Ml-knn: A Lazy Learning Approach to Multi-Label Learning

Min-Ling Zhang, Zhi-Hua Zhou [*]

National Laboratory for Novel Software Technology
Nanjing University, Nanjing 210093, China

Abstract: Multi-label learning originated from the investigation of text categorization problem, where each document may belong to several predefined topics simultaneously. In multi-label learning, the training set is composed of instances each associated with a set of labels, and the task is to predict the label sets of unseen instances through analyzing training instances with known label sets. In this paper, a multi-label lazy learning approach named Ml-knn is presented, which is derived from the traditional k-Nearest Neighbor (kNN) algorithm. In detail, for each unseen instance, its k nearest neighbors in the training set are firstly identified. After that, based on statistical information gained from the label sets of these neighboring instances, i.e. the number of neighboring instances belonging to each possible class, maximum a posteriori (MAP) principle is utilized to determine the label set for the unseen instance. Experiments on three different real-world multi-label learning problems, i.e. Yeast gene functional analysis, natural scene classification and automatic web page categorization, show that Ml-knn achieves superior performance to some well-established multi-label learning algorithms.

Keywords: Machine learning, multi-label learning, lazy learning, k-nearest neighbor, functional genomics, natural scene classification, text categorization

1 Introduction

Multi-label learning tasks are omnipresent in real-world problems. For instance, in text categorization, each document may belong to several predefined topics, such as *government* and *health* [14,19]; in functional genomics, each gene may be associated with a set of functional classes, such as *metabolism*, *transcription* and *protein synthesis* [7]; in scene classification, each scene im-

[*] Corresponding author. Tel.: +86-25-8368-6268; Fax: +86-25-8368-6268; E-mail: zhouzh@nju.edu.cn

age may belong to several semantic classes, such as *beach* and *urban* [2]. In all these cases, each instance in the training set is associated with a set of labels, and the task is to output a label set whose size is unknown *a priori* for each unseen instance.

Traditional two-class and multi-class problems can both be cast into multi-label ones by restricting each instance to have only one label. On the other hand, the generality of multi-label problems inevitably makes it more difficult to learn. An intuitive approach to solving multi-label problem is to decompose it into multiple independent binary classification problems (one per category). However, this kind of method does not consider the correlations between the different labels of each instance and the expressive power of such a system can be weak [14,19,7]. Fortunately, several approaches specially designed for multi-label learning tasks have been proposed, such as multi-label text categorization algorithms [14,19,11,13], multi-label decision trees [3,4], multi-label kernel methods [7,2] and multi-label neural networks [23]. In this paper, a lazy learning algorithm named ML-KNN, i.e. Multi-Label K-Nearest Neighbor, is proposed, which is the first multi-label lazy learning algorithm. As its name implied, ML-KNN is derived from the popular K-Nearest Neighbor (KNN) algorithm [1]. Firstly, for each test instance, its K nearest neighbors in the training set are identified. Then, according to statistical information gained from the label sets of these neighboring instances, i.e. the number of neighboring instances belonging to each possible class, maximum a posteriori (MAP) principle is utilized to determine the label set for the test instance. The effectiveness of ML-KNN is evaluated through three different multi-label learning problems, i.e. Yeast gene functional analysis [7], natural scene classification and automatic web page categorization [21]. Experimental results show that the performance of ML-KNN is superior to those of some well-established multi-label learning methods.

The rest of this paper is organized as follows. In Section 2, notations and evaluation metrics used in multi-label learning are briefly introduced. In Section 3, previous works on multi-label learning are reviewed. In Section 4, ML-KNN is proposed. In Section 5, experimental results of ML-KNN and other multi-label learning algorithms are presented. Finally in Section 6, the main contribution of this paper is summarized.

2 Preliminaries

Let $\mathcal{X}$ denote the domain of instances and let $\mathcal{Y} = \{1, 2, \ldots, Q\}$ be the finite set of labels. Given a training set $T = \{(x_1, Y_1), (x_2, Y_2), ..., (x_m, Y_m)\}$ ($x_i \in \mathcal{X}$, $Y_i \subseteq \mathcal{Y}$) i.i.d. drawn from an unknown distribution D, the goal of the learning system is to output a multi-label classifier $h : \mathcal{X} \rightarrow 2^{\mathcal{Y}}$ which optimizes some specific evaluation metric. In most cases however, instead of outputting a multi-label classifier, the learning system will produce a real-valued function of the form $f : \mathcal{X} \times \mathcal{Y} \rightarrow \mathcal{R}$. It is supposed that, given an instance x_i and its associated label set Y_i, a successful learning system will tend to output larger values for labels in Y_i than those not in Y_i, i.e. $f(x_i, y_1) > f(x_i, y_2)$ for any $y_1 \in Y_i$ and $y_2 \notin Y_i$.

The real-valued function $f(\cdot,\cdot)$ can be transformed to a ranking function $rank_f(\cdot,\cdot)$, which maps the outputs of $f(x_i, y)$ for any $y \in \mathcal{Y}$ to $\{1, 2, \ldots, Q\}$ such that if $f(x_i, y_1) > f(x_i, y_2)$ then $rank_f(x_i, y_1) < rank_f(x_i, y_2)$. Note that the corresponding multi-label classifier $h(\cdot)$ can also be derived from the function $f(\cdot,\cdot)$: $h(x_i) = \{y | f(x_i, y) > t(x_i),\ y \in \mathcal{Y}\}$, where $t(\cdot)$ is a threshold function which is usually set to be the zero constant function.

Performance evaluation of multi-label learning system is different from that of classic single-label learning system. Popular evaluation metrics used in single-label system include accuracy, precision, recall and F-measure [20]. In multi-label learning, the evaluation is much more complicated. For a test set $S = \{(x_1, Y_1), (x_2, Y_2), ..., (x_p, Y_p)\}$, the following multi-label evaluation metrics proposed in [19] are used in this paper:

(1)*hamming loss*: evaluates how many times an instance-label pair is misclassified, i.e. a label not belonging to the instance is predicted or a label belonging to the instance is not predicted. The performance is perfect when $\text{hloss}_S(h) = 0$; the *smaller* the value of $\text{hloss}_S(h)$, the better the performance.

$$\text{hloss}_S(h) = \frac{1}{p}\sum_{i=1}^{p}\frac{1}{Q}|h(x_i)\Delta Y_i| \tag{1}$$

where Δ stands for the symmetric difference between two sets. Note that when $|Y_i| = 1$ for all instances, a multi-label system is in fact a multi-class single-label one and the hamming loss is $\frac{2}{Q}$ times the usual classification error.

