Computational Biology and Chemistry 63 (2016) 30–40

Contents lists available at ScienceDirect

Computational Biology and Chemistry

journal homepage: www.elsevier.com/locate/compbiolchem

Research Article

Multi-instance multi-label distance metric learning for genome-wide protein function prediction

Yonghui Xu [a], Huaqing Min [b], Hengjie Song [b], Qingyao Wu [b,*]

[a] School of Computer Science and Engineering, South China University of Technology, Guangzhou 510006, China
[b] School of Software Engineering, South China University of Technology, Guangzhou 510006, China

ARTICLE INFO

Article history:
Received 20 January 2016
Accepted 1 February 2016
Available online 13 February 2016

Keywords:
Protein function prediction
Genome wide
Distance metric learning
Machine learning
Multi-instance multi-label learning

ABSTRACT

Multi-instance multi-label (MIML) learning has been proven to be effective for the genome-wide protein function prediction problems where each training example is associated with not only multiple instances but also multiple class labels. To find an appropriate MIML learning method for genome-wide protein function prediction, many studies in the literature attempted to optimize objective functions in which dissimilarity between instances is measured using the Euclidean distance. But in many real applications, Euclidean distance may be unable to capture the intrinsic similarity/dissimilarity in feature space and label space. Unlike other previous approaches, in this paper, we propose to learn a multi-instance multi-label distance metric learning framework (MIMLDML) for genome-wide protein function prediction. Specifically, we learn a Mahalanobis distance to preserve and utilize the intrinsic geometric information of both feature space and label space for MIML learning. In addition, we try to deal with the sparsely labeled data by giving weight to the labeled data. Extensive experiments on seven real-world organisms covering the biological three-domain system (*i.e.*, archaea, bacteria, and eukaryote; Woese et al., 1990) show that the MIMLDML algorithm is superior to most state-of-the-art MIML learning algorithms.

1. Introduction

As more genomic sequences become available, functional annotation of genes is becoming one of the most important challenges in bioinformatics. And the computational methods for genome-wide protein function prediction (Marcotte et al., 1999; Radivojac et al., 2013; Wu et al., 2015) have emerged as an urgent problem at the forefront of the post-genomic era since it is expensive and time-consuming to determinate the protein structure and function with experimental (Andorf et al., 2007). With the computational methods, we can annotate hundreds or thousands of proteins in a matter of minutes. In such case, we can save a significant amount of labeling effort.

During the past few years, various computational methods have been developed for genome-wide protein function prediction (Andorf et al., 2007; Radivojac et al., 2013; Wu et al., 2014). For example, in EnMIMLmetric (Wu et al., 2014), the protein function prediction problem has been solved as a naturally and inherently multi-instance multi-label learning task (Zhou et al., 2012; Wu et al., 2014). MIML learning tasks deal with the problem where each training example is involved in not only multiple instances but also multiple class labels. In addition to EnMIMLmetric, there are many other MIML learning methods which can be used to tackle the protein function prediction problem (*i.e.*, MIMLkNN (Zhang, 2010), MIMLNN (Zhou et al., 2012), MIMLSVM (Zhou et al., 2012), MIMLBOOST (Zhou et al., 2012), Markov-Miml (Wu et al., 2013)).

As far as we know, most of the existing MIML learning methods were designed by using the Euclidean distance to measure the dissimilarity between instances. It sometimes makes MIML learning suffer from the limitations which are associated with the Euclidean distance. From the perspective of classification, the objective functions that are described by using Euclidean distance may be inappropriate to maximize the distance between classes, while minimizing that within each class for some real-world applications (Li et al., 2006; Weinberger et al., 2005; Yang and Jin, 2006; Long et al., 2014) because Euclidean distance may not be able to capitalize on any statistical regularities in the data that might be estimated from a large training set of labeled data (Weinberger et al., 2005). In such case, using a pre-defined and data-independent distance to learn a model for MIML problem may be not applicable. In addition, different from traditional metric learning setting, each bag in MIML learning setting is associated with a unique label vector, and the instances in the same bag are associated with the same label vector. In traditional metric learning methods, we always

* Corresponding author.
E-mail addresses: hqmin@scut.edu.cn (H. Min), qyw@scut.edu.cn (Q. Wu).

http://dx.doi.org/10.1016/j.compbiolchem.2016.02.011
1476-9271/

maximize the distance between classes. If we do not consider the differences between the bags' distances and use the same strategy to maximize the distance between bags for MIML learning, we may lose the intrinsic geometric information among the label space. We also notice that the label space of some MIML learning datasets are sparse (*i.e.*, the datasets used in Wu et al., 2014). In such case, labeled data and unlabeled data from a same class may be unbalanced (Cieslak and Chawla, 2008; Ando, 2015). However, many existing MIML learning methods (*i.e.*, Jin et al., 2009) ignore this problem and treat the labeled data and the unlabeled data equally.

To address these issues, in this paper, we proposed a new MIML algorithm, called multi-instance multi-label distance metric learning (MIMLDML). Compared with other state-of-the-art MIML learning algorithms, the main contributions of our approach are three folds,

- We propose a multi-instance multi-label distance metric learning framework that is applicable to MIML learning problems. Due to the advantages of Mahalanobis distance (*i.e.*, unit less, scale-invariant and taking into account the correlations of the data set), this framework can more efficiently preserve and utilize the intrinsic geometric information among the instances from different bags. By this way, MIMLDML improves the performance of genome-wide protein function prediction.
- Different from traditional metric learning methods, we consider the difference between the bags' distances, and bind the margin between the bags with the label vector distance between the bags. By doing this, the learned Mahalanobis distance can preserve more intrinsic geometric information of the label space.
- We find that the label space of some MIML learning datasets are sparse. By giving weight to the labeled data in our learning framework, MIMLDML increases the weight for sparsely labeled data and improves the average recall rate for genome-wide protein function prediction.

Experimental results on seven real-world datasets show that learning performance can be significantly enhanced when the geometrical structure is exploited and the weight-trick approach is considered using our proposed method.

The rest of this paper is organized as follows: In Section 2, we formulate the protein function prediction task and present MIMLDML in details. Furthermore, to solve the problems presented in Section 2, an alternating optimization method for our approach is presented in Section 3. We report the experimental results of this paper on seven real-world organisms datasets in Section 4. Finally, we conclude this paper and discuss future work in Section 5.

2. The proposed method

In this section, we first formulate the protein function prediction task. Then we present MIMLDML in details.

2.1. The formulation of the protein function prediction task

In our approach, the protein function prediction problem is solved as a MIML learning task (Wu et al., 2014). Before describing this task, we give some definitions.

We denote by $D = \{(X_i, Y_i) | i = 1, 2, \ldots, n_{bag}\}$ the training dataset, where $X_i = (x_{i_1}, \ldots, x_{i_{n_i}})$ is a bag of n_i instances, and every instance $x_{i_j} \in R^d$ is a vector of d dimensions. n_{bag} indicates the bag number in D. $Y_i \in R^L$ is a binary vector, and Y_{i_k} is the kth element in Y_i. $Y_{i_k} = 1$ indicates that bag X_i is assigned to class c_k, and $Y_{i_k} = 0$ otherwise. We assume that bag X_i is assigned to c_k at least one instance in X_i belongs to c_k.

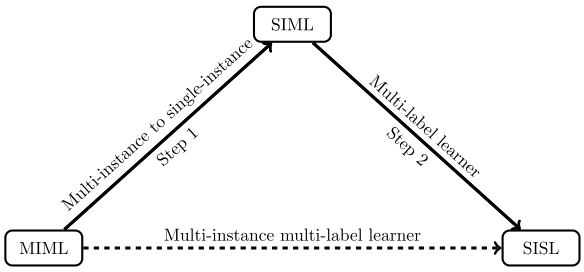

Fig. 1. Schematic illustration of multi-instance multi-label learning framework using multi-label learning as the bridge.

In our approach, the MIML learning task aims to find a hypothesis $h : X \rightarrow Y$ from the training data D. We notice that, without explicit relationship between an instance x_{i_j} and a label Y_{i_k}, this learning problem is more difficult than traditional supervised learning methods that learn concepts from objects represented by a single instance which is associated with a single label (Wu et al., 2014).

2.2. The MIMLDML learning framework

As presented in Zhou et al. (2012), traditional supervised learning, multi-instance learning and multi-label learning are all degenerated versions of MIML. Armed with this idea, Zhou et al. (2012) tackles MIML problem by identifying its equivalence in the traditional supervised learning framework. Fig. 1 shows the schematic illustration of the MIML learning framework using multi-label learning as the bridge. From the figure, we can find that MIML learning task is divided into two steps. In the first step, MIML learning task ($h : X \rightarrow Y$) is transformed into a single-instance multi-label (SIML) learning task (Zhou et al., 2012). Then, in the second step, the SIML learning task is transformed into a single-instance single-label (SISL) learning task (Zhou et al., 2012).

