	**0.971±0.003**	**0.971±0.005**
CHD-49	0.515±0.02	0.533±0.025	0.531±0.022	0.470±0.018	0.490±0.031	0.464±0.008	0.470±0.018	**0.706±0.011**	0.703±0.013
Enron	0.425±0.015	0.467±0.019	0.376±0.020	0.414±0.012	0.411±0.013	0.151±0.009	0.414±0.012	0.953±0.001	**0.954±0.000**
Langlog	0.232±0.027	0.237±0.023	0.231±0.024	0.250±0.026	DNF	0.142±0.022	0.084±0.019	0.820±0.003	**0.830±0.001**
Medical	0.755±0.024	0.767±0.025	0.754±0.024	0.752±0.033	0.718±0.040	0.000±0.000	0.752±0.033	0.986±0.001	**0.987±0.000**
VirusGo	0.861±0.058	0.859±0.056	0.872±0.043	0.861±0.058	0.872±0.058	0.000±0.000	0.861±0.058	**0.956±0.003**	**0.956±0.005**
Water-qy	0.393±0.007	0.414±0.010	0.204±0.019	0.318±0.010	0.402±0.006	0.157±0.03	0.374±0.007	**0.715±0.004**	0.707±0.007
3s-bbc1000	0.044±0.01	0.123±0.027	0.195±0.027	0.144±0.027	0.144±0.019	0.000±0.000	0.144±0.027	0.805±0.006	**0.810±0.005**
Dataset	Hamming loss↓								
	EBR	ECC	EPS	RAkEL	CDE	AdaBoost.MH	MLS	MLWSE-L1	MLWSE-L21
Emotions	0.197±0.015	0.205±0.016	0.211±0.015	0.264±0.018	0.212±0.019	0.306±0.010	0.264±0.018	0.194±0.007	**0.193±0.007**
Flags	0.249±0.044	**0.243±0.045**	0.258±0.041	0.253±0.036	0.258±0.052	0.278±0.026	0.253±0.036	0.273±0.014	0.257±0.014
Scene	0.093±0.003	0.094±0.004	0.099±0.003	0.135±0.007	0.136±0.003	0.179±0.002	0.135±0.007	**0.083±0.001**	0.085±0.003
Yeast	0.205±0.006	0.210±0.004	0.212±0.007	0.248±0.008	0.228±0.006	0.232±0.007	0.249±0.008	**0.197±0.002**	0.199±0.002
Birds	**0.042±0.003**	0.043±0.004	0.046±0.002	0.051±0.006	0.047±0.006	0.053±0.002	0.051±0.006	0.051±0.003	0.045±0.001
GpositiveGO	**0.027±0.004**	0.030±0.009	0.031±0.005	**0.027±0.006**	0.031±0.009	0.255±0.007	**0.027±0.006**	0.029±0.003	0.029±0.005
CHD-49	0.299±0.013	0.304±0.020	0.307±0.016	0.325±0.013	0.323±0.022	0.307±0.004	0.325±0.013	**0.294±0.011**	0.297±0.013
Enron	0.048±0.001	0.048±0.002	0.052±0.002	0.051±0.001	0.051±0.001	0.062±0.001	0.051±0.001	0.047±0.001	**0.046±0.000**
Langlog	**0.016±0.001**	**0.016±0.001**	**0.016±0.001**	0.020±0.002	DNF	**0.016±0.001**	0.037±0.002	0.180±0.003	0.170±0.001
Medical	**0.010±0.001**	**0.010±0.001**	0.012±0.001	**0.010±0.001**	0.012±0.001	0.028±0.001	**0.010±0.001**	0.014±0.001	0.013±0.000
VirusGo	0.045±0.012	0.045±0.014	0.047±0.019	**0.042±0.017**	**0.042±0.019**	0.203±0.013	**0.042±0.017**	0.044±0.003	0.044±0.005
Water-qy	0.293±0.009	0.295±0.009	0.323±0.002	0.329±0.004	0.303±0.010	0.338±0.008	0.311±0.005	**0.286±0.004**	0.293±0.007
3s-bbc1000	0.209±0.011	0.223±0.012	0.206±0.010	0.251±0.029	0.250±0.013	**0.188±0.008**	0.251±0.029	0.195±0.006	0.190±0.005
Dataset	Ranking loss↓								
	EBR	ECC	EPS	RAkEL	CDE	AdaBoost.MH	MLS	MLWSE-L1	MLWSE-L21
Emotions	0.171±0.019	0.171±0.013	0.196±0.015	0.316±0.031	0.176±0.019	0.427±0.029	0.326±0.036	0.159±0.013	**0.149±0.011**
Flags	0.201±0.032	0.217±0.041	0.220±0.051	0.318±0.042	0.256±0.060	0.238±0.034	0.272±0.035	0.233±0.021	**0.200±0.011**
Scene	0.079±0.009	0.092±0.009	0.101±0.008	0.195±0.015	0.138±0.010	0.472±0.013	0.227±0.021	**0.068±0.003**	0.069±0.003
Yeast	0.185±0.010	0.191±0.010	0.202±0.008	0.336±0.015	0.219±0.009	0.363±0.029	0.316±0.012	0.171±0.001	**0.168±0.001**
Birds	**0.098±0.012**	0.111±0.013	0.140±0.014	0.199±0.026	0.134±0.015	0.229±0.037	0.168±0.012	0.120±0.008	0.110±0.003
GpositiveGO	0.025±0.005	0.027±0.008	0.031±0.011	0.034±0.012	0.029±0.012	0.301±0.019	0.025±0.006	0.026±0.005	**0.024±0.004**
CHD-49	0.222±0.015	0.230±0.020	0.226±0.021	0.313±0.014	0.255±0.027	0.222±0.011	0.313±0.020	0.215±0.006	**0.210±0.007**
Enron	**0.085±0.008**	0.150±0.014	0.161±0.011	0.302±0.011	0.198±0.001	0.240±0.011	0.175±0.005	0.105±0.003	0.092±0.007
Langlog	**0.121±0.005**	0.273±0.017	0.291±0.013	0.413±0.011	DNF	0.470±0.015	0.166±0.039	0.248±0.005	0.230±0.004
Medical	0.031±0.003	0.042±0.011	0.057±0.011	0.097±0.016	0.074±0.005	0.285±0.010	0.070±0.016	0.033±0.009	**0.025±0.004**
VirusGo	**0.030±0.015**	0.033±0.015	**0.030±0.017**	0.067±0.055	0.043±0.018	0.264±0.045	0.042±0.025	0.031±0.004	0.032±0.005
Water-qy	0.253±0.006	0.256±0.006	0.347±0.007	0.368±0.007	0.275±0.005	0.374±0.011	0.325±0.006	**0.247±0.008**	0.262±0.006
3s-bbc1000	0.404±0.034	0.417±0.031	0.383±0.037	0.497±0.035	0.434±0.003	0.422±0.027	0.497±0.058	**0.381±0.020**	0.389±0.025