While hamming loss is based on the multi-label classifier $h(\cdot)$, the following metrics are defined based on the real-valued function $f(\cdot,\cdot)$ which concern the ranking quality of different labels for each instance:

(2)*one-error*: evaluates how many times the top-ranked label is not in the set of proper labels of the instance. The performance is perfect when $\text{one-error}_S(f) = 0$; the *smaller* the value of $\text{one-error}_S(f)$, the better the performance.

$$\text{one-error}_S(f) = \frac{1}{p}\sum_{i=1}^{p}[\![[\arg\max_{y\in\mathcal{Y}} f(x_i, y)] \notin Y_i]\!] \tag{2}$$

where for any predicate π, $[\![\pi]\!]$ equals 1 if π holds and 0 otherwise. Note that, for single-label classification problems, the *one-error* is identical to ordinary

classification error.
(3)*coverage*: evaluates how far we need, on the average, to go down the list of labels in order to cover all the proper labels of the instance. It is loosely related to precision at the level of perfect recall. The *smaller* the value of $\text{coverage}_S(f)$, the better the performance.

$$\text{coverage}_S(f) = \frac{1}{p}\sum_{i=1}^{p}\max_{y\in Y_i} rank_f(x_i, y) - 1 \tag{3}$$

(4)*ranking loss*: evaluates the average fraction of label pairs that are reversely ordered for the instance. The performance is perfect when $\text{rloss}_S(f) = 0$; the *smaller* the value of $\text{rloss}_S(f)$, the better the performance.

$$\text{rloss}_S(f) = \frac{1}{p}\sum_{i=1}^{p}\frac{1}{|Y_i||\overline{Y}_i|}|\{(y_1, y_2)|f(x_i, y_1) \leq f(x_i, y_2),\ (y_1, y_2) \in Y_i \times \overline{Y_i}\}| \tag{4}$$

where $\overline{Y}$ denotes the complementary set of Y in $\mathcal{Y}$.
(5)*average precision*: evaluates the average fraction of labels ranked above a particular label $y \in Y$ which actually are in Y. It is originally used in information retrieval (IR) systems to evaluate the document ranking performance for query retrieval [17]. The performance is perfect when $\text{avgprec}_S(f) = 1$; the *bigger* the value of $\text{avgprec}_S(f)$, the better the performance.

$$\text{avgprec}_S(f) = \frac{1}{p}\sum_{i=1}^{p}\frac{1}{|Y_i|}\sum_{y\in Y_i}\frac{|\{y'|rank_f(x_i, y') \leq rank_f(x_i, y),\ y' \in Y_i\}|}{rank_f(x_i, y)} \tag{5}$$

3 Previous Works Review

As stated in Section 1, the generality of multi-label problems inevitably makes it more difficult to solve than traditional single-label (two-class or multi-class) problems. Until now, only a few literatures on multi-label learning are available, which mainly concern the problems of text categorization [14,19,4,21,11,13], bioinformatics [3,7,23] and scene classification [2].

Research of multi-label learning was initially motivated by the difficulty of concept ambiguity encountered in text categorization, where each document may belong to several topics (labels) simultaneously. One famous approach to solving this problem is BoosTexter proposed by Schapire and Singer [19], which is in fact extended from the popular ensemble learning method AdaBoost [9]. In the training phase, BoosTexter maintains a set of weights over both training examples and their labels, where training examples and their corresponding labels that are hard (easy) to predict correctly get incrementally higher (lower) weights. McCallum [14] proposed a Bayesian approach to multi-label document classification, where a mixture probabilistic model (one mixture component per category) is assumed to generate each document and EM [5] algorithm is utilized to learn the mixture weights and the word distributions in each mixture component. Ueda and Saito [21] presented two types of probabilistic generative models for multi-label text called parametric mixture models (Pmm1, Pmm2), where the basic assumption under Pmms is that multi-label text has a mixture of characteristic words appearing in single-label text that belong to each category of the multi-categories. It is worth noting that the generative models used in [14] and [21] are both based on learning text frequencies in documents, and are thus specific to text applications. Comité et al. [4] extended alternating decision tree [8] to handle multi-label data, where the AdaBoost.MH algorithm proposed by Schapire and Singer [18] is employed to train the multi-label alternating decision trees.

Gao et al. [11] generalized the maximal figure-of-merit (MFoM) approach [10] for binary classifier learning to the case of multiclass, multi-label text categorization. They defined a continuous and differentiable function of the classifier parameters to simulate specific performance metrics, such as precision and recall etc. (micro-averaging F_1 in their paper). Their method assigns a uniform score function to each category of interest for each given test example, and thus the classical Bayes decision rules can be applied. Kazawa et al. [13] converted the original multi-label learning problem of text categorization into a multiclass single-label problem by regarding a set of topics (labels) as a new class. To cope with the data sparseness caused by the huge number of possible classes (Q topics will yield 2^Q classes), they embedded labels into a similarity-induced vector space in which prototype vectors of similar labels will be placed close to each other. They also provided an approximation method in learning and efficient classification algorithms in testing to overcome the demanding computational cost of their method.

In addition to text categorization, multi-label learning has also manifested its effectiveness in other real-world applications, such as bioinformatics and scene classification. Clare and King [3] adapted C4.5 decision tree [16] to handle multi-label data (gene expression in their case) through modifying the definition of entropy. They chose decision trees as the baseline algorithm because of its output (equivalently a set of symbolic rules) is interpretable and can be compared with existing biological knowledge. It is also noteworthy that their goal is to learn a set of accurate rules, not necessarily a complete classification. Through defining a special cost function based on *ranking loss* (as shown in Eq.(4)) and the corresponding margin for multi-label models, Elisseeff and Weston [7] proposed a kernel method for multi-label classification and tested their algorithm on a Yeast gene functional classification problem with positive results. Zhang and Zhou [23] designed the multi-label version of BP neural network through employing a novel error function capturing the characteristics of multi-label learning, i.e. the labels belonging to an instance should be ranked higher than those not belonging to that instance. Boutell et al. [2] applied multi-label learning techniques to scene classification. They decomposed the multi-label learning problem into multiple independent binary classification problems (one per category), where each example associated with label set Y will be regarded as positive example when building classifier for class $y \in Y$ while regarded as negative example when building classifier for class $y \notin Y$. They also provided various labeling criteria to predict a set of labels for each test instance based on its output on each binary classifier. Note that although most works on multi-label learning assume that an instance can be associated with multiple valid labels, there are also works assuming that only one of the labels associated with an instance is correct [12] [1].

4 ML-KNN

For convenience, several notations are introduced before presenting ML-KNN. Given an instance x and its associated label set $Y \subseteq \mathcal{Y}$, suppose K nearest neighbors are considered in the ML-KNN method. Let $\vec{y}_x$ be the category vector for x, where its l-th component $\vec{y}_x(l)$ $(l \in \mathcal{Y})$ takes the value of 1 if $l \in Y$ and 0 otherwise. In addition, let $N(x)$ denote the set of K nearest neighbors of x identified in the training set. Thus, based on the label sets of these neighbors, a *membership counting* vector can be defined as:

$$\vec{C}_x(l) = \textstyle\sum_{a \in N(x)} \vec{y}_a(l), \ l \in \mathcal{Y} \tag{6}$$

where $\vec{C}_x(l)$ counts the number of neighbors of x belonging to the l-th class.

