A Locally Adaptive Multi-Label k-Nearest Neighbor Algorithm

Dengbao Wang[1], Jingyuan Wang[1], Fei Hu[1], Li Li[1], and Xiuzhen Zhang[2]

[1] College of Computer and Information Science, Southwest University, China
[2] School of Computer Science and Information Technology, RMIT University, Australia

Abstract. In the field of multi-label learning, ML-kNN is the first lazy learning approach and one of the most influential approaches. The main idea of it is to adapt k-NN method to deal with multi-label data, where maximum a posteriori rule is utilized to adaptively adjust decision boundary for each unseen instance. In ML-kNN, all test instances which get the same number of votes among k nearest neighbors have the same probability to be assigned a label, which may cause improper decision since it ignores the local difference of samples. Actually, in real world data sets, the instances with (or without) label l from different locations may have different numbers of neighbors with the label l. In this paper, we propose a locally adaptive Multi-Label k-Nearest Neighbor method to address this problem, which takes the local difference of samples into account. We show how a simple modification to the posterior probability expression, previously used in ML-kNN algorithm, allows us to take the local difference into account. Experimental results on benchmark data sets demonstrate that our approach has superior classification performance with respect to other kNN-based algorithms.

1 Introduction

1.1 Background

Multi-Label classification has received considerable attention over the past several years. In multi-label classification, each instance in the dataset is associated with a set of labels, and the task of multi-label classification problem is to output a label set whose size is unknown for each test instances. Multi-label problems are ubiquitous in the real world, for example, in image categorization, each image can be associated with multiple labels, such as *sea*, *desert* and *mountain* [1]; in text categorization, each text may belong to a set of topics, such as *economics, poetry* and *health* [2]; in bioinformatics, a gene may be related to multiple functions, such as *metabolism* and *protein synthesis* [3].

Formally, let $\mathcal{X} = \mathcal{R}^d$ denote the d-dimensional feature space and $\mathcal{Y} = \{0,1\}^L$ be the label space with L possible labels, then the goal of multi-label classifier is to learn a function $\boldsymbol{f}\colon \mathcal{X} \mapsto \mathcal{Y}$. Given a multi-label dataset $\mathcal{D}$, we can divide it into feature space $\mathcal{X}$ and label space $\mathcal{Y}$. An instance x_i is associated with a subset of labels $Y_i \subseteq \mathcal{Y}$ (finite set of labels), and a multi-label dataset is composed of m examples (x_1, Y_1),(x_2, Y_2), ... ,(x_n, Y_n) [4].

Given a multi-label learning task, it can be transformed into other well-established learning tasks. This category of approaches is formally defined as $Problem\ Transformation$ method. In this way, we can decompose a multi-label problem into multiple single-label problems, and each single-label problems can be tackled by a binary classifier. Thus, the multi-label classification function can be represented in another form $\boldsymbol{f}=\{f_1, f_2, \ldots, f_L\}$ in this way. Problem Transformation is widely used in multi-label learning problems for its greater flexibility [8,9,11]. Another way to tackle multi-label classification problems is so called $Algorithm\ Adaptation$ method [5]. This category of approaches tackles multi-label learning problem by adapting existing popular learning approaches such as AdaBoost, Neural Networks or kNN to deal with the multi-label problems directly [2,12,13].

According to the idea of Algorithm Adaptation, Zhang and Zhou [6] proposed Multi-Label k-Nearest Neighbor (ML-kNN). It is the first lazy learning approach and one of the most influential multi-label classification approaches. The basic idea of this approach is to adapt the classic kNN algorithm to deal with multi-label classification problems, where maximum a posteriori (MAP) rule is utilized to adaptively adjust decision boundary for each new instance. In this method, the test instances which get the same number of votes among k nearest neighbors have the same probability to be assigned a label. It may cause improper decision since it ignores the local difference of samples. Actually, in real world data sets, the instances with (or without) label l from different locations may have different numbers of neighbors with the label l. Thus, in this paper, we propose a locally adaptive Multi-Label k-Nearest Neighbor method to address this problem.

1.2 Motivation

We begin by conducting a simple experiment to try to show the local difference of samples. The local difference here means the instances with (or without) the l-th label from different locations may have different numbers of neighbors with the l-th label.

