Fast Multi-Instance Multi-Label Learning*

Sheng-Jun Huang Wei Gao Zhi-Hua Zhou
National Key Laboratory for Novel Software Technology
Nanjing University, Nanjing 210023, China
{huangsj, gaow, zhouzh}@lamda.nju.edu.cn

Abstract

In multi-instance multi-label learning (MIML), one object is represented by multiple instances and simultaneously associated with multiple labels. 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, 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; particularly, on a data set with 30K bags and 270K instances, where none of existing approaches can return results in 24 hours, MIMLfast takes only 12 minutes. 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 semantics.

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 decomposed into multiple instances, and has multiple class labels simultaneously. For example, in image classification problems, an image usually contains multiple objects, and can be divided into several segments, where each segment is represented with an instance, and corresponds to a semantic label (Zhou and Zhang 2007); in text categorization tasks, an article may belong to multiple categories, and can be represented by a bag of instances, one for a paragraph (Yang, Zha, and Hu 2009). Multi-instance multi-label learning (MIML) is a recent proposed framework for such complicated objects (Zhou et al. 2012).

During the past years, many MIML algorithms were proposed (Luo and Orabona 2010; Nguyen 2010; Zhou et al. 2012; Nguyen, Zhan, and Zhou 2013). For example, MIMLSVM (Zhou and Zhang 2007) degenerated the MIML problem into single-instance multi-label tasks to solve. MIMLBoost (Zhou and Zhang 2007) degenerated MIML to multi-instance single-label learning. A generative model for MIML was proposed by Yang et al. (Yang, Zha, and Hu 2009). Zha et al. proposed a hidden conditional random field model for MIML image annotation (Zha et al. 2008). Nearest neighbor approach for MIML was proposed in (Zhang 2010). Briggs et al. proposed to optimize ranking loss for MIML instance annotation (Briggs, Fern, and Raich 2012). Existing MIML approaches achieved decent performances and validated the superiority of MIML in different 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 MIML.

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. To make the learning efficient, we employ stochastic gradient descent (SGD) to optimize an approximated ranking loss. At each iteration, 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. While most MIML methods focus on improving generalization, another important task of MIML is to understand the relation between input patterns and output semantics (Li et al. 2012). MIMLfast can naturally identify the most representative instance for each label. In addition, we propose to discover sub-concepts for complicated labels, which frequently occur in MIML tasks.

*This research was supported by the 973 Program (2010CB327903) and NSFC (61321491).

There are some related work, for example, learning a subspace from the feature space was well studied before (Ando and Zhang 2005; Ji et al. 2008), however, has not been applied to the MIML setting. In (Weston, Bengio, and Usunier 2011), a similar technique was used to optimize the approximated ranking loss for image annotation; however, it dealt with single-instance single-label problem, which is quite different from the MIML problem. In (Zhou et al. 2012), 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 discovers sub-concepts using supervised model rather than heuristic clustering.

The rest of the paper is organized as follows. The MIMLfast approach is proposed in the next section, followed by the experimental study. At last we conclude this work with future issues.

The MIMLfast Approach

We denote by $\{(X_1, Y_1), (X_2, Y_2), \cdots, (X_n, Y_n)\}$ the training data that consists of n examples, where each bag X_i has z_i instances $\{\mathbf{x}_{i,1}, \mathbf{x}_{i,2}, \cdots, \mathbf{x}_{i,z_i}\}$ and Y_i contains the labels associated with X_i, which is a subset of all possible labels $\{y_1, y_2 \cdots y_L\}$.

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 problems 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. In such a way, examples from each label will contribute the optimization of the shared space, and related labels are expected to help each other. 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.