Table 5. Comparison of the experimental results obtained for each algorithm (mean ± std) in terms of F1, Macro-F1, and Micro-F1

Dataset	F1↑								
	EBR	ECC	EPS	RAkEL	CDE	AdaBoost.MH	MLS	MLWSE-L1	MLWSE-L21
Emotions	0.597±0.037	0.612±0.037	0.615±0.018	0.509±0.036	0.608±0.031	0.037±0.02	0.509±0.036	**0.639±0.024**	0.614±0.014
Flags	0.711±0.057	**0.735±0.050**	0.699±0.049	0.721±0.043	0.721±0.065	0.631±0.063	0.721±0.043	0.700±0.020	0.721±0.025
Scene	0.620±0.007	0.675±0.014	0.655±0.006	0.573±0.016	0.573±0.009	0.000±0.000	0.573±0.016	**0.708±0.005**	0.672±0.010
Yeast	0.599±0.014	0.611±0.007	0.599±0.013	0.556±0.012	0.595±0.007	0.456±0.019	0.556±0.012	**0.647±0.006**	0.625±0.004
Birds	0.618±0.022	**0.631±0.016**	0.616±0.019	0.603±0.037	0.621±0.04	0.456±0.015	0.603±0.037	0.152±0.024	0.140±0.009
GpositiveGO	0.938±0.012	0.931±0.017	0.940±0.008	0.934±0.018	0.933±0.018	0.000±0.000	0.934±0.018	**0.945±0.009**	0.941±0.008
CHD-49	0.628±0.022	0.643±0.024	0.643±0.016	0.587±0.016	0.610±0.032	0.580±0.007	0.587±0.016	**0.659±0.008**	0.654±0.016
Enron	0.537±0.015	**0.579±0.017**	0.472±0.020	0.525±0.012	0.523±0.012	0.231±0.013	0.525±0.012	0.578±0.011	0.576±0.006
Langlog	0.239±0.026	0.246±0.020	0.236±0.024	0.267±0.025	DNF	0.142±0.022	0.115±0.026	0.487±0.004	**0.496±0.002**
Medical	0.785±0.025	**0.795±0.026**	0.779±0.024	0.783±0.031	0.751±0.043	0.000±0.000	0.783±0.031	0.773±0.015	0.770±0.011
VirusGo	0.883±0.057	0.879±0.055	0.893±0.037	0.880±0.056	0.893±0.047	0.000±0.000	0.880±0.056	**0.913±0.008**	0.905±0.013
Water-qy	0.532±0.007	0.556±0.011	0.299±0.022	0.452±0.011	0.543±0.006	0.244±0.043	0.513±0.006	0.550±0.009	**0.557±0.011**
3s-bbc1000	0.047±0.012	0.128±0.027	**0.207±0.028**	0.162±0.029	0.159±0.019	0.000±0.000	0.162±0.029	0.051±0.022	0.043±0.021
Dataset	Macro-F1 ↑								
	EBR	ECC	EPS	RAkEL	CDE	AdaBoost.MH	MLS	MLWSE-L1	MLWSE-L21
Emotions	0.639±0.029	**0.641±0.027**	0.631±0.022	0.551±0.039	0.635±0.037	0.038±0.018	0.551±0.039	0.608±0.023	0.584±0.013
Flags	0.657±0.063	0.671±0.086	0.587±0.065	0.658±0.077	0.668±0.077	0.560±0.129	0.658±0.077	0.687±0.024	**0.711±0.025**
Scene	0.709±0.009	**0.728±0.013**	0.707±0.003	0.634±0.015	0.629±0.002	0.000±0.000	0.634±0.015	0.700±0.005	0.665±0.010
Yeast	0.385±0.009	0.398±0.006	0.374±0.005	0.383±0.010	0.405±0.011	0.122±0.003	0.384±0.009	**0.619±0.006**	0.593±0.004
Birds	0.321±0.055	0.291±0.012	0.265±0.052	**0.349±0.048**	0.336±0.057	0.053±0.033	**0.349±0.048**	0.141±0.022	0.133±0.010
GpositiveGO	0.871±0.045	0.854±0.062	0.901±0.047	0.859±0.054	0.845±0.056	0.000±0.000	0.859±0.054	**0.943±0.008**	0.940±0.007
CHD-49	0.498±0.015	0.512±0.026	0.510±0.017	0.470±0.022	0.490±0.030	0.270±0.002	0.470±0.022	**0.629±0.007**	0.624±0.017
Enron	0.219±0.015	0.225±0.016	0.182±0.010	0.214±0.021	0.157±0.000	0.085±0.014	0.214±0.021	**0.548±0.009**	0.547±0.005
Langlog	0.270±0.047	0.273±0.048	0.264±0.043	0.284±0.048	DNF	0.237±0.047	0.051±0.001	0.474±0.006	**0.478±0.003**
Medical	0.653±0.029	0.630±0.031	0.616±0.058	0.669±0.037	0.468±0.002	0.324±0.036	0.669±0.037	**0.758±0.015**	0.755±0.011
VirusGo	0.796±0.078	0.833±0.072	0.844±0.090	0.803±0.069	0.858±0.089	0.067±0.082	0.803±0.069	**0.902±0.009**	0.894±0.011
Water-qy	0.502±0.005	0.523±0.011	0.177±0.019	0.413±0.012	0.503±0.004	0.091±0.020	0.466±0.011	0.518±0.011	**0.526±0.010**
3s-bbc1000	0.062±0.032	0.115±0.027	**0.246±0.028**	0.189±0.051	0.180±0.002	0.000±0.000	0.189±0.051	0.049±0.021	0.036±0.023
Dataset	Micro-F1 ↑								
	EBR	ECC	EPS	RAkEL	CDE	AdaBoost.MH	MLS	MLWSE-L1	MLWSE-L21
Emotions	0.662±0.028	0.663±0.025	0.654±0.023	0.564±0.038	0.654±0.034	0.063±0.032	0.564±0.038	**0.664±0.013**	0.658±0.013
Flags	0.746±0.051	**0.760±0.051**	0.725±0.05	0.745±0.046	0.741±0.063	0.693±0.064	0.745±0.046	0.719±0.017	0.737±0.017
Scene	0.705±0.007	0.718±0.012	0.700±0.006	0.624±0.015	0.617±0.003	0.000±0.000	0.624±0.015	**0.750±0.004**	0.733±0.009
Yeast	0.628±0.011	0.636±0.006	0.625±0.012	0.581±0.012	0.617±0.006	0.480±0.016	0.581±0.011	**0.644±0.006**	0.621±0.004
Birds	0.431±0.054	0.450±0.031	0.402±0.034	0.444±0.048	**0.456±0.055**	0.000±0.000	0.444±0.048	0.365±0.031	0.359±0.027
GpositiveGO	**0.947±0.008**	0.939±0.018	0.939±0.009	0.946±0.013	0.938±0.018	0.000±0.000	0.946±0.013	0.942±0.005	0.942±0.009
CHD-49	0.655±0.017	**0.667±0.025**	0.663±0.018	0.619±0.019	0.638±0.028	0.598±0.004	0.619±0.019	0.658±0.006	0.653±0.017
Enron	0.562±0.004	**0.583±0.013**	0.481±0.016	0.550±0.009	0.544±0.002	0.245±0.014	0.550±0.009	0.565±0.007	0.566±0.004
Langlog	0.159±0.022	0.174±0.012	0.156±0.027	0.191±0.014	DNF	0.000±0.000	0.192±0.029	0.532±0.006	**0.544±0.003**
Medical	0.810±0.016	**0.815±0.024**	0.780±0.028	0.813±0.026	0.781±0.027	0.000±0.000	0.813±0.026	0.754±0.013	0.759±0.007
VirusGo	0.890±0.033	0.890±0.036	0.881±0.047	0.897±0.042	**0.898±0.046**	0.000±0.000	0.897±0.042	0.894±0.008	0.894±0.011
Water-qy	0.563±0.006	**0.585±0.011**	0.304±0.024	0.480±0.010	0.570±0.008	0.259±0.045	0.544±0.008	0.559±0.007	0.557±0.009
3s-bbc1000	0.079±0.023	0.173±0.034	**0.277±0.033**	0.215±0.036	0.208±0.030	0.000±0.000	0.215±0.036	0.086±0.033	0.084±0.042