For a dataset, we first find the k nearest neighbors of each instance x and denote as $\mathcal{N}(x)$. Then we can count the number of neighbors of x with label l. The counting vector can be defined as:

$$\boldsymbol{C}_x(l) = \sum\nolimits_{(x^*, Y^*) \in \mathcal{N}(x)} Y^*(l), l \in \mathcal{Y} \tag{1}$$

After calculating above statistics, we can figure out if the distribution of $\boldsymbol{C}_x(l)$ is related to the location information. In our experiments, we separate the dataset into five clusters, and use the cluster index to represent location information. For each cluster S_j and each label l, we calculate: (1) the average $\boldsymbol{C}_x(l)$ of the instances with label l (defined as Equation (2)); (2) the average $\boldsymbol{C}_x(l)$ of the instances without label l (defined as Equation (3)).

$$\overline{\boldsymbol{C}}(S_j, l) = \frac{1}{|S_j^{l1}|} \sum\nolimits_{(x,Y) \in S_j^{l1}} \boldsymbol{C}_x(l) \tag{2}$$

$$\overline{\boldsymbol{C}^*}(S_j, l) = \frac{1}{|S_j^{l0}|} \sum\nolimits_{(x,Y)\in S_j^{l0}} \boldsymbol{C}_x(l) \tag{3}$$

where $S_j^{l1} = \{(x,Y)|(x,Y) \in S_j, Y(l) = 1\}$ and $S_j^{l0} = \{(x,Y)|(x,Y) \in S_j, Y(l) = 0\}$.

We conduct the experiment on an image data set *scene*, which has 2407 instances and 6 labels. The results are shown in Table 1. The first part and the second part respectively show $\overline{\boldsymbol{C}}(S_j, l)$ and $\overline{\boldsymbol{C}^*}(S_j, l)$ of each cluster S_j and each label l. As is shown in Table 1, for a same label l, the $\overline{\boldsymbol{C}}(S_j, l)$ and $\overline{\boldsymbol{C}^*}(S_j, l)$ of different clusters may vary tremendously.

Table 1: Each cell of the table means the average $\boldsymbol{C}_x(l)$ of the instances with (or without) label l in each cluster.

	label	$beach$	$sunset$	$fall$	$field$	$mountain$	$urban$
	cluster 1	1.432	0.926	1.470	2.000	1.785	5.532
	cluster 2	1.600	1.500	2.047	6.188	2.256	1.000
$\overline{\boldsymbol{C}}$	cluster 3	1.250	5.265	6.055	1.571	0.500	1.090
	cluster 4	1.044	1.607	1.214	1.936	4.761	2.000
	cluster 5	4.863	1.333	1.000	2.333	1.444	1.956
	cluster 1	0.401	0.029	0.083	0.176	1.073	4.503
	cluster 2	0.221	0.005	0.219	4.462	1.116	0.167
$\overline{\boldsymbol{C}^*}$	cluster 3	0.098	0.248	1.696	0.201	0.134	0.198
	cluster 4	0.346	0.067	0.123	0.556	3.759	1.369
	cluster 5	3.543	0.006	0.195	0.430	0.891	1.084

The above results hint that the distribution of $\boldsymbol{C}_x(l)$ is significantly related to the location information. In ML-kNN, however, the local difference of samples is ignored, which may cause the improper decision. To take the local difference into account, we propose a locally adaptive Multi-Label k-Nearest Neighbor method in this paper. In our approach, the test instances which get the same number of votes among k nearest neighbors may have different probabilities to be assigned a label if they come from different regions. Experimental results on benchmark data sets demonstrate that our approach has superior classification performance with respect to previous ML-kNN algorithm, especially on large scale data sets[3].

1.3 Paper Organization

The rest of this paper is organized as follows. The related work is discussed in Section 2. The details of our approach are proposed in Section 3. After that, the experiment results are reported in Section 4. Finally, the conclusion is summarized in Section 5.

[3] The code available at https://github.com/DENGBAODAGE/LAMLKNN

2 Related Work

The k-nearest neighbors (kNN) rule [7] is one of the oldest and simplest methods for pattern classification. For traditional single-label classification problems, the kNN rule usually classifies each unlabeled instance by the majority label among its k nearest neighbors in the training data. The kNN-based methods often yield competitive results and have been widely used in practical applications mainly due to its implementation simplicity. However, for multi-label classification, the traditional kNN rule is inappropriate mainly due to the severe class-imbalance issue.

