Improvement of Learning Algorithm for the Multi-instance Multi-label RBF Neural Networks Trained with Imbalanced Samples

CUNHE LI, GUOQIANG SHI
College of Computer and Communication Engineering
China University of Petroleum
Qingdao, 266555 P.R. China

Multi-instance multi-label learning (MIML) is a novel learning framework where each sample is represented by multiple instances and associated with multiple class labels. In several learning situations, the multi-instance multi-label RBF neural networks (MIMLRBF) can exploit connections between the instances and the labels of an MIML example directly. However, it is quite often that the numbers of samples in different categories are discrete, i.e., the class distribution is imbalanced. When an MIMLRBF is trained with imbalanced samples, it will produce poor performance for setting the consistent fraction parameter \( \alpha \) for all classes. This paper presents an improved approach in learning algorithms used for training MIMLRBF with imbalanced samples. In the first cluster stage, the methodology calculates the initial medoids for each category based on the data density. Afterwards, \( k \)-medoids is been invoked to optimize the medoids. The network will take advantage of the well-adjusted units. In the second stage, the weights between the first and second layer are optimized by the singular value decomposition method. The improved approaches could be used in applications with imbalanced samples. Comparing results employing diverse learning strategies shows interesting outcomes as have come out of this paper.

Keywords: Machine learning; Radial basis function; Multi-instance multi-label learning; Class imbalance

1. INTRODUCTION

In traditional supervised learning, each example is denoted by an instance, and associated with a single class label (Single-Instance Single-Label learning, SISL). However, numerous real-world learning problems cannot be resolved directly under this framework. In many cases, it is possible to represent a complicated object using a set of instances rather than a single instance. For instance, a web page usually consists of several parts, such as hyperlinks, content, copyright etc., where each part can be described by an instance, and the page may be assigned to a set of predefined labels. Such problems can be formalized properly under multi-instance multi-label learning (MIML) where each sample is represented by multiple instances and associated with multiple class labels [1].

Compared to SISL learning frameworks, the MIML framework is more convenient and natural for representing complicated objects. The SISL learning can be regarded as a reduced version of MIML learning. Hence, the intuitive way to solve MIML problem is identifying its equivalence in SISL via problem reduction. However, the reduction process may lose some useful information. Therefore, the MIML tasks can be tackled by
directly exploiting connections between the instances and the labels of an MIML example [2].

Neural networks are commonly used in many areas, such as time series prediction [3], text categorization [4], image reconstruction [5], etc. Multi-instance multi-label radial basis function neural networks (MIMLRBF) which are derived from traditional radial basis function (RBF) neural networks [6], can exploit connections between the instances and the labels of an MIML example directly [2]. However, when the number of samples of some classes is much smaller than those in the other classes, the class imbalance problem will appear, and this problem is very common in MIML training set. This problem will result in poor performance by trained MIMLRBF networks. This paper presents an improved MIMLRBF named IMIMLRBF is proposed to relieve this problem by attaining balanced numbers of hidden units for different categories.

Briefly, the first layer of a MIMLRBF neural network consists of medoids (i.e., bags of instances) formed by performing an improved k-medoids clustering on an MIML data set, where the number and medoids of clusters are initialized by an algorithm based on average Hausdorff distance. In the second stage, the weights of a MIMLRBF neural network are assigned through singular value decomposition (SVD) [2].

The organization of this paper is as follows. Section 2 reviews related works. Section 3 presents the IMIMLRBF in detail. Section 4 reports experimental results. Finally, Section 5 concludes and indicates some issues for future work.

2. RELATED WORK

Firstly, we introduce the formal definition of multi-instance, multi-label and MIML learning.

