MimlRbf: RBF Neural Networks for Multi-Instance Multi-Label Learning

Min-Ling Zhang* and Zhi-Jian Wang

College of Computer and Information Engineering, Hohai University, Nanjing 210098, China

Abstract. In multi-instance multi-label learning (Miml), each example is not only represented by multiple instances but also associated with multiple class labels. Several learning frameworks, such as the traditional supervised learning, can be regarded as degenerated versions of Miml. Therefore, an intuitive way to solve Miml problem is to identify its equivalence in its degenerated versions. However, this identification process would make useful information encoded in training examples get lost and thus impair the learning algorithm's performance. In this paper, RBF neural networks are adapted to learn from Miml examples. Connections between instances and labels are directly exploited in the process of first layer clustering and second layer optimization. The proposed method demonstrates superior performance on two real-world Miml tasks.

Keywords: Machine learning; Multi-instance multi-label learning; Radial basis function; Scene classification; Text categorization

1 Introduction

Multi-instance multi-label learning (Miml) is a novel learning framework where each example corresponds to *a bag of* instances as well as *a set of* labels [21, 22]. Many real-world learning problems can be properly formalized under this framework. For instance, in image classification, an image generally contains several naturally-partitioned patches each can be represented as an instance, while such an image can be related to multiple semantic classes simultaneously, such as *clouds*, *grassland* and *lions*; In text categorization, each document usually consists of several sections or paragraphs each can be regarded as an instance, while the document may be assigned to a set of predefined topics, such as *sports*, *Beijing Olympics* and *opening ceremony*; In bioinformatics, an gene sequence generally encodes a number of segments each can be expressed an instance, while this sequence may be associated with several functional classes, such as *metabolism*, *transcription* and *protein synthesis*.

The *traditional supervised learning* (Sisl, i.e. single-instance single-label learning) can be viewed as a *degenerated* version of Miml, where each example is restricted to have only single instance and single label. Hence, it is intuitive to solve Miml problem by identifying its equivalence in Sisl via problem reduction. Although this kind of identification strategy is feasible, the performance of the resultant algorithm may suffer from the loss of information incurred during the reduction process. Therefore, whether Miml tasks can be tackled by directly exploiting connections between the instances and the labels of an Miml example is an interesting problem to be further investigated [21, 22].

In this paper, an innovative neural network style algorithm named MimlRbf, i.e. Multi-Instance Multi-Label Radial Basis Function, is proposed. As its name implied, MimlRbf is derived from the popular radial basis function (RBF) method [2]. Briefly, the first layer of MimlRbf neural network consists of *medoids* (i.e. bags of instances) formed by performing k-Medoids clustering on Miml examples for each possible class, where a variant of Hausdorff metric [5] is utilized to measure the distance between bags [19]. Second layer weights of MimlRbf neural network are optimized by minimizing a sum-of-squares error function and worked out through singular value decomposition (SVD) [12].

In the stage of first layer clustering, different clusters

*Corresponding author. Tel.: +86-25-83786520; Fax: +86-25-83787793. *Email address:* zhangml@hhu.edu.cn

are formed for MIML examples from each possible class. In this way, the inherent structure of each class is encoded in the medoids returned by the clustering algorithm. In the stage of second layer optimization, medoids from all classes are involved in calculating the weights associated with each class. In this way, the connections between the instances and the labels of an MIML example is explicitly addressed by MIMLRBF.

The rest of this paper is organized as follows. Section 2 reviews related works. Section 3 presents MIMLRBF. Section 4 reports experimental results on two real-world MIML data sets. Finally, Section 5 concludes and indicates some issues for future work.

2 Related Works

In this section, we first give the formal definition of MIML and then review existing MIML algorithms. Let $\mathcal{X} = \mathbb{R}^d$ denote the input space of instances and $\mathcal{Y} = \{1, 2, \ldots, Q\}$ be the set of class labels. The task of MIML is to learn a function $f_{\text{MIML}} : 2^{\mathcal{X}} \to 2^{\mathcal{Y}}$ from a set of MIML training examples $\{(X_i, Y_i)|1 \leq i \leq N\}$, where $X_i \subseteq \mathcal{X}$ is a bag of instances $\{\mathbf{x}_1^i, \mathbf{x}_2^i, \ldots, \mathbf{x}_{n_i}^i\}$ and $Y_i \subseteq \mathcal{Y}$ is a set of labels $\{y_1^i, y_2^i, \ldots, y_{l_i}^i\}$ associated with X_i. Here n_i is the number of instances in X_i and l_i the number of labels in Y_i. The MIML framework is closely related to the learning frameworks of *multi-instance learning* [4], *multi-label learning* [11, 13] and traditional supervised learning.

