Association Rules-Based Classifier Chains Method

DING JIAMAN, ZHOU SHUJIE, LI RUNXIN, FU XIAODONG, AND JIA LIANYIN

Faculty of Information Engineering and Automation, Kunming University of Science and Technology, Kunming 650500, China
Artificial Intelligence Key Laboratory of Yunnan Province, Kunming 650500, China
Corresponding author: Jia Lianyin (jlianyin@163.com)

This work was supported by the National Natural Science Foundation of China under Grant 61562054. The fund title is Research on Key Technologies of Set-Valued Indexes, and the fund principal named JIA LIANYIN.

ABSTRACT The order for label learning is very important to the classifier chains method, and improper order can limit learning performance and make the model very random. Therefore, this paper proposes a classifier chains method based on the association rules (ARECC in short). ARECC first designs strong association rules based label dependence measurement strategy by combining the idea of frequent patterns; then based on label dependence relationship, a directed acyclic graph is constructed to topologically sort all vertices in the graph; next, the linear topological sequence obtained is used as the learning order of labels to train each label’s classifier; finally, ARECC uses association rules to modify and update the probability of the prediction for each label. By mining the label dependencies, ARECC writes the correlation information in the topological sequence, which improves the utilization of the correlation information. Experimental results of a variety of public multi-label datasets show that ARECC can effectively improve classification performance.

INDEX TERMS Multi-label learning, classifier chains, label correlations, association rules.

I. INTRODUCTION

Traditional supervised learning usually assumes that each instance is associated with one label. In real-world applications, an instance usually has multiple labels. For example, a document about the Olympics may be associated with “sports”, “business” and “economic” at the same time. Traditional supervised learning based on one label per instance can no longer solve this problem, while multi-label learning processing instances associated with a set of labels has become a research hotspot in the field of machine learning and data mining, and is widely used in the fields of text classification, image annotation, and functional gene prediction [1], [2].

The common strategy for dealing with multi-label learning problems is to construct n independent binary classifiers and convert them into single-label classification problems. Such as BR (Binary Relevance), but this method ignores the implicit relationship between labels [3]. Rational use of the implicit relationship between labels can improve the performance of multi-label classification, which is also one of the hotspots of multi-label learning research in recent years. For this reason, Read et al. [4], [5] proposed a classifier chains method, arrange the labels in a chain in a certain order, train the classifier in turn for the labels on the chains, and add all the label learning results before the target label to the training features of the target label classifier. The method is simple and easy to implement and uses the implicit relationship between labels to achieve better prediction performance than the binary relevance method. However, the classifier chains method needs to pre-set the learning order for the labels during the learning process, and a good label learning order is often difficult to determine. In addition, improper label learning order will affect the predictive performance of the method, and may even transmit wrong information [1]. Read et al. proposed an improved ECC method (Ensemble Classifier Chains) using ensemble ideas. It randomly generates multiple different label learning orders, integrates the learning results of multiple orders, weakens the unstable influence caused by a single label sequence, and alleviates the problem of limited performance of the classifier chains method. As the number of labels increases, the number of permutations and combinations of labels order increases factorially, which greatly increases the randomness and uncertainty of the prediction of the ensemble classifier chains method. The ensemble classifier chains method still faces the difficulty of order selection [6]. On the other hand, limited by the classifier chains model structure, its utilization of the label correlations information is limited. When the feature dimension of the dataset is too high, the information transmitted
between labels is easy to be submerged in the features, which affects the performance and accuracy of the model.

In response to the above problems, this paper proposes a classifier chains method based on association rules (ARECC for short). The method first combines the idea of frequent patterns to design a calculation method of related information between labels based on association rules. Then, a directed acyclic graph is constructed through the pairwise dependency and the degree of dependency between the labels. Next, topologically sort the directed acyclic graph to get the sequence that implies the dependency between labels, and the obtained topological sequence is used as the learning order of the labels in the classifier chains method. Finally, the association rules are used to modify and update the prediction results of the classifier chains model. Experimental results on a variety of public multi-label datasets show that ARECC can effectively improve classification performance.

The organizational structure of the rest of this article: Section II introduces the related work of multi-label learning; Section III gives the principle and implementation process of the method in this article; Section IV reports the experimental results and comparative analysis; Section V concludes the full text.

II. RELATED WORD
The classifier chains algorithm [4], [5] is a simple and practical multi-label classification algorithm developed from the binary relevance algorithm [3]. The binary relevance algorithm converts the multi-label classification problem into several binary classification problems, but this algorithm ignores the correlation information between the labels, which leads to performance limitations. The classifier chains algorithm is developed from the binary relevance, by adding several labels into the feature space of labels waiting to be predicted, multiple independent binary classifiers are linked one by one so that the current classifier learns the previous label information which improves the performance of the classifier. It is a multi-label learning method that uses the “high-order strategy” of label correlation information [1]. In multi-label learning, reasonable use of the implicit correlation information between labels can bring more ideal prediction performance, which is also the research focus of multi-label learning in recent years. For example, Wang et al. combined deep belief network and backpropagation neural network to transfer information between labels [7]; Dai et al. proposed asymmetric uncertainty association between labels based on fuzzy mutual information [8]; Lin et al. constructed a projection matrix model which used to determine the relationship between multiple labels [9]. From the perspective of using the correlation information between labels, multi-label learning methods can be divided into three categories [1]:

The “first-order strategy” considers the independence of labels and ignores the coexistence of labels. For example, the binary relevance algorithm proposed by Boutell et al. [3], converts a multi-label classification problem into multiple independent binary classification problems. This method has a simple concept and high efficiency, but it ignores the implicit correlation information between labels and has limited learning performance.

The “second-order strategy” is to learn the pairwise correlation information between labels. For example, the Rank-SVM (support vector machines) algorithm proposed by Elisseeff and Weston [10], the CML (collective multi-label classification) algorithm proposed by Ghamrawi and McCallum [11], and a three-way selection ensemble model proposed by Zhang et al. [12]. The second-order strategy is simple and effective, and obtains good generalization performance, but the implicit correlation information between labels in practical applications often exceeds the second-order.

The “high-order strategy”, that is, to learn the implicit correlation information between multiple labels. High-order strategy methods describe the implicit correlation information between labels more accurately and have stronger capabilities of modeling. For example, Huang et al. proposed a new cost-sensitive LE (label embedding) algorithm [13]. In addition, the classifier chains method is a simple and effective algorithm that belongs to the “high-order strategy”.

