IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received December 11, 2018, accepted December 30, 2018, date of publication January 11, 2019, date of current version February 6, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2892120

A Randomized Clustering Forest Approach for Efficient Prediction of Protein Functions

HONG TANG[1], YUANYUAN WANG[2], SHAOMIN TANG[3], DIANHUI CHU[4], AND CHUNSHAN LI[4]

[1]School of Electronics and Information Technology, Sun Yat-sen University, Guangzhou 510275, China
[2]College of Education, Shenzhen University, Shenzhen 518060, China
[3]School of Mechanical and Automotive Engineering, South China University of Technology, Guangzhou 510630, China
[4]Department of Computer Science, Harbin Institute of Technology, Weihai 264209, China

Corresponding author: Chunshan Li (lics@hit.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61772159, in part by the National Natural Science Foundation of Shandong Province under Grant ZR201702150244, and in part by the University Co-Construction Project.

ABSTRACT With the advances in genetic sequencing technology, the automated assignment of protein function has become a key challenge in bioinformatics and computational biology. In nature, many kinds of proteins consist of a variety of structural domains, and each domain almost holds its own function independently or implements a new function in cooperation with neighbors. Thus, a multi-domain protein function prediction problem can be converted into multi-instance multi-label (MIML) learning tasks. In this paper, we propose a novel ensemble MIML algorithm called multi-instance multi-label randomized clustering forest (MIMLRC-Forest) for protein function prediction. In MIMLRC-Forest, we develop a set of hierarchical clustering trees and conduct a label transfer mechanism to identify the relevant function labels in learning process. The clustering tree with a hierarchical structure can handle the multi-label problem by exploiting more discriminable label concepts at higher-level nodes and by transferring less discriminable labels into the lower-level nodes. Then, the label dependency can be computed by aggregating tree labels for protein function prediction. Extensive experiments on five real-world protein data sets show the effectiveness of the proposed algorithm compared with several state-of-the-art baselines, including MIMLSVM, MIMLNN, MIML-kNN, EnMIMLNN, and M3MIML.

INDEX TERMS Protein function prediction, multi-instance multi-label learning, randomized clustering tree, Hausdorff distance, ensemble learning.

I. INTRODUCTION

With a revolution in genomics and proteomics in last decade, biological sequence data are undergoing an explosive growth in the content and quality. Detection of functions to unknown proteins has become a key challenge in genome sequencing projects. Due to its inherent time-consuming and labor-intensive, it is hard to apply this large-scale prediction to biological experiment method. In recent years, computational methods have been successful applied to those prediction tasks, in which the machine-learning based methods have obtained good predictive performance [1].

A protein domain can be defined as a conserved part of a given protein sequence and structure. Each of them forms a compact three-dimensional structure and often can be independently stable and folded. Many proteins are composed of a variety of structural domains. One domain may appear in many different proteins. Molecular evolution proceeds based on these domains which may be recombined in different arrangements to create proteins with different functions. Multi-domain proteins always have emerged during evolution to generate new functions [2], and they appeared in majority of genomic proteins, two-thirds in unicellular organisms and more than 80% in metazoa [3].

Recently, *Multi-instance multi-label* (MIML) learning framework [4], [5] has been widely used to solve problems involving multiple instance associated with multiple class concepts, where an object is described by a bag of instances and then mapped to multiple labels simultaneously. Obviously multi-domain protein can be represented in MIML framework [24]. Specifically, each protein domain may be seen as an instance of input object, e.g., BAR Domain, PX Domain, VPS29, etc. and each multi-domain protein is usually annotated with multiple function labels. Therefore, it is a possible way to apply MIML framework to tackle the protein function prediction problem [9], [10]. Previous studies have shown that the learning performance of protein

function prediction can be significantly improved using the MIML framework [11].

A straightforward approach for MIML learning is to degenerate the problem to *multi-label* (ML) or *multi-instance* (MI) problem based on predefined degeneration strategy so that existing MI or ML techniques can be directly applied to solve the problem. For instance, *multi-instance multi-label SVM* (MIMLSVM) and *multi-instance multi-label Boosting*(MIML Boosting) are well-known degeneration strategy-based methods for MIML learning task achieving impressive performances on certain type of data sets [4]. However, these approaches are not suitable for protein data due to its underlying relationships among multi-function structure. Direct application of the degeneration strategies may lead to a loss of useful classification information, e.g., the correlation between instance distribution and labels. Moreover, a specific protein function may require multiple domains to cooperate with each other. A protein represented by multi-instance may exploit more relationship information between domains and functions, which would facilitate effective learning.

As reported in [12], label dependency relations can significantly improve MIML learning performance. In order to exploit the dependency relations, some approaches assume that the relations are externally provided, e.g., predefined label hierarchies of data sets [13], [14]. However, such prior dependency relations are usually unavailable in practice. In contrast, some other approaches [1], [15] tempted to automatically learn the label dependency relations from limited resource, e.g., using label's co-occurrences in their training set. These approaches may lead to an overfitting problem in the learning phase [16]. In addition, exploiting the correlation between instances and labels in one object is also an effective manner for providing useful information to MIML learning [17], [18].

Motivated by recent researches in MIML and label dependency learning, this paper proposes a novel ensemble MIML learning framework called Multi-Instance Multi-Label Randomized Clustering Forest (MIMLRC-Forest) for protein function prediction. The MIMLRC-Forest constructs a set of hierarchical trees to learn label dependency relations. These randomized clustering trees are combined as an ensemble to predict protein functions. To seek a suitable hierarchical tree structure for locating relevant proteins with strong label dependency relations at the same tree node, this paper designs an automatic tree structure generation algorithm to distinguish relevant labels for nodes through a label transfer mechanism.

The algorithm is implemented in three tasks: Firstly, a data clustering algorithm is applied to allocate the protein data as objects to tree nodes according to clustering results. For each node, a set of relevant labels are exploited by computing its label purity over the classes. Secondly, a label transfer mechanism is proposed for recursively propagating the relevant labels from root to leaf nodes. In case a relevant label is assigned to a node, all of its child nodes are automatically associated with the label and the algorithm finds a new relevant label (if any) which complies with the label dependencies of the objects in the child node. Eventually each node is denoted by multiple relevant labels obtained from the nodes at different levels. Hence, a new label dependency representation is formed that the learning models at different levels cooperate to exploit multi-label concepts for a given protein. Intuitively, the relevant labels at higher hierarchy levels are thematically more general and that at lower levels are more specific.

To evaluate the proposed MIMLRC-Forest approach, we compare it with five widely-used MIML learning algorithms, i.e. multi-instance multi-label support vector machine (MIMLSVM) [4], multi-instance multi-label nearest neighbor (MIMLNN) [5], multi-instance multi-label k nearest neighbor (MIMLKNN) [22], Maximum Margin Method for Multi-Instance Multi-Label (M3MIML) [18] and ensemble multi-instance multi-label nearest neighbor (EnMIMLNN) [11]. Experimens on five protein multi-instance multi-label data sets show that our MIMLRC-Forest approach outperforms the baseline methods, demonstrating its effectiveness in protein function prediction.

The main contributions of this paper are given as follows:

- proposing a new hierarchical classification approach to model the inherent label dependency relations between data into a tree structure to handle the multi-label classification task.
- designing a label transfer mechanism to assign protein data with the relevant class labels in the hierarchy way.
- developing an ensemble framework that builds a multitude of hierarchical multi-label trees and combines the predictions of different trees as an ensemble to make more effective predictions.

The rest of this paper is organized as follows: Section 2 presents the related work. The proposed MIMLRC-Forest approach is described in Section 3. The experimental results and evaluations are discussed in Section 4 and Section 5 concludes the work.

II. RELATED WORKS

Let $\mathcal{X} = \mathcal{R}^d$ be the d-dimensional input space and $\mathcal{Y} = \{1, 2, \cdots, n\}$ be a finite set of labels. Given a labeled training set $\mathcal{D} = \{(X_1, Y_1), (X_2, Y_2), \cdots, (X_i, Y_i) | 1 \leq i \leq n\}$ with n examples, the i-th example $X_i = \{x_1^{(i)}, \cdots, x_{n_i}^{(i)}\}$ contains a bag of n_i instances, and $Y_i = \{y_1^{(i)}, \cdots, Y_j^{(i)}\}$ is a set of labels associated with the example X_i. $Y_j^{(i)} = 1$ if the j-th label is a relevant label of example X_i and $Y_j^{(i)} = 0$ otherwise. The goal of multi-instance multi-label learning is to learn a function $f : 2^{\mathcal{X}} \rightarrow 2^{\mathcal{Y}}$ from $\mathcal{D}$ to map each example from $\mathcal{D}$ to a set of proper labels.

During the past few years, multi-instance multi-label learning has received a lot of attentions both in academic and industry. A number of MIML learning approaches have been developed for solving various types of real-world applications, such as text categorization [6], [7], [22],

bioinformatics [11], [23], [24], image classification [8], [25], [26], and video annotation [27].