Multi-instance learning [4] (MISL, i.e. multi-instance single-label learning) was coined by Dietterich et al. in their investigation of drug activity prediction problem. The task of MISL is to learn a function $f_{\text{MISL}} : 2^{\mathcal{X}} \to \{+1, -1\}$ from a set of MISL training examples $\{(X_i, y_i)|1 \leq i \leq N\}$, where $X_i \subseteq \mathcal{X}$ is a bag of instances $\{\mathbf{x}_1^i, \mathbf{x}_2^i, \ldots, \mathbf{x}_{n_i}^i\}$ and $y_i \in \{+1, -1\}$ is the binary label of X_i. Multi-label learning [11, 13] (SIML, i.e. single-instance multi-label learning) originated from the investigation of text categorization problems. The task of SIML is to learn a function $f_{\text{SIML}} : \mathcal{X} \to 2^{\mathcal{Y}}$ from a set of SIML training examples $\{(\mathbf{x}_i, Y_i)|1 \leq i \leq N\}$, where $\mathbf{x}_i \in \mathcal{X}$ is an instance and $Y_i \subseteq \mathcal{Y}$ is a set of labels $\{y_1^i, y_2^i, \ldots, y_{l_i}^i\}$ associated with $\mathbf{x}_i$. Considerable research results have been achieved on both learning frameworks and related works can be found in [20] for MISL and [16] for SIML, together with references therein.

It is obvious that traditional supervised learning (SISL) can be regarded as a degenerated version of either MISL or SIML. Furthermore, according to the above definitions, SISL, MISL and SIML are all degenerated versions of MIML. Therefore, a natural way of solving MIML problem is to identify its equivalence in SISL, using MISL or SIML as the bridge. Actually, while formalizing the MIML framework, Zhou and Zhang [21] adopted this strategy and proposed two MIML algorithms named MIMLBOOST and MIMLSVM.

MIMLBOOST reduces the MIML problem into an SISL one using MISL as the intermediate link. Firstly, MIMLBOOST transforms the original MIML task into an MISL one by converting each MIML example (X_i, Y_i) into $|\mathcal{Y}|$ number of MISL examples $\{([X_i, y], \Phi[X_i, y])\,|y \in \mathcal{Y}\}$. Here, $[X_i, y]$ contains n_i instances $\{(\mathbf{x}_1^i, y), \ldots, (\mathbf{x}_{n_i}^i, y)\}$ formed by concatenating each of X_i's instance with label y, while $\Phi[X_i, y] = +1$ if $y \in Y_i$ and -1 otherwise. After that, MIMLBOOST solves the derived MISL problem by employing a specific algorithm named MIBOOSTING [18]. This algorithm deals with MISL problem by reducing it into an SISL one under the assumption that each instance in the bag contributes equally and independently to a bag's label.

In contrast to MIMLBOOST, MIMLSVM reduces the MIML problem into an SISL one using SIML as the intermediate link other than MISL. Firstly, MIMLSVM transforms the original MIML task into an SIML one by converting each MIML example (X_i, Y_i) into an SIML example $(\tau(X_i), Y_i)$. Here, the function $\tau(\cdot)$ maps a bag of instances X_i into a single instance $\mathbf{z}_i$ with the help of *constructive clustering* [21]. Specifically, k-medoids clustering is performed on $\Lambda = \{X_1, X_2, \ldots, X_N\}$ at the level of bags and components of $\mathbf{z}_i$ correspond to the distances between X_i and the medoids of the clustered groups. After that, MIMLSVM solves the derived SIML problem by employing a specific algorithm named MLSVM [3]. This algorithm deals with SIML problem by decomposing it into a number of independent SISL problems (one per class), where instance $\mathbf{x}_i$ associated with label set Y_i will be regarded as positive instance when building classifier for class $y \in Y_i$ while regarded as negative instance when building classifier for class $y \notin Y_i$.