MIMLSVM is a state-of-the-art multi-instance multi-label learning algorithm which follows the MIML learning framework in Zhou et al. (2012). In the first step of MIMLSVM, a k-medoids clustering is performed on the training data D under Euclidean distance. Then, with the help of these medoids, MIMLSVM transforms the MIML learning task into the SIML learning task. In the second step of MIMLSVM, a multi-label learner is used to transform the SIML learning task into SISI learning task. Then, SVM (Chang and Lin, 2011) is used for each SISI learning task. MIMLSVM uses Euclidean distance to measure the similarity/dissimilarity between instances. Although it is theoretical simple and widely applied in many MIML applications, Euclidean distance may not be able to capitalize on statistical regularities in the data which might be estimated from a large training set of labeled instances (Weinberger et al., 2005). And the objective functions that are described by using the Euclidean distance may be inappropriate to maximize the distance between bags while minimizing the distance within each bag. In such case, MIMLSVM may suffer from the limitations of Euclidean distance.

Different from Euclidean distance, Mahalanobis distance has been proven to be effective for preserving and utilizing the intrinsic geometric information among the instances. A lot of previous work has shown that an appropriate metric can significantly benefit classification in terms of prediction accuracy (Yang and Jin, 2006; Kulis, 2012). The class of Mahalanobis distance is one of the most popular metrics in machine learning. Herein, Fig. 2 provides an example to demonstrate the advantage of Mahalanobis distance on single-instance single-label classification (Zhou et al., 2012). Fig. 2(left) displays the original data which are generated according to three Gaussian distributions with different means. From this figure, we observe that it is not easy to distinguish the data subject to

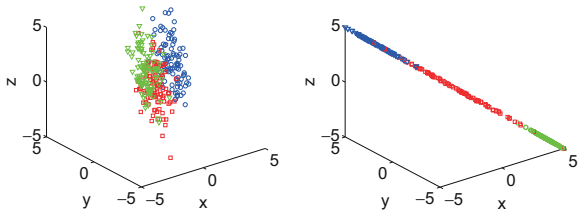

Fig. 2. An example to demonstrate the advantage of Mahalanobis distance on classification. (Left) displays the original data which are generated according to three Gaussian distributions with different means. (Right) shows the scaled data which are obtained by transforming the three Gaussian distributions data with learned Mahalanobis distance. From this figure, we observe that it is not easy to distinguish different distributions under Euclidean distance. However, different distributions can be easily distinguished under Mahalanobis distance.

different distributions under Euclidean distance. Fig. 2(right) shows the data which are obtained by transforming the three Gaussian distributions data with Mahalanobis distance (*i.e.*, the Mahalanobis distance learned by information-theoretic metric learning (Davis et al., 2007)). From Fig. 2(b), we observe that different distributions can easily be distinguished under Mahalanobis distance.

Motivated by previous progress in MIML and metric learning with Mahalanobis distance, in this paper, we propose a novel algorithm, called multi-instance multi-label distance metric learning (MIMLDML). It is worthwhile to highlight several differences between our proposed approach and the MIMLSVM algorithm here: (1) Different from MIMLSVM, MIMLDML uses Mahalanobis distance instead of Euclidean distance to measure the distance between instances. (2) MIMLSVM treats the labeled data and unlabeled data equally and ignores the unbalance characteristic between them. In contrast, MIMLDML takes the unbalance characteristic into consideration and increases the weight for the sparsely labeled data.

2.2.1. The first step of MIMLDML

The learning process of MIMLDML is divided into two steps. The first step is to transform the MIML learning task ($h : X \to Y$) a single-instance multi-label learning (SIML) task ($h_{SIML} : Z \to Y$, where Z is the instance space which is transformed from X, Zhou et al., 2012). MIMLSVM performs this learning in the Euclidean space, but it fails to reveal the intrinsic geometrical structure of the MIML data, which is essential to improve the performance of MIML learning. In MIMLDML, we introduce a novel Mahalanobis distance based metric learning function which avoids this limitation.

Let $M \in R^{d\times d}$ be a positive semi-definite matrix, then the Mahlanobis distance between a pair of instances x_i and x_j is defined as follows,

$$d_{ij} = \sqrt{(x_i - x_j)^T M (x_i - x_j)}. \tag{1}$$

As M is positive semi-definite, it can be decomposed as $M = A^T A$ where $A \in R^{d\times d}$. Therefore, learning the Mahalanobis distance d is equivalent to learn the matrix A. With the defined Mahalanobis distance, we define the distance (margin) between two bags X_i and X_j as the distance of the centers of X_i and X_j, *i.e.*,

$$D(X_i, X_j) = \sqrt{(\bar{X}_i - \bar{X}_j)^T A^T A (\bar{X}_i - \bar{X}_j)}, \tag{2}$$

where $\bar{X}_i$ and $\bar{X}_j$ are the average values of all the instances in X_i and X_j, respectively.

In order to learn an appropriate Mahalanobis distance, we make two principles to construct an objective function for MIML distance metric learning. In particular, we take the following principles into consideration to learn optimal distance metric by using the MIML data D: (a) minimizing the distance in each bag, and (b) the larger the label vector distance between the bags, the larger the margin between the bags. Fig. 3 shows the schematic illustration of the margin between bags for multi-instance multi-label distance metric learning framework.

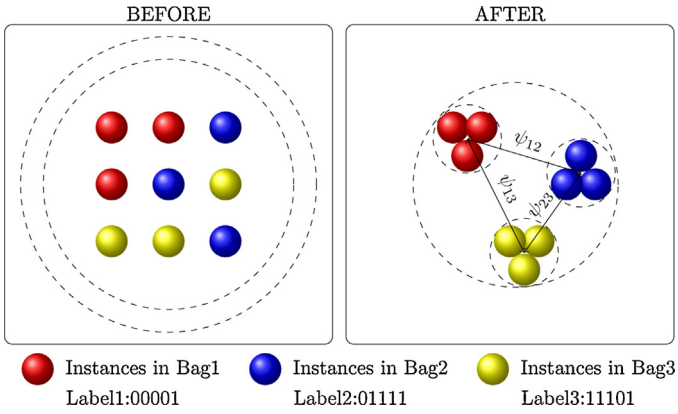

Fig. 3. Schematic illustration of the margin between bags for multi-instance multi-label distance metric learning frameworks, before training (left) versus after training (right). After training, the larger the label vector distance between the bags, the larger the margin between the bags. By binding the label vector distance between the bags with the margin between the bags, MIMLDML can encode much structural information of the label space.

On the one hand, to minimize the distance in each bag, we restrict all the instances into a minimum enclosing ball whose radius is set to 1,

$$\|Ax_i - c\|^2 \le 1, \tag{3}$$

where c is the center of the instances. With a preprocessing step to centralize the input data,

$$x_i \leftarrow x_i - \frac{1}{n}\sum_{j=1}^{n_{all}} x_j, \tag{4}$$

where n_{all} indicates all the instances number in D, (3) can be represented as,

$$\|Ax_i\|^2 \le 1. \tag{5}$$

In the following, without special declaration the data is supposed to be centralized. Note that, the choice of the constant 1 in the right hand side of (5) is arbitrary but not important, and changing it to any other positive constant τ results only in A being replaced by $\sqrt{\tau}A$.

On the other hand, to bind the margin between the bags with the label vector distance between the bags, we require the distance $D(X_i, X_j)$ between bags larger than a margin $\psi(Y_i, Y_j)$.

$$D(X_i, X_j) \ge \psi(Y_i, Y_j), \tag{6}$$

where $\psi(Y_i, Y_j)$ indicates the margin between ith bag and jth bag. In our approach, the margin $\psi(Y_i, Y_j)$ should increase rapidly with increasing (squared) distance $D(X_i, X_j)$. Hence, we define $\psi(Y_i, Y_j)$ as follows,

$$\psi(Y_i, Y_j) = \sqrt{\exp(\delta \ell_i) - 1}, \tag{7}$$

where ℓ_i indicates the hamming distance (Zhou et al., 2012) between Y_i and Y_j. $\delta > 0$ is a margin factor which is used to tune the margin between X_i and X_j. The larger δ, the larger margin between X_i and X_j. The setting of δ is studied in the experiment setting. Different from other approaches (Jin et al., 2009), we bind the hamming distance between the labels of the bags with the margin between bags. In our approach, the larger of the distance between labels, the larger the margin between bags. By doing so, we can encode much structural information of the label space.

Combined (5)–(7), the proposed method for multi-instance multi-label classification distance metric learning can be formulated as follows,

$$\min_A f(A) = \|A\|_F^2 \quad s.t. \begin{cases} \|Ax_i\|^2 \le 1, \quad x_i \in X_j, \quad j = 1, 2, \ldots, n_{bag}, \\ \|Av_i\|^2 \ge \exp(\delta \ell_i) - 1, \quad v_i \in \Gamma, \end{cases} \tag{8}$$

where $\Gamma = \{(\bar{X}_{i_1} - \bar{X}_{i_2}) | i_1, i_2 = 1, 2, \ldots, n_{bag}, Y_{i_1} \ne Y_{i_2}\}$ and $\bar{X}_{i_1}$ is the average value of all the instances in the bag X_{i_1}. We denote by n_{out} the size of Γ. $\|A\|_F^2$ is a regularization term to control the generalization error of the learned metric (Kulis, 2012).