Multi-instance learning [7] (MIL) studies the problem where a real-world object described by a number of instances is associated with one class label. Given a data set \( \{ (X_1, y_1), (X_2, y_2), \ldots, (X_m, y_m) \} \), where \( X_i \subseteq \mathcal{X} \) is a set of instances \( \{ x_{1_i}, x_{2_i}, \ldots, x_{n_i} \} \), \( x_{j_i} \in \mathcal{X} \) \( (j = 1, 2, \ldots, n_i) \), \( y_i \in \{-1, +1\} \) is the label of \( X_i \). The task is to learn a function \( f_{MIL}: 2^{\mathcal{X}} \rightarrow \{-1, +1\} \) [1]. Multi-label learning [8] (MLL) studies the problem where a real-world object described by one instance is associated with a number of class labels. Formally, the task is to learn a function \( f_{MLL}: \mathcal{X} \rightarrow 2^{\mathcal{Y}} \) from a given data set \( \{(x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m)\} \), where \( x_i \in \mathcal{X} \) is an instance and \( Y_i \subseteq \mathcal{Y} \) is a set of labels \( \{y_{1_i}, y_{2_i}, \ldots, y_{n_i}\} \), \( y_{k_i} \in \mathcal{Y} \) \( (k = 1, 2, \ldots, l_i) \). Both of the MIL and MLL have been successfully applied to diverse applications including scene classification [9][10].

The following is the definition of MIML learning. 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 \( f_{MIML}: 2^{\mathcal{X}} \rightarrow 2^{\mathcal{Y}} \) 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 \( \{x_{1_i}, x_{2_i}, \ldots, x_{n_i}\} \) and \( Y_i \subseteq \mathcal{Y} \) is a set of labels \( \{y_{1_i}, y_{2_i}, \ldots, y_{n_i}\} \) associated with \( X_i \). Here \( n_i \) is the number of instances in \( X_i \) and \( l_i \) is the number of labels in \( Y_i \) [2]. It is obvious that traditional supervised learning (SISL, i.e., single-instance single-label learning) can be regarded as a degenerate version of either MIL or MLL which is degenerate version of MIML. Therefore, there are three ways to solve MIML problems:
(1) Conversion the MIML problem into an SISL one using MIL as the intermediate link: MIMLBOOST [1].

(2) Conversion the MIML problem into an SISL one using MLL as the intermediate link: MIMLSVM [1].

(3) Designing the original MIML algorithms: D-MIMLSVM [11], MIMLRBF [2].

The first two methods convert the MIML problem into an SISL problem using MIL or MLL as the intermediate link, so we can solve the MIML problem using traditional SISL algorithms. However, this kind of method may lose information during the converting process. People began to designing the original MIML algorithms for this reason. Zhou et al. [11] proposed D-MIMLSVM based on regularization. D-MIMLSVM assumes that the classification system $f$ is formed by $[\mathcal{Y}]$ functions $f_y: 2^X \rightarrow \{+1, -1\}$ ($y \in \mathcal{Y}$), each determining whether label $y$ can be associated with bag $X \subseteq X$. D-MIMLSVM defines an objective function over $f$ which balances the loss between the labels and predictions on the bags and 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 $2^X$, such as the set kernel [12]. The resultant non-linear optimization problem is solved by a standard constraint concave convex procedure [13] and its efficiency is further improved by utilizing cutting plane techniques [14]. Although D-MIMLSVM may achieve better performance than those degenerate algorithms, it can only deal with moderate size of training set due to the associated demanding optimization problem [11].

Min-Ling Zhang and Zhi-Jian Wang [2] proposed MIMLRBF which is derived from the traditional RBF neural networks. The first layer consists of medoids that formed by performing $k$-medoids clustering on MIML examples for each possible class, where a variant of Hausdorff metric [15] called average Hausdorff distance [16] is employed to measure the distance between bags. The $k$-medoids algorithm partition training set $U_l$ which denotes the set of MIML examples with $l$-th label into $M_l$ disjoint groups of bags $G^l_j (1 \leq j \leq M_l)$, whose medoids $C^l_j$ are determined as Eq. (1),

$$C^l_j = \arg\min_{A \in G^l_j} \sum_{B \in G^l_j} aveH(A, B)$$

where $A$ and $B$ denote two bags of instances, $aveH(A, B)$ represents the average Hausdorff distance between $A$ and $B$. The number of medoids $M_l$ for each class is set to be fraction $\alpha$ of the number of MIML examples in $U_l$, i.e., $M_l = \alpha \times |U_l|$. Second layer weights are optimized by minimizing a sum-of-squares error function and worked out through singular value decomposition (SVD). Applications to two real-world MIML tasks show that MIMLRBF is highly competitive to other MIML algorithms [2].

