2015-12

Lazy multi-label learning algorithms based on mutuality strategies

Journal of Intelligent and Robotic Systems, Dordrecht, v. 80, suppl 1, p. S261-S276, Dec. 2015
http://www.producao.usp.br/handle/BDPI/50106

J Intell Robot Syst (2015) 80 (Suppl 1):S261–S276
DOI 10.1007/s10846-014-0144-4

Lazy Multi-label Learning Algorithms Based on Mutuality Strategies

Everton Alvares Cherman · Newton Spolaôr · Jorge Valverde-Rebaza · Maria Carolina Monard

Received: 28 February 2014 / Accepted: 26 September 2014 / Published online: 17 October 2014

Abstract Lazy multi-label learning algorithms have become an important research topic within the multi-label community. These algorithms usually consider the set of standard k-Nearest Neighbors of a new instance to predict its labels (multi-label). The prediction is made by following a voting criteria within the multi-labels of the set of k-Nearest Neighbors of the new instance. This work proposes the use of two alternative strategies to identify the set of these examples: the Mutual and Not Mutual Nearest Neighbors rules, which have already been used by lazy single-learning algorithms. In this work, we use these strategies to extend the lazy multi-label algorithm *BRkNN*. An experimental evaluation carried out to compare both mutuality strategies with the original *BRkNN* algorithm and the well-known *MLkNN* lazy algorithm on 15 benchmark datasets showed that *MLkNN* presented the best predictive performance for the *Hamming-Loss* evaluation measure, although it was significantly outperformed by the mutuality strategies when *F-Measure* is considered. The best results of the lazy algorithms were also compared with the results obtained by the Binary Relevance approach using three different base learning algorithms.

Keywords Machine learning · Multi-label learning · Lazy algorithms · Nearest Neighbors

E. A. Cherman (✉) · N. Spolaôr · J. Valverde-Rebaza · M. C. Monard
Laboratory of Computational Intelligence, Institute of Mathematics and Computer Science, University of São Paulo, 13560-970, São Carlos, SP, Brazil
e-mail: evertoncherman@gmail.com

N. Spolaôr
e-mail: newtonspolaor@gmail.com

J. Valverde-Rebaza
e-mail: jvalverr@icmc.usp.br

M. C. Monard
e-mail: mcmonard@icmc.usp.br

1 Introduction

Traditional supervised learning algorithms are single-label, in which only one label from a set of labels L is associated to each example in the dataset. On the other hand, in multi-label learning, a set of labels (called multi-label) is associated to each example in the dataset. Thus, an example is associated to one or more labels from the set of labels L. It is worth noting that an example may be associated to more than one label not due to ambiguity (*fuzzy* membership), but because of multiplicity (*full* membership). Nevertheless, this makes the multi-label learning task more difficult than the traditional single-label learning task.

Multi-label learning is an emerging and promising research topic of machine learning due to the increasing number of new applications where examples are annotated with more than one label. Multi-label classification has been widely applied in applications such as semantic annotation of video and image, functional

genomics, web mining, information retrieval, tag recommendation, music categorization into emotions and automatic image annotation [18], to name just a few.

Multi-label learning methods can be grouped into two main categories: *problem transformation* and *algorithm adaptation* [12]. The first group of methods are algorithm independent and they transform the multi-label classification problem into either one or more single-label classification problems where any state-of-the-art single-label learning method can be used as a base-algorithm. The second group of methods extends specific learning algorithms in order to handle multi-label data directly.

In this second group, the instance-based (or *lazy*) multi-label learning has become an important research topic since the k-Nearest Neighbors concept was introduced to cope with multi-label datasets. The lazy multi-label learning methods usually take standard k-Nearest Neighbors as the input. Then, they predict the labels (multi-label) of a new instance based on the labels in the multi-labels of the k-Nearest Neighbors by following a voting criteria within them. The multi-label algorithms proposed in the literature differ in such voting criteria. Typical examples include *BRkNN* [11] and *MLkNN* [17]. The underlying reason of the lazy learning methods popularity is due to the fact that they are conceptually simple, efficient and can be easily implemented.