ML-kNN was proposed based on the traditional kNN algorithm to deal with multi-label classification problems. Rather than classifying new instance by the majority label among its k nearest neighbors, ML-kNN employs maximum a posteriori (MAP) principle to predict the set of labels of the new instance.

$$\begin{aligned} Y_t(l) &= \underset{b\in\{0,1\}}{\arg\max}\, P(H_b^l | E^l_{\boldsymbol{C_t}(l)}) \\ &= \underset{b\in\{0,1\}}{\arg\max}\, P(H_b^l) P(E^l_{\boldsymbol{C_t}(l)} | H_b^l) \end{aligned} \tag{4}$$

where $Y_t(l)$ is the label vector for the new instance t. $\boldsymbol{C_t}(l)$ is the same as described previously. H_1^l represents the event that t has label l, while H_0^l represents the event that t doesn't have label l. $E^l_{\boldsymbol{C_t}(l)}$ denotes the event that, among the k nearest neighbors of t, there are exactly $\boldsymbol{C_t}(l)$ instances which have label l. The prior probability $P(H_b^l)$ and the conditional probability $P(E^l_{\boldsymbol{C_t}(l)}|H_b^l)$ in Equation (4) can all be estimated from the training dataset in advance.

The reported experiment results show that ML-kNN performed well on several real world data sets. However, it ignores the local differece when using utilizing maximum a posteriori rule, and we think the location information of the new instance is helpful especially for large scale data sets.

There are also some other kNN based approaches to handle multi-label classification problems. Note that ML-kNN is a first-order approach which reasons the relevance of each label separately. Considering that this method is ignorant of exploiting label correlations, a dependent multi-label classification method derived from ML-kNN is proposed in [14], which takes into account the dependencies between labels. In order to exploit the non-parametric property of classical kNN method, Wang et al. [15] further developed classical KNN method, and proposed a Class Balanced K-Nearest Neighbor (BKNN) approach for multi-label classification. This method picks up the most representative training data points from every class with equal number, such that the label of a test data point is determined via the information from all the classes in a balanced manner. In [18], a kNN based ranking approach is proposed to solve the multi-label classification problem. This approach exploits a ranking model to learn which neighbor′s labels are more trustable candidates for a weighted KNN-based strategy, and then assigns higher weights to those candidates when making weighted-voting decisions.

3 Methodology

As described in previous sections, we try to take the local difference into account by modifying the posterior probability expression used in ML-kNN algorithm. How to exploit the location information when using MAP principle to assign labels to a new instance? In this section, we introduce a Locally Adaptive Multi-Label k-Nearest Neighbor algorithm to address this problem.

Inspired by the results presented in Section 1.2, we firstly separate the training data into m groups $S_1, S_2, \ldots, S_m$ via clustering, where the average $\boldsymbol{C}_x(l)$ of instances in the different clusters may vary tremendously. For each test instance t, we can identify which group should it be assigned to by measuring the distance between the test instance and each cluster center.

$$w_t = \operatorname*{arg\,min}_{1 \leq j \leq m} \|x_t - c_j\|^2 \tag{5}$$

where w_t is the index of cluster to which should the test instance t assign. c_j stands for the center point of cluster S_j.

Therefore we can get two important information of the test instance t: $\boldsymbol{C}_t$ (records the numbers of x′s neighbors with each label) and w_t (stands for the index of cluster to which should the test instance t assign). Then based on the membership counting vector $\boldsymbol{C}_t$ and the location information w_t, the category vector Y_t can be determined using the following maximum a posteriori principle:

$$Y_t(l) = \operatorname*{arg\,max}_{b \in \{0,1\}} P(H_b^l | E^l_{\boldsymbol{C_t}(l)}, W_{w_t}) \tag{6}$$

where H_b^l and $E^l_{\boldsymbol{C_t}(l)}$ is the same as described in Section 2. W_{w_t} denotes the event that the test instance t can be assigned to the cluster S_{w_t}. Based on Bayes theorem, we have:

$$Y_t(l) = \operatorname*{arg\,max}_{b \in \{0,1\}} P(H_b^l) P(E^l_{\boldsymbol{C_t}(l)}, W_{w_t} | H_b^l) \tag{7}$$

The prior probability $P(H_b^l)$ and the likelihood $P(E^l_{\boldsymbol{C_t}(l)}, W_{w_t} | H_b^l)$ can be estimated from the training data.