Sometimes, there may be noises in the training data and our algorithm may be affected by the noise. In order to avoid this problem, we introduce two slack vectors $\xi \in R^{n_{all}}$ and $\zeta \in R^{n_{out}}$ for (8). By doing so, we improve the robustness of our algorithm. Then, we obtain the optimization problem for distance metric learning in multi-instance multi-label classification as follows,

$$\min_{A,\xi,\zeta} f(A, \xi, \zeta) = \|A\|_F^2 + \lambda \sum_{i=1}^{n_{all}} \xi_i + \beta \sum_{i=1}^{n_{out}} \zeta_i \quad s.t. \begin{cases} \|Ax_i\|^2 \le 1 + \xi_i, \quad x_i \in X_j, \quad j = 1, 2, \ldots, n_{bag}, \\ \|Av_i\|^2 \ge \exp(\delta \ell_i) - 1 - \zeta_i, \quad i \in \Gamma, \\ \xi \ge 0 \quad \text{and} \quad \zeta \ge 0, \end{cases} \tag{9}$$

where ξ_i and ζ_i are elements of ξ and ζ, respectively.

After learning the Mahalanobis distance metric, a k-medoids clustering is performed on all the instances of D with the learned Mahalanobis distance. We define C_k as the cluster number for the k-medoids clustering. In our experiment, we empirically set C_k to be $0.2 * n_{bag}$. With these medoids, the original instance X_{i_j} is transformed into a k-dimensional numerical vector $z_j \in Z$, where the mth ($m = 1, 2, \ldots, k$) component of z_j is the distance between X_{i_j} and mth medoid M_m. By doing so, we transfer the MIML learning task into a multi-label learning task.

2.2.2. The second step of MIMLDML

In the second step of MIMLDML, the single-instance multi-label learning task that obtained from the first step of MIMLDML is further transformed into a traditional supervised learning task. This transformation is achieved by decomposing the multi-label learning problem into multiple independent binary classification problems (one per class). For each subtask, the instance associated with the label data $Y_{i_k} = 1$ is considered as a positive instance, while being regarded as a negative instance when $Y_{i_k} = 0$. With the statistical analysis of the datasets (Tables 1 and 2), we find that the proportion of positive and negative instance in each class is unbalanced. However, many researches have shown that learning from unbalanced datasets presents a convoluted problem in which traditional learning algorithms may perform poorly (Cieslak and Chawla, 2008; Ando, 2015). To solve this problem, we give a weight to each class (*i.e.*, we fix the weight of negative class to be 1, and set the weight of positive class to be w), when learning with SVM (Chang and Lin, 2011) for each binary classification problem. With a weight $\omega > 1$, we can give more important weight for the loss of the mis-classified positive instances when learning. In this case, classification for positive instance can be more accurately. By this method, we improve the recall rate of the classification method, even with a sparsely labeled data. The setting of w is studied in the experiment setting.

Algorithm 1. Distance metric learning in MIMLDML

Require: training set D, a margin factor δ, a weight parameter w, step size's γ_1, γ_2 and γ_3, tradeoff parameters (λ, β), a penalty coefficient σ, and a threshold ε.
1: Initialize A^0, ξ^0 and ζ^0.
2: Centralize the input data: $x_i \leftarrow x_i - \frac{1}{n}\sum_{j=1}^{n_{all}} x_j$.
3: **procedure MIMLDML** A, ξ, ζ
4: **while** *true* **do**
5: Update A by $A^{t+1} = A^t - \gamma_1 \frac{\partial f(A,\xi,\zeta)}{\partial A}\Big|_{A^t}$.
6: Update each $\xi_i \in \xi$ by $\xi_i^{t+1} = \xi_i^t - \gamma_2 \frac{\partial f(A,\xi,\zeta)}{\partial \xi_i}\Big|_{\xi_i^t}$.
7: Update each $\zeta_i \in \zeta$ by $\zeta_i^{t+1} = \zeta_i^t - \gamma_3 \frac{\partial f(A,\xi,\zeta)}{\partial \zeta_i}\Big|_{\zeta_i^t}$.
8: **if** $\left|f(A^{t+1}, \xi^{t+1}, \zeta^{t+1}) - f(A^t, \xi^t, \zeta^t)\right| < \varepsilon$ **then**
9: $A = A^{t+1}$, $\xi = \xi^{t+1}$, $\zeta = \zeta^{t+1}$, break.
10: **end if**
11: **end while**
12: **end procedure**
Ensure: $(A, \xi, \zeta) = \arg\min f(A, \xi, \zeta)$.

3. Optimizations

In this section, we derive approaches to solve the optimization problem constructed in (9). We first convert the constrained problem to an unconstrained problem by adding penalty functions. The resulting optimization problem becomes,

$$\begin{aligned} \min_{A,\xi,\zeta} f(A, \xi, \zeta) = \|A\|_F^2 + \lambda \sum_{i=1}^{n_{all}} \xi_i + \beta \sum_{i=1}^{n_{out}} \zeta_i \\ + \sigma \sum_{i=1}^{n_{all}} \{[\max(0, \|Ax_i\|^2 - 1 - \xi_i)]^2 + [\max(0, -\xi_i)]^2\} \\ + \sigma \sum_{i=1}^{n_{out}} \{[\max(0, \exp(\delta \ell_i) - \|Av_i\|^2 - 1 - \zeta_i)]^2 \\ + [\max(0, -\zeta_i)]^2\}, \end{aligned} \tag{10}$$

where σ is the penalty coefficient.

Then we use the gradient-projection method (Bertsekas, 1999) to solve (10). To be precise, in the first step, we initialize A^0, ξ^0 and ζ^0, and centralize the input data by (4). In the second step, we update the value of A, ξ and ζ using gradient descent based on the following rules,

$$A^{t+1} = A^t - \gamma_1 \frac{\partial f(A, \xi, \zeta)}{\partial A}\Big|_{A^t}, \tag{11}$$

$$\xi_i^{t+1} = \xi_i^t - \gamma_2 \frac{\partial f(A, \xi, \zeta)}{\partial \xi_i}\Big|_{\xi_i^t}, \tag{12}$$

$$\zeta_i^{t+1} = \zeta_i^t - \gamma_3 \frac{\partial f(A, \xi, \zeta)}{\partial \zeta_i}\Big|_{\zeta_i^t}. \tag{13}$$

The derivatives of the objective f with respect to A, ξ and ζ in (11)–(13) are,

$$\begin{aligned} \frac{\partial f(A, \xi, \zeta)}{\partial A} = 2A + 4\sigma A\{\sum_{i=1}^{n_{all}} \max(0, \|Ax_i\|^2 - 1 - \xi_i) \\ - \sum_{i=1}^{n_{out}} v_i v_i^T \max(0, \exp(\delta \ell_i) - \|Av_i\|^2 - 1 - \zeta_i)\}, \end{aligned} \tag{14}$$

$$\frac{\partial f(A, \xi, \zeta)}{\partial \xi_i} = \lambda - 2\sigma[\max(0, \|Ax_i\|^2 - 1 - \xi_i) + \max(0, -\xi_i)], \tag{15}$$

Table 1
Characteristics of the datasets.

	Genome	Bags	Classes	Instances	Instances per bag (mean ± std)	Labels per instance (mean ± std)
Archaea	*Haloarcula marismortui*	304	234	950	3.13 ± 1.09	3.25 ± 3.02
	Pyrococcus furiosus	425	321	1317	3.10 ± 1.09	4.48 ± 6.33
Bacteria	*Azotobacter vinelandii*	407	340	1251	3.07 ± 1.16	4.00 ± 6.97
	Geobacter sulfurreducens	379	320	1214	3.20 ± 1.21	3.14 ± 3.33
Eukaryote	*Caenorhabditis elegans*	2512	940	8509	3.39 ± 4.20	6.07 ± 11.25
	Drosophila melanogaster	2605	1035	9146	3.51 ± 3.49	6.02 ± 10.24
	Saccharomyces cerevisiae	3509	1566	6533	1.86 ± 1.36	5.89 ± 11.52

Table 2
Details information about positive and negative instances of the datasets.

	HM	PF	AV	GS	CE	DM	SC
Positive instances per classes (mean ± std)	4.2 ± 9.4	5.9 ± 13.0	4.8 ± 9.8	3.7 ± 9.0	16.2 ± 48.9	15.2 ± 48.5	13.2 ± 43.9
Positives instances/negative instances	1.41%	1.42%	1.19%	0.99%	0.65%	0.59%	0.38%

$$\frac{\partial f(A,\xi,\zeta)}{\partial \zeta_i} = \beta - 2\sigma \max(0, \exp(\delta\ell_i) - \|Av_i\|^2 - 1 - \zeta_i) - 2\sigma \max(0, -\zeta_i). \quad (16)$$

We repeat the second step until the change of the objective function f is less than a threshold ε. A detailed procedure is given in Algorithm 1.