Considering that either MIMLBOOST or MIMLSVM may lose information during the reduction process, Zhou et al. [22] proposed another method for MIML called D-MIMLSVM based on regularization. D-MIMLSVM assumes that the classification system $\boldsymbol{f}$ is formed by $|\mathcal{Y}|$ functions $f_y : 2^{\mathcal{X}} \to \{+1, -1\}$ $(y \in \mathcal{Y})$, each determining whether label y can be associated with bag $X \subseteq \mathcal{X}$.

D-MimlSvm defines an objective function over $\boldsymbol{f}$ which balances the loss between the labels and predictions on the bags as well as on the constituent instances. Furthermore, each function f_y is set to be a linear model in a feature space induced by some kernel k defined on $2^{\mathcal{X}}$, such as the *set kernel* [6]. The resultant non-linear optimization problem is solved by a standard constraint concave-convex procedure (Cccp) [15] and its efficiency is further improved by utilizing *cutting plane* techniques [8]. Note that although D-MimlSvm may achieve better performance than those degenerated algorithms, it can only deal with moderate size of training set due to the associated demanding optimization problem [22].

In addition to the design of various Miml algorithms, there are some recent research progresses on Miml framework, such as metric learning from Miml data [7] as well as applications of Miml techniques in bioinformatics [9].

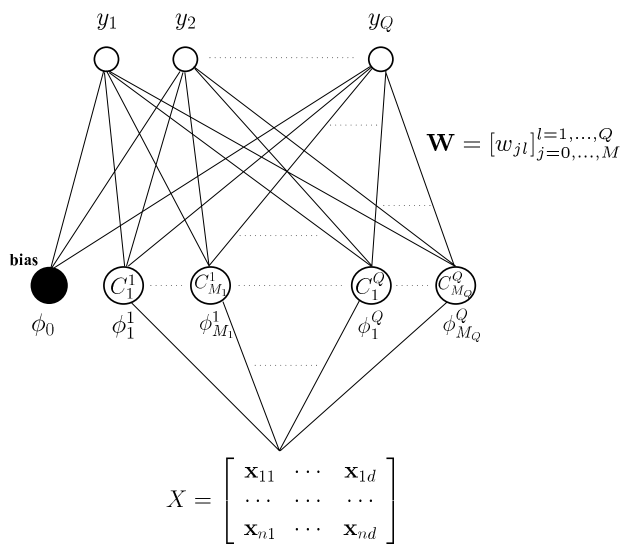


Figure 1: Architecture of an MimlRbf neural network.

3 The Proposed Approach

3.1 Neural Network Architecture

Generally, an RBF neural network is composed of two layers of units. In the first layer, each hidden unit (basis function) is associated with a prototype vector; In the second layer, each output unit corresponds to a possible class. In most cases, parameters of basis functions are determined using unsupervised methods while weights between first and second layers are obtained by solving a linear problem. Detailed description and theoretical foundations of RBF neural networks can be found in [2].

Figure 1 illustrates the typical architecture of an MimlRbf neural network. As shown in the figure, there are two major architectural differences between the MimlRbf and the conventional RBF neural networks. Firstly, the input to an MimlRbf neural network is a bag X consisting of n instances $\{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n\}$, where each instance $\mathbf{x}_k$ is a d-dimensional feature vector $[\mathbf{x}_{k1}, \mathbf{x}_{k2}, \dots, \mathbf{x}_{kd}]^{\mathrm{T}}$. For a conventional RBF neural network however, its input usually takes the form of *single* feature vector; Secondly, the first layer of an MimlRbf neural network is composed of Q sets of bags $\bigcup_{l=1}^{Q}\{C_1^l, C_2^l, \dots, C_{M_l}^l\}$, where M_l is the number of bags retained for the l-th class and the total number of bags in the first layer then equals $M = \sum_{l=1}^{Q} M_l$. For a conventional RBF neural network however, each node in its first layer usually stores a prototype vector acting as the center of basis function $\phi(\cdot)$.