For each test instance t, ML-KNN firstly identifies its K nearest neighbors $N(t)$ in the training set. Let H_1^l be the event that t has label l, while H_0^l be the event that t has not label l. Furthermore, let E_j^l $(j \in \{0, 1, \ldots, \mathrm{K}\})$ denote the event that, among the K nearest neighbors of t, there are exactly j instances which have label l. Therefore, based on the membership counting vector $\vec{C}_t$, the category vector $\vec{y}_t$ is determined using the following maximum a posteriori principle:

$$\vec{y}_t(l) = \arg\max_{b \in \{0,1\}} P(H_b^l | E^l_{\vec{C}_t(l)}), \ l \in \mathcal{Y} \tag{7}$$

Using the Bayesian rule, Eq.(7) can be rewritten as:

$$\begin{aligned} \vec{y}_t(l) &= \arg\max_{b \in \{0,1\}} \frac{P(H_b^l) P(E^l_{\vec{C}_t(l)} | H_b^l)}{P(E^l_{\vec{C}_t(l)})} \\ &= \arg\max_{b \in \{0,1\}} P(H_b^l) P(E^l_{\vec{C}_t(l)} | H_b^l) \end{aligned} \tag{8}$$

[1] In this paper, only the former formalism of multi-label learning is studied.

```
$[\vec{y}_t, \vec{r}_t]$=ML-KNN($T$, K, $t$, $s$)

%Computing the prior probabilities $P(H_b^l)$
(1)  for $l \in \mathcal{Y}$ do
(2)      $P(H_1^l) = (s + \sum_{i=1}^{m} \vec{y}_{x_i}(l)) / (s \times 2 + m)$ ;   $P(H_0^l) = 1 - P(H_1^l)$;
%Computing the posterior probabilities $P(E_j^l|H_b^l)$
(3)  Identify $N(x_i),\ i \in \{1, 2, \ldots, m\}$;
(4)  for $l \in \mathcal{Y}$ do
(5)      for $j \in \{0, 1, \ldots, \text{K}\}$ do
(6)          $c[j] = 0;\quad c'[j] = 0;$
(7)      for $i \in \{1, 2, \ldots, m\}$ do
(8)          $\delta = \vec{C}_{x_i}(l) = \sum_{a \in N(x_i)} \vec{y}_a(l);$
(9)          if $(\vec{y}_{x_i}(l) == 1)$ then $c[\delta] = c[\delta] + 1;$
(10)                                   else $c'[\delta] = c'[\delta] + 1;$
(11)     for $j \in \{0, 1, \ldots, \text{K}\}$ do
(12)         $P(E_j^l|H_1^l) = (s + c[j])/(s \times (\text{K} + 1) + \sum_{p=0}^{\text{k}} c[p])$;
(13)         $P(E_j^l|H_0^l) = (s + c'[j])/(s \times (\text{K} + 1) + \sum_{p=0}^{\text{k}} c'[p])$;
%Computing $\vec{y}_t$ and $\vec{r}_t$
(14) Identify $N(t)$;
(15) for $l \in \mathcal{Y}$ do
(16)     $\vec{C}_t(l) = \sum_{a \in N(t)} \vec{y}_a(l);$
(17)     $\vec{y}_t(l) = \arg\max_{b \in \{0,1\}} P(H_b^l) P(E_{\vec{C}_t(l)}^l|H_b^l);$
(18)     $\vec{r}_t(l) = P(H_1^l|E_{\vec{C}_t(l)}^l) = (P(H_1^l) P(E_{\vec{C}_t(l)}^l|H_1^l))/P(E_{\vec{C}_t(l)}^l)$
              $= (P(H_1^l) P(E_{\vec{C}_t(l)}^l|H_1^l))/(\sum_{b \in \{0,1\}} P(H_b^l) P(E_{\vec{C}_t(l)}^l|H_b^l));$
```

Fig. 1. Pseudo code of ML-KNN.

As shown in Eq.(8), in order to determine the category vector $\vec{y}_t$, all the information needed is the prior probabilities $P(H_b^l)$ ($l \in \mathcal{Y}$, $b \in \{0, 1\}$) and the posterior probabilities $P(E_j^l|H_b^l)$ ($j \in \{0, 1, \ldots, \text{K}\}$). Actually, these prior and posterior probabilities can all be directly estimated from the training set based on frequency counting.

Fig. 1 gives the complete description of ML-KNN. T is the training set as shown in Section 2 and the meanings of the input arguments K, t and the output argument $\vec{y}_t$ are the same as described previously. Furthermore, the input argument s is a smoothing parameter controlling the strength of uniform prior (In this paper, s is set to be 1 which yields the Laplace smoothing). $\vec{r}_t$ is a real-valued vector calculated to rank labels in $\mathcal{Y}$, where $\vec{r}_t(l)$ corresponds to the posterior probability $P(H_1^l|E^l_{\vec{C}_t(l)})$. As shown in Fig. 1, based on the multi-label training instances, steps (1) and (2) estimate the prior probabilities $P(H_b^l)$. Steps from (3) to (13) estimate the posterior probabilities $P(E_j^l|H_b^l)$, where $c[j]$ used in each iteration of l counts the number of training instances with label l whose k nearest neighbors contain exactly j instances with label l. Correspondingly, $c'[j]$ counts the number of training instances without label l whose k nearest neighbors contain exactly j instances with label l. Finally, using the Bayesian rule, steps from (14) to (18) compute the algorithm's outputs based on the estimated probabilities.

5 Experiments

As reviewed in Section 3, there have been several approaches to solving multi-label problems. In this paper, ML-KNN is compared with the boosting-style algorithm BOOSTEXTER [19] [2], multi-label decision tree ADTBOOST.MH [4] [3], and the multi-label kernel method RANK-SVM [7], which are all general-purpose multi-label learning algorithms applicable to various multi-label problems.

For ML-KNN, Euclidean metric is used to measure distances between instances. For BOOSTEXTER and ADTBOOST.MH, the number of boosting rounds is set to be 500 and 50 respectively because on all data sets studied in this paper, the performance of these two algorithms will not significantly change after the specified boosting rounds; For RANK-SVM, polynomial kernels with degree 8 are used which yield the best performance as shown in the literature [7].

In this paper, comparative studies of those algorithms are performed on one bioinformatic data [7], one natural scene classification data, and one automatic web page categorization data [21].

[2] Program available at http://www.cs.princeton.edu/~schapire/boostexter.html.

[3] The algorithm and a graphical user interface are available at http://www.grappa.univ-lille3.fr/grappa/index.php3?info=logiciels.

5.1 Yeast Gene Functional Analysis

In this paper, the effectiveness of multi-label learning algorithms is firstly evaluated through predicting the gene functional classes of the Yeast *Saccharomyces cerevisiae*, which is one of the best studied organisms. Specifically, the Yeast data set studied in the literatures [7] and [15] is investigated. Each gene is described by the concatenation of micro-array expression data and phylogenetic profile and is associated with a set of functional classes whose maximum size can be potentially more than 190. In order to make it easier, Elisseeff and Weston [7] preprocessed the data set where only the known structure of the functional classes are used. Actually, the whole set of functional classes is structured into hierarchies up to 4 levels deep [4]. In this paper, as what has been done in the literature [7], only functional classes in the top hierarchy are considered. The first level of the hierarchy is depicted in Fig. 2. The resulting multi-label data set contains 2 417 genes each represented by a 103-dimensional feature vector. There are 14 possible class labels and the average number of labels for each gene is 4.24 ± 1.57.