4. Experiments

In this section, we compare the performance of the proposed algorithm with the performances of other previously proposed algorithms: MIMLkNN (Zhang, 2010), MIMLSVM (Zhou et al., 2012), MIMLNN (Zhou et al., 2012), and EnMIMLNNmetric (Wu et al., 2014) on seven real-world organisms covering the biological three-domain system (Woese and Fox, 1977; Woese et al., 1978, 1990) (*i.e.*, archaea, bacteria, and eukaryote) which shows that the proposed algorithm outperforms other algorithms.

To make a fair comparison, all the experiments were conducted over 10 random permutations for each dataset and the results are reported by averaging over those 10 runs.

4.1. Datasets

Before we proceed to present empirical results, we offer a description of the used datasets. The datasets are outlined below.

Seven real-world organisms datasets[1] have been used in the prior research on genome-wide protein function prediction (Wu et al., 2014). These seven real-world organisms include two archaea genomes: *Haloarcula marismortui* (HM) and *Pyrococcus furiosus* (PF); two bacteria genomes: *Azotobacter vinelandii* (AV) and *Geobacter sulfurreducens* (GS); three eukaryote genomes: *Caenorhabditis elegans* (CE), *Drosophila melanogaster* (DM) and *Saccharomyces cerevisiae* (SC). In our experiments, each bag including several instances is used to represent the protein in organisms, each instance is represented by a 216-dimensions vector in which each dimension denotes the frequency of a triad type (Wu et al., 2011), and each instance is labeled with a group of GO molecular function terms (Ashburner et al., 2006). The characteristics of the datasets are summarized in Table 1. For example, there are 3509 proteins (bags) with a total of 1566 gene ontology terms (label classes) on molecular function in the *S. cerevisiae* dataset (Table 1). The total instance number of *S. cerevisiae* dataset is 6533. The average number of instances per bag (protein) is 1.86 ± 1.36, and the average number of labels (GO terms) per instance is 5.89 ± 11.52.

[1] http://lamda.nju.edu.cn/files/MIMLprotein.zip.

4.2. Evaluation metrics

For evaluating the performance of our proposed MIMLDML algorithm, we used four evaluation criteria, i.e., Ranking Loss (RL) is selected to evaluate the average fraction of misordered label pairs produced by our algorithm, Coverage is selected to assess the performance of MIMLDML for all the possible labels of documents, Average-Recall (avgRecall) and Average-F1 (avgF1) are selected to evaluate the performance of the weight trick in MIMLDML. These four measures are often used for evaluating the performance of MIML learning problem (Zhang, 2010; Jin et al., 2009; Radivojac et al., 2013).

We first give some definitions, and then we present the four evaluation criteria briefly. For a given test set $S = (X_1, Y_1), (X_2, Y_2), \ldots, (X_N, Y_N)$, we use $h(X_i)$ to represent the returned labels for X_i; $h(X_i, y)$ to represent the returned confidence (real-value) for X_i; $rank^h(X_i, y)$ to represent the rank of y which is derived from $h(X_i, y)$; $\bar{Y}_i$ to represent the complementary set of Y_i.

1. *Ranking Loss*: The ranking loss evaluates the average fraction of misordered label pairs for the test bag. The smaller the value of ranking loss, the better the performance.

$$Rankingloss(h) = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{|Y_i||\bar{Y}_i|}|\{(y_1, y_2)| h(X_i, y_1) \le h(X_i, y_2), (y_1, y_2) \in Y_i \times \bar{Y}_i\}|,$$

2. *Coverage*: The coverage evaluates the average fraction of how far it is needed to go down the list of labels in order to cover all the proper labels of the test bag. The smaller the value of coverage, the better the performance.

$$Coverage(h) = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{|Y_i||\bar{Y}_i|}\max_{y \in Y_i} rank^h(X_i, y) - 1.$$

3. *Average-Recall*: The average recall evaluates the average fraction of correct labels that have been predicted. The larger the value of Average-Recall, the better the performance.

$$avgRecall(h) = \frac{1}{N}\sum_{i=1}^{N}\frac{|\{y|rank^h(X_i, y) \le |h(X_i)|, y \in Y_i\}|}{|Y_i|}.$$

4. *Average-F1*: The average F1 indicates a tradeoff between the average precision (Zhou et al., 2012) and the average recall. The larger the value of Average-F1, the better the performance.

$$avgF1(h) = \frac{2 \times avgPrec(h) \times avgRecall(h)}{avgPrec(h) + avgRecall(h)},$$

where $avgPrec(h)$ indicates the average precision (Zhou et al., 2012).

4.3. Comparison methods

In this paper, we compare our proposed method with the following MIML classification algorithms since the protein function prediction was degenerated as a multi-label learning framework by some previous researches (Yu et al., 2012, 2013; Mostafavi and Morris, 2010):

(1) *MIMLkNN*: MIMLkNN is proposed for MIML by utilizing the popular *k*-nearest neighbor techniques. Motivated by the advantage of the citers that is used in Citation-kNN approach (Wang and Zucker, 2000), MIMLkNN not only considers the test instances' neighboring examples in the training set, but also considers those training examples which regard the test instance as their own neighbors (*i.e.*, the citers). In this way, MIMLkNN makes multi-label predictions for the unseen MIML example.

(2) *MIMLSVM*: Different from MIMLkNN, MIMLSVM first degenerates MIML learning task to a simplified multi-label learning (MLL) task by using a clustering-based representation transformation (Zhou and Zhang, 2007; Zhou et al., 2012). Then the MLL task is further transformed into a traditional supervised learning task by MLSVM. In this way, MIMLSVM uses multi-label learning as a bridge to transfer the MIML problem into a traditional supervised learning framework.

(3) *MIMLNN*: This baseline method follows the same MIML learning framework as MIMLSVM. MIMLNN is obtained by using the two-layer neural network structure (Zhang and Zhou, 2007) to replace the MLSVM (Zhou et al., 2012) used in MIMLSVM. Zhou et al. (2012) has shown that MIMLNN preforms quite well on some MIML datasets.

(4) *EnMIMLNNmetric*: This baseline method is the metric-based ensemble multi-instance multi-label classification method of EnMIMLNN (Wu et al., 2014). Different from MIMLNN, EnMIMLNN combines three different Hausdorff distances (*i.e.*, average, maximal and minimal) to denote the distance between two proteins. In EnMIMLNN, two voting-based models (*i.e.*, EnMIMLNN$voting_1$ and EnMIMLNN$voting_2$) have also been proposed. In our experiments, we only compare with EnMIMLNNmetric since EnMIMLNNmetric outperforms the other two voting-based models in most cases (Wu et al., 2014).

The codes [2] of these four MIML classification algorithms have been shared by their authors. To make a fair comparison, these algorithms are set to the best parameters which are reported in the papers. Specifically, for MIMLkNN, the number of citers and the number of nearest neighbors are set to 20 and 10, respectively (Zhang, 2010); for MIMLNN, the regularization parameter used to compute matrix inverse is set to 1 and the number of clusters is set to 40% of the training bags; for MIMLSVM, the number of clusters is set to 20% of the training bags and the SVM used in MIMLSVM is implemented by LIBSVM (Chang and Lin, 2011) package with radial basis function whose parameter "$-c$" is set to 1 and "$-g$" is set to 0.2; for EnMIMLNNmetric, the fraction parameter and the scaling factor are set to 0.1 and 0.8, respectively (Wu et al., 2014).

[2] http://lamda.nju.edu.cn/CH.Data.ashx.

4.4. Parameter configurations

Our proposed approach that has been described in Section 2 involves some tunable parameters (*i.e.*, the margin factor δ and the weight of positive class w). In the following experiments, we test to investigate how would different values of the parameters δ and w affect the performance of MIMLDML. For each test, we randomly select 50% of the bags in the dataset as training data and use remained bags as test data.

Parameter δ is used as the margin factor to separate different bags. To illustrate the sensitivity of this parameter, we show the performance of the proposed algorithm with various δ on different datasets in Fig. 4. We observe from Fig. 4 that there are several plateaus in the coverage curves which indicate that MIMLDML is quite insensitive to the specific setting of δ on coverage. We also observe that when $\delta = 0.001$, the ranking loss and avgRecall of MIMLDML are high on most datasets, and the avgF1 is low. In fact, the smaller δ, the shorter the constraint distances between bags. In this case, it is difficult to separate unique bags and this may lead to a worse performance of MIMLDML. When $\delta \le 0.1$, the avgRecall of MIMLDML on the seven datasets keeps high, but the avgF1 is low. Considering the fact that avgF1 is a tradeoff between the average precision (Zhou et al., 2012) and the average recall, we find that the lower avgF1 may indicate that a lower $\delta \le 0.1$ contributes little to the precision of MIMLDML. As δ increasing to the other extreme (*i.e.*, $\delta = 10$), the performances of MIMLDML reach the perk on most datasets which indicates that MIMLDML can construct an effective Mahalanobis distance for the MIML problem on these datasets. Hence, we set $\delta = 10$ for MIMLDML on all datasets.

