Front. Comput. Sci., 2018, 12(2): 191–202
https://doi.org/10.1007/s11704-017-7031-7

REVIEW ARTICLE

Binary relevance for multi-label learning: an overview

Min-Ling ZHANG (✉)[1,2,3], Yu-Kun LI[1,2,3], Xu-Ying LIU[1,2,3], Xin GENG[1,2,3]

1 School of Computer Science and Engineering, Southeast University, Nanjing 210096, China
2 Key Laboratory of Computer Network and Information Integration (Southeast University), Ministry of Education, China
3 Collaborative Innovation Center for Wireless Communications Technology, Nanjing 211100, China

Abstract Multi-label learning deals with problems where each example is represented by a single instance while being associated with multiple class labels simultaneously. Binary relevance is arguably the most intuitive solution for learning from multi-label examples. It works by decomposing the multi-label learning task into a number of *independent* binary learning tasks (one per class label). In view of its potential weakness in ignoring correlations between labels, many correlation-enabling extensions to binary relevance have been proposed in the past decade. In this paper, we aim to review the state of the art of binary relevance from three perspectives. First, basic settings for multi-label learning and binary relevance solutions are briefly summarized. Second, representative strategies to provide binary relevance with label correlation exploitation abilities are discussed. Third, some of our recent studies on binary relevance aimed at issues other than label correlation exploitation are introduced. As a conclusion, we provide suggestions on future research directions.

Keywords machine learning, multi-label learning, binary relevance, label correlation, class-imbalance, relative labeling-importance

1 Introduction

Multi-label learning is a popular learning framework for modeling real-world objects with multiple semantic meanings [1,2]. For instance, in text categorization, a news document on government reform can cover multiple topics, such as *politics*, *economics*, and *society* [3]; in image classification, a natural scene image can depict multiple types of scenery, such as *sky*, *sand*, *sea*, and *yacht* [4]. Multi-label objects exist in many real-world applications, including information retrieval [5], bioinformatics [6], multimedia content annotation [7], and Web mining [8].

The goal of multi-label learning is to induce a multi-label predictor that can assign a set of relevant labels for the unseen instance. In order to achieve this, the most intuitive solution is to learn one binary classifier for each class label, where the relevance of each class label for the unseen instance is determined by the prediction yielded by the corresponding binary classifier [9]. Specifically, the *binary relevance* procedure works in an *independent* manner, where the binary classifier for each class label is learned by ignoring the existence of other class labels. Due to its conceptual simplicity, binary relevance has attracted considerable attention in multi-label learning research.[1)]

However, a consensus assumption for multi-label learning is that the correlations between labels should be exploited in order to build multi-label prediction models with strong generalization ability [1,2,10,11]. The decomposition nature of binary relevance leads to its inability to exploit label correlations. Therefore, many correlation-enabling extensions to binary relevance have been proposed in the past decade [12–29]. Generally, representative strategies to provide binary relevance with label correlation exploitation abilities include the *chaining* structure assuming random label correlations, the

Received January 25, 2017; accepted July 4, 2017

E-mail: zhangml@seu.edu.cn

[1)] According to Google Scholar (June 2017), the seminal work on binary relevance [9] has received more than 1,100 citations *stacking* structure assuming full-order label correlations, and the *controlling* structure assuming pruned label correlations.

Although label correlation plays an essential role in inducing effective multi-label learning models, recent studies have shown that some inherent properties of multi-label learning should also be investigated in order to achieve good generalization performance. On one hand, class labels in the label space typically have *imbalanced distributions*, meaning the number of positive instances w.r.t. each class label is far less than its negative counterpart [30–39]. On the other hand, class labels in the label space typically have *different labeling-importance*, meaning the importance degrees of each class label for characterizing the semantics of a multi-label example are relative to each other [40–45]. Therefore, in order to enhance the generalization performance of binary relevance models, it is beneficial to consider these inherent properties in addition to label correlation exploitation during the learning procedure.

In this paper, we aim to provide an overview of the state of the art of binary relevance for multi-label learning. In Section 2, formal definitions for multi-label learning, as well as the canonical binary relevance solution are briefly summarized. In Section 3, representative strategies to provide label correlation exploitation abilities to binary relevance are discussed. In Section 4, some of our recent studies on related issues regarding binary relevance are introduced. Finally, Section 5 provides suggestions for several future research directions regarding binary relevance.

2 Binary relevance

Let $\mathcal{X} = \mathbb{R}^d$ denote the d-dimensional instance space and let $\mathcal{Y} = \{\lambda_1, \lambda_2, \ldots, \lambda_q\}$ denote the label space, consisting of q class labels. The goal of multi-label learning is to induce a multi-label predictor $f : \mathcal{X} \mapsto 2^{\mathcal{Y}}$ from the multi-label training set $\mathcal{D} = \{(\boldsymbol{x}^i, \boldsymbol{y}^i) \mid 1 \leqslant i \leqslant m\}$. Here, for each multi-label training example $(\boldsymbol{x}^i, \boldsymbol{y}^i)$, $\boldsymbol{x}^i \in \mathcal{X}$ is a d-dimensional feature vector $[x_1^i, x_2^i, \ldots, x_d^i]^\top$ and $\boldsymbol{y}^i \in \{-1, +1\}^q$ is a q-bit binary vector $[y_1^i, y_2^i, \ldots, y_q^i]^\top$, with $y_j^i = +1$ (-1) indicating that y_j^i is a relevant (or irrelevant) label for $\boldsymbol{x}^i$.[2) Equivalently, the set of relevant labels $Y^i \subseteq \mathcal{Y}$ for $\boldsymbol{x}^i$ corresponds to $Y^i = \{\lambda_j \mid y_j^i = +1, 1 \leqslant j \leqslant q\}$. Given an unseen instance $\boldsymbol{x}^* \in \mathcal{X}$, its relevant label set Y^* is predicted as $Y^* = f(\boldsymbol{x}^*) \subseteq \mathcal{Y}$.

Binary relevance is arguably the most intuitive solution for learning from multi-label training examples [1,2]. It decomposes the multi-label learning problem into q independent binary learning problems. Each binary classification problem corresponds to one class label in the label space $\mathcal{Y}$ [9]. Specifically, for each class label λ_j, binary relevance derives a binary training set $\mathcal{D}_j$ from the original multi-label training set $\mathcal{D}$ in the following manner:

$$\mathcal{D}_j = \{(\boldsymbol{x}^i, y_j^i) \mid 1 \leqslant i \leqslant m\}. \tag{1}$$

In other words, each multi-label training example $(\boldsymbol{x}^i, \boldsymbol{y}^i)$ is transformed into a binary training example based on its *relevancy* to λ_j.

Next, a binary classifier $g_j : \mathcal{X} \mapsto \mathbb{R}$ can be induced from $\mathcal{D}_j$ by applying a binary learning algorithm $\mathcal{B}$, i.e., $g_j \leftarrow\!\shortmid \mathcal{B}(\mathcal{D}_j)$. Therefore, the multi-label training example $(\boldsymbol{x}^i, \boldsymbol{y}^i)$ will contribute to the learning process for all binary classifiers g_j $(1 \leqslant j \leqslant q)$, where $\boldsymbol{x}^i$ is utilized as a positive (or negative) training example for inducing g_j based on its relevancy (or irrelevancy) to λ_j.[3)

Given an unseen instance $\boldsymbol{x}^*$, its relevant label set Y^* is determined by querying the outputs of each binary classifier:

$$Y^* = \{\lambda_j \mid g_j(\boldsymbol{x}^*) > 0,\ 1 \leqslant j \leqslant q\}. \tag{2}$$

As shown in Eq. (2), the predicted label set Y^* will be empty when all binary classifiers yield negative outputs for $\boldsymbol{x}^*$. In this case, one might choose the so-called *T-Criterion* method [9] to predict the class label with the *greatest* (least negative) output. Other criteria for aggregating the outputs of binary classifiers can be found in [9].

Algorithm 1 summarizes the pseudo-code for binary relevance. As shown in Algorithm 1, there are several properties that are noteworthy for binary relevance:

- First, the most prominent property of binary relevance lies in its conceptual simplicity. Specifically, binary relevance is a *first-order* approach that builds a classification model in a label-by-label manner and ignores the existence of other class labels. The modeling complexity of binary relevance is linear to the number of class labels q in the label space;
- Second, binary relevance falls into the category of *problem transformation* approaches, which solve multi-label learning problems by transforming them into other well-established learning scenarios (binary classification in this case) [1,2]. Therefore, binary relevance is not restricted to a particular learning technique and can

[2) Without loss of generality, binary assignment of each class label is represented by +1 and −1 (rather than 1 and 0) in this paper

[3) In the seminal literature on binary relevance [9], this training procedure is also referred to as *cross-training* be instantiated with various binary learning algorithms with diverse characteristics;

- Third, binary relevance optimizes macro-averaged *label-based* multi-label evaluation metrics, which evaluate the learning system's performance on each class label separately, and then return the mean value across all class labels. Therefore, the actual multi-label metric being optimized depends on the binary loss, which is minimized by the binary learning algorithm $\mathcal{B}$ [46,47];
- Finally, binary relevance can be easily adapted to learn from multi-label examples with missing labels, where the labeling information for training examples is incomplete due to various factors, such as high labeling cost, carelessness of human labelers, etc. [48–50]. In order to accommodate this situation, binary relevance can derive the binary training set in Eq. (1) by simply excluding examples whose labeling information y_j^i is not available.

