Fast Multi-Instance Multi-Label Learning

Sheng-Jun Huang, Wei Gao, and Zhi-Hua Zhou, *Fellow, IEEE*

Abstract—In many real-world tasks, particularly those involving data objects with complicated semantics such as images and texts, one object can be represented by multiple instances and simultaneously be associated with multiple labels. Such tasks can be formulated as multi-instance multi-label learning (MIML) problems, and have been extensively studied during the past few years. Existing MIML approaches have been found useful in many applications; however, most of them can only handle moderate-sized data. To efficiently handle large data sets, in this paper we propose the MIMLfast approach, which first constructs a low-dimensional subspace shared by all labels, and then trains label specific linear models to optimize approximated ranking loss via stochastic gradient descent. Although the MIML problem is complicated, MIMLfast is able to achieve excellent performance by exploiting label relations with shared space and discovering sub-concepts for complicated labels. Experiments show that the performance of MIMLfast is highly competitive to state-of-the-art techniques, whereas its time cost is much less. Moreover, our approach is able to identify the most representative instance for each label, and thus providing a chance to understand the relation between input patterns and output label semantics.

Index Terms—Multi-instance multi-label learning, fast, key instance, sub-concepts

1 Introduction

In traditional supervised learning, one object is represented by a single instance and associated with only one label. However, in many real world applications, one object can be naturally described by multiple instances, and has multiple class labels simultaneously. For example, in image classification tasks, an image may be associated with multiple semantic labels, and can be divided into several segments, each represented by an instance, as shown in Fig. 1. In text categorization tasks, an article may belong to multiple categories, and can be represented by a bag of instances, one for a paragraph [44]; in gene function prediction tasks, a gene usually has multiple labels since it is related to multiple functions, and can be represented with a set of images with different views [23]. Multi-instance multi-label learning (MIML) is a recent proposed framework for such complicated objects [55].

During the past years, many MIML algorithms were proposed [8], [46], [55]. Some of them work in a degenerating way. For example, MIMLBoost [54] degenerates the MIML task into a series of multi-instance single-label learning tasks, one for a label; and MIMLSVM [54] degenerates the MIML task into multi-label learning tasks by converting a bag into a single instance. In contrast, the DMIMLSVM [55] approach directly optimize the loss function for the original MIML task. There are also some other works try to adapt existing techniques to solve the MIML task [29], [44], [47], [49].

These approaches achieved decent performances and validated the superiority of MIML framework in various applications. However, along with the enhancing of expressive power, the hypothesis space of MIML expands dramatically, resulting in the high complexity and low efficiency of existing approaches. These approaches are usually time-consuming, and cannot handle large scale data, thus strongly limit the application of multi-instance multi-label learning.

To overcome this challenge, in this paper, we propose a novel approach MIMLfast to learn on multi-instance multi-label data fast. Though simple linear models are employed for efficiency, MIMLfast provides an effective approximation of the original MIML problem. Specifically, to utilize the relations among multiple labels, we first learn a shared space for all the labels from the original features, and then train label specific linear models from the shared space. To identify the key instance to represent a bag for a specific label, we train the classification model on the instance level, and then select the instance with maximum prediction as the key instance. To make the learning efficient, we employ stochastic gradient descent (SGD) to optimize an approximated ranking loss. At each step of SGD, MIMLfast randomly samples a triplet which consists of a bag, a relevant label of the bag and an irrelevant label, and optimizes the model to rank the relevant label before the irrelevant one if such an order is violated. Compared to state-of-the-art MIML methods, the proposed MIMLfast approach achieves highly competitive performance, and improves the efficiency for more than 100 times on large data sets.

While most existing approaches focus on improving generalization, another important task of MIML learning is to understand the relation between input patterns and output label semantics [24], i.e., to exploit the correspondence

- *S.-J. Huang is with National Key Laboratory of Novel Software Technology, Nanjing University, Nanjing 210023, China, and also with the College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing 211106, China. E-mail: huangsj@nuaa.edu.cn.*
- *W. Gao and Z.-H. Zhou are with National Key Laboratory of Novel Software Technology, Nanjing University, Nanjing 210023, China E-mail: {gaow, zhouzh}@lamda.nju.edu.cn.*

Manuscript received 19 June 2015; revised 18 June 2018; accepted 16 July 2018. Date of publication 30 July 2018; date of current version 10 Oct. 2019. (Corresponding author: Zhi-Hua Zhou).
Recommended for acceptance by G. Shakhnarovich.
For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.
Digital Object Identifier no. 10.1109/TPAMI.2018.2861732

Fig. 1. A MIML example: The image is represented with multiple instances and associated with multiple labels.

between instances and labels. Our approach can naturally identify the most representative instance for each label based on the instance-level predictions.

In addition, our approach tries to exploit sub-concepts for complicated labels. By adaptively learning multiple models for each label, the proposed method can distinguish diverse sub-concepts embedded in a complicated label.

The rest of the paper is organized as follows. Section 2 reviews some related work, Section 3 presents the MIMLfast approach, which is experimentally validated in Section 4, followed by the conclusion in Section 5.

2 Related Work

We first briefly review two learning frameworks which are related to MIML: multi-label learning and multi-instance learning.

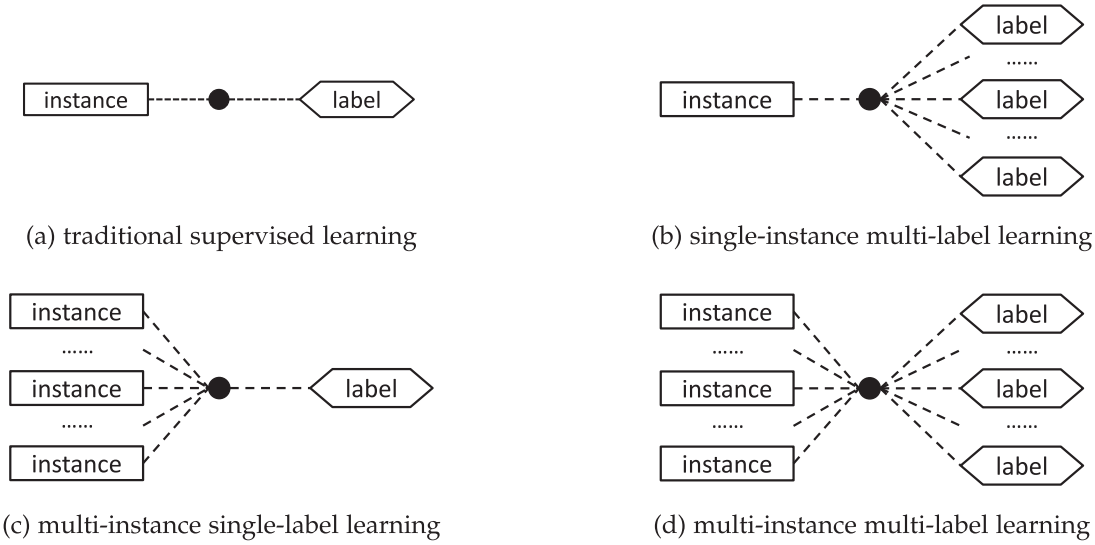
In multi-label learning, each object is represented by a single instance while associated with multiple labels. Multi-label learning has been well studied during the past decades; and many algorithms have been proposed. Existing methods can be roughly classified into two groups: problem transformation methods and algorithm adaptation methods [51]. In the first group, the simplest method is to decompose a multi-label task into a series of binary classification problems [48], one for a label, and then learn each label independently [7]. Some other methods try to transform the multi-label task into multi-class problems, where each of the multiple classes represents a possible label subset [37]. There are also some methods trying to transform the multi-label task into label ranking problems, where the objective is to rank relevant labels before irrelevant ones for every instance [14]. Algorithm adaptation methods try to modify popular learning techniques to multi-label setting. Representative approaches include boosting style algorithm AdaBoost.MH [34], lazy learning algorithm ML-kNN [50], decision tree based algorithm ML-DT [9], neural networks based algorithm [27]. Besides, weakly supervised multi-label learning has been studied, by either active querying [16] or partial annotations [42]. Recently, exploiting the relationship among multiple labels has attracted more and more interests. Efforts in this direction include utilizing prior knowledge on label structures [4], [52] and exploiting correlations among labels automatically [19], [20].

Multi-instance learning was first proposed by Dietterich et al. for drug activity prediction [10]. Under this framework, each object is described with a bag of instances and associated with a single label. The learning task is weakly supervised without instance annotations, and thus is rather challenging [53]. Many popular machine learning algorithms have been adapted to multi-instance representation. For example, the decision tree algorithm MITI [5], kernel methods Mi-Kernel [15] and Marginalized Mi-Kernel [21], lazy learning algorithms Citation-$k$nn and Bayesian-$k$nn [39], and ensemble method HSMILE [45].

Multi-instance multi-label learning is a more general framework. As sated in [55], the *multi-* learning frameworks are resulted from the ambiguities in representing complicated real-world objects. MIML considers the ambiguities in both the input and output spaces, and thus is more natural and convenient to deal with tasks involving such objects. The differences among the four learning frameworks are summarized in Fig. 2.

Many MIML approaches were proposed during the past few years. For example, MIMLSVM [54] degenerated the MIML problem into single-instance multi-label tasks to solve. MIMLBoost [54] degenerated MIML to multi-instance single-label learning. A generative model for MIML was proposed by Yang et al. [44]. Nearest neighbor and neural network approaches for MIML were proposed in [47] and [49], respectively. Zha et al. [46] proposed a hidden conditional random field model for MIML image annotation. Briggs et al. [8] proposed to optimize ranking loss for MIML instance annotation. In [24], the authors tried to discover what patters trigger what labels in MIML learning by constructing a prototype for each label with clustering. Recently, there are some studies trying to extend the MIML framework to novel settings, including class expanding [32], multi-view learning [28] and instance clustering [30]. Existing MIML approaches achieved success in many applications, most with moderate-sized data owing to the high computational load. To handle large-scale data, MIML approaches with high efficiency are demanded.

During the review period of this paper, some new MIML studies have been reported. To name a few, a discriminative probabilistic model is proposed in [31]. This method focuses on instance annotation instead of bag annotation under the MIML setting, and presents a dynamic programming solution for computing the instance label posterior probability. In [12], authors propose a deep multi-instance multi-label model, which automatically learns the instance description instead of manually designing the representation. In [43], another deep model named MIML-FCN+ is proposed for multi-instance multi-label learning. This method tries to utilize privileged bags by a two-stream fully convolutional network with privileged information loss. In [1], a scalable optimization method is proposed for multivariate performance measures. In addition, active label querying [17] and



(a) traditional supervised learning (b) single-instance multi-label learning

(c) multi-instance single-label learning (d) multi-instance multi-label learning

Fig. 2. Four different learning frameworks [55].

novel class detection [56] are also studied in multi-instance multi-label learning setting.

There are some studies trying to implement classification via ranking [25], [40]. In [40], a similar technique was used to optimize WARP loss for image annotation; however, it dealt with single-instance single-label problem, which is quite different from our MIML problem. In [55], an approach of discovering sub-concepts for complicated concepts was proposed based on clustering. However, it was focused on single label learning, quite different from our MIML task. Moreover, MIMLfast exploits label information and discovers sub-concepts using supervised model rather than heuristic clustering.