Objects in multi-instance multi-label learning tasks usually have complicated semantic; and thus examples with diverse contents may be assigned the same label. For example, the content of an image labeled *apple* can be a mobile phone, a laptop or just a real apple. It is difficult to train a single model to classify images with such diverse contents into the same category. Instead, we propose to learn multiple models for a complicated label, one for a sub-concept, and automatically decide which sub-concept one example belongs to. The model of each sub-concept is much simpler and 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:

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

where $\mathbf{w}_{l,k}$ corresponds to the k-th 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 (Dietterich, Lathrop, and Lozano-Pérez 1997; Briggs, Fern, and Raich 2012). 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:

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

We call the instance with maximum prediction the key instance of X on label l. With the above model, for an example X and one of its relevant labels l, we define $R(X, l)$ as

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

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 (Usunier, Buffoni, and Gallinari 2009; Weston, Bengio, and Usunier 2011) with respect to an example X on label l as

$$\epsilon(X, l) = \sum\nolimits_{i=1}^{R(X,l)} \frac{1}{i}. \qquad (3)$$

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 data set:

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

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

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

Here we use the convention $0/0 = 0$ if $R(X, l) = 0$. 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 (Ben-David et al. 2012),

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

where $|q|_+ = \max\{q, 0\}$. Accordingly, we penalize $R(X, l)$ with a margin 1, and redefine it as $R(X, l) =$

$\sum_{j \in \bar{Y}} I[f_j(X) > f_l(X) - 1]$. Obviously, Eq. 5 is an upper bound of Eq. 4. We then employ stochastic gradient descent (SGD) (Robbins and Monro 1951) 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:

$$\mathcal{L}(X, y, \bar{y}) = \epsilon(X, y)|1 + f_{\bar{y}}(X) - f_y(X)|_+. \quad (6)$$

We call $\bar{y}$ a violated label if it induces a nonzero loss, i.e., $f_{\bar{y}}(X) > f_y(X) - 1$. In the cases $R(X, y) > 0$, by excluding the inviolated irrelevant labels (which do not induce losses) from $\bar{Y}$, the probability of randomly choosing a violated irrelevant label $\bar{y}$ is $1/R(X, y)$, and thus $\Psi(X, y)$ can be viewed as the expectation of $\mathcal{L}(X, y, \bar{y})$.

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. (Weston, Bengio, and Usunier 2011). 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. Without loss of generality, we assume that the first violated label is found at the v-th sampling step, and then, $R(X, y)$ can be approximated by $\lfloor |\bar{Y}|/v \rfloor$. We assume that the triplet sampled at the t-th SGD iteration is $(X, y, \bar{y})$, on label y, the key instance is $\mathbf{x}$, and achieves the maximum prediction on the k-th 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:

$$\mathcal{L}(X, y, \bar{y}) = \epsilon(X, y)|1 + f_{\bar{y}}(X) - f_y(X)|_+$$
$$\approx \begin{cases} 0 & \text{if } \bar{y} \text{ is not violated;} \\ S_{\bar{Y},v}(1 + [\mathbf{w}^t_{\bar{y},\bar{k}}]^\top W^t_0 \bar{\mathbf{x}} - [\mathbf{w}^t_{y,k}]^\top W^t_0 \mathbf{x}) & \text{otherwise.} \end{cases}$$

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:

$$W^{t+1}_0 = W^t_0 - \gamma_t S_{\bar{Y},v}(\mathbf{w}^t_{\bar{y},\bar{k}} \bar{\mathbf{x}}^\top - \mathbf{w}^t_{y,k} \mathbf{x}^\top) \quad (7)$$

$$\mathbf{w}^{t+1}_{y,k} = \mathbf{w}^t_{y,k} + \gamma_t S_{\bar{Y},v} W^t_0 \mathbf{x} \quad (8)$$

$$\mathbf{w}^{t+1}_{\bar{y},\bar{k}} = \mathbf{w}^t_{\bar{y},\bar{k}} - \gamma_t S_{\bar{Y},v} W^t_0 \bar{\mathbf{x}} \quad (9)$$

where γ_t is the step size of SGD. 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 $\mathbf{w}^k_l$ for all labels l 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. 7 to 9. At last, the updated parameters are normalized such that their norms will be upper bounded by C. This procedure is repeated until the stop criterion reached. In our experiments, we sample