Algorithm 1 Pseudo-code for binary relevance [9]

```
Inputs:
$\mathcal{D}$: Multi-label training set $\{(\boldsymbol{x}^i, \boldsymbol{y}^i) \mid 1 \leqslant i \leqslant m\}$
    $(\boldsymbol{x}^i \in \mathcal{X}, \boldsymbol{y}^i \in \{-1,+1\}^q, \mathcal{X} = \mathbb{R}^d, \mathcal{Y} = \{\lambda_1, \lambda_2, \ldots, \lambda_q\})$
$\mathcal{B}$: Binary learning algorithm
$\boldsymbol{x}^*$: Unseen instance $(\boldsymbol{x}^* \in \mathcal{X})$

Outputs:
$Y^*$: Predicted label set for $\boldsymbol{x}^*$ $(Y^* \subseteq \mathcal{Y})$

Process:
for $j = 1$ to $q$ do
    Derive the binary training set $\mathcal{D}_j$ according to Eq. (1);
    Induce the binary classifier $g_j : \mapsfrom \mathcal{B}(\mathcal{D}_j)$;
end for
return $Y^* = \{\lambda_j \mid g_j(\boldsymbol{x}^*) > 0,\ 1 \leqslant j \leqslant q\}$
```

3 Correlation-enabling extensions

As discussed in Section 2, binary relevance has been used widely for multi-label modeling due to its simplicity and other attractive properties. However, one potential weakness of binary relevance lies in its inability to exploit label correlations to improve the learning system's generalization ability [1,2]. Therefore, a natural consideration is to attempt to provide binary relevance with label correlation exploitation abilities while retaining its linear modeling complexity w.r.t. the number of class labels.

In light of the above consideration, many correlation-enabling extensions have been proposed following the seminal work on binary relevance. In the following sections, three representative extension strategies are discussed: the chaining structure assuming *random* label correlations [12–18], the stacking structure assuming *full-order* label correlations [19–23], and the controlling structure assuming *pruned* label correlations [24–29].

3.1 Binary relevance with the chaining structure

In the chaining structure, a total of q binary classifiers are induced based on a chaining order specified over the class labels. Specifically, one binary classifier is built for each class label based on the predictions of the preceding classifiers in the chain [12,14].

Given the label space $\mathcal{Y} = \{\lambda_1, \lambda_2, \ldots, \lambda_q\}$, let $\pi : \{1, 2, \ldots, q\} \mapsto \{1, 2, \ldots, q\}$ be the permutation used to specify a chaining order over all class labels, i.e., $\lambda_{\pi(1)} \succ \lambda_{\pi(2)} \succ \cdots \succ \lambda_{\pi(q)}$. Thereafter, for the jth class label $\lambda_{\pi(j)}$ in the ordered list, the *classifier chain* approach [12,14] works by deriving a corresponding binary training set $\mathcal{D}_{\pi(j)}$ from $\mathcal{D}$ in the following manner:

$$\mathcal{D}_{\pi(j)} = \left\{ \left(\left[\boldsymbol{x}^i, y^i_{\pi(1)}, \ldots, y^i_{\pi(j-1)} \right], y^i_{\pi(j)} \right) \,\middle|\, 1 \leqslant i \leqslant m \right\}. \tag{3}$$

Here, the binary assignments of preceding class labels in the chain, i.e., $\left[y^i_{\pi(1)}, \ldots, y^i_{\pi(j-1)} \right]$, are treated as additional features to append to the original instance $\boldsymbol{x}^i$.

Next, a binary classifier $g_{\pi(j)} : \mathcal{X} \times \{-1,+1\}^{j-1} \mapsto \mathbb{R}$ can be induced from $\mathcal{D}_{\pi(j)}$ by applying a binary learning algorithm $\mathcal{B}$, i.e., $g_{\pi(j)} \mapsfrom \mathcal{B}(\mathcal{D}_{\pi(j)})$. In other words, $g_{\pi(j)}$ determines the relevancy of $\lambda_{\pi(j)}$ by exploiting its correlations with the preceding labels $\lambda_{\pi(1)}, \ldots, \lambda_{\pi(j-1)}$ in the chain.

Given an unseen instance $\boldsymbol{x}^*$, its relevant label set Y^* is determined by iteratively querying the outputs of each binary classifier along the chaining order. Let $\eta^{\boldsymbol{x}^*}_{\pi(j)} \in \{-1,+1\}$ denote the predicted binary assignment of $\lambda_{\pi(j)}$ on $\boldsymbol{x}^*$, which is recursively determined as follows:

$$\begin{aligned} \eta^{\boldsymbol{x}^*}_{\pi(1)} &= \operatorname{sign}\left[g_{\pi(1)}(\boldsymbol{x}^*) \right], \\ \eta^{\boldsymbol{x}^*}_{\pi(j)} &= \operatorname{sign}\left[g_{\pi(j)}\left(\left[\boldsymbol{x}^*, \eta^{\boldsymbol{x}^*}_{\pi(1)}, \ldots, \eta^{\boldsymbol{x}^*}_{\pi(j-1)} \right] \right) \right]. \end{aligned} \tag{4}$$

Here, sign[$\cdot$] represents the signed function. Therefore, the relevant label set Y^* is derived as:

$$Y^* = \left\{ \lambda_{\pi(j)} \mid \eta^{\boldsymbol{x}^*}_{\pi(j)} = +1,\ 1 \leqslant j \leqslant q \right\}. \tag{5}$$

Algorithm 2 presents the pseudo-code of the classifier chain. As shown in Algorithm 2, the classifier chain is a *high-order* approach that considers correlations between labels in a random manner specified by the permutation π. In order to account for the randomness introduced by the permutation ordering, one effective choice is to build an *ensemble* of classifier chains with n random permutations $\{\pi^r \mid 1 \leqslant r \leqslant n\}$. One classifier chain can be learned based on each random permutation, and then the outputs from all classifier chains are aggregated to yield the final prediction [12,14,16].

Algorithm 2 Pseudo-code of the classifier chain [12,14]

```
Inputs:
$\mathcal{D}$: Multi-label training set $\{(\boldsymbol{x}^i, \boldsymbol{y}^i) \mid 1 \leqslant i \leqslant m\}$
    $(\boldsymbol{x}^i \in \mathcal{X}, \boldsymbol{y}^i \in \{-1,+1\}^q, \mathcal{X} = \mathbb{R}^d, \mathcal{Y} = \{\lambda_1, \lambda_2, \ldots, \lambda_q\})$
$\pi$: Permutation used to specify chaining order
$\mathcal{B}$: Binary learning algorithm
$\boldsymbol{x}^*$: Unseen instance $(\boldsymbol{x}^* \in \mathcal{X})$

Outputs:
$Y^*$: Predicted label set for $\boldsymbol{x}^*$ $(Y^* \subseteq \mathcal{Y})$

Process:
for $j = 1$ to $q$ do
    Derive the binary training set $\mathcal{D}_{\pi(j)}$ using Eq. (3);
    Induce the binary classifier $g_{\pi(j)} : \leftarrow\!\!\mid \mathcal{B}(\mathcal{D}_{\pi(j)})$;
end for
Determine the binary assignments $\eta^{\boldsymbol{x}^*}_{\pi(j)}$ $(1 \leqslant j \leqslant q)$ using Eq. (4);
return $Y^* = \left\{\lambda_{\pi(j)} \mid \eta^{\boldsymbol{x}^*}_{\pi(j)} = +1,\ 1 \leqslant j \leqslant q\right\}$ w.r.t. Eq. (4)
```

It is also worth noting that predictive errors incurred in preceding classifiers will be propagated to subsequent classifiers along the chain. These undesirable influences become more pronounced if error-prone class labels happen to be placed at early chaining positions [12,14,28,51]. Furthermore, during the training phase, the additional features appended to the input space $\mathcal{X}$ correspond to the ground-truth labeling assignments (Eq. (3)). However, during the testing phase, the additional features appended to $\mathcal{X}$ correspond to predicted labeling assignments (Eq. (4)). One way to rectify this discrepancy is to replace the extra features $\left[y^i_{\pi(1)}, \ldots, y^i_{\pi(j-1)}\right]$ in Eq. (3) with $\left[\eta^{\boldsymbol{x}^i}_{\pi(1)}, \ldots, \eta^{\boldsymbol{x}^i}_{\pi(j-1)}\right]$ such that the predicted labeling assignments are appended to $\mathcal{X}$ in both the training and testing phases [17,51].