In multi-label classification, it is quite often that the numbers of samples in different categories are different, i.e., the class distribution is imbalanced. When the class imbalance problem occurs, the MIMLRBF will produce poor performance for setting the same fraction $\alpha$ for all classes. Experimental results in Section 4 show that this problem does exist. Let $U_m$ and $U_n$ denote two classes in a training set. When the class imbalance problem appears, we can get $|U_m| \gg |U_n|$ by assuming that the number of samples in class $U_m$ is much bigger than the number of samples in class $U_n$. After performing the $k$-medoids clustering on the two classes, we can obtain the number of medoids for each class, i.e., $M_m = \alpha \times |U_m|, M_n = \alpha \times |U_n|$. So we have $|M_m| \gg |M_n|$,
that means the number of hidden units on class $U_m$ is much bigger than the number of hidden units on class $U_n$ in the trained MIMLRBF; i.e., the number of hidden units is unbalanced. The conventional least square error training algorithm will overcompensate for the dominant class, especially when the data are not easily discernible. Therefore, the training method will theoretically lead the network to ignore the small class [17]. This problem is the reason of poor performance of the MIMLRBF in classification process. This paper presents an alternative method to solving this problem.

3. THE IMPROVED APPROACH

The MIMLRBF neural network is derived from conventional RBF neural network [2]. Like conventional RBF neural network, the MIMLRBF neural network is composed of two layers of nodes. Fig. 1 illustrates the typical architecture of an MIMLRBF neural network. As shown in the figure, the input to MIMLRBF is a bag $X$ consisting of $n$ instances $\{x_1, x_2, ..., x_n\}$, where each instance $x_k$ is a $d$-dimensional feature vector $[x_{k1}, x_{k2}, ..., x_{kd}]^T$. The first layer called hidden layer consists of $Q$ sets of bags $\bigcup_{l=1}^{Q} \{C_{l1}, C_{l2}, ..., C_{lM_l}\}$, where $M_l$ is the number of bags retained for the $l$-th class, then the total number of bags in the first layer equals $M = \Sigma_{l=1}^{Q} M_l$. Output unit compose the second layer (i.e., output layer) where each unit represents a possible class. Lines from basis functions to output nodes denote the weights $W = [w_{jl}]_{(M+1) \times Q}$ between first and second layers.

Similar as conventional RBF neural network, MIMLRBF is also trained with a two-phase learning method. The first phase consists of determining the number and medoids of hidden units and medoid widths. The three parameters should reflect the density of data points. Thus various clustering techniques can be used for this reason. Here we use a variant of $k$-medoids based on average Hausdorff distance to set the parameters. In the second stage, weights between first and second layers are calculated by minimizing a sum-of-squares error function and worked out through SVD.

![Fig. 1 Architecture of an MIMLRBF neural network](image-url)
3.1 The First Clustering Phase

Let $S = \{(X_i, Y_i)|1 \leq i \leq N\}$ corresponds to the MIML training set, $U_l = \{X_i|(X_i, Y_i) \in S, l \in Y_i\}$ denote the set of MIML examples with the $l$-th label. Regarding each bag of instances as an atomic object, a variant of $k$-medoids based on average Hausdorff distance is employed to partition $U_l$ into $M_l$ disjoint groups of bags. This algorithm has the following steps.