Parameter w is used to reweight the importance of the positive class since the proportion of positive and negative instances in each class is unbalanced (Table 2). To illustrate the sensitivity of this parameter, we show the performances of the proposed algorithm with various w on different datasets in Fig. 5. From the figure, we can find that with the increasing of w, the avgRecalls of MIMLDML on all the seven datasets increase and the avgF1s of MIMLDML decrease slightly. This is because a larger weight for positive class may influence the accuracy of the negative class. From Fig. 5(a) and (b), we notice that with the increasing of w, the performances of MIMLDML on HM, PF, AV and GS are gradually worse, while the performances of MIMLDML on CE, DM and SC become better. These results indicate that w is sensitive to diverse datasets, a small w is suitable for HM, PF, AV and GS datasets and a larger w is suitable for CE, DM and SC datasets. Hence, we set $w = 10$ for HM, PF, AV and GS datasets and we set $w = 100$ for CE, DM and SC datasets.

4.5. Performance comparison

In this section, we conduct four experiments to verify the performance of MIMLDML. For each experiment, the performance is measured in evaluation criteria which have been introduced in Section 4.2.

In Section 2, we have presented the MIMLDML in details. In MIMLDML, Mahalanobis distance is used to measure the distance between instances. And we also use Mahalanobis distance to conduct the distance between bags in (2). However, some recent researches have used Hausdorff distance to measure the distance between bags. To compare the advantage of Mahalanobis distance

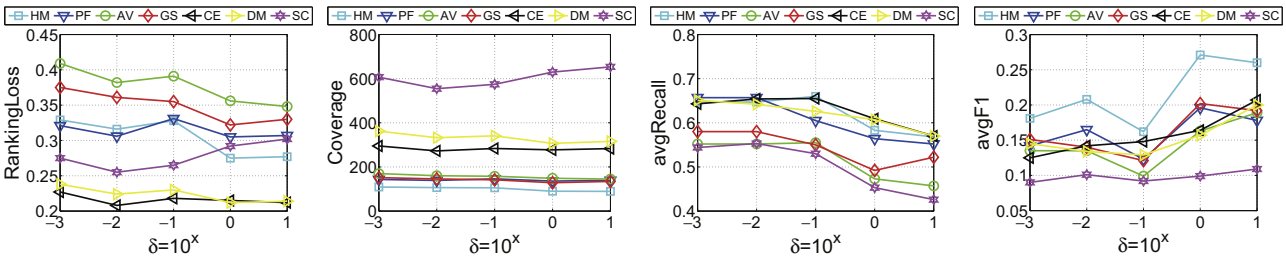

Fig. 4. The performance of MIMLDML on HM, PF, AV, GS, CE, DM and SC datasets under different values of the margin factor δ when the weight parameter w is fixed to 100.

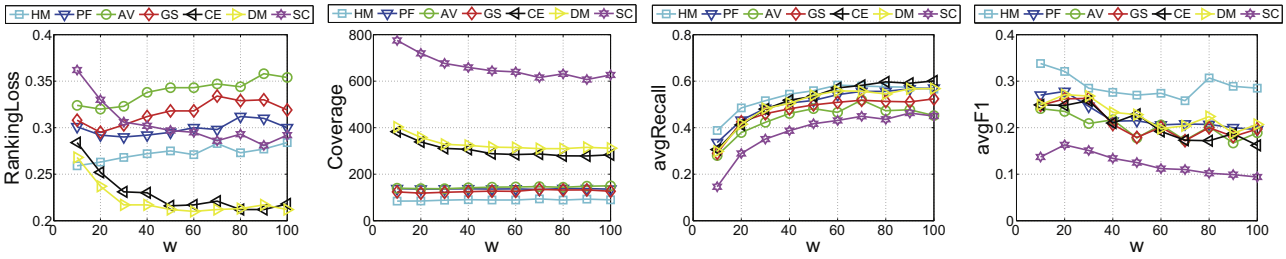

Fig. 5. The performance of MIMLDML on HM, PF, AV, GS, CE, DM and SC datasets under different values of the positive class weight parameter w when the margin factor δ is fixed to 1.

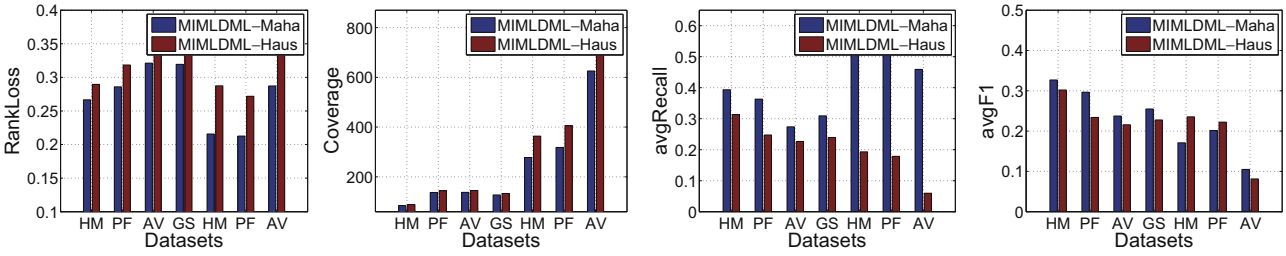

Fig. 6. Comparison results of MIMLDML-Maha and MIMLDML-Haus.

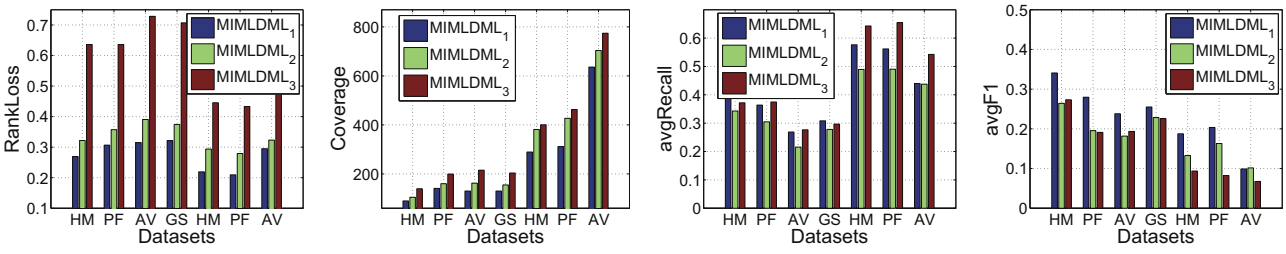

Fig. 7. Comparison results of MIMLDML_1, MIMLDML_2 and MIMLDML_3.

with Hausdorff distance, we implement two versions of MIMLDML algorithm (*i.e.*, MIMLDML-Maha and MIMLDML-Haus), in this section. Keeping other settings unchanged, MIMLDML-Maha uses the learned Mahalanobis distance to measure the distance between bags while MIMLDML-Haus uses Hausdorff distance (Edgar, 2007). To examine the difference between Mahalanobis distance and Hausdorff distance (Edgar, 2007), we compare the performances of MIMLDML-Maha and MIMLDML-Haus on the seven datasets. Fig. 6 shows the experimental results. From the figure, we can find that the ranking loss and coverage of MIMLDML-Maha on these datasets are significantly lower than that of MIMLDML-Haus; the avgF1 and avgRecall of MIMLDML-Maha on most datasets are higher than MIMLDML-Haus. The experimental results in Fig. 6 indicate that the effectiveness of the learned Mahalanobis distance is particularly significant. In fact, comparing with MIMLDML-Haus, MIMLDML-Maha can preserve and utilize the intrinsic geometric information among feature space and label space. By using this method, MIML-Maha can utilize the intrinsic geometric information among the bags more efficiently.

To learn an appropriate Mahalanobis distance for MIML problems, we propose two principles (*i.e.*, (a) and (b)) in Section 2. Based on the principles, two types of constraints (*i.e.*, (5) and (6)) have been added to our learning framework. To compare the impact of the two principles on MIMLDML, we conduct the second experiment. In this experiment, MIMLDML_1 denotes the MIMLDML using both principles (a) and (b). MIMLDML_2 denotes the MIMLDML only using principle (a) and MIMLDML_3 denotes the MIMLDML only using principle (b). Experimental results are shown in Fig. 7. From the figure, we can find that both (a) and (b) enhance the effectiveness of our framework MIMLDML. The effectiveness of (b) is particularly significant and (a) further strengthens the effectiveness of MIMLDML on the basis of (b).