Table 6. Experimental results for each of the algorithms (mean ± std) on the CCD dataset

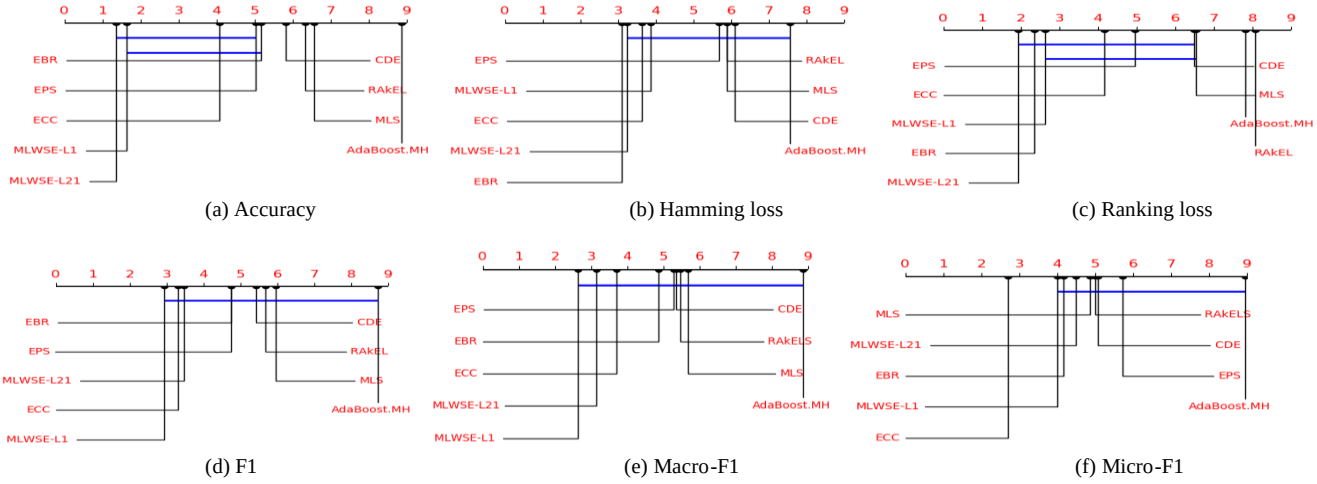
Algorithm	Accuracy ↑	Hamming loss ↓	Ranking loss ↓	F1 ↑	Macro-F1 ↑	Micro-F1 ↑
EBR	0.6923±0.0118	0.0910±0.0050	0.0395±0.0040	0.7694±0.0102	0.4038±0.0464	0.8079±0.0100
ECC	0.7041±0.0082	**0.0896±0.0041**	0.0472±0.0045	0.7800±0.0064	0.4196±0.0495	**0.8156±0.0074**
EPS	0.6904±0.0069	0.0935±0.0034	0.0492±0.0057	0.7673±0.0060	0.4045±0.0508	0.8063±0.0069
RAkEL	0.6797±0.0047	0.0957±0.0028	0.0853±0.0046	0.7597±0.0040	0.3985±0.0477	0.7982±0.0058
CDE	0.6953±0.0060	0.0913±0.0034	0.0579±0.0053	0.7718±0.0049	0.4096±0.0458	0.8094±0.0066
AdaBoost.MH	0.6178±0.0126	0.1201±0.0045	0.0536±0.0044	0.7213±0.0110	0.2146±0.0442	0.7405±0.0094
MLS	0.6797±0.0047	0.0957±0.0028	0.0814±0.0030	0.7597±0.0040	0.3985±0.0477	0.7982±0.0058
MLWSE-L1	**0.9090±0.0015**	0.0910±0.0015	**0.0388±0.0016**	**0.7979±0.0035**	**0.7686±0.0027**	0.8102±0.0038
MLWSE-L21	**0.9101±0.0009**	0.0899±0.0009	**0.0384±0.0078**	**0.7968±0.0035**	**0.7681±0.0033**	0. 8116±0.0023