**Step 1:** Select $M_l$ initial medoids
1-1. Regarding each bag as a cluster.
1-2. Using average Hausdorff distance as a dissimilarity measure, compute the distance between any two clusters Eq. (2),

$$d_{ij} = \frac{\sum_{A \in G_l^i} \sum_{B \in G_l^j} \text{aveH}(AB)}{|A| \times |B|}$$  

where $|\cdot|$ gives the cardinality of a cluster and $G_l^i$ denote the $i$-th cluster in $U_l$. $\text{aveH}(\cdot, \cdot)$ is the average Hausdorff distance. It is defined as Eq. (3),

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

where $A = \{a_1, a_2, ..., a_{n_A}\}$ and $B = \{b_1, b_2, ..., b_{n_B}\}$ are two bags of instances.
1-3. Choosing a pair of clusters $G_l^i$ and $G_l^j$ which has the minimum average Hausdorff distance $d_{ij}^{\text{min}}$. If $d_{ij}^{\text{min}} < \alpha$, we merge $G_l^i$ and $G_l^j$ into one cluster. The $\alpha$ is set to be fraction $\mu$ of the average distance between any two bags of instances in $U_l$, i.e., Eq. (4).

$$\alpha = \mu \times \frac{2 \sum_{i=1}^{[U_l]} \sum_{j=1}^{[U_l]} \text{aveH}(x_i, x_j)}{n \times (n-1)}$$  

1-4. Repeat process 1-2 to 1-4 until no cluster need to be merged.
1-5. Set $M_l$ equals to the number of clusters at last. The medoid $C_l^i$ of cluster $G_l^i$ is determined as Eq. (5).

$$C_l^i = \text{argmin}_{A \in G_l^i} \sum_{B \in G_l^i} \text{aveH}(A, B)$$  

**Step 2:** Optimize $M_l$ medoids using $k$-medoids
Perfrom $k$-medoids on $U_l$, where the initial medoids is set to be $C^l = \{C_l^i|1 \leq i \leq M_l\}$. Eventually $M_l$ optimized medoids could be obtained.

After invoking the algorithm on every class, $M = \sum_{l=1}^{Q} M_l$ clusters are formed, i.e., units in the hidden layer. The basis function $\phi_j(\cdot)$ based on each cluster makes Gaussian style activation as Eq. (6) [2],

$$\phi_j(X_l) = \exp \left( - \frac{\text{aveH}(X_l, C_j)^2}{2 \sigma_j^2} \right) (1 \leq j \leq M)$$  

where $\text{aveH}(X_l, C_j)$ measures the average Hausdorff distance between $X_l$ and the
The $j$-th medoid in the hidden 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 $\sigma_j$ (i.e., width) is a parameter controlling the smoothness of basis function $\phi_j(\cdot)$. Same as in paper [2], all the $\sigma_j (1 \leq j \leq M)$ takes the same value of $\sigma$, which is some multiple of the average distance between each pair of medoids in the first layer, i.e., Eq. (7),

$$\sigma = \gamma \times \left( \frac{\sum_{p=1}^{M-1} \sum_{q=p+1}^{M} \text{ave}_{H(C_p,C_q)}}{M(M-1)/2} \right)$$

where $\gamma$ is the parameter of scaling factor.

---

**Fig. 2.** Pseudo code of IMIMLRBF

**Y = IMIMLRBF(S, µ, γ, X)**

**Inputs:**
- S: the MIML training set $\{(X_1, Y_1), \ldots, (X_N, Y_N)\}$
- $\mu$: the fraction parameter
- $\gamma$: the scaling factor
- $X$: the test MIML example ($X \subseteq X$)