From a statistical point of view, the task of multi-label learning is equivalent to learning the conditional distribution $p(\boldsymbol{y} \mid \boldsymbol{x})$ with $\boldsymbol{x} \in \mathcal{X}$ and $\boldsymbol{y} \in \{-1,+1\}^q$. Therefore, $p(\boldsymbol{y} \mid \boldsymbol{x})$ can be factorized w.r.t. the chaining order specified by π as follows:

$$p(\boldsymbol{y} \mid \boldsymbol{x}) = \prod_{j=1}^{q} p\left(y_{\pi(j)} \mid \boldsymbol{x}, y_{\pi(1)}, \ldots, y_{\pi(j-1)}\right). \tag{6}$$

Here, each term on the RHS of Eq. (6) represents the conditional probability of observing $y_{\pi(j)}$ given $\boldsymbol{x}$ and its preceding labels in the chain. Specifically, this term can be estimated by utilizing a binary learning algorithm $\mathcal{B}$, which is capable of yielding probabilistic outputs (e.g., Naive Bayes). Thereafter, the relevant label set for the unseen instance is predicted by performing exact inference [13] or approximate inference (when q is large) over the probabilistic classifier chain [15,18].

3.2 Binary relevance with the stacking structure

In the stacking structure, a total of $2q$ binary classifiers are induced by stacking a set of q meta-level binary relevance models over another set of q base-level binary relevance models. Specifically, each meta-level binary classifier is built upon the predictions of all base-level binary classifiers [19].

Following the notations in Section 2, let g_j $(1 \leqslant j \leqslant q)$ denote the set of base-level classifiers learned by invoking the standard binary relevance procedure on the multi-label training set, i.e., $g_j \leftarrow\!\!\mid \mathcal{B}(\mathcal{D}_j)$. Thereafter, for each class label λ_j, the *stacking aggregation* approach [1,19] derives a meta-level binary training set $\mathcal{D}^M_j$ in the following manner:

$$\mathcal{D}^M_j = \left\{\left(\left[\boldsymbol{x}^i, \mathrm{sign}[g_1(\boldsymbol{x}^i)], \ldots, \mathrm{sign}[g_q(\boldsymbol{x}^i)]\right], y^i_j\right) \,\middle|\, 1 \leqslant i \leqslant m\right\}. \tag{7}$$

Here, the signed predictions of base-level classifiers, i.e., $\left[\mathrm{sign}[g_1(\boldsymbol{x}^i)], \ldots, \mathrm{sign}[g_q(\boldsymbol{x}^i)]\right]$, are treated as additional features to append to the original instance $\boldsymbol{x}^i$ in the meta-level.

Next, a meta-level classifier $g^M_j : \mathcal{X} \times \{-1,+1\}^q \mapsto \mathbb{R}$ can be induced from $\mathcal{D}^M_j$ by applying a binary learning algorithm $\mathcal{B}$, i.e., $g^M_j \leftarrow\!\!\mid \mathcal{B}(\mathcal{D}^M_j)$. In other words, g^M_j determines the relevancy of λ_j by exploiting its correlations with all the class labels.

Given an unseen instance $\boldsymbol{x}^*$, its relevant label set Y^* is determined by using the outputs of the base-level classifiers as extra inputs for the meta-level classifiers:

$$\begin{gathered} Y^* = \left\{\lambda_j \mid g^M_j(\tau^{\boldsymbol{x}^*}) > 0,\ 1 \leqslant j \leqslant q\right\}, \\ \text{where } \tau^{\boldsymbol{x}^*} = \left[\boldsymbol{x}^*, \mathrm{sign}[g_1(\boldsymbol{x}^*)], \ldots, \mathrm{sign}[g_q(\boldsymbol{x}^*)]\right]. \end{gathered} \tag{8}$$

Algorithm 3 presents the pseudo-code for stacking aggregation. As shown in Algorithm 3, stacking aggregation is a *full-order* approach that assumes that each class label has correlations with all other class labels. It is worth noting that stacking aggregation employs ensemble learning [52] to combine two sets of binary relevance models with deterministic label correlation exploitation. Ensemble learning can also be applied to the classifier chain to compensate for its randomness of label correlation exploitation.

Rather than using the outputs of the base-level classifiers $\left[\mathrm{sign}[g_1(\boldsymbol{x}^i)], \ldots, \mathrm{sign}[g_q(\boldsymbol{x}^i)]\right]$ by appending them to the inputs of the meta-level classifiers, it is also feasible to use the ground-truth labeling assignments $\left[y_1^i, \ldots, y_q^i\right]$ to instantiate the meta-level binary training set (i.e., Eq. (21)) [21]. However, similar to the standard classifier chain approach, this practice would also lead to the discrepancy issue regarding the additional features appended to the input space $\mathcal{X}$ in the training and testing phases.

Algorithm 3 Pseudo-code of stacking aggregation [19]

```
Inputs:
  $\mathcal{D}$: Multi-label training set $\{(\boldsymbol{x}^i, \boldsymbol{y}^i) \mid 1 \leqslant i \leqslant m\}$
       $(\boldsymbol{x}^i \in \mathcal{X}, \boldsymbol{y}^i \in \{-1, +1\}^q, \mathcal{X} = \mathbb{R}^d, \mathcal{Y} = \{\lambda_1, \lambda_2, \ldots, \lambda_q\})$
  $\mathcal{B}$: Binary learning algorithm
  $\boldsymbol{x}^*$: Unseen instance $(\boldsymbol{x}^* \in \mathcal{X})$

Outputs:
  $Y^*$: Predicted label set for $\boldsymbol{x}^*$ $(Y^* \subseteq \mathcal{Y})$

Process:
  for $j = 1$ to $q$ do
      Derive the binary training set $\mathcal{D}_j$ using Eq. (1);
      Induce the base-level binary classifier $g_j : \leftarrow\!\dashv \mathcal{B}(\mathcal{D}_j)$;
  end for
  for $j = 1$ to $q$ do
      Derive the binary training set $\mathcal{D}_j^M$ using Eq. (7);
      Induce the meta-level binary classifier $g_j^M : \leftarrow\!\dashv \mathcal{B}(\mathcal{D}_j^M)$;
  end for
  return $Y^* = \left\{\lambda_j \mid g_j^M(\tau^{\boldsymbol{x}^*}) > 0,\ 1 \leqslant j \leqslant q\right\}$ w.r.t Eq. (8)
```

There are other ways to make use of the stacking strategy to induce a multi-label prediction model. Given the base-level classifiers g_j $(1 \leqslant j \leqslant q)$ and the meta-level classifiers g_j^M $(1 \leqslant j \leqslant q)$, rather than relying only on the meta-level classifiers to yield final predictions (i.e., Eq. (8)), one can also aggregate the outputs of both the base-level and meta-level classifiers to accomplish this task [20]. Furthermore, rather than using the binary labeling assignments as additional features for stacking, one can also adapt specific techniques to generate tailored features for stacking, such as discriminant analysis [22] or rule learning [23].

3.3 Binary relevance with the controlling structure

In the controlling structure, a total of $2q$ binary classifiers are induced based on a dependency structure specified over the class labels. Specifically, one binary classifier is built for each class label by exploiting the pruned predictions of q binary relevance models [25].

A Bayesian network (or *directed acyclic graph*, DAG) is a convenient tool to explicitly characterize correlations between class labels in a compact manner [25–27]. As mentioned in Subsection 3.1, a statistical equivalence for multi-label learning corresponds to modeling the conditional distribution $p(\boldsymbol{y} \mid \boldsymbol{x})$ with $\boldsymbol{x} \in \mathcal{X}$ and $\boldsymbol{y} = \{-1, +1\}^q$. Given the Bayesian network structure $\mathcal{G}$ specified over $(\boldsymbol{x}, \boldsymbol{y})$, the conditional distribution $p(\boldsymbol{y} \mid \boldsymbol{x})$ can be factorized based on $\mathcal{G}$ as follows:

$$p(\boldsymbol{y} \mid \boldsymbol{x}) = \prod_{j=1}^{q} p(y_j \mid \mathbf{pa}_j, \boldsymbol{x}). \tag{9}$$

Here, $\boldsymbol{x}$ serves as the common parent for each y_j $(1 \leqslant j \leqslant q)$ because all class labels inherently depend on the feature space $\mathcal{X}$. Additionally, $\mathbf{pa}_j$ represents the set of parent class labels of y_j implied by $\mathcal{G}$. Figure 1 illustrates two examples of how the conditional distribution $p(\boldsymbol{y} \mid \boldsymbol{x})$ can be factorized based on the given Bayesian network structure.