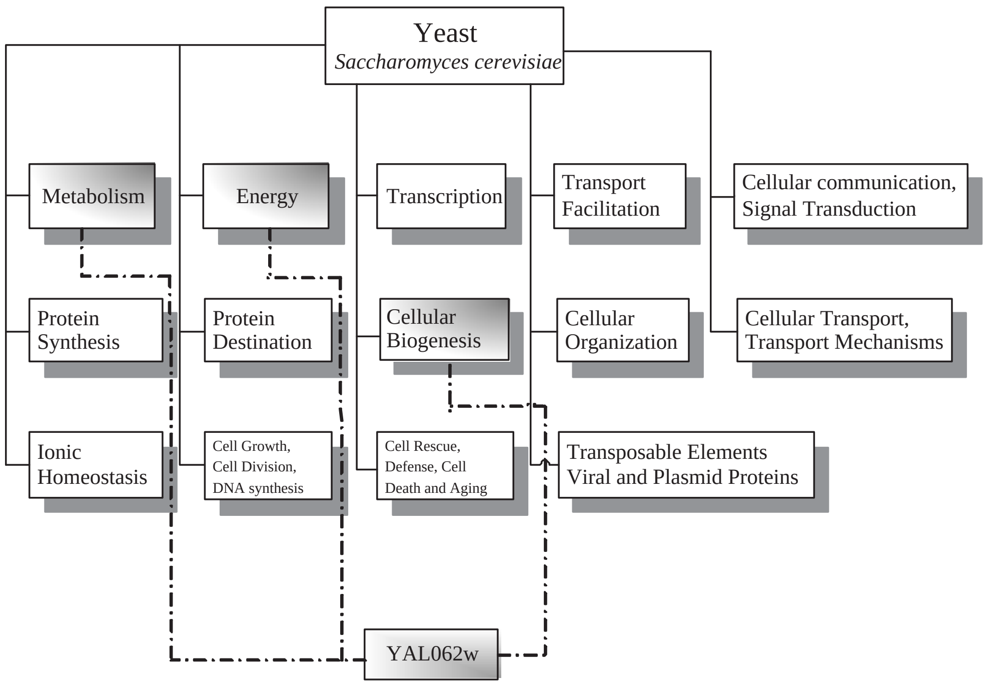
Yeast
Saccharomyces cerevisiae
Metabolism
Energy
Transcription
Transport Facilitation
Cellular communication, Signal Transduction
Protein Synthesis
Protein Destination
Cellular Biogenesis
Cellular Organization
Cellular Transport, Transport Mechanisms
Ionic Homeostasis
Cell Growth, Cell Division, DNA synthesis
Cell Rescue, Defense, Cell Death and Aging
Transposable Elements Viral and Plasmid Proteins
YAL062w

Fig. 2. First level of the hierarchy of the Yeast gene functional classes. One gene, for instance the one named YAL062w, can belong to several classes (shaded in grey) of the 14 possible classes.

Ten-fold cross-validation is performed on this data set. The experimental re-

[4] See http://mips.gsf.de/proj/yeast/catalogues/funcat/ for more details.

Table 1
Experimental results of Ml-knn (mean±std) on the Yeast data with different number of nearest neighbors considered.

Evaluation	Number of Nearest Neighbors Considered				
Criterion	K=8	K=9	K=10	K=11	K=12
Hamming Loss	0.195±0.010	0.193±0.009	0.194±0.010	0.193±0.008	**0.192±0.010**
One-error	0.233±0.032	0.230±0.041	0.230±0.030	**0.225±0.036**	0.232±0.032
Coverage	6.291±0.238	6.297±0.223	**6.275±0.240**	6.285±0.208	6.283±0.228
Ranking Loss	0.169±0.016	0.168±0.016	**0.167±0.016**	**0.167±0.015**	**0.167±0.015**
Average Precision	0.763±0.021	0.764±0.022	0.765±0.021	**0.766±0.021**	0.765±0.020

Table 2
Experimental results of each multi-label learning algorithm (mean±std) on the Yeast data.

Evaluation	Algorithm			
Criterion	Ml-knn	BoosTexter	Adtboost.MH	Rank-svm
Hamming Loss	**0.194±0.010**	0.220±0.011	0.207±0.010	0.207±0.013
One-error	**0.230±0.030**	0.278±0.034	0.244±0.035	0.243±0.039
Coverage	**6.275±0.240**	6.550±0.243	6.390±0.203	7.090±0.503
Ranking Loss	**0.167±0.016**	0.186±0.015	N/A	0.195±0.021
Average Precision	**0.765±0.021**	0.737±0.022	0.744±0.025	0.749±0.026

sults of Ml-knn are reported in Table 1, where the number of nearest neighbors considered by Ml-knn (i.e. the parameter K as shown in Fig. 1) varies from 8 to 12. The value following "±" gives the standard deviation and the best result on each metric is shown in bold face. Table 1 shows that the number of nearest neighbors used by Ml-knn does not significantly affect the performance of the algorithm. Therefore, all the results of Ml-knn shown in the rest of this paper are obtained with the parameter K set to be the moderate value of 10.

Table 2 reports the experimental results of Ml-knn and other multi-label learning algorithms on the Yeast data, where the best result on each metric is shown in bold face. To make a clearer view of the relative performance between each algorithm, a partial order "$\succ$" is defined on the set of all comparing algorithms for each evaluation criterion, where $A1 \succ A2$ means that the performance of algorithm $A1$ is statistically better than that of algorithm $A2$ on the specific metric (based on two-tailed paired t-test at 5% significance level). The partial order on all the comparing algorithms in terms of each evaluation criterion is summarized in Table 3.

Note that the partial order "$\succ$" only measures the relative performance between two algorithms $A1$ and $A2$ on one specific evaluation criterion. However, it is quite possible that $A1$ performs better than $A2$ in terms of some metrics but worse that $A2$ in terms of other ones. In this case, it is hard to judge

Table 3
Relative performance between each multi-label learning algorithm on the Yeast data.

Evaluation Criterion	Algorithm: $A1$-Ml-knn; $A2$-BoosTexter; $A3$-Adtboost.MH; $A4$-Rank-svm
Hamming Loss	$A1 \succ A2$, $A1 \succ A3$, $A1 \succ A4$, $A3 \succ A2$, $A4 \succ A2$
One-error	$A1 \succ A2$, $A3 \succ A2$, $A4 \succ A2$
Coverage	$A1 \succ A2$, $A1 \succ A3$, $A1 \succ A4$, $A2 \succ A4$, $A3 \succ A2$, $A3 \succ A4$
Ranking Loss	$A1 \succ A2$, $A1 \succ A4$
Average Precision	$A1 \succ A2$, $A1 \succ A3$
Total Order	Ml-knn(11)>Adtboost.MH(1)>Rank-svm(-3)>BoosTexter(-9)

which algorithm is superior. Therefore, in order to give an overall performance assessment of an algorithm, a score is assigned to it which takes account of its relative performance with other algorithms on all metrics. Concretely, for each evaluation criterion, for each possible pair of algorithms $A1$ and $A2$, if $A1 \succ A2$ holds, then $A1$ is rewarded by a positive score +1 and $A2$ is penalized by a negative score -1. Based on the accumulated score of each algorithm on all evaluation criteria, a total order ">" is defined on the set of all comparing algorithms as shown in the last line of Table 3, where $A1 > A2$ means that $A1$ performs better than $A2$ on the Yeast data. The accumulated score of each algorithm is also shown in the parentheses.