**Outputs:**
- $Y$: predicted label set for $X$ ($Y \subseteq Y$)

**Process:**
1. for $l \in Y$
2. Set $U_i = \{X_i, (X_i, Y_i) \in S, i \in Y \}$;
3. Set $G^i = \{G_i^j | G_i^j = \{X_i | X_i \in U_i\} \}$;
4. Set $\text{isMerged} = \text{false}$;
5. do
6. Set $D^i = \{d_{ij} | G_i^j \in G^i, G_i^j \in G^i, d_{ij} \text{ is defined as Eq. (2)} \}$;
7. $d_{ij} = \text{min}(D^i)$;
8. if $d_{ij} < \mu$
9. $G^i = G^i - \{G_i^j \}$;
10. $G_i^j = G_i^j \cup G_i^j$;
11. $G^i = G^i + \{G_i^j \}$;
12. isMerged = true;
13. else
14. isMerged = false;
15. while isMerged = true;
16. Calculate $M_i = |G_i^1|$ initial medoids of clusters in $G^i$ using Eq.(4).
17. Optimize $M_i$ medoids using k-medoids.
18. Form matrix $\Phi$ using Eq.(6);
19. Compute weights $W$ by solving Eq.(10) with SVD;
20. $Y = \{\phi_j(X) = \sum_{j=1}^{M} w_{ij} \phi_j(X) > 0, i \in Y \}$;
3.2 Weights Optimization

To evaluate the weights between first and second layer, we used the SVD method as in [2].

Let \( \mathbf{Y}^d = [y^d_{ij}]_{Q \times N} \) denote the desired output matrix of MIMLRBF on the training set, \( \mathbf{Y} = \mathbf{W} \mathbf{\Phi} \) corresponds the estimated output matrix where \( \mathbf{\Phi} = [\phi_j(X_i)]_{(M+1) \times N} \) is the output matrix of the hidden layer. It is obvious that the error matrix is

\[
E = Y^d - Y = Y^d - W \Phi
\]

and the sum squared error, which should be minimized through the learning process, will be

\[
J = \frac{1}{2} \text{tr}(EE^T)
\]

where \( \text{tr}(\cdot) \) is trace function of a matrix.

Differentiating the error function of Eq. (9) with respect to \( \mathbf{W} \) and setting the derivative to zero matrix \( \mathbf{O} \) gives the normal equations as follows Eq. (10).

\[
\mathbf{O}J = -E \Phi^T = (W \Phi - Y^d) \Phi^T = 0 \Rightarrow W \Phi \Phi^T = Y^d \Phi^T
\]

The weights matrix \( \mathbf{W} \) can be computed by solving Eq. (10) using linear matrix inversion techniques of SVD.

Fig. 2 gives the pseudo code description of IMIMLRBF.

4. EXPERIMENTS

4.1 Experimental Setting

To benchmark the performance of the proposed learning method for class imbalance problem, we compared IMIMLRBF with MIMLRBF, MIMLBOOST and MIMLSVM on two MIML tasks. The first is the benchmark scene classification task. The chosen data set is same as used in [1]. The scene classification data contains 2000 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 [18], where each instance corresponds to an image patch.

The second MIML task is text categorization. The text data for multi-instance multi-label learning [19] is employed in this experiment, which is derived from the widely used Reuters-21578 collection [20]. In this data set, 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 2000 documents is obtained. Documents with multiple labels comprise around 15% of the data set and the average number of labels per document is \(1.15 \pm 0.37\). Each document is represented as a bag of instances based on the techniques of sliding windows [21], 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. 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 [19].

For the purpose of studying class imbalance, we reduce the size of some classes stochastically in the experiments. Then we calculate the CV (Coefficient of Variation) [22] value on the two data sets. The CV represents the ratio of the standard deviation to the mean, and it is a useful statistic for comparing the degree of variation from one data series to another, even if the means are drastically different from each other. In general the larger the CV value is, the more unbalanced the data is. Table 1 summarizes characteristics of the two data sets used in the experiment.