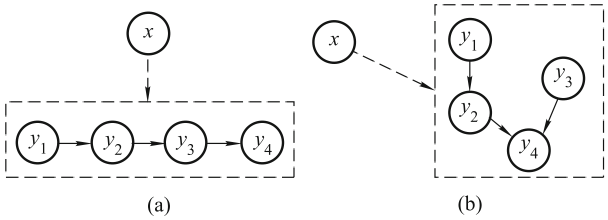


Fig. 1 Examples of two Bayesian network (DAG) structures with $\boldsymbol{x}$ serving as the common parent. The conditional distribution $p(\boldsymbol{y} \mid \boldsymbol{x})$ factorizes based on each structure as: (a) $p(\boldsymbol{y} \mid \boldsymbol{x}) = p(y_1 \mid \boldsymbol{x}) \cdot p(y_2 \mid y_1, \boldsymbol{x}) \cdot p(y_3 \mid y_2, \boldsymbol{x}) \cdot p(y_4 \mid y_3, \boldsymbol{x})$; (b) $p(\boldsymbol{y} \mid \boldsymbol{x}) = p(y_1 \mid \boldsymbol{x}) \cdot p(y_2 \mid y_1, \boldsymbol{x}) \cdot p(y_3 \mid \boldsymbol{x}) \cdot p(y_4 \mid y_2, y_3, \boldsymbol{x})$

Learning a Bayesian network structure $\mathcal{G}$ from a multi-label training set $\mathcal{D} = \{(\boldsymbol{x}^i, \boldsymbol{y}^i) \mid 1 \leqslant i \leqslant m\}$ is difficult. Existing Bayesian network learning techniques [53] are not directly applicable for two major reasons. First, variables in the Bayesian network have mixed types with $\boldsymbol{y}$ (class labels) being discrete and $\boldsymbol{x}$ (feature vector) being continuous. Second, computational complexity is prohibitively high when input dimensionality (i.e., number of features) is too large.

These two issues are brought about by the involvement of the feature vector $\boldsymbol{x}$ when learning the Bayesian network structure. In light of this information, the *LEAD* approach [25] chooses to eliminate the effects of features in order to simplify the Bayesian network generation procedure. Following the notations in Section 2, let g_j $(1 \leqslant j \leqslant q)$ denote the binary classifiers induced by the standard binary relevance procedure, i.e., $g_j \leftarrow\!\dashv \mathcal{B}(\mathcal{D}_j)$. Accordingly, a set of *error* random variables are derived to decouple the influences of $\boldsymbol{x}$ from all class labels:

$$e_j = y_j - \operatorname{sign}(g_j(\boldsymbol{x})) \ \ (1 \leqslant j \leqslant q). \tag{10}$$

Thereafter, the Bayesian network structure $\mathcal{G}$ for all class labels (conditioned on $\boldsymbol{x}$) can be learned from e_j $(1 \leqslant j \leqslant q)$ using off-the-shelf packages [54–56].

Based on the DAG structure implied by $\mathcal{G}$, for each class label λ_j, the LEAD approach derives a binary training set $\mathcal{D}_j^{\mathcal{G}}$

from $\mathcal{D}$ in the following manner:

$$\mathcal{D}_j^{\mathcal{G}} = \left\{\left(\left[\boldsymbol{x}^i, \mathbf{pa}_j^i\right], y_j^i\right) \;\middle|\; 1 \leqslant i \leqslant m\right\}. \tag{11}$$

Here, the binary assignments of parent class labels, i.e., $\mathbf{pa}_j^i$, are treated as additional features to append to the original instance $\boldsymbol{x}^i$.

Next, a binary classifier $g_j^{\mathcal{G}} : \mathcal{X} \times \{-1,+1\}^{|\mathbf{pa}_j|} \mapsto \mathbb{R}$ can be induced from $\mathcal{D}_j^{\mathcal{G}}$ by applying a binary learning algorithm $\mathcal{B}$, i.e., $g_j^{\mathcal{G}} \mapsfrom \mathcal{B}(\mathcal{D}_j^{\mathcal{G}})$. In other words, $g_j^{\mathcal{G}}$ determines the relevancy of λ_j by exploiting its correlations with the parent class labels $\mathbf{pa}_j$ implied by $\mathcal{G}$.

Given an unseen instance $\boldsymbol{x}^*$, its relevant label set Y^* is determined by iteratively querying the outputs of each binary classifier w.r.t. the Bayesian network structure. Let $\pi^{\mathcal{G}} : \{1,2,\ldots,q\} \mapsto \{1,2,\ldots,q\}$ be the causal order implied by $\mathcal{G}$ over all class labels, i.e., $\lambda_{\pi^{\mathcal{G}}(1)} \succ \lambda_{\pi^{\mathcal{G}}(2)} \succ \cdots \lambda_{\pi^{\mathcal{G}}(q)}$. Furthermore, let $\eta_{\pi^{\mathcal{G}}(j)}^{\boldsymbol{x}^*} \in \{-1,+1\}$ denote the predicted binary assignment of $\lambda_{\pi^{\mathcal{G}}(j)}$ on $\boldsymbol{x}^*$, which is recursively determined as follows:

$$\begin{aligned}\eta_{\pi^{\mathcal{G}}(1)}^{\boldsymbol{x}^*} &= \text{sign}\left[g_{\pi^{\mathcal{G}}(1)}^{\mathcal{G}}(\boldsymbol{x}^*)\right],\\ \eta_{\pi^{\mathcal{G}}(j)}^{\boldsymbol{x}^*} &= \text{sign}\left[g_{\pi^{\mathcal{G}}(j)}^{\mathcal{G}}\left(\left[\boldsymbol{x}^*, \langle \eta_a^{\boldsymbol{x}^*}\rangle_{y_a \in \mathbf{pa}_{\pi^{\mathcal{G}}(j)}}\right]\right)\right].\end{aligned} \tag{12}$$

Accordingly, the relevant label set Y^* becomes:

$$Y^* = \left\{\lambda_{\pi^{\mathcal{G}}(j)} \mid \eta_{\pi^{\mathcal{G}}(j)}^{\boldsymbol{x}^*} = +1,\ 1 \leqslant j \leqslant q\right\}. \tag{13}$$

Algorithm 4 Pseudo-code of LEAD [25]

```
Inputs:
  $\mathcal{D}$: Multi-label training set $\{(\boldsymbol{x}^i, \boldsymbol{y}^i) \mid 1 \leqslant i \leqslant m\}$
      $(\boldsymbol{x}^i \in \mathcal{X}, \boldsymbol{y}^i \in \{-1,+1\}^q, \mathcal{X} = \mathbb{R}^d, \mathcal{Y} = \{\lambda_1, \lambda_2, \ldots, \lambda_q\})$
  $\mathcal{B}$: Binary learning algorithm
  $\mathcal{L}$: Bayesian network structure learning algorithm
  $\boldsymbol{x}^*$: Unseen instance $(\boldsymbol{x}^* \in \mathcal{X})$

Outputs:
  $Y^*$: Predicted label set for $\boldsymbol{x}^*$ $(Y^* \subseteq \mathcal{Y})$

Process:
  for $j = 1$ to $q$ do
      Derive the binary training set $\mathcal{D}_j$ using Eq. (1);
      Induce the binary classifier $g_j : \mapsfrom \mathcal{B}(\mathcal{D}_j)$;
  end for
  Derive the error random variables $e_j$ $(1 \leqslant j \leqslant q)$ using Eq. (10);
  Learn the Bayesian network structure $\mathcal{G} \mapsfrom \mathcal{L}(e_1, e_2, \ldots, e_q)$;
  for $j = 1$ to $q$ do
      Derive the binary training set $\mathcal{D}_j^{\mathcal{G}}$ using Eq. (11);
      Induce the binary classifier $g_j^{\mathcal{G}} : \mapsfrom \mathcal{B}(\mathcal{D}_j^{\mathcal{G}})$;
  end for
  Specify the causal order $\pi^{\mathcal{G}}$ over all class labels w.r.t. $\mathcal{G}$;
  return $Y^* = \left\{\lambda_{\pi^{\mathcal{G}}(j)} \mid \eta_{\pi^{\mathcal{G}}(j)}^{\boldsymbol{x}^*} = +1,\ 1 \leqslant j \leqslant q\right\}$ w.r.t. Eq. (12)
```

Algorithm 4 presents the pseudo-code of LEAD. As shown in Algorithm 4, LEAD is a *high-order* approach that controls the order of correlations using the number of parents of each class label implied by $\mathcal{G}$. Similar to stacking aggregation, LEAD also employs ensemble learning to combine two sets of binary classifiers g_j $(1 \leqslant j \leqslant q)$ and $g_j^{\mathcal{G}}$ $(1 \leqslant j \leqslant q)$ to yield the multi-label prediction model. Specifically, predictions of the q binary classifiers g_j are *pruned* w.r.t. the parents for label correlation exploitation.

There are also other ways to consider pruned label correlations with specific controlling structures. First, a tree-based Bayesian network can be utilized as a simplified DAG structure where second-order label correlations are considered by pruning each class label with (up to) one parent [26,27]. Second, the stacking structure can be adapted to fulfill controlled label correlation exploitation by pruning the uncorrelated outputs of base-level classifiers for stacking meta-level classifiers [24,29]. Third, class labels with error-prone predictions can be filtered out of the pool of class labels for correlation exploitation [28].