The classifier chains algorithm effectively uses the associated information between labels to improve the accuracy of prediction, but its chain structure is sensitive to the order of labels, resulting in unstable performance and randomness. A large number of experimental results show that the choice of learning order will seriously affect the learning performance of the classifier chains algorithm [6], [14]. In response to this problem, Read et al. [4], [5] proposed the ensemble classifier chains, which weakens the randomness brought by a single chain by integrating multiple classifier chains with different label orders. However, the number of label orders increases in a factorial order with the number of labels, and the algorithm still faces strong randomness. For this reason, Wang and Li [15] proposed a classifier circle algorithm, which uses a model of circle structure to avoid the influence of learning order on the classifier chains; Li Na et al [16] proposed a classifier chains algorithm based on multi-label importance ranking, which uses the degree of interaction between the labels as the basis for measuring the importance of the labels, and uses the importance ranking result as the learning order of the classifier chains model; Hu Tianlei et al [17] proposed the two-way classifier chains solution, extracting the labels information again by introducing a reverse chain. What’s more, the classifier chains algorithm does not make full use of the correlation information between labels. Its chain structure determines that when label A learns the information of label B, label B can no longer learn the information of label A. When the feature dimension of the dataset is high, the correlation information between labels is easily concealed by excessively long redundant features. For this reason, Wei et al [18] proposed a classifier chains method based on label feature selection, which reduces the redundant information in labels transmission through feature selection. Liu et al [19] considered the imbalance problem
and proposed a classifier chains method based on random undersampling. In addition, Lu and Mineichi [20] optimized the performance of the classifier chains from the perspective of conditional likelihood maximization. For the learning order that the classifier chains rely on, most of these classifier chains related algorithms overcome the label order problem based on training labels iteratively, which still faces strong randomness.

To overcome the above difficulties, this paper proposes a classifier chains method based on association rules.

III. THE PROPOSED ALGORITHM

The method is divided into three steps. The first is the mining of the correlation information between labels. This paper proposes a new method for calculating the correlation information between labels, migrating the idea of strong association rules from the frequent pattern, analogy a group of multi-label vectors in the dataset to a transaction, exploring the association rules between label items.

The second step is to construct a directed acyclic graph using the association rules between the label 1-itemset as the pairwise dependency and the degree of dependency between the labels, and then obtain its topological sequence. The pairwise dependency relationship and the degree of dependency between labels are regarded as the constraint between labels, and a directed acyclic graph is constructed to represent the dependency between labels so that the label dependency information is implicit in the topological sequences. Then the topological sequences are used as the learning orders in the CC model (Classifier Chains).

Finally, the strong association rules between the label k-itemset are used to modify and update the prediction results, so that the model can learn the correlation information between the labels again, enhance the utilization of the information between the labels in the classifier chains model, and improve the performance of the model.

A. ASSOCIATION RULES BETWEEN LABELS

Frequent pattern mining searches for recurring connections in a given dataset. A typical example is shopping basket analysis. This process analyzes the customer’s shopping habits by discovering the correlation between the products that the customer puts in the “shopping basket”. If we imagine that the universe is a collection of commodities in a store, then each commodity has a boolean variable that indicates whether the commodity appears. Each shopping basket can be represented by a boolean vector. Boolean vectors can be analyzed to obtain purchase patterns reflecting the frequent associations or simultaneous appearance of commodities. These patterns can be expressed in the form of association rules [22].

Similarly, in the multi-label classification problem, all labels can be regarded as the entire domain, and the overall label situation of each sample can be represented by a boolean vector, and then by analyzing the boolean vector, we can obtain the pattern reflecting the frequent association or simultaneous occurrence of the labels.

Let \( \gamma = \{l_1, l_2, \ldots, l_y\} \) be the set of labels, and \( D_Y \) is the set of label vectors of the dataset, where each label vector \( Y \) is a set of non-empty label items, such that \( Y \subseteq \gamma \). Let \( A \) be a set of label items, and the label vector \( Y \) contains \( A \), if and only if \( A \subseteq Y \). The association rule is an implication of the form \( A \Rightarrow B \), where \( A \subseteq \gamma, B \subseteq \gamma, A \neq \emptyset, B \neq \emptyset, \) and \( A \cap B \neq \emptyset \). Rule \( A \Rightarrow B \) is established in dataset \( D_Y \) and has a support degree of support, where support is the percentage of the label vector in dataset \( D_Y \) that contains \( A \cup B \) (that is, the union of sets \( A \) and \( B \) or both \( A \) and \( B \)). It is probability \( P(A \cup B) \). Rule \( A \Rightarrow B \) has a confidence level confidence in dataset \( D_Y \), where confidence is the percentage of the label vector that contains \( A \) and also contains \( B \) in the dataset \( D_Y \). This is conditional probability \( P(B|A) \), which is

\[
\text{support}(A \Rightarrow B) = P(A \cup B) \\
\text{confidence}(A \Rightarrow B) = P(B|A)
\]

The rule that meets the minimum support threshold \( \text{min_sup} \) and the minimum confidence threshold \( \text{min_conf} \) at the same time is strong [22]. Here we call the collection of labels a label itemset. A label itemset containing \( k \) labels is called a label \( k \)-itemset. The occurrence frequency of a label itemset is the number of label vectors containing the label itemset in the dataset \( D_Y \), referred to as the frequency, support count, or count of the label itemset.

From the formula (2), there is

\[
\text{confidence}(A \Rightarrow B) = \frac{P(A \cup B)}{P(A)} = \frac{\text{support}(A \cup B)}{\text{support}(A)} \\
\text{support}(A \cup B) = \text{support_count}(A \cup B) \\
\text{support}(A) = \text{support_count}(A)
\]

where \( \text{support_count}(A \cup B) \) is the number of label vectors containing label itemset \( A \cup B \), and \( \text{support_count}(A) \) is the number of label vectors containing label itemset \( A \).

