Algorithm for Double-Layer Structure Multi-label Classification with Optimal Sequence Based on Attention Mechanism

Geqiao Liu[1,2(✉)] and Mingjie Tan[1,2]

[1] College of Engineering and Technology, Sichuan Radio and Television University, Chengdu, China
liugq@scrtvu.net

[2] Research Center for Educational Information Management and Information Systems, Open University of China, Beijing, China

Abstract. A common approach to multi-label classification is to perform problem transformation, whereby a multi-label problem is transformed into one or more single-label problems. Problem transformation considers label correlations by extending the attributes, but ignores the importance of each feature attribute for different classification targets, weakens the sensitivity of the classifier, and reduces the classification accuracy. Attention mechanism is a model that simulates the mechanism of human brain attention. It mainly emphasizes the influence of some crucial inputs on the output by calculating the attention probability distribution, which has a good optimization effect on the traditional model. Based on this, this paper proposes a two-layer chain structure multi-label classification (ATDCC-OS) algorithm, which incorporates the attention mechanism. This algorithm constructs a two-layer multi-label classification model in order to realize the correlation between labels through inter-layer and intra-layer interaction. At the same time, the attention mechanism is introduced to focus selectively on the sample features, identify more important information for the current task, and further improve the classification performance of the algorithm. Furthermore, an optimal sequence selection algorithm (OSS) is proposed, seeking to label the pecking order, solving the problem of reduced classification accuracy caused by randomly selecting the class label sequence to train the binary classifier by the chain classification model. The OSS will be used to optimize the second-layer chain classification model of ATDCC-OS. Comparisons on seven benchmark data sets with related algorithms verify the effectiveness of ATDCC-OS.

Keywords: Multi-label classification · Double layer · Attention mechanism

1 Introduction

Classification is a popular branch of data mining techniques, it is aimed at training sample data set to construct a classification model, and use classification model to the measured

H. Song and D. Jiang (Eds.): SIMUtools 2020, LNICST 369, pp. 370–390, 2021.
https://doi.org/10.1007/978-3-030-72792-5_31

data to predict the category information. In traditional single label classification, a sample instance belongs to only one category. However, There are a lot of ambiguities examples in the real world, namely a sample instance might also belong to two different categories, the corresponding classification problems referred to as multi-label classification. Initially, multi-label learning originates from the document classified in the ambiguity problem. After decades of development, multi-label classification technology has been widely applied to medical diagnosis, recommender systems, information retrieval, image, video and other fields [1–8]. In recent years, the frequency of "multi-label" discussions have continued to increase from international conferences related to machine learning such as ACL, NIPS, CIKM, COLING, AAAI, INTERSPEECH, ICML, KDD, ICDM, and IJCAI. These make multi-label classification a popular research direction in machine learning, and it has also attracted the attention of the authoritative publication "Machine Learning" in the international machine learning community. Therefore, a large number of multi-label classification algorithms have been proposed.

The existing multi-label learning algorithms are mainly divided into two categories: problem transformation and algorithm adaptation. Problem transformation will transform a multi-label problem into one or more single-label problems. In this way, single-label classifiers are employed; and their single label predictions are transformed into multi-label predictions [9]. Algorithm adaptation is to modify an algorithm (such as AdaBoost, decision trees) directly to make multi-label predictions. This article will focus on the research of problem transformation.

After studying the problem transformation algorithm, it is found that many algorithms (such as BR [10], CC [9], MBR [11], and DLMC-OS [12]) often neglect the correlation between labels, and randomly select label sequences and redundant interactive label information in the design process, which results in the reduction of classification accuracy. At the same time, problem transformation mainly considers the correlation between labels by extending attributes, but ignores the importance of each feature attribute for different classification targets, thus weakening the sensitivity of classifiers and affecting the classification effect of each classifier. Attention Mechanism [13] is a model that simulates the human brain's attention mechanism. It highlights the effect of some key inputs on the output by calculating the probability distribution of attention, which has a better optimization effect on the traditional model. Based on this, this paper proposes a multi-label classification algorithm based on attention mechanism (ATDCC-OS), which combines attention mechanism and double-layer chain structure.

The ATDCC-OS algorithm integrates three classical problem transformation types of multi-label classification frameworks (BR, CC, and MBR), and constructs a multi-label classification model based on a double-layer chain structure. In the first layer of the model, a binary association classification framework is used to accomplish the first classification of unseen instances, and the label information interacts with the second layer. In the second layer of the model, a chain classification model with an "update process" is used to accomplish the final classification of unseen instances. On this basis, attention mechanism is introduced to calculate dynamically the weight of each feature attribute in the second layer, which can identify more important attributes for the current classification task and further improve the classification performance of the algorithm. In order to solve the random chain order problem of the chain model in ATDCC-OS,

an optimal sequence selection algorithm (OSS) is proposed. The OSS combines mutual information, PageRank, Kruskal's algorithm, and hierarchical traversal algorithm to find a tag priority order, and uses this sequence to guide the construction of classifiers in the chain classification model and further optimize the ATDCC-OS.

The rest of the paper is organized as follows: Sect. 2describes related work, Sect. 3 describes the proposed ATDCC-OS model, Sect. 4 introduces experimental data, evaluation methods, experimental set up, and experimental results and discussion. This paper is concluded in Sect. 5.

2 Related Work

2.1 Multi-label Classification

At present, problem transformation and algorithm adaptation are two popular research directions in the field of multi-label classification. In the problem transformation method, the multi-label classification task transforms one or more single-label classification, regression, or sorting tasks [11]. The basic classification algorithms often used in problem transformation are supporting vector machine [14], naive Bayesian, and k-nearest neighbor algorithm. The algorithm adaptations often modify the single-label classification algorithm to adapt to multi-label classification data. Representative algorithms include ML-RBF [15, 16], ML-kNN [17, 18], Rank-SVM [19], associated classification algorithm LRwAR [20, 21], etc.

The most common problem transformation method is BR [10], which converts a multi-label problem into multiple binary problems, training a binary model for each label to determine whether it belongs to the class label. The classification prediction for a new instance will be the sum of all binary classifier results. Although the computational complexity of the BR algorithm is low and linearly related to the class label, the correlation between the class labels is not considered in the process of classification, which leads to a certain degree of information loss. S. Godbole et al. [11] proposed the MBR of the two-layer BR model. It considers label correlations by adding the output of the first layer as the sample attribute to the second layer. However, when performing the second-layer model training, there is a problem of redundant consideration of the label value.

J. Read et al. [9] introduced the "chain" to consider the correlation between labels, and proposed CC. It links all binary classifiers into a random chain, and the output of the previous classifier is added to its sample attributes as input to the next classifier. However, the random chain still has some disadvantages. First, when training the binary classifier, the CC only considers the output predictions of the previous classifiers, and ignores the back-end classifier, so that the correlation between labels cannot be fully considered. Secondly, the implicit directionality in the chain will also affect the classification accuracy. If there is a low-precision binary classifier at the head of the chain, the low-precision classification results will propagate backward along the chain, thus affecting the accuracy of the overall classification. Finally, CC is a random chain, and the randomness will increase with the number of labels, which will seriously affect the stability of the algorithm [22, 23].