3 The MIMLfast Approach

We denote by $\{(X_1, Y_1), (X_2, Y_2), \ldots, (X_n, Y_n)\}$ a MIML dataset that consists of n examples, where each bag X_i has z_i instances $\{\mathbf{x}_{i,1}, \mathbf{x}_{i,2}, \ldots, \mathbf{x}_{i,z_i}\}$ and Y_i contains the labels associated with X_i, which is a subset of all possible labels $\{y_1, y_2 \ldots y_L\}$. In the rest of this section, we will first introduce the proposed model for MIML prediction, and then present the objective of our learning task. After that, a fast algorithm will be proposed to optimize the objective function. At last, we introduce an enhanced version of the algorithm to further utilize instance label information.

3.1 The Prediction Model

We first discuss on how to build the classification model on the instance level, and then try to get the labels of bags from instance predictions. To handle a problem with multiple labels, the simplest way is to degenerate it into a series of single label problems by training one model for each label independently. However, such a degenerating approach may lose information since it treats the labels independently and ignores the relations among them. In our approach, we formulate the model as a combination of two components. The first component learns a linear mapping from the original feature space to a low dimensional space, which is shared by all the labels. Then the second component learns label specific models based on the shared space. The two components are optimized interactively to fit training examples from all labels.

Formally, given an instance $\mathbf{x}$, we define the classification model on label l as

$$f_l(\mathbf{x}) = \mathbf{w}_l^\top W_0 \mathbf{x},$$

where W_0 is a $m \times d$ matrix which maps the original feature vectors to the shared space, and $\mathbf{w}_l$ is the m-dimensional weight vector for label l. d and m are the dimensionalities of the feature space and the shared space, respectively. The two-level model is demonstrated in Fig. 3. Such a model brings two significant advantages. First, given that we usually have $m \ll d$, the number of variables to be learned is reduced from $d \times L$ to $(d + L) \times m$. This will lead to significant reduction on both the memory and computational cost. Second, relationship among labels can be utilized. Examples from each label will contribute the optimization of the shared space, and labels with strong relations are expected to help each other. For example, assuming l is a rare label, and it would be difficult to train an accurate model for it with the very few positive examples alone; however, with our model, by fitting the examples from other labels, the shared space (W_0) may be trained to be good enough, and thus it might be much easier to optimize $\mathbf{w}_l$ based on the well-trained W_0.

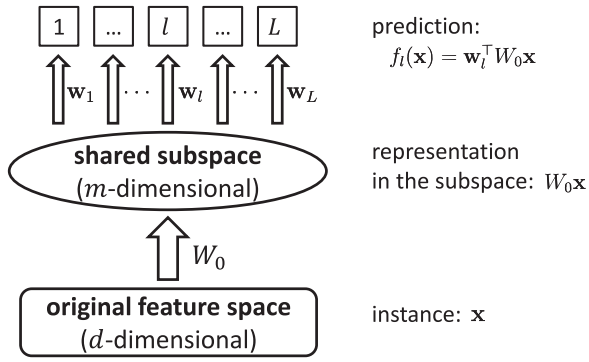


Fig. 3. The two-level model: W_0 maps the original feature vectors to a shared subspace, and $\mathbf{w}_l$ is the weight vector for label l.

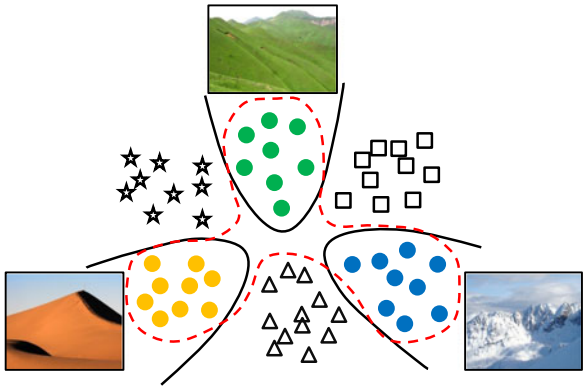


Fig. 4. An example of sub-concepts. An image with label *mountain* can be a mountain of sand, a snow mountain or a mountain covered with trees.

We then enhance the model to deal with complicated labels with sub-concepts. Objects in multi-instance multi-label learning tasks usually have complicated semantic; and thus examples with diverse contents may be assigned with the same label. Fig. 4 shows an example of complicated label with multiple sub-concepts in the image classification task. In the figure, different shapes (circles, squares, triangles and stars) represent examples belong to different labels, and different colors denote different sub-concepts. As we can see, the content of an image with the same label *mountain* can be a mountain of sand, a snow mountain or a mountain covered with trees. It is difficult to train a single model (corresponds to the dashed red line) to classify images with such diverse contents into the same category. Instead, we propose to learn multiple models for a label, one for a sub-concept, and automatically decide which sub-concept one example belongs to. The model of each sub-concept (corresponds to the black lines) is much simpler and thus may be more easily trained to fit the data. We assume that there are K sub-concepts for each label. For a given example with label l, the sub-concept it belongs to is automatically determined by first examining the prediction values of the K models, and then selecting the sub-concept with maximum prediction value. Now we can redefine the prediction of instance $\mathbf{x}$ on label l as:

$$f_l(\mathbf{x}) = \max_{k=1\cdots K} f_{l,k}(\mathbf{x}) = \max_{k=1\cdots K} \mathbf{w}_{l,k}^{\top} W_0 \mathbf{x}, \tag{1}$$

where $\mathbf{w}_{l,k}$ corresponds to the kth sub-concept of label l. Note that although we assume there are K sub-concepts for each label, empty sub-concepts are allowed, i.e., examples of a simple label may be distributed in only a few or even one sub-concept.

We then look at how to get the predictions of bags from the instance level models. It is usually assumed that a bag is positive if and only if it contains at least one positive instance [8], [10]. Under this assumption, the prediction of a bag X on label l can be defined as the maximum of predictions of all instances in this bag:

$$f_l(X) = \max_{\mathbf{x}\in X} f_l(\mathbf{x}).$$

We call the instance with maximum prediction the key instance of X on label l.

3.2 The Objective

We transform the MIML classification problem into a label ranking problem. And the objective is to rank relevant labels before irrelevant labels for all examples. With the model introduced in the previous subsection, for an example X and one of its relevant labels l, we define $R(X,l)$ as

$$R(X,l) = \sum_{j\in\bar{Y}} I[f_j(X) > f_l(X)], \tag{2}$$

where $\bar{Y}$ denotes the set of irrelevant labels of X, and $I[\cdot]$ is the indicator function which returns 1 if the argument is true and 0 otherwise. Essentially, $R(X,l)$ counts how many irrelevant labels are ranked before label l on the bag X.

Based on $R(X,l)$, we further define the ranking error [38] with respect to an example X on label l as

$$\epsilon(X,l) = \sum_{i=1}^{R(X,l)} \frac{1}{i}. \tag{3}$$

It is obvious that the ranking error ϵ would be larger for lower l being ranked. Finally, we have the ranking error on the whole dataset as

$$\text{Rank Error} = \sum_{i=1}^{n} \sum_{l\in Y_i} \epsilon(X,l).$$

Based on Eq. (2), the ranking error $\epsilon(X,l)$ can be spread into all irrelevant labels in $\bar{Y}$ as:

$$\epsilon(X,l) = \sum_{j\in\bar{Y}} \epsilon(X,l) \frac{I[f_j(X) > f_l(X)]}{R(X,l)}. \tag{4}$$

Due to non-convexity and discontinuousness, it is rather difficult to optimize the above equation directly because such optimization often leads to NP-hard problems. We instead explore the following hinge loss, which has been shown as an optimal choice among all convex surrogate losses [3],

$$\Psi(X,l) = \sum_{j\in\bar{Y}} \epsilon(X,l) \frac{|1 + f_j(X) - f_l(X)|_+}{R(X,l)}, \tag{5}$$

where $|q|_+ = q$ if $q \geq 0$; otherwise, $|q|_+ = 0$. The surrogate loss $\Psi(X,l)$ can be viewed as an upper bound of $\epsilon(X,l)$ because $I[q] \leq |1+q|_+$.

To rank the relevant labels before irrelevant labels, we only need to minimize the rank loss on the training data, leading to the following objective function:

$$\min \sum_{i=1}^{n} \sum_{l\in Y_i} \Psi(X_i,l). \tag{6}$$

3.3 The Algorithm

We then employ stochastic gradient descent [33] to minimize the ranking error. At each iteration of SGD, we randomly sample a bag X, one of its relevant labels y, and one of its irrelevant labels $\bar{y} \in \bar{Y}$ to form a triplet $(X, y, \bar{y})$, which will induce a loss:

$$\mathcal{L}(X,y,\bar{y}) = \epsilon(X,y)|1 + f_{\bar{y}}(X) - f_y(X)|_+. \tag{7}$$

We then have the following lemma to disclose the relation between $\Psi(X,y)$ and $\mathcal{L}(X,y,\bar{y})$.

Lemma 1. *$\Psi(X,y) = E_{\bar{y}}[\mathcal{L}(X,y,\bar{y})]$, where $E[\cdot]$ denotes the expectation on the uniform distribution over $\bar{Y}$.*

Proof. This lemma follows from the fact that probability of randomly choosing $\bar{y}$ in $\bar{Y}$ is $1/R(X,y)$. □

To minimize $\mathcal{L}(X,y,\bar{y})$, it is required to calculate $R(X,y)$ in advance, i.e., we have to compare $f_y(X)$ with $f_{\bar{y}}(X)$ for each $\bar{y} \in \bar{Y}$, whereas this could be time consuming when the number of possible labels is large. Therefore, we use an approximation to estimate $R(X,y)$ in our implementation, inspired by Weston et al. [40]. Specifically, at each SGD iteration, we randomly sample labels from the irrelevant label set $\bar{Y}$ one by one, until a violated label $\bar{y}$ occurs. Here we call $\bar{y}$ a violated label if it was ranked before y, i.e., $f_{\bar{y}}(X) > f_y(X) - 1$. Without loss of generality, we assume that the first violated label is found at the vth sampling step, and then, $R(X,y)$ can be approximated by $\lfloor |\bar{Y}|/v \rfloor$ with the following lemma:

Lemma 2. *We denote by θ a random event with $\theta = i$ representing the event that first violated label is at the ith sampling step. We have*

$$\frac{R(X,y)}{|\bar{Y}|} \approx E_\theta\left[\frac{1}{\theta}\right].$$

Proof. For convenience, we set $p = R(X,y)/|\bar{Y}|$ and assume $0 < p < 1$ without loss of generality. It is easy to derive the probability

$$\Pr[\theta = i] = (1-p)^{i-1}p \text{ for } i \geq 1,$$

and we further have

$$\begin{aligned} E_\theta\left[\frac{1}{\theta}\right] &= \sum_{i=1}^{\infty}\frac{1}{i}p(1-p)^{i-1} = \frac{p}{1-p}\sum_{i=1}^{\infty}\frac{1}{i}(1-p)^i \\ &= \frac{-p}{1-p}\ln(1-(1-p)) \approx p, \end{aligned}$$

where we use $\sum_{i=1}^{\infty}\frac{1}{i}(1-p)^i = -\ln(p)$ and $\ln(1+q) \approx q$. This completes the proof. □