In previous work [2], and based on the k-Nearest Neighbors of a new instance, we proposed two strategies, *MLMUT* and *MLnotMUT*, to find the set of examples B which will be used by the voting criteria to predict its multi-label. While the first strategy might restrict to less than k the number of examples in B, the second one might increase it. One advantage of these two strategies is that the number of selected examples in B, which are used to classify a new instance, is a variable determined by the training set.

Although both strategies have been evaluated for graph single-label learning, it is not of our knowledge that they have been considered for multi-label learning. In [2], these strategies were experimentally evaluated on 10 datasets and compared with *BRkNN*, as well as with algorithms which follow the problem transformation approach, specifically, the binary relevance approach. The best results of the lazy algorithms were also compared with the results obtained by the Binary Relevance approach using three different base learning algorithms: *J48*, *Naive Bayes* and *SMO*. Good results were shown by the proposed *MLMUT* and *MLnotMUT* approaches, mainly when compared to the results obtained by the algorithms which follow the problem transformation approach. Thus, in this work we carry out a more thorough experimental evaluation of *MLMUT* and *MLnotMUT* on 15 datasets, and compare these results with the ones obtained by *BRkNN* and *MLkNN*, where the latter is considered a state-of-the-art lazy algorithm. Good results were obtained by the proposed algorithms *MLMUT* and *MLnotMUT*. It was found, for example, that using *MLMUT* and *MLnotMUT* led to significant improvements in terms of a traditional multi-label evaluation measure.

This work is organized as follows: Section 2 briefly describes multi-label learning, as well as the lazy algorithms *BRkNN* and *MLkNN*, the multi-label classifier evaluation measures and a baseline classifier $General_B$ used in this work. Section 3 explains the lazy algorithms *MLMUT* and *MLnotMUT* which implement, respectively, the Mutual and Not Mutual Nearest Neighbors strategies. The experimental evaluation is described in Section 4 and Section 5 shows the conclusions.

2 Multi-label Learning

In *single-label* learning, the input to the learning algorithm consists of a set of N classified instances (or examples E_i) $\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)\}$ for some unknown function $y = f(\mathbf{x})$. The $\mathbf{x}_i$ are tipically vectors of the form $(x_{i1}, x_{i2}, \ldots, x_{iM})$ whose components are discrete or real values called *features* or *attributes*. Thus, x_{ij} denotes the value of the j-th feature X_j of $\mathbf{x}_i$, as shown in Table 1(a). Moreover, in single-label learning, each example in the dataset is associated with only one label from a set of labels. The task is called *binary classification* if there are only two possible labels (Yes/No), and *multi-class classification* when the number of labels is greater than two. Given the set of classified instances (the training set), a single-label learning algorithm generates a *classifier* H which is a hypothesis about the true unknown function f, *i.e.*, $H(\mathbf{x}) \approx f(\mathbf{x})$. Given a new example $\mathbf{x}$, $H(\mathbf{x})$ predicts the corresponding y value.

On the other hand, in *multi-label* learning each example $E_i = (\mathbf{x}_i, Y_i)$ is associated with a set of labels Y_i, where $Y_i \subseteq L$, $Y_i \neq \emptyset$ and $L =$

Table 1 Single and multi-label datasets in the attribute-value format

	X_1	X_2	...	X_M	Y
(a) Single-label					
E_1	x_{11}	x_{12}	...	x_{1M}	y_1
E_2	x_{21}	x_{22}	...	x_{2M}	y_2
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
E_N	x_{N1}	x_{N2}	...	x_{NM}	y_N
(b) Multi-label					
E_1	x_{11}	x_{12}	...	x_{1M}	Y_1
E_2	x_{21}	x_{22}	...	x_{2M}	Y_2
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
E_N	x_{N1}	x_{N2}	...	x_{NM}	Y_N

$\{y_1, y_2, y_3, \ldots y_q\}$, as shown in Table 1(b). In this case, the multi-label learning algorithm generates a classifier H which, given a new example $\mathbf{x}$, $H(\mathbf{x})$ predicts the corresponding multi-label Y.

As already mentioned, several approaches have been proposed to develop multi-label learning algorithms. In this work we use algorithms which follow the *algorithm adaptation* approach. This approach considers methods which extend specific algorithms to handle multi-label data directly, such as: *C4.5 Multi-label* [3], which extends the single-label decision tree *C4.5* algorithm; *BP-MLL* [16], which adapts the back-propagation strategy to multi-label data; *MLkNN* [17], *DMLkNN* [15] and *BRkNN* [11], which propose different adaptations of the *lazy* single-label *kNN* algorithm. In this work we use the *MLkNN* algorithm, as well as an extention of the *BRkNN* algorithm. Both algorithms are described next.