However, the association rules derived from the confidence level are deceptive in special circumstances, and sometimes interesting association rules [22]. Therefore, we introduce the lift in the association rules as the identification of strong associations. Lift is a simple correlation measure, \( A \) appears independent of \( B \), if \( P(A \cup B) = P(A)P(B) \); otherwise, \( A \) and \( B \) are dependent and correlated [22]. The lift between the appearance of \( A \) and \( B \) can be obtained by the following formula

\[
\text{lift}(A, B) = \frac{P(A \cup B)}{P(A)P(B)} = \frac{P(B|A)}{P(B)} = \frac{\text{confidence}(A \Rightarrow B)}{P(B)}
\]

Calculate the lift between label itemsets by the occurrence frequency of label itemsets. If the lift is greater than 1, then label itemset \( A \) and label itemset \( B \) are positively correlated, which means that the appearance of each label itemset \( A(B) \) may imply the appearance of label itemset \( B(A) \); the lift is less than 1, then the appearance of label itemset \( A \) and the appearance of label itemset \( B \) is negatively correlated, which
means that the appearance of each label itemset \( A(B) \) may cause label itemset \( B(A) \) not to appear.

1) \textbf{PAIRED DEPENDENCY BETWEEN LABELS}

First, count the occurrence frequencies of all labels (\( L = \{l_1, l_2, \ldots, l_n\} \)) and pairwise combinations between labels in the set \( D_Y = \{Y_i|1 \leq i \leq n\} \) of the label vector of the dataset, and obtain the label 1-itemset frequency \( s_1 \) and the label 2-itemset frequency \( s_2 \). Defined as:

\[
\begin{align*}
    s_1^i &= \text{support}_\text{count}(l_i) = \sum_{j=1}^{n} \begin{cases} 
        1 & y_j = l_i \\
        0 & \text{other} \end{cases} \\
    s_2^i,j &= \text{support}_\text{count}(l_i \cup l_j) = \sum_{k=1}^{n} \begin{cases} 
        1 & y_k^i = 1 \text{ and } y_k^j = 1 \\
        0 & \text{other} \end{cases}
\end{align*}
\]

Then the confidence of the association between the label 1-itemset is calculated by the formula (3), and the lift of the association between the label 1-itemset is calculated by the formula (4), which is defined as:

\[
\begin{align*}
    \text{confidence}(l_i \Rightarrow l_j) &= \frac{\text{support}_\text{count}(l_i \cup l_j)}{\text{support}_\text{count}(l_i)} = \frac{s_2^i,j}{s_1^i} \quad (7) \\
    \text{lift}(l_i, l_j) &= \frac{\text{confidence}(l_i \Rightarrow l_j)}{\text{support}(l_j)} = \frac{\text{support}_\text{count}(l_i \cup l_j)/\text{support}_\text{count}(l_i)}{\text{support}(l_j)} = \frac{s_2^i,j \times n}{s_1^j} \quad (8)
\end{align*}
\]

where \( n \) is the number of samples, a lift greater than 1 indicates that label \( l_i \) and label \( l_j \) are positively correlated, and the positive correlation between the two is directly proportional to the lift value. At this time, the dependence between the two is considered to be significantly effective. Set the confidence threshold \( \text{min}_\text{conf} \), the association rules smaller than the confidence threshold is weak association rules, and the confidence value is set to 0. Algorithm 1 gives the calculation process of the dependence between labels.

\textbf{Algorithm 1 Calculate the Label Dependency Matrix}

\textbf{Input}: the confidence threshold \( \text{min}_\text{conf} \), label vector set of training dataset \( D_Y = \{Y_i|1 \leq i \leq n\} \)

\textbf{Output}: label dependency matrix \( W \in \mathbb{R}^{q \times q} \)

1. Initialize the space of \( s_1 \in \mathbb{O}^q, s_2 \in \mathbb{O}^{q \times q}, W \in \mathbb{R}^{q \times q} \)

2. for \( i \leftarrow 1 \) to \( q \)

3. for \( j \leftarrow 1 \) to \( n \)

4. if \( Y_i^j = 1 \) then \( s_1^i \leftarrow s_1^i + 1 \)

5. for \( k \leftarrow 1 \) to \( q \)

6. if \( Y_j^k = 1 \) and \( Y_j^k = 1 \) then \( s_2^i,j \leftarrow s_2^i,j + 1 \)

7. end for

8. end for

9. for \( i \leftarrow 1 \) to \( q \)

10. for \( j \leftarrow 1 \) to \( q \)

11. if \( s_2^i,j/s_1^i > \text{min}_\text{conf} \) and \( s_1^i > \text{min}_\text{conf} \)

12. \( w_{ij} \leftarrow s_2^i,j/s_1^i \)

13. end for

14. end for

2) \textbf{ASSOCIATION RULES}

The formula(3) indicates that the confidence of rule \( A \Rightarrow B \) is easily deduced from the calculation of support of \( A \) and \( A \cup B \), and the minimum confidence threshold can be used to determine whether the rule is a strong rule. Therefore, mining the association rules of label itemsets can be divided into two steps:

(a) Find all frequent label itemsets: Find the set of label itemsets whose occurrences are greater than or equal to the minimum support threshold \( \text{min}_\text{sup} \).

(b) Generate strong association rules from frequent itemsets: Find the rules that meet the minimum support and minimum confidence among frequent itemsets.

By calculating the support count of the label itemsets, the association rules between the label itemsets are calculated, and the rule that meets the minimum support and the minimum confidence threshold is the strong association rule. The specific algorithm can use Apriori, FP-Growth, and other association rule mining algorithms to get the association rules set \( R = \{r_i|1 \leq i \leq p\} \) between labels, and the association rules are in the form of \( \{y_{q_1}, y_{q_2}, \ldots, y_{q_m}\} \Rightarrow \{y_{y_1}, y_{y_2}, \ldots, y_{y_n}\} \).

\textbf{B. DAG AND TOPOLOGICAL SEQUENCE}

A directed acyclic graph is usually an effective tool to describe the progress of a project or system. Except for the simplest cases, almost all projects can be divided into several active sub-projects, and these sub-projects are usually restricted by certain conditions. For example, the start of some sub-projects must be completed after the completion of other sub-projects, the DAG diagram is often used to
represent the driving dependency between events and manage the scheduling between tasks.

Similar to the constraint dependency relationship between sub-projects, the dependency relationship between labels can also be regarded as the constraint condition between labels. For example, label $A$ depends on label $B$ and label $C$, which means that the start of label $A$ prediction needs to be completed after the prediction of label $B$ and label $C$ is completed. Use the obtained dataset label dependency to draw a directed graph of dependency between labels.