G. Q. Liu and T. Guo [12] proposed a multi-label classification with optimal sequence based on double layers (DLMC-OS). The algorithm constructs a two-layer classification model. Each classifier in the second layer receives all the extended features transmitted from the first layer, and links the backward update to consider the correlations existing between labels. Although the algorithm effectively solves the randomness problem of the chain classification model, it does not solve the uniqueness of the sequence.

2.2 Attention Mechanism

The attention mechanism [24–26] was first applied to the field of computer vision to simulate human visual attention mechanism. It quickly scans the global image to obtain the target area that needs attention, and then puts more attention resources into the area to obtain more detailed information and suppress other useless information. This improves the efficiency and accuracy of visual information processing exceptionally. Attention mechanism in deep learning is essentially similar to human selective visual attention mechanism, and the core goal is to select more critical information for the current task goal from a large amount of information. D. Bahdanau et al. [27] applied attention mechanism to machine translation tasks for the first time and obtained good results. The Google Mind [28] team was inspired by the human attention mechanism and published an article in 2014 that introduced people not looking at the pixels of the entire image at once when viewing images. Instead, they focus on a specific part of the image according to their needs. Moreover, humans will obtain a position that needs to be focused on in the future based on the observation of the previous image. A. M. Rush et al. [29] proposed a completely data-driven abstract sentence summarization method based on the local attention model. H. Chen et al. [30] captured key components at different semantic levels by introducing user-product attention mechanism for emotional analysis. Z. Yang et al. [31] proposed a hierarchical attention network for document classification, enabling it to differentiate between unimportant content. R. He et al. [32] used attention mechanism to extract viewpoint entities from text. W. Yin et al. [33] combined the attention mechanism with CNN for machine translation, which is an early exploratory work of the attention mechanism in CNN. In "Attention is all you need" published by the Google Machine Translation Team [34], the self-attention mechanism is widely used to learn text representation. It is not only separated from traditional RNN/CNN, but also uses a novel multi-head mechanism.

3 ATDCC-OS

3.1 Preliminaries

Let $\chi \in \Re^d$ be the input domain of all possible d-dimensional attribute values. The set $\gamma = \{y_1, y_2, \nu, y_L\}$ is the output domain of L-dimensional label values. Each instance x is associated with a subset of these labels. This set is represented by a L-vector $Y = \left[y_1, y_2, \nu, y_L\right]$, where $y_j = 1$ if and only if label j is associated with instance x, and 0 otherwise. Given a multi-label training set $D = \{(x_i, Y_i)|1 \leq i \leq m\}$, where $x_i \in \chi$ is the d-dimensional attribute vector $(x_{i1}, x_{i2}, \nu, x_{id})^T$ and $Y_i \subset \gamma$ is a set of labels

corresponding to x_i. Learning multi-label classification is learning a multi-label classifier $H : \chi \rightarrow 2^{\gamma}.H^f = \left(H_1^f, H_2^f, \nu, H_L^f\right)$ and $H^s = \left(H_1^s, H_2^s, \nu, H_L^s\right)$ are the classifiers constructed in the first and second layer, respectively. $c^f = \left(y_1^f, y_2^f, \nu, y_L^f\right)$ and $c^S = \left(y_1^S, y_2^S, \nu, y_L^S\right)$ are the classification results of the corresponding layer.

3.2 The ATDCC Framework

According to the design idea of the DLMC-OS algorithm, a Double-Layer Chain Classification Model Based on Attention Mechanism (ATDCC) is constructed. ATDCC sets two layers to decompose the multi-label classification problem into independent binary-classification problems. In the first layer, the ATDCC model involves L binary transformations one for each label and each binary model is trained to predict the relevance of one of the labels [12]. The first layer of ATDCC implements the first classification of the instance, and then the classification result is transmitted to the second layer by extending the attribute. In the second layer, ATDCC builds a chained classification model with an update process, which uses the chain to pass and update label information, achieves the second interaction of label information, and simultaneously accomplishes the final classification of the instance. ATDCC fully considers the correlation between labels through inter-layer label information interaction and intra-layer label information transfer.

ATDCC$^{\text{First-layer}}$. ATDCC$^{\text{First-layer}}$ inherits the idea of the binary relevance classification model to construct a corresponding binary classifier for each label, and then combines the classification results of all binary classifiers as the first classification of unseen instances (see Fig. 1).

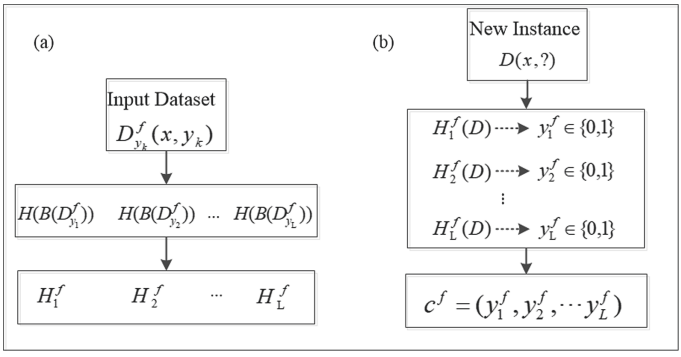

Fig. 1. a The procedure of ATDCC$^{\text{First-layer}}$ in training stage. **b** The procedure of ATDCC$^{\text{First-layer}}$ in training stage.

In the first step, for a dataset with L labels, ATDCC$^{\text{First-layer}}$ constructs a corresponding data set for each class label according to Eq. (1):

$$D_{y_k}^f = \{(x_i, y_k) | 1 \leq i \leq m\}$$

$$\text{Where} \quad y_k = \begin{cases} 1, \text{ if } y_k \in Y_k \\ 0, \text{ otherwise} \end{cases} \tag{1}$$

In the second step, for any multi-label training example, some binary algorithm B (such as SMO) is used to induce a binary classifier: $H^f_{y_k} \leftarrow B(D^f_{y_k})$.

In the third step, the unseen instance X is classified and predicted using the constructed binary classifier.

$$\begin{aligned} &H^f_{y_k} : X \rightarrow \{0, 1\} \\ &y^f_k = \left\{ H^f_{y_k}(X) | 1 \le k \le L \right\} \end{aligned} \tag{2}$$

Finally, ATDCC will combine the prediction value (i.e.$c^f = (y^f_1, y^f_2, \cdots, y^f_L)$) of each classifier as the first classification of the unseen instance, and add c^f to the original sample attribute to form a new sample attribute $x' = \{(x_i, c^f) | 1 \le i \le m\}$.$x'$ will be taken as the input of the next layer of ATDCC.

ATDCC$^{\text{AT-layer}}$. The attention mechanism has been successfully applied to sequence-to-sequence learning tasks. For classification tasks, the attention mechanism is able to learn the weight of each attribute in the sample to reflect its impact on the final classification result.

ATDCC$^{\text{AT-layer}}$ introduces the attention mechanism to calculate dynamically the weight value of the attribute values passed to the second layer model, discriminates the more important information for the current classification task for each classifier in the second layer, and enhances the sensitivity of each classifier.