We assume that the triplet sampled at the tth SGD iteration is $(X,y,\bar{y})$, on label y, the key instance is $\mathbf{x}$, and achieves the maximum prediction on the kth sub-concept, while on label $\bar{y}$, the instance $\bar{\mathbf{x}}$ achieves the maximum prediction on the $\bar{k}$th sub-concept. Then we have the approximated ranking loss for the triplet:

$$\begin{aligned} &\mathcal{L}(X,y,\bar{y}) = \epsilon(X,y)|1 + f_{\bar{y}}(X) - f_y(X)|_+ \\ &\approx \begin{cases} 0 & \text{if } \bar{y} \text{ is not violated;} \\ S_{\bar{Y},v}(1 + [\mathbf{w}^t_{\bar{y},\bar{k}}]^\top W_0^t\bar{\mathbf{x}} - [\mathbf{w}^t_{y,k}]^\top W_0^t\mathbf{x}) & \text{otherwise.} \end{cases} \end{aligned}$$

Here we introduce $S_{\bar{Y},v} = \sum_{i=1}^{\lfloor\frac{|\bar{Y}|}{v}\rfloor}\frac{1}{i}$ for the convenience of presentation. So, if a violated label $\bar{y}$ is sampled, we perform the gradient descent on the three parameters according to:

$$W_0^{t+1} = W_0^t - \gamma_t S_{\bar{Y},v}(\mathbf{w}^t_{\bar{y},\bar{k}}\bar{\mathbf{x}}^\top - \mathbf{w}^t_{y,k}\mathbf{x}^\top) \tag{8}$$

$$\mathbf{w}^{t+1}_{y,k} = \mathbf{w}^t_{y,k} + \gamma_t S_{\bar{Y},v} W_0^t\mathbf{x} \tag{9}$$

$$\mathbf{w}^{t+1}_{\bar{y},\bar{k}} = \mathbf{w}^t_{\bar{y},\bar{k}} - \gamma_t S_{\bar{Y},v} W_0^t\bar{\mathbf{x}}, \tag{10}$$

where γ_t is the step size of SGD at the tth iteration. After the update of the parameters, $\mathbf{w}_{y,k}$, $\mathbf{w}_{\bar{y},\bar{k}}$ and each column of W_0 are normalized to have a L2 norm smaller than a constant C.

The pseudo code of MIMLfast is presented in Algorithm 1. First, each column of W_0 and $\mathbf{w}^k_y$ for all labels y and all sub-concepts k are initialized at random with mean 0 and standard deviation $1/\sqrt{d}$. Then at each iteration of SGD, a triplet $(X,y,\bar{y})$ is randomly sampled, and their corresponding key instance and sub-concepts are identified. After that, gradient descent is performed to update the three parameters: W_0, $\mathbf{w}_{y,k}$ and $\mathbf{w}_{\bar{y},\bar{k}}$ according to Eqs. 8 to 10. At last, the updated parameters are normalized such that their norms will be upper bounded by C. This procedure is repeated until some stop criteria reached. In our experiments, we sample a small subset from the training data to form a validation set, and stop the training if the ranking loss does not decrease anymore on the validation set.

Algorithm 1. The MIMLfast Algorithm

1: *INPUT:*
2: training data, parameters m, C, K and γ_t
3: *TRAIN:*
4: initialize W_0 and $\mathbf{w}_{l,k}$ $(l = 1\cdots L, k = 1\cdots K)$
5: **repeat**:
6: randomly sample a bag X and one of its label y
7: $(\mathbf{x},k) = \arg\max_{\mathbf{x}\in X, k\in\{1\cdots K\}} f_{y,k}(\mathbf{x})$
8: **for** $i = 1 : |\bar{Y}|$
9: sample an irrelevant label $\bar{y}$ from $\bar{Y}$
10: $(\bar{\mathbf{x}},\bar{k}) = \arg\max_{\mathbf{x}\in X,\bar{k}\in\{1\cdots K\}} f_{\bar{y},\bar{k}}(\mathbf{x})$
11: **if** $f_{\bar{y}}(X) > f_y(X) - 1$
12: $v = i$
13: update W_0, $\mathbf{w}_{y,k}$ and $\mathbf{w}_{\bar{y},\bar{k}}$ as Eqs. 8 to 10
14: normalize W_0, $\mathbf{w}_{y,k}$ and $\mathbf{w}_{\bar{y},\bar{k}}$
15: break
16: **end if**
17: **end for**
18: **until** stop criterion reached
19: *TEST:*
20: Relevant labels set for the test bag X_{test} is: $\{l|1 + f_l(X_{\text{test}}) > f_{\hat{y}}(X_{\text{test}})\}$

In the test phase of the algorithm, for a bag X_{test}, we can get the prediction value on each label, and consequently the rank of all labels. For single label classification problem, it is very easy to get the label of X_{test} by selecting the one with largest prediction value. However, in multi-label learning, the bag X_{test} may have more than one relevant label; and thus one do not know how many labels should be selected as relevant ones from the ranked label list [14]. To solve this problem, we assign each bag a dummy label, denoted by $\hat{y}$, and train the model to rank the dummy label before all irrelevant labels while after the relevant ones. To implement this idea, we pay a special consideration on constructing the irrelevant labels set $\bar{Y}$. Specifically, when X and its label y are sampled (in Line 6 of Algorithm 1), the algorithm will first examine whether y is the dummy label. If $y = \hat{y}$, then $\bar{Y}$ consists of all the irrelevant labels; otherwise, $\bar{Y}$ contains both the dummy label and all the irrelevant labels. In such a way, the model will be trained to rank the dummy label between relevant labels and irrelevant ones. For a test bag, the labels with larger prediction value than that on the dummy label are selected as relevant labels.

At last, we present some theoretical guarantees on the convergence rate of the proposed algorithm. Denoting by

$$\mathcal{L}_t(W_0, \mathbf{w}_{y,k}, \mathbf{w}_{\bar{y},\bar{k}}) = S_{\bar{Y},v}(1 + \mathbf{w}_{\bar{y},\bar{k}}^\top W_0 \bar{\mathbf{x}}_t - \mathbf{w}_{y,k}^\top W_0 \mathbf{x}_t)_+$$

the loss of tth SGD iteration with model parameters W_0, $\mathbf{w}_{y,k}$ $\mathbf{w}_{\bar{y},\bar{k}}$, and

$$(W_0^*, \mathbf{w}_{l,k}^*) \in \arg\min \sum_t \mathcal{L}_t(W_0, \mathbf{w}_{y,k}, \mathbf{w}_{\bar{y},\bar{k}})$$

the optimal solution, we have the following theorem.

Theorem 1. *Suppose* $\|\mathbf{x}_t\| \le 1$, $\|W_0^t\| \le C\sqrt{d}$, $\|\mathbf{w}_{y,k}^t\| \le C$ *and* $\|\mathbf{w}_{\bar{y},\bar{k}}^t\| \le C$. *By choosing proper* W_0^0, $\mathbf{w}_{y,k}^0$ *and* γ_t, *it holds that*

$$\sum_t^T \mathcal{L}_t(W_0^t, \mathbf{w}_{y,k}^t, \mathbf{w}_{\bar{y},\bar{k}}^t) - \sum_t^T \mathcal{L}_t(W_0^*, \mathbf{w}_{y,k}^*, \mathbf{w}_{\bar{y},\bar{k}}^*) \le B\sqrt{T}$$

where $B = 4 + (d + 2\sqrt{d})C^2 \sum_{i=1}^L \frac{1}{i}$.

Proof. Because the function $\mathcal{L}_t(W_0^t, \mathbf{w}_{y,k}^t, \mathbf{w}_{\bar{y},\bar{k}}^t)$ is convex with respect to $\mathbf{w}_{y,k}^t$ and $\mathbf{w}_{\bar{y},\bar{k}}^t$, we have

$$\begin{aligned}
&\mathcal{L}_t(W_0^t, \mathbf{w}_{y,k}^t, \mathbf{w}_{\bar{y},\bar{k}}^t) - \mathcal{L}_t(W_0^t, \mathbf{w}_{y,k}^*, \mathbf{w}_{\bar{y},\bar{k}}^*) \\
&\quad \le [\partial \mathcal{L}_t(W_0^t, \mathbf{w}_{y,k}^t, \mathbf{w}_{\bar{y},\bar{k}}^t)/\partial \mathbf{w}_{y,k}^t]^\top (\mathbf{w}_{y,k}^t - \mathbf{w}_{y,k}^*) \\
&\quad + [\partial \mathcal{L}_t(W_0^t, \mathbf{w}_{y,k}^t, \mathbf{w}_{\bar{y},\bar{k}}^t)/\partial \mathbf{w}_{\bar{y},\bar{k}}^t]^\top (\mathbf{w}_{\bar{y},\bar{k}}^t - \mathbf{w}_{\bar{y},\bar{k}}^*).
\end{aligned}$$

From Eqs. (9) and (10), we have

$$\begin{aligned}
\|\mathbf{w}_{y,k}^{t+1} - \mathbf{w}_{y,k}^*\|^2 &= \|\mathbf{w}_{y,k}^t - \mathbf{w}_{y,k}^*\|^2 + \Delta_t \\
&\quad - 2\gamma_t [\partial \mathcal{L}_t(W_0^t, \mathbf{w}_{y,k}^t, \mathbf{w}_{\bar{y},\bar{k}}^t)/\partial \mathbf{w}_{y,k}^t]^\top (\mathbf{w}_{y,k}^t - \mathbf{w}_{y,k}^*),
\end{aligned}$$

where

$$\Delta_t = \gamma_t^2 \left\| S_{\bar{Y},v} W_0^t \right\|^2 \le dC^2 \gamma_t^2 \sum_{i=1}^L \frac{1}{i}.$$

This follows that

$$\begin{aligned}
\|\mathbf{w}_{y,k}^{t+1} - \mathbf{w}_{y,k}^*\|^2 &\le \|\mathbf{w}_{y,k}^t - \mathbf{w}_{y,k}^*\|^2 + dC^2 \gamma_t^2 \sum_{i=1}^L 1/i \\
&\quad - 2\gamma_t [\partial \mathcal{L}_t(W_0^t, \mathbf{w}_{y,k}^t, \mathbf{w}_{\bar{y},\bar{k}}^t)/\partial \mathbf{w}_{y,k}^t]^\top (\mathbf{w}_{y,k}^t - \mathbf{w}_{y,k}^*).
\end{aligned}$$

In a similar manner, we have

$$\begin{aligned}
\|\mathbf{w}_{\bar{y},\bar{k}}^{t+1} - \mathbf{w}_{\bar{y},\bar{k}}^*\|^2 &\le \|\mathbf{w}_{\bar{y},\bar{k}}^t - \mathbf{w}_{\bar{y},\bar{k}}^*\|^2 + dC^2 \gamma_t^2 \sum_{i=1}^L 1/i \\
&\quad - 2\gamma_t [\partial \mathcal{L}_t(W_0^t, \mathbf{w}_{y,k}^t, \mathbf{w}_{\bar{y},\bar{k}}^t)/\partial \mathbf{w}_{\bar{y},\bar{k}}^t]^\top (\mathbf{w}_{\bar{y},\bar{k}}^t - \mathbf{w}_{\bar{y},\bar{k}}^*).
\end{aligned}$$