In particular, Zhou and Zhang [4] was the first to apply the degeneration strategy to MIML learning and presented two approaches, namely MIMLSVM and MIMLBoosting methods. The MIMLSVM is firstly used a similarity matrix of examples to cluster training data by k-means clustering algorithm at instance level. Based on the medoids of the obtained clusters, a feature space is transformed to a distance matrix between the examples and the medoids. In this way, it successfully degenerates the MIML learning task to a ML problem, which can be easily handled by MLSVM method [4] or other ML learning methods [25], [28]. The MIMLBoosting converts each multi-label object into multiple single-label objects by assigning a single label to all the instances that belongs to the same object. It then applies boosting algorithms, such as MIBoosting [29] or previous MI learning approaches [30]–[32], to solve the derived multiple instances learning problem. In order to optimize the above algorithms, Zhou *et al*. [5] further explored MIMLSVM and MIMLNN methods which use MI-SVM [20] to replace the MIBoosting [29] in MIMLBoost and use a two-layer neural networks structure [33] to replace the MLSVM [25] in MIMLSVM, respectively.

In addition, some existing researchers found that exploiting correlations between instances (or objects) and labels in an effective manner was essential for achieving a good performance in a MIML learning task. For instance, Wu *et al*. [17] proposed a markov chain-based MIML learning algorithm, where the correlations between objects and labels are exploited by computing label ranks. They demonstrated the importance of a set of label to an object in determining the MIML learning performance. Although this method achieved a good performance on text and image data sets, it was ineffective for the protein function prediction task as the algorithm was always terminated by the steady-state probability matrix (termination conditions) at the first iteration. After that, Zhang and Zhou [18] developed a maximum margin approach for multi-instance multi-label learning (M3MIML) which directly discovers the connections between instances and labels. This approach assumes a linear model for each label, where the output of a specific label was the maximum prediction of MIML objects with respect to the relevant linear models. This method could not be well applied in practical applications due to it's expensive computation. Ge *et al*. [21] proposed LPBNI approach to identify potential lnc RNA-interacting proteins, by making full use of the known lnc RNA-protein interactions. Li *et al*. [19] applied a feature-based approach to extract protein interactions(LPIs) from biomedical literatures. Meng *et al*. [24] proposed PDAMIML and PDAMIML-Ensemble to predict the annotation of domains without knowing the whole protein sequences and annotations of other coexisting domains.

Recently, a number of other well-known MIML approaches was proposed. For instance, the MIMLNN [5] and EnMIMLNN methods [11] both employ a two-layer architecture to train a model for MIML learning task, where it's first layer exploited a relation matrix between the objects and the mediods of obtained label cluster and then transformed the matrix into an one-to-one weight matrix revealing the correlations between objects and labels. It's worth noting that EnMIMLNN was an optimized version of the method MIMLNN. Besides, MIMLKNN [22] approach utilizes k-nearest neighbor technique to solve the MIML learning task. What's more, many researchers utilized MIML learning framework to handle multi-graph learning task. Wu *et al*. built a MI model to represent multi-graph features in a multi-graph bag [34]. bMGC [35] approach employs dynamic weight adjustment at both bag- and graph-levels to select one subgraph as a weak classifier. MSVBL [36] approach explores subgraph features across different structure views to generate a bag constrained graph classification. puMGL [37] framework assigns each bag a confidence weight which can be dynamically adjusted to select ''reliable negative bags''. MILDM [38] approach identifies the best instances to directly discriminate bags in a new mapping space.

The protein function prediction task is essentially a MIML learning problem, where each multi-domain protein is annotated by a list of relevant function labels. There is a strategy to predict protein function by exploiting the label dependency of multi-domain protein data and building a tree structure model. Hierarchical tree-based model includes a series of learning algorithms with simple theoretical foundation and wide applications. Although tree-based models are rarely mentioned in MIML learning task, many tree-based multi-label classification methods have been proposed and applied [10], [39]–[44]. For instance, Clare and King [10] adjusted the C4.5 tree to deal with multi-label data by modifying the common formula for calculating the entropy. Blockeel *et al*. [45] developed a predictive clustering trees (PCT), while Vens *et al*. [46] presented several multi-label learning methods based on PCT framework where class labels are organized in a hierarchical pattern. Tsoumakas *et al*. [39] presented a tree structure-based algorithm called HOMER to handle data containing large number of labels. HOMER algorithm disjointed the entire label set into several subsets so as to build a tree by a balance clustering. ML-FOREST algorithm [52] constructed a set of hierarchical trees which can identify the multiple relevant labels in a hierarchy way.

Different from the existing tree structure-based multi-label learning methods, in our MIMLRC-Forest method, an ensemble of tree classifiers is constructed by addressing label dependency generated from root to leave nodes via a label transfer mechanism. The underlying label dependency transfer can be explicitly expressed in a hierarchical structure, offering a natural and comprehensive way in a MIML learning task.

III. METHOD

This section presents the detail of MIMLRC-Forest method for MIML learning. The method includes a classifier-tree

construction algorithm, a label transfer mechanism, and a new ensemble framework.

A. RANDOMIZED CLUSTERING TREES

Statistically, the joint conditional probability distribution $p(y|x)$ states the probability of label combination for a particular instance and it can be decomposed as:

$$\begin{aligned} p(y|x) &= p(y^1|x)p(y^2, \cdots, y^q|y^1, x) \\ &= p(y^1|x)p(y^2|y^1, x)p(y^3, \cdots, y^q|y^1, y^2, x) \\ &= p(y^1|x)p(y^2|y^1, x)\cdots p(y^q|y^1, \cdots, y^{q-1}, x) \end{aligned} \quad (1)$$

Our goal is to construct a hierarchical randomized clustering tree and find relevant labels assigned to the data examples at each tree node. The proposing new hierarchical tree algorithm studies the inherent label dependency hierarchically. More specifically, three folds construct the hierarchical structure: 1) At each internal node, the training data is partitioned into smaller subsets as its child nodes; 2) A group of relevant labels are associated with each node; 3) The relevant labels of a higher level node are transferred into its child nodes.

The pseudo code of the algorithm is described in Algorithm 1. It consists of the following major steps: 1) construct a new feature matrix $\mathcal{D}'$ if the node is the root (line 1-2); 2) decide whether it is an internal node or leaf node (line 3-5); 3) determine the set of reused labels from parent nodes (line 9); 4) randomly select an attribute and a threshold for generating child nodes (line 11-16); 5) calculate the score according to the current strategy of node partition (line 17); 6) find the best strategy for node partition (line 19); 7) create a decision node to make prediction (line 20-22); 8) invoking above steps for dealing examples in left and right child nodes recursively. These major steps to construct a hierarchical tree are detailed as follows.

The training examples of each function label are clustered using k-Medoids algorithm and the medoids of each obtained clusters are used to construct a new input feature space $\mathcal{D}'$ for the randomized clustering tree. More specifically, given a set of training objects $\mathcal{D} = \{X_i|1 \leq i \leq n\}$ and relevant label set $\mathcal{Y} = \{y_1, y_2, \cdots, y_l\}$, $\mathcal{D}_r = \{X_i|1 \leq i \leq m, m \leq n\}$ denotes the training set associated with r-th function label, where $\mathcal{D} = \mathcal{D}_1 \cup \mathcal{D}_2 \cup, \cdots, \cup \mathcal{D}_l$ and $|\mathcal{D}| \leq |\mathcal{D}_1| + |\mathcal{D}_2| +, \cdots, +|\mathcal{D}_l|$. Given two objects with multiple instances $X_a = \{x_1^{(a)}, x_2^{(a)}, \cdots, x_{n_a}{}^{(a)}\}$ and $X_b = \{x_1^{(b)}, x_2^{(b)}, \cdots, x_{n_b}{}^{(b)}\}$, the distance between the two objects can be computed by a novel Hausdorff distance, which is defined as Eq. 2 and Eq. 3:

$$H(X_a, X_b) = \frac{1}{3}\sum_{d\in\eta} H^d(X_a, X_b) \quad (2)$$

In Eq. 3, as shown at the bottom of the next page, $MinDist(x^{(b)}, X_a) = \min_{x^{(a)}\in X_a}||x^{(a)} - x^{(b)}||$ and $||.||$ denote the Euclidean distance between the two instances. According to the metric $H(X_a, X_b)$, the k-Medoids algorithm partitions the $\mathcal{D}_s$ into Q_s groups of clusters $C_j^s (1 \leq s \leq l, 1 \leq j \leq Q_s)$, where $C_j^s \cap C_k^s = \phi$ and the number of medoids Q_s for each function label is determined by the fraction parameter α as well as the size of $\mathcal{D}_i$,i.e, $Q_s = \alpha \times |\mathcal{D}_s|$. Thus, the feature dimension of training objects is $Q = \sum_{s=1}^{l} Q_s$ and the medoids M_j^s of cluster C_j^s are determined as:

$$M_j^s = \{arg \min_{X_a\in C_j^s} \sum_{X_b\in C_j^s} H(X_a, X_b)\}(1 \leq s \leq l, 1 \leq j \leq Q_s) \quad (4)$$

Based on the acquired cluster mediods, the correlation between j-th medoid and object X_i is computed by radial basis function neural networks rather than similarity measures in MIMLSVM. The computation function is defined as Eq. 5 and Eq. 6. The parameter σ is determined by the average Hausdorff distance between every two medoids. It is further used to control the smoothness of function $\psi_j^s(.)$. μ is a scaling factor.