2.1 Algorithms *BRkNN* and *MLkNN*

The multi-label learning algorithm *BRkNN* is an adaptation of the single-label *lazy* k-Nearest Neighbor (*kNN*) algorithm to classify multi-label examples. To tackle directly the multi-label problem, the extensions *BRkNN-a* and *BRkNN-b* are proposed in [11]. Both extensions are based on a label confidence score, which is estimated for each label from the percentage of the k-Nearest Neighbors having this label. *BRkNN-a* classifies a new instance E using the labels in the multi-labels of the k-Nearest Neighbors which have a confidence score greater than 0.5, *i.e.*, labels included in the multi-labels of at least half of the k-Nearest Neighbors of E. If no label satisfies this condition, it outputs the label with the greatest confidence score. On the other hand, *BRkNN-b* classifies E with the $[s]$ (nearest integer of s) labels which have the greatest confidence score, where s is the average size of the multi-labels of the k-Nearest Neighbors of E.

MLkNN [17] is another multi-label lazy learning algorithm derived from the traditional *kNN* algorithm. Similar to *BRkNN*, *MLkNN* first identifies the k-Nearest Neighbors of the new instance as well as its multi-labels. After that, the maximum a posteriori (MAP) principle is used to determine the multi-label of the new instance, estimating the prior and the posterior probabilities from the training set.

In this work we use the *BRkNN-b* extension and *MLkNN*, both implemented in the *Mulan framework* [13], which provides different multi-label algorithms and is frequently used by the scientific community

2.2 Evaluation Measures

The multi-label dataset characteristics, as well as the quality of multi-label classifiers, are evaluated using different measures. To characterize a multi-label dataset D, two measures are frequently used: the Label Cardinality (LC), defined by Eq. 1, which is the average number of single-labels associated with each example, and the Label Density (LD), defined by Eq. 2, which is the normalized cardinality, *i.e.* $LD(D) = LC(D)/|L|$

$$LC(D) = \frac{1}{|D|}\sum_{i=1}^{|D|}|Y_i| \tag{1}$$

$$LD(D) = \frac{1}{|D|}\sum_{i=1}^{|D|}\frac{|Y_i|}{|L|}. \tag{2}$$

Although the evaluation of single-label classifers has only two possible outcomes, *correct* or *incorrect*, the evaluation of multi-label classifiers may also take into account *partially correct* classification. To this end, several multi-label evaluation measures, which use different criteria to evaluate the partial correction of multi-label classifiers, have been proposed. In [4], the connection among these criteria are established, showing that some of these criteria are uncorrelated or even negatively correlated. In other words, some

loss functions are essentially conflicting. Thus, several multi-label evaluation measures have been proposed, highlighting different aspects of this important characteristic of multi-label learning, as illustrated in [8].

The multi-label classification measures can be divided into three categories: (i) example-based; (ii) label-based; and (iii) ranking-based. A complete discussion of all these measures is out of the scope of this work, and can be found in [12]. In what follows, we briefly describe the example-based evaluation measures used in this work, which consider the difference between Y_i, the true multi-labels, and Z_i the predicted multi-labels. All these performance measures range in the interval [0..1].

Hamming Loss is defined by Eq. 3

$$Hamming\ Loss(H, D) = \frac{1}{N}\sum_{i=1}^{N}\frac{|Y_i \Delta Z_i|}{|L|}. \quad (3)$$

where Δ denotes the symmetric difference between two sets. *Hamming Loss* evaluates the frequency that labels in the multi-label are misclassified, *i.e.*, the example is associated to a wrong label or a label belonging to the true instance which is not predicted. Thus, unlike all other evaluation measures described in this section, for *Hamming-Loss*, the smaller the value, the better the multi-label classifier performance is. The other measures, defined next, were inspired in single-label classifier evaluation measures. For these measures, the greater the value, the better the multi-label classifier performance is.