Equation (6) can also be rewritten by another way(based on Bayes theorem):

$$\begin{aligned} Y_t(l) &= \operatorname*{arg\,max}_{b \in \{0,1\}} P(H_b^l,\, E^l_{\boldsymbol{C_t}(l)},\, W_{w_t}) \\ &= \operatorname*{arg\,max}_{b \in \{0,1\}} P(H_b^l, W_{w_t}) P(E^l_{\boldsymbol{C_t}(l)} | H_b^l,\, W_{w_t}) \\ &= \operatorname*{arg\,max}_{b \in \{0,1\}} P(W_{w_t}) P(H_b^l | W_{w_t}) P(E^l_{\boldsymbol{C_t}(l)} | H_b^l,\, W_{w_t}) \end{aligned} \tag{8}$$

where $P(W_{w_t})$ represents the prior probability that W_{w_t} holds. $P(H_b^l | W_{w_t})$ represents the conditional probability that H_b^l holds when W_{w_t} holds. Furthermore, the conditional probability $P(E^l_{\boldsymbol{C_t}(l)} | H_b^l,\, W_{w_t})$ represents the likelihood that the instance x has $\boldsymbol{C}_t(l)$ neighbors with label l when H_b^l and W_{w_t} both hold.

By comparing Equation (8) and Equation (4) (in Section 2), it is intuitive to understand how we exploit the location information by involving W_{w_t} in posterior probability expression. In our method, the category vector Y_t of the new instance t depends on the membership counting vector $\boldsymbol{C_t}$ as well as the location information w_t. Unlike ML-kNN, our approach can derive different probabilities of assigning a label to new instances which get the same number of votes among k nearest neighbors but come from different regions. Actually, ML-kNN can be regarded as a special case of our approach with m=1. Note that Equation (7) and (8) we described above are actually equivalent. We choose the latter version in our implementation.

All the three terms in Equation (8) can be estimated from the training data. Firstly, the prior probability $P(W_{w_t})$ is estimated by calculating the proportion of the cluster S_{w_t} in training data:

$$P(W_{w_t}) = \frac{|S_{w_t}|}{|S_{train}|} \tag{9}$$

where $|S_{w_t}|$ and $|S_{train}|$ is the size of cluster w_t and training dataset.

Then the conditional probability $P(H_b^l|W_{w_t})$ are estimated by counting the number of training examples associated with each label in each cluster:

$$\begin{aligned} P(H_1^l|W_{w_t}) &= \frac{s + \sum_{(x,Y)\in S_{w_t}} Y(l)}{2 \times s + |S_{w_t}|} \qquad (l \in \mathcal{Y}) \\ P(H_0^l|W_{w_t}) &= 1 - P(H_1^l|W_{w_t}) \qquad\quad (l \in \mathcal{Y}) \end{aligned} \tag{10}$$

where s is the smoothing parameter controlling the effect of uniform prior on the estimation [6].

Finally, the estimation process for likelihoods $P(E^l_{\boldsymbol{C_t}(l)}|H_b^l, W_{w_t})$ is involved. For each label l, we calculate:

$$\begin{aligned} \mathcal{K}_l(r) &= \sum\nolimits_{(x,Y)\in \boldsymbol{S_{w_t}}} Y(l) \cdot [\![\boldsymbol{C}_x(l) = r]\!] \qquad (l \in \mathcal{Y},\ 0 \le r \le k) \\ \mathcal{K}'_l(r) &= \sum\nolimits_{(x,Y)\in \boldsymbol{S_{w_t}}} (1 - Y(l)) \cdot [\![\boldsymbol{C}_x(l) = r]\!] \quad (l \in \mathcal{Y},\ 0 \le r \le k) \end{aligned} \tag{11}$$

$\mathcal{K}_l(C)$ counts the number of training examples which have label l and have exactly C neighbors with label l, while $\mathcal{K}'_l(C)$ counts the number of training examples which don't have label l and have exactly C neighbors with label l. For any $\cdot$, $[\![\cdot]\!]$equals 1 if $\cdot$ holds and 0 otherwise. After calculate $\mathcal{K}_l(C)$ and $\mathcal{K}'_l(C)$, we can estimate the likelihood in Equation (8):

$$\begin{aligned} P(E^l_{\boldsymbol{C_t}(l)}|H_1^l, W_{w_t}) &= \frac{s + \mathcal{K}_l(\boldsymbol{C_t(l)})}{s \times (k+1) + \sum_{r=0}^k \mathcal{K}_l(r)} \\ P(E^l_{\boldsymbol{C_t}(l)}|H_0^l, W_{w_t}) &= \frac{s + \mathcal{K}'_l(\boldsymbol{C_t(l)})}{s \times (k+1) + \sum_{r=0}^k \mathcal{K}'_l(r)} \end{aligned} \tag{12}$$

The following pseudo-code illustrates the complete description of our method. In training phase, we estimate the prior probability $P(W_j)$, the conditional probabilities $P(H_1^l|W_j)$, $P(H_1^0|W_j)$, the statistics $\mathcal{K}_l(r)$, and $\mathcal{K}'_l(r)$ (steps from 5 to 13). In classifying phase, the predicted label set of test instance t can be determined using the maximum a posteriori principle (by substituting Equation (9), Equation (10) and Equation (12) into Equation (8)).