In addition, each output unit of an MimlRbf neural network is related to a possible class label. The weights $\mathbf{W} = [w_{jl}]_{(M+1)\times Q}$ between first and second layers are shown as lines from basis functions to output units. Furthermore, the biases are shown as weights w_{0l} from an extra basis function ϕ_0 whose output is fixed at 1.

Similar as conventional RBF neural networks, MimlRbf is also trained with a two-stage procedure. In the first stage, for each possible class $l \in \mathcal{Y}$, Miml training examples with label l are clustered by invoking k-Medoids and then the medoids of the clustered groups are retained in the first layer. In the second stage, weights between first and second layers are optimized by minimizing a sum-of-squares error function. The two stages are scrutinized in the following subsection in succession.

3.2 Training and Testing

Adopting the notations in Section 2, $S = \{(X_i, Y_i)|1 \leq i \leq N\}$ corresponds to the Miml training set. Let $U_l = \{X_i|(X_i, Y_i) \in S, l \in Y_i\}$ denote the set of Miml examples with the l-th label. Then, by regarding each bag as an atomic object, the popular k-Medoids clustering algorithm is adapted to partition U_l into M_l disjoint groups of bags. Previous works reveal that Hausdorff metric [5] is an effective measure to calculate distances between objects with multi-instance representations [17, 19]. In this paper, a variant of Hausdorff metric called *average Hausdorff distance* [19] is used, which is shown to be less sensitive to outlying points compared to *maximal* Hausdorff distance [5], and be able of considering more geometric relationships between instances in the bags compared to *minimal* Hausdorff distance [17].

Formally, given two bags of instances $A = \{\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_{n_a}\}$ and $B = \{\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_{n_b}\}$, the average Hausdorff distance between A and B is defined as:

$$\mathrm{aveH}(A, B) = \frac{\sum_{\mathbf{a} \in A} \min_{\mathbf{b} \in B} \mathrm{dist}(\mathbf{a}, \mathbf{b}) + \sum_{\mathbf{b} \in B} \min_{\mathbf{a} \in A} \mathrm{dist}(\mathbf{b}, \mathbf{a})}{|A| + |B|} \tag{1}$$

where $|\cdot|$ gives the cardinality of a set and $\mathrm{dist}(\cdot, \cdot)$ measures the distance between two instances (taking the form of Euclidean distance in this paper). Conceptually, for each instance in one bag, $\mathrm{aveH}(\cdot, \cdot)$ calculates the distance to its nearest neighbor in the other bag. After traversing all the instances in both bags, the average value out of the above distances is reported as $\mathrm{aveH}(\cdot, \cdot)$'s output.

Combined with average Hausdorff distance, k-Medoids algorithm is employed to partition U_l into M_l disjoint groups of bags G_j^l $(1 \leq j \leq M_l)$, whose medoids C_j^l are intuitively determined as:

$$C_j^l = \arg \min_{A \in G_j^l} \sum\nolimits_{B \in G_j^l} \mathrm{aveH}(A, B) \tag{2}$$

In this paper, the number of retained medoids M_l for each class is set to be fraction α of the number of Miml examples in U_l, i.e. $M_l = \alpha \times |U_l|$.

Exploiting those medoids from each possible class, the weight matrix $\mathbf{W} = [w_{jl}]_{(M+1) \times Q}$ between first and second layers is optimized by minimizing the following sum-of-squares error function:

$$E = \frac{1}{2} \sum_{i=1}^{N} \sum_{l=1}^{Q} \left(y_l(X_i) - t_l^i \right)^2 \tag{3}$$

where t_l^i is the desired output of X_i on the l-th class, which takes the value of +1 if $l \in Y_i$ and -1 otherwise. Correspondingly, $y_l(X_i) = \sum_{j=0}^{M} w_{jl} \phi_j(X_i)$ is actual output of X_i on the l-th class, where $\phi_j(X_i)$ is the activation of the j-th basis function on X_i. Similar as conventional RBF neural network, the basis function $\phi_j(\cdot)$ makes Gaussian style activation as follows:

$$\phi_j(X_i) = \exp\left(-\frac{\mathrm{aveH}(X_i, C_j)^2}{2\sigma_j^2} \right) \quad (1 \leq j \leq M) \tag{4}$$