In the first step, according to the dimension of the incoming data of the first layer, a weight matrix W is defined. ATDCC$^{\text{AT-layer}}$ will use the tanh function to quantize the correlation between the input data and the i-th label. In Eq. (3), W is the weight matrix that needs to be learned, and b is the bias of the model.

$$e_{ij} = \tanh(W_{ij} x^{'}_{ij} + b)(1 \le i \le m) \tag{3}$$

In the second step, ATDCC$^{\text{AT-layer}}$ will convert the result of Eq. (3) into a probability value by the softmax function to obtain attention weights:

$$W^{'}_{ij} = \text{softmax}(e_{ij}) = \frac{\exp(e_{ij})}{\sum_{j=1}^{m} \exp(e_{ij})} \tag{4}$$

Finally, the original input will be weighted using the attention weights from Eq. (4):

$$x^{''} = \sum\nolimits_{i=1}^{m} x^{'}_{ij} \omega^{'}_{ij} \tag{5}$$

We update and optimize the parameters in the model by minimizing the loss function. The loss function used here is the cross-entropy loss function (Eq. 6), which is the negative log likelihood of the actual and predicted labels for each sample:

$$J(\theta) = -\frac{1}{l} \sum_{k=1}^{l} \log p(y_k | y'_k) \tag{6}$$

ATDCC$^{\text{Second-layer}}$. ATDCC$^{\text{Second-layer}}$ (see Fig. 2) is the second layer of the ATDCC model, which uses the chained classification model with the update process to complete the second classification of the instances. The instance with attribute space for each binary model is extended with label relevance of all previous classifiers to form a classifier chain. The attribute space for each binary model is augmented with the 0/1 label prediction coming from first-layer classifiers and all prior binary relevance predictions from the second layer in which the classifier chain is built. The correlation between each label is fully considered after the procedure of the second layer. Each classifier in the chain is responsible for learning and predicting the binary association of the label, given the attribute space.

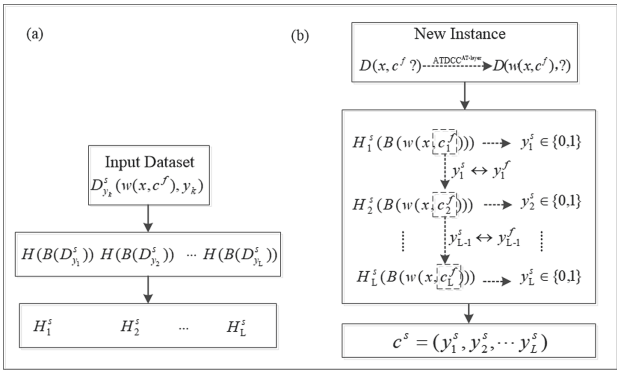

Fig. 2. a The procedure of ATDCC$^{\text{Second-layer}}$ in training stage. **b** The procedure of ATDCC$^{\text{Second-layer}}$ in training stage.

In the first step, ATDCC$^{\text{Second-layer}}$ constructs a corresponding data set $D^s_{y_k}(1 \le k \le L)$ for each class label according to Eq. (7), where W is the attribute weight obtained from the ATDCC$^{\text{AT-layer}}$.

$$D^s_{y_k} = \left\{ \left(\left[w_i x_i, w_{i+1} y^f_1, \cdots, w_{i+k-1} y^f_{k-1}, w_{i+k+1} y^f_{k+1} \cdots, w_{i+L} y^f_L \right], y^f_k \right) | 1 \le i \le m \right\} \tag{7}$$

In the second step, a binary algorithm B (such as SMO)is used to train the corresponding binary classifier on the constructed data set, i.e. $H^s_{y_k} \leftarrow B(D^s_{y_k})$.

In the third step, the unseen instance X is classified and predicted using the constructed binary classifier.

$$\begin{aligned} H^s_{y_k} &: X \to \{0, 1\} \\ y^s_k &= \left\{ H^s_{y_k}(X) | 1 \le k \le L \right\} \end{aligned} \tag{8}$$

During the classification process, the latest predicted label value is used to update the corresponding label value of the sample attribute space. For example, for the third classifier $H^s_{y_3}$ in the chain, the incoming sample attribute value is $\left[x, y^s_1, y^s_2, y^f_3, y^f_4, \cdots, y^f_L\right]$ instead of $\left[x, y^f_1, y^f_2, y^f_3, y^f_4, \cdots, y^f_L\right]$.

Finally, ATDCC will combine the classification prediction value $c^s = (y_1^s, y_2^s, \cdots, y_L^s)$ of each classifier as the final classification for the unseen instance.

3.3 OSS

Because the chain classification model has a random chain order problem, if there is a lower-precision binary classifier at the front of the chain, the low-precision classification result will propagate backward along the chain, affecting the classification accuracy of the subsequent classifier. It even affects the classification accuracy of the entire chain, and the randomness of the chain orders increases with the number of labels. Although the DLMC-OS algorithm alleviates the randomness problem of the chain classification model, it cannot find the optimal label sequence because it cannot fix the root node. Then the most straightforward solution is to sequence the random chain order, but the formation of the label sequence is not a simple arrangement, and needs to be combined with the characteristics of the chain classification model. To this end, the labeling order sought in this article needs to have the following points:

A. The label sequence is an ordered sequence containing all label information.
B. The label sequence is a sequence with the greatest correlation.
C. The label sequence is an optimal sequence.

Based on the above design principles, OSS is proposed. The algorithm combines mutual information, PageRank, Kruskal's algorithm, and hierarchical traversal algorithm to find a label with the most relevance. The sequence is used to guide the construction order for each classifier in the chain classification model, and the second layer of ATDCC is optimized using OSS.

Related Sub-algorithm of OSS. (1) *Mutual Information (MI).* In probability theory and information theory, the mutual information (MI) of two random variables is a measure of the mutual dependence between the two variables. More specifically, it quantifies the "amount of information" obtained about one random variable by observing the other random variable. The MI of the two variables is defined by Eq. (9). With the constant research of experts and scholars, the theory of MI has gradually infiltrated all walks of life. In machine learning, MI is applied to feature selection [35, 36]. In search engine technology, MI between phrases and contexts is used to discover semantic clusters [37]. In statistical mechanics, MI is combined with Loschmidt's paradox to solve mechanical problems [38, 39].
Based on the MI design idea and application, this work will measure the correlation between tags by calculating the MI between the two labels, and use it as the weight of the edges of the full connection graph.

(2) *PageRank.* PageRank (PR) [40] was proposed by Sergey Brin and Larry Page in 1998 to solve the problem of page ranking in link analysis. The core idea is that the importance of a page depends on the number and quality of other pages pointing to it. Reference [41] mentioned that Google's search engine uses the PageRank algorithm based on the importance (popularity) of Web pages. The importance of

a Webpage is discovered through the analysis of its link structure, and does not depend on the specific search request. Personalized PageRank is used by Twitter to present users with other accounts they may wish to follow [42].
Based on PageRank's design idea and priority search principle, this work will use the PageRank algorithm to find an important tag as a chain head node, which solves the problem of low-precision classifier in the chain.

Algorithm 1. Description of PageRank algorithm

Input: M, alpha, M: label matrix, alpha: damping factor (default value 0.85)
Output: root

```
V ← {}
N ← M. shape [1]
V ← random Matrix (n, 1)
Last_V ← Matrix (n, 1, 10)
M_hat ← (alpha * M) + (((1 - alpha) / N) * Matrix. ones ((N, N)))
While (Last_V -V > epsilon)
        Last_V ← V
        V ← M_hat * V
end while
root ← MaxIndex(V)
Return root
```

(3) *Edge-Weight Graph Algorithm.* A weighted graph [43] is a graph in which a number (the weight) is assigned to each edge. Such weights might represent, for example, costs, lengths, or capacities, depending on the problem at hand [44–47]. This work will use Edge-weight graph algorithm to construct a fully connected graph with the association with labels.