Summing over $t = 0, \ldots, T-1$, and by setting $\gamma_t = 1/\sqrt{t}$ and simple calculation, we have

$$\begin{aligned}
&\sum_{t=1}^{T-1} \mathcal{L}_t(W_0^t, \mathbf{w}_{y,k}^t, \mathbf{w}_{\bar{y},\bar{k}}^t) - \sum_{t=1}^{T-1} \mathcal{L}_t(W_0^t, \mathbf{w}_{y,k}^*, \mathbf{w}_{\bar{y},\bar{k}}^*) \\
&\quad \le \frac{2}{\gamma_T} + B \sum_{t=1}^{T-1} \frac{\gamma_t}{2} \le (2 + dC^2 \sum_{i=1}^L \frac{1}{i})\sqrt{T}.
\end{aligned}$$

Further, we have

$$\begin{aligned}
&\sum_{t=1}^{T-1} \mathcal{L}_t(W_0^t, \mathbf{w}_{y,k}^*, \mathbf{w}_{\bar{y},\bar{k}}^*) - \sum_{t=1}^{T-1} \mathcal{L}_t(W_0^*, \mathbf{w}_{y,k}^*, \mathbf{w}_{\bar{y},\bar{k}}^*) \\
&= \sum_{t=1}^{\sqrt{T}-1} \mathcal{L}_t(W_0^t, \mathbf{w}_{y,k}^*, \mathbf{w}_{\bar{y},\bar{k}}^*) - \sum_{t=1}^{\sqrt{T}-1} \mathcal{L}_t(W_0^*, \mathbf{w}_{y,k}^*, \mathbf{w}_{\bar{y},\bar{k}}^*) \\
&\quad + \sum_{t=\sqrt{T}}^{T-1} \mathcal{L}_t(W_0^t, \mathbf{w}_{y,k}^*, \mathbf{w}_{\bar{y},\bar{k}}^*) - \sum_{t=\sqrt{T}}^{T-1} \mathcal{L}_t(W_0^*, \mathbf{w}_{y,k}^*, \mathbf{w}_{\bar{y},\bar{k}}^*) \\
&\quad \le 2\left(1 + \sqrt{d}C^2 \sum_{i=1}^L 1/i\right)\sqrt{T}.
\end{aligned}$$

by selecting proper initial values and simple calculation. This theorem follows as desired. □

3.4 Learning with Instance Labels

The above introduced algorithm can identify the key instance of a bag for each label. In some MIML tasks, in addition to bag labels, the label assignments for instances are also available, i.e., the key instance of each label is known. For example, in image annotation tasks, annotators may be asked to not only assign tags to the image but also identify the corresponding region for each tag. In such cases, we can adapt our algorithm to further utilize such supervision information. The basic idea is to rank the key instance before other instances for each relevant label. Formally, given a bag X and one of its relevant label y, the ranking error of its key instance $\mathbf{x}^*$ can be defined as:

$$\xi(y, \mathbf{x}^*) = \sum_{i=1}^{R(y,\mathbf{x}^*)} \frac{1}{i}, \tag{11}$$

where

$$R(y, \mathbf{x}^*) = \sum_{\mathbf{x} \in X} I[f_y(\mathbf{x}) > f_y(\mathbf{x}^*)] \tag{12}$$

counts how many instances are ranked before the key instance on label y. Obviously, the error reaches zero when the key instance $\mathbf{x}^*$ optimally achieves the largest prediction value on y. Similar to the case of label ranking, if an instance $\tilde{\mathbf{x}}$ is ranked before the key instance $\mathbf{x}^*$, then it will induce a surrogate ranking error as follows:

$$\mathcal{J}(y, \mathbf{x}^*, \tilde{\mathbf{x}}) = \xi(y, \mathbf{x}^*)|1 + f_y(\tilde{\mathbf{x}}) - f_y(\mathbf{x}^*)|_+. \tag{13}$$

With similar deviations in Section 3.3, we can have the following gradient descent rules for updating the model:

$$W_0^{t+1} = W_0^t - \gamma_t \xi(y, \mathbf{x}^*) \mathbf{w}_{y,k}^t (\tilde{\mathbf{x}}^\top - \mathbf{x}^{*\top}) \tag{14}$$

$$\mathbf{w}_{y,k}^{t+1} = \mathbf{w}_{y,k}^t - \gamma_t \xi(y, \mathbf{x}^*) W_0^t (\tilde{\mathbf{x}} - \mathbf{x}^*). \tag{15}$$

Note that because the number of instances in a bag is usually not large, $\xi(y, \mathbf{x}^*)$ can be efficiently calculated according to Eq. (11). One can also get an approximation value with Lemma 2 in case of large bag size. In summary, when the instance labels are available, we can combine Eqs. (14), (15) with Eqs. (8), (9), (10) to jointly optimize the rank of labels and instances.

TABLE 1
Statistics of Experimental Data Sets
(6 with Moderate Size and 2 with Large Size)

Data	# ins.	# bag	# label	# label per bag
Letter Frost	565	144	26	3.6
Letter Carroll	717	166	26	3.9
MSRC v2	1758	591	23	2.5
Reuters	7119	2000	7	1.2
Bird Song	10232	548	13	2.1
Scene	18000	2000	5	1.2
Corel5K	47,065	5,000	260	3.4
MSRA	270,000	30,000	99	2.7

4 Experiments

4.1 Settings

We compare MIMLfast with six state-of-the-art MIML methods: DBA [44], a generative model for MIML learning; KISAR [24], a MIML algorithm tries to discover instance-label relation; MIMLBoost [54], a boosting method decomposes MIML into multi-instance single label problems; MIMLkNN [47], a MIML nearest neighbor algorithm; MIMLSVM [54], a SVM style algorithm which decomposes MIML into single instance multi-label problems; and RankLossSIM [8], a MIML algorithm minimizes ranking loss for instance annotation.

We perform the experiments on 6 moderate-sized data sets and 2 large data sets. Among the moderate-sized data sets, *Scene* and *Reuters* are two benchmark data sets commonly used in existing MIML works. *Scene* [54] consists of 2000 images for scene classification, and is associated with 5 possible labels: desert, mountains, sea, sunset and trees. For each image, a bag of 9 instances is extracted via SBN [26]. *Reuters* is constructed based on the Reuters-21578 data set [36] with the sliding window technique in [2]. The other four moderate-sized data sets are collected by Fern et al. in their recent work [8]: *Letter Carroll* and *Letter Frost* are constructed

TABLE 2
Comparison Results (mean ± std.) on Moderate-Sized Data Sets

Methods	MIMLfast	DBA	KISAR	MIMLBoost	MIMLkNN	MIMLSVM	RankLossSIM
Letter Carroll							
hamming loss ↓	.134 ± .012	.180 ± .010•	.150 ± .008•	.153 ± .008•	.170 ± .017•	.154 ± .007•	.132 ± .006
one error ↓	.119 ± .050	.248 ± .036•	.058 ± .096○	.645 ± .062•	.312 ± .043•	.554 ± .043•	.167 ± .050•
coverage ↓	.380 ± .029	.909 ± .023•	.870 ± .018•	.730 ± .039•	.460 ± .030•	.905 ± .020•	.389 ± .037
ranking loss ↓	.130 ± .013	.622 ± .033•	.873 ± .043•	.477 ± .035•	.194 ± .019•	.710 ± .029•	.134 ± .017
average precision ↑	.715 ± .032	.324 ± .029•	.181 ± .027•	.263 ± .020•	.611 ± .023•	.350 ± .022•	.708 ± .026
Letter Frost							
hamming loss ↓	.136 ± .014	.166 ± .010•	.200 ± .013•	.139 ± .007	.139 ± .010	.154 ± .013•	.136 ± .010
one error ↓	.151 ± .041	.228 ± .056•	.380 ± .064•	.257 ± .101•	.288 ± .077•	.581 ± .045•	.203 ± .055•
coverage ↓	.375 ± .042	.857 ± .032•	.906 ± .019•	.728 ± .038•	.463 ± .035•	.884 ± .028•	.372 ± .038
ranking loss ↓	.134 ± .019	.580 ± .033•	.705 ± .036•	.478 ± .030•	.199 ± .018•	.810 ± .101•	.138 ± .019
average precision ↑	.704 ± .034	.358 ± .030•	.264 ± .028•	.235 ± .014•	.612 ± .027•	.226 ± .060•	.686 ± .035•
MSRC v2							
hamming loss ↓	.100 ± .007	.140 ± .006•	.086 ± .004○	N/A	.131 ± .007•	.084 ± .003○	.110 ± .004•
one error ↓	.295 ± .025	.415 ± .026•	.341 ± .031•	N/A	.440 ± .031•	.320 ± .029•	.302 ± .028
coverage ↓	.238 ± .014	.837 ± .018•	.254 ± .015•	N/A	.312 ± .020•	.256 ± .018•	.239 ± .013
ranking loss ↓	.108 ± .009	.675 ± .017•	.131 ± .010•	N/A	.165 ± .013•	.125 ± .011•	.107 ± .007
average precision ↑	.688 ± .017	.326 ± .016•	.666 ± .018•	N/A	.591 ± .018•	.685 ± .018	.687 ± .013
Reuters							
hamming loss ↓	.028 ± .004	.043 ± .004•	.032 ± .003•	N/A	.034 ± .004•	.042 ± .004•	.037 ± .003•
one error ↓	.044 ± .008	.077 ± .011•	.057 ± .010•	N/A	.065 ± .011•	.100 ± .015•	.055 ± .007•
coverage ↓	.035 ± .004	.089 ± .010•	.036 ± .004•	N/A	.043 ± .004•	.050 ± .006•	.036 ± .004•
ranking loss ↓	.014 ± .004	.062 ± .008•	.016 ± .003•	N/A	.023 ± .004•	.031 ± .005•	.016 ± .003•
average precision ↑	.972 ± .005	.922 ± .008•	.966 ± .006•	N/A	.958 ± .006•	.939 ± .009•	.967 ± .005•
Bird Song							
hamming loss ↓	.073 ± .009	.116 ± .005•	.098 ± .011•	N/A	.081 ± .007•	.073 ± .005	.087 ± .008•
one error ↓	.055 ± .017	.101 ± .020•	.159 ± .039•	N/A	.122 ± .029•	.111 ± .025•	.064 ± .046
coverage ↓	.150 ± .013	.292 ± .015•	.186 ± .018•	N/A	.175 ± .015•	.173 ± .013•	.133 ± .011○
ranking loss ↓	.036 ± .007	.132 ± .010•	.067 ± .012•	N/A	.059 ± .010•	.054 ± .006•	.027 ± .008○
average precision ↑	.921 ± .014	.786 ± .013•	.847 ± .026•	N/A	.878 ± .017•	.888 ± .011•	.930 ± .025
Scene							
hamming loss ↓	.188 ± .009	.269 ± .009•	.194 ± .005•	N/A	.196 ± .007•	.200 ± .008•	.204 ± .007•
one error ↓	.351 ± .023	.386 ± .025•	.351 ± .020	N/A	.370 ± .018•	.380 ± .021•	.392 ± .019•
coverage ↓	.207 ± .012	.334 ± .011•	.204 ± .008○	N/A	.222 ± .009•	.225 ± .010•	.237 ± .010•
ranking loss ↓	.189 ± .014	.348 ± .012•	.185 ± .010	N/A	.207 ± .011•	.212 ± .011•	.222 ± .010•
average precision ↑	.770 ± .015	.600 ± .013•	.772 ± .012	N/A	.757 ± .011•	.750 ± .012•	.738 ± .011•

↑(↓) indicates that the larger (smaller) the value, the better the performance; •(○) indicates that MIMLfast is significantly better(worse) than the corresponding method based on paired t-tests at 95% significance level; N/A indicates that no result was obtained in 24 hours.