4 Related issues

As discussed in Section 3, in order to enhance binary relevance, it is necessary to enable label correlation exploitation during the learning process. However, it is also noteworthy that some inherent properties of multi-label learning should be investigated in order to further enhance the generalization ability of binary relevance. Specifically, recent studies on the issue of *class-imbalance*, i.e., the number of positive instances and negative instances w.r.t. each class label are imbalanced [30–39], and the issue of *relative labeling-importance*, i.e., each class label has different labeling-importance [40–45], are introduced.

4.1 Class-imbalance

The issue of class-imbalance exists in many multi-label learning tasks, especially those where the label space consists of a significant number of class labels. For each class label $\lambda_j \in \mathcal{Y}$, let $\mathcal{D}_j^+ = \{(\boldsymbol{x}^i, +1) \mid y_j^i = +1, 1 \leqslant i \leqslant m\}$ and $\mathcal{D}_j^- = \{(\boldsymbol{x}^i, -1) \mid y_j^i = -1, 1 \leqslant i \leqslant m\}$ denote the set of *positive* and *negative* training examples w.r.t. λ_j. The level of class-imbalance can then be characterized by the imbalance ratio:

$$ImR_j = \frac{\max\left(|\mathcal{D}_j^+|, |\mathcal{D}_j^-|\right)}{\min\left(|\mathcal{D}_j^+|, |\mathcal{D}_j^-|\right)}. \tag{14}$$

Here, $|\cdot|$ returns the cardinality of a set and, in most cases,

$|\mathcal{D}_j^+| < |\mathcal{D}_j^-|$ holds. Generally, the imbalance ratio is high for most benchmark multi-label data sets [1,57]. For instance, among the 42 class labels of the `rcv1` benchmark data set, the average imbalance ratio (i.e., $\frac{1}{q}\sum_{j=1}^{q} ImR_j$) is greater than 15 and the maximum imbalance ratio (i.e., $\max_{1\leqslant j\leqslant q} ImR_j$) is greater than 50 [38].

In order to handle the issue of class-imbalance in multi-label learning, existing approaches employ binary relevance as an intermediate step in the learning procedure. Specifically, by decomposing the multi-label learning task into q independent binary learning tasks, each of them can be addressed using prevalent binary imbalance learning techniques, such as over-/under-sampling [32,36,37], thresholding the decision boundary [31,33,34], or optimizing imbalance-specific metrics [30,35,39]. Because standard binary relevance is applied prior to subsequent modeling, existing approaches handle class-imbalance in multi-label learning at the expense of ignoring the exploitation of label correlations.

Therefore, a favorable solution to class-imbalance in multi-label learning is to consider the exploitation of label correlations and the exploration of class-imbalance simultaneously. In light of this information, the COCOA approach was proposed based on a specific strategy called *cross-coupling aggregation* [38]. For each class label λ_j, a binary classifier g_j^I is induced from $\mathcal{D}_j$ (i.e., Eq. (1)) by applying a binary imbalance learning algorithm $\mathcal{B}^I$ [58], i.e., $g_j^I \mapsfrom \mathcal{B}^I(\mathcal{D}_j)$. Additionally, a random subset of K class labels $J_K \subset \mathcal{Y} \setminus \{\lambda_j\}$ is extracted for pairwise cross-coupling with λ_j. For each coupling label $\lambda_k \in J_K$, COCOA derives a tri-class training set $\mathcal{D}_{jk}^{\mathbf{tri}}$ for the label pair (λ_j, λ_k) from $\mathcal{D}$ in the following manner:

$$\mathcal{D}_{jk}^{\mathbf{tri}} = \{(\boldsymbol{x}^i, \psi^{\mathbf{tri}}(\boldsymbol{y}^i, \lambda_j, \lambda_k)) \mid 1 \leqslant i \leqslant m\},$$

$$\text{where } \psi^{\mathbf{tri}}(\boldsymbol{y}^i, \lambda_j, \lambda_k) = \begin{cases} 0, & \text{if } y_j^i = -1 \text{ and } y_k^i = -1; \\ +1, & \text{if } y_j^i = -1 \text{ and } y_k^i = +1; \\ +2, & \text{if } y_j^i = +1. \end{cases} \quad (15)$$

Among the three derived class labels, the first two labels (i.e., 0 and +1) exploit label correlations by considering the joint labeling assignments of λ_j and λ_k w.r.t. $\boldsymbol{y}^i$, and the third class label (i.e., +2) corresponds to the case of λ_j being a relevant label.

Next, a tri-class classifier $g_{jk}^I : \mathcal{X} \times \{0, +1, +2\} \mapsto \mathbb{R}$ can be induced from $\mathcal{D}_{jk}^{\mathbf{tri}}$ by applying a multi-class imbalance learning algorithm $\mathcal{M}^I$ [59–61], i.e., $g_{jk}^I \mapsfrom \mathcal{M}^I(\mathcal{D}_{jk}^{\mathbf{tri}})$. In other words, a total of $K+1$ classifiers, including g_j^I and g_{jk}^I $(\lambda_k \in J_K)$, are induced for the class label λ_j.

Given an unseen instance $\boldsymbol{x}^*$, its relevant label set Y^* is determined by aggregating the predictions of the classifiers induced by the binary and multi-class imbalanced learning algorithms:

$$Y^* = \{\lambda_j \mid f_j(\boldsymbol{x}^*) > t_j,\ 1 \leqslant j \leqslant q\}$$
$$\text{where } f_j(\boldsymbol{x}^*) = g_j^I(\boldsymbol{x}^*) + \sum\nolimits_{\lambda_k \in J_K} g_{jk}^I(\boldsymbol{x}^*, +2). \quad (16)$$

Here, t_j is a bipartition threshold, which is set by optimizing an empirical metric (e.g., F-measure) over $\mathcal{D}_j$.

Algorithm 5 presents the pseudo-code of COCOA. As shown in Algorithm 5, COCOA is a *high-order* approach that considers correlations between labels in a random manner via the K coupling class labels in J_K. Specifically, during the training phase, label correlation exploitation is enabled by an ensemble of pairwise cross-couplings between class labels. During the testing phase, class-imbalance exploration is enabled by aggregating the classifiers induced from the class-imbalance learning algorithms.

Algorithm 5 Pseudo-code of COCOA [38]

```
Inputs:
  $\mathcal{D}$: Multi-label training set $\{(\boldsymbol{x}^i, \boldsymbol{y}^i) \mid 1 \leqslant i \leqslant m\}$
     $(\boldsymbol{x}^i \in \mathcal{X}, \boldsymbol{y}^i \in \{-1,+1\}^q, \mathcal{X} = \mathbb{R}^d, \mathcal{Y} = \{\lambda_1, \lambda_2, \ldots, \lambda_q\})$
  $\mathcal{B}^I$: Binary imbalance learning algorithm
  $\mathcal{M}^I$: Bulti-class imbalance learning algorithm
  $K$: Number of coupling class labels
  $\boldsymbol{x}^*$: Unseen instance $(\boldsymbol{x}^* \in \mathcal{X})$

Outputs:
  $Y^*$: Predicted label set for $\boldsymbol{x}^*$ $(Y^* \subseteq \mathcal{Y})$

Process:
  for $j = 1$ to $q$ do
     Derive the binary training set $\mathcal{D}_j$ using Eq. (1);
     Induce the binary classifier $g_j^I : \mapsfrom \mathcal{B}^I(\mathcal{D}_j)$;
     Extract a random subset $J_K \subset \mathcal{Y} \setminus \{\lambda_j\}$ with $K$ coupling class labels;
     for each $\lambda_k \in J_K$ do
        Derive the tri-class training set $\mathcal{D}_{jk}^{\mathbf{tri}}$ using Eq. (15);
        Induce the tri-class classifier $g_{jk}^I : \mapsfrom \mathcal{M}^I(\mathcal{D}_{jk}^{\mathbf{tri}})$;
     end for
  end for
  Return $Y^* = \{\lambda_j \mid f_j(\boldsymbol{x}^*) > t_j,\ 1 \leqslant j \leqslant q\}$ w.r.t. Eq. (16)
```

4.2 Relative labeling-importance

Existing approaches to multi-label learning, including binary relevance, make the common assumption of equal labeling-importance. Here, class labels associated with the training example are regarded to be relevant, while their relative importance in characterizing the example's semantics is not differentiated [1,2]. Nevertheless, the degree of labeling importance for each associated class label is generally different and not directly accessible from multi-label training examples. Figure 2 shows an example multi-label natural scene image with descending relative labeling-importance: *sky* ≻ *water* ≻ *cloud* ≻ *building* ≻ *pedestrian*. Similar situations hold for other types of multi-label objects, such as multi-category documents with different topical importance and multi-functionality genes with different expression levels.