Table 3 shows that Ml-knn performs fairly well in terms of all the evaluation criteria, where on all these metrics no algorithm has outperformed Ml-knn. Especially, Ml-knn outperforms all the other algorithms with respect to *hamming loss*, *coverage* and *ranking loss* [5]. It is also worth noting that BoosTexter performs quite poorly compared to other algorithms. As indicated in the literature [7], the reason may be that the simple decision function realized by this method is not suitable to learn from the Yeast data set. On the whole (as shown by the total order), Ml-knn substantially outperforms all the other algorithms on the Yeast data.

5.2 Natural Scene Classification

In natural scene classification, each natural scene image may belong to several image types (classes) simultaneously, e.g. the image shown in Fig. 3(a) can be classified as a mountain scene as well as a tree scene, while the image shown in Fig. 3(b) can be classified as a sea scene as well as a sunset scene. Through analyzing images with known label sets, a multi-label learning system will automatically predict the sets of labels for unseen images. The above process of semantic scene classification can be applied to many areas, such as content-

[5] Note that *ranking loss* is not provided by the Adtboost.MH program.

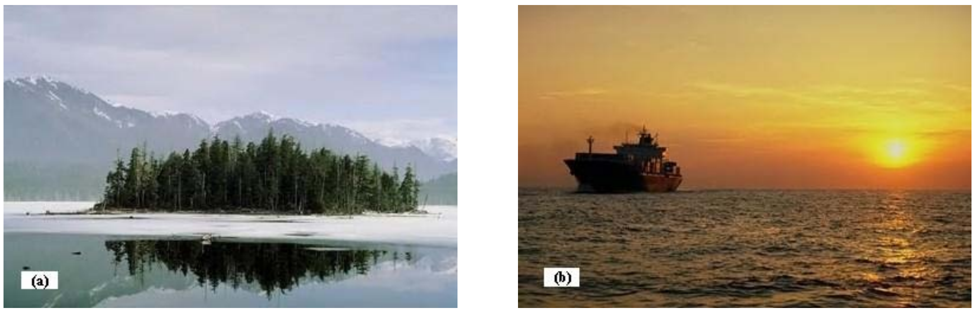

Fig. 3. Examples of multi-labelled images.

Table 4
Summary of the natural scene image data set.

Label Set	#Images	Label Set	#Images	Label Set	#Images
desert	340	desert+sunset	21	sunset+trees	28
mountains	268	desert+trees	20	desert+mountains+sunset	1
sea	341	mountains+sea	38	desert+sunset+trees	3
sunset	216	mountains+sunset	19	mountains+sea+trees	6
trees	378	mountains+trees	106	mountains+sunset+trees	1
desert+mountains	19	sea+sunset	172	sea+sunset+trees	4
desert+sea	5	sea+trees	14	*Total*	2 000

based indexing and organization and content-sensitive image enhancement, etc [2]. In this paper, the effectiveness of multi-label learning algorithms is also evaluated via this specific kind of multi-label learning problem.

The experimental data set consists of 2 000 natural scene images, where a set of labels is manually assigned to each image. Table 4 gives the detailed description of the number of images associated with different label sets, where all the possible class labels are *desert*, *mountains*, *sea*, *sunset* and *trees*. The number of images belonging to more than one class (e.g. sea+sunset) comprises over 22% of the data set, many combined classes (e.g. mountains+sunset+trees) are extremely rare. On average, each image is associated with 1.24 class labels. In this paper, each image is represented by a feature vector using the same method employed in the literature [2]. Concretely, each color image is firstly converted to the CIE Luv space, which is a more perceptually uniform color space such that perceived color differences correspond closely to Euclidean distances in this color space. After that, the image is divided into 49 blocks using a 7×7 grid, where in each block the first and second moments (mean and variance) of each band are computed, corresponding to a low-resolution image and to computationally inexpensive texture features respectively. Finally, each image is transformed into a $49 \times 3 \times 2 = 294$-dimensional feature vector.

Ten-fold cross-validation is also performed on this image data set. Experimen-

Table 5
Experimental results of each multi-label learning algorithm (mean±std) on the natural scene image data set.

Evaluation	Algorithm			
Criterion	ML-KNN	BOOSTEXTER	ADTBOOST.MH	RANK-SVM
Hamming Loss	**0.169±0.016**	0.179±0.015	0.193±0.014	0.253±0.055
One-error	**0.300±0.046**	0.311±0.041	0.375±0.049	0.491±0.135
Coverage	**0.939±0.100**	**0.939±0.092**	1.102±0.111	1.382±0.381
Ranking Loss	**0.168±0.024**	**0.168±0.020**	N/A	0.278±0.096
Average Precision	**0.803±0.027**	0.798±0.024	0.755±0.027	0.682±0.093

Table 6
Relative performance between each multi-label learning algorithm on the natural scene image data set.

Evaluation	Algorithm
Criterion	$A1$-ML-KNN; $A2$-BOOSTEXTER; $A3$-ADTBOOST.MH; $A4$-RANK-SVM
Hamming Loss	$A1 \succ A2$, $A1 \succ A3$, $A1 \succ A4$, $A2 \succ A3$, $A2 \succ A4$, $A3 \succ A4$
One-error	$A1 \succ A3$, $A1 \succ A4$, $A2 \succ A3$, $A2 \succ A4$, $A3 \succ A4$
Coverage	$A1 \succ A3$, $A1 \succ A4$, $A2 \succ A3$, $A2 \succ A4$, $A3 \succ A4$
Ranking Loss	$A1 \succ A4$, $A2 \succ A4$
Average Precision	$A1 \succ A3$, $A1 \succ A4$, $A2 \succ A3$, $A2 \succ A4$, $A3 \succ A4$
Total Order	ML-KNN(10)>BOOSTEXTER(8)>ADTBOOST.MH(-4)>RANK-SVM(-14)

tal results of ML-KNN and other multi-label learning algorithms are reported in Table 5, where the best result on each evaluation criterion is shown in bold face. Similarly as the Yeast data, the partial order "$\succ$" and the total order "$>$" are also defined on the set of all comparing algorithms which are shown in Table 6.

As shown in Table 6, it is obvious that both ML-KNN and BOOSTEXTER are superior to ADTBOOST.MH and RANK-SVM in terms of all evaluation criteria. Furthermore, ADTBOOST.MH outperforms RANK-SVM on all evaluation metrics and ML-KNN outperforms all the other algorithms in terms of *hamming loss*. On the whole (as shown by the total order), ML-KNN slightly outperforms BOOSTEXTER and is far superior to ADTBOOST.MH and RANK-SVM on the natural scene image data set. Fig. 4 shows some example images on which ML-KNN works better than BOOSTEXTER, ADTBOOST.MH and RANK-SVM, where *null* means that the predicted label set is empty.