$$\psi_j^s(X_i) = exp\Big(-\frac{H(X_i, M_j^s)^2}{2\sigma^2}\Big) \quad (1 \leq i \leq n, 1 \leq s \leq l, 1 \leq j \leq Q_s) \quad (5)$$

$$\sigma = \mu \times \Big(\frac{\sum_{i=1}^{Q-1}\sum_{j=i+1}^{Q} H(M_i, M_j)}{Q(Q-1)/2}\Big) \quad (6)$$

Using the definitions above, we can transform training objects $\mathcal{D}$ to a new feature space $\mathcal{D}' = \{\psi(X_1), \psi(X_2), \cdots, \psi(X_i)\}$, where $\psi(X_i) = \{\psi_1(X_i), \psi_2(X_i), \cdots, \psi_q(X_i)\}(1 \leq q \leq Q)$. Taking the $\mathcal{D}'$ as input, the training data is recursively partitioned into smaller subsets while moving from tree root nodes down to leaf nodes. Such recursion continues until all leaf nodes are pure (all the objects associated to a node belong to the same classes). For each node, we randomly select the i-th feature f_i from the bag of each training example as well as a threshold θ_i which are used to determine a suitable child node that example belongs to. We repeat the random selection for t times and compute the score of node splitting to determine the best training example partition. In this way, a randomized clustering tree, whose structure and predictions are independent from the labels, is constructed. In the procedure, a boolean test τ is used to partition training examples into left child node or right one. That is, the example assigned to left child if $f_i \geq \theta_i$ and is assigned to right child otherwise. Here, we denote $\mathcal{D}' = \mathcal{D}'_l \cup \mathcal{D}'_r$ and $\mathcal{D}'_l \cap \mathcal{D}'_r = \phi$ where $\mathcal{D}'$ is a set of examples for parent node, $\mathcal{D}'_l$ and $\mathcal{D}'_r$ represents the examples for left child and right child nodes, respectively. We compute the node splitting score using the Eq. 7 and Eq. 8, where $|\mathcal{D}'|$ is the total number of examples in set $\mathcal{D}'$ and Hc denotes the entropy of the class distribution of the current node examples. $Y = 1, 2, \cdots, l$ is a set of l possible class labels in the total data set and $\mathcal{D}'_y$ is the examples in set $\mathcal{D}'$ annotated with the y-th label.

$$Sc(\mathcal{D}', \mathcal{D}'_l, \mathcal{D}'_r) = Hc(\mathcal{D}') - \sum_{p\in\{l,r\}} \frac{|\mathcal{D}'_p|}{|\mathcal{D}'|} Hc(\mathcal{D}'_p) \quad (7)$$

Algorithm 1 MIMLRC-Tree($\mathcal{D}$)

Input: A training data set $\mathcal{D}$
Output: A MIML randomized clustering tree

if root = true **then**
  Construct a new feature space $\mathcal{D}'$
else
  if stopsplitting($\mathcal{D}'$) = true **then**
    create a leaf node
    return
  else
    $iterations = 0$
    compute the reuse label set b from parent node
    repeat
      $iterations \leftarrow iterations + 1$
      select an attribute f_i from each training example randomly(with the same random index i in each iteration)
      select a threshold θ randomly
      split the examples in $\mathcal{D}'$ into two children $\mathcal{D}'_l$ and $\mathcal{D}'_r$ according to f_i and θ:
      $\mathcal{D}'_l \leftarrow \{d \in \mathcal{D}' | f_i < \theta\}$
      $\mathcal{D}'_r \leftarrow \{d \in \mathcal{D}' | f_i \geq \theta\}$
      calculate $score \leftarrow Sc(\mathcal{D}', \mathcal{D}'_l, \mathcal{D}'_r)$
    until $iterations \geq I_{max}$
    select f_i^*, θ^* with the highest score
    calculate the relevant label set $\hat{b}$ for the current node
    combine b and $\hat{b}$ to the final prediction b^*
    create a decision node by a group of selected parameters f_i^*, θ^*, b^*
    return MIMLRC-Tree($\mathcal{D}'_l$) and MIMLRC-Tree($\mathcal{D}'_r$)
  end if
end if

$$Hc(\mathcal{D}') = -\sum_{y \in \mathcal{Y}} \frac{|\mathcal{D}'_y|}{|\mathcal{D}'|} \log_2 \frac{|\mathcal{D}'_y|}{|\mathcal{D}'|} \tag{8}$$

At each node, a set of relevant labels are exploited by computing its purity over the classes. Formally, given a set of examples and labels assigned to a node $\mathcal{D} = \{(x_i, Y_i) | 1 \leq i \leq m\}$, the label purity of a node is represented as a vector $p = [p(1), \cdots, p(q)]$), which is computed using Eq. 9

$$p(j) = \frac{1}{|\mathcal{D}|} \sum_{x_i \in D} Y_i(j) \tag{9}$$

where $p(j)$ is in the interval [0, 1] and $Y_i(j) = 1$ if the example X_i is assigned to the j-th label and $Y_i(j) = 0$ otherwise. The classes with purity score larger than a predefined threshold $\lambda(\lambda \in [0, 1])$ are regarded as a relevant label vector $b = [b(1), \cdots, b(q)])$. Formally, we have:

$$b(j) = \begin{cases} 1, & if\ p(j) \geq \lambda, \\ 0, & otherwise. \end{cases} \tag{10}$$

In addition, we propose a label transfer mechanism to transfer the obtained relevant label vector to its lower-layer nodes. In this mechanism, the identified relevant label vector from the parent node at higher level is transferred to its child node as an additional indicator incorporating in the relevant label vector of the child node to get final relevant labels. Specifically, suppose the relevant labels of the parent node is denoted as $\hat{b}$, the non-zero elements (i.e., the relevant labels) in $\hat{b}$ can be reused in the child nodes, and the zero elements (i.e., the non-relevant labels) can be computed using Eq. (7) and Eq. (10). The results are stored as another relevant label vector $\hat{b}$. The final output $b^* = [b^*(1), \cdots, b^*(q)]$ is given as

$$H^d(X_a, X_b) = \begin{cases} \dfrac{\sum_{x^{(a)} \in X_a} MinDist(x^{(a)}, X_b) + \sum_{x^{(b)} \in X_b} MinDist(x^{(b)}, X_a)}{|X_a| + |X_b|}, & d = average; \\ \max\left\{\max_{x^{(a)} \in X_a} MinDist(x^{(a)}, X_b), \max_{x^{(b)} \in X_b} MinDist(x^{(b)}, X_a)\right\}, & d = \max; \\ \min_{x^{(a)} \in X_a} MinDist(x^{(a)}, X_b), & d = min. \end{cases} \tag{3}$$

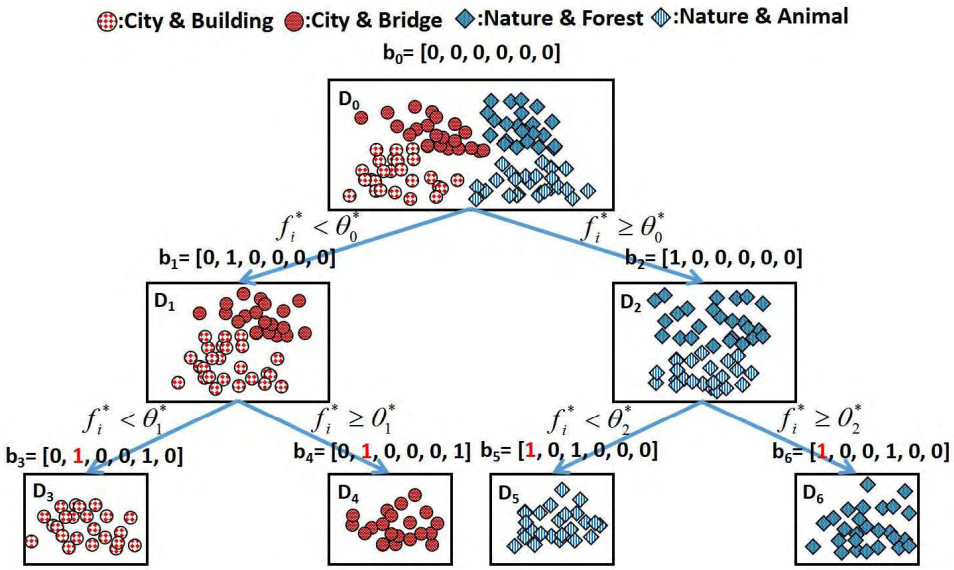


FIGURE 1. An example of randomized clustering tree construction.

follows:

$$b^*(j) = \begin{cases} 1, & if\ b(j) = 1\ or\ \hat{b}(j) = 1, \\ 0, & otherwise. \end{cases} \tag{11}$$

Fig. 1 shows an example to illustrate the randomized clustering tree construction. At the top level, all objects are assigned to the root node D_0, and then are partitioned into two child nodes D_1 and D_2 through the boolean test $\tau = f > \theta$. The objects in left node D_1 are all annotated with the same label *city*, while the objects on right node D_2 are associated with label *nature*. After that, the nodes D_1 and D_2 are further partitioned into two nodes $\{D_3, D_4\}$ and $\{D_5, D_6\}$ at next layer in the same way, respectively. The red labels assigned to nodes D_3-D_6 are transferred from the parent nodes D_1 and D_2. The final predictions are determined on leaf nodes and we define non-leaf nodes as composite nodes, e.g., the nodes D_1 and D_2 above.