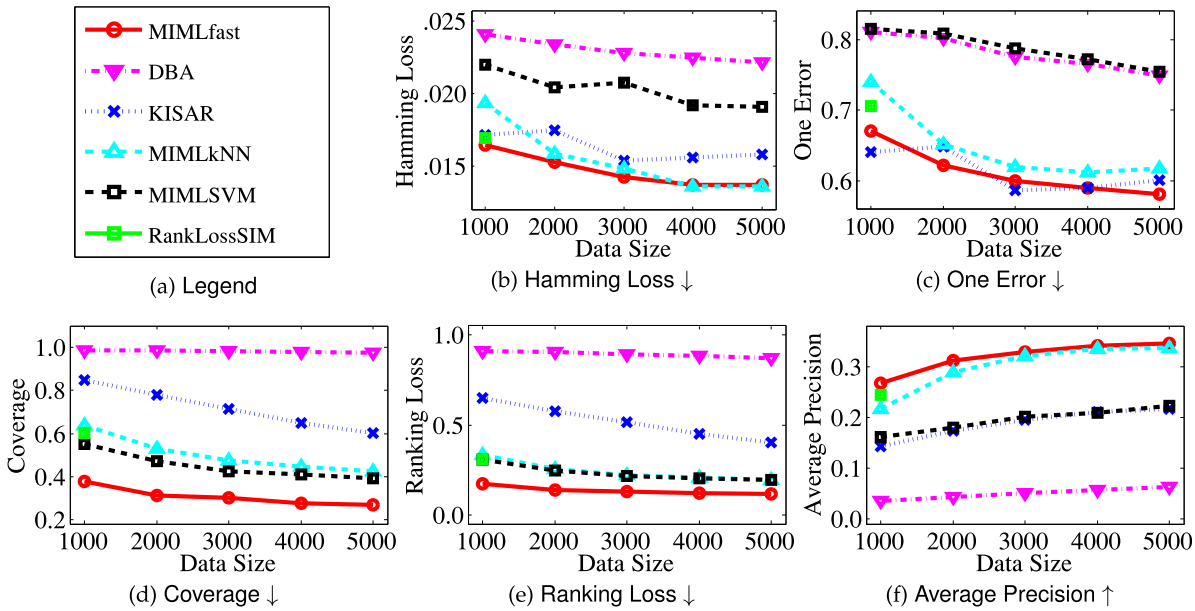


Fig. 5. Comparison results on *Corel5K* with varying data size; ↑(↓) indicates that the larger (smaller) the value, the better the performance.

using the UCI Letter Recognition dataset [13], where a bag is created for each word, and labels correspond to the letters. *Bird Song* consists of bird song recordings at the H. J. Andrews (HJA) Experimental Forest. Each bag is extracted from a 10-seconds audio recording while labels correspond to species of birds. *MSRC v2* is a subset of the Microsoft Research Cambridge (MSRC) image dataset [41]. Based on the ground-truth segmentation, histograms of gradients and colors are extracted to form an instance for each segment. The two large data sets are *Corel5K* and *MSRA*. *Corel5K* [11] contains 5000 segmented images and 260 class labels, and each image is represented by 9 instances on average. *MSRA* [22] is a multimedia database collected by Microsoft Research Asia, the subset used in this work contains 30000 images with 99 possible labels, and each image is represented with a bag of 9 instances. The detailed characteristics of these data sets are summarized in Table 1.

For *MSRA* and *Corel5K*, since existing MIML approaches cannot handle large scale data, we examine the performances of compared approaches on a series of subsets with different number of training bags (which will be specified later). For each data set, 2/3 of the data are randomly sampled for training, and the remaining examples are taken as test set. We repeat the random data partition for thirty times, and report the average results over the thirty repetitions.

For MIMLfast, the step size is in the form $\gamma_t = \gamma_0/(1+\eta\gamma_0 t)$ according to [6]. The parameters are selected by 3-fold cross validation on the training data with regard to ranking loss. The candidate values for the parameters are as below: $m \in \{50, 100, 200\}$, $C \in \{1, 5, 10\}$, $K \in \{1, 5, 10, 15\}$, $\gamma_0 \in \{0.0001, 0.0005, 0.001, 0.005\}$ and $\eta \in \{10^{-5}, 10^{-6}\}$. In our experience, the algorithm is not very sensitive to m and C; and the influence of K will be studied in Section 3.5. For the compared approaches, parameters are determined in the same way if no value suggested in their literatures.

Performance evaluation in MIML tasks is more complicated than that in traditional supervised learning. And traditional criteria such as accuracy, precision, recall, etc. cannot be directly used. In our experiments, we evaluated the performance with five commonly used MIML criteria: hamming loss, one error, coverage, ranking loss and average precision. For average precision, a larger value implies a better performance, while for the other four criteria, the smaller, the better. Coverage is normalized by the number of labels such that all criteria are in the interval $[0, 1]$. The definition of these criteria can be found in [35], [55].

4.2 Performance Comparison

We first report the comparison results on the six moderate-sized data sets in Table 2. As shown in the table, our approach MIMLfast achieves the best performance in most cases. DBA tends to favor text data, and is outperformed by MIMLfast on all the data sets. KISAR achieves comparable results with MIMLfast on *Scene* while is less effective on the other data sets. MIMLBoost can handle only the two smallest data sets, and does not yield good performance. MIMLkNN and MIMLSVM work steady on all the data sets, but are not competitive when compared with MIMLfast. At last, RankLossSIM is comparable to MIMLfast on 3 of 6 data sets, and even achieves better coverage and ranking loss on the *Bird Song* data set. However, on the other two data sets with relative more bags, i.e., *Reuters* and *Scene*, it is significantly worse than our approach on all the five criteria.

MSRA and *Corel5K* contain 30000 and 5000 bags respectively, which are too large for most existing MIML approaches. We thus perform the comparison on subsets of them with different data sizes. We vary the number of bags from 1000 to 5000 for *Corel5K*, and 5000 to 30000 for *MSRA*, and plot the performance curves in Figs. 5 and 6, respectively. MIMLBoost did not return results in 24 hours even for the smallest data size, and thus it is not included in the comparison. RankLossSIM is not presented on *MSRA* for the same reason. We also exclude DBA on *MSRA* because its performance is too bad. As observable in Figs. 5 and 6, MIMLfast is apparently better than the others on these two large data sets. Particularly, when data size reaches 25K, other methods cannot work, while MIMLfast still works well.

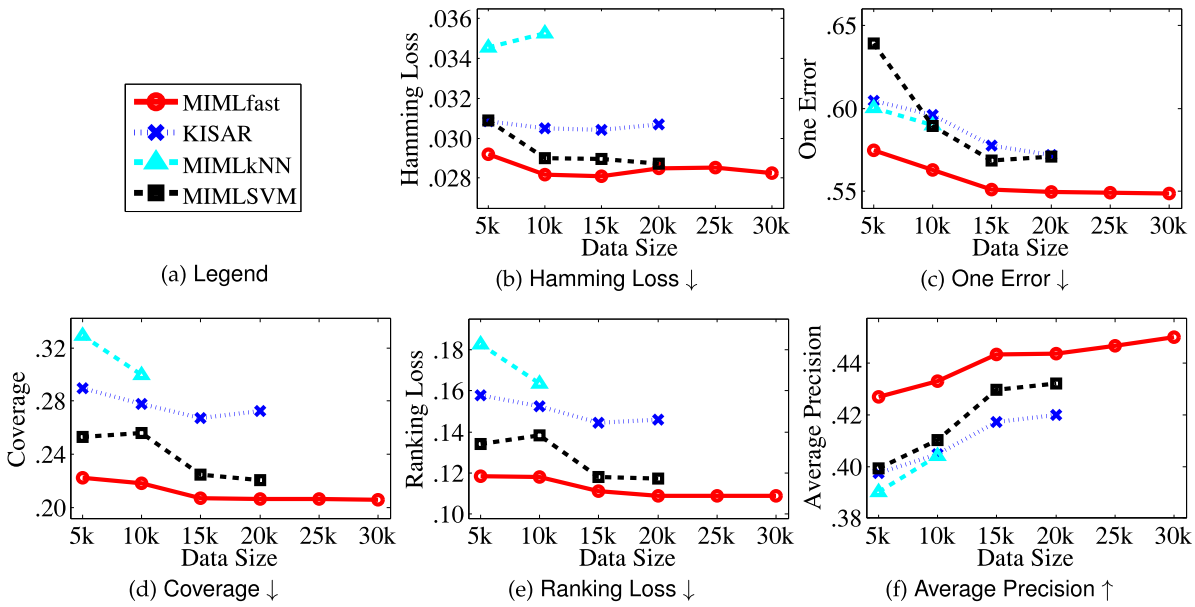


Fig. 6. Comparison results on *MSRA* with varying data size; ↑(↓) indicates that the larger (smaller) the value, the better the performance; only MIMLfast can work when data size reaches 25,000.

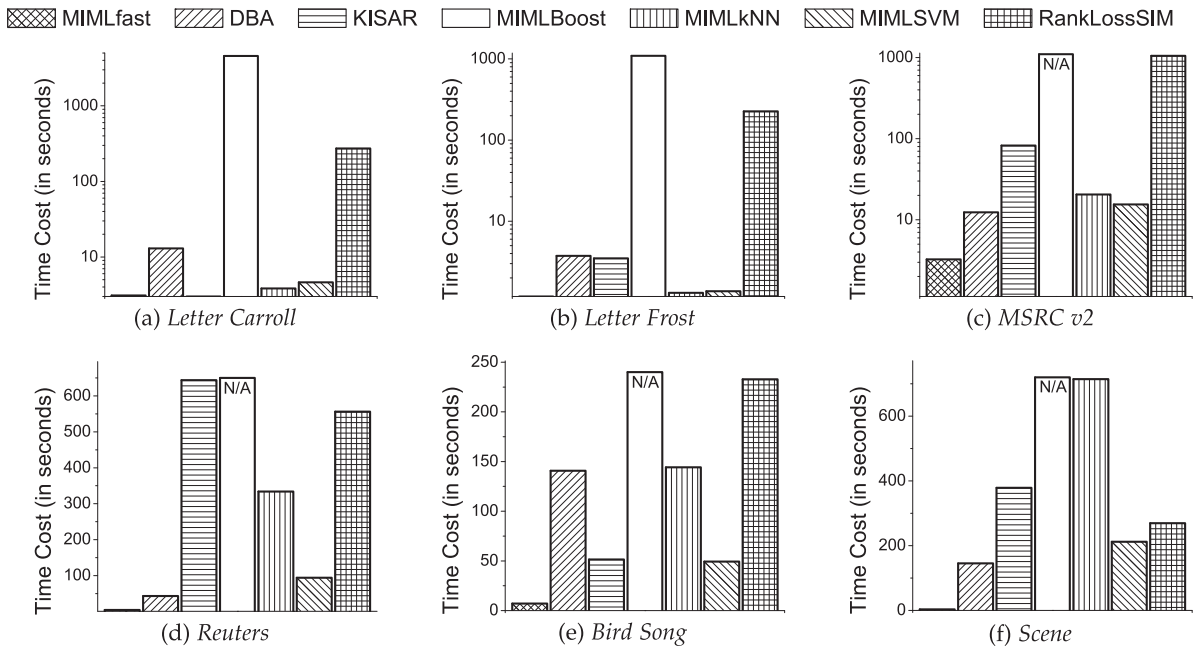


Fig. 7. Comparison of time cost on six moderate-sized data sets; N/A indicates that no result was obtained in 24 hours; the *y*-axis in (a) (b) and (c) are log-scaled.