Topological sorting of a directed acyclic graph (DAG) is to arrange all nodes in the graph into a linear sequence, so that any pair of nodes $u$ and $v$ in the graph, if there is a directed edge from $u$ to $v$, then $u$ appears before $v$ in the sequence. Generally, such a linear sequence is called a sequence that satisfies the topological order, referred to as a topological sequence. The topological sequence is often used to determine the sequence of events in a dependency set. By topological sorting on a directed acyclic graph (DAG), a topological sequence is obtained. The dependency relationship between labels can be implicit in the sequence.

Based on the dependency matrix between labels, the label relationship directed acyclic graph can be constructed. To ensure that the constructed graph is a directed graph acyclic, Compare the two dependency values of mutually dependent labels (that is, both wacyclic, Compare the two dependency values of mutually dependent labels (that is, both wacyclic, Compare the two dependency values of mutually dependent labels (that is, both wacyclic, $w_{ij}$ and $w_{ji}$ are greater than 0), and set the weaker dependency value to 0 to eliminate the loop caused by the interdependence between labels. Although the ring formed by the interdependence between labels no longer exists, the ring caused by the interdependence of three or more labels may still appear. Therefore, it is necessary to carry out ring detection and adjust the value of min_conf to eliminate the ring structure in the figure. The ring detection process is as follows:
(a) Traverse all current nodes and calculate the in-degree of each node (the sum of the number of times that a point in the directed graph is the endpoint of the edge in the graph).
(b) Count the number of nodes with an in-degree of 0, if not, the graph has a ring structure, and the detection ends.
(c) Delete the nodes with an in-degree of 0 and the associations of these nodes to other nodes. If the number of remaining nodes after deletion is 0, the graph has no loop structure and the detection ends; otherwise, skip to step a.

Finally, according to the directed acyclic graph structure of the label dependency, the topological sequence of the label dependency can be obtained. Due to the characteristics of directed acyclic graphs, often more than one topological sequence is obtained, so this paper sets the maximum topological sequence number threshold $\text{max}_\text{tplen}$. If the number of sequences is greater than the maximum topological sequence number threshold $\text{max}_\text{tplen}$, then $k$ sequences are randomly retained. In the end, $k$ topological sequences are retained, and a topological sequences set $T$ is obtained, which is defined as follows:

$$T = \{t_1, t_2, \cdots, t_k\}, 1 \leq k \leq \text{max}_\text{tplen}$$

Among them, $t_i$ is the sequence combination of the $L = \{l_1, l_2, \cdots, l_q\}$ labels set ($1 \leq i \leq \text{max}_\text{tplen}$). Algorithm 2 presents the process of generating the complete sequences of topological sorting from the label dependence matrix $W$.

**Algorithm 2 Get the Full Topological Sequences**

**Input**: label dependency matrix $W \in \mathbb{R}^{q \times q}$, threshold of the number of topological sequences $\text{max}_\text{tplen}$

**Output**: full sequences of topological sorting $T = \{t_1, t_2, \cdots, t_k\}, 1 \leq k \leq \text{max}_\text{tplen}$

1. Initialize the space of sequence $T$, current sequence preList
2. topologicalSort $(W, \text{preList}, T)$
3. if length (preList) = $q$ then $T \leftarrow T \cup$ preList
4. if length $(T) = \text{max}_\text{tplen}$ then end
5. for $i \leftarrow 1$ to $q$
6. if sum $(w_i) = 0$ and $i$ not in preList
7. preList $\leftarrow$ preList $\cup$ i
8. for $j \leftarrow 1$ to $q$
9. $w_{j,i} \leftarrow 0$
10. end for
11. topologicalSort $(W, \text{preList}, T)$
12. end for

**C. MODIFY AND UPDATE STRATEGY**

Use the mined label itemsets association rules to modify and update the results of the classifier chains prediction, and learn the association information between labels again to improve the model’s utilization of the association information between labels.

For a label association rule in the form of $\{y_{l_1}, y_{l_2}, \cdots, y_{l_n}\} \Rightarrow \{y_{q_1}, y_{q_2}, \cdots, y_{q_m}\}$, when its confidence meets the minimum support and minimum confidence thresholds, the rule is a strong association rule. The label $n$-itemset can be used to correct the label probability of the label $m$-itemset. The predicted probability of each label of the label $n$-itemset and the predicted probability of the label $m$-itemset to be corrected are added and the average value is obtained to obtain the new corrected probability. The formula is as follows

$$P(y_{q_i})^\prime = \frac{\sum_{j=1}^{n} P(y_{q_j}) + P(y_{q_i})}{n + 1} \quad (10)$$

When the newly revised probability $P(y_{q_i})^\prime$ is greater than the originally predicted probability $P(y_{q_i})$, the newly revised probability $P(y_{q_i})^\prime$ replaces the originally predicted probability $P(y_{q_i})$, otherwise, the original predicted probability is maintained.

Use the mined association rules between label itemset to modify and update the obtained prediction result $y' = \{(Y'_i) | 1 \leq i \leq n'\}$, improving the model’s utilization of the correlation information between labels. Algorithm 3 gives the process of correction and completion.
TABLE 1. Experimental datasets

| Datasets  | Field     | Number of examples | Feature dimension | Number of labels | Label cardinality | Label density | Label distinct |
|-----------|-----------|--------------------|-------------------|-----------------|-------------------|--------------|---------------|
| birds     | music     | 645                | 260               | 19              | 1.014             | 0.053        | 133           |
| emotions  | music     | 593                | 72                | 6               | 1.869             | 0.311        | 27            |
| flags     | image     | 194                | 19                | 7               | 3.392             | 0.485        | 54            |
| yeast     | biology   | 2417               | 103               | 14              | 4.237             | 0.303        | 198           |
| medical   | text      | 978                | 1449              | 45              | 1.245             | 0.028        | 94            |
| enron     | text      | 1702               | 1001              | 53              | 3.378             | 0.064        | 753           |
| genbase   | biology   | 662                | 1186              | 27              | 1.252             | 0.046        | 32            |
| bookmarks | text      | 87856              | 2150              | 208             | 2.028             | 0.010        | 18716         |