B. RANDOMIZED CLUSTERING FORESTS

To reduce the class label variance for a given single randomized clustering tree aiming at improving the prediction performance, we propose the MIMLRC-Forest algorithm (as described in Table 1) to build an ensemble of tree classifiers. The popular bootstrap replica scheme [47] is used to generate training data so that diverse tree classifiers in the ensemble can be built. The ensemble building procedure involves a random selection of learning examples and a random clustering. Considering the trees diversity, a best node splitting strategy with score ranking is used to generate trees partitions to yield a relative strong tree.

According to the generalization error bound which is defined by Breiman [48], our method has a better error bound. MIMLRC-Forest outputs a confidence vector, denoted as $\hat{w} = [\hat{w}(1), \cdots, \hat{w}(q)] \in \mathbb{R}^q$ for MIML learning, where $\hat{w}(j)$ is the confidence for the j-th class. Given a test example $\hat{X}$, we navigate $\hat{X}$ across all the trees and find its predictions on leaves where $\hat{X}$ ends. Suppose the predictions on these leaves are denoted as $\hat{b}_1^*, \cdots, \hat{b}_K^*$ (where K denotes the number of trees), an ensemble confidence $\hat{w} = [\hat{w}(1), \cdots, \hat{w}(q)]$, representing the importance of the classes to $\hat{X}$ can be conputed as follows:

$$\hat{w}(j) = \frac{1}{K} \sum_{k=1}^{K} b_k^*(j) \tag{12}$$

MIMLRC-Forest also outputs a binary vector, denoted as $\hat{Y} = [\hat{Y}(1), \cdots, \hat{Y}(q)]$ where $\hat{Y}(j) = 1$ represents the j-th label as a relevant class for $\hat{x}$ and $\hat{Y}(j) = 0$ otherwise. Specifically, a threshold function f is used to separate the relevant and irrelevant classes according to the $\hat{w}$ value. There are many ways for selecting a threshold value for MIML prediction, e.g., a threshold $t = 0.5$ commonly used for its simplification and low computational complexity. In this paper, we propose a *maximum drop thresholding* method to optimize the threshold for multi-instance multi-label learning.

Intuitively, a multi-instance multi-label example is assigned with a set of relevant labels and it is expected that the confidences of the relevant labels are much higher than those of other labels. Given the confidence $\hat{w}$ of the class labels for an example $\hat{X}$, the labels are sorted by $\hat{w}$ to identify two classes with largest drops based on their confidence values. The median value of these two classes are then calculated as a threshold to establish a separation that separates relevant and irrelevant labels to $\hat{X}$ where the relevant labels are labels with confidences higher than the threshold and the irrelevant labels are the rest. As shown in Table 2, in the example, the largest drop in confidences of the sorted classes is observed between 1 and 0.25 and their

TABLE 1. The pseudo-code of the MIMLRC-Forest algorithm to build an ensemble of tree.

MIMLRC-Forest
Training Phase
Input: A training data-set $\mathcal{D}$, the number of ensemble classifiers K
Output: A set of classifiers $\mathcal{F}$
1: $\mathcal{F}$ be set to ϕ
2: **for** $i = 1$ to K **do**
3: using bootstrap($\mathcal{D}$) to select the sub-set of $\mathcal{D}_i$
4: using MIMLRC-Tree($\mathcal{D}$) to train tree classifier T_i
5: $\mathcal{F} = \mathcal{F} \cup T_i$
6: **return** $\mathcal{F}$

Prediction Phase
1: For a given sample $\hat{x}$, let $b_1^*, \cdots, b_K^*$ be the predicting labels, computing the confidence for each category label, $\hat{w}(j)$, via following method:

$$\hat{w}(j) = \frac{1}{K} \sum_{k=1}^{K} b_k^*(j)$$

2: Assign sample $\hat{x}$ to the category label with the confidences larger than a predefined threshold

TABLE 2. An example of MIMLRC-Forest prediction procedure, where $K = 4$, $q = 7$ and the threshold is $t = 0.625$ for the function $f_t(\hat{w})$.

$\hat{b}_1^* =$	[1	0	1	0	0	1	0]
$\hat{b}_2^* =$	[1	0	0	1	0	1	0]
$\hat{b}_3^* =$	[1	0	0	0	0	1	0]
$\hat{b}_4^* =$	[1	0	0	0	0	1	0]
$\hat{w} =$	[1	0	0.25	0.25	0	1	0]
$\hat{Y} = f_{t=0.625}(\hat{w}) =$	[1	0	0	0	0	1	0]

median value 0.625 is used as a threshold for further multi-instance multi-label classification. We find that this threshold optimization method has better classification performance against using an arbitrary threshold for all the examples and data sets.

IV. RESULTS

In this section, we compare the performance of our proposed MIMLRC-Forest algorithm with five popular MIML learning algorithms as baselines on real-world genome-wide protein data sets. The baseline methods are MIMLNN, MIMLKNN, MIML-SVM, EnMIMLNN and M3MIML and the experimental results present that our algorithm outperforms all the baselines.

A. DATA SETS

The experiments use five genome-wide protein data sets represented as multi-instance multi-label data pattern [11]. These real-world protein data sets are mainly from three biological domains containing two Archaea genomes (Haloarcula marismortui and Pyrococcus furiosus), two Bacteria genomes (Azotobacte vinelandii and Geobacter sulfurreducens) and a Eukaryota genome(Caenorhabditis elegans) [49]. In the data sets, each multi-domain protein was manually annotated with a set of function labels, where each domain is represented as an 216-dimensional independent bags and all the annotation terms are based on one of the most famous and widely used Molecular function called Gene Ontology Consortium [50]. More details about the protein data sets are shown in Table 3. Specifically, the data sets about Archaea and Bacteria biological domains are more sparse as each example contains 3.07 to 3.20 instances and no more than 4.5 labels only. The Caenorhabditis elegans data set in the instance number and label number per example goes up to 3.64 and 5.34 with the max values 115 and 107, respectively.

B. BASELINE METHODS

To evaluate the effectiveness of our method, we use five widely used MIML learning algorithms, i.e., MIMLSVM, MIMLkNN, MIMLNN, M3MIML and EnMIMLNN, as baselines. For MIMLSVM algorithm, it applies a degeneration strategy to simplify the MIML learning task to a ML learning problem. All instances are partitioned into different clusters according to the similarity of each pair instances. A feature space then is built by the distance matrix between the examples and the medoids of clusters learned from a supporting vector machine (SVM) classifier. There are two parameters, Gaussian kernel parameter and the value of the cost in MIMLSVM algorithm. We set them to the optimal values 0.2 and 1 in the experiment, respectively. As for MIMLkNN, it degenerates MIML learning problem to ML problem by using another well-known method k-nearest neighbor algorithm to explore the correlation of an example among its neighbors. The number of the nearest neighbors k is set to default value 10. The method MIMLNN is trained with two layer architectures. It exploits the medoids of clusters for each class labels by invoking k-Medoids algorithm in first layer. The basic functions related to the medoids and examples are determined in second layer. Finally, it utilizes a weight matrix to solve MIML learning problem. The regularization parameter in the method is set 1. For optimizing the MIMLNN algorithm, the EnMIMLNN applies a novel Hausdroff distance to compute the medoids of each cluster in first layer. It implements the basic functions by radial basis function neural networks rather than back-propagation neural networks in MIMLNN in second layer. We set the scaling

TABLE 3. The characteristics of the five genome-wide protein data sets.