Here $\mathrm{aveH}(X_i, C_j)$ measures the average Hausdorff distance between X_i and the j-th medoid in MimlRbf's first layer. In addition, the activation of $\phi_0(X_i)$ is fixed at 1 and w_{0l} acts as the bias value for the l-th class. The standard deviation σ_j is a parameter controlling the smoothness of the basis function $\phi_j(\cdot)$. In this paper, all the σ_j $(1 \leq j \leq M)$ take the same value of σ, which is some multiple of the average distance between each pair of medoids in the first layer:

$$\sigma = \mu \times \left(\frac{\sum_{p=1}^{M-1} \sum_{q=p+1}^{M} \mathrm{aveH}(C_p, C_q)}{M(M-1)/2} \right) \tag{5}$$

where μ is the parameter of scaling factor.

Differentiating the error function of Eq.(3) with respect to w_{jl} and setting the derivative to zero gives the normal equations for the least sum-of-squares problem as follows:

$$\left(\mathbf{\Phi}^{\mathrm{T}} \mathbf{\Phi} \right) \mathbf{W} = \mathbf{\Phi}^{\mathrm{T}} \mathbf{T} \tag{6}$$

Here $\mathbf{\Phi} = [\phi_{ij}]_{N \times (M+1)}$ with elements $\phi_{ij} = \phi_j(X_i)$, $\mathbf{W} = [w_{jl}]_{(M+1) \times Q}$ with elements w_{jl}, and $\mathbf{T} = [t_{il}]_{N \times Q}$ with elements $t_{il} = t_l^i$. In this paper, the weight parameters $\mathbf{W}$ are computed by solving Eq.(6) using linear matrix inversion techniques of SVD [12].

Y=MimlRbf(S, α, μ, X)

Inputs:

S: the Miml training set $\{(X_1, Y_1), \ldots, (X_N, Y_N)\}$

α: the fraction parameter

μ: the scaling factor

X: the test Miml example $(X \subseteq \mathcal{X})$

Outputs:

Y: predicted label set for X $(Y \subseteq \mathcal{Y})$

Process:

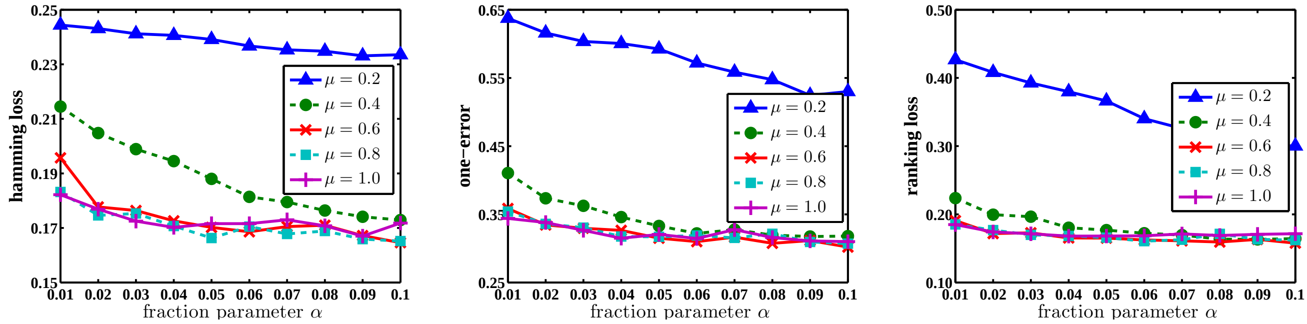
for $l \in \mathcal{Y}$ **do**
    Set $U_l = \{X_i | (X_i, Y_i) \in S, l \in Y_i\}$;
    Partition U_l into $M_l = \alpha \times |U_l|$ groups of bags G_j^l using k-Medoids with distance metric of Eq.(1);
    Determine medoid C_j^l of each group using Eq.(2);
Form matrix $\mathbf{\Phi}$ (using Eqs.(4) and (5)) and $\mathbf{T}$;
Compute weights $\mathbf{W}$ by solving Eq.(6) with SVD;
$Y = \{l | y_l(X) = \sum_{j=0}^{M} w_{jl} \phi_j(X) > 0, l \in \mathcal{Y}\}$;

Figure 2: Pseudo code of MimlRbf.