Algorithm 2. Description of Edge-weight graph algorithm

Input: $Y = \{y_1, y_2, \ldots, y_L\}$
Output: G

```
G ← {}
G. V ← Y
For each (u, v) in G.V:
    Calculate the mutual information of MI (u, v) according to definition 1.
    G. E ← MI (u, v)
G ← G(V, E)
Return G
```

(4) *Kruskal's algorithm.* Kruskal's algorithm [48] was proposed by Joseph Kruskal in 1956 to find the minimum spanning tree. This paper will introduce Kruskal's algorithm to find the largest-label spanning tree, which provides the basis of finding the sequence of the largest association with labels.

Algorithm 3. Description of Kruskal's algorithm

```
Input: G (V, E)
Output: MWT
1. MWT ← {}
2. For each v ∈ G.V:
3.     MAKE-SET(v)
4. For each (u, v) in G.E ordered by weight (u, v) increasing:
5.     If FIND-SET(u) ≠ FIND-SET(v):
6.         MST=MST ⋃ {(u, v)}
7.         UNION (u, v)
8. Return MWT ← MST
```

(5) *Breadth-First Search Algorithm.* Breadth-first search (BFS) and its application for finding connected components of graphs was invented in 1945 by Konrad Zuse. BFS is an algorithm for traversing or searching tree or graph data structures. In this paper, the BFS algorithm will take the tag node obtained by PageRank as the starting point, and traverse the largest-tag spanning tree to obtain the final tag order.

Algorithm 4. Description of Breadth-first search algorithm

```
Input: MWT (V, E)
Output: OS
1. Queue Q ← {}
2. For each v ∈ MWT.V:
3.   Q ← Q ⋃ (v)
4.   while(Q!=∅)
5.       v ← Q.head , w ← Q.next
6.       while(w!=∅)
7.           Q ← Q ⋃ (w)
8.       end while
9.   end while
10. end for
11. OS ← Q
12. Return OS
```

The OSS Framework. The design steps for the OSS algorithm are as follows (see Fig. 3):

In the first step, this work uses mutual information to measure the correlation between tags. If there are N labels $y_1, y_2 \cdots, y_n$, the mutual information on any two tags y_i and y_j is calculated by formula (9).

Definition 1: MI of two variables:

$$I(x_i, x_j) = \sum_{x_i x_j} p(x_i, x_j) \log\left(\frac{p(x_i, x_j)}{p(x_i)p(x_j)}\right) \text{ and must be non - negative.} \tag{9}$$

In the second step, a full-connection graph G representing the association with labels is constructed, the label is used as the vertex of the graph, and the mutual information value between the labels is used as the weight of the edge. In order to enable the

Kruskal algorithm to obtain the maximum weight label tree, it is necessary to reverse the mutual information value. In this way, the minimum-weight spanning tree obtained by the Kruskal algorithm is the maximum-weight spanning tree.

In the third step, the PageRank algorithm is used to perform "voting" sorting through each label in the data set to find the label node with the highest PR value. The selected node will be taken as the root node of the maximum weight tree and the starting node of the hierarchical traversal algorithm, which can solve the problem of reducing the overall classification accuracy caused by the existence of a low-precision classifier at the head of the label chain.

In the fourth step, the Kruskal's algorithm is used to generate a minimum-weight label tree (also MWT) for the full connection graph G. The label tree contains all the labels and edges connecting the label nodes, and the sum of weights is the largest.

In the fifth step, The BFS will use the label node obtained by PageRank as a starting node, and traverse MWT to obtain a label sequence. The sequence is used to guide the construction order for each classifier in the chain model to solve the randomness problem of the classification model.

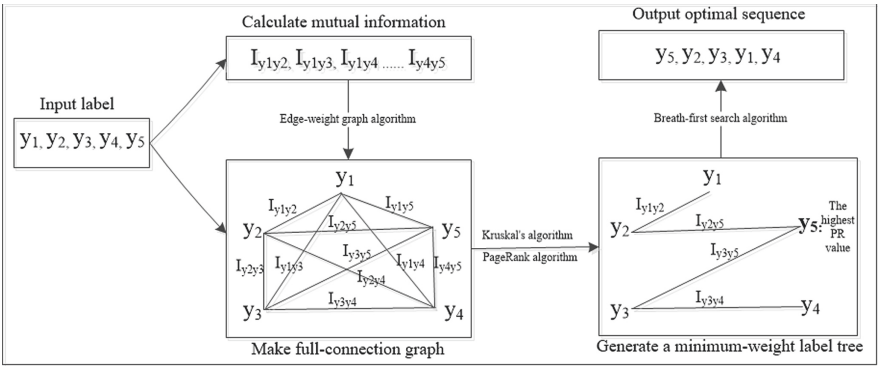

Fig. 3. Calculation process of the OSS

Algorithm 5. Description of OSS algorithm

Find the labels' optimized chain order.
Input: $D=(x_1,x_2,\cdots,x_n|y_1,y_2,\cdots,y_L)$
Output: $OS(y_1,y_2,\cdots,y_L)$

1. ▷ Calculate mutual information according to definition 1
2. *for* i = 1,2 ⋯ , L
3. *for* j=i+1,2,⋯,L
4. $I_{ij} \leftarrow MI(y_i,y_j)$
5. $Array\ \mathrm{A} \leftarrow \overline{\mathrm{I_{ij}}}$
6. End for
7. End for
8. ▷ Make a fully connected graph
9. $G \leftarrow Edge\text{-}weighted\ graph(L,A)$
10. ▷ Determine the root node by PageRank
11. $V \leftarrow PageRank(D)$
12. ▷ Get the maximum weight label Tree
13. $T \leftarrow Kruskal(G,V)$
14. ▷ Get the optimal sequence
15. $OS(y_1,y_2,\cdots,y_L) \leftarrow Breadth\text{-}first\ search(\mathrm{T})$
16. Return OS

3.4 The ATDCC-OS Framework

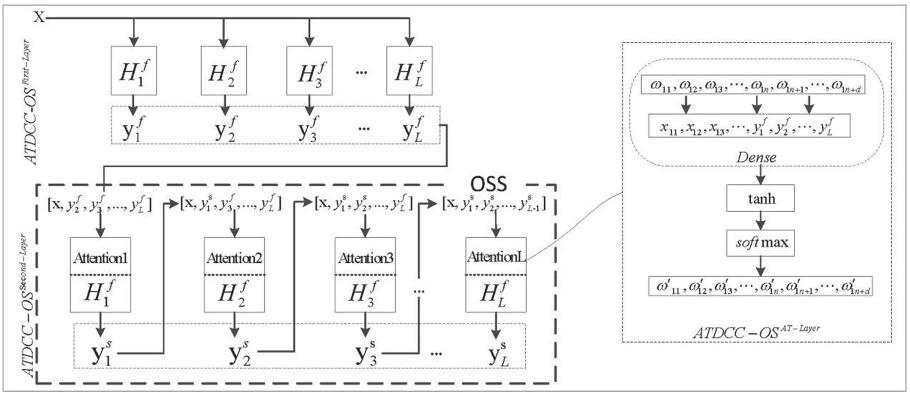

Fig. 4. Framework of the ATDCC-OS model

The ATDCC-OS framework is shown in Fig. 4, where ATDCC$^{\text{First-layer}}$ and ATDCC$^{\text{Second-layer}}$ are the first and second layers of the model, respectively. The OSS algorithm is used to optimize the second-layer chain model of ATDCC-OS, find a label

optimal sequence in it, and use the optimal sequence as the training order for each classifier in the second-layer model. The ATDCC$^{\text{AT-layer}}$ is the attention mechanism layer in the second layer, which is used to identify more important attributes to current classification tasks for each classifier in the second layer.