Note that in this data set, the average number of class labels associated with each image is relatively small (e.g. 1.24). Therefore, to further evaluate the performance of the multi-label learning algorithms on the problem of natural scene classification, images with only one class label are excluded from the

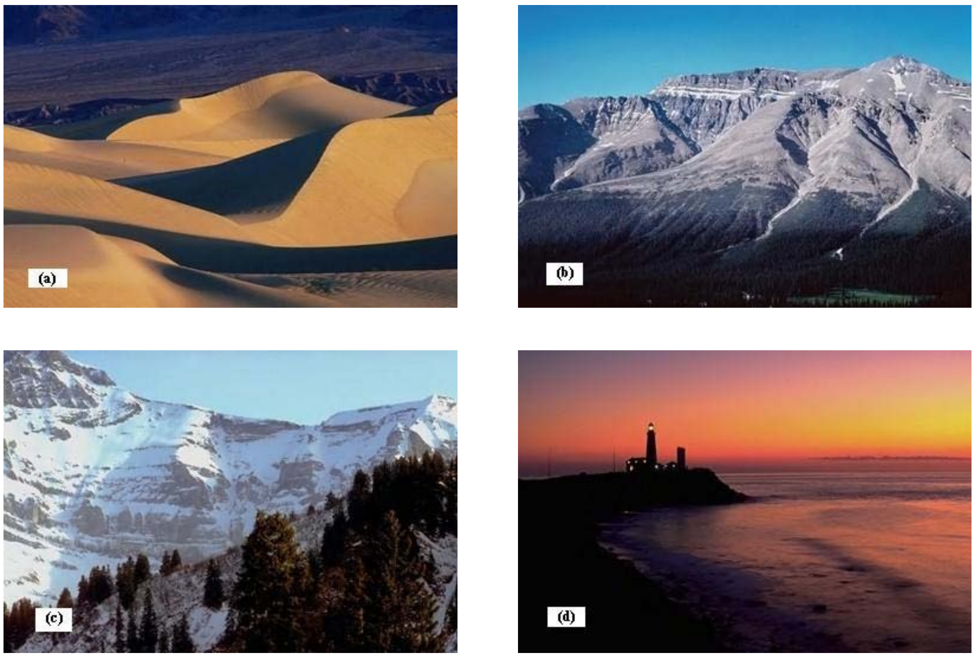

Fig. 4. Some example images on which Ml-knn works better than BoosTexter, Adtboost.MH and Rank-svm. (a) Ground-truth: *desert*, Ml-knn: *desert*, BoosTexter: *desert+trees*, Adtboost.MH: *null*, Rank-svm: *mountains*; (b) Ground-truth: *mountains*, Ml-knn: *mountains*, BoosTexter: *mountains+trees*, Adtboost.MH: *trees*, Rank-svm: *trees*; (c) Ground-truth: *mountains+trees*, Ml-knn: *mountains+trees*, BoosTexter: *null*, Adtboost.MH: *mountains*, Rank-svm: *mountains*; (d) Ground-truth: *sea+sunset*, Ml-knn: *sea+sunset*, BoosTexter: *sunset*, Adtboost.MH: *null*, Rank-svm: *sunset*.

original data set. Thus, a *filtered* data set containing 457 images is obtained, in which each image is associated with 2.03 class labels on average. Ten-fold cross-validation is again performed on the filtered image data set, where experimental results of the multi-label learning algorithms are reported in Table 7 with the best result on each evaluation criterion shown in bold face. Similarly as the Yeast data, the partial order "$\succ$" and the total order "$>$" are also defined on the set of all comparing algorithms which are shown in Table 8.

Table 8 shows that both Ml-knn and BoosTexter are superior or at least comparable to Adtboost.MH and Rank-svm in terms of all evaluation criteria. Furthermore, Ml-knn outperforms Adtboost.MH on all evaluation metrics while BoosTexter outperforms Adtboost.MH and Ml-knn outperforms Rank-svm on all evaluation metrics except *one-error*. On the whole (as shown by the total order), the same as the original (unfiltered) data set, Ml-knn again slightly outperforms BoosTexter and is far superior to Adtboost.MH and Rank-svm on the filtered natural scene image data set. These results show that Ml-knn can also work well on the problem of natural scene classification when more class labels are associated with each image.

Table 7
Experimental results of each multi-label learning algorithm (mean±std) on the *filtered* natural scene image data set.

Evaluation	Algorithm			
Criterion	Ml-knn	BoosTexter	Adtboost.MH	Rank-svm
Hamming Loss	**0.235±0.019**	0.240±0.043	0.270±0.027	0.275±0.035
One-error	**0.205±0.051**	0.211±0.058	0.250±0.070	0.236±0.064
Coverage	**1.875±0.072**	1.921±0.140	2.080±0.151	2.054±0.162
Ranking Loss	**0.195±0.021**	0.204±0.034	N/A	0.228±0.040
Average Precision	**0.832±0.018**	0.828±0.029	0.799±0.034	0.805±0.032

Table 8
Relative performance between each multi-label learning algorithm on the *filtered* natural scene image data set.

Evaluation	Algorithm
Criterion	$A1$-Ml-knn; $A2$-BoosTexter; $A3$-Adtboost.MH; $A4$-Rank-svm
Hamming Loss	$A1 \succ A3$, $A1 \succ A4$, $A2 \succ A3$, $A2 \succ A4$
One-error	$A1 \succ A3$
Coverage	$A1 \succ A3$, $A1 \succ A4$, $A2 \succ A3$
Ranking Loss	$A1 \succ A4$
Average Precision	$A1 \succ A3$, $A1 \succ A4$, $A2 \succ A3$, $A2 \succ A4$
Total Order	Ml-knn(8)>BoosTexter(5)>Rank-svm(-6)>Adtboost.MH(-7)

5.3 Automatic Web Page Categorization

Recently, Ueda and Saito [21] presented two types of probabilistic generative models called parametric mixture models (Pmm1, Pmm2) for multi-label text. They also designed efficient learning and prediction algorithms for Pmms and tested the effectiveness of their method with application to the specific text categorization problem of WWW page categorization [6]. Specifically, they tried to categorize real Web pages linked from the “yahoo.com” domain, where it consists of 14 top-level categories (i.e. “Arts&Humanities”, “Business&Economy”, etc.) and each category is classified into a number of second-level subcategories. By focusing on the second-level categories, they used 11 out of the 14 *independent* text categorization problems. For each problem, the training set contains 2 000 documents while the test set contains 3 000 documents.

In this paper, these data sets are used to further evaluate the performance of each multi-label learning algorithm. The simple term selection method based

[6] Data set available at http://www.kecl.ntt.co.jp/as/members/ueda/yahoo.tar.gz.

Table 9
Characteristics of the web page data sets (after term selection). *PMC* denotes the percentage of documents belonging to more than one category, *ANL* denotes the average number of labels for each document, and *PRC* denotes the percentage of *rare* categories, i.e. the kind of category where only less than 1% instances in the data set belong to it.