<table>
<thead>
<tr>
<th>Data set</th>
<th>No. of examples</th>
<th>No. of classes</th>
<th>Instances per bag</th>
<th>Labels per example ((k))</th>
<th>Min class size</th>
<th>Max class size</th>
<th>Training set size</th>
<th>Test set size</th>
<th>CV</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scene</td>
<td>1500</td>
<td>5</td>
<td>9</td>
<td>1040 400 15</td>
<td>89</td>
<td>831</td>
<td>1000</td>
<td>500</td>
<td>0.92</td>
</tr>
<tr>
<td>Reuters</td>
<td>1500</td>
<td>7</td>
<td>2</td>
<td>1200 240 9</td>
<td>100</td>
<td>580</td>
<td>1000</td>
<td>500</td>
<td>0.53</td>
</tr>
</tbody>
</table>

The performances of the four MIML algorithms are evaluated with hamming loss, one-error, coverage, ranking loss and average precision metrics [8]. For the first four metrics, the smaller the value the better performance. For average precision, the bigger the value the better the performance.

4.2 Experimental Results

All of the other three MIML algorithms (i.e., MIMLRBF, MIMLBOOST and MIMLSVM) adopt the best parameters which we evaluated each of them by conducting tenfold cross-validation on the Reuters data set described in Table 1. The fraction parameter and the scaling factor of MIMLRBF are set to 9% and 0.55 separately. The number of boosting rounds for MIMLBOOST is set to 23, and Gaussian kernel with width parameter of 0.15 is used to implement MIMLSVM.

As section 3.1 mentioned, the IMIMLRBF algorithm involves two parameters, i.e., the fraction parameter \(\mu\) and the scaling parameter \(\gamma\). Fig. 3 shows the results when the proposed algorithm performs on the text training set under different parameter configurations, where the performance is evaluated in terms of hamming loss, one-error, ranking loss and average precision. Parameter \(\mu\) increases from 40% to 130% with an
interval of 10% and $\gamma$ varies from 0.2 to 1.0 with an interval of 0.2.

From Fig. 3, we can observe that, when scaling parameter $\gamma$ is fixed, the network’s performance on the four metrics improves slightly until the fraction parameter $\mu$ increasing to 0.7 and then obviously goes down as $\mu$ increasing. On the contrary, when $\mu$ is fixed, the performance of proposed network changes drastically as $\gamma$ increasing to 0.6. However, any value bigger than 0.6 will not improve the performance evidently. Therefore, in this paper, the proposed algorithm is implemented by setting the fraction parameter $\mu$ to 0.7 and the scaling parameter $\gamma$ to 0.6.

Table 2 and Table 3 show the experimental results of each compared algorithm with the five metrics on the both data set respectively. For the first four metrics, the smaller value indicates the better performance. For average precision, the bigger value states the better performance. The best result is highlighted in boldface. We can see that the IMIMLRBF yield the best performance in terms of all metrics on the both data set.

Fig. 3 Performance of IMIMLRBF on text training data set under different parameters.
### Table 2 Experimental results on the scene data set

<table>
<thead>
<tr>
<th>Metric</th>
<th>IMIMLRBF</th>
<th>MIMLRBF</th>
<th>MIMLBOOST</th>
<th>MIMLSVM</th>
</tr>
</thead>
<tbody>
<tr>
<td>hamming loss</td>
<td>0.145±0.007</td>
<td>0.165±0.008</td>
<td>0.250±0.007</td>
<td>0.187±0.017</td>
</tr>
<tr>
<td>one-error</td>
<td>0.301±0.020</td>
<td>0.350±0.025</td>
<td>0.402±0.022</td>
<td>0.389±0.025</td>
</tr>
<tr>
<td>coverage</td>
<td>0.871±0.023</td>
<td>0.903±0.061</td>
<td>0.965±0.054</td>
<td>1.041±0.050</td>
</tr>
<tr>
<td>ranking loss</td>
<td>0.132±0.012</td>
<td>0.175±0.010</td>
<td>0.202±0.008</td>
<td>0.200±0.015</td>
</tr>
<tr>
<td>average precision</td>
<td>0.905±0.005</td>
<td>0.813±0.014</td>
<td>0.755±0.010</td>
<td>0.77±0.015</td>
</tr>
</tbody>
</table>

Furthermore, Table 4 demonstrates the training and testing time consumed by each algorithm. As showing in Table 4, the IMIMLRBF is a little slower than MIMLRBF while better than MIMLBOOST and MIMLSVM, especially than MIMLBOOST. Compared to MIMLRBF, the extra cost of IMIMLRBF is consumed by the proposed method in the first cluster stage when calculate the cluster number on each class according to data density, while MIMLRBF simply takes a fraction of the number of examples as the cluster number on each class. In summary, the experimental results reveal that the IMIMLRBF is a very effective and efficient approach to the MIML classification task when the samples are unbalanced.