Table 3
Comparison results (mean ± std) with four state-of-the-art MIML methods on seven real-world organisms. ↓ (↑) indicates the smaller (larger), the better of the performance. The best results are marked in boldface.

	Genome	Methods	RL	Coverage	avgRecall	avgF1
Archaea	*Haloarcula marismortui*	MIMLkNN	0.425 ± 0.025	128.595 ± 7.013	0.257 ± 0.021	0.238 ± 0.020
		MIMLNN	0.315 ± 0.022	102.507 ± 5.119	0.063 ± 0.008	0.107 ± 0.011
		MIMLSVM	0.346 ± 0.013	106.145 ± 4.233	0.168 ± 0.009	0.216 ± 0.011
		EnMIMLNNmetric	0.310 ± 0.024	99.594 ± 4.691	0.180 ± 0.017	0.246 ± 0.017
		MIMLDML	**0.271 ± 0.025**	**85.469 ± 8.113**	**0.385 ± 0.027**	**0.3182 ± 0.023**
	Pyrococcus furiosus	MIMLkNN	0.435 ± 0.022	190.568 ± 4.477	0.264 ± 0.026	0.230 ± 0.020
		MIMLNN	0.317 ± 0.018	153.506 ± 6.774	0.053 ± 0.010	0.090 ± 0.015
		MIMLSVM	0.356 ± 0.014	158.709 ± 5.062	0.126 ± 0.014	0.168 ± 0.014
		EnMIMLNNmetric	0.323 ± 0.017	156.654 ± 7.148	0.142 ± 0.015	0.199 ± 0.015
		MIMLDML	**0.291 ± 0.019**	**140.504 ± 9.949**	**0.369 ± 0.026**	**0.290 ± 0.024**
Bacteria	*Azotobacter vinelandii*	MIMLkNN	0.473 ± 0.021	198.033 ± 4.641	0.251 ± 0.023	0.198 ± 0.007
		MIMLNN	0.372 ± 0.016	168.592 ± 6.140	0.055 ± 0.012	0.090 ± 0.015
		MIMLSVM	0.380 ± 0.019	157.809 ± 5.809	0.115 ± 0.009	0.151 ± 0.010
		EnMIMLNNmetric	0.371 ± 0.013	157.951 ± 5.514	0.128 ± 0.013	0.177 ± 0.016
		MIMLDML	**0.327 ± 0.016**	**139.947 ± 4.912**	**0.283 ± 0.023**	**0.241 ± 0.020**
	Geobacter sulfurreducens	MIMLkNN	0.483 ± 0.019	192.254 ± 7.343	0.247 ± 0.047	0.194 ± 0.018
		MIMLNN	0.369 ± 0.020	161.777 ± 7.135	0.051 ± 0.012	0.086 ± 0.017
		MIMLSVM	0.381 ± 0.025	149.809 ± 7.525	0.129 ± 0.013	0.165 ± 0.015
		EnMIMLNNmetric	0.393 ± 0.014	160.181 ± 4.556	0.127 ± 0.018	0.176 ± 0.020
		MIMLDML	**0.321 ± 0.015**	**127.426 ± 4.870**	**0.305 ± 0.013**	**0.252 ± 0.008**
Eukaryote	*Caenorhabditis elegans*	MIMLkNN	0.229 ± 0.008	311.087 ± 8.630	0.228 ± 0.014	0.285 ± 0.015
		MIMLNN	0.231 ± 0.003	317.182 ± 4.236	0.167 ± 0.008	0.231 ± 0.008
		MIMLSVM	**0.193 ± 0.010**	**265.731 ± 12.645**	0.218 ± 0.008	0.284 ± 0.008
		EnMIMLNNmetric	0.210 ± 0.006	287.861 ± 6.816	0.317 ± 0.013	**0.381 ± 0.011**
		MIMLDML	0.212 ± 0.008	283.004 ± 10.824	**0.570 ± 0.022**	0.207 ± 0.023
	Drosophila melanogaster	MIMLkNN	0.233 ± 0.007	373.013 ± 10.662	0.215 ± 0.014	0.268 ± 0.013
		MIMLNN	0.232 ± 0.008	371.994 ± 15.573	0.156 ± 0.010	0.219 ± 0.012
		MIMLSVM	**0.189 ± 0.005**	**307.607 ± 8.213**	0.193 ± 0.009	0.258 ± 0.010
		EnMIMLNNmetric	0.214 ± 0.008	348.628 ± 13.808	0.300 ± 0.008	**0.361 ± 0.008**
		MIMLDML	0.217 ± 0.009	323.875 ± 10.715	**0.505 ± 0.024**	0.233 ± 0.029
	Saccharomyces cerevisiae	MIMLkNN	0.362 ± 0.004	804.251 ± 9.862	0.076 ± 0.005	0.099 ± 0.005
		MIMLNN	0.309 ± 0.007	726.703 ± 13.001	0.035 ± 0.003	0.059 ± 0.004
		MIMLSVM	**0.250 ± 0.006**	**564.790 ± 7.773**	0.061 ± 0.004	0.093 ± 0.006
		EnMIMLNNmetric	0.335 ± 0.007	754.132 ± 11.807	0.074 ± 0.005	0.106 ± 0.006
		MIMLDML	0.287 ± 0.011	625.563 ± 27.876	**0.459 ± 0.040**	**0.109 ± 0.011**

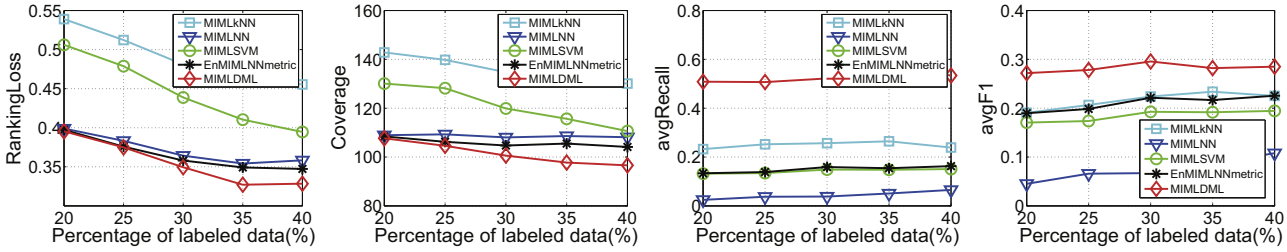

Fig. 8. Experimental results of MIMLDML on *Haloarcula marismortui* dataset.

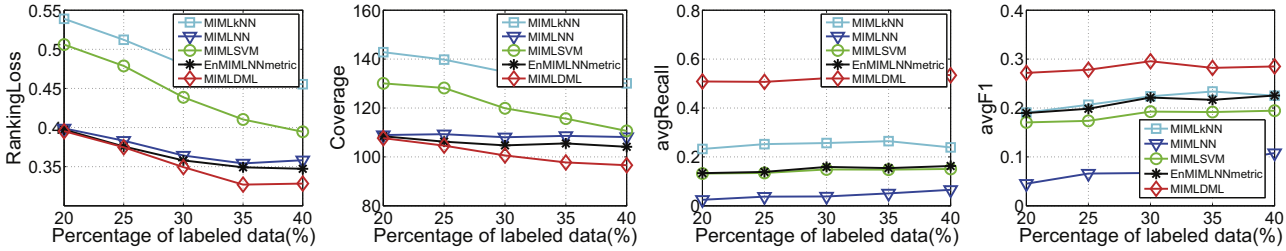

Fig. 9. Experimental results of MIMLDML on *Pyrococcus furiosus* dataset.

In the third experiment, we compare the performance of MIMLDML with other state-of-the-art methods, *i.e.*, MIMLkNN, MIMLNN, MIMLSVM and EnMIMLmetric. For this experiment, 50% of the bags in the dataset are randomly selected as training data and remained bags are treated as testing data. Table 3 reports the experimental results. From the table, we can find that MIMLDML algorithm performs better than all other state-of-the-art methods in terms of all criteria in most cases. Specifically, if we examine the results for each of the criteria individually, we notice that MIMLDML has dramatically improved the avgRecall on all the seven datasets. We also notice that the evaluation criteria used in the experiments measure the learning performance from different

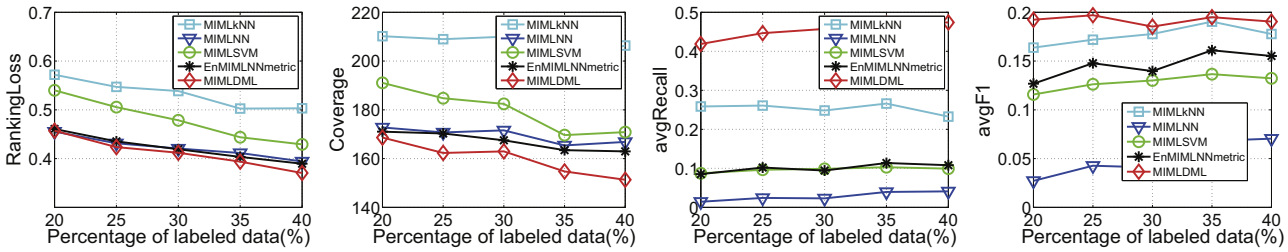

Fig. 10. Experimental results of MIMLDML on *Azotobacter vinelandii* dataset.