	Number of	Vocabulary	Training Set			Test Set		
Data Set	Categories	Size	*PMC*	*ANL*	*PRC*	*PMC*	*ANL*	*PRC*
Arts&Humanities	26	462	44.50%	1.627	19.23%	43.63%	1.642	19.23%
Business&Economy	30	438	42.20%	1.590	50.00%	41.93%	1.586	43.33%
Computers&Internet	33	681	29.60%	1.487	39.39%	31.27%	1.522	36.36%
Education	33	550	33.50%	1.465	57.58%	33.73%	1.458	57.58%
Entertainment	21	640	29.30%	1.426	28.57%	28.20%	1.417	33.33%
Health	32	612	48.05%	1.667	53.13%	47.20%	1.659	53.13%
Recreation&Sports	22	606	30.20%	1.414	18.18%	31.20%	1.429	18.18%
Reference	33	793	13.75%	1.159	51.52%	14.60%	1.177	54.55%
Science	40	743	34.85%	1.489	35.00%	30.57%	1.425	40.00%
Social&Science	39	1 047	20.95%	1.274	56.41%	22.83%	1.290	58.97%
Society&Culture	27	636	41.90%	1.705	25.93%	39.97%	1.684	22.22%

on *document frequency* (the number of documents containing a specific term) is used to reduce the dimensionality of each data set. Actually, only 2% words with highest document frequency are retained in the final vocabulary[7]. Note that other term selection methods such as *information gain* and *mutual information* could also be adopted. After term selection, each document in the data set is described as a feature vector using the "*Bag-of-Words*" representation [6], i.e. each dimension of the feature vector corresponds to the number of times a word in the vocabulary appearing in this document. Table 9 summarizes the characteristics of the web page data sets. It is worth noting that, compared with the Yeast data and the natural scene image data, instances are represented by much higher dimensional feature vectors and a large portion of them (about 20% $\sim$ 45%) are multi-labelled over the 11 problems. Furthermore, in those 11 data sets, the number of categories are much larger (minimum 21, maximum 40) and many of them are *rare* categories (about 20% $\sim$ 55%). Therefore, the web page data sets are more difficult to learn from than the previous data collections.

The experimental results on each evaluation criterion are reported in Tables 10 to 14, where the best result on each data set is shown in bold face. Similarly as the Yeast data, the partial order "$\succ$" and total order "$>$" are also defined

[7] Based on a series experiments, Yang and Pedersen [22] have shown that based on document frequency, it is possible to reduce the dimensionality by a factor of 10 with no loss in effectiveness and by a factor of 100 with just a small loss.

Table 10
Experimental results of each multi-label learning algorithm on the web page data sets in terms of *hamming loss*.

	Algorithm			
Data Set	Ml-knn	BoosTexter	Adtboost.MH	Rank-svm
Arts&Humanities	0.0612	0.0652	**0.0585**	0.0615
Business&Economy	**0.0269**	0.0293	0.0279	0.0275
Computers&Internet	0.0412	0.0408	0.0396	**0.0392**
Education	**0.0387**	0.0457	0.0423	0.0398
Entertainment	0.0604	0.0626	**0.0578**	0.0630
Health	0.0458	**0.0397**	**0.0397**	0.0423
Recreation&Sports	0.0620	0.0657	**0.0584**	0.0605
Reference	0.0314	0.0304	**0.0293**	0.0300
Science	**0.0325**	0.0379	0.0344	0.0340
Social&Science	**0.0218**	0.0243	0.0234	0.0242
Society&Culture	**0.0537**	0.0628	0.0575	0.0555
Average	0.0432	0.0459	**0.0426**	0.0434

Table 11
Experimental results of each multi-label learning algorithm on the web page data sets in terms of *one-error*.

	Algorithm			
Data Set	Ml-knn	BoosTexter	Adtboost.MH	Rank-svm
Arts&Humanities	0.6330	**0.5550**	0.5617	0.6653
Business&Economy	**0.1213**	0.1307	0.1337	0.1237
Computers&Internet	0.4357	0.4287	0.4613	**0.4037**
Education	0.5207	0.5587	0.5753	**0.4937**
Entertainment	0.5300	**0.4750**	0.4940	0.4933
Health	0.4190	**0.3210**	0.3470	0.3323
Recreation&Sports	0.7057	0.5557	**0.5547**	0.5627
Reference	0.4730	0.4427	0.4840	**0.4323**
Science	0.5810	0.6100	0.6170	**0.5523**
Social&Science	**0.3270**	0.3437	0.3600	0.3550
Society&Culture	0.4357	0.4877	0.4845	**0.4270**
Average	0.4711	0.4463	0.4612	**0.4401**

on the set of all comparing algorithms which are shown in Table 15.

As shown in Table 15, Ml-knn achieves comparable results in terms of all the evaluation criteria, where on all these metrics no algorithm has outperformed Ml-knn. On the other hand, although BoosTexter performs quite well in terms of *one-error*, *coverage*, *ranking loss* and *average precision*, it performs almost worst among all the comparing algorithms in terms of *hamming loss*.

Table 12
Experimental results of each multi-label learning algorithm on the web page data sets in terms of *coverage*.

	Algorithm			
Data Set	Ml-knn	BoosTexter	Adtboost.MH	Rank-svm
Arts&Humanities	5.4313	5.2973	**5.1900**	9.2723
Business&Economy	**2.1840**	2.4123	2.4730	3.3637
Computers&Internet	**4.4117**	4.4887	4.4747	8.7910
Education	**3.4973**	4.0673	3.9663	8.9560
Entertainment	3.1467	3.0883	**3.0877**	6.5210
Health	3.3043	**3.0780**	3.0843	5.5400
Recreation&Sports	5.1010	4.4737	**4.3380**	5.6680
Reference	3.5420	**3.2100**	3.2643	6.9683
Science	**6.0470**	6.6907	6.6027	12.4010
Social&Science	**3.0340**	3.6870	3.4820	8.2177
Society&Culture	5.3653	5.8463	**4.9545**	6.8837
Average	4.0968	4.2127	**4.0834**	7.5075

Table 13
Experimental results of each multi-label learning algorithm on the web page data sets in terms of *ranking loss*.

	Algorithm			
Data Set	Ml-knn	BoosTexter	Adtboost.MH	Rank-svm
Arts&Humanities	0.1514	**0.1458**	N/A	0.2826
Business&Economy	**0.0373**	0.0416	N/A	0.0662
Computers&Internet	**0.0921**	0.0950	N/A	0.2091
Education	**0.0800**	0.0938	N/A	0.2080
Entertainment	0.1151	**0.1132**	N/A	0.2617
Health	0.0605	**0.0521**	N/A	0.1096
Recreation&Sports	0.1913	**0.1599**	N/A	0.2094
Reference	0.0919	**0.0811**	N/A	0.1818
Science	**0.1167**	0.1312	N/A	0.2570
Social&Science	**0.0561**	0.0684	N/A	0.1661
Society&Culture	**0.1338**	0.1483	N/A	0.1716
Average	**0.1024**	0.1028	N/A	0.1930

It is also worth noting that all the algorithms perform quite poorly in terms of *one-error* (around 45% for all comparing algorithms). The reason may be that there are much more categories in those 11 data sets which makes the top-ranked label be in the set of proper labels of an instance much more difficult. On the whole (as shown by the total order), Ml-knn is comparable to BoosTexter and both of them slightly outperform Adtboost.MH and are far superior to Rank-svm on the web page data sets.