Algorithm 1 The MIMLfast algorithm

```
 1: INPUT:
 2:    training data, parameters $m$, $C$, $K$ and $\gamma_t$
 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 relevant
            label $y$
 7:       select key instance and sub-concept by
            $(\mathbf{x}, k) = \arg\max_{\mathbf{x} \in X, k \in \{1 \cdots K\}} f_{y,k}(\mathbf{x})$
 8:       for $i = 1 : |\bar{Y}|$
 9:          sample an irrelevant label $\bar{y}$ from $\bar{Y}$
10:          select key instance and sub-concept by
               $(\bar{\mathbf{x}}, \bar{k}) = \arg\max_{\mathbf{x} \in X, \bar{k} \in \{1 \cdots K\}} f_{\bar{y},\bar{k}}(\mathbf{x})$
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. 7 to 9,
                  and perform normalization
14:             break
15:    until stop criterion reached
16: TEST:
17:    Relevant labels set for the test bag $X_{\text{test}}$ is:
         $\{l | 1 + f_l(X_{\text{test}}) > f_{\hat{y}}(X_{\text{test}})\}$
```

a small validation set from the training data, and stop the training once the ranking loss does not decrease on the validation set. We have presented some theoretical results on the convergence of the algorithm in a technical report (Huang and Zhou 2013b). Also, we have employed this technique in (Huang and Zhou 2013a) for multi-label active learning.

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, one can 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 label; and thus one does not know how many labels should be selected as relevant ones from the ranked label list (Fürnkranz et al. 2008). 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 to 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 (i.e., whether $y = \hat{y}$). If $y = \hat{y}$, then $\bar{Y}$ consists of all the irrelevant labels, which implies that y (the dummy label) will be ranked before all irrelevant labels; otherwise, $\bar{Y}$ contains both the dummy label and all the irrelevant labels, which implies that the relevant label y will be ranked before both dummy label and 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 ranked before the dummy label are selected as relevant labels.

Table 1: Data sets (6 moderate size and 2 large size)

Data	# ins.	# bag	# label	# label/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

Experiments

Experiments are performed on 6 moderate-sized data sets, including *LetterFrost* (Briggs, Fern, and Raich 2012), *LetterCarroll* (Briggs, Fern, and Raich 2012), *MSRC v2* (Winn, Criminisi, and Minka 2005), *Reuters* (Sebastiani 2002) *Bird Song* (Briggs, Fern, and Raich 2012) and *Scene* (Zhou and Zhang 2007), and 2 large data sets, including *Corel5K* (Duygulu et al. 2002) and *MSRA* (Li, Wang, and Hua 2009). The detailed characteristics of these data sets are summarized in Table 1. Note that MIML is a new learning framework different from multi-instance learning (MI) or multi-label learning (ML); MI and ML algorithms cannot be applied to these MIML data sets directly. We compare MIMLfast with six state-of-the-art MIML methods: DBA (Yang, Zha, and Hu 2009), KISAR (Li et al. 2012), MIMLBoost (Zhou and Zhang 2007), MIMLkNN (Zhang 2010), MIMLSVM (Zhou and Zhang 2007) and RankLossSIM (Briggs, Fern, and Raich 2012).

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. Although MIMLfast is designed to optimize ranking loss, we evaluate the performances of the compared approaches on five commonly used MIML criteria: hamming loss, one error, coverage, ranking loss and average precision. Note that 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 (Schapire and Singer 2000; Zhou et al. 2012). For MIMLfast, the step size is in the form $\gamma_t = \gamma_0/(1 + \eta\gamma_0 t)$ according to (Bottou 2010). 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 $\eta \in \{10^{-5}, 10^{-6}\}$. For the compared approaches, parameters are determined in the same way if no value suggested in their literatures.

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 4 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 Figures 1 and 2, 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 Figures 1 and 2, MIMLfast is apparently better than the others on these two large data sets. Particularly, when data size reaches 25K, other methods cannot work, but MIMLfast still works well.

To have an overall evaluation of the performances of the methods over all datasets, we performed the Bonferroni-Dunn test (Demšar 2006) at 90% significance level. Results show that on both coverage and ranking loss, MIMLfast is significantly better than all the other approaches. And on the other three measures, MIMLfast achieves the best performance along with KISAR and MIMLkNN.