Subset Accuracy is defined by Eq. 4

$$Subset\ Accuracy(H, D) = \frac{1}{N}\sum_{i=1}^{N} I(Z_i = Y_i). \quad (4)$$

where, $I(true) = 1$ and $I(false) = 0$. Thus, *Subset Accuracy* is a very strict evaluation measure as it requires an exact match of the predicted and the true multi-label.

Accuracy is defined by Eq. 5

$$Accuracy(H, D) = \frac{1}{N}\sum_{i=1}^{N}\frac{|Y_i \cap Z_i|}{|Y_i \cup Z_i|}. \quad (5)$$

which is the proportion of the predicted correct labels to the total number of labels in the predicted and the truth label set of an instance.

F-Measure, defined by Eq. 6, is the harmonic mean between the multi-label evaluation measures $Precision(H, D) = \frac{1}{N}\sum_{i=1}^{N}\frac{|Y_i \cap Z_i|}{|Z_i|}$ and $Recall(H, D) = \frac{1}{N}\sum_{i=1}^{N}\frac{|Y_i \cap Z_i|}{|Y_i|}$.

$$F\text{-}Measure(H,D) = \frac{1}{N}\sum_{i=1}^{N}\frac{2|Y_i \cap Z_i|}{|Z_i| + |Y_i|}. \quad (6)$$

2.3 *Baseline* Algorithm $General_B$

Finding a simple multi-label baseline classifier by only looking at the multi-labels is not as straightforward as in single-label, where the classification of a new instance has only two possible outcomes, and the error rate is often considered an important single objective to be achieved.

To this end, in [8] $General_B$ is proposed, a simple multi-label baseline classifier which is constructed by only looking at the multi-labels of the dataset D. $General_B$ does not focus on maximizing/minimizing any specific measure, and can be used to find general baselines for any bipartite multi-label evaluation measure. The rationale behind $General_B$ to find the predicted multi-label Z is simple. It consists of ranking the single-labels in L according to their individual relative frequencies on the multi-labels in D, and the σ most frequent single-labels are included in Z. As Z should have a reasonable number of single-labels such that it is not too strict (including too few single-labels) or too flexible (including too many single-labels), σ is defined as the closest integer value of the label cardinality of dataset D — Eq. 1.

In this work, we use $General_B$, which is also implemented using Mulan, as a *baseline* of the evaluation measures used to experimentally evaluate the proposed strategies.

3 Proposed Algorithms

BRkNN-b, as well as *MLkNN* use the k-Nearest Neighbors strategy to identify the set of examples B_k that will be taken into account to predict the multi-label of a new instance E, *i.e.*, $|B_k| = k$.

In this work, we propose the use of the following two other strategies, which have already been used for single-label learning [6], **but not for multi-label learning**, to identify the set of these examples:

1. Mutual Nearest Neighbors (set B_{mut})
2. Not Mutual Nearest Neighbors (set B_{nmut})

Let $B_k(E)$ be the set of k-Nearest Neighbors of a new instance E. The set $B_{mut}(E)$ consists of examples E_i which verify

$$E_i \in B_{mut}(E)\ if\ E_i \in B_k(E)\ and\ E \in B_k(E_i) \quad (7)$$

As this strategy requires that examples in $B_{mut}(E)$ not only have to be among the k-Nearest Neighbors of E, but the new instance E itself should also be among the k-Nearest Neighbors of the examples in $B_{mut}(E)$, the Mutual Nearest Neighbors strategy restricts the number of examples in $B_{mut}(E)$. In other words, $|B_{mut}(E)| \leq |B_k(E)|$. The *MLMUT* algorithm implements this strategy for multi-label learning, in which, case $B_{mut}(E) = 0$ then $|B_k(E)| = 1$ (default value).

On the other hand, the set of Not Mutual Nearest Neighbors B_{nmut}, consists of examples E_i which verify

$$E_i \in B_{nmut}(E)\ if\ E_i \in B_k(E)\ or\ E \in B_k(E_i) \quad (8)$$

then, the set $B_{nmut}(E)$ might contain more examples than $B_k(E)$, *i.e.*, $|B_{nmut}(E)| \geq |B_k(E)|$, as any example which is not in $B_k(E)$, but has the new instance E among its k-Nearest Neighbors, will be included in $B_{nmut}(E)$. The *MLnotMUT* algorithm implements this strategy for multi-label learning.