Algorithm 6. Description of the train stage of the ATDCC-OS

D is training set, L is number of labels
TRAINING $D=\{(x_i,Y_i)\,|\,i=1,2,\cdots,m\}$

1. *for* j = 1,2,⋯,L
2. $D_{y_j}^{f} \leftarrow \{\ \}$
3. *do* $\mathrm{x} \leftarrow [\mathrm{x}_{i1},\mathrm{x}_{i2},\cdots,\mathrm{x}_{im}]$
4. $D_{y_j}^{f} \leftarrow x \bigcup y_j$
5. $H_j^{f} \leftarrow \mathrm{B}(D_{y_j}^{f})$
6. end for
7. for j=sort1,2,⋯,LbyOSS(D)
8. $D_{y_j}^{s} \leftarrow \{\ \}$
9. *do* $\mathrm{x} \leftarrow [\mathrm{x}_{i1},\mathrm{x}_{i2},\cdots,\mathrm{x}_{im},\mathrm{y}_1,\cdots,\mathrm{y}_{j-1},\mathrm{y}_{j+1},\cdots,\mathrm{y}_L]$
10. $W \leftarrow \{W_{i1},W_{i2},\cdots,W_{im+L}\}$
11. $W \leftarrow attention\ (x)$
12. $x' \leftarrow x \times W$
13. $D_{y_j}^{s} \leftarrow x' \bigcup y_j$
14. $H_j^{s} \leftarrow \mathrm{B}(D_{y_j}^{s})$
15. $y_j = H_j^{s}(x')$
16. End for

Algorithm 7. Description of the test stage of the ATDCC-OS

Classify (x)

1. Global $c^{f}=(y_1^{f},y_2^{f},\cdots,y_L^{f})$, $c^{s}=(y_1^{s},y_2^{s},\cdots,y_L^{s})$
2. *for* j = 1,2,⋯,L
3. *do* $\mathrm{x} \leftarrow [\mathrm{x}_{i1},\mathrm{x}_{i2},\cdots,\mathrm{x}_{im}]$
4. $\mathrm{y}_j^{f} \leftarrow H_{y_j}^{f}(\mathrm{x})$
5. End for
6. $\mathrm{x}' \leftarrow [\mathrm{x},\mathrm{y}_1^{f},\mathrm{y}_2^{f},\cdots,\mathrm{y}_L^{f}]$
7. *for* $\mathrm{j}=\mathrm{sort}1,2,\cdots,\mathrm{L}\ \mathrm{by}\ \mathrm{OS\text{-}CC(D)}$
8. *do* $\mathrm{x}'' \leftarrow \mathrm{x}' \times \mathrm{W}_j'$
9. $y_j^{s} \leftarrow H_j^{s}(x'')$
10. End for
11. $c^{s} \leftarrow (y_1^{s},y_2^{s},\cdots,y_L^{s})$
12. *Return* c^{s}

4 Experiments

In this section, we will compare ATDCC-OS with DLMC-OS, BR, CC, and MBR on the basis of five evaluation metrics on seven benchmark multi-label data sets. We will introduce the multi-label benchmark datasets, evaluation measurements, and experiment setup in turn, and finally show the experimental results and discuss them.

4.1 Data Sets

The multi-label benchmark datasets used in the experiments are derived from the standard dataset provided by the Mulan [49] platform. Table 1 displays multi-label datasets from a variety of domains, and their associated statistics. Here, N is the number of instances in the dataset. F is the number of attributes included in each instance of the dataset. L is the number of labels in the dataset. Label Cardinality (LCard) is a standard measure of "multi-labelled-ness," introduced in Tsoumaskas and Katakis [50]. It is simply the average number of labels associated with each example. This measure gives a good idea of label frequency, but gives no indication of the regularity or uniformity of the labelling scheme.

Table 1. Multi-label datasets

Dataset	N	F	L	LCard	Type
Flags	194	19	7	3.392	Images
Emotion	593	72	6	1.87	Music
Birds	654	300	21	1.104	Audio
Medical	978	1449	45	1.245	Text
Enron	1702	1001	53	3.38	Text
Yeast	2417	103	14	4.24	Biology
Bibtex	7395	1836	159	2.40	Text

4.2 Evaluation Methods

In multi-label classification experiments, it is important to evaluate the performance of each algorithm. In order to evaluate the algorithm better, we use Average Precision, Coverage, One-Error, Ranking Loss, and Micro-averaged AUC.

(1) Average Precision: Average precision [12] is a measure that combines recall and precision for ranked retrieval results. It evaluates the average fraction of relevant labels ranked higher than a particular label $y \in y_i$. For this indicator, the bigger the value, the better. The formula for Average Precision is as follows:

$$avgprec_D(H) = \frac{1}{P}\sum_{i=1}^{P}\frac{1}{|Y_i|}\sum_{y\in Y_i}\frac{\left|\left\{y'\middle|rank_C(x,y') \le rank_C(x_i,y), y' \in Y_i\right\}\right|}{rank_C(x_i,y)}$$

Where $rank(.)$ is a sort function.

(2) Coverage [12]: the coverage measure is for assessing the performance of a system for all the possible labels of documents. That is, coverage measures how far we need, on average, to go down the list of labels in order to cover all the possible labels assigned to a document. Coverage is loosely related to precision at the level of perfect recall. For this indicator, the smaller value is better. The formula for Coverage is as follows:

$$coverage_D(H) = \frac{1}{P}\sum_{i=1}^{P} \max rank_f(x_i, y) - 1$$

Where $rank(.)$ is the sort function that matches the classifier $H(.)$.

(3) One-Error [51]: The one-error evaluates the fraction of examples whose top-ranked label is not in the relevant label set. For this indicator, the smaller value is better. The formula for One-Error is as follows:

$$one - error_D(H) = \frac{1}{P}\sum_{i=1}^{P} \left\| \left[\arg\max_{y \in \gamma} f(x_i, y) \notin Y_i \right] \right\|$$

Where $f(.)$ is a real-valued function corresponding to the multi-label classifier $H(.)$.