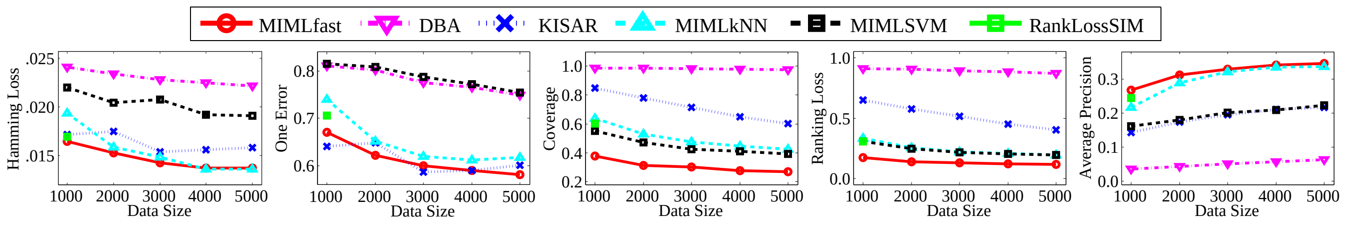
Efficiency Comparison

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 data. All the experiments are performed on a machine with 16×2.60 GHz CPUs and 32GB main memory.

Again, we first show the time cost of each algorithm on the six moderate-sized data sets in Figure 3. Since the results on the two smallest data sets *Letter Carroll* and *Letter Frost* are similar, we take one of them as representative to save space. 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. Based on paired t-tests at 95% significance level, the superiority of MIMLfast to all the other methods is significant. The superiority of MIMLfast is more distinguished on larger data sets. As shown in Figure 4, 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 Figure 4(b). In Figure 4(c), on the largest *MSRA* data, 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 most efficient existing approach; when the data size becomes larger, none of existing approaches can return result in 24 hours, and MIMLfast takes only 12 minutes.

Table 2: Comparison results (mean±std.) on moderate-sized data sets. ↑(↓) 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.

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



(a) Hamming Loss ↓ (b) One Error ↓ (c) Coverage ↓ (d) Ranking Loss ↓ (e) Average Precision ↑

Figure 1: Comparison results on *Corel5K* with varying data size; ↑(↓) indicates that the larger (smaller) the value, the better the performance.

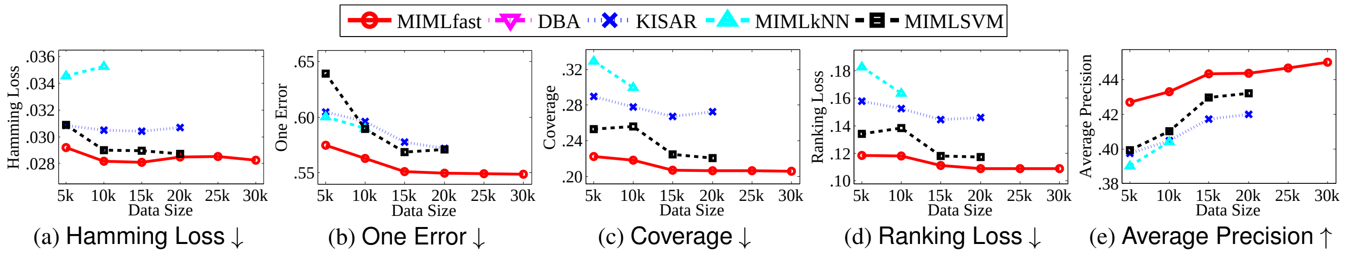


Figure 2: Comparison results on *MSRA* with varying data size; only MIMLfast can work when data size reaches 25,000.