Algorithm 3 Process of the Modify and Update Strategy

Input: predict result \( P(y_q) = \{P(y_{qj}) | 1 \leq j \leq n\}, 1 \leq i \leq q \), the association rules set \( R = \{r_i|1 \leq i \leq p\} \)

Output: newly prediction result \( P(y_{qi}'), 1 \leq i \leq q \)

1. Initialize the new prediction result space \( P(y_{qi})' \leftarrow P(y_{qi}) \)
2. for \( i \rightarrow 1 \) to \( p \)
3. for \( j \rightarrow 1 \) to \( m \)
4. \( P(y_{qi})' \leftarrow \left[ \sum_{k=1}^{n} P(y_{ki}) + P(y_{qi})' \right] / n + 1 \)
5. if \( P(y_{qi})' < P(y_{qi}) \) then \( P(y_{qi}) \leftarrow P(y_{qi})' \)
6. end for
7. end for

IV. EXPERIMENTS AND RESULTS

A. DATASETS

In this paper, eight multi-label datasets are tested to explore the performance of the algorithm. These datasets come from different application fields of multi-label: flags from image classification, birds, emotions from music annotation, yeast, genbase from biological gene function prediction, and medical, bookmarks, enron from text classification.

These datasets can be downloaded from the home page (http://mlkd.csd.auth.gr/multilabel.html) of the open-source project Mulan [23]. Detailed statistics of the datasets are given in TABLE 1.

Label cardinality, the average number of relevant labels per sample:

\[
LCard(D) = \frac{1}{m} \sum_{i=1}^{m} \sum_{j=1}^{q} y_{ij} \quad (11)
\]

Label density, the ratio of label cardinality to the number of labels:

\[
LDen(D) = \frac{1}{q} LCard(D) \quad (12)
\]

Label distinct, total number of sample related label sets:

\[
LDis(D) = | \{ y | \exists x : (x, y) \in D \} | \quad (13)
\]

B. COMPARATIVE ALGORITHM

The five algorithms compared in this paper including BR, CC, ECC, CCE, and ML-kNN.

1. BR (binary relevant) method, which does not consider the relationship between labels and independently trains the binary classifier of each label [3].
2. CC (classifier chains) method, classifier chains algorithm [4, 5].
3. ECC (ensemble classifier chains) method, ensemble classifier chains method [4, 5].
4. CCE (classifier circle) method is an improved classifier circle algorithm based on the classifier chains algorithm, which uses circle structure to avoid the uncertainty of label sequence [15].
5. ML-kNN (multi-label k-nearest neighbor) method, which extends the k-nearest neighbor method to deal with multi-label learning problems [24].
6. ARECC (association rules based classifier chains method) method is the classifier chains algorithm based on association rules in this paper.

C. EVALUATION METRICS

Multi-label classification problem has many evaluation metrics, and different evaluation metrics measure different aspects of performance. In this paper, six commonly used indicators hammingloss, macro-F1, micro-F1, rankingloss, oneerror and coverage [1] in the multi-label learning field are used to measure the prediction performance of the method. The definitions are as follows:

1. hammingloss: it measures the proportion of wrong labels, the proportion of correct labels that are not predicted, and the proportion of wrong labels that are predicted.
2. macro-F1: average of each label F1 indicator.
3. micro-F1: calculate the overall accuracy (micro-P) and recall (micro-R) of all samples on all markers, and then calculate the F1 score. The methods used in this paper are three classifier chains correlation algorithms and two other multi-label algorithms.
4. rankingloss: The ranking loss evaluates the fraction of reversely ordered label pairs, i.e. an irrelevant label is ranked higher than a relevant label.
5. oneerror: The one-error evaluates the fraction of examples whose top-ranked label is not in the relevant label set.
6. Coverage: The coverage evaluates how many steps are needed, on average, to move down the ranked label list so as to cover all the relevant labels of the example.
### D. EXPERIMENTAL RESULTS

During the experiment, for each dataset, this paper randomly selects 70% of the examples as the training samples and the remaining 30% as the test samples. To reduce the influence of randomness, this paper repeats the experiment 30 times for each method of each dataset and calculates the average value and standard deviation of each metrics as the final result of the experiment.

All experiments in this paper are completed on the python platform. The binary classifiers used in the classifier chains include AdaBoost, SVM (support vector machines). The base classifiers and evaluation metrics (hammingloss, macro-F1, micro-F1, rankingloss, oneerror, coverage) are implemented by the scikit-learn toolkit on the python platform.

Table 2, Table 3 shows the average results of hammingloss, macro-F1, micro-F1, rankingloss, oneerror, and coverage on eight datasets compared with BR, CC, ECC, CCE, and ML-kNN (the classifier chains related algorithm are based on AdaBoost classifier). Among them, the bold font indicates that the algorithm obtains the optimal value under the dataset and the corresponding metrics (paired-samples t-test at 95% confidence level).

↑ (↓) indicates that the larger (smaller) the evaluation metrics, the better the performance of the corresponding method. And the number in brackets represents the ranking of the method under this indicator in the current dataset.

By analyzing the data in the table, it can be seen that ARECC has achieved good performance in most cases. Among them, the performance on the datasets of birds, emotions, flags, medical is particularly outstanding, with the best performance in 4, 5, 4, and 3 of the 6 evaluation metrics. The reason is that the association degree between labels is high and the label dimension is low. ARECC algorithm makes full use of the dependencies hidden in the label sequence. In the yeast dataset, the ML-kNN algorithm that uses the similarity between samples to propagate labels achieves the best of the two metrics among six. Similarly, the BR algorithm that does not consider the correlation information between labels also achieves the best of the two metrics and the second-best of the two metrics. The reason is that the label dimension of the yeast dataset is high, the number of permutations and combinations in the label order is large, the classifier chains related algorithm that relies on the label sequence is highly random, and the excessively long-chain structure enhances the transmission of error information, leading to the performance is limited. The ARECC algorithm achieves better performance than other classifier chains algorithms. The average ranking of the six metrics is 2.67, compared with ECC 4.17, CC 5.17, and CCE 2.83. The reason is that ARECC’s modify and update strategy further utilizes the correlation information between labels based on the classifier chains, and is not affected by error transmission. In the remaining dataset, ARECC achieves the best average ranking on enron and bookmarks, and the second-best average ranking on genbase. The above experimental results reflect the overall performance of the ARECC method in this paper.