In both strategies, the multi-label of a new example $E = (\mathbf{x}, ?)$ is predicted as *BRkNN* does, but considering the examples in the set $B_{mut}(E)$ (algorithm *MLMUT*) or $B_{nmut}(E)$ (algorithm *MLnotMUT*). One advantage of adopting these strategies is that the number of selected examples used to classify a new instance E is a variable determined by the training set, whereas *BRkNN*, as well as *MLkNN*, use a fixed k.

The computational complexity of these algorithms is higher than the complexity of *BRkNN* and *MLkNN*, which is $O(N \times M)$, where N is the number of examples in the training set and M is the number of attributes in the dataset. For *MLMUT* the complexity is $O(k \times N \times M)$, and for *MLnotMUT* $O(N^2 \times M)$. Nevertheless, their complexity can be improved by using appropriate data structures [6].

We have implemented *MLMUT* and *MLnotMUT* in the *Mulan framework* [13]. The implementations of both algorithms are available at http://www.labic.icmc.usp.br/pub/mcmonard/Implementations/Multilabel/MUT-NOTMUT.zip. In what follows *BRkNN*, *MLkNN* (both already implemented in the *Mulan framework*), *MLMUT* and *MLnotMUT* are experimentally evaluated and compared.

All algorithms use the *Heterogeneous Euclidean-Overlap Metric* ($HEOM$), defined by Eq. 9, to quantify the similarity among pairs of examples. $HEOM$ uses the *overlap* metric for nominal attributes and the $range\ normalized\ diff$ distance for numeric attributes [14].

$$HEOM(x_{aj}, x_{bj}) = \begin{cases} overlap(x_{aj}, x_{bj}), & \text{if } X_j \text{ is nominal} \\ range\ normalized\ diff(x_{aj}, x_{bj}), & \text{if } X_j \text{ is numeric.} \end{cases} \quad (9)$$

where $overlap(x_{aj}, x_{bj}) = 0$ if $x_{aj} = x_{bj}$, and 1 otherwise. In all algorithms, the normalized Euclidean distance is used as *range normalized diff*(x_{aj}, x_{bj}).

4 Experimental Evaluation

Using the *Mulan framewok*, we compare the four lazy algorithms *BRkNN-b*, *MLkNN*, *MLMUT* and *MLnotMUT*. To this end, the algorithms were executed on 15 datasets. The reported results were obtained using 10-fold cross validation with paired folds, *i.e.*, using the same training and testing partitions. The evaluation measures defined in Section 2.2 were used to evaluate the performance of the classifiers.

Supplementary material related to the experiments, such as figures and tables with standard deviations, is available at http://www.labic.icmc.usp.br/pub/mcmonard/ExperimentalResults/MUT-NOTMUT.pdf. In what follows, we discuss the most relevant results.

4.1 Datasets

Table 2 summarizes the characteristics of the 15 datasets used in this work. For each dataset it shows: dataset name (Dataset); dataset domain (Domain); number of instances (N); number of features (M); feature type ($Type$); number of labels ($|L|$); label cardinality (LC); label density (LD); and the number of different multi-labels (#Distinct).

Except for datasets *7-Fapesp* and *10-Magtag5k*, the other datasets are available at the Mulan[1] and Meka[2] repositories. In particular, dataset *7-Fapesp*

[1] http://mulan.sourceforge.net/datasets.html

[2] http://meka.sourceforge.net/#datasets

Table 2 Dataset description

Dataset	Domain	N	M	$Type$	$\|L\|$	LC	LD	#Distinct
1-*Bibtex*	text	7395	1836	discrete	159	2.402	0.015	2856
2-*Cal500*	music	502	68	numeric	174	26.044	0.150	502
3-*Corel16k(sample 1)*	image	13766	500	discrete	153	2.859	0.019	4803
4-*Corel5k*	image	5000	499	discrete	374	3.522	0.009	3175
5-*Emotions*	music	593	72	numeric	6	1.869	0.311	27
6-*Enron*	text	1702	1001	discrete	53	3.378	0.064	753
7-*Fapesp*	text	332	8669	discrete	66	1.774	0.027	206
8-*Genbase**	biology	662	1185	discrete	27	1.252	0.046	32
9-*Llog-c**	text	1253	1004	discrete	75	1.375	0.018	303
10-*Magtag5k*	music	5260	68	numeric	136	4.839	0.036	4163
11-*Medical*	text	978	1449	discrete	45	1.245	0.028	94
12-*Ohsumed*	text	13929	1002	discrete	23	1.663	0.072	1147
13-*Scene*	image	2407	294	numeric	6	1.074	0.179	15
14-*Slashdot*	text	3782	1079	numeric	22	1.181	0.041	156
15-*Yeast*	biology	2417	103	numeric	14	4.237	0.303	198