Figure 2 gives the complete description of MimlRbf. Firstly, the first layer of MimlRbf neural network is

Table 1: Characteristics of the data sets.

				Instances per bag			Labels per example (k)		
Data set	#examples	#classes	#features	min	max	mean±std.	k=1	k=2	k≥3
Scene	2,000	5	15	9	9	9.00±0.00	1,543	442	15
Reuters	2,000	7	243	2	26	3.56±2.71	1,701	290	9



(a) *hamming loss* (b) *one-error* (c) *ranking loss*

Figure 3: The performance of MimlRbf on the scene data changes as α increases under different values of μ.

formed by performing k-Medoids clustering on training examples of each possible class, where average Hausdorff metric is employed to measure distance between bags (steps 1 to 4); After that, the weights between first and second layers are calculated through minimizing the sum-of-squares error function as shown in Eq.(3) (steps 5 to 6); Finally, the test Miml example is fed to the trained neural network for prediction (step 7).

4 Experiments

4.1 Experimental Setup

The performance of MimlRbf is compared with MimlBoost and MimlSvm on two real-world Miml tasks. The first is the benchmark scene classification task studied in seminal works on Miml framework [19]. The scene classification data contains 2,000 natural scene images. All the possible class labels are *desert*, *mountains*, *sea*, *sunset* and *trees* and a set of labels is manually assigned to each image. Images with multiples labels comprise over 22% of the data set and the average number of labels per image is 1.24±0.44. Each image is represented as a bag of nine 15-dimensional instances using the Sbn image bag generator [10], where each instance corresponds to an image patch. The data set with detailed description can be found at http://lamda.nju.edu.cn/datacode/miml-image-data.htm.

In addition to scene classification, we have also studied another Miml task of text categorization. Specifically, the widely used Reuters-21578 collection [14] is employed in experiment. The seven most frequent categories are considered. After removing documents without labels or main texts and randomly removing some documents with only one label, a text categorization data set containing 2,000 documents is obtained. Documents with multiple labels comprise around 15% of the data set and the average number of labels per document is 1.15±0.37. Each document is represented as a bag of instances based on the techniques of sliding windows [1], where each instance corresponds to a text segment enclosed in a sliding window of size 50 (overlapped with 25 words). Function words are excluded from the vocabulary and stemming is performed on the remaining words. Instances in the bags adopt the *"Bag-of-Words"* representation based on term frequency [14]. Without loss of effectiveness, dimensionality reduction is conducted and the top 2% words with highest document frequency are retained. Finally, each instance is represented as a 243-dimensional feature vector. The data set with detailed description can be found at http://lamda.nju.edu.cn/datacode/miml-text-data.htm.

Table 1 summarizes characteristics of both data sets. Since MIML algorithms make multi-label predictions, their performance are evaluated with five popular multi-label metrics, i.e. *hamming loss*, *one-error*, *coverage*, *ranking loss* and *average precision*. For the first four metrics, the *smaller* the value the better the performance. For *average precision*, the *bigger* the value the better the performance. Details on the metrics can be found in [13].

4.2 Experimental Results

MIMLRBF is compared with MIMLBOOST and MIMLSVM, both of which adopt the best parameters as reported in [21]. Concretely, the number of boosting rounds for MIMLBOOST is set to 25 while Gaussian kernel with width parameter of 0.2 is used to implement MIMLSVM. The performance of each compared algorithm is evaluated by conducting *tenfold cross-validation* on the given data set.

As shown in Figure 2, the MIMLRBF algorithm involves two different parameters, i.e. the fraction parameter α and the scaling factor μ. Figure 3 illustrates how MIMLRBF performs on the scene data under different parameter configurations, where the performance is evaluated in terms of *hamming loss*, *one-error* and *ranking loss*[1]. Here, α increases from 1% to 10% with an interval of 1% and μ varies from 0.2 to 1.0 with an interval of 0.2.

It is obvious from Figure 3 that, when μ is fixed, the performance of MIMLRBF improves evidently in the *initial* increasing phase of α ($\alpha < 4\%$) and then tends to level up in the remaining increasing phase of α. Furthermore, when α is fixed, the performance of MIMLRBF will not significantly change as long as μ is greater than 0.4. Therefore, in this paper, MIMLRBF is implemented by setting the fraction parameter α to 10% and the scaling factor μ to moderate value of 0.6.