Train(S_{train}, k, m)

Divide training data into m clusters $\{S_1, S_2 \ldots, S_m\}$ with k-means
for $i = 1$ to $|S_{train}|$ **do**:
  Identify k nearest neighbors $\mathcal{N}(x_i)$ for x_i
end
for $j = 1$ to m **do**:
  $P(W_j) = \dfrac{|S_j|}{|S_{train}|}$
  for $l = 1$ to L **do**:
    $P(H_1^l|W_j) = \dfrac{s + \sum_{(x,Y)\in S_j} Y(l)}{2 \times s + |S_j|}$
    $P(H_0^l|W_j) = 1 - P(H_1^l|W_j)$
    $\mathcal{K}_l(r) = \sum_{(x,Y)\in \boldsymbol{S_j}} Y(l) \cdot [\![\boldsymbol{C}_x(l) = r]\!] \qquad (\ 0 \leq r \leq k)$
    $\mathcal{K}'_l(r) = \sum_{(x,Y)\in \boldsymbol{S_j}} (1 - Y(l)) \cdot [\![\boldsymbol{C}_x(l) = r]\!] \qquad (\ 0 \leq r \leq k)$
  end
end

Classify(t, k)

Identify S_{w_t} (the cluster should t be assigned to) using Equation (5)
Identify k nearest neighbors $\mathcal{N}(t)$ for t
for $l = 1$ to L **do**:
  Calculate $C_t(l)$ according to Equation (1)
  Estimate $P(E^l_{\boldsymbol{C_t}(l)}|H_1^l, W_{w_t})$ and $P(E^l_{\boldsymbol{C_t}(l)}|H_0^l, W_{w_t})$ according to (12)
  $Y_t(l) = \arg\max_{b\in\{0,1\}} P(H_b^l|E^l_{\boldsymbol{C_t}(l)}, W_{w_t})$
  $\qquad = \arg\max_{b\in\{0,1\}} P(W_{w_t})P(H_b^l|W_{w_t})P(E^l_{\boldsymbol{C_t}(l)}|H_b^l, W_{w_t})$
end

4 Experiment

We compare the our proposed method with other multi-label lazy learning algorithms on several data sets. In the following sections, we first describe the experiment setup including the data sets, the evaluation metrics, and the compared algorithms; Then we discuss the experiment results.

4.1 Experiment Setup

Data sets: We evaluated the algorithm presented in the previous section on twelve data sets[4] of varying size and difficulty. The statistics of the data sets are shown in Table 2. Six regular-scale data sets (first part) as well as six large-scale

[4] Data sets were downloaded from http://mulan.sourceforge.net/datasets.html and http://meka.sourceforge.net/#datasets

data sets (second part) are included (the data sets are roughly ordered by the number of instances). There are two additional properties [10] to measure the density of labels:

- The cardinality of a dataset $\mathcal{S}$ is the mean of the number of labels of the instances that belong to $\mathcal{S}$, defined as:

$$cardinality(S) = \frac{1}{n}\sum_{i=1}^{n} |Y_i| \tag{13}$$

- The density of $\mathcal{S}$ is the mean of the number of labels of the instances that belong to $\mathcal{S}$ divided by L, defined as:

$$density(S) = \frac{1}{n}\sum_{i=1}^{n} \frac{|Y_i|}{L} \tag{14}$$

Table 2: Multi-label data sets used in experiments.

name	*domain*	*instances*	*dimension*	*labels*	*cardinality*	*density*
emotions	music	593	72	6	1.869	0.311
birds	audio	645	260	19	1.014	0.053
enron	text	1702	1001	53	3.378	0.064
scene	image	2407	294	6	1.074	0.179
yeast	biology	2417	103	14	4.237	0.3003
slashdot	text	3782	1079	22	1.181	0.054
bibtex	text	7395	1836	159	2.402	0.015
corel5k	image	5000	499	374	3.522	0.009
corel16k (1)	image	13766	500	153	2.859	0.019
corel16k (2)	image	13761	500	164	2.882	0.018
corel16k (3)	image	13760	500	154	2.829	0.018
ohsumed	text	13929	1002	23	1.663	0.072

Metrics: In multi-label learning, the evaluation is more complicated than that in single-label learning. Various evaluation metrics have been proposed to measure the performance of multi-label classifier. We use five commonly used metrics: *hamming lass*,*ranking loss*, *coverage*, *one error* and *average precision* [17]. These above five metrics evaluate the performance of a multi-label classifier from different horizon. Note that for average precision, the larger the values the better the performance, while for other four metrics, the smaller the values the better the performance.