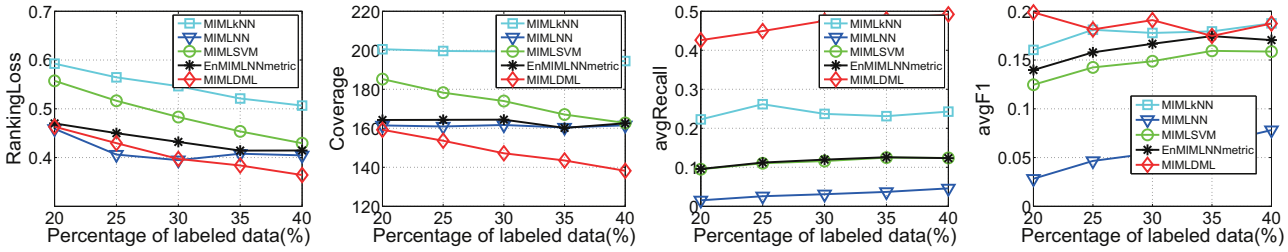

Fig. 11. Experimental results of MIMLDML on *Geobacter sulfurreducens* dataset.

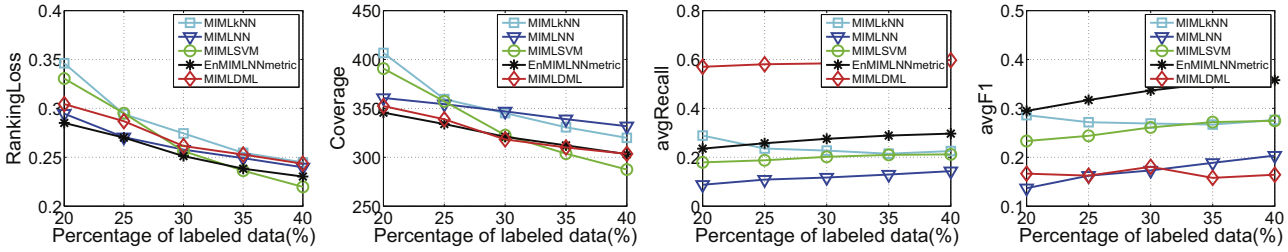

Fig. 12. Experimental results of MIMLDML on *Geobacter sulfurreducens* dataset.

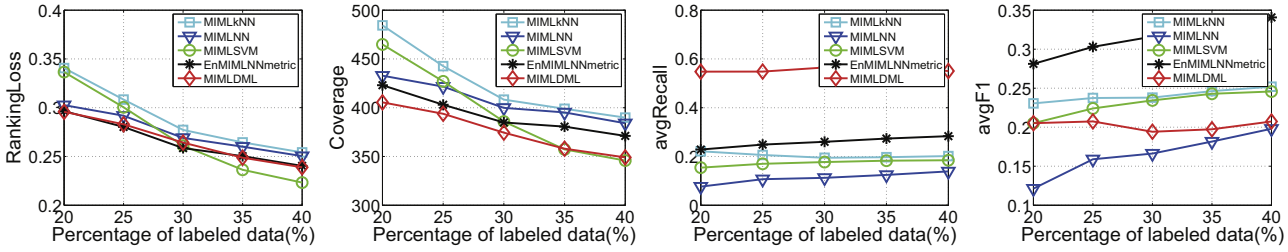

Fig. 13. Experimental results of MIMLDML on *Drosophila melanogaster* dataset.

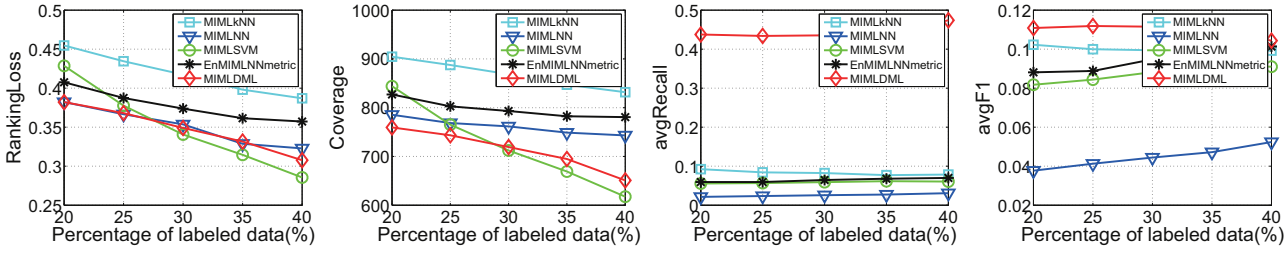

Fig. 14. Experimental results of MIMLDML on *Saccharomyces cerevisiae* dataset.

aspects, and one algorithm rarely outperforms another algorithm on all criteria. If we compare MIMLDML with EnMIMLmetric, we can find that though EnMIMLmetric combines three different Hausdorff distances to denote the distance between bags, EnMIMLmetric performs poorly than MIMLDML on most of the datasets. This may because Hausdorff distance is a pre-defined and data-independent distance which is not able to learn a precise model for MIML problem. Different from EnMIMLmetric, with the advantages of Mahalanobis distance (*i.e.*, unitless, scale-invariant and taking into account the correlations of the data set), MIMLDML can more efficiently preserve and utilize the intrinsic geometric information among the proteins with similar/dissimilar labels. By doing this

Table 4
Runtime comparison (in seconds).

	Datasets	MIMLkNN	MIMLNN	MIMLSVM	EnMIMLNNmetric	MIMLDML
Archaea	*Haloarcula marismortui*	5	2	2	3	17
	Pyrococcus furiosus	9	3	4	5	26
Bacteria	*Azotobacter vinelandii*	9	3	3	5	25
	Geobacter sulfurreducens	8	3	3	4	20
Eukaryote	*Caenorhabditis elegans*	399	73	673	123	540
	Drosophila melanogaster	459	88	949	148	562
	Saccharomyces cerevisiae	810	139	3614	213	1475
	Mean	243	44	750	72	381

way, MIMLDML improves the effectiveness of classification for genome-wide protein function prediction.

For the fourth experiment, to further evaluate the effectiveness of MIMLDML, we test the performances of different compared algorithms with respect to the number of training instances. For each test, the percentage of labeled data varies from 20% to 40% and remained data is used as test data. The performance results for each algorithm on the seven datasets are shown in Figs. 8–14 . From the figures, we find that the avgRecalls of MIMLDML are dramatically higher than other algorithms for all the percentages on all datasets. We notice that MIMLDML performs better than the other state-of-the-art methods on most of the datasets while maintaining the excellent avgRecall. With the increasing of the percentage of labeled data, the performance of MIMLDML becomes more significant. In summary, the experimental results demonstrate the effectiveness of the proposed MIMLDML method.

4.6. Efficiency comparison

The average run time of MIMLkNN, MIMLSVM, MIMLNN, EnMIMLmetric and MIMLDML on seven real-world organisms are recorded and summarized in Table 4. The experiments were implemented in MATLAB R2013a and run on a Windows machine with 4×2.6 G CPU processors and 8 GB memory. From the table, we can find that the average runtime of MIMLDML is higher than MIMLkNN, MIMLNN and EnMIMLNNmetric. This is because we use the same framework as MIMLSVM and most of the lost time is spent on predicting for each class. We also notice that the run time of the MIMLDML algorithms is better than MIMLSVM. This is because MIMLDML benefits from dimensionality reduction. In our experiments, the principal component analysis (PCA) (Jolliffe, 2002) is employed to transform the data from the high-dimensional space to a lower-dimensional space and we only use the first 30 principal components for our method on all the seven datasets.

5. Conclusion

In this paper, we present a multi-instance multi-label distance metric learning approach to address the genome-wide protein function prediction problems. By defining the objective functions on the basis of Mahalanobis distance, instead of Euclidean distance, MIMLDML makes it possible to more efficiently preserve and utilize the intrinsic geometric information among the feature space and label space. With these advantages, MIMLDML can minimize the distance of the instances within each bag and keep the geometry information between bags. In this method, MTLF improves the performance of genome-wide protein function prediction. In addition, we find that the sparseness of label space in some MIML learning datasets. The sparse label space may lead to the unbalanced proportion of positive and negative instance in each class. MIMLDML tries to deal with the sparsely labeled data by giving weight to the labeled data in our learning framework. In this way, MIMLDML improves the average recall rate for genome-wide protein function prediction.