(4) Ranking Loss [12]: Ranking Loss is used in the case where the class label sorting of the response sample is out of order, that is, in the class label sorting queue, the class label unrelated to the instance is located before the related class label. For this indicator, the smaller value is better. The formula for Ranking Loss is as follows:

$$rloss_D(H) = \frac{1}{P}\sum_{i=1}^{P} \frac{1}{|Y_i|\left|\overline{Y_i}\right|} \left| \left\{ \left(y', y''\right) | f\left(x_i, y'\right) \leq f\left(x_i, y''\right), \left(y', y''\right) \in Y_i \times \overline{Y_i} \right\} \right|$$

(5) Micro-averaged AUC [51]: This indicator is the area under the ROC curve, which is between 0.1 and 1. AUC as a numerical value can be used to evaluate the quality of the classifier intuitively. For this indicator, the bigger value is better. The formula for Micro-averaged AUC is as follows:

$$AUC_{micro} = \frac{\left| \left\{ \left(x', x'', y', y''\right) | f\left(x', y'\right) \geq f\left(x'', y''\right), \left(x', y'\right) \in S^+, \left(x'', y''\right) \in S^- \right\} \right|}{\left|S^+\right|\left|S^-\right|}$$

The right side of the equation follows from the close relation between AUC and the Wilcoxon-Mann-Whitney statistic [52], where $f(.)$ is a real-valued function, $S^+ = \{(x_i, y) | y \in Y_i, 1 \leq i \leq p\}$ and $S^- = \{(x_i, y) | y \notin Y_i, 1 \leq i \leq p\}$ is a set of related (unrelated) label pairs.

4.3 Experimental Set up

In this experiment, we evaluate all algorithms under a Mulan platform. Mulan [49] is a multi-label classification open source library developed based on Weka. SMO is used as

the base classification algorithm. We have considered four classifiers to perform comparisons: DLMC-OS, MBR, CC, and BR. We set 80% of each complete dataset as training sets and the remaining part is used as testing sets. Because of its high computational efficiency, low memory requirements, and fast convergence, the Adam algorithm is very popular with the field of deep learning. Adam [53] is used to optimize the loss function. Adam's hyperparameters are set to the default parameters suggested in [53]: alpha = 0.001, beta1 = 0.9, beta2 = 0.999, and epslon = 10^{-8}.

4.4 Results and Discussion

Tables 2–6 give the comparisons of the performances of ATDCC-OS with DLMC, MBR, CC, and BR for Average Precision, Coverage, One-Error, Ranking Loss, and Micro-averaged AUC. The Fredman [54] test based on the method average ranking (Ave. Rank) is used to evaluate the difference between the algorithms.

From the overall analysis of Tables 2–6, it can be seen that ATDCC-OS has the best performance on all datasets, DLMC-OS ranks second, and other algorithms are relatively stable. From Ave. Rank's point of view, ATDCC-OS is far superior to other algorithms, and the highest gap is about four times.

Among the evaluation metrics, Average Precision and Micro-averaged AUC are mainly used to evaluate and reflect the classification performance of the classifier visually. The larger the metrics value, the better. According to Tables 2 and 3, ATDCC-OS and DLMC-OS are superior to other algorithms. This conclusion is not surprising; the main reason is that these two algorithms benefit from the design of their two-layer structure classification model and the interaction of tag information. On this basis, ATDCC-OS also introduces attention mechanism enhancing classifier sensitivity. The three types of indicators—Coverage, Ranking Loss, and One-Error—are mainly used to measure irrelevant labels before the relevant labels in the classification results. From Tables 4, 5, and 6, it can be found that ATDCC-OS and DLMC-OS are still superior to other algorithms, BR is centered, and MBR and CC are the worst. ATDCC-OS and DLMC-OS benefit from the optimization algorithm (such as OSS), so that the training of each classifier becomes orderly. The random sequence of CC and MBR is the cause of poor results. Instead, BR without considering the label sequence is better than them.

From the dataset point of view, ATDCC-OS performs better than other algorithms on most datasets, but it does not perform well on Yeast and Birds. As J. Read [9] stated, an algorithm cannot perform best on all types of datasets. The quality of the algorithm depends not only on the design of the structure, but also on the type and size of the data.

Table 2. Performance comparison of algorithms based on average precision

Dataset	BR	CC	MBR	DLMC-OS	ATDCC-OS
Flags	0.7952(2.5)	0.7950(4)	0.7308(5)	0.7952(2.5)	0.8315(1)
Emotion	0.7157(4)	0.7018(5)	0.7742(2)	0.7438(3)	0.8085(1)
Birds	0.4311(3.5)	0.4312(2)	0.4198(5)	0.4311(3.5)	0.4884(1)
Medical	0.8260(3)	0.8206(5)	0.8207(4)	0.8285(2)	0.8606(1)
Enron	0.3802(4)	0.3675(5)	0.3890(2)	0.3815(3)	0.4224(1)
Yeast	0.6669(2.5)	0.6618(4)	0.6671(1)	0.6669(2.5)	0.6533(4)
Bibtex	0.3938(5)	0.3961(2.5)	0.3961(2.5)	0.3949(4)	0.4307(1)
Ave. rank	3.5(4)	3.93(5)	3.07(3)	2.93(2)	1.43(1)

Table 3. Performance comparison of algorithms based on micro-averaged AUC

Dataset	BR	CC	MBR	DLMC-OS	ATDCC-OS
Flags	0.7671(3)	0.7544(4)	0.7010(5)	0.7710(2)	0.7941(1)
Emotion	0.7336(4)	0.7321(5)	0.7750(2)	0.7527(3)	0.8615(1)
Birds	0.6731(3.5)	0.6798(2)	0.6803(1)	0.6731(3.5)	0.6495(5)
Medical	0.9012(2.5)	0.8991(5)	0.8994(4)	0.9012(2.5)	0.9565(1)
Enron	0.7085(2)	0.7020(5)	0.7081(4)	0.7083(3)	0.7923(1)
Yeast	0.7363(4)	0.7303(5)	0.7364(2.5)	0.7364(2.5)	0.7681(1)
Bibtex	0.6925(2)	0.6896(4)	0.6838(5)	0.6922(3)	0.7165(1)
Ave. Rank	3(3)	4.29(5)	3.35(4)	2.79(2)	1.57(1)

Table 4. Performance comparison of algorithms based on coverage

Dataset	BR	CC	MBR	DLMC-OS	ATDCC-OS
Flags	4.5385(2.5)	4.6923(4)	4.8462(5)	4.5385(2.5)	3.8718(1)
Emotion	2.6639(5)	2.6471(4)	2.4202(2)	2.6218(3)	1.7815(1)
Birds	4.6406(3)	4.6406(3)	4.8438(5)	4.6406(3)	4.2656(1)
Medical	5.6939(3)	5.9745(5)	5.8367(4)	5.6888(2)	2.8367(1)
Enron	33.3588(4)	34.0000(5)	32.6206(2)	33.1647(3)	27.4676(1)
Yeast	9.0787(3.5)	8.6046(2)	9.0828(5)	9.0787(3.5)	7.0373(1)
Bibtex	68.9277(3)	68.6748(2)	70.4834(5)	69.0730(4)	61.5463(1)
Ave. rank	3.42(4)	3.57(3)	4(5)	3(2)	1(1)