4.3 Friedman Statistics Analysis

We employed the Friedman test [52,53] to statistically analyze the performance of the different algorithms systematically. Table 7 provides the Friedman statistics F_F and the corresponding critical value in terms of each evaluation metric. As shown in Table 7, at the significance level α=0.05, the null hypothesis of "equal" performance is rejected for each evaluation metric. Hence, we can use the post-hoc test [53] for relative performance comparisons. We employed the Nemenyi test [53] to analyze the performance of our proposed method MLWSE-L1 and MLWSE-L21 against that of the other algorithms. The performance of two classifiers is significantly different if the corresponding average ranks differ by at least the critical difference $CD=q_a\sqrt{\frac{k(k+1)}{6N}}$. At the significance level α=0.05, the value of q_a=3.102 [53], for the Nemenyi test with $k=9$, $N=14$ (including 13 benchmark datasets and a real-world dataset), and thus CD=3.211. Fig. 5 shows the CD diagrams for each evaluation metric, where the average rank of each algorithm is marked along the axis (lower ranks to the left). In each subfigure, any algorithm connected with MLWSE is considered to have significantly different performance between them.

Table 7. Summary of the Friedman statistics $F_F(k=9, N=14)$ and the Critical Value in term of each evaluation Metric

Metric	F_F	Critical Value (α=0.05)
Accuracy	35.075	3.211
Hamming loss	6.348	
Ranking loss	37.824	
F1	9.261	
Macro-F1	10.243	
Micro-F1	8.312	



(a) Accuracy (b) Hamming loss (c) Ranking loss

(d) F1 (e) Macro-F1 (f) Micro-F1

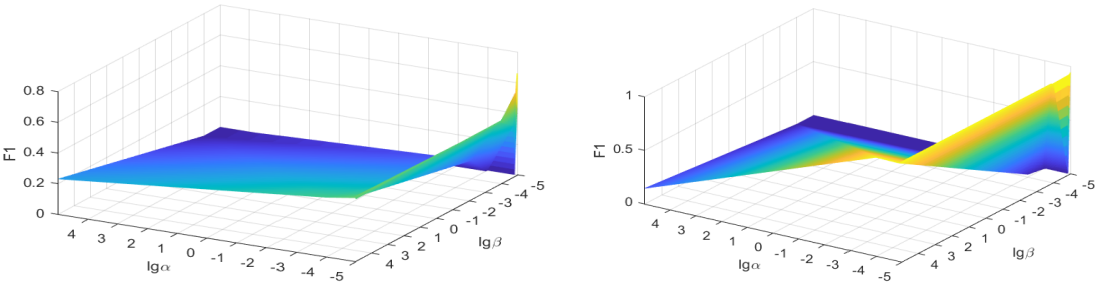
Fig. 5. CD diagrams of the algorithms that were compared under each evaluation criterion.

The Friedman statistics results indicate that: 1) EBR outperforms the other approaches on *Hamming Loss*. This is because EBR is a *first-order* approach (which does not consider label correlations) that tries to optimize the Hamming Loss. But the proposed MLWSE-L21 outperforms MLWSE-L1 and the other approaches in terms of *Hamming Loss*. 2) ECC outperforms the other approaches on *Micro-F1*, because ECC is a *high-order* approach (which considers global label correlations) that tries to model global label correlations. But the proposed MLWSE-L1 outperforms MLWSE-L21 and the other approaches in terms of *Micro-F1*. 3) MLWSE outperforms the other approaches in terms of the other four evaluation metrics. The superior performance of MLWSE against these approaches indicates the effectiveness of learning label correlations and the classifier weight.

To summarize, MLWSE achieves competitive performance against other well-established multi-label ensemble approaches.