Data sets		examples	instances	classes	Instances per bag		Labels per example	
					max	mean	max	mean
Archaea	Haloarcula marismortui	304	950	234	7	3.13	21	3.25
	Pyrococcus furiosus	425	1317	321	8	3.10	65	4.48
Bacteria	Azotobacter vinelandii	407	1251	340	8	3.07	69	4.0
	Geobacter sulfurreducens	379	1214	320	8	3.20	46	3.14
Eukaryota	Caenorhabditis elegans	521	1894	940	115	3.64	107	5.34

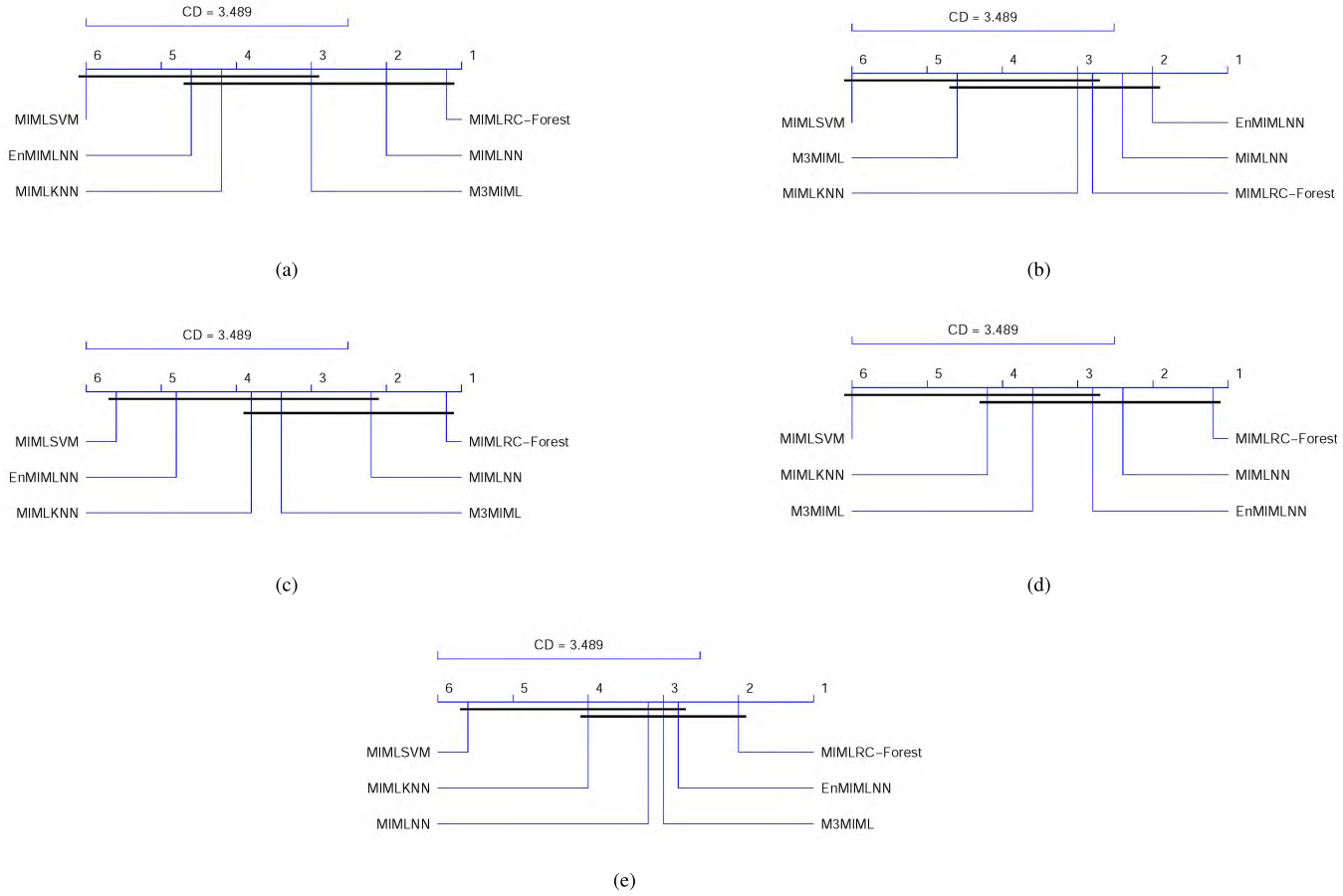


FIGURE 2. Graphical presentation of results from the Neminyi post hoc test at 0.05 significance level for the ranking-based measures. (a) Ranking-loss. (b) Hamming-loss. (c) Coverage. (d) One-error (e) Average precision.

parameter $\mu = 0.4$ and the fraction parameter $\alpha = 0.1$ in EnMIMLNN algorithm. The last baseline M3MIML transforms a MIML learning task into a quadratic programming problem and implements its dual form by exploiting the correlations between instances and labels. The parameters γ and λ are set to 1 and 10^{-4}, respectively.

C. EVALUATION METRICS

In the comparison, we use hamming loss, one-error, coverage, ranking loss, and average precision as evaluation metrics. In particular, accuracy and kappa-error will not be used as evaluation metrics. Because the accuracy is a key evaluation metric for single label prediction, the kappa coefficient is a different statistic which always used to evaluate average on multi-label prediction. The meaning of accuracy and kappa coefficient are similar as average precision. For the detailed definitions of these metrics, one can refer to [5]. As multiple algorithms and diverse data sets are used in the comparison, the effectiveness of the algorithms across different data sets is evaluated by comparing their performance in a two-step statistical test procedure (the corrected Friedman test and the post-hoc Nemenyi test) [51]. The Friedman test is used for multiple hypotheses test without any parameter. The algorithms are ranked (each row in each section of the table) in a descending order according to their performance in each data set separately and the one which shows best performance for each data set is ranked first. It also computes the average rank of each algorithm across all the datasets (each column in each section of the table). Based on the average ranks of the algorithms, the Nemenyi post-hoc test detects the algorithms which are significantly different from each other. Performance of two algorithms is viewed as significantly

TABLE 4. Performance of the MIML learning algorithms on hamming loss ↓.

Dataset	MIMLSVM	MIMLNN	MIMLKNN	M3MIML	EnMIMLNN	MIMLRC-Forest
Haloarcula marismortui	0.0175(6)	0.0134(4)	0.0131(3)	0.01467(5)	**0.0103(1)**	0.0123(2)
Pyrococcus furiosus	0.0146(6)	0.0117(3)	0.0119(4)	0.0133(5)	**0.0102(1)**	0.0111(2)
Azotobacter vinelandii	0.0183(6)	0.0151(3)	0.0152(4)	0.0166(5)	**0.0087(1)**	0.0146(2)
Geobacter sulfurreducens	0.0137(6)	**0.0105(1)**	0.0106(2)	0.0112(4)	0.0123(5)	0.0115(3)
Caenorhabditis elegans	0.0054(6)	**0.0042(1)**	0.0043(2)	0.0045(4)	0.0044(3)	0.0053(5)
Avg. rank	6	2.4	3	4.6	2	2.8

TABLE 5. Performance of the MIML learning algorithms on ranking loss ↓.

Dataset	MIMLSVM	MIMLNN	MIMLKNN	M3MIML	EnMIMLNN	MIMLRC-Forest
Haloarcula marismortui	0.9957(6)	0.2537(3)	0.2754(5)	0.2235(2)	0.2570(4)	**0.2158(1)**
Pyrococcus furiosus	0.9942(6)	0.3126(2)	0.3654(4)	0.3480(3)	0.3823(5)	**0.2837(1)**
Azotobacter vinelandii	0.9948(6)	**0.2967(1)**	0.3539(3)	0.3549(4)	0.3630(5)	0.3426(2)
Geobacter sulfurreducens	0.9938(6)	0.3537(2)	0.3930(4)	0.3665(3)	0.4555(5)	**0.2544(1)**
Caenorhabditis elegans	0.9923(6)	0.3226(2)	0.3369(5)	0.3289(3)	0.3353(4)	**0.2119(1)**
Avg. rank	6	2	4.2	3	4.6	1.2

TABLE 6. Performance of the MIML learning algorithms on coverage ↓.

Dataset	MIMLSVM	MIMLNN	MIMLKNN	M3MIML	EnMIMLNN	MIMLRC-Forest
Haloarcula marismortui	126.4333(5)	109.6000(3)	126.7000(6)	106.8667(2)	121.6333(4)	**102.3000(1)**
Pyrococcus furiosus	204.5714(6)	148.5952(2)	184.4286(4)	173.2143(3)	187.2857(5)	**129.1544(1)**
Azotobacter vinelandii	177.9512(6)	**150.4878(1)**	165.3902(3)	169.4634(5)	167.6829(4)	164.0500(2)
Geobacter sulfurreducens	184.2368(5)	162.8947(2)	182.5921(4)	169.3553(3)	226.1053(6)	**150.0541(1)**
Caenorhabditis elegans	566.4808(6)	409.1442(3)	388.2212(2)	433.9519(4)	437.2788(5)	**377.6400(1)**
Avg. rank	5.6	2.2	3.8	3.4	4.8	1.2

TABLE 7. Performance of the MIML learning algorithms on one error ↓.

Dataset	MIMLSVM	MIMLNN	MIMLKNN	M3MIML	EnMIMLNN	MIMLRC-Forest
Haloarcula marismortui	0.9667(6)	**0.5333(1)**	0.6333(3)	0.6333(3)	0.6333(3)	0.6000(2)
Pyrococcus furiosus	0.9524(6)	0.7380(3)	0.7857(5)	0.7381(4)	0.6667(1)	**0.6667(1)**
Azotobacter vinelandii	1.0000(6)	0.6341(2)	0.6585(3)	0.7317(5)	0.6829(4)	**0.6250(1)**
Geobacter sulfurreducens	1.0000(6)	0.6842(2)	0.7500(5)	0.7237(4)	0.6579(3)	**0.5405(1)**
Caenorhabditis elegans	1.0000(6)	0.7212(4)	0.7788(5)	0.6154(2)	0.7115(3)	**0.5800(1)**
Avg. rank	6	2.4	4.2	3.6	2.8	1.2

TABLE 8. Performance of the MIML learning algorithms on average precision ↑.