Compared Algorithms: We compare the performance of our proposed method with that of three other kNN-based multi-label approaches: BRkNN, ML-kNN and DML-kNN. BRkNN [16] is an adaptation of the kNN algorithm that is conceptually equivalent to using BR method in conjunction with the traditional kNN algorithm. As we discussed in Section 2, DML-kNN is an extension approach based on ML-kNN, which takes into account the dependencies between labels.

4.2 Results

Following the experiment setup described above, we conduct the comparison experiments. The experimental results of each algorithm on each data set are respectively reported in Table 3 and Table 4. For each algorithm, the k value is determined by cross-validation. We can see that our proposed method LAML-kNN outperform the compared methods in most cases. Furthermore, the advantages of our approach are more obvious on the large-scale data sets (in Table 4) than that on the regular-scale data sets (in Table 3).

Table 3: Experimental results of each algorithm on regular-scale data sets.

Metrics	algorithms	emotions	birds	enron	scene	yeast	slashdot
Hamming Loss	LAMLkNN	0.197	0.045	**0.050**	0.097	**0.198**	**0.050**
	MLkNN	0.191	**0.044**	0.051	**0.096**	0.198	0.053
	BRkNN	0.193	0.045	0.058	0.105	0.203	0.090
	DMLkNN	**0.187**	0.045	0.051	0.097	0.198	0.051
Ranking Loss	LAMLkNN	0.151	**0.093**	**0.088**	0.090	**0.170**	**0.157**
	MLkNN	**0.145**	0.102	0.093	0.096	0.171	0.168
	BRkNN	0.151	0.119	0.152	0.106	0.183	0.242
	DMLkNN	0.147	0.101	0.092	**0.083**	0.170	0.161
OneError	LAMLkNN	**0.243**	**0.709**	**0.252**	**0.230**	**0.236**	**0.610**
	MLkNN	0.253	0.728	0.280	0.233	0.242	0.645
	BRkNN	0.267	0.726	0.459	0.291	0.242	0.891
	DMLkNN	0.253	0.721	0.282	0.238	0.237	0.612
Coverage	LAMLkNN	0.307	**0.138**	**0.240**	0.093	**0.454**	**0.172**
	MLkNN	**0.298**	0.147	0.249	0.096	0.455	0.184
	BRkNN	0.303	0.172	0.382	0.105	0.472	0.253
	DMLkNN	0.300	0.145	0.246	**0.086**	0.455	0.176
Avg-Precision	LAMLkNN	**0.818**	**0.609**	**0.654**	0.856	**0.759**	**0.530**
	MLkNN	0.818	0.578	0.640	0.852	0.757	0.502
	BRkNN	0.810	0.570	0.564	0.824	0.754	0.334
	DMLkNN	0.816	0.580	0.643	**0.857**	0.758	0.526

The experimental results on benchmark data sets and diverse evaluation metrics validate the superior effectiveness of our approach to existing kNN-based multi-label approaches. Meanwhile, the experimental results demonstrate the number of clusters does not significantly affect the classifier′s performance on large-scale data sets. We fix the k value as well as change m (the number of clusters) for our proposed approach, then compare the *average precision* of each case with that of ML-kNN. From Figure 2 we can see, on these six large-scale data sets, across all the m value, our approach superior to ML-kNN. But the performance of our approach is sensitive to the cluster number m on small-scale

Table 4: Experimental results of each algorithm on large-scale data sets.