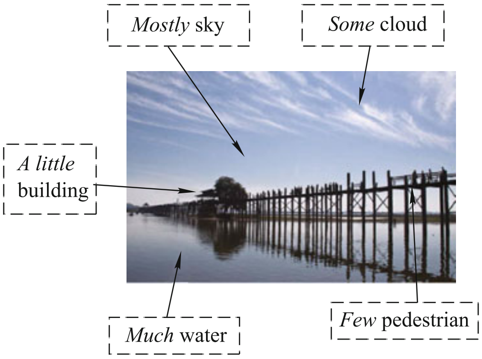


Fig. 2 An example natural scene image annotated with multiple class labels (The relative labeling-importance of each annotation is also illustrated in this figure, although it is not explicitly provided by the annotator [42])

It is worth noting that there have been studies on multi-label learning that have aimed to make use of *auxiliary* labeling-importance information. Different forms of auxiliary information exist, including *ordinal scale* over each class label [40], *full ranking* over relevant class labels [41], *importance distribution* over all class labels [43,44], and *oracle feedbacks* over queried labels of unlabeled examples [45]. However, in standard multi-label learning, this auxiliary information are not assumed to be available and the only accessible labeling information is the relevancy/irrelevancy of each class label.

By leveraging *implicit* relative labeling-importance information, further improvement in the generalization performance of the multi-label learning system can be expected. In light of this information, the RELIAB approach is proposed to incorporate relative labeling-importance information in the learning process [42]. Formally, for each instance $\boldsymbol{x}$ and class label λ_j, the relative labeling-importance of λ_j in characterizing $\boldsymbol{x}$ is denoted $\mu_{\boldsymbol{x}}^{\lambda_j}$. Specifically, the terms $\mu_{\boldsymbol{x}}^{\lambda_j}$ $(1 \leqslant j \leqslant q)$ satisfy the non-negativity constraint $\mu_{\boldsymbol{x}}^{\lambda_j} \geqslant 0$ and the normalization constraint $\sum_{j=1}^{q} \mu_{\boldsymbol{x}}^{\lambda_j} = 1$.

In the first stage, RELIAB estimates the implicit relative labeling-importance information $\mathcal{U} = \{\mu_{\boldsymbol{x}_i}^{\lambda_j} \mid 1 \leqslant i \leqslant m,\ 1 \leqslant j \leqslant q\}$ through iterative label propagation. Let $G = (V, E)$ be a fully-connected graph constructed over all the training examples with $V = \{\boldsymbol{x}^i \mid 1 \leqslant i \leqslant m\}$. Additionally, a similarity matrix $\mathbf{W} = [w_{ik}]_{m\times m}$ is specified for G as follows:

$$\forall_{i,k=1}^{m}: \quad w_{ik} = \begin{cases} \exp\left(-\frac{\|\boldsymbol{x}^i-\boldsymbol{x}^k\|_2^2}{2\sigma^2}\right), & \text{if } i \neq k; \\ 0, & \text{if } i = k. \end{cases} \tag{17}$$

Here, $\sigma > 0$ is the width constant for similarity calculation. The corresponding label propagation matrix $\mathbf{P}$ is set as follows:

$$\mathbf{P} = \mathbf{D}^{-\frac{1}{2}}\mathbf{W}\mathbf{D}^{-\frac{1}{2}},$$
$$\text{where } \mathbf{D} = \mathrm{diag}[d_1, d_2, \ldots, d_m] \text{ with } d_i = \sum\nolimits_{k=1}^{m} w_{ik}. \tag{18}$$

Additionally, the labeling-importance matrix $\mathbf{R} = [r_{ij}]_{m\times q}$ is initialized with $\mathbf{R}^{(0)} = \mathbf{\Phi} = [\phi_{ij}]_{m\times q}$ as follows:

$$\forall 1 \leqslant i \leqslant m,\ \forall 1 \leqslant j \leqslant q: \ \ \phi_{ij} = \begin{cases} 1, & \text{if } y_j^i = +1; \\ 0, & \text{if } y_j^i = -1. \end{cases} \tag{19}$$

Next, the label propagation procedure works by iteratively updating $\mathbf{R}$ as: $\mathbf{R}^{(t)} = \alpha\mathbf{P}\mathbf{R}^{(t-1)} + (1-\alpha)\mathbf{\Phi}$. In practice, $\mathbf{R}^{(t)}$ will converge to $\mathbf{R}^*$ as t grows to infinity [42,62,63]:

$$\mathbf{R}^* = (1-\alpha)(\mathbf{I}-\alpha\mathbf{P})^{-1}\mathbf{\Phi}. \tag{20}$$

Here, $\alpha \in (0, 1)$ is the trade-off parameter that balances the information flow from label propagation and initial labeling. Next, the implicit relative labeling-importance information $\mathcal{U}$ is obtained by normalizing each row of $\mathbf{R}^*$ as follows:

$$\forall 1 \leqslant i \leqslant m,\ \forall 1 \leqslant j \leqslant q: \ \ \mu_{\boldsymbol{x}_i}^{\lambda_j} = \frac{r_{ij}^*}{\sum_{j=1}^{q} r_{ij}^*}. \tag{21}$$

In the second stage, in order to make use of the information conveyed by $\mathcal{U}$, RELIAB chooses the maximum entropy model [64] to parametrize the multi-label predictor as follows:

$$f_j(\boldsymbol{x}) = \frac{1}{Z(\boldsymbol{x})}\exp\left(\boldsymbol{\theta}_j^{\top}\boldsymbol{x}\right) \quad (1 \leqslant j \leqslant q),$$
$$\text{where } Z(\boldsymbol{x}) = \sum\nolimits_{j=1}^{q} \exp\left(\boldsymbol{\theta}_j^{\top}\boldsymbol{x}\right). \tag{22}$$

In order to induce the prediction model $\mathbf{\Theta} = [\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \ldots, \boldsymbol{\theta}_q]$, RELIAB chooses to minimize the following objective function:

$$V(\mathbf{\Theta}, \mathcal{U}, \mathcal{D}) = V_{dis}(\mathbf{\Theta}, \mathcal{U}) + \beta \cdot V_{emp}(\mathbf{\Theta}, \mathcal{D}). \tag{23}$$

Here, the first term $V_{dis}(\mathbf{\Theta}, \mathcal{U})$ evaluates how well the prediction model $\mathbf{\Theta}$ fits the estimated relative labeling-importance information $\mathcal{U}$ (e.g., by Kullback-Leibler divergence) and the second term evaluates how well the prediction model $\mathbf{\Theta}$ classifies the training examples in $\mathcal{D}$ (e.g., by empirical ranking loss). Furthermore, β is the regularization parameter that balances the two terms of the objective function.

Given an unseen instance $\boldsymbol{x}^*$, its relevant label set Y^* is determined by thresholding the parametrized prediction model as follows:

$$Y^* = \{\lambda_j \mid f_j(\boldsymbol{x}^*) > t(\boldsymbol{x}^*),\ 1 \leqslant j \leqslant q\}. \tag{24}$$

Here, $t(\boldsymbol{x}^*)$ is a thresholding function, which can be learned from the training examples [1,34,42].

Algorithm 6 presents the pseudo-code of RELIAB. As shown in Algorithm 6, RELIAB employs a two-stage procedure to learn from multi-label examples, where the relative labeling-importance information estimated in the first stage contributes to the model induction in the second stage. Furthermore, the order of label correlations considered by RELIAB depends on the empirical loss chosen to instantiate $V_{emp}(\boldsymbol{\Theta}, \mathcal{D})$.

Algorithm 6 Pseudo-code of RELIAB [42]

```
Inputs:
  $\mathcal{D}$: Multi-label training set $\{(\boldsymbol{x}^i, \boldsymbol{y}^i) \mid 1 \leqslant i \leqslant m\}$
      $(\boldsymbol{x}^i \in \mathcal{X}, \boldsymbol{y}^i \in \{-1,+1\}^q, \mathcal{X} = \mathbb{R}^d, \mathcal{Y} = \{\lambda_1, \lambda_2, \ldots, \lambda_q\})$
  $\alpha$: Trade-off parameter in (0,1)
  $\beta$: Regularization parameter
  $\boldsymbol{x}^*$: Unseen instance $(\boldsymbol{x}^* \in \mathcal{X})$
Outputs:
  $Y^*$: Predicted label set for $\boldsymbol{x}^*$ $(Y^* \subseteq \mathcal{Y})$
Process:
  Construct the fully-connected graph $G = (V, E)$ with $V = \{\boldsymbol{x}^i \mid 1 \leqslant i \leqslant m\}$;
  Specify the weight matrix $\mathbf{W}$ using Eq. (17);
  Set the label propagation matrix $\mathbf{P}$ using Eq. (18);
  Initialize the labeling-importance matrix $\mathbf{R}$ using Eq. (19), and then derive the converged solution $\mathbf{R}^*$ using Eq. (20);
  Obtain the relative labeling-importance information $\mathcal{U}$ using Eq. (21);
  Learn the parametrized prediction model $\boldsymbol{\Theta}$ by minimizing the objective function specified in Eq. (23);
  Return $Y^* = \{\lambda_j \mid f_j(\boldsymbol{x}^*) > t(\boldsymbol{x}^*),\ 1 \leqslant j \leqslant q\}$ w.r.t. Eq. (24)
```

5 Conclusion

In this paper, the state of the art of binary relevance, which is one of the most important solutions for multi-label learning, was reviewed. Specifically, the basic settings for binary relevance, a few representative correlation-enabling extensions, and related issues on class-imbalance and relative labeling-importance have been discussed. Code packages for the learning algorithms introduced in this paper are publicly-available at the MULAN toobox [57] (binary relevance [9], classifier chain [12,14], stacking aggregation [19]) and the first author's homepage (LEAD [25], COCOA [38], RELIAB [42]).