4.4 Parameter Sensitivity Analysis

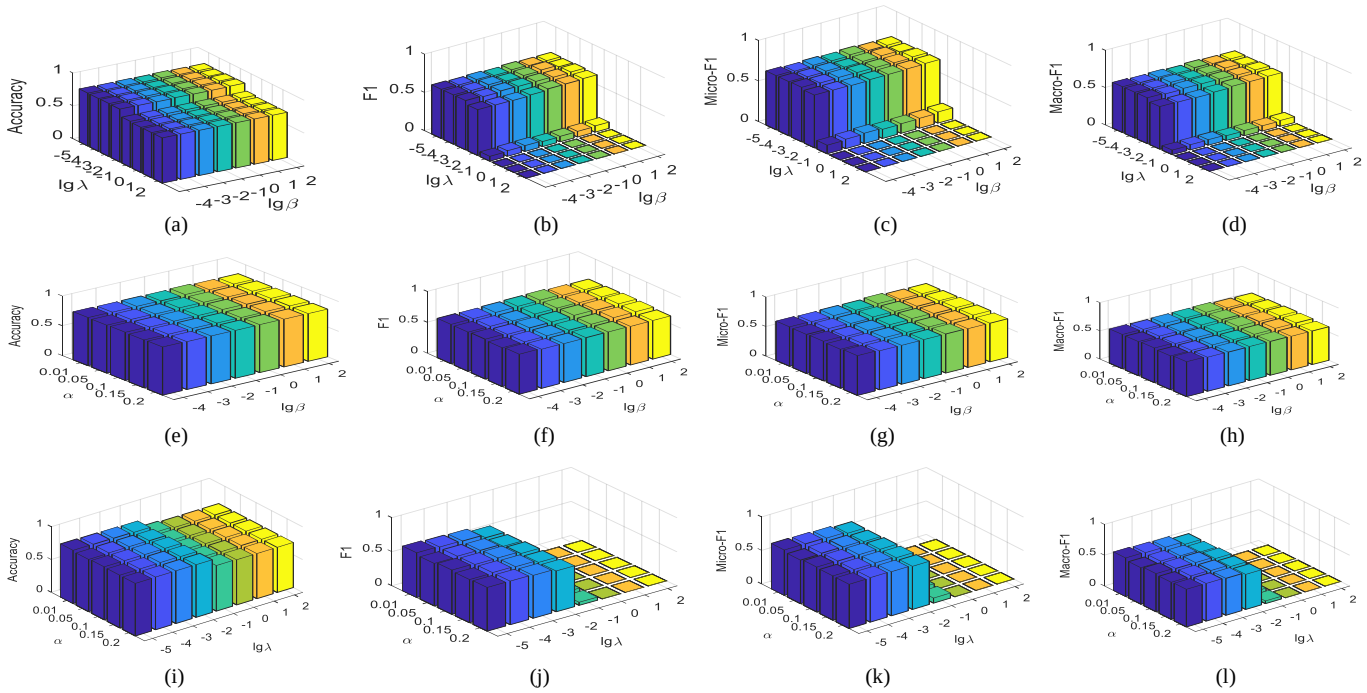
We analyzed the parameter sensitivity of MLWSE-L1 and MLWSE-L21 by conducting experiments on the *Emotions* and *GpositiveGO* datasets. We performed a five-fold cross validation on each dataset and analyzed the relative parameter.



(a) Emotions dataset (b) GpositiveGO dataset

Fig. 6. Parameter sensitivity analysis of MLWSE-L1

In MLWSE-L1, the regularization parameters α and β were searched in $\{10^{-5},10^{-4},...,10^{3},10^{4}\}$, and η was set to 0.1. For each (α,β)-pair, we recorded the mean value of F1. Fig. 6a and Fig. 6b report the influence of parameters α and β on the *Emotions* and *GpositiveGO* datasets, respectively. It can be seen from Fig. 5 that in most cases: (1) The performance of MLWSE-L1 is worse when the value of α is large, especially, $\alpha>10$ is often harmful; (2) The performance of MLWSE-L1 initially improves and then degrades with the increasing of β. Therefore, we fixed the regularization parameters α and β to 10^{-4} and 10^{-3}, respectively.



(a) (b) (c) (d)

(e) (f) (g) (h)

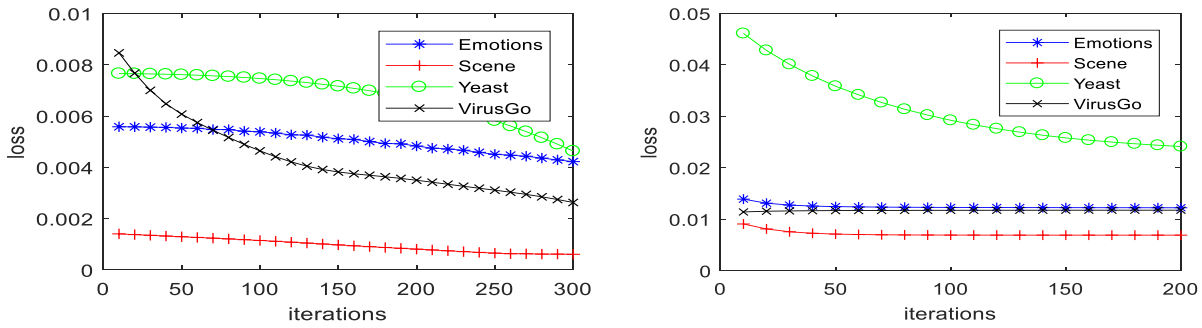
(i) (j) (k) (l)

Fig. 7. Parameter sensitivity analysis of MLWSE-L21. (a)-(d): the parameter analysis of MLWSE-L21 with fixed α; (e)-(h): the parameter analysis of MLWSE-L21 with fixed λ; (i)-(l): the parameter analysis of MLWSE-L21 with fixed β.

MLWSE-L21 searches for the regularization parameters α in {0.01, 0. 05, 0.1, 0.15, 0.2}, λ in $\{10^{-5},10^{-4},\ldots,10^{1},10^{2}\}$, β in $\{10^{-4},10^{-3},\ldots,10^{1},10^{2}\}$, and η is set to 0.1. We first find the group of best configurations for the parameters by using five-fold cross validation on the training data of the *Emotions* dataset, and then keep the value of one parameter constant and vary the values of the other two parameters. The average results of MLWSE-L21 with different values of α , β , and λ are depicted in Fig. 7(a)-(l). The figure shows that, in most cases, (1) With a fixed setting of α , the candidate sets for λ and β can be employed in $\{10^{-4},10^{-3},10^{-2}\}$ to obtain satisfactory performance. (2) With a fixed setting of λ , the performance of MLWSE-L21 is stable with different values of each (α,β) -pair. (3) With a fixed setting of β , the candidate sets for α and λ can be employed by searching λ in $\{10^{-5},10^{-4},10^{-3},10^{-2}\}$ to obtain satisfactory performance.