Table 2 and Table 3 summarize the experimental results of each compared algorithm on the scene data and Reuters data respectively. For each evaluation criterion, "↓" indicates "the smaller the better" while "↑" indicates "the bigger the better". Furthermore, the best result on each evaluation criterion is highlighted in boldface. It is obvious from Tables 2 and 3 that MIMLRBF performs quite well on both data sets in terms of all metrics.

Specifically, pairwise t-tests at 0.01 significance level are further conducted to test whether the performance difference between two algorithms is statistically significant. On the scene data set, MIMLRBF is comparable to MIMLBOOST in terms of *one-error* while achieves superior performance than MIMLBOOST on the other evaluation metrics. In addition, MIMLRBF outperforms MIMLSVM on the scene data in terms of all the evaluation criterions. On the Reuters data set, it is even more impressive that MIMLRBF is superior to both MIMLBOOST and MIMLSVM in terms of all the concerned metrics.

Furthermore, Table 4 reports the training and testing time consumed by each compared algorithm on both data sets[2]. As shown by Table 4, the training and testing efficiency of MIMLRBF is slightly better than MIMLSVM while far superior than MIMLBOOST. All the above results indicate that MIMLRBF turns out to be a very effective as well as efficient approach to the task of MIML.

Table 2: Experimental results on the scene data set.

Evaluation Metric	Compared Algorithm MIMLRBF	MIMLBOOST	MIMLSVM
hamming loss↓	**0.165±0.009**	0.189±0.007	0.185±0.011
one-error↓	**0.302±0.024**	0.335±0.021	0.347±0.026
coverage↓	**0.894±0.066**	0.947±0.056	1.031±0.068
ranking loss↓	**0.158±0.015**	0.172±0.011	0.191±0.015
average precision↑	**0.806±0.015**	0.785±0.012	0.774±0.015

Table 3: Experimental results on the Reuters data set.

Evaluation Metric	Compared Algorithm MIMLRBF	MIMLBOOST	MIMLSVM
hamming loss↓	**0.034±0.005**	0.053±0.009	0.043±0.007
one-error↓	**0.058±0.020**	0.107±0.022	0.101±0.024
coverage↓	**0.272±0.055**	0.417±0.047	0.379±0.062
ranking loss↓	**0.017±0.006**	0.039±0.007	0.033±0.008
average precision↑	**0.966±0.011**	0.930±0.012	0.937±0.015

Table 4: Time cost of each compared algorithm on both data sets (*measured in minutes*).

		Compared Algorithm MIMLRBF	MIMLBOOST	MIMLSVM
Training Phase	*Scene*	4.95±0.04	6462.40±86.40	6.12±0.09
	Reuters	2.42±0.10	4354.98±64.89	3.01±0.12
Testing Phase	*Scene*	0.16±0.01	467.73±4.42	0.52±0.02
	Reuters	0.10±0.02	369.24±12.18	0.22±0.04

[1] Experimental results on other metrics yield same observations.

[2] An HP sever with 4G RAM and four Intel Xeron™ CPUs each running at 2.80GHz is used to conduct the experiments.

5 Conclusion

In this paper, a novel algorithm MimlRbf deriving from conventional RBF neural networks is proposed for Miml. In both stages of first layer clustering and second layer optimization, connections between the instances and the labels of Miml examples are explicitly addressed. Applications to two real-world Miml tasks show that MimlRbf is highly competitive to other Miml algorithms.

Investigate whether better performance can be achieved by setting different value of α for each possible class is an interesting future work. In addition, designing other kinds of Miml algorithms is worth further study.

Acknowledgements

The authors wish to thank the associate editor and the anonymous reviewers for their helpful comments. This work is supported by the National Science Foundation of China (60805022), Ph.D. Programs Foundation of Ministry of Education of China (200802941009), Open Foundation of National Key Laboratory for Novel Software Technology of China (KFKT2008B12), and Startup Foundation for Excellent New Faculties of Hohai University.