was built by members of our research laboratory by extracting bag-of-related-words features from stories (examples) from a Brazilian scientific magazine[3], which in turn were annotated by the staff magazine according to the scientific branch(es) reported (labels) [10][4]. The *10-Magtag5k* dataset[5] is further described in [7]. Moreover, datasets *8-Genbase** and *9-Llog-c** are pre-processed versions of the publicly available datasets *genbase* and *language log*, respectively. The pre-processing consisted of removing one feature which uniquely identifies the examples in the dataset (*8-Genbase**), and the removal of unlabeled examples (*9-Llog-c**).

Table 3 shows, for each dataset, the lowest (Min) and the highest (Max) single-label frequencies, as well as the first ($1Q$), second (median Med) and third ($3Q$) quartiles. Figure 1 shows this information in the boxplot format.

Table 3 also shows the percentage of these values taken into account the number of examples (N) in the datasets. For instance, dataset 1-*Bibtex* consist of $N = 7395$ examples. As $Min = 51$, the least frequent label(s) participate almost in 1% of the multi-labels of the 7395 examples. On the other hand, as $Max = 1042$ the most frequent label(s) participate in 14% of these multi-labels. Moreover, as the third quartile ($3Q$) indicates 75% of the $|L| = 159$ single-labels, approximately 117 single-labels have a frequency less than 129.5, which represents approximately 2% of the examples in the dataset. This shows that the single-labels are highly unbalanced in this dataset. Beside 1-*Bibtex*, datasets 3-*Corel16k (sample 1)*, 4-*Corel5k* and 9-*Llog-c** show similar characteristics — Table 3.

4.2 Experimental Setting

Initially, using 10-folds cross-validation, we evaluated the parameter k for the lazy classification algorithms *BRkNN-b*, *MLkNN*, *MLMUT* and *MLnotMUT*, in order to find its best value for each algorithm. To this end, the influence of k on the learning performance of the algorithms was analyzed. This influence was estimated using the four evaluation measures described in Section 2.2. We varied the value of k in the range [1..27] with step 2 and in the range [29..99] with step 10. Whenever an evaluation measure was improving at the end of this process, we continue the process by further varying the value of k in the range [109..299] with step 10. Although these results might be overfitted as no validation set was used, the experimental comparison still holds as all the methods are equally overfitted. Figure 2 illustrates this procedure for dataset 13-*Scene* using as evaluation measures *Hamming-Loss* (lower values are better) and

[3] http://www.revistapesquisa.fapesp.br

[4] The dataset can be obtained from the authors.

[5] http://tl.di.fc.ul.pt/t/magtag5k.zip

Table 3 Statistics of single-label frequency in the multi-labels of the datasets

Dataset	*Min*		1*Q*		*Med*		3*Q*		*Max*	
1-*Bibtex*	51	(1%)	61	(1%)	82	(1%)	129.5	(2%)	1042	(14%)
2-*Cal500*	5	(1%)	15	(3%)	39.5	(8%)	109	(22%)	444	(88%)
3-*Corel16k (sample 1)*	25	(0%)	67	(0%)	115	(1%)	264	(2%)	3170	(23%)
4-*Corel5k*	1	(0%)	6	(0%)	15	(0%)	38.75	(1%)	1120	(22%)
5-*Emotions*	148	(25%)	166.5	(28%)	170.5	(29%)	185	(31%)	264	(45%)
6-*Enron*	1	(0%)	13	(1%)	26	(2%)	107	(6%)	913	(54%)
7-*Fapesp*	1	(0%)	4	(1%)	6	(2%)	11	(3%)	37	(11%)
8-*Genbase**	1	(0%)	3.5	(1%)	17	(3%)	45	(7%)	171	(26%)
9-*Llog-c**	1	(0%)	4	(0%)	11	(1%)	22.5	(2%)	171	(14%)
10-*Magtag5k*	9	(0%)	28.75	(1%)	86.5	(2%)	258.5	(5%)	1591	(30%)
11-*Medical*	1	(0%)	2	(0%)	8	(1%)	34	(3%)	266	(27%)
12-*Ohsumed*	135	(1%)	386.5	(3%)	712	(5%)	1220	(9%)	3952	(28%)
13-*Scene*	364	(15%)	404.5	(17%)	429	(18%)	432.5	(18%)	533	(22%)
14-*Slashdot*	0	(0%)	26	(1%)	179.5	(5%)	250.5	(7%)	584	(15%)
15-*Yeast*	34	(1%)	323.75	(13%)	659.5	(27%)	952.75	(39%)	1816	(75%)