Dataset	MIMLSVM	MIMLNN	MIMLKNN	M3MIML	EnMIMLNN	MIMLRC-Forest
Haloarcula marismortui	0.0944(6)	0.3794(4)	0.3538(5)	**0.4232(1)**	0.4159(2)	0.3846(3)
Pyrococcus furiosus	0.0598(6)	0.2469(3)	0.2114(5)	0.2354(4)	0.2578(2)	**0.2660(1)**
Azotobacter vinelandii	0.0645(6)	0.3622(3)	**0.3702(1)**	0.3194(5)	0.3667(2)	0.3323(4)
Geobacter sulfurreducens	0.05756(6)	0.3047(2)	0.2606(4)	0.2778(3)	0.2327(5)	**0.3263(1)**
Caenorhabditis elegans	0.0346(6)	0.2355(4)	0.2105(5)	0.3136(2)	0.2671(3)	**0.3643(1)**
Avg. rank	5.6	3.2	4	3	2.8	2

different if their average rank difference is larger than a critical distance (CD) which is computed in terms of the number of algorithms, the number of data sets and a given significance level p. The results of the Nemenyi post-hoc test can be visualized with diagrams [51] as shown in Fig. 2, where the critical distance is 2.849 and the significant level is $p = 0.05$. For each evaluation metric, the average ranks of the algorithms are displayed on the horizontal axis in the way that the algorithm ranked best (worst) is shown at the rightmost(leftmost) side of the diagram. Algorithms that are not significantly different are connected by bold horizontal lines.

D. PARAMETER TUNING

Four parameters are set for our method, which include the fraction parameter α and the scaling parameter μ used to transform the input data to a new feature space, the number of trees K and the purity threshold ν which is used to construct the tree classifiers. We set $\alpha = 0.1$, $\mu = 0.4$, $K = 65$, and we use 3-fold cross validation on the training set to automatically obtain the optimized value of parameter ν.

In this experiment, we test the impacts of the number of trees (K) in our proposed algorithm MIMLRC-Forest. We vary the number of trees from 5 to 65 with an interval 10, and the variation tendency of all the evaluation metrics on different data sets are shown in Fig. 3. Intuitively, except average precision, the values of the other four evaluation metrics are approximately declining with the increasing number of trees (K). More specifically, the values of each metric trend to be stable when the number of trees reaching 45, on the other side, a fulfilling performance can not be guaranteed with a small number of trees, i.e., $K = 5$.

E. PERFORMANCE COMPARISON

In the comparison, the performances of all baselines are evaluated on five different evaluation metrics including ranking loss, hamming loss, one error, coverage and average precision. The performances of all the methods on protein function

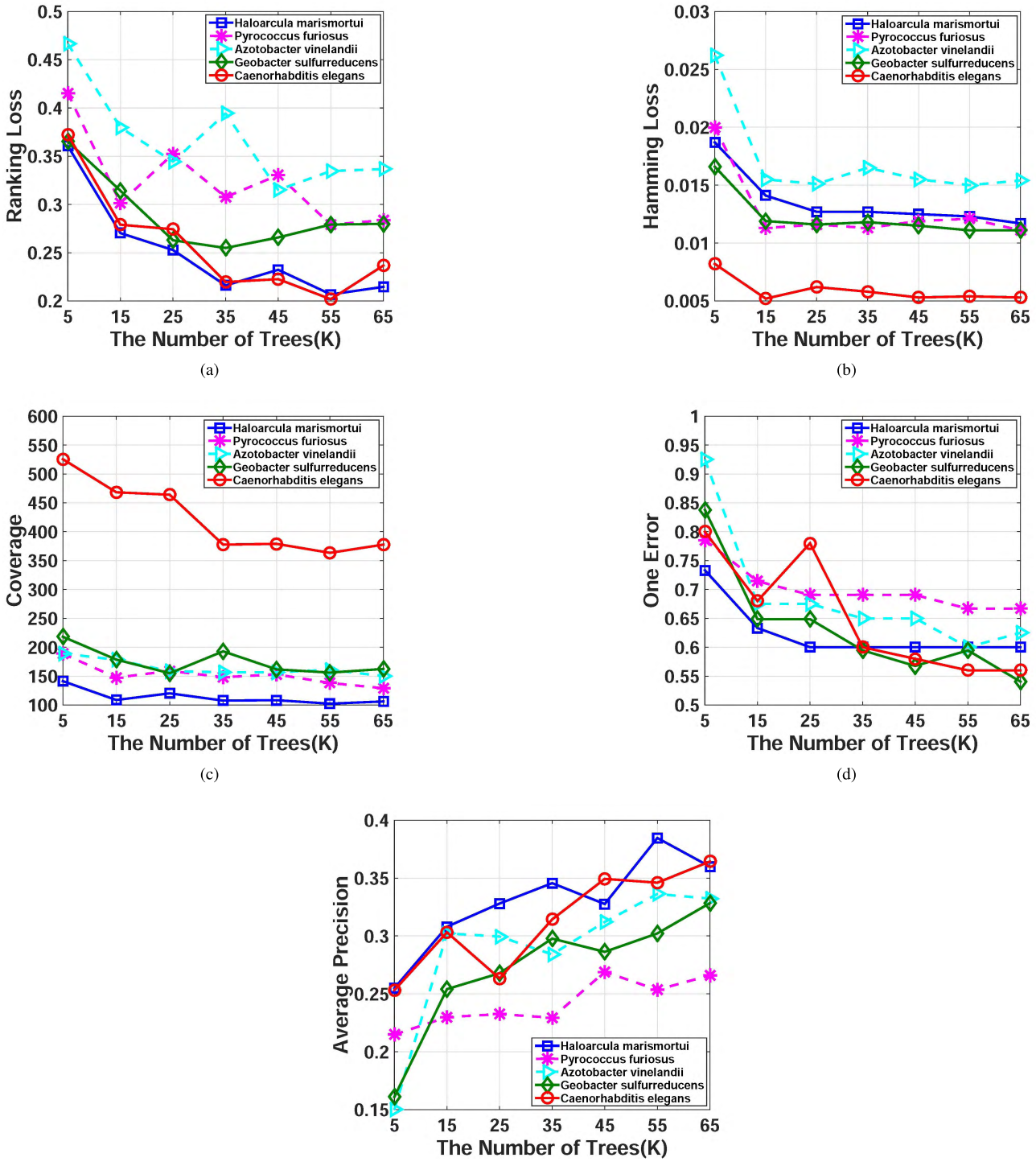


FIGURE 3. The impacts of MIMLRC-Forest algorithm for each evaluation metric on protein data sets with a varying parameter K. (a) Ranking-loss. (b) Hamming-loss. (c) Coverage. (d) One-error (e) Average precision.

prediction task with respect to the metrics are shown in Fig. 2. A list of tables from Table 4 to Table 8 present specific experimental results of the performances of all the methods. In the tables, the up arrow ↑ (down arrow ↓) indicates that the value of this particular evaluation metric is more competitive with a larger (smaller) value. The average ranks of the method performances are listed in the last row of each table. In the Fig. 2, the numbers in brackets denote he performance ranks of the algorithms according to one particular evaluation metric and the numbers in boldface show the best ranking algorithm.