Metrics	algorithms	bibtex	corel5k	corel16k(1)	corel16k(2)	corel16k(3)	ohsumed
Hamming Loss	LAMLkNN	**0.014**	**0.009**	**0.019**	**0.016**	**0.017**	**0.070**
	MLkNN	0.014	0.009	0.019	0.016	0.017	0.071
	BRkNN	0.015	0.010	0.019	0.016	0.017	0.072
	DMLkNN	0.014	0.009	0.019	0.016	0.017	0.071
Ranking Loss	LAMLkNN	**0.145**	**0.118**	**0.160**	0.180	0.184	**0.214**
	MLkNN	0.217	0.127	0.175	0.181	0.183	0.231
	BRkNN	0.297	0.292	0.268	0.279	0.259	0.277
	DMLkNN	0.208	0.127	0.174	**0.179**	**0.179**	0.231
OneError	LAMLkNN	**0.542**	**0.670**	**0.698**	**0.731**	**0.732**	**0.613**
	MLkNN	0.578	0.706	0.736	0.782	0.769	0.639
	BRkNN	0.680	0.742	0.771	0.917	0.769	0.706
	DMLkNN	0.576	0.722	0.729	0.767	0.764	0.640
Coverage	LAMLkNN	**0.222**	**0.272**	**0.312**	**0.316**	**0.331**	**0.292**
	MLkNN	0.354	0.298	0.342	0.326	0.336	0.311
	BRkNN	0.431	0.591	0.493	0.475	0.476	0.361
	DMLkNN	0.332	0.299	0.339	0.327	0.332	0.311
Avg-Precision	LAMLkNN	**0.395**	**0.288**	**0.305**	**0.276**	**0.267**	**0.470**
	MLkNN	0.349	0.275	0.288	0.255	0.253	0.442
	BRkNN	0.268	0.210	0.200	0.170	0.222	0.394
	DMLkNN	0.350	0.265	0.291	0.266	0.258	0.441

data sets (see in Figure 1). The proposed approach may inferior to ML-kNN if we select improper m for small-scale data sets (e.g. *emotions* and *yeast*). We think one possible reason may be due to lack of prior acknowledge when the size of each cluster is too small.

5 CONCLUSION

To achieve more effective multi-label classification using lazy learning method, in this paper, we introduced an original kNN-based multi-label classification algorithm. We show how to take into account the local difference of samples by modifying the posterior probability expression previously used in ML-kNN algorithm. The experimental results on benchmark data sets demonstrate effective classification of our approach, especially on large scale data sets.

6 Acknowledgement

It was supported by NSF Chongqing China (cstc2017zdcy-zdyf0366). Li Li is the corresponding author for the paper.

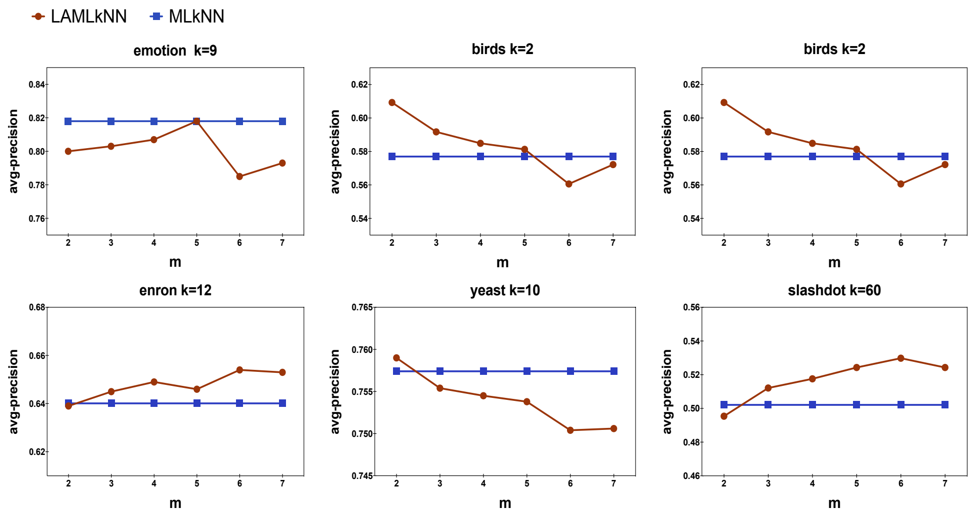

Fig. 1: Comparison results on six regular-scale data sets.