F-Measure (higher values are better). The figures for all datasets and evaluation measures can be found in the supplementary material.

Recall that *BRkNN-b*, as well as *MLkNN* use the k-Nearest Neighbors strategy to identify the set of examples B_k that will be taken into account to predict the multi-label of a new instance E, *i.e.*, $|B_k| = k$ — Section 3. On the other hand, using the set of k-Nearest Neighbors B_k, the *MLMUT* algorithm constructs the set $B_{mut}(E)$, where $|B_{mut}(E)| \leq |B_k(E)|$,

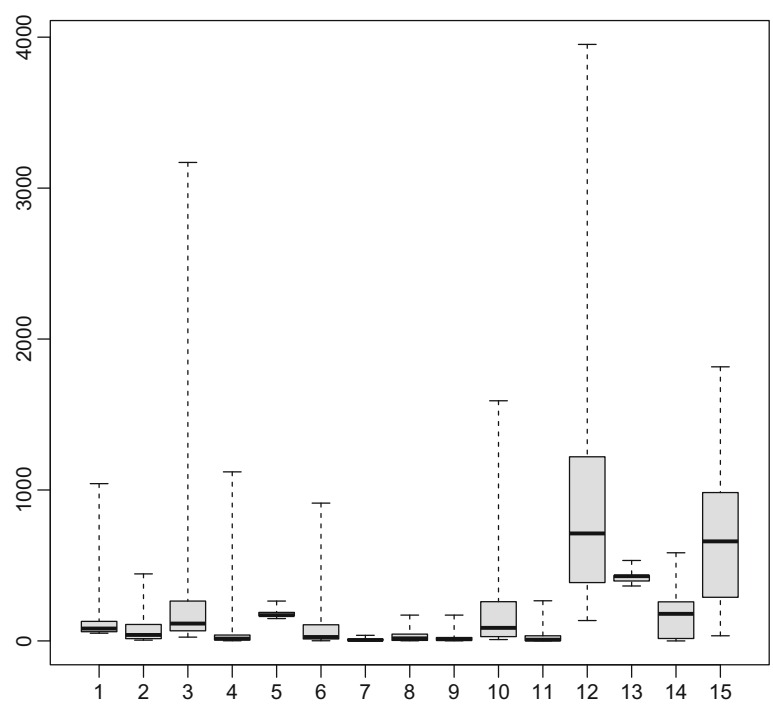

Fig. 1 Boxplots of the datasets single-label frequency

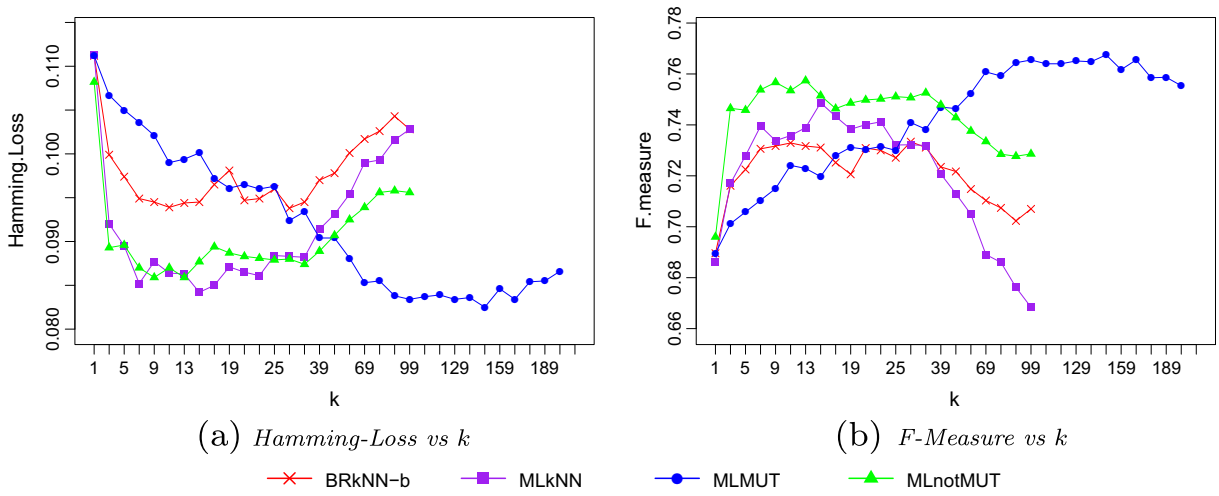

Fig. 2 *Hamming-Loss* and *F-Measure* evaluation measures for different values of k on dataset 13-Scene

while *MLnotMUT* constructs the set $B_{nmut}(E)$, where $|B_{nmut}(E)| \geq |B_k(E)|$, which will be respectively used to predict the multi-label of a new instance E. Thus, it could happen that $B_{mut}(E)$ is empty. In such a case, *MLMUT* inserts the nearest neighbor in $B_{mut}(E)$.

Table 4 shows, for each dataset, the best $k = |B_k|$ value found considering *Hamming-Loss* (HL), *F-Measure* ($F1$), *Accuracy* (AC) and *Subset Accuracy* (SA) as evaluation measures.

The best k value for *MLMUT* is generally higher than for *MLnotMUT*, which is an expected behavior

Table 4 Best number of neighbors used by each algorithm considering *F-Measure* (*F1*) and *Hamming-Loss* (*HL*) as evaluation measures