Table 5. Performance comparison of algorithms based on one-error

Dataset	BR	CC	MBR	DLMC-OS	ATDCC-OS
Flags	0.2308(3)	0.2308(3)	0.3590(5)	0.2308(3)	0.2051(1)
Emotion	0.3277(4)	0.3782(5)	0.2521(1)	0.2773(3)	0.2689(2)
Birds	0.7813(2.5)	0.8125(5)	0.7969(4)	0.7813(2.5)	0.7500(1)
Medical	0.1582(3.5)	0.1582(3.5)	0.1582(3.5)	0.1531(1)	0.1582(3.5)
Enron	0.5882(2)	0.6235(5)	0.6118(4)	0.6088(3)	0.5676(1)
Yeast	0.2609(2.5)	0.2609(2.5)	0.2609(2.5)	0.2609(2.5)	0.4472(5)
Bibtex	0.5531(3)	0.5571(5)	0.5321(2)	0.5544(4)	0.4956(1)
Ave. rank	2.92(3)	3.71(5)	3.14(4)	2.71(2)	2.07(1)

Table 6. Performance comparison of algorithms based on ranking loss

Dataset	BR	CC	MBR	DLMC-OS	ATDCC-OS
Flags	0.2615(2.5)	0.2697(4)	0.4009(5)	0.2615(2.5)	0.1915(1)
Emotion	0.3141(4)	0.3147(5)	0.2444(2)	0.2780(3)	0.1579(1)
Birds	0.1832(2.5)	0.1836(4)	0.1894(5)	0.1832(2.5)	0.1556(1)
Medical	0.0956(3)	0.0984(5)	0.0967(4)	0.0954(2)	0.0424(1)
Enron	0.3258(4)	0.3350(5)	0.3159(2)	0.3242(3)	0.2435(1)
Yeast	0.3150(3.5)	0.3270(5)	0.3149(2)	0.3150(3.5)	0.2433(1)
Bibtex	0.2789(2)	0.2793(3)	0.2877(5)	0.2794(4)	0.2595(1)
Ave. rank	3.07(3)	4.43(5)	3.57(4)	2.93(2)	1(1)

5 Conclusions

Many problem-transformation multi-label classification algorithms consider the correlation between labels by extending attributes, but ignore the different importance of each attribute value of different classification tasks. Based on the wide application and good results from attention mechanism in various types of deep-learning tasks, such as natural language processing, image recognition, and speech recognition, this paper proposes the ATDCC-OS algorithm. This algorithm integrates the multi-label classification framework of three classic problem-conversion types, and combines their common advantages to solve the problem of ignoring the correlation between labels in the classification process. The algorithm also introduces the "update replacement" idea to solve the problem that the label information interaction is not real-time. At the same time, the algorithm also introduces the attention mechanism to calculate the weight of each feature attribute dynamically, and then distinguishes the more important attribute values to the current classification target for each classifier, enhances the sensitivity of each classifier, and improves the classification accuracy. The ATDCC-OS method solves the

defect that the BR algorithm does not consider the correlation between labels, corrects the redundant interaction and information loss of the MBR algorithm, and optimizes the problem that the DLMC-OS does not find the optimal label sequence. It also solves the problem of neglecting the importance of different feature attributes to different classification targets by extending the attribute method to consider the correlation between labels. In order to verify the classification performance of the algorithm, we used five evaluation indicators to compare ATDCC-OS with DLMC-OS, MBR, CC, and BR on seven standard data sets. The experimental results show that ATDCC-OS achieves good results compared to other algorithms.

Acknowledgements. The authors gratefully acknowledge the financial support of the Planning Subject for the 13th Five Year Plan of National Education Sciences under Grant No. DCA160258 and the Key Research Project of Education Department of Sichuan Province of China under Grant No. 18ZA319.

References

1. Shao, H., Li, G., Liu, G., Wang, Y.: Symptom selection for multi-label data of inquiry diagnosis in traditional Chinese medicine. Sci. China Inf. Sci. **56**(5), 1–3 (2013)
2. Glinka, K., Wosiak, A., Zakrzewska, D.: Improving children diagnostics by efficient multi-label classification method. In: Piętka, E., Badura, P., Kawa, J., Wieclawek, W. (eds.) Information Technologies in Medicine. AISC, vol. 471, pp. 253–266. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-39796-2_21
3. Yao, L., Poblenz, E., Dagunts, D., Covington, B., Bernard, D., Lyman, K.: Learning to diagnose from scratch by exploiting dependencies among labels. arXiv preprint arXiv:1710.10501 (2017)
4. Zhu, H., et al.: Learning tree-based deep model for recommender systems. In: The 2018 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (2018)
5. Son, J., Kim, S.B., Kim, H., Cho, S.: Review and analysis of recommender systems. J. Korean Inst. Ind. Eng. **41**(2), 185–208 (2015)
6. Bogaert, M., Lootens, J., Van den Poel, D., Ballings, M.: Evaluating multi-label classifiers and recommender systems in the financial service sector. Eur. J. Oper. Res. **279**(2), 620–634 (2019)
7. Ray, J., Heng Wang, D., Tran, Y.W., Feiszli, M., Torresani, L., Paluri, M.: Scenes-objects-actions: a multi-task, multi-label video dataset. In: Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y. (eds.) Computer Vision – ECCV 2018: 15th European Conference, Munich, Germany, September 8–14, 2018, Proceedings, Part XIV, pp. 660–676. Springer International Publishing, Cham (2018). https://doi.org/10.1007/978-3-030-01264-9_39
8. Chen, Z.M., Wei, X.S., Wang, P., Guo, Y.: Multi-label image recognition with graph convolutional networks. In: CVPR (2019)
9. Read, J., Pfahringer, B., Holmes, G., Frank, E.: Classifier chains for multi-label classification. In: Buntine, W., Grobelnik, M., Mladenić, D., Shawe-Taylor, J. (eds.) ECML PKDD 2009. LNCS (LNAI), vol. 5782, pp. 254–269. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-04174-7_17
10. Boutell, M.R., Luo, J., Shen, X., Brown, C.M.: Learning multi-label scene classification. Patt. Recogn. **37**(9), 1757–1771 (2004)