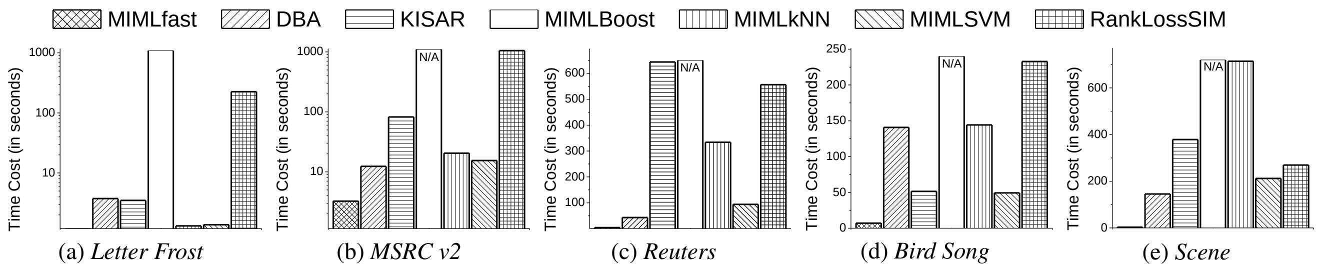


Figure 3: Comparison of time cost on six moderate-sized data sets; N/A indicates that no result was obtained in 24 hours; the y-axis in (a) and (b) are log-scaled.

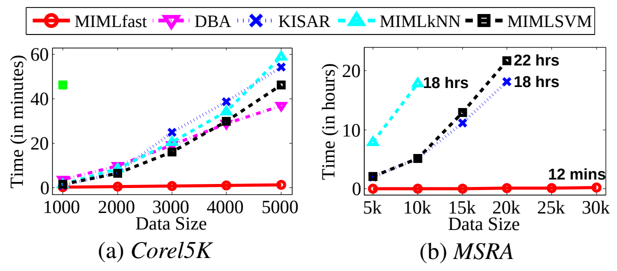


Figure 4: Comparison of time cost on *Corel5K* and *MSRA*.

Key Instance Detection

In MIML, 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 semantics. 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. On 4 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, and will be compared with our approach. For MIMLfast and RankLossSIM, the key instance for a specific label is identified by selecting the instance with maximum prediction value, while for KISAR, key instance is the one closest to the prototype as in (Li et al. 2012). We examine the ground truth of the detected key instances and present the accuracies in Table 3. We can observe that KISAR is less accurate than the other two methods, probably because it does not build the model

Table 3: Key instance detection accuracy (mean±std.). The best results are bolded.

	MIMLfast	KISAR	RankLossSIM
LetterCarroll	**0.67±0.03**	0.41±0.03	**0.67±0.03**
LetterFrost	0.67±0.03	0.47±0.04	**0.70±0.03**
MSRC v2	**0.66±0.03**	0.62±0.03	0.64±0.02
Bird Song	**0.58±0.04**	0.31±0.03	0.42±0.02

on the instance level, and detects key instance based on unsupervised prototypes. When compared with RankLossSIM, which is specially designed for instance annotation, MIMLfast is more accurate on the two larger data sets, while comparable on *Letter Carroll*, and slightly worse on *Letter Frost*.

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. In this paper, we propose the MIMLfast approach to learn with MIML examples fast. 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, MIMLfast can detect key instance for each label, 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 and non-linear models will be studied.

References

Ando, R. K., and Zhang, T. 2005. A framework for learning predictive structures from multiple tasks and unlabeled data. *The Journal of Machine Learning Research* 6:1817–1853.

Ben-David, S.; Loker, D.; Srebro, N.; and Sridharan, K. 2012. Minimizing the misclassification error rate using a surrogate convex loss. In *Proceedings of the 29th International Conference on Machine Learning*.

Bottou, L. 2010. Large-scale machine learning with stochastic gradient descent. *Compstat*.

Briggs, F.; Fern, X.; and Raich, R. 2012. Rank-loss support instance machines for miml instance annotation. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 534–542.

Demšar, J. 2006. Statistical comparisons of classifiers over multiple data sets. *The Journal of Machine Learning Research* 7:1–30.

Dietterich, T.; Lathrop, R.; and Lozano-Pérez, T. 1997. Solving the multiple instance problem with axis-parallel rectangles. *Artificial Intelligence* 89(1):31–71.