Table 4 and Table 5 show the experimental results under 60% of the training samples. It can be seen that the ARECC method also achieves better performance than ECC and other multi-label learning algorithms on most datasets. In general, the ARECC method achieves the best average ranking in 5 datasets, and the second-best ranking in the remaining three datasets.
datasets. Figure 1 and Figure 2 show the average ranking of each compared algorithm on eight datasets and six evaluation metrics. This result verifies the effectiveness of the method in this paper and applies to different numbers of training samples.

1) ALGORITHM STABILITY EXPERIMENT

When the label dimension of the multi-label dataset is too high, the classifier chains and its related algorithm faces the exponentially increasing label sequence, which is prone to unstable performance, resulting in strong prediction randomness. The ARECC method selects the learning order by mining the correlation information between the labels, which is different from the classifier chains and its related algorithm that randomly selects the learning order, so it can have more stable performance. This article compares the performance of ARECC, CC, and ECC methods on the medical, enron, and bookmarks dataset (label dimensions 45, 53, and 208). The

### TABLE 3. Average performance for ARECC method and compared methods on medical, enron, genbase, bookmarks.

| Datasets | Algorithm | Hammingloss | Macro-F1 | Micro-F1 | Rankingloss | Oneerror | Coverage |
|----------|-----------|-------------|----------|----------|-------------|----------|----------|
| Medical  | ARECC     | 0.0115±0.0006 | 0.3425±0.0121 | 0.7770±0.0148 | 0.0024±0.0004 | 0.2217±0.0168 | 5.891±4.192 |
|          | BR        | 0.0199±0.0044 | 0.3244±0.0149 | 0.7702±0.0183 | 0.0021±0.0003 | 0.2341±0.0173 | 5.608±5.110 |
|          | ECC       | 0.014±0.0011  | 0.3317±0.0163 | 0.7765±0.0207 | 0.0023±0.0002 | 0.2170±0.0179 | 5.875±4.277 |
|          | CC        | 0.017±0.0038  | 0.3259±0.0130 | 0.7760±0.0151 | 0.0024±0.0002 | 0.2201±0.0221 | 5.897±4.982 |
|          | CCE       | 0.017±0.0077  | 0.3318±0.0118 | 0.7725±0.0162 | 0.0022±0.0004 | 0.2232±0.0196 | 5.914±4.513 |
|          | ML-KNN    | 0.0196±0.0010 | 0.3288±0.0144 | 0.7593±0.0187 | 0.0032±0.0012 | 0.4999±0.0184 | 4.457±5.041 |

### TABLE 4. Average performance for ARECC method and compared methods on birds, emotions, flags, yeast(60% as training samples).

| Datasets | Algorithm | Hammingloss | Macro-F1 | Micro-F1 | Rankingloss | Oneerror | Coverage |
|----------|-----------|-------------|----------|----------|-------------|----------|----------|
| Birds    | ARECC     | 0.0502±0.0099 | 0.0595±0.0024 | 0.433±0.0141 | 0.0043±0.0006 | 0.3851±0.0197 | 3.888±6.229 |
|          | BR        | 0.0511±0.0122 | 0.0522±0.0034 | 0.406±0.0157 | 0.0040±0.0011 | 0.3868±0.0207 | 3.858±6.024 |
|          | ECC       | 0.0508±0.0143 | 0.0574±0.0021 | 0.412±0.0135 | 0.0042±0.0007 | 0.3958±0.0205 | 3.892±6.193 |
|          | CC        | 0.0507±0.0100 | 0.0574±0.0028 | 0.419±0.0149 | 0.0045±0.0008 | 0.3872±0.0212 | 3.879±6.488 |
|          | CCE       | 0.0507±0.0008 | 0.0572±0.0023 | 0.420±0.0137 | 0.0044±0.0008 | 0.3862±0.0243 | 3.848±6.220 |
|          | ML-KNN    | 0.0514±0.0014 | 0.0887±0.0037 | 0.493±0.0152 | 0.0115±0.0030 | 0.4974±0.0234 | 4.167±5.791 |

B. Jiaman et al.: Association Rules-Based Classifier Chains Method
TABLE 5. Average performance for ARECC method and compared methods on medical, enron, genbase, bookmarks (60% as training samples).

| Datasets       | Algorithm | HammingLoss  | Macro-F1    | Micro-F1    | RankingLoss | Oneerror | Coverage |
|----------------|-----------|--------------|-------------|-------------|-------------|----------|----------|
| Medical        | ARECC     | 0.110±.0068  | 0.341±.0140 | 0.784±.0110 | 0.0027±.0003 | 0.2175±.0135 | 6.166±36.62 |
|                | BR        | 0.117±.0069  | 0.329±.0302 | 0.775±.0136 | 0.0023±.0002 | 0.2314±.0164 | 6.465±57.09 |
|                | ECC       | 0.115±.0070  | 0.336±.0171 | 0.781±.0149 | 0.0026±.0003 | 0.2172±.0227 | 6.356±54.77 |
|                | CC        | 0.112±.0069  | 0.342±.0193 | 0.791±.0125 | 0.0025±.0003 | 0.2131±.0121 | 6.177±60.41 |
|                | CCE       | 0.113±.0082  | 0.340±.0186 | 0.782±.0147 | 0.0025±.0002 | 0.2193±.0167 | 6.170±62.29 |
|                | ML-KNN    | 0.117±.0080  | 0.372±.0173 | 0.607±.0129 | 0.0585±.0001 | 0.4418±.0183 | 3.644±43.98 |

The experimental results show that when faced with multi-label datasets with high label dimensions, the excessively high label dimension weakens the effect of chains dissemination of related information, and the prediction accuracy and performance of the classifier chains and its related algorithm are roughly the same. However, the ARECC method’s active selection of learning order rather than random selection strategy makes it have the best stability. These experimental results prove that the ARECC method has better stability than other classifier chain related methods such as ECC when facing datasets with high label dimensions.

2) EFFECTIVENESS EXPERIMENT

To analyze the effectiveness of the modify and update strategy in ARECC, this paper introduces three algorithm variants for ARECC, ECC, and BR: ARECC-N, ECC-MU and BR-MU. ‘-MU’ means to adopt the modify and update strategy, while ‘-N’ to not adopt the strategy (ARECC adopts). Refer to chapter III section C for the modify and update strategy. The experiment is the same as the above experiment setting. Table 6 shows the experiment results of the ARECC, ECC, and BR algorithms with and without the modify and update strategy.