Table 14
Experimental results of each multi-label learning algorithm on the web page data sets in terms of *average precision*.

	Algorithm			
Data Set	Ml-knn	BoosTexter	Adtboost.MH	Rank-svm
Arts&Humanities	0.5097	0.5448	**0.5526**	0.4170
Business&Economy	**0.8798**	0.8697	0.8702	0.8694
Computers&Internet	0.6338	**0.6449**	0.6235	0.6123
Education	**0.5993**	0.5654	0.5619	0.5702
Entertainment	0.6013	**0.6368**	0.6221	0.5637
Health	0.6817	**0.7408**	0.7257	0.6839
Recreation&Sports	0.4552	0.5572	**0.5639**	0.5315
Reference	0.6194	**0.6578**	0.6264	0.6176
Science	**0.5324**	0.5006	0.4940	0.5007
Social&Science	**0.7481**	0.7262	0.7217	0.6788
Society&Culture	**0.6128**	0.5717	0.5881	0.5717
Average	0.6249	**0.6378**	0.6318	0.6046

Table 15
Relative performance between each multi-label learning algorithm on the web page data sets.

Evaluation	Algorithm
Criterion	$A1$-Ml-knn; $A2$-BoosTexter; $A3$-Adtboost.MH; $A4$-Rank-svm
Hamming Loss	$A3 \succ A2$, $A4 \succ A2$
One-error	$A2 \succ A3$
Coverage	$A1 \succ A4$, $A2 \succ A4$, $A3 \succ A4$
Ranking Loss	$A1 \succ A4$, $A2 \succ A4$
Average Precision	$A2 \succ A4$
Total Order	{Ml-knn(2), BoosTexter(2)}>Adtboost.MH(1)>Rank-svm(-5)

6 Conclusion

In this paper, a lazy learning algorithm named Ml-knn, which is the multi-label version of kNN, is proposed. Based on statistical information derived from the label sets of an unseen instance's neighboring instances, i.e. the *membership counting* statistic as shown in Section 4, Ml-knn utilizes maximum a posteriori principle to determine the label set for the unseen instance. Experiments on three real-world multi-label learning problems, i.e. Yeast gene functional analysis, natural scene classification and automatic web page categorization, show that Ml-knn outperforms some well-established multi-label learning algorithms.

In this paper, the distance between instances is simply measured by Euclidean metric. Therefore, it is interesting to see whether other kinds of distance metrics could further improve the performance of Ml-knn. On the other hand, investigating more complex statistical information other than the *membership counting* statistic to facilitate the usage of maximum a posteriori principle is another interesting issue for future work.

Acknowledgment

The authors wish to thank the anonymous reviewers for their constructive suggestions. Many thanks to A. Elisseeff and J. Weston for providing the authors with the Yeast data and the implementation details of Rank-svm. This work was supported by the National Science Foundation of China under Grant Nos. 60473046 and 60635030, and the Jiangsu Science Foundation under Grant No. BK2005412.

References

[1] D. W. Aha, Lazy learning: Special issue editorial, Artificial Intelligence Review 11 (1-5) (1997) 7–10.

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

[3] A. Clare, R. D. King, Knowledge discovery in multi-label phenotype data, in: L. D. Raedt, A. Siebes (Eds.), Lecture Notes in Computer Science 2168, Springer, Berlin, 2001, pp. 42–53.

[4] F. D. Comité, R. Gilleron, M. Tommasi, Learning multi-label altenating decision tree from texts and data, in: P. Perner, A. Rosenfeld (Eds.), Lecture Notes in Computer Science 2734, Springer, Berlin, 2003, pp. 35–49.

[5] A. P. Dempster, N. M. Laird, D. B. Rubin, Maximum likelihood from incomplete data via the EM algorithm, Journal of the Royal Statistics Society -B 39 (1) (1977) 1–38.

[6] S. T. Dumais, J. Platt, D. Heckerman, M. Sahami, Inductive learning algorithms and representation for text categorization, in: Proceedings of the 7th ACM International Conference on Information and Knowledge Management, Bethesda, MD, 1998, pp. 148–155.

[7] A. Elisseeff, J. Weston, A kernel method for multi-labelled classification, in: T. G. Dietterich, S. Becker, Z. Ghahramani (Eds.), Advances in Neural Information Processing Systems 14, MIT Press, Cambridge, MA, 2002, pp. 681–687.

[8] Y. Freund, L. Mason, The alternating decision tree learning algorithm, in: Proceedings of the 16th International Conference on Machine Learning, Bled, Slovenia, 1999, pp. 124–133.

[9] Y. Freund, R. E. Schapire, A decision-theoretic generalization of on-line learning and an application to boosting, Journal of Computer and System Sciences 55 (1) (1997) 119–139.

[10] S. Gao, W. Wu, C.-H. Lee, T.-S. Chua, A maximal figure-of-merit learning approach to text categorization, in: Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Toronto, Canada, 2003, pp. 174–181.

[11] S. Gao, W. Wu, C.-H. Lee, T.-S. Chua, A MFoM learning approach to robust multiclass multi-label text categorization, in: Proceedings of the 21st International Conference on Machine Learning, Banff, Canada, 2004, pp. 329–336.

[12] R. Jin, Z. Ghahramani, Learning with multiple labels, in: S. Becker, S. Thrun, K. Obermayer (Eds.), Advances in Neural Information Processing Systems 15, MIT Press, Cambridge, MA, 2003, pp. 897–904.

[13] H. Kazawa, T. Izumitani, H. Taira, E. Maeda, Maximal margin labeling for multi-topic text categorization, in: L. K. Saul, Y. Weiss, L. Bottou (Eds.), Advances in Neural Information Processing Systems 17, MIT Press, Cambridge, MA, 2005, pp. 649–656.

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

[15] P. Pavlidis, J. Weston, J. Cai, W. N. Grundy, Combining microarray expression data and phylogenetic profiles to learn functional categories using support vector machines, in: Proceedings of the 5th Annual International Conference on Computational Biology, Montréal, Canada, 2001, pp. 242–248.

[16] J. R. Quinlan, C4.5: Programs for Machine Learning, Morgan Kaufmann, San Mateo, California, 1993.

[17] G. Salton, Developments in automatic text retrieval, Science 253 (1991) 974–980.

[18] R. E. Schapire, Y. Singer, Improved boosting algorithms using confidence-rated predictions, in: Proceedings of the 11th Annual Conference on Computational Learning Theory, New York, 1998, pp. 80–91.

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

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

[21] N. Ueda, K. Saito, Parametric mixture models for multi-label text, in: S. Becker, S. Thrun, K. Obermayer (Eds.), Advances in Neural Information Processing Systems 15, MIT Press, Cambridge, MA, 2003, pp. 721–728.

[22] Y. Yang, J. O. Pedersen, A comparative study on feature selection in text categorization, in: Proceedings of the 14th International Conference on Machine Learning, Nashville, TN, 1997, pp. 412–420.

[23] M.-L. Zhang, Z.-H. Zhou, Multilabel neural networks with applications to functional genomics and text categorization, IEEE Transactions on Knowledge and Data Engineering 18 (10) (2006) 1338–1351.