Duygulu, P.; Barnard, K.; Freitas, J.; and Forsyth, D. 2002. Object recognition as machine translation: Learning a lexicon for a fixed image vocabulary. In *Proceedings of the 7th European Conference on Computer Vision*, 97–112.

Fürnkranz, J.; Hüllermeier, E.; Loza Mencía, E.; and Brinker, K. 2008. Multilabel classification via calibrated label ranking. *Machine Learning* 73(2):133–153.

Huang, S.-J., and Zhou, Z.-H. 2013a. Active query driven by uncertainty and diversity for incremental multi-label learning. In *Proceedings of the 13th IEEE International Conference on Data Mining*, 1079–1084.

Huang, S.-J., and Zhou, Z.-H. 2013b. Fast multi-instance multi-label learning. *CoRR* abs/1310.2049.

Ji, S.; Tang, L.; Yu, S.; and Ye, J. 2008. Extracting shared subspace for multi-label classification. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 381–389.

Li, Y.-F.; Hu, J.-H.; Jiang, Y.; and Zhou, Z.-H. 2012. Towards discovering what patterns trigger what labels. In *Proceedings of the 26th AAAI Conference on Artificial Intelligence*, 1012–1018.

Li, H.; Wang, M.; and Hua, X. 2009. Msra-mm 2.0: A large-scale web multimedia dataset. In *Proceedings of the IEEE International Conference on Data Mining Workshops*, 164–169.

Luo, J., and Orabona, F. 2010. Learning from candidate labeling sets. In *Advances in Neural Information Processing Systems 23*. Cambridge, MA: MIT Press.

Nguyen, C.-T.; Zhan, D.-C.; and Zhou, Z.-H. 2013. Multimodal image annotation with multi-instance multi-label lda. In *Proceedings of the Twenty-Third international joint conference on Artificial Intelligence*, 1558–1564.

Nguyen, N. 2010. A new svm approach to multi-instance multi-label learning. In *Proceedings of the 10th IEEE International Conference on Data Mining*, 384–392.

Robbins, H., and Monro, S. 1951. A stochastic approximation method. *The Annals of Mathematical Statistics* 22(3):400–407.

Schapire, R. E., and Singer, Y. 2000. BoosTexter: A boosting-based system for text categorization. *Machine Learning* 39(2-3):135–168.

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

Usunier, N.; Buffoni, D.; and Gallinari, P. 2009. Ranking with ordered weighted pairwise classification. In *Proceedings of the 26th International Conference on Machine Learning*, 1057–1064.

Weston, J.; Bengio, S.; and Usunier, N. 2011. Wsabie: Scaling up to large vocabulary image annotation. In *Proceedings of the 22nd International Joint Conference on Artificial Intelligence*, 2764–2770.

Winn, J.; Criminisi, A.; and Minka, T. 2005. Object categorization by learned universal visual dictionary. In *10th IEEE International Conference on Computer Vision*, 1800–1807.

Yang, S.; Zha, H.; and Hu, B. 2009. Dirichlet-bernoulli alignment: A generative model for multi-class multi-label multi-instance corpora. In *Advances in Neural Information Processing Systems 22*. Cambridge, MA: MIT Press. 2143–2150.

Zha, Z.; Hua, X.; Mei, T.; Wang, J.; Qi, G.; and Wang, Z. 2008. Joint multi-label multi-instance learning for image classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 1–8.

Zhang, M.-L. 2010. A k-nearest neighbor based multi-instance multi-label learning algorithm. In *Proceedings of the 22nd IEEE International Conference on Tools with Artificial Intelligence*, 207–212.

Zhou, Z.-H., and Zhang, M.-L. 2007. Multi-instance multi-label learning with application to scene classification. In *Advances in Neural Information Processing Systems 19*. Cambridge, MA: MIT Press. 1609–1616.

Zhou, Z.-H.; Zhang, M.-L.; Huang, S.-J.; and Li, Y.-F. 2012. Multi-instance multi-label learning. *Artificial Intelligence* 176(1):2291–2320.