Experiments show that compared with ECC, ECC-MU, BR, and BR-MU algorithms, ARECC has achieved relatively better performance in most cases. Specifically, on the birds, emotions, and flags datasets, ARECC scored 3, 3, and 4 best in the six evaluation metrics respectively; on the yeast dataset, ARECC achieved an average ranking of 2.67, compared to ARECC-NMU (3.5), ECC-MU (4.17), ECC (3.83), BR-MU (3.33), BR (3). In general, the average ranking of these six
TABLE 6. Average performance for ARECC method and compared methods with and without ‘mu’ strategy on birds, emotions, flags, yeast.

| Datasets  | Algorithm | Hammingloss | Macro-F1 | Micro-F1 | Rankingloss | Oneerror | Coverage |
|-----------|-----------|-------------|----------|----------|-------------|----------|----------|
|          |           |             |          |          |             |          |          |
|          |           | 0.039±0.0042 | 0.334±0.0322 | 0.508±0.0303 | 0.009±0.0025 | 0.522±0.0383 | 4.559±2.985 |
|          | ARECC-NMU | 0.039±0.0047 | 0.332±0.0337 | 0.506±0.0355 | 0.010±0.0029 | 0.522±0.0383 | 4.465±3.522 |
|          | ECC-MU    | 0.0402±0.0373 | 0.328±0.0417 | 0.509±0.0338 | 0.011±0.0033 | 0.520±0.0458 | 4.581±4.055 |
|          | ECC       | 0.0397±0.0372 | 0.325±0.0420 | 0.499±0.0346 | 0.009±0.0019 | 0.520±0.0458 | 4.385±4.561 |
|          | BR-MU     | 0.0404±0.0173 | 0.321±0.0332 | 0.510±0.0338 | 0.011±0.0054 | 0.517±0.0840 | 4.503±1.822 |
|          | BR        | 0.0401±0.0023 | 0.319±0.0339 | 0.494±0.0371 | 0.010±0.0054 | 0.517±0.0840 | 4.509±1.242 |

FIGURE 3. The average rank of the variance on medical, enron, bookmarks.

FIGURE 4. The average rank of the variance on six evaluation metrics.

algorithms on the four datasets based on the six evaluation metrics is ARECC>EARECC-NMU>ECC-MU>ECC>BR-MU>BR. Figure 5 and Figure 6 show the average rankings of the six algorithms on four datasets and six evaluation metrics. This result verifies the effectiveness of the “MU” strategy in the ARECC algorithm.

3) TIME COMPLEXITY

The ARECC method is composed of the generation of the label learning order, model training, and prediction. The time complexity of the mining of association rules, the calculation of topological sequence, and the modify and update strategy is relatively small. Therefore, the time complexity of the ARECC method mainly depends on the training and prediction of the model, and the time complexity is $k \cdot \sum_{i=1}^{n} O_{\beta}(d, d + i - 1) \text{ and } k \cdot \sum_{i=1}^{n} O'_{\beta}(d + i - 1)$ respectively, where $O_{\beta}(\cdot, \cdot)$ and $O'_{\beta}(\cdot)$ are the time complexity of binary classifier training and prediction, while $k$ is the number of
integrations of the classifier chains. When the number of integrations of the classifier chains of ARECC and ECC algorithms is the same, they have the same time complexity. Therefore, the ARECC method has the same time complexity as the ECC algorithm, but experiments show that the ARECC method has achieved better performance and stability.

In order to analyze the actual operation efficiency of ARECC method and each comparative algorithm, this paper counts the running time of each method on different datasets under the same experimental platform, and calculates the average time of five independent runs of each method, as shown in Table 7.

It can be seen from Table 7 that the running time of ARECC is roughly the same as that of ECC and other algorithms. Since CC and BR do not need to ensemble multiple chains (ECC, CCE and ARECC ensemble five chains in this experiment), the time is always less than the other three classifier chains related algorithms. The actual running time of ARECC is comparable to that of ECC, and when faced with high label dimension datasets, thanks to the “MU” strategy, the running time of ARECC is shorter than that of ECC. The above experiments show that ARECC achieves better performance with comparable running time to ECC.

V. CONCLUSION

The classifier chains method is sensitive to the label learning order, and different orders often lead to different classification performance. In response to this problem, the ARECC method used the topological sequence of labels dependencies to record the correlation information between labels, which improves the utilization of the correlation information and reduces the randomness of the model. On the other hand, the classifier chains model is limited by its chain structure. When faced with datasets with high label dimensions, the effectiveness of information transfer between labels is reduced, and the information between labels can only be learned in one direction. The ARECC method improved the classification performance of the model by the modify and update strategy, using the correlation information between labels recorded by the association rules, so that each label can learn from each other. The experimental results show that the ARECC method has achieved better performance than the ensemble classifier chains and its related algorithms. In future work, we will consider how to adapt and optimize association rules mining parameters to further improve model prediction performance.