4.5 Convergence Analysis

We analysis convergence of MLWSE-L1 and MLWSE-L21 by conducting experiments on the *Emotions*, *Scene*, *Yeast* and *VirusGO* datasets. In this work, our approach is solved by using accelerated proximal gradient and block coordinate descent, which can be seen as iterative shrinkage-thresholding algorithms. Accelerated proximal gradient was proven to converge in function values as $O(1/t^2)$ with a backtracking step size rule [43]. Block coordinate descent is proven to converge in function values as $O((\log t/t)^2)$ [43,44,45]. Fig. 8 shows the value of the loss function of MLWSE-L1 (8) and MLWSE-L21 (9) with the number of iterative cycles, respectively. For MLWSE-L1, the loss value tends to stabilize after 300 iterative cycles. Fig. 8(a) shows that when the number of iterative cycles exceeds 200, the loss value (which is less than 0.008) has little effect on the performance; hence, the number of cycles in the experiment was specified to be 200. For MLWSE-L21, the loss value tends to stabilize after 200 cycles. Here, the number of iterative cycles refers to the outer loop, whereas the number of cycles of the inner loop in the experiment was set to 100. The experimental results show that the proposed methods are more efficient than most of the multi-label ensemble methods.



(a) MLWSE-L1 (b) MLWSE-L21

Fig. 8. Loss of MLWSE-L1 and MLWSE-L21 with the number of iterative cycles.

5. Discussion

We conducted a comprehensive investigation based on a series of simulations. As demonstrated by our experiments, our approach is able to achieve high-quality generalization performance by implementing a simple iterative shrinkage-thresholding algorithm. Thus, a promising yet easy-to-use technique for multi-label ensemble classification is introduced. We summarize the advantages of our approach as follows.

First, we proposed a novel weighted stacked ensemble approach for multi-label classification and used sparsity for regularization to facilitate classifier selection and ensemble construction with the ultimate goal of developing a simple and efficient method to select multi-label base classifiers. Our approach is geometrically explained in Section 2.2, and the results of the weighted classifier selection ability are presented in Table 3. Further, the proposed approach was tested on datasets from a variety of domains such as Text, Imaging, Biology, and Medicine. The experimental results in Table 4 and Table 5 indicate that our approach outperforms state-of-the-art multi-label ensemble algorithms. In addition, our model can be used as base classifier with any existing multi-label classification algorithm such as MLKNN [12] and ML-DT [13]. As illustrated in Fig. 8, the proposed approach only needs a small number of iterative steps to reach convergence, thereby confirming that our technique is easy to use for multi-label classification tasks and applications.

Second, our approach can be seen as label meta-specific-feature selection methods because of their exploitation of label correlations. Here, generating the confidence score matrix $\mathbf{S}$ can be seen as a meta feature. Because we simultaneously exploit classifier weights and label correlation, the weight matrix $\mathbf{W}$ can be regarded as selecting a label meta-specific-feature. We made

the assumption that the classifier weight vector pair would be highly similar if a strong correlation between any two labels existed. In other words, any two strongly correlated class labels would share more meta features with each other than two uncorrelated or weakly correlated ones. Fig. 4 shows the correctness of our proposed hypothesis. Similar approaches to learning label specific features have been proposed, such as MLSF [56] and LLSF [57]. In our methods, parameter α can be used to adjust feature sparsity, and parameter β balances the contribution of label correlations and weight learning. The results in Fig. 6 and Fig. 7 demonstrate the effect of parameter adjustments, and show that the proposed MLWSE-L21 method is more stable than the MLWSE-L1 method. Finally, the results in Fig. 5 and Table 6 indicate that our approach is much more robust to the different evaluation metrics than the other algorithms used in the comparison.

A subsequent literature survey revealed that Zhou and Tao [58] proposed a multi-label subspace ensemble method based on the group sparsity Lasso, which does not consider the stacked ensemble. Büyükçakır et al. [59] also proposed a weighted stacked ensemble method for multi-label classification, named GOOWE-ML. In their approach, the weights are obtained by utilizing component classifiers. Although this algorithm uses a principle similar to that of a weighted stacked ensemble, it was mainly developed for multi-label stream classification, and their approach neither considered a classifier selection ensemble nor pairwise label correlation. In contrast, our approach adopts sparsity for regularization of the classifier selection ensemble and uses the cosine similarity to calculate the label correlation matrix. Therefore, our approach not only improves the interaction between base classifiers but also the computing efficiency.

Although the experimental results demonstrate that our proposed approach achieves competitive performance on a variety of datasets, it still causes a few problems that could directly affect its practical application. Similar to other multi-label stacking algorithms, our approach also needs to generate meta-level features, which adds to the computational cost. Therefore, our approach could be problematic for extreme multi-label classification [60]. Reaching a compromise between computational efficiency and classification accuracy for extreme multi-label ensemble classification would be an interesting and challenging research topic.

6. Conclusion

In this paper, we proposed a novel weighted classifier selection and stacked ensemble for multi-label classification, MLWSE, which uses sparsity for regularization to facilitate classifier selection and ensemble construction, and simultaneously exploits the classifier weights and label correlation to improve the classification performance. On the other hand, our model can be seen as a label meta-specific-feature selection method, and it can be used with any existing multi-label classification algorithm as its base classifier. We compared our method MLWSE-L1 and MLWSE-L21 with several well-established multi-label ensemble classification algorithms on 13 multi-label benchmark datasets and real Cardiovascular and Cerebrovascular Disease datasets. The results of the comparison confirmed the competitive performance of our proposed method, and verified the effectiveness of the weighted stacked ensemble.