4.3 Efficiency Comparison

It is crucial to study the efficiency of the compared MIML approaches, because our basic motivation is to develop a method that can work on large scale MIML datasets. All the experiments are performed on a machine with 16×2.60 GHz CPUs and 32 GB main memory. Again, we first show the time cost of each algorithm on the six moderate-sized data sets in Fig. 7. Note that *y*-axis in the first 3 sub-figures are log-scaled. Obviously, our approach is the most efficient one on all the data sets. MIMLBoost is the most time-consuming one, followed by RankLossSIM and MIMLkNN.

We show the time costs of the compared algorithms on *Corel5K* and *MSRA* in Fig. 8, with varying data size. The superiority of our approach is more distinguished on these two larger data sets. On *Corel5K*, MIMLBoost failed to get result in 24 hours even with the smallest subset, while RankLossSIM can handle only 1000 examples. The time costs of existing methods increase dramatically as the data size increases. In contrast, MIMLfast takes only 1 minute even for the largest size in Fig. 8a. In Fig. 8b, on the largest *MSRA* data, the superiority of MIMLfast is even more apparent. None of existing approaches can deal with more than 20K examples. In contrast, on data of 20,000 bags and 180,000 instances, MIMLfast is more than 100 times faster than the other compared approaches; when the data size becomes larger, none of compared approaches can return result in 24 hours, and MIMLfast takes only 12 minutes.

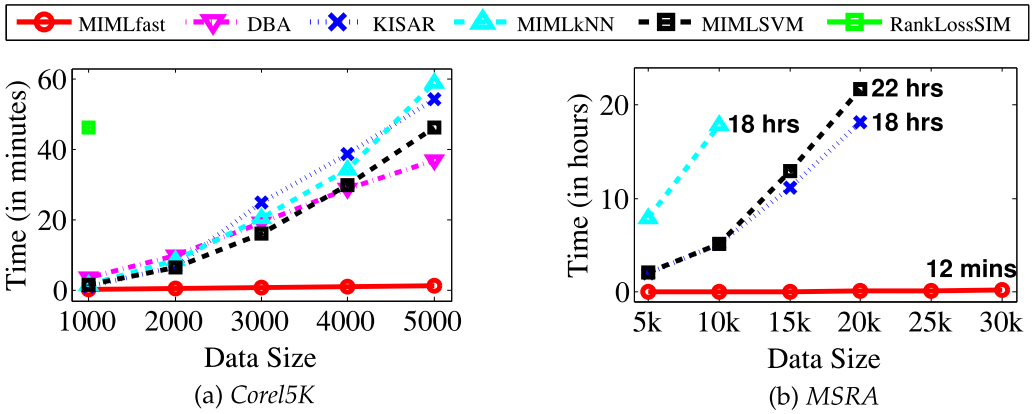


Fig. 8. Comparison of time cost on *Corel5K* and *MSRA* with varying data size.

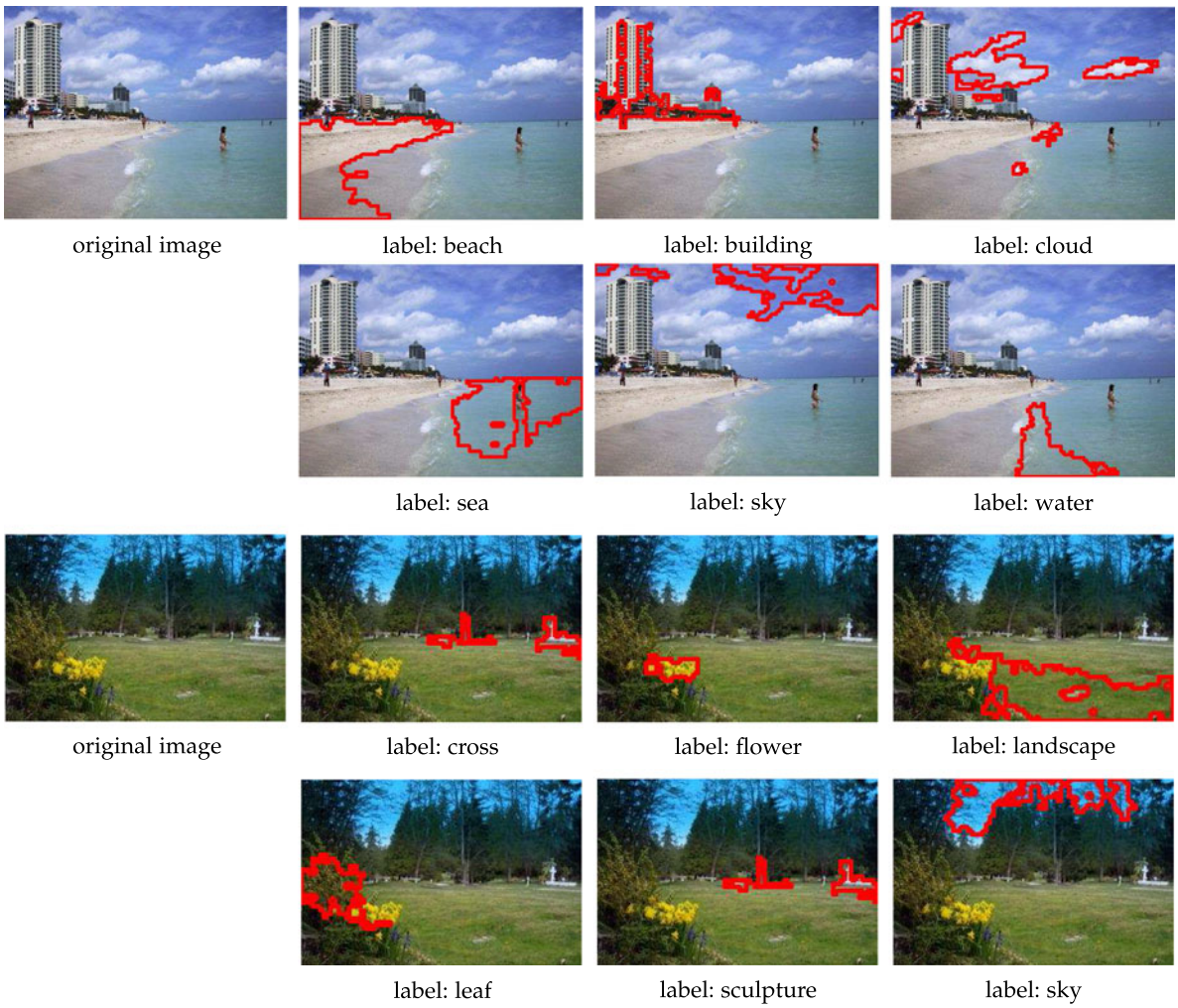


Fig. 9. Key instances identified by MIMLfast for each label. Image regions corresponding to key instances are highlighted with red contours.

4.4 Key Instance Detection

In multi-instance multi-label learning, a set of labels are assigned to a group of instances, and thus it is interesting to understand the relation between input patterns and output label semantics. Inspired by [24], by assuming that each label is triggered by its most positive instance, our MIMLfast approach is able to identify the key instance for each label.

We first give an intuitive evaluation of the key instance detection of MIMLfast. On *MSRA*, following [24], we first partition each image into a set of patches with k-means clustering, and then extract an instance from each cluster. In Fig. 9, we show two example images, and highlight the regions corresponding to the key instance detected by our approach for each label. Note that since the image regions are obtained by clustering, an instance may correspond to multiple regions in the same cluster rather than a single region. The results clearly show that MIMLfast can detect reasonable key instances for the labels.

TABLE 3
Key Instance Detection Accuracy (mean ± std.)

data	*Letter C.*	*Letter F.*	*MSRC v2*	*Bird Song*
MIMLfast-*ins*	**0.81 ± 0.02**	**0.81 ± 0.03**	**0.82 ± 0.02**	**0.83 ± 0.02**
MIMLfast	0.67 ± 0.03	0.67 ± 0.03	0.66 ± 0.03	0.58 ± 0.04
KISAR	0.41 ± 0.03	0.47 ± 0.04	0.62 ± 0.03	0.31 ± 0.03
RankLossSIM	0.67 ± 0.03	0.70 ± 0.03	0.64 ± 0.02	0.42 ± 0.02

The best results based on paired t-tests at 95 percent significance level are bolded.

We also evaluate the key instance detection accuracy quantitatively. On 4 of the 8 MIML data sets, i.e., *Letter Carroll*, *Letter Frost*, *MSRC v2* and *Bird Song*, the instance labels are available, and thus providing a test bed for key instance detection. Among the existing MIML methods, RankLossSIM and KISAR are able to detect key instance for each label, and will be compared with our MIMLfast approach. Note that all of the above three methods do not utilize the instance label information during the training process. As introduced in Section 3.4, when instance labels are available, we can further train the model to optimize the ranking of instances, and thus may detect key instances more accurately. We denote this enhanced version of our algorithm by MIMLfast-*ins*. For KISAR, the key instance is the one closest to the prototype of the label as in [24], while for the other three methods, the key instance for a specific label is identified by selecting the instance with maximum prediction value on that label. We examine the ground truth of the detected key instances and present the accuracies of instance classification in Table 3. As expected, MIMLfast-*ins*, which utilizes the instance labels, achieves the highest accuracy on all of the four datasets. Among the three methods that do not use instance labels, we can observe that KISAR is less accurate than the other two methods, probably because it does not build the model on the instance level, and detects key instance based on unsupervised prototypes. When comparing with RankLossSIM, which is specially designed for MIML instance annotation, our algorithm MIMLfast is more accurate on the two larger data sets, while comparable on *Letter Carroll*, and slightly worse on *Letter Frost*.

4.5 Sub-Concept Discovery

To examine the effectiveness of sub-concept discovery, we run MIMLfast with varying number of sub-concepts on the two benchmark data sets: *Scene* for image classification and *Reuters* for text categorization. Table 4 presents the results with K varying from 1 to 15 with step size of 5. For each value of K, we run 10-fold cross validation and report the average results as well as standard deviations. Note that K is selected by cross validation on the training data in Section 3.2. As shown in Table 4, compared with neglecting the sub-concepts ($K = 1$), the exploitation of sub-concepts is helpful ($K = 5$, 10 and 15 are all better than $K = 1$). When the K gets larger, the difference between results with different K values is not very significant. This may owe to that if we set a K value larger than what is really needed, some sub-concepts might capture no examples, and thus a overly-large K will not make the performance degenerate too much, although it might hamper the efficiency.

We further examine the sub-concepts discovered by MIMLfast. We take the *Scene* data set as an illustration and show some example images of the top-four sub-concepts discovered for the label *sea* in Fig. 10. It is interesting to see that these four sub-concepts are with reasonable but different perceptions: the first sub-concept corresponds to sea with beach and blue sky, the second sub-concept corresponds to big wave in the sea, etc.