For binary relevance, there are several research issues that require further investigation. First, the performance evaluation of multi-label learning is more complicated than single-label learning. A number of popular multi-label evaluation metrics have been proposed [1,2,10,11]. It is desirable to design correlation-enabling extensions for binary relevance that are tailored to optimize designated multi-label metrics, suitable for the multi-label learning task at hand. Second, in binary relevance, the same set of features is used to induce the classification models for all class labels. It is appropriate to develop binary relevance style learning algorithms that are capable of utilizing label-specific features to characterize distinct properties of each class label [65–67]. Third, the modeling complexities of binary relevance, as well as its extensions, are linear to the number of class labels in the label space. It is necessary to adapt binary relevance to accommodate extreme multi-label learning scenarios with huge (e.g., millions) numbers of class labels [68–72].

Acknowledgements The authors would like to thank the associate editor and anonymous reviewers for their helpful comments and suggestions. This work was supported by the National Natural Science Foundation of China (Grant Nos. 61573104, 61622203), the Natural Science Foundation of Jiangsu Province (BK20141340), the Fundamental Research Funds for the Central Universities (2242017K40140), and partially supported by the Collaborative Innovation Center of Novel Software Technology and Industrialization.

References

1. Zhang M-L, Zhou Z-H. A review on multi-label learning algorithms. IEEE Transactions on Knowledge and Data Engineering, 2014, 26(8): 1819–1837
2. Zhou Z-H, Zhang M-L. Multi-label learning. In: Sammut C, Webb G I, eds. Encyclopedia of Machine Learning and Data Mining. Berlin: Springer, 2016, 1–8
3. Schapire R E, Singer Y. Boostexter: a boosting-based system for text categorization. Machine Learning, 2000, 39(2–3): 135–168
4. Cabral R S, De la Torre F, Costeira J P, Bernardino A. Matrix completion for multi-label image classification. In: Proceedings of Advances in Neural Information Processing Systems. 2011, 190–198
5. Sanden C, Zhang J Z. Enhancing multi-label music genre classification through ensemble techniques. In: Proceedings of the 34th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. 2011, 705–714
6. Barutcuoglu Z, Schapire R E, Troyanskaya O G. Hierarchical multilabel prediction of gene function. Bioinformatics, 2006, 22(7): 830–836
7. Qi G-J, Hua X-S, Rui Y, Tang J, Mei T, Zhang H-J. Correlative multilabel video annotation. In: Proceedings of the 15th ACM International Conference on Multimedia. 2007, 17–26
8. Tang L, Rajan S, Narayanan V K. Large scale multi-label classification

via metalabeler. In: Proceedings of the 19th International Conference on World Wide Web. 2009, 211–220

9. Boutell M R, Luo J, Shen X, Brown C M. Learning multi-label scene classification. Pattern Recognition, 2004, 37(9): 1757–1771
10. Tsoumakas G, Katakis I, Vlahavas I. Mining multi-label data. In: Maimon O, Rokach L, eds. Data Mining and Knowledge Discovery Handbook. Berlin: Springer, 2010, 667–686
11. Gibaja E, Ventura S. A tutorial on multilabel learning. ACM Computing Surveys, 2015, 47(3): 52
12. Read J, Pfahringer B, Holmes G, Frank E. Classifier chains for multilabel classification. In: Proceedings of Joint European Conference on Machine Learning and Knowledge Discovery in Databases. 2009, 254–269
13. Dembczyński K, Cheng W, Hüllermeier E. Bayes optimal multilabel classification via probabilistic classifier chains. In: Proceedings of the 27th International Conference on Machine Learning. 2010, 279–286
14. Read J, Pfahringer B, Holmes G, Frank E. Classifier chains for multilabel classification. Machine Learning, 2011, 85(3): 333–359
15. Kumar A, Vembu S, Menon A K, Elkan C. Learning and inference in probabilistic classifier chains with beam search. In: Proceedings of Joint European Conference on Machine Learning and Knowledge Discovery in Databases. 2012, 665–680
16. Li N, Zhou Z-H. Selective ensemble of classifier chains. In: Proceedings of International Workshop on Multiple Classifier Systems. 2013, 146–156
17. Senge R, del Coz J J, Hüllermeier E. Rectifying classifier chains for multi-label classification. In: Proceedings of the 15th German Workshop on Learning, Knowledge, and Adaptation. 2013, 162–169
18. Mena D, Montañés E, Quevedo J R, del Coz J J. A family of admissible heuristics for A* to perform inference in probabilistic classifier chains. Machine Learning, 2017, 106(1): 143–169
19. Godbole S, Sarawagi S. Discriminative methods for multi-labeled classification. In: Proceedings of Pacific-Asia Conference on Knowledge Discovery and Data Mining. 20004, 22–30
20. Montañés E, Quevedo J R, del Coz J J. Aggregating independent and dependent models to learn multi-label classifiers. In: proceedings of Joint European Conference on Machine Learning and Knowledge Discovery in Databases. 2011, 484–500
21. Montañés E, Senge R, Barranquero J, Quevedo J R, del Coz J J, Hüllermeier E. Dependent binary relevance models for multi-label classification. Pattern Recognition, 2014, 47(3): 1494–1508
22. Tahir M A, Kittler J, Bouridane A. Multi-label classification using stacked spectral kernel discriminant analysis. Neurocomputing, 2016, 171: 127–137
23. Loza Mencía E, Janssen F. Learning rules for multi-label classification: a stacking and a separate-and-conquer approach. Machine Learning, 2016, 105(1): 77–126
24. Tsoumakas G, Dimou A, Spyromitros E, Mezaris V, Kompatsiaris I, Vlahavas I. 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, 101–116
25. Zhang M-L, Zhang K. Multi-label learning by exploiting label dependency. In: Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. 2010, 999–1007
26. Alessandro A, Corani G, Mauá D, Gabaglio S. An ensemble of Bayesian networks for multilabel classification. In: Proceedings of the 23rd International Joint Conference on Artificial Intelligence. 2013, 1220–1225
27. Sucar L E, Bielza C, Morales E F, Hernandez-Leal P, Zaragoza J H, Larrañaga P. Multi-label classification with Bayesian network-based chain classifiers. Pattern Recognition Letters, 2014, 41: 14–22
28. Li Y-K, Zhang M-L. Enhancing binary relevance for multi-label learning with controlled label correlations exploitation. In: Proceedings of Pacific Rim International Conference on Artificial Intelligence. 2014, 91–103
29. Alali A, Kubat M. Prudent: a pruned and confident stacking approach for multi-label classification. IEEE Transactions on Knowledge and Data Engineering, 2015, 27(9): 2480–2493
30. Petterson J, Caetano T. Reverse multi-label learning. In: Proceedings of the Neural Information Processing Systems Comference. 2010, 1912–1920
31. Spyromitros-Xioufis E, Spiliopoulou M, Tsoumakas G, Vlahavas I. Dealing with concept drift and class imbalance in multi-label stream classification. In: Proceedings of the 22nd International Joint Conference on Artificial Intelligence. 2011, 1583–1588
32. Tahir M A, Kittler J, Yan F. Inverse random under sampling for class imbalance problem and its application to multi-label classification. Pattern Recognition, 2012, 45(10): 3738–3750
33. Quevedo J R, Luaces O, Bahamonde A. Multilabel classifiers with a probabilistic thresholding strategy. Pattern Recognition, 2012, 45(2): 876–883
34. Pillai I, Fumera G, Roli F. Threshold optimisation for multi-label classifiers. Pattern Recognition, 2013, 46(7): 2055–2065
35. Dembczynski K, Jachnik A, Kotłowski W, Waegeman W, Hüllermeier E. Optimizing the F-measure in multi-label classification: plug-in rule approach versus structured loss minimization. In: Proceedings of the 30th International Conference on Machine Learning. 2013, 1130–1138
36. Charte F, Rivera A J, del Jesus M J, Herrera F. Addressing imbalance in multilabel classification: measures and random resampling algorithms. Neurocomputing, 2015, 163: 3–16
37. Charte F, Rivera A J, del Jesus M J, Herrera F. Mlsmote: approaching imbalanced multilabel learning through synthetic instance generation. Knowledge-Based Systems, 2015, 89: 385–397
38. Zhang M-L, Li Y-K, Liu X-Y. Towards class-imbalance aware multi-label learning. In: Proceedings of the 24th International Joint Conference on Artificial Intelligence. 2015, 4041–4047
39. Wu B, Lyu S, Ghanem B. Constrained submodular minimization for missing labels and class imbalance in multi-label learning. In: Proceedings of the 30th AAAI Conference on Artificial Intelligence. 2016, 2229–2236
40. Cheng W, Dembczynski K J, Hüllermeier E. Graded multilabel classification: the ordinal case. In: Proceedings of the 27th International Conference on Machine Learning. 2010, 223–230
41. Xu M, Li Y-F, Zhou Z-H. Multi-label learning with PRO loss. In: Proceedings of the 27th AAAI Conference on Artificial Intelligence. 2013, 998–1004
42. Li Y-K, Zhang M-L, Geng X. Leveraging implicit relative labeling-importance information for effective multi-label learning. In: Proceedings of the 15th IEEE International Conference on Data Mining. 2015, 251–260
43. Geng X, Yin C, Zhou Z-H. Facial age estimation by learning from label distributions. IEEE Transactions on Pattern Analysis and Machine