REFERENCES

[1] M.-L. Zhang and Z.-H. Zhou, “A review on multi-label learning algorithms,” IEEE Trans. Knowl. Data Eng., vol. 26, no. 8, pp. 1819–1837, Aug. 2014, doi: 10.1109/TKDE.2013.39.
[2] G. Tsoumakas and I. Katakis, “Multi-label classification: An overview,” Int. J. Data Warehousing Mining, vol. 3, no. 3, pp. 1–13, 2006, doi: 10.1007/s8070101.
[3] M. R. Boutell, J. Luo, X. Shen, and C. M. Brown, “Learning multi-label scene classification,” Pattern Recognit., vol. 37, no. 9, pp. 1757–1771, Sep. 2004, doi: 10.1016/j.patcog.2004.03.009.
[4] J. Read, B. Pfahringer, G. Holmes, and E. Frank, “Classifier chains for multi-label classification,” in Machine Learning and Knowledge Discovery in Databases (Lecture Notes in Artificial Intelligence), vol. 5782, 2009, pp. 254–269, doi: 10.1007/978-3-642-01474-7_17.
[5] J. Read, B. Pfahringer, G. Holmes, and E. Frank, “Classifier chains for multi-label classification,” J. Mach. Learn. Res., vol. 8, no. 3, pp. 333–359, Dec. 2011.
[6] K. Dembczyński, W. Cheng, and E. Hüllermeier, “Bayes optimal multi-label classification via probabilistic classifier chains,” in Proc. 27th Int. Conf. Mach. Learn. (ICML), 2010, pp. 279–286.
[7] T. Wang, L. Liu, N. Liu, H. Zhang, L. Zhang, and S. Feng, “A multi-label text classification method via dynamic semantic representation model and deep neural network,” Int. J. Speech Technol., vol. 50, no. 8, pp. 2339–2351, Aug. 2020, doi: 10.1007/s10498-020-01680-w.
[8] J. Dai, J. Chen, Y. Liu, and H. Hu, “Novel multi-label feature selection via label symmetric uncertainty correlation learning and feature redundancy evaluation,” Knowl.-Based Syst., vol. 207, Nov. 2020, Art. no. 106342, doi: 10.1016/j.knosys.2020.106342.
[9] L. Lin, B. Liu, X. Zheng, Y. Xiao, Z. Liu, and H. Cai, “An efficient multi-label learning method with label projection,” Knowl.-Based Syst., vol. 207, Nov. 2020, Art. no. 106298, doi: 10.1016/j.knosys.2020.106298.
[10] A. Elissaeff and J. Weston, “A kernel method for multi-labeled classification,” in Proc. 14th Int. Conf. Neural Inf. Process. Syst., Natural Synth. (NIPS), Cambridge, MA, USA: MIT Press, Jan. 2001, pp. 681–687.
[11] N. Ghamrawi and A. McCallum, “Collective multi-label classification,” in Proc. 14th ACM Int. Conf. Inf. Knowl. Manage. (CIKM), New York, NY, USA: Association for Computing Machinery, 2005, pp. 195–200, doi: 10.1145/1099554.1099591.
[12] Y. Zhang, D. Miao, Z. Zhang, J. Xu, and S. Luo, “A three-way selective ensemble model for multi-label classification,” Int. J. Approx. Reason., vol. 103, pp. 394–413, Dec. 2018, doi: 10.1016/j.ijar.2018.10.009.
[13] K.-H. Huang and H.-T. Lin, “Cost-sensitive label embedding for multi-label classification,” Mach. Learn., vol. 106, no. 9, pp. 1725–1746, 2017.
[14] G. Madjarov, D. Koccev, D. Gjorgjevikj, and S. Džeroski, “An extensive experimental comparison of methods for multi-label learning,” Pattern Recognit., vol. 45, no. 9, pp. 3084–3104, Sep. 2012, doi: 10.1016/j.patcog.2012.03.004.
[15] S.-B. Wang and Y.-F. Li, “Classifier circle method for multi-label learning,” J. Softw., vol. 26, pp. 2811–2819, Nov. 2015, doi: 10.13238/j.cnki.jetp.2016060011.
T.-L. Hu, H.-B. Wang, and W.-D. Yin, “Multi-label news classification algorithm based on deep bi-directional classifier chains,” J. Zhejiang Univ., Eng. Ed., vol. 53, no. 11, pp. 2110–2117, Jan. 2019, doi: 10.3785/j.issn.1008-973X.2019.11.008.

W. Weng, D.-H. Wang, C.-L. Chen, J. Wen, and S.-X. Wu, “Label specific features-based classifier chains for multi-label classification,” IEEE Access, vol. 8, pp. 51265–51275, 2020, doi: 10.1109/ACCESS.2020.2980551.

B. Liu and G. Tsoumakas, “Dealing with class imbalance in classifier chains via random undersampling,” Knowl.-Based Syst., vol. 192, Mar. 2020, Art. no. 105292, doi: 10.1016/j.knosys.2019.105292.

L. Sun and M. Kudo, “Optimization of classifier chains via conditional likelihood maximization,” Pattern Recognit., vol. 74, pp. 503–517, Feb. 2018, doi: 10.1016/j.patcog.2017.09.034.

B. Wu, F. Jia, W. Liu, B. Ghanem, and S. Lyu, “Multi-label learning with missing labels using mixed dependency graphs,” Int. J. Comput. Vis., vol. 126, no. 8, pp. 875–896, Aug. 2018, doi: 10.1007/s11263-018-1085-3.

J. Han, M. Kamber, and J. Pei, “Data Mining: Concepts and Techniques,” 3rd ed., San Mateo, CA, USA: Morgan Kaufmann, 2012, pp. 157–162.

G. Tsoumakas, E. Spyromitros-Xioufis, J. Vilcek, and I. Vlahavas, “MULAN: A Java library for multi-label learning,” J. Mach. Learn. Res., vol. 12, pp. 2411–2414, Jul. 2011.

M.-L. Zhang and Z.-H. Zhou, “ML-KNN: A lazy learning approach to multi-label learning,” Pattern Recognit., vol. 40, no. 7, pp. 2038–2048, Jul. 2007, doi: 10.1016/j.patcog.2006.12.019.

ZHUO SHUJIE was born in 1996. He is currently pursuing the M.S. degree. His current research interests include data mining and machine learning.

LI RUNXIN was born in 1983. He is currently pursuing the Ph.D. degree. His current research interests include data mining and optimization algorithm.

FU XIAODONG was born in 1975. He is currently pursuing the Ph.D. degree. His current research interests include service computing, intelligent decision making, big data, and software engineering.

JIA LIANYIN was born in 1978. He is currently pursuing the Ph.D. degree. He is also an Associate Professor. His current research interests include database, data mining, information retrieval, and parallel computing.

DING JIAMAN was born in 1974. He received the B.S. degree from the Faculty of Information Engineering and Automation, Kunming University of Science and Technology, in 1998, and the M.S. degree, in 2005. He is currently a Professor. His current research interests include data mining, cloud computing, and machine learning.

ZHOU SHUJIE was born in 1996. He is currently pursuing the M.S. degree. His current research interests include data mining and machine learning.

LI RUNXIN was born in 1983. He is currently pursuing the Ph.D. degree. His current research interests include data mining and optimization algorithm.

FU XIAODONG was born in 1975. He is currently pursuing the Ph.D. degree. His current research interests include service computing, intelligent decision making, big data, and software engineering.

JIA LIANYIN was born in 1978. He is currently pursuing the Ph.D. degree. He is also an Associate Professor. His current research interests include database, data mining, information retrieval, and parallel computing.