Experimental results on seven real-world organisms covering the biological three-domain system, *i.e.*, archaea, bacteria, and eukaryote, show that the MIMLDML algorithms are superior to most state-of-the-art MIML learning algorithms in most cases. Although MIMLDML works well on genome-wide protein function prediction problems, there are still many issues to be considered. To train an efficient model, we need more labeled data. However, experimental determination of protein structure and function is expensive and time-consuming. In this situation, it is meaningful and challenge to learn for genome-wide protein function prediction with little labeled data. If we have many labeled data in previously annotation tasks, transferring the knowledge information from previously annotated proteins may be a promising method to solve this problem (Pan and Yang, 2010; Mei, 2012). In future work, we would like to expand our framework for these situations.

Acknowledgements

This research was supported by the Guangzhou Key Laboratory of Robotics and Intelligent Softwareunder Grant No. 15180007, and Fundamental Research Funds for the Central Universities under Grant Nos. D215048w and 2015ZZ029, and National Natural Science Foundation of China (NSFC) under Grant Nos. 61005061 and 61502177.

References

Ando, S., 2015. Classifying imbalanced data in distance-based feature space. Knowl. Inf. Syst., 1–24, http://dx.doi.org/10.1007/s10115-015-0846-3.
Andorf, C., Dobbs, D., Honavar, V., 2007. Exploring inconsistencies in genome-wide protein function annotations: a machine learning approach. BMC Bioinform. 8 (1), 284, http://dx.doi.org/10.1186/1471-2105-8-284.
Ashburner, M., Ball, C., Blake, J., et al., 2006. Gene ontology: tool for the unification of biology. The Gene Ontology Consortium Database Resources of the National Center for Biotechnology Information. Nucleic Acids Res. 34, http://dx.doi.org/10.1038/75556.
Bertsekas, D.P., 1999. Nonlinear Programming: 2nd Edition. Athena Scientific, Belmont, Mass.
Chang, C.-C., Lin, C.-J., 2011. LIBSVM: a library for support vector machines. ACM Trans. Intell. Syst. Technol. 2 (3), 27, http://dx.doi.org/10.1145/1961189.1961199.
Cieslak, D.A., Chawla, N.V., 2008. Learning decision trees for unbalanced data. In: Machine Learning and Knowledge Discovery in Databases. Springer, pp. 241–256, http://dx.doi.org/10.1007/978-3-540-87479-9_34.
Davis, J.V., Kulis, B., Jain, P., Sra, S., Dhillon, I.S., 2007. Information-theoretic metric learning. In: Proceedings of the 24th International Conference on Machine Learning, pp. 209–216, http://dx.doi.org/10.1145/1273496.1273523.
Edgar, G., 2007. Measure, Topology, and Fractal Geometry. Springer Science & Business Media, http://dx.doi.org/10.1007/978-1-4757-4134-6.
Jin, R., Wang, S., Zhou, Z.-H.,2009. Learning a distance metric from multi-instance multi-label data. In: IEEE Conference on Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE, pp. 896–902, http://dx.doi.org/10.1109/CVPRW.2009.5206684.
Jolliffe, I., 2002. Principal Component Analysis. Wiley Online Library, http://dx.doi.org/10.1002/9781118445112.stat06472.
Kulis, B., 2012. Metric learning: a survey. Found. Trends Mach. Learn. 5 (4), 287–364, http://dx.doi.org/10.1561/2200000019.

Li, H., Jiang, T., Zhang, K., 2006. Efficient and robust feature extraction by maximum margin criterion. IEEE Trans. Neural Netw. 17 (1), 157–165, http://dx.doi.org/10.1109/TNN.2005.860852.

Long, M., Wang, J., Ding, G., Pan, S.J., Yu, P.S., 2014. Adaptation regularization: a general framework for transfer learning. IEEE Trans. Knowl. Data Eng. 26 (5), 1076–1089, http://dx.doi.org/10.1109/TKDE.2013.111.

Marcotte, E.M., Pellegrini, M., Thompson, M.J., Yeates, T.O., Eisenberg, D., 1999. A combined algorithm for genome-wide prediction of protein function. Nature 402 (6757), 83–86, http://dx.doi.org/10.1038/47048.

Mei, S., 2012. Predicting plant protein subcellular multi-localization by Chou's PseAAC formulation based multi-label homolog knowledge transfer learning. J. Theor. Biol. 310, 80–87, http://dx.doi.org/10.1016/j.jtbi.2013.09.014.

Mostafavi, S., Morris, Q., 2010. Fast integration of heterogeneous data sources for predicting gene function with limited annotation. Bioinformatics 26 (14), 1759–1765, http://dx.doi.org/10.1093/bioinformatics/btq262.

Pan, S.J., Yang, Q., 2010. A survey on transfer learning. IEEE Trans. Knowl. Data Eng. 22 (10), 1345–1359, http://dx.doi.org/10.1109/TKDE.2009.191.

Radivojac, P., Clark, W.T., Oron, T.R., Schnoes, A.M., Wittkop, T., Sokolov, A., Graim, K., Funk, C., Verspoor, K., Ben-Hur, A., et al., 2013. A large-scale evaluation of computational protein function prediction. Nat. Methods 10 (3), 221–227, http://dx.doi.org/10.1038/NMETH.2340.

Wang, J., Zucker, J.-D.,2000. Solving multiple-instance problem: a lazy learning approach. In: Proc. 17th International Conf. on Machine Learning. Morgan Kaufmann, pp. 1119–1125.

Weinberger, K.Q., Blitzer, J., Saul, L.K., 2005. Distance metric learning for large margin nearest neighbor classification. In: Advances in Neural Information Processing Systems., pp. 1473–1480.

Woese, C.R., Fox, G.E., 1977. Phylogenetic structure of the prokaryotic domain: the primary kingdoms. Proc. Natl. Acad. Sci. U. S. A. 74 (11), 5088–5090, http://dx.doi.org/10.1073/pnas.74.11.5088.

Woese, C.R., Magrum, L.J., Fox, G.E., 1978. Archaebacteria. J. Mol. Evol. 11 (3), 245–252, http://dx.doi.org/10.1007/BF01734485.

Woese, C.R., Kandler, O., Wheelis, M.L., 1990. Towards a natural system of organisms: proposal for the domains archaea, bacteria, and eucarya. Proc. Natl. Acad. Sci. U. S. A. 87 (12), 4576–4579, http://dx.doi.org/10.1073/pnas.87.12.4576.

Wu, J., Hu, D., Xu, X., Ding, Y., Yan, S., Sun, X., 2011. A novel method for quantitatively predicting non-covalent interactions from protein and nucleic acid sequence. J. Mol. Graph. Model. 31, 28–34, http://dx.doi.org/10.1016/j.jmgm.2011.08.001.

Wu, Q., Ng, M.K., Ye, Y., 2013. Markov-MIML: a Markov chain-based multi-instance multi-label learning algorithm. Knowl. Inf. Syst. 37 (1), 83–104, http://dx.doi.org/10.1007/s10115-012-0567-9.

Wu, J.-S., Huang, S.-J., Zhou, Z.-H., 2014. Genome-wide protein function prediction through multi-instance multi-label learning. IEEE/ACM Trans. Comput. Biol. Bioinform. 11 (5), 891–902, http://dx.doi.org/10.1109/TCBB.2014.2323058.

Wu, Q., Wang, Z., Li, C., Ye, Y., Li, Y., Sun, N., 2015. Protein functional properties prediction in sparsely-label PPI networks through regularized non-negative matrix factorization. BMC Syst. Biol. 9 (Suppl. 1), S9, http://dx.doi.org/10.1186/1752-0509-9-S1-S9.

Yang, L., Jin, R., 2006. Distance Metric Learning: A Comprehensive Survey. Michigan State University 2.

Yu, G., Domeniconi, C., Rangwala, H., Zhang, G., Yu, Z.,2012. Transductive multi-label ensemble classification for protein function prediction. In: Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM, pp. 1077–1085, http://dx.doi.org/10.1145/2339530.2339700.

Yu, G., Rangwala, H., Domeniconi, C., Zhang, G., Zhang, Z.,2013. Protein function prediction by integrating multiple kernels. In: Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence. AAAI Press, pp. 1869–1875.

Zhang, M.-L., Zhou, Z.-H., 2007. Multi-label learning by instance differentiation. In: AAAI, vol. 7, pp. 669–674.

Zhang, M.-L.,2010. A k-nearest neighbor based multi-instance multi-label learning algorithm. In: 2010 22nd IEEE International Conference on Tools with Artificial Intelligence (ICTAI), vol. 2. IEEE, pp. 207–212, http://dx.doi.org/10.1109/ICTAI.2010.102.

Zhou, Z.-H., Zhang, M.-L., 2010. Multi-instance multi-label learning with application to scene classification. In: Schölkopf, B. (Ed.), In: Advances in Neural Information Processing Systems, vol. 19.

Zhou, Z.-H., Zhang, M.-L., Huang, S.-J., Li, Y.-F., 2012. Multi-instance multi-label learning. Artif. Intell. 176 (1), 2291–2320, http://dx.doi.org/10.1016/j.artint.2011.10.002.