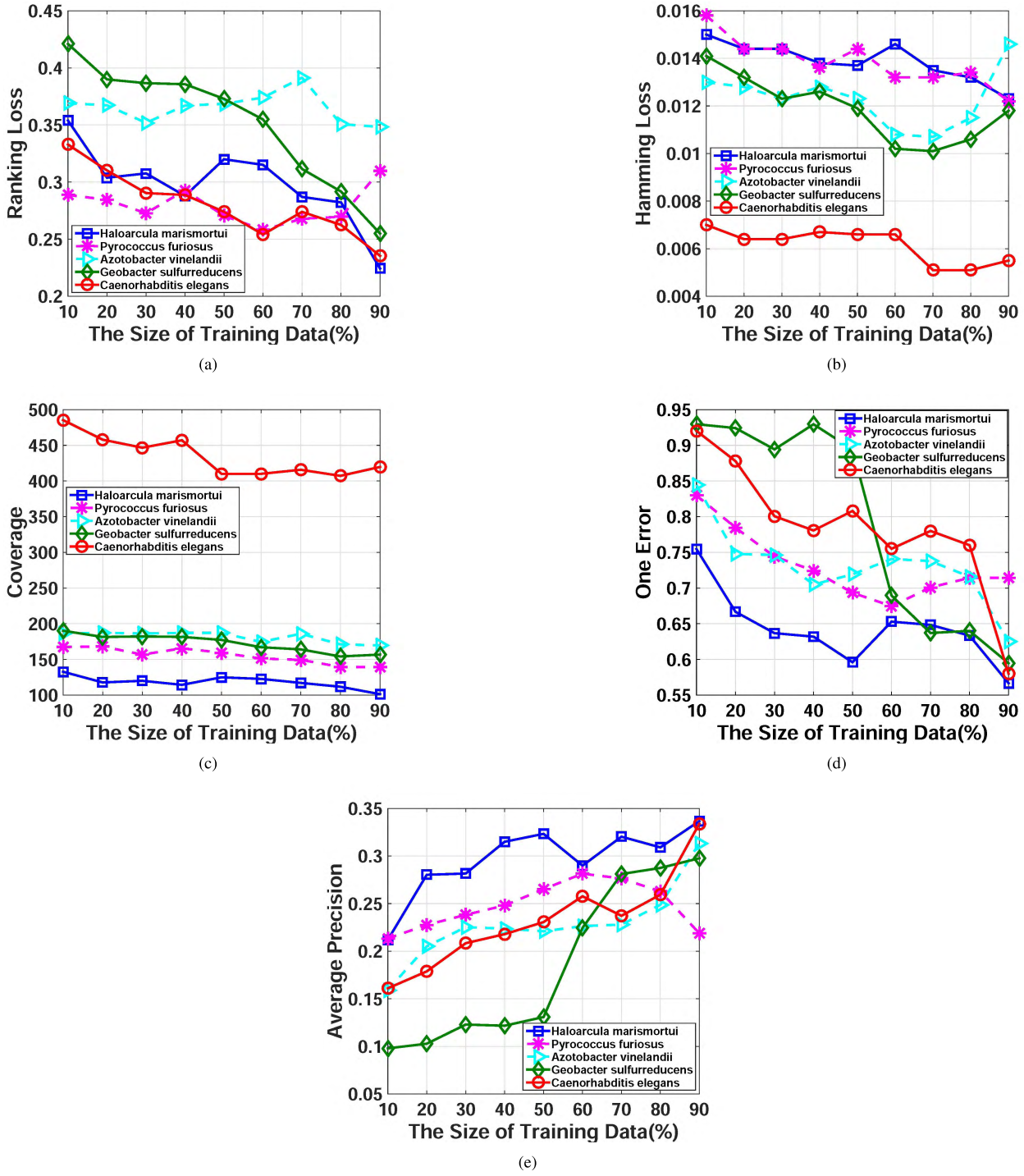


FIGURE 4. The impacts of MIMLRC-Forest algorithm for each evaluation metric on protein data sets with different sizes training data. (a) Ranking-loss. (b) Hamming-loss. (c) Coverage. (d) One-error (e) Average precision.

All the experimental results present:

- We observe that our proposed algorithm MIMLRC-Forest highly outperforms the five baseline methods using all the ranking-loss, coverage, one-error and average precision evaluation metrics. The average algorithm performance rank is: MIMLRC-Forest $\succ$ MIMLNN $\succ$ M3MIML $\succ$ EnMIMLNN $\succ$ MIMLkNN $\succ$ MIMLSVM. Using ranking-loss and coverage evaluation metrics, the performance of MIMLRC-Forest is higher than MIMLSVM, MIMLkNN and EnMIMLNN. As for average precision and one-error evaluation metrics, the MIMLRC-Forest presents much better performance compared with MIMLSVM and MIMLkNN.

- Using the hamming-loss metric, MIMLRC-Forest fails to achieve the best performance, where it is ranked as 3rd on average. Both MIMLNN and EnMIMLNN apply two layers architecture to solve the MIML learning task and obtain better performance. This result proves a recent research [11] about EnMIMLNN reporting its significant effectiveness with hamming-loss metric on protein function prediction task.
- In general, the average performance ranks of the compared MIMLRC-Forest, MIMLSVM, MIMLNN, MIML*k*NN and EnMIMLNN based on these five evaluation metrics are 1.68, 5.84, 2.44, 3.84 and 3.52, respectively. This implies that MIMLSVM method is evidently unavailable on protein function prediction problem while MIMLRC-Forest and MIMLNN approaches are more effective on protein data sets. Although MIMLRC-Forest is the optimal choice for protein function prediction task, MIMLNN steadily achieves satisfied performance on almost all of the evaluation metrics.

In our experiments, we also study the performance of MIMLRC-Forest algorithm with different sizes of training set. We randomly pick up 10% to 90% of the data set as training data with an interval 10%, and the remaining data set is used as test data. Fig. 4 shows the performances of MIMLRC-Forest with different sizes of training data on protein data sets. From Fig 4(d) and Fig. 4(e), with the increasing size of training data, MIMLRC-Forest achieves better performances in terms of one error and average precision. In the contrast, from the perspective of Fig. 4(a) to 4(c), the performances using ranking loss, hamming loss and coverage are slightly influenced by the different size of training data. The only exception is on the Caenorhabditis elegans data set using the ranking loss.

V. CONCLUSION

We propose a novel ensemble algorithm called Multi-Instance Multi-Label Randomized Clustering Forest (MIMLRC-Forest) in the paper to build a set of randomized clustering trees for protein function prediction task. To exploit the label dependency, a label transfer mechanism is also developed to find relevant labels at each node of the tree hierarchically and seek a decision path to do the prediction. Based on the experiments on real-world protein data sets, the results shows that the proposed MIMLRC-Forest method achieves better performance against five state-of-the-art MIML learning algorithms (MIMLSVM, MIMLNN, MIML*k*NN, M3MIML and EnMIMLNN) using ranking loss, hamming loss, coverage, one-error and average precision as metrics, demonstrating the effectiveness of our method in protein function prediction. Using the proposed MIMLRC-Forest algorithm, researchers can perform the structural analysis, functional annotations which can guide the design of biological experiments and identify functional units or domains of proteins. Moreover, the proposed MIMLRC-Forest algorithm can provide a target for genetic manipulation and a reliable basis for designing new proteins or modifying existing ones. For future work, it will be interesting to explore the effect of the fraction parameter α and the scaling parameter μ. Also, we will model the relations among proteins and considering label relations as input to improve the performance of MIMLRC-Forest.

ACKNOWLEDGEMENT

(Hong Tang, Yuanyuan Wang, and Shaomin Tang contributed equally to this work.)

REFERENCES

[1] M.-L. Zhang and K. Zhang, "Multi-label learning by exploiting label dependency," in *Proc. 16th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2010, pp. 999–1008.
[2] C. Chothia, "One thousand families for the molecular biologist," *Nature*, vol. 357, pp. 543–544, Jun. 1992.
[3] G. Apic, J. Gough, and S. A. Teichmann, "Domain combinations in archaeal, eubacterial and eukaryotic proteomes," *J. Mol. Biol.*, vol. 310, no. 2, pp. 311–325, 2001.
[4] Z.-H. Zhou and M.-L. Zhang, "Multi-instance multi-label learning with application to scene classification," in *Proc. Adv. Neural Inf. Process. Syst.*, 2006, pp. 1609–1616.
[5] Z.-H. Zhou, M.-L. Zhang, S.-J. Huang, and Y.-F. Li, "Multi-instance multi-label learning," *Artif. Intell.*, vol. 176, pp. 2291–2320, Jan. 2012.
[6] K. Yan, Z. Li, and C. Zhang, "A new multi-instance multi-label learning approach for image and text classification," *Multimedia Tools Appl.*, vol. 75, no. 13, pp. 7875–7890, 2016.
[7] T. Sumathi, C. L. Devasena, R. Revathi, S. Priya, and M. Hemalatha, "Automatic image annotation and retrieval using multi-instance multi-label learning," *Bonfring Int. J. Adv. Image Process.*, vol. 176, pp. 1–5, Dec. 2011.
[8] Z.-J. Zha, X.-S. Hua, T. Mei, J. Wang, G.-J. Qi, and Z. Wang, "Joint multi-label multi-instance learning for image classification," in *Proc. Comput. Vis. Pattern Recognit.*, Jun. 2008, pp. 1–8.
[9] A. Elisseeff and J. Weston, "A kernel method for multi-labelled classification," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 2002, pp. 681–687.
[10] A. Clare and R. D. King, "Knowledge discovery in multi-label phenotype data," in *Principles of Data Mining and Knowledge Discovery*. 2001, pp. 42–53.
[11] J.-S. Wu, S.-J. Huang, and Z.-H. Zhou, "Genome-wide protein function prediction through multi-instance multi-label learning," *IEEE/ACM Trans. Comput. Biol. Bioinf.*, vol. 11, no. 5, pp. 891–902, Sep./Oct. 2014.
[12] N. Alaydie, C. K. Reddy, and F. Fotouhi, "Exploiting label dependency for hierarchical multi-label classification," in *Proc. Pacific-Asia Conf. Knowl. Discovery Data Mining*, 2012, pp. 294–305.
[13] L. Cai and T. Hofmann, "Hierarchical document categorization with support vector machines," in *Proc. 13th ACM Int. Conf. Inf. Knowl. Manage.*, 2004, pp. 78–87.
[14] N. Cesa-Bianchi, C. Gentile, and L. Zaniboni, "Hierarchical classification: Combining Bayes with SVM," in *Proc. 23rd Int. Conf. Mach. Learn.*, 2006, pp. 177–184.
[15] L. Sun, S. Ji, and J. Ye, "Hypergraph spectral learning for multi-label classification," in *Proc. 14th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2008, pp. 668–676.
[16] S.-J. Huang, Y. Yu, and Z. H. Zhou, "Multi-label hypothesis reus," in *Proc. 18th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2012, pp. 525–533.
[17] Q. Wu, M. K. Ng, and Y. Ye, "Markov-miml: A Markov chain-based multi-instance multi-label learning algorithm," *Knowl. Inf. Syst.*, vol. 37, pp. 83–104, Oct. 2013.
[18] M.-L. Zhang and Z.-H. Zhou, "M3miml: A maximum margin method for multi-instance multi-label learning," in *Proc. 8th IEEE Int. Conf. Data Mining*, Dec. 2008, pp. 688–697.
[19] A. Li, Q. Zang, D. Sun, and M. Wang, "A text feature-based approach for literature mining of lncRNA–protein interactions," *Neurocomputing*, vol. 206, pp. 73–80, Sep. 2016.
[20] S. Andrews, I. Tsochantaridis, and T. Hofmann, "Support vector machines for multiple-instance learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2003, pp. 577–584.
[21] M. Ge, A. Li, and M. Wang, "A bipartite network-based method for prediction of long non-coding RNA–protein interactions," *Genomics, Proteomics Bioinformatics*, vol. 14, no. 1, pp. 62–71, 2016.