Intelligence, 2013, 35(10): 2401–2412

44. Geng X. Label distribution learning. IEEE Transactions on Knowledge and Data Engineering, 2016, 28(7): 1734–1748

45. Gao N, Huang S-J, Chen S. Multi-label active learning by model guided distribution matching. Frontiers of Computer Science, 2016, 10(5): 845–855

46. Dembczyński K, WaegemanW, Cheng W, Hüllermeier E. On label dependence and loss minimization in multi-label classification. Machine Learning, 2012, 88(1–2): 5–45

47. Gao W, Zhou Z-H. On the consistency of multi-label learning. In: Proceedings of the 24th Annual Conference on Learning Theory. 2011, 341–358

48. Sun Y-Y, Zhang Y, Zhou Z-H. Multi-label learning with weak label. In: Proceedings of the 24th AAAI Conference on Artificial Intelligence. 2010, 593–598

49. Xu M, Jin R, Zhou Z-H. Speedup matrix completion with side information: application to multi-label learning. In: Proceedings of the Neural Information Processing Systems Conference. 2013, 2301–2309

50. Cabral R, De la Torre F, Costeira J P, Bernardino A. Matrix completion for weakly-supervised multi-label image classification. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2015, 37(1): 121–135

51. Senge R, del Coz J J, Hüllermeier E. On the problem of error propagation in classifier chains for multi-label classification. In: Spiliopoulou M, Schmidt-Thieme L, Janning R, eds. Data Analysis, Machine Learning and Knowledge Discovery. Berlin: Springer, 2014, 163–170

52. Zhou Z-H. Ensemble Methods: Foundations and Algorithms. Boca Raton, FL: Chap-man & Hall/CRC, 2012

53. Koller D, Friedman N. Probabilistic Graphical Models: Principles and Techniques. Cambridge, MA: MIT Press, 2009

54. Koivisto M. Advances in exact Bayesian structure discovery in Bayesian networks. In: Proceedings of the 22nd Conference on Uncertainty in Artificial Intelligence. 2006, 241–248

55. Smith V, Yu J, Smulders T, Hartemink A, Jarvis E. Computational inference of neural information flow networks. PLoS Computational Biology, 2006, 2: 1436–1449

56. Murphy K. Software for graphical models: a review. ISBA Bulletin, 2007, 14(4): 13–15

57. Tsoumakas G, Spyromitros-Xioufis E, Vilcek J, Vlahavas I. MULAN: a Java library for multi-label learning. Journal of Machine Learning Research, 2011, 12: 2411–2414

58. He H, Garcia E A. Learning from imbalanced data. IEEE Transactions on Knowledge and Data Engineering, 2009, 21(9): 1263–1284

59. Wang S, Yao X. Multiclass imbalance problems: analysis and potential solutions. IEEE Transactions on Systems, Man, and Cybernetics-Part B: Cybernetics, 2012, 42(4): 1119–1130

60. Liu X-Y, Li Q-Q, Zhou Z-H. Learning imbalanced multi-class data with optimal dichotomy weights. In: Proceedings of the 13th IEEE International Conference on Data Mining. 2013, 478–487

61. Abdi L, Hashemi S. To combat multi-class imbalanced problems by means of over-sampling techniques. IEEE Transactions on Knowledge and Data Engineering, 2016, 28(1): 238–251

62. Zhou D, Bousquet O, Lal T N, Weston J, Schölkopf B. Learning with local and global consistency. In: Proceedings of the Neural Information Processing Systems Conference. 2004, 284–291

63. Zhu X, Goldberg A B. Introduction to semi-supervised learning. In: Brachman R, Stone P, eds. Synthesis Lectures to Artificial Intelligence and Machine Learning. San Francisco, CA: Morgan & Claypool Publishers, 2009, 1–130

64. Della Pietra S, Della Pietra V, Lafferty J. Inducing features of random fields. IEEE Transactions on Pattern Analysis and Machine Intelligence, 1997, 19(4): 380–393

65. Zhang M-L, Wu L. LIFT: multi-label learning with label-specific features. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2015, 37(1): 107–120

66. Xu X, Yang X, Yu H, Yu D-J, Yang J, Tsang E C C. Multi-label learning with label-specific feature reduction. Knowledge-Based Systems, 2016, 104: 52–61

67. Huang J, Li G, Huang Q, Wu X. Learning label-specific features and class-dependent labels for multi-label classification. IEEE Transactions on Knowledge and Data Engineering, 2016, 28(12): 3309–3323

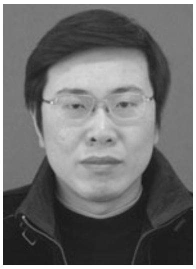
68. Weston J, Bengio S, Usunier N. WSABIE: scaling up to large vocabulary image annotation. In: Proceedings of the 22nd International Joint Conference on Artificial Intelligence. 2011, 2764–2770

69. Agrawal R, Gupta A, Prabhu Y, Varma M. Multi-label learning with millions of labels: recommending advertiser bid phrases for Web pages. In: Proceedings of the 22nd International Conference on World Wide Web. 2013, 13–24

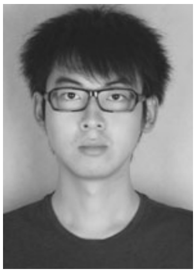
70. Xu C, Tao D, Xu C. Robust extreme multi-label learning. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. 2016, 1275–1284

71. Jain H, Prabhu Y, Varma M. Extreme multi-label loss functions for recommendation, tagging, ranking & other missing label applications. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. 2016, 935–944

72. Zhou W J, Yu Y, Zhang M-L. Binary linear compression for multi-label classification. In: Proceedings of the 26th International Joint Conference on Artificial Intelligence. 2017



Min-Ling Zhang received the BS, MS, and PhD degrees in computer science from Nanjing University, China in 2001, 2004 and 2007, respectively. Currently, he is a professor at the School of Computer Science and Engineering, Southeast University, China. In recent years, he has served as the Program Co-Chairs of ACML'17, CCFAI'17, PRICAI'16, Senior PC member or Area Chair of AAAI'18/'17, IJCAI'17/'15, ICDM'17/'16, PAKDD'16/'15, etc. He is also on the editorial board of Frontiers of Computer Science, ACM Transactions on Intelligent Systems and Technology, Neural Networks. He is the secretary-general of the CAAI (Chinese Association of Artificial Intelligence) Machine Learning Society, standing committee member of the CCF (China Computer Federation) Artificial Intelligence & Pattern Recognition Society. He is an awardee of the NSFC Excellent Young Scholars Program in 2012.



Yu-Kun Li received the BS and MS degrees in computer science from Southeast University, China in 2012 and 2015 respectively. Currently, he is an R&D engineer at the Baidu Inc. His main research interests include machine learning and data mining, especially in learning from multi-label data.

Xu-Ying Liu received the BS degree at Nanjing University of Aeronautics and Astronautics, China, the MS and PhD degrees at Nanjing University, China in 2006 and 2010 respectively. Now she is an assistant professor at the PALM Group, School of Computer Science and Engineering, Southeast University, China. Her research interests mainly include machine learning and data mining, especially cost-sensitive learning and class imbalance learning.

Xin Geng is currently a professor and the director of the PALM lab of Southeast University, China. He received the BS (2001) and MS (2004) degrees in computer science from Nanjing University, China, and the PhD (2008) degree in computer science from Deakin University, Australia. His research interests include pattern recognition, machine learning, and computer vision. He has published more than 50 refereed papers in these areas, including those published in prestigious journals and top international conferences.