11. Godbole, S., Sarawagi, S.: Discriminative methods for multi-labeled classification. In: Dai, H., Srikant, R., Zhang, C. (eds.) PAKDD 2004. LNCS (LNAI), vol. 3056, pp. 22–30. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-24775-3_5
12. Liu, G.Q., Guo, T.: Algorithm for multi-label classification with optimal sequence based on double layers. Comput. Eng. Des. **37**(4), 921-927+948 (2016)
13. Pan, C., Tan, J., Feng, D., Li, Y.: Very short-term solar generation forecasting based on LSTM with temporal attention mechanism. In: ICCC (2019)
14. Chen, W.J., Shao, Y.H., Li, C.N., Deng, N.Y.: MLTSVM: a novel twin support vector machine to multi-label learning. Patt. Recogn. **52**, 61–74 (2016)
15. Xu, X., Shan, D., Li, S., Sun, T., Xiao, P., Fan, J.: Multi-label learning method based on ML-RBF and laplacian ELM. Neurocomputing **331**, 213–219 (2019)
16. Zhang, N., Ding, S., Zhang, J.: Multi layer ELM-RBF for multi-label learning. Appl. Soft. Comput. **43**, 535–545 (2016)
17. Roseberry, M., Krawczyk, B., Cano, A.: Multi-label punitive kNN with self-adjusting memory for drifting data streams. ACM Trans. Knowl. Disc. Data **13**(6), 1–31 (2019). https://doi.org/10.1145/3363573
18. Yapu, D.: An Improved ML-KNN Approach for Weibo text classification. Chin. Comput. Commun. **7**, 18 (2018)
19. Wu, G., Zheng, R., Tian, Y., Liu, D.: Joint ranking SVM and binary relevance with robust Low-rank learning for multi-label classification. Neural Netw. **122**, 24–39 (2020)
20. Charte, F., Rivera, A., del Jesus, M.J., Herrera, F.: Improving multi-label classifiers via label reduction with association rules. In: Corchado, E., Snášel, V., Abraham, A., Woźniak, M., Graña, M., Cho, S.-B. (eds.) HAIS 2012. LNCS (LNAI), vol. 7209, pp. 188–199. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-28931-6_18
21. Luo, F., Guo, W., Yu, Y., Chen, G.: A multi-label classification algorithm based on kernel extreme learning machine. Neurocomputing **260**, 313–320 (2017)
22. Kulessa, M., Mencía, E.L.: Dynamic classifier chain with random decision trees. In: Soldatova, L., Vanschoren, J., Papadopoulos, G., Ceci, M. (eds.) DS 2018. LNCS (LNAI), vol. 11198, pp. 33–50. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-01771-2_3
23. Ali, T., Asghar, S.: Efficient label ordering for improving multi-label classifier chain accuracy. J. Nat. Sci. Found. Sri Lanka **47**(2), 175 (2019). https://doi.org/10.4038/jnsfsr.v47i2.9159
24. Firat, O., Cho, K., Bengio, Y.: Multi-way, multilingual neural machine translation with a shared attention mechanism. arXiv preprint. arXiv:1601.0107 (2016)
25. Mnih, V., Heess, N., Graves, A.: Recurrent models of visual attention. In: NIPS, pp. 2204–2212. MIT Press, US (2014)
26. Borji, A., Itti, L.: State-of-the-art in visual attention modeling. IEEE Trans. Pattern Anal. Mach. Intell. **35**(1), 185–207 (2012)
27. Bahdanau, D., Cho, K., Bengio, Y.: Neural machine translation by jointly learning to align and translate. arXiv preprint. arXiv:1409.0473 (2014)
28. Mnih, V., Heess, N., Graves, A.: Recurrent models of visual attention. In:NIPS, pp. 2204–2212 (2014)
29. Rush, A.M., Chopra, S., Weston, J.: A neural attention model for abstractive sentence summarization. arXiv preprint. arXiv:1509.00685 (2015)
30. Chen, H., Sun, M., Tu, C., Lin, Y., Liu, Z.: Neural sentiment classification with user and product attention. In: EMNLP (2016)
31. Yang, Z., Yang, D., Dyer, C., He, X., Smola, A., Hovy, E.: Hierarchical attention networks for document classification. In:NAACL HLT, pp. 1480–1489. ACL, California (2016)
32. He, R., Lee, W. S., Ng, H. T., Dahlmeier, D.: An unsupervised neural attention model for aspect extraction. In: the 55th Annual Meeting of the Association for Computational Linguistics, pp. 388–397. ACL, Canada (2017)

33. Yin, W., Schütze, H., Xiang, B., Zhou, B.: Abcnn: Attention-based convolutional neural network for modeling sentence pairs. Trans. Assoc. Comput. Linguist. **4**, 259–272 (2016)
34. Vaswani, A., et al.: Attention is all you need. In: NIPS, pp. 5998–6008. MIT Press, US (2017)
35. Coelho, F., Braga, A.P., Verleysen, M.: A mutual information estimator for continuous and discrete variables applied to feature selection and classification problems. Int. J. Comput. Intell. Syst. **9**(4), 726–733 (2016)
36. Sefidian, A.M., Daneshpour, N.: Missing value imputation using a novel grey based fuzzy c-means, mutual information based feature selection, and regression model. Exp. Syst. Appl. **115**, 68–94 (2019)
37. Cao, X., Cong, G., Jensen, C.S.: Mining significant semantic locations from GPS data. Proc. VLDB Endow. **3**(1–2), 1009–1020 (2010)
38. Yanagisawa, T., et al.: Electrocorticographic control of a prosthetic arm in paralyzed patients. Ann. Neurol. **71**(3), 353–361 (2012)
39. Liu, X.S., et al.: High-resolution peripheral quantitative computed tomography can assess microstructural and mechanical properties of human distal tibial bone. J. Bone Miner. Res. **25**(4), 746–756 (2010)
40. Brin, S., Page, L.: The anatomy of a large-scale hypertextual web search engine. Comput. Netw. ISDN Syst. **30**(1–7), 107–117 (1998). https://doi.org/10.1016/S0169-7552(98)00110-X
41. Singhal, R., Srivastava, S.R.: Enhancing the page ranking for search engine optimization based on weightage of in-linked web pages. In: ICRAIE (2016)
42. Gupta, P., Goel, A., Lin, J., Sharma, A., Wang, D., Zadeh, R.: Wtf: the who to follow service at Twitter. In: IW3C2 (2013)
43. Fletcher, P., Hoyle, H., Patty, C.W.: Foundations of Discrete Mathematics. PWS-KENT Pub. Co., Boston (1991)
44. Jiang, D., Wang, Y., Lv, Z., Qi, S., Singh, S.: Big data analysis based network behavior insight of cellular networks for industry 4.0 applications. IEEE Trans. Ind. Inf. **16**(2), 1310–1320 (2020). https://doi.org/10.1109/TII.2019.2930226
45. Jiang, D., Huo, L., Lv, Z., Song, H., Qin, W.: A joint multi-criteria utility-based network selection approach for vehicle-to-infrastructure networking. IEEE Trans. Intell. Transp. Syst. **19**(10), 3305–3319 (2018)
46. Wang, Y., Jiang, D., Huo, L., Zhao, Y.: A new traffic prediction algorithm to software defined networking. Mob. Netw. Appl., 1–10 (2019).https://doi.org/10.1007/s11036-019-01423-3
47. Wang, F., Jiang, D., Qi, S.: An adaptive routing algorithm for integrated information networks. China Commun. **16**(7), 195–206 (2019)
48. Kruskal, J.B.: On the shortest spanning subtree of a graph and the traveling salesman problem. Proc. Am. Math. Soc. **7**(1), 48–50 (1956)
49. Tsoumakas, G., Spyromitros-Xioufis, E., Vilcek, J., Vlahavas, I.: Mulan: a java library for multi-label learning. J. Mach. Learn. Res. **12**, 2411–2414 (2011)
50. Tsoumakas, G., Katakis, I.: Multi-label classification: an overview. Int. J. Data Warehouse. Min. **3**(3), 1–3 (2007)
51. Zhang, M.L., Zhou, Z.H.: A review on multi-label learning algorithms. IEEE Trans. Knowl. Data Eng. **26**(8), 1819–1837 (2013)
52. Hanley, J.A., McNeil, B.J.: The meaning and use of the area under a receiver operating characteristic (ROC) curve. Radiology **143**(1), 29–36 (1982)
53. Da, K.: A method for stochastic optimization. arXiv preprint. arXiv:1412.6980 (2014)
54. DemšCar, J.: Statistical comparisons of classifiers over multiple data sets. J. Mach. Learn. Res. **7**(1), 30 (2006)