### Table 3 Experimental results on the text data set

<table>
<thead>
<tr>
<th>Metric</th>
<th>IMIMLRBF</th>
<th>MIMLRBF</th>
<th>MIMLBOOST</th>
<th>MIMLSVM</th>
</tr>
</thead>
<tbody>
<tr>
<td>hamming loss</td>
<td>0.025±0.003</td>
<td>0.034±0.005</td>
<td>0.054±0.009</td>
<td>0.036±0.007</td>
</tr>
<tr>
<td>one-error</td>
<td>0.040±0.010</td>
<td>0.055±0.020</td>
<td>0.108±0.022</td>
<td>0.100±0.024</td>
</tr>
<tr>
<td>coverage</td>
<td>0.211±0.003</td>
<td>0.265±0.055</td>
<td>0.413±0.047</td>
<td>0.361±0.062</td>
</tr>
<tr>
<td>ranking loss</td>
<td>0.012±0.003</td>
<td>0.018±0.006</td>
<td>0.037±0.007</td>
<td>0.029±0.005</td>
</tr>
<tr>
<td>average precision</td>
<td>0.975±0.005</td>
<td>0.959±0.011</td>
<td>0.934±0.012</td>
<td>0.937±0.015</td>
</tr>
</tbody>
</table>

### Table 4 Time cost of each algorithm on both data sets

<table>
<thead>
<tr>
<th></th>
<th>IMIMLRBF</th>
<th>MIMLRBF</th>
<th>MIMLBOOST</th>
<th>MIMLSVM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Training</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(minutes)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Scene</td>
<td>8.11±0.05</td>
<td><strong>6.34±0.05</strong></td>
<td>7253.35±30.20</td>
<td>9.24±0.10</td>
</tr>
<tr>
<td>Reuters</td>
<td>3.81±0.11</td>
<td><strong>2.55±0.10</strong></td>
<td>4404.21±64.89</td>
<td>4.23±0.12</td>
</tr>
<tr>
<td>Testing</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(minutes)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Scene</td>
<td>0.61±0.02</td>
<td><strong>0.36±0.04</strong></td>
<td>534.63±5.51</td>
<td>1.99±0.04</td>
</tr>
<tr>
<td>Reuters</td>
<td>0.55±0.05</td>
<td><strong>0.31±0.02</strong></td>
<td>375.14±12.18</td>
<td>1.21±0.09</td>
</tr>
</tbody>
</table>

### 5. CONCLUSIONS

A Founder server with 4G RAM and Intel Xeon E5504 CPU is used to conduct the experiments.
In this paper, an improved MIMLRBF named IMIMLRBF is proposed for MIML classification task with imbalanced samples. In the first cluster stage, a variant of $k$-medoids based on the average Hausdorff distance is employed to produce medoids of the first layer. The methodology considers the data density when calculate the cluster number on each class. In this way, the numbers of hidden units for different categories could be more balanced. The network with well-adjusted units in the training process will result in better performance in networks response. In the second stage, weights between the first layer and the second layer are optimized by SVD method. Experimental results show that the IMIMLRBF is very competitive to other MIML algorithms on two data sets with imbalanced samples.

ACKNOWLEDGMENTS

This work is supported by the Fundamental Research Funds for the Central Universities (No. 09CX04031A). The authors are grateful for the anonymous reviewers who made constructive comments.
REFERENCES