4.6 Comparison with Variants

To further examine how MIMLfast works, we study two variants, V1 and V2. V1 gives up W_0 in Eq. (1) and directly learns a linear model for each label. It is constructed to examine whether learning the shared space is helpful. V2 simply selects the top r labels as relevant ones, where r is the average number of relevant labels on the training data. It is constructed to examine whether the dummy label provides a good separation of relevant and irrelevant labels.

Table 5 shows the results on the two benchmark data sets. V1 is significantly worse than MIMLfast on all criteria, implying that learning the shared space for all the labels is better than learning each label independently. On hamming loss, MIMLfast achieves significantly better performance than V2, while on the other four criteria, they achieve comparable performances, implying that the use of dummy label does not affect the rank of the labels but providing a reasonable separation of relevant and irrelevant labels.

TABLE 4
Results (mean ± std.) Obtained by Identifying Different Numbers of Sub-Concepts

	hamming loss$^{\downarrow}$	one error$^{\downarrow}$	coverage$^{\downarrow}$	ranking loss$^{\downarrow}$	average precision$^{\uparrow}$
Scene					
$K = 1$	0.191 ± 0.011	0.366 ± 0.038	0.224 ± 0.018	0.209 ± 0.020	0.754 ± 0.023
$K = 5$	**0.186 ± 0.009**	0.354 ± 0.026	0.213 ± 0.015	0.196 ± 0.017	0.764 ± 0.018
$K = 10$	**0.182 ± 0.014**	**0.338 ± 0.030**	**0.202 ± 0.017**	**0.184 ± 0.018**	**0.777 ± 0.020**
$K = 15$	**0.181 ± 0.011**	**0.344 ± 0.031**	0.210 ± 0.014	0.192 ± 0.016	**0.769 ± 0.019**
Reuters					
$K = 1$	0.027 ± 0.008	0.042 ± 0.013	0.036 ± 0.007	0.015 ± 0.006	0.972 ± 0.010
$K = 5$	0.026 ± 0.006	0.040 ± 0.009	**0.035 ± 0.006**	**0.014 ± 0.005**	0.974 ± 0.007
$K = 10$	**0.025 ± 0.006**	**0.037 ± 0.007**	**0.034 ± 0.006**	**0.013 ± 0.005**	**0.976 ± 0.006**
$K = 15$	**0.025 ± 0.007**	**0.040 ± 0.010**	**0.035 ± 0.007**	**0.014 ± 0.006**	**0.974 ± 0.008**

The best performance and its comparable results based on paired t-tests at 95 percent significance level are bolded.

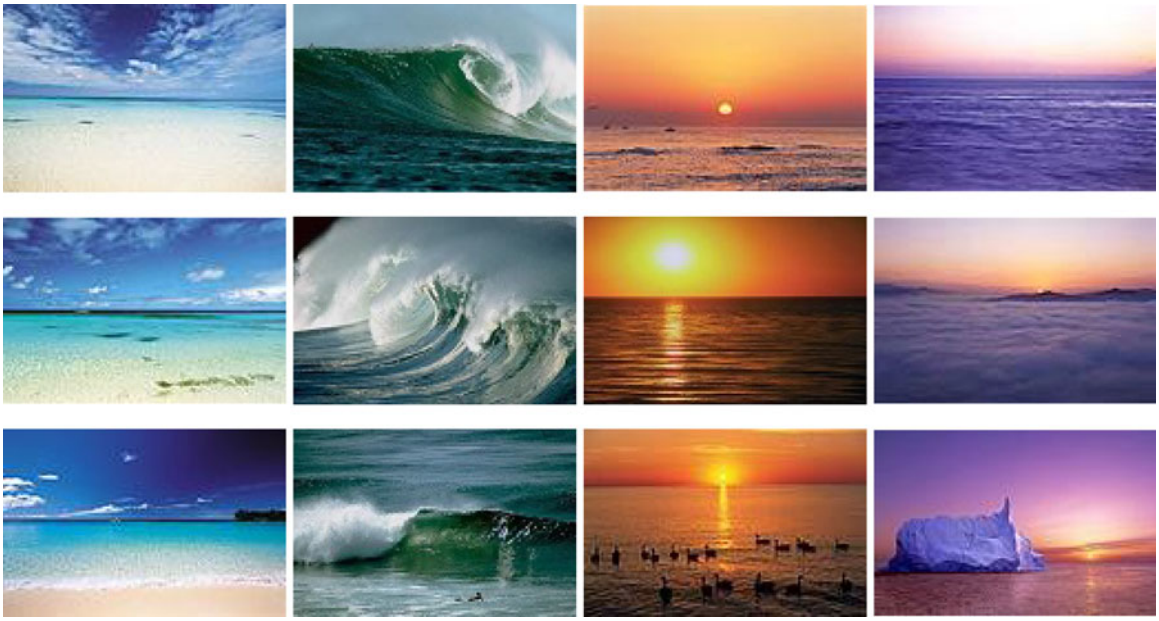


Fig. 10. Example images of different sub-concepts identified for label *sea*, where one column corresponds to one sub-concept.

TABLE 5
Comparison Results (mean ± std.) of MIMLfast with Two Variants (V1 and V2)

	hamming loss$^{\downarrow}$	one error$^{\downarrow}$	coverage$^{\downarrow}$	ranking loss$^{\downarrow}$	average precision$^{\uparrow}$
Scene					
MIMLfast	**0.188 ± 0.009**	**0.351 ± 0.023**	**0.207 ± 0.012**	**0.189 ± 0.014**	**0.770 ± 0.015**
MIMLfast-V1	0.211 ± 0.009	0.409 ± 0.023	0.239 ± 0.011	0.228 ± 0.013	0.730 ± 0.014
MIMLfast-V2	0.196 ± 0.012	**0.358 ± 0.030**	**0.208 ± 0.014**	**0.192 ± 0.016**	**0.767 ± 0.018**
Reuters					
MIMLfast	**0.028 ± 0.004**	**0.044 ± 0.008**	**0.035 ± 0.004**	**0.014 ± 0.004**	**0.972 ± 0.005**
MIMLfast-V1	0.038 ± 0.004	0.060 ± 0.011	0.038 ± 0.005	0.019 ± 0.004	0.963 ± 0.007
MIMLfast-V2	0.035 ± 0.003	0.046 ± 0.010	**0.035 ± 0.004**	**0.015 ± 0.003**	**0.971 ± 0.006**

The best performance and its comparable results based on paired t-tests at 95% significance level are bolded.

5 Conclusion

MIML is a framework for learning with complicated objects, and has been proved to be effective in many applications. However, existing MIML approaches are usually too time-consuming to deal with large scale problems. This paper proposes the MIMLfast approach to learn with MIML examples fast, which extends our preliminary research [18]. On one hand, efficiency is highly improved by optimizing the approximated ranking loss with SGD based on a two level linear model; on the other hand, effectiveness is achieved by exploiting label relations in a shared space and discovering sub-concepts for complicated labels. Moreover, our approach can naturally detect key instance for each label, and thus providing a chance to discover the relation between input patterns and output label semantics. In the future, we will try to optimize other loss functions rather than ranking loss. Also, larger scale problems will be studied.

Acknowledgments

The authors would like to thank the associate editor and reviewers for helpful comments and suggestions. This research was partially supported by National Key R&D Program of China (2018YFB1004300), NSFC (61751306, 61503182, 61503179), Jiangsu SF (BK20150754, BK20150586) and Collaborative Innovation Center of Novel Software Technology and Industrialization.

References

[1] A. Aggarwal, S. Ghoshal, A. Shetty, S. Sinha, G. Ramakrishnan, P. Kar, and P. Jain, "Scalable optimization of multivariate performance measures in multi-instance multi-label learning," in *Proc. 31st AAAI Conf. Artif. Intell.*, 2017, pp. 1698–1704.

[2] S. Andrews, I. Tsochantaridis, and T. Hofmann, "Support vector machines for multiple-instance learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2002, pp. 561–568.

[3] S. Ben-David, D. Loker, N. Srebro, and K. Sridharan, "Minimizing the misclassification error rate using a surrogate convex loss," in *Proc. 29th Int. Conf. Mach. Learn.*, 2012, pp. 83–90.

[4] W. Bi and J. Kwok, "Multi-label classification on tree-and dag-structured hierarchies," in *Proc. 28th Int. Conf. Mach. Learn.*, 2011, pp. 17–24.

[5] H. Blockeel, D. Page, and A. Srinivasan, "Multi-instance tree learning," in *Proc. 22nd Int. Conf. Mach. Learn.*, 2005, pp. 57–64.

[6] L. Bottou, "Large-scale machine learning with stochastic gradient descent," in *Proc. Compstat*, 2010, pp. 177–186.

[7] M. Boutell, J. Luo, X. Shen, and C. Brown, "Learning multi-label scene classification," *Pattern Recognit.*, vol. 37, no. 9, pp. 1757–1771, 2004.

[8] F. Briggs, X. Fern, and R. Raich, "Rank-loss support instance machines for miml instance annotation," in *Proc. 18th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2012, pp. 534–542.

[9] A. Clare and R. D. King, "Knowledge discovery in multi-label phenotype data," in *Proc. 5th Eur. Conf. Principles Practice Knowl. Discovery Databases*, 2001, pp. 42–53.

[10] T. Dietterich, R. Lathrop, and T. Lozano-Pérez, "Solving the multiple instance problem with axis-parallel rectangles," *Artif. Intell.*, vol. 89, no. 1, pp. 31–71, 1997.

[11] P. Duygulu, K. Barnard, J. Freitas, and D. Forsyth, "Object recognition as machine translation: Learning a lexicon for a fixed image vocabulary," in *Proc. 7th Eur. Conf. Comput. Vis.*, 2002, pp. 97–112.

[12] J. Feng and Z.-H. Zhou, "Deep miml network," in *Proc. 31st AAAI Conf. Artif. Intell.*, 2017, pp. 1884–1890.

[13] P. Frey and D. Slate, "Letter recognition using holland-style adaptive classifiers," *Mach. Learn.*, vol. 6, no. 2, pp. 161–182, 1991.

[14] J. Fürnkranz, E. Hüllermeier, E. Loza Mencía, and K. Brinker, "Multilabel classification via calibrated label ranking," *Mach. Learn.*, vol. 73, no. 2, pp. 133–153, 2008.

[15] T. Gärtner, P. A. Flach, A. Kowalczyk, and A. J. Smola, "Multi-instance kernels," in *Proc. 19th Int. Conf. Mach. Learn.*, 2002, pp. 179–186.

[16] S.-J. Huang, S. Chen, and Z.-H. Zhou, "Multi-label active learning: query type matters," in *Proc. 24th Int. Joint Conf. Artif. Intell.*, 2015, pp. 946–952.

[17] S.-J. Huang, N. Gao, and S. Chen, "Multi-instance multi-label active learning," in *Proc. 26th Int. Joint Conf. Artif. Intell.*, 2017, pp. 1886–1892.

[18] S.-J. Huang, W. Gao, and Z.-H. Zhou, "Fast multi-instance multi-label learning," in *Proc. 28th AAAI Conf. Artif. Intell.*, 2014, pp. 1868–1874.

[19] S.-J. Huang, Y. Yu, and Z.-H. Zhou, "Multi-label hypothesis reuse," in *Proc. 18th ACM SIGKDD Conf. Knowl Discovery Data Mining*, 2012, pp. 525–533.