Acknowledgment

The authors would like to thank KDIS Research Group for providing the multi-label benchmark datasets and Mulan code online to enable us to complete the comparative studies. This work was supported in part by the Natural Science Foundation of China under Grant 61402397 and Grant 61663046, and in part by the Program for Excellent Young Talents of Yunnan University.

References

[1] A. McCallum, Multi-label text classification with a mixture model trained by EM, in: AAAI Workshop Text Learn. 1999.
[2] C. Sanden and J. Z. Zhang, Enhancing multi-label music genre classification through ensemble techniques, in: Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval, 2011, pp. 705-714, doi:10.1145/2009916.2010011.
[3] H. Weng, Z. Liu, A. Maxwell, X. Li, C. Zhang, E. Peng, G. Li, Multi-label symptom analysis and modeling of TCM diagnosis of hypertension, in: IEEE International Conference on Bioinformatics and Biomedicine (BIBM), 2018, pp. 1922-1929, doi:10.1109/bibm.2018.8621173.
[4] B. Jin, B. Muller, C. Zhai, X. Lu, Multi-label literature classification based on the Gene Ontology graph, BMC Bioinformatics, 2008, doi: 10.1186/1471-2105-9-525.
[5] M. Zhang, Z. Zhou, A review on multi-label learning algorithms, IEEE Trans. Knowl. Data Eng. 26 (8) (2014) 1819-1837.
[6] P. Li, H. Li, M. Wu, Multi-label ensemble based on variable pairwise constraint projection, Information Sciences, 222 (2013) 269-281.
[7] G. Madjarov, D. Kocev, D. Gjorgjevikj, S. Džeroski, An extensive experimental comparison of methods for multi-label learning, Pattern Recognition, 45 (9) (2012) 3084-3104.
[8] M. R. Boutell, J. Luo, X. Shen, C. M. Brown, Learning multi-label scene classification, Pattern Recognition, 37 (9) (2004) 1757-1771.
[9] J. Read, B. Pfahringer, G. Holmes, E. Frank, Classifier chains for multi-label classification, Machine Learning, 85 (2011) 333-359.
[10] J. Fürnkranz, E. Hüllermeier, E. Loza Mencía, K. Brinker, Multilabel classification via calibrated label ranking, Machine Learning, 73 (2) (2008) 133-153.
[11] G. Tsoumakas, I. Katakis, I. Vlahavas, Random k-labelsets for multi-label classification, IEEE Trans. Knowl. Data Eng., 23 (7) (2011) 1079-1089.
[12] M. Zhang, Z. Zhou, ML-KNN: A lazy learning approach to multi-label learning, Pattern Recognition, 40 (7) (2007) 2038-2048.
[13] A. Clare, A, R. D. King, Knowledge discovery in multi-label phenotype data, in: European Conference on Principles of Data Mining and Knowledge Discovery, 2001, pp. 42-53.
[14] A. Elisseeff, J. Weston, A kernel method for multi-labelled classification, in: Advances in Neural Information Processing Systems, 2002, pp. 681-687.
[15] Y. Yang, J. Jiang, Bi-weighted ensemble via HMM-based approaches for temporal data clustering, Pattern Recognition, 76 (2018) 391-403.
[16] M. Amozegar, K. Khorasani, An ensemble of dynamic neural network identifiers for fault detection and isolation of gas turbine engines, Neural Networks,76 (2016) 106-121.

[17] O. Gharroudi, H. Elghazel, A. Aussem, Ensemble multi-label classification: A comparative study on threshold selection and voting methods, in: IEEE 27th International Conference on Tools with Artificial Intelligence (ICTAI), 2015, pp. 377-384.

[18] J. M. Moyano, E. L. Gibaja, K. J. Cios, S. Ventura, Review of ensembles of multi-label classifiers: Models, experimental study and prospects, Information Fusion, 44 (2018) 33-45.

[10] E. Gibaja, S. Ventura, A tutorial on multilabel learning, ACM Comput. Surv., 47 (3) (2015) 52:1-38, doi:10.1145/2716262.

[20] Y. Yang, J. Jiang, Adaptive Bi-weighting toward automatic initialization and model selection for HMM-based hybrid meta-clustering ensembles, IEEE Transactions on Cybernetics, 49 (5) (2018) 1657-1668.

[21] G. Tsoumakas, A. Dimou, E. Spyromitros, V. Mezaris, I. Kompatsiaris, I. Vlahavas, Correlation-based pruning of stacked binary relevance models for multi-label learning, in: Proceedings of the 1st International Workshop on Learning from Multi-label Data, 2009, pp. 101–116.

[22] J. Read, B. Pfahringer, G. Holmes, E. Frank, Classifier chains for multi-label classification, in: Joint European Conference on Machine Learning and Knowledge Discovery in Databases, 2009, pp. 254-269.

[23] J. Read, B. Pfahringer , G. Holmes, Multi-label classification using ensmeble pruned sets, in: 8th IEEE International Conference on Data Mining, 2008, pp. 995-1000.

[24] G. Tsoumakas, I. Katakis, I. Vlahavas, Mining multi-label data, in: Data Mining and Knowledge Discovery Handbook, 2009, pp. 667-685.

[25] L. Tenenboim-Chekina, L. Rokach, B. Shapira, Identification of label dependencies for multi-label classification, in: Working Notes of the Second International Workshop on Learning from Multi-Label Data, 2010, pp. 53-60.

[26] D. Kocev, C. Vens, J. Struyf, S. Džeroski, Ensembles of multi-objective decision trees, in: European Conference on Machine Learning, 2007, pp. 624-631.

[27] D. H. Wolpert, Stacked generalization. Neural networks, 5 (2) (1992) 241-259.

[28] L. Wang, H. Shen, H. Tian, Weighted ensemble classification of multi-label data streams, in: Pacific-Asia Conference on Knowledge Discovery and Data Mining, 2017, pp. 551-562.