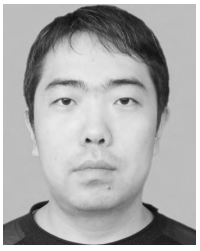
[22] M.-L. Zhang, "A k-nearest neighbor based multi-instance multi-label learning algorithm," in *Proc. 22nd IEEE Int. Conf. Tools Artif. Intell.*, Oct. 2010, pp. 207–212.
[23] Y.-X. Li, S. Ji, S. Kumar, J. Ye, and Z.-H. Zhou, "Drosophila gene expression pattern annotation through multi-instance multi-label learning," *IEEE/ACM Trans. Comput. Biol. Bioinf.*, vol. 9, no. 1, pp. 98–112, Jan./Feb. 2011.
[24] M. Yang *et al.*, "A multi-instance multi-label learning approach for protein domain annotation," in *Proc. Int. Conf. Intell. Comput.* Springer, 2014, pp. 104–111.
[25] 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, 2004.
[26] N. Nguyen, "A new SVM approach to multi-instance multi-label learning," in *Proc. IEEE 13th Int. Conf. Data Mining*, Dec. 2010, pp. 384–392.
[27] X.-S. Xu, Y. Jiang, X. Xue, and Z.-H. Zhou, "Semi-supervised multi-instance multi-label learning for video annotation task," in *Proc. 20th ACM Int. Conf. Multimedia*, 2012, pp. 737–740.
[28] M.-L. Zhang and Z.-H. Zhou, "A k-nearest neighbor based algorithm for multi-label classification," in *Proc. IEEE Int. Conf. Granular Comput.*, Jul. 2005, pp. 718–721.
[29] X. Xu and E. Frank, "Logistic regression and boosting for labeled bags of instances," in *Proc. Pacific–Asia Conf. Knowl. Discovery Data Mining*, 2004, pp. 272–281.
[30] T. G. Dietterich, R. H. Lathrop, and T. Lozano-Pérez, "Solving the multiple instance problem with axis-parallel rectangles," *Artif. Intell.*, vol. 89, pp. 31–71, Jan. 1997.
[31] Y. Chen and J. Z. Wang, "Image categorization by learning and reasoning with regions," *J. Mach. Learn. Res.*, vol. 5, pp. 913–939, Aug. 2004.
[32] Q. Zhang and S. A. Goldman, "EM-DD: An improved multiple-instance learning technique," in *Proc. Adv. Neural Inf. Process. Syst.*, 2002, pp. 1073–1080.
[33] M.-L. Zhang and Z.-H. Zhou, "Multi-label learning by instance differentiation," in *Proc. Nat. Conf. Artif. Intell.*, 2007, pp. 669–674.
[34] J. Wu, X. Zhu, C. Zhang, and P. S. Yu, "Bag constrained structure pattern mining for multi-graph classification," *IEEE Trans. Knowl. Data Eng.*, vol. 26, no. 10, pp. 2382–2396, Oct. 2014.
[35] J. Wu, S. Pan, X. Zhu, and Z. Cai, "Boosting for multi-graph classification," *IEEE Trans. Cybern.*, vol. 45, no. 3, pp. 416–429, Mar. 2015.
[36] J. Wu, S. Pan, X. Zhu, C. Zhang, and P. S. Yu, "Multiple structure-view learning for graph classification," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 26, no. 7, pp. 3236–3251, Jul. 2018.
[37] J. Wu, S. Pan, X. Zhu, C. Zhang, and X. Wu, "Positive and unlabeled multi-graph learning," *IEEE Trans. Cybern.*, vol. 47, no. 4, pp. 818–829, Apr. 2017.
[38] J. Wu, S. Pan, X. Zhu, C. Zhang, and X. Wu, "Multi-instance learning with discriminative bag mapping," *IEEE Trans. Knowl. Data Eng.*, vol. 30, no. 6, pp. 1065–1080, Jun. 2018.
[39] G. Tsoumakas and I. Katakis, "Effective and efficient multilabel classification in domains with large number of labels," in *Proc. ECML/PKDD Workshop Mining Multidimensional Data*, 2008, pp. 53–59.
[40] S. Bengio, J. Weston, and D. Grangier, "Label embedding trees for large multi-class tasks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2010, pp. 163–171.
[41] J. Deng, S. Satheesh, A. C. Berg, and F.-F. Li, "Fast and balanced: Efficient label tree learning for large scale object recognition," in *Proc. NIPS*, vol. 2, 2011, pp. 567–575.
[42] G. Madjarov and D. Gjorgjevikj, "Hybrid decision tree architecture utilizing local SVMs for multi-label classification," in *Hybrid Artificial Intelligent Systems*. 2012, pp. 1–12.
[43] B. Fu, Z. Wang, R. Pan, G. Xu, and P. Dolog, "Learning tree structure of label dependency for multi-label learning," in *Advances in Knowledge Discovery and Data Mining*, 2012, pp. 159–170.
[44] W. Bi and J. T. Kwok, "Multi-label classification on tree-and dag-structured hierarchies," in *Proc. 28th Int. Conf. Mach. Learn. (ICML)*, 2011, pp. 17–24.
[45] H. Blockeel, L. De Raedt, and J. Ramon. (2000). "Top-down induction of clustering trees." [Online]. Available: https://arxiv.org/abs/cs/0011032
[46] C. Vens, J. Struyf, L. Schietgat, S. Džeroski, and H. Blockeel, "Decision trees for hierarchical multi-label classification," *Mach. Learn.*, vol. 73, no. 2, pp. 185–214, Nov. 2008.
[47] L. Breiman, "Bagging predictors," *Mach. Learn.*, vol. 24, no. 2, pp. 123–140, 1996.
[48] L. Breiman, "Random forests," *Mach. Learn.*, vol. 45, no. 1, pp. 5–32, 2001.
[49] C. R. Woese, O. Kandler, and M. L. Wheelis, "Towards a natural system of organisms: Proposal for the domains Archaea, Bacteria, and Eucarya," *Proc. Nat. Acad. Sci. USA*, vol. 87, pp. 4576–4579, Jun. 1990.
[50] R. Apweiler *et al.*, "UniProt: The universal protein knowledgebase," *Nucleic Acids Res.*, vol. 32, pp. D115–D119, Jan. 2004.
[51] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," *J. Mach. Learn. Res.*, vol. 7, pp. 1–30, Jan. 2006.
[52] W. Qingyao, T. Mingkui, S. Hengjie, C. Jian, and M. K. Ng, "ML-FOREST: A multi-label tree ensemble method for multi-label classification," *IEEE Trans. Knowl. Data Eng.*, vol. 28, no. 10, pp. 2665–2680, Oct. 2016.

HONG TANG received the B.Sc. degree from the Huazhong University of Science and Technology, China, in 1997, and the M.Sc. degree in optical engineering from Jinan University, China, in 2003. He is currently pursuing the Ph.D. degree with the School of Information Science and Technology, Sun Yat-sen University, Guangzhou, China. He is also a Network Architect with the Guangzhou Institute of China Telecom. His main research interests include traffic planning in IP backbone networks, and traffics scheduling in software-defined networks and network function virtualization.

YUANYUAN WANG received the B.Sc. degree from Jilin University, China, the M.Sc. degree from the Harbin Institute of Technology, China, and the Ph.D. degree from The Hong Kong Polytechnic University, Hong Kong, all in computer science. She is currently a Teacher with Shenzhen University, China. Her research interests include data mining, recommender systems, and education.

SHAOMIN TANG received the B.S. degree in process equipment and control engineering and the M.S. degree in materials processing engineering from the South China University of Technology, Guangzhou, China, in 2009 and 2012, respectively, where she is currently pursuing the Ph.D. degree in mechanical engineering. Her research interests include machinery condition monitoring and fault diagnosis, prognostics of RUL of rotating machinery, and machine learning in the field of prognostics.

DIANHUI CHU received the Ph.D. degree from the Department of Computer Science and Technology, Harbin Institute of Technology, Weihai, where he is currently a Professor. His research interests include service computing, service engineering, and software architecture.



CHUNSHAN LI received the B.S., M.S., and Ph.D. degrees in computer science from the Harbin Institute of Technology, Weihai, China, in 2005, 2009, and 2014, respectively, where he is currently an Assistant Professor. His research interests include data mining, text mining, and service computing.