[20] S.-J. Huang and Z.-H. Zhou, "Multi-label learning by exploiting label correlations locally," in *Proc. 26th AAAI Conf. Artif. Intell.*, 2012, page 949–955.

[21] J. T. Kwok and P.-M. Cheung, "Marginalized multi-instance kernels," in *Proc. 20th Int. Joint Conf. Artif. Intell.*, 2007, pp. 901–906.

[22] H. Li, M. Wang, and X. Hua, "Msra-mm 2.0: A large-scale web multimedia dataset," in *Proc. IEEE Int. Conf. Data Mining Workshops*, 2009, pp. 164–169.

[23] Y. Li, S. Ji, S. Kumar, J. Ye, and Z. Zhou, "Drosophila gene expression pattern annotation through multi-instance multi-label learning," in *Proc. 21st Int. Joint Conf. Artif. Intell.*, 2009, pp. 1445–1450.

[24] Y.-F. Li, J.-H. Hu, Y. Jiang, and Z.-H. Zhou, "Towards discovering what patterns trigger what labels," in *Proc. 26th AAAI Conf. Artif. Intell.*, 2012, pp. 1012–1018.

[25] L. Ma, D. Song, L. Liao, and J. Wang, "PSVM: A preference-enhanced SVM model using preference data for classification," *Sci. China Inf. Sci.*, vol. 60, no. 12, pp. 1–14, 2017.

[26] O. Maron and A. Ratan, "Multiple-instance learning for natural scene classification," in *Proc. 15th Int. Conf. Mach. Learn.*, 1998, pp. 341–349.

[27] J. Nam, J. Kim, I. Gurevych, and J. Fürnkranz, "Large-scale multi-label text classification-revisiting neural networks," in *Proc. Eur. Conf. Mach. Learn. Principles Practice Knowl. Discovery Databases*, 2014, pp. 437–452.

[28] C.-T. Nguyen, X. Wang, J. Liu, and Z.-H. Zhou, "Labeling complicated objects: Multi-view multi-instance multi-label learning," in *Proc. 28th AAAI Conf. Artif. Intell.*, 2014, pp. 2013–2019.

[29] N. Nguyen, "A new svm approach to multi-instance multi-label learning," in *Proc. 10th IEEE Int. Conf. Data Mining*, 2010, pp. 384–392.

[30] Y. Pei and X. Z. Fern, "Constrained instance clustering in multi-instance multi-label learning," *Pattern Recognit. Lett.*, vol. 37, pp. 107–114, 2014.

[31] A. Pham, R. Raich, and X. Fern, "Dynamic programming for instance annotation in multi-instance multi-label learning," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 12, pp. 2381–2394, Dec. 2017.

[32] A. Pham, R. Raich, X. Fern, and J. P. Arriaga, "Multi-instance multi-label learning in the presence of novel class instances," in *Proc. 32nd Int. Conf. Mach. Learn.*, 2015, pp. 2427–2435.

[33] H. Robbins and S. Monro, "A stochastic approximation method," *Ann. Math. Statist.*, vol. 22, no. 3, pp. 400–407, 1951.

[34] R. E. Schapire and Y. Singer, "Improved boosting algorithms using confidence-rated prediction," *Mach. Learn.*, vol. 37, no. 3, pp. 297–336, 1999.

[35] R. E. Schapire and Y. Singer, "BoosTexter: A boosting-based system for text categorization," *Mach. Learn.*, vol. 39, no 2/3, pp. 135–168, 2000.

[36] F. Sebastiani, "Machine learning in automated text categorization," *ACM Comput. Surveys*, vol. 34, no. 1, pp. 1–47, 2002.

[37] G. Tsoumakas, I. Katakis, and L. Vlahavas, "Random k-labelsets for multilabel classification," *IEEE Trans. Knowl. Data Eng.*, vol. 23, no. 7, pp. 1079–1089, Jul. 2011.

[38] N. Usunier, D. Buffoni, and P. Gallinari, "Ranking with ordered weighted pairwise classification," in *Proc. 26th Int. Conf. Mach. Learn.*, 2009, pp. 1057–1064.

[39] J. Wang and J.-D. Zucker, "Solving the multi-instance problem: A lazy learning approach," in *Proc. 17th Int. Conf. Mach. Learn.*, 2000, pp. 1119–1125.

[40] J. Weston, S. Bengio, and N. Usunier, "Wsabie: Scaling up to large vocabulary image annotation," in *Proc. 22nd Int. Joint Conf. Artif. Intell.*, 2011, pp. 2764–2770.

[41] J. Winn, A. Criminisi, and T. Minka, "Object categorization by learned universal visual dictionary," in *Proc. 10th IEEE Int. Conf. Comput. Vis.*, 2005, pp. 1800–1807.

[42] M.-K. Xie and S.-J. Huang, "Partial multi-label learning," in *Proc. 32nd AAAI Conf. Artif. Intell.*, 2018, pp. 946–952.

[43] H. Yang, J. T. Zhou, J. Cai, and Y. S. Ong, "Miml-fcn+: Multi-instance multi-label learning via fully convolutional networks with privileged information," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 5996–6004.

[44] S. Yang, H. Zha, and B. Hu, "Dirichlet-Bernoulli alignment: A generative model for multi-class multi-label multi-instance corpora," in *Proc. Adv. Neural Inf. Process. Syst.*, 2009, pp. 2143–2150.

[45] H. Yuan, M. Fang, and X. Zhu, "Hierarchical sampling for multi-instance ensemble learning," *IEEE Trans. Knowl. Data Eng.*, vol. 25, no. 12, pp. 2900–2905, Dec. 2013.

[46] Z. Zha, X. Hua, T. Mei, J. Wang, G. Qi, and Z. Wang, "Joint multi-label multi-instance learning for image classification," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2008, pp. 1–8.

[47] M.-L. Zhang, "A k-nearest neighbor based multi-instance multi-label learning algorithm," in *Proc. 22nd IEEE Int. Conf. Tools Artif. Intell.*, 2010, pp. 207–212.

[48] M.-L. Zhang, Y.-K. Li, X.-Y. Liu, and X. Geng, "Binary relevance for multi-label learning: An overview," *Frontiers Comput. Sci.*, vol. 12, no. 2, pp. 191–202, 2018.

[49] M.-L. Zhang and Z.-J. Wang, "Mimlrbf: Rbf neural networks for multi-instance multi-label learning," *Neurocomputing*, vol. 72, no. 16, pp. 3951–3956, 2009.

[50] M.-L. Zhang and Z.-H. Zhou, "ML-kNN: A lazy learning approach to multi-label learning," *Pattern Recognit.*, vol. 40, no. 7, pp. 2038–2048, 2007.

[51] M.-L. Zhang and Z.-H. Zhou, "A review on multi-label learning algorithms," *IEEE Trans. Knowl. Data Eng.*, vol. 26, no. 8, pp. 1819–1837, Aug. 2014.

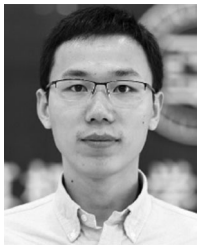
[52] Y. Zhang and Z.-H. Zhou, "Multilabel dimensionality reduction via dependence maximization," *ACM Trans. Knowl. Discovery Data*, vol. 4, no. 3, 2010, Art. no. 14.

[53] Z.-H. Zhou, "A brief introduction to weakly supervised learning," *Nat. Sci. Rev.*, vol. 5, no. 1, pp. 44–53, 2018.

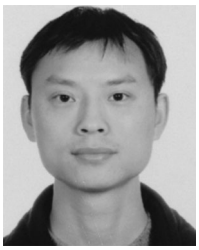
[54] Z.-H. Zhou and M.-L. Zhang, "Multi-instance multi-label learning with application to scene classification," in *Proc. 19th Int. Conf. Neural Inf. Process. Syst.*, pp. 2007, 1609–1616.

[55] Z.-H. Zhou, M.-L. Zhang, S.-J. Huang, and Y.-F. Li, "Multi-instance multi-label learning," *Artif. Intell.*, vol. 176, no. 1, pp. 2291–2320, 2012.

[56] Y. Zhu, K. M. Ting, and Z.-H. Zhou, "Discover multiple novel labels in multi-instance multi-label learning," in *Proc. 31st AAAI Conf. Artif. Intell.*, 2017, pp. 2977–2984.



Sheng-Jun Huang received the BSc and PhD degrees in computer science from Nanjing University, China, in 2008 and 2014, respectively. He is now an associate professor with the College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics. His main research interests include machine learning and data mining. He has been selected to the Young Elite Scientists Sponsorship Program by CAST in 2016, and won the China Computer Federation Outstanding Doctoral Dissertation Award in 2015, the KDD Best Poster Award at the in 2012, and the Microsoft Fellowship Award in 2011. He is a junior associate editor of *the Frontiers of Computer Science*.



Wei Gao received the MSc and PhD degrees from Nankai and Nanjing University, China, in 2009 and 2014, respectively. He joined the Department of Computer Science & Technology, Nanjing University as an assistant professor, in 2014. His research interest is mainly in machine learning theory.

Zhi-Hua Zhou (S'00-M'01-SM'06-F'13) received the BSc, MSc, and PhD degrees in computer science from Nanjing University, China, in 1996, 1998, and 2000, respectively, all with the highest honors. He joined the Department of Computer Science & Technology, Nanjing University as an assistant professor, in 2001, and is currently a professor and standing deputy director of the National Key Laboratory for Novel Software Technology; he is also the founding director of the LAMDA group. His research interests include artificial intelligence, machine learning and data mining. He has authored the books *Ensemble Methods: Foundations and Algorithms* and *Machine Learning* (in Chinese), and published more than 150 papers in top-tier international journals or conference proceedings. He has received various awards/honors including the National Natural Science Award of China, the PAKDD Distinguished Contribution Award, the IEEE ICDM Outstanding Service Award, the Microsoft Professorship Award, etc. He also received 22 patents. He is an executive editor-in-chief of the *Frontiers of Computer Science*, associate editor-in-chief of the *Science China Information Sciences*, action or associate editor of the *Machine Learning*, *the IEEE Transactions on Pattern Analysis and Machine Intelligence*, *the ACM Transactions on Knowledge Discovery from Data*, etc. He served as an associate editor-in-chief for *Chinese Science Bulletin* (2008-2014), associate editor for *the IEEE Transactions on Knowledge and Data Engineering* (2008-2012), *the IEEE Transactions on Neural Networks and Learning Systems* (2014-2017), *the ACM Transactions on Intelligent Systems and Technology* (2009-2017), the *Neural Networks* (2014-2016), *the Knowledge and Information Systems* (2003-2008), etc. He founded ACML (Asian Conference on Machine Learning), served as Advisory Committee member for IJCAI (2015-2016), Steering Committee member for ICDM, PAKDD and PRICAI, and chair of various conferences such as general co-chair of PAKDD 2014 and ICDM 2016, program co-chair of SDM 2013 and IJCAI 2015 Machine Learning Track, and Area chair of NIPS, ICML, AAAI, IJCAI, KDD, etc. He is/was the chair of the IEEE CIS Data Mining Technical Committee (2015-2016), the chair of the CCF-AI(2012-), and the chair of the Machine Learning Technical Committee of CAAI (2006-2015). He is a foreign member of the Academy of Europe, and a fellow of the ACM, AAAI, AAAS, IEEE, IAPR, IET/IEE, CCF, and CAAI.