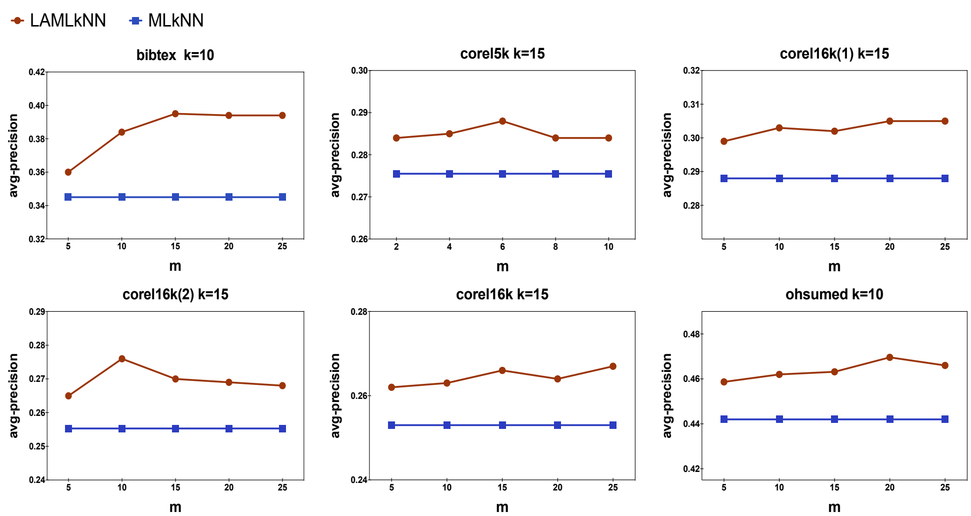

Fig. 2: Comparison results on six large-scale data sets.

References

1. Wang, J., Yang, Y., Mao, J., Huang, Z., Huang, C., Xu, W.: Cnn-rnn: A unified framework for multi-label image classification. In: Proceedings of CVPR, pp. 2285-2294 (2016)
2. Nam, J., Kim, J., Menca, E. L., Gurevych, I., Frnkranz, J.: Large-scale multi-label text classification—revisiting neural networks. In: Proceedings of ECML/PKDD 2014, pp. 437-452 (2014)

3. Elisseeff, A., Weston, J.: A kernel method for multi-labelled classification. In: Proceedings of NIPS, Vol.14, pp.681-687 (2001)
4. Zhang, M. L., Zhou, Z. H.: A review on multi-label learning algorithms. IEEE Trans. Knowl. Data Eng. 26(8), 1819-1837 (2014)
5. Tsoumakas, G., Katakis, I., Taniar, D.: Multi-label classification: an overview. International Journal of Data Warehousing & Mining, 3(3), 1-13 (2007)
6. Zhang, M. L., Zhou, Z. H.: Ml-knn: a lazy learning approach to multi-label learning. Pattern Recogn. 40(7), 2038-2048 (2007)
7. Cover, T. M., Hart, P. E.: Nearest neighbor pattern classification. IEEE Trans. Inf. Theor. 13(1), 21-27 (1967)
8. Read, J., Pfahringer, B., Holmes, G., Frank, E.: Classifier chains for multi-label classification. Mach. Learn. 85(3), 333 (2011)
9. Brinker, K.: Multilabel classification via calibrated label ranking. Mach. Learn. 73(2), 133-153 (2008)
10. Tsoumakas, G., Katakis, I., Vlahavas, I.: Mining multi-label data. Data mining and knowledge discovery handbook, pp. 667-685 (2009)
11. Zhang, M. L.: LIFT: multi-label learning with label-specific features. In: IJCAI, Vol.37, pp.1609-1614 (2011)
12. Zhang, M. L., Zhou, Z. H.: Multilabel neural networks with applications to functional genomics and text categorization. IEEE Trans. Knowl. Data Eng. 18(10), 1338-1351 (2006)
13. Zhang, M. L.: Ml-rbf: rbf neural networks for multi-label learning. Neural Process. Lett. 29(2), 61-74 (2009)
14. Younes, Z., Abdallah, F., Denoeux, T., Snoussi, H.: A dependent multilabel classification method derived from the k -nearest neighbor rule. Eurasip J. Adv. Sig. Process. 2011(1), 1-14 (2011)
15. Wang, H., Ding, C. H. Q., Huang, H.: Multi-Label Classification: Inconsistency and Class Balanced K-Nearest Neighbor. In: Proceedings of AAAI (2010)
16. Spyromitros, E., Tsoumakas, G., Vlahavas, I.: An Empirical Study of Lazy Multilabel Classification Algorithms. In: Proceedings of the fifth Hellenic Conference on Artificial Intelligence, pp. 401-406 (2008)
17. Wu, X. Z., Zhou, Z. H.: A Unified View of Multi-Label Performance Measures. arXiv preprint arXiv: 1609.00288 (2016)
18. Chiang, T. H., Lo, H. Y., Lin, S. D.: A ranking-based knn approach for multi-label classification. In: Proceedings of ACML, pp. 81-96 (2012)