References

[1] S. Andrews, I. Tsochantaridis, and T. Hofmann. Support vector machines for multiple-instance learning. In S. Becker, S. Thrun, and K. Obermayer, editors, *Advances in Neural Information Processing Systems 15*, pages 561–568. MIT Press, Cambridge, MA, 2003.

[2] C. M. Bishop. *Neural Networks for Pattern Recognition.* Oxford University Press, New York, 1995.

[3] M. R. Boutell, J. Luo, X. Shen, and C. M. Brown. Learning multi-label scene classification. *Pattern Recognition*, 37(9):1757–1771, 2004.

[4] T. G. Dietterich, R. H. Lathrop, and T. Lozano-Pérez. Solving the multiple-instance problem with axis-parallel rectangles. *Artificial Intelligence*, 39(1-2):31–71, 1997.

[5] G. A. Edgar. *Measure, Topology, and Fractal Geometry, 3rd print.* Springer-Verlag, Berlin, 1995.

[6] T. Gärtner, P. A. Flach, A. Kowalczyk, and A. J. Smola. Multi-instance kernels. In *Proceedings of the 19th International Conference on Machine Learning*, pages 179–186, Sydney, Australia, 2002.

[7] R. Jin, S. Wang, and Z.-H. Zhou. Learning a distance metric from multi-instance multi-label data. In *Proceedings of the 2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Miami, FL, 2009.

[8] J. E. Kelley. The cutting-plane method for solving convex programs. *Journal of the Society for Industrial and Applied Mathematics*, 8(4):703–712, 1960.

[9] Y.-X. Li, S. Ji, J. Ye, S. Kumar, and Z.-H. Zhou. Drosophila gene expression pattern annotation through multi-instance multi-label learning. In *Proceedings of the 21st International Joint Conference on Artificial Intelligence*, Pasadena, CA, 2009.

[10] O. Maron and A. L. Ratan. Multiple-instance learning for natural scene classification. In *Proceedings of the 15th International Conference on Machine Learning*, pages 341–349, Madison, WI, 1998.

[11] A. McCallum. Multi-label text classification with a mixture model trained by EM. In *Working Notes of the AAAI'99 Workshop on Text Learning*, Orlando, FL, 1999.

[12] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. *Numerical Recipes in C: the Art of Scientific Computing.* Cambridge University Press, New York, 1992.

[13] R. E. Schapire and Y. Singer. Boostexter: a boosting-based system for text categorization. *Machine Learning*, 39(2/3):135–168, 2000.

[14] F. Sebastiani. Machine learning in automated text categorization. *ACM Computing Surveys*, 34(1):1–47, 2002.

[15] A. J. Smola, S. V. N. Vishwanathan, and T. Hofmann. Kernel methods for missing variables. In *Proceedings of the 10th International Workshop on Artificial Intelligence and Statistics*, pages 325–332, Barbados, 2005.

[16] G. Tsoumakas and I. Katakis. Multi-label classification: an overview. *International Journal of Data Warehousing and Mining*, 3(3):1–13, 2007.

[17] J. Wang and J.-D. Zucker. Solving the multiple-instance problem: a lazy learning approach. In *Proceedings of the 17th International Conference on Machine Learning*, pages 1119–1125, Stanford, CA, 2000.

[18] X. Xu and E. Frank. Logistic regression and boosting for labeled bags of instances. In H. Dai, R. Srikant, and C.-Q. Zhang, editors, *Lecture Notes in Computer Science 3056*, pages 272–281. Springer, Berlin, 2004.

[19] M.-L. Zhang and Z.-H. Zhou. Multi-instance clustering with applications to multi-instance prediction. *Applied Intelligence*, in press.

[20] Z.-H. Zhou. Multi-instance learning: a survey. Technical report, AI Lab, Department of Computer Science & Technology, Nanjing University, Nanjing, China, 2004.

[21] Z.-H. Zhou and M.-L. Zhang. Multi-instance multi-label learning with application to scene classification. In B. Schölkopf, J. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems 19*, pages 1609–1616. MIT Press, Cambridge, MA, 2007.

[22] Z.-H. Zhou, M.-L. Zhang, S.-J. Huang, and Y.-F Li. MIML: A framework for learning with ambiguous objects. *CORR abs/0808.3231*, 2008.