[29] W. Qu, Y. Zhang, J. Zhu, Q. Qiu, Mining multi-label concept-drifting data streams using dynamic classifier ensemble, in: Asian Conference on Machine Learning 2009, pp. 308-321.

[30] R. E. Schapire, Y. Singer, BoosTexter: A boosting-based system for text categorization, Machine learning, 39 (2000) 135-168.

[31] Y. Yang, J. Jiang, Hybrid sampling-based clustering ensemble with global and local constitutions, IEEE Transactions on Neural Networks and Learning Systems, 27 (5) (2015) 952-965.

[32] F. Tai, H. T. Lin, Multilabel classification with principal label space transformation. Neural Computation, 24 (9) (2012), 2508-2542.

[33] E. Spyromitros-Xioufis, G. Tsoumakas, W. Groves, I. Vlahavas, Multi-target regression via input space expansion: treating targets as inputs, Machine Learning, 104 (1) (2016), 55-98.

[34] M. U. Sen, H. Erdogan, Max-margin stacking and sparse regularization for linear classifier combination and selection, arXiv:1106.1684, 2011.

[35] F. Gunes, SAS Institute Inc, Penalized regression methods for linear models in SAS/STAT, 2015.

[36] R. Tibshirani, Regression shrinkage and selection via the lasso: a retrospective, Journal of the Royal Statistical Society: Series B (Statistical Methodology), 73 (3) (2011) 273-282.

[37] C. Cui, D. Wang, High dimensional data regression using Lasso model and neural networks with random weights, Information Sciences, 372 (2016) 505-517.

[38] M. Yuan, Y. Lin, Model selection and estimation in regression with grouped variables. Journal of the Royal Statistical Society: Series B (Statistical Methodology), 68 (1) (2006) 49-67.

[39] A. Jalali, S. Sanghavi, C. Ruan, P. K. Ravikumar, A dirty model for multi-task learning, in: Advances in Neural Information Processing Systems, 2010, pp. 964-972.

[40] S. Pan, J. Wu, X. Zhu, G. Long, C. Zhang, Task sensitive feature exploration and learning for multitask graph classification. IEEE Transactions on Cybernetics, 47 (3) (2017) 744-758.

[41] S. Vluymans, C. Cornelis, F. Herrera, Y. Saeys, Multi-label classification using a fuzzy rough neighborhood consensus, Information Sciences, 433-434 (2018) 96-114.

[42] A. Beck, M. Teboulle, A fast iterative shrinkage-thresholding algorithm for linear inverse problems, SIAM journal on Imaging Sciences, 2 (1) (2009) 183-202.

[43] N. Ito, A. Takeda, K. C. Toh, A unified formulation and fast accelerated proximal gradient method for classification, Journal of Machine Learning Research, 18 (2017) 510-558.

[44] V. Kumar, A. K. Pujari, V. Padmanabhan, V. R. Kagita, Group preserving label embedding for multi-label classification, Pattern Recognition, 90 (2019) 23-34.

[45] J. Nutini, M. Schmidt, I. Laradji, M. Friedlander, H. Koepke, Coordinate descent converges faster with the gauss-southwell rule than random selection, in: International Conference on Machine Learning, 2015, pp. 1632-1641.

[46] J. Huang, G. Li, Q. Huang, X. Wu, Joint feature selection and classification for multilabel learning, IEEE Transactions on Cybernetics, 48 (3) (2017) 876-889.

[47] N. Simon, J. Friedman, T. Hastie, R. Tibshirani, A sparse-group lasso, Journal of Computational and Graphical Statistics, 22 (2) (2013) 231-245.

[48] A. Catalina, C. M. Alaíz, J. R. Dorronsoro, Accelerated Block Coordinate Descent for Sparse Group Lasso, in: IJCNN, 2018, pp. 1-8.

[49] G. Tsoumakas, E. Spyromitros-Xioufis, J. Vilcek, I. Vlahavas, Mulan: A java library for multi-label learning, Journal of Machine Learning Research, 12 (2011) 2411-2414.

[50] J. Read, P. Reutemann, B. Pfahringer, G. Holmes, Meka: a multi-label/multi-target extension to weka, Journal of Machine Learning Research, 17 (2016) 667-671.

[51] P. Szymanski, T. Kajdanowicz, Scikit-multilearn: a scikit-based Python environment for performing multi-label classification, Journal of Machine Learning Research, 20 (2019), 209-230.

[52] M. Friedman. A comparison of alternative tests of significance for the problem of m rankings. Annals of Mathematical Statistics, 11 (1940) 86-92.

[53] J. Demšar, Statistical comparisons of classifiers over multiple data sets, Journal of Machine learning research, 7 (2006) 1-30.

[54] R. Subha, K. Anandakumar, A. Bharathi, Study on Cardiovascular Disease Classification Using Machine Learning Approaches, International Journal of Applied Engineering Research, 11 (2016) 4377-4380.

[55] M. Deng, C. Wang, M. Tang, T. Zheng, Extracting cardiac dynamics within ECG signal for human identification and cardiovascular diseases classification, Neural Networks, 100 (2018) 70-83.

[56] L. Sun, M. Kudo, K. Kimura, Multi-label classification with meta-label-specific features, in: ICPR, 2016, pp. 1612-1617.

[57] J. Huang, G. Li, Q. Huang, X. Wu, Learning label specific features for multi-label classification, in: IEEE International Conference on Data Mining, 2015, pp. 181-190.

[58] T. Zhou, D. Tao, Multi-label subspace ensemble, Journal of Machine Learning Research, 2012, pp. 1444-1452.

[59] A. Büyükçakir, H. Bonab, F. Can, A novel online stacked ensemble for multi-label stream classification, in: Proceedings of the 27th ACM International Conference on Information and Knowledge Management, 2018, pp. 1063-1072.

[60] W. Zhang, J. Yan, X. Wang, H. Zha, Deep extreme multi-label learning, in: Proceedings of the 2018 ACM on International Conference on Multimedia Retrieval, 2018, pp. 100-107.