	$\|B_k\|$-*MLMUT*				$\|B_k\|$-*MLnotMUT*				$\|B_k\|$-*BRkNN-b*				$\|B_k\|-$*MLkNN*			
	HL	*F1*	*AC*	*SA*	*HL*	*F1*	*AC*	*SA*	*HL*	*F1*	*AC*	*SA*	*HL*	*F1*	*AC*	*SA*
1-*Bibtex*	99	119	49	49	1	1	1	1	1	1	1	1	7	7	7	7
2-*Cal500*	279	279	89	99	79	79	79	79	59	59	59	59	1	59	1	59
3-*Corel16k*	199	199	99	99	13	89	23	1	5	49	49	49	11	49	99	49
4-*Corel5k*	249	249	249	249	23	23	23	1	21	21	21	1	5	23	49	23
5-*Emotions*	15	39	15	15	9	15	9	9	9	15	15	9	21	21	13	9
6-*Enron*	149	149	149	119	13	13	9	1	11	11	1	1	13	13	5	13
7-*Fapesp*	89	89	89	89	1	29	29	7	1	29	29	29	29	29	29	29
8-*Genbase**	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
9-*Llog-c**	299	269	99	99	23	15	15	15	25	13	13	13	9	9	9	9
10-*Magtag5k*	99	99	99	1	15	17	15	1	17	17	17	1	19	19	13	13
11-*Medical*	9	9	9	9	7	7	7	7	3	3	3	5	21	5	3	3
12-*Ohsumed*	139	139	139	139	25	99	79	19	19	119	119	7	139	139	139	149
13-*Scene*	149	149	149	149	13	13	13	13	27	27	27	11	15	15	15	15
14-*Slashdot*	189	189	189	189	1	1	1	1	1	1	1	1	149	99	99	99
15-*Yeast*	69	69	69	1	29	29	29	59	21	21	21	1	9	13	9	13

as *MLMUT* is more restrictive to consider the neighbors in its equation to predict a multi-label. Recall that the set B_{mut} constructed by *MLMUT* to classify a new example might have less examples than B_k, while the set B_{nmut} constructed by *MLnotMUT* might have more examples than B_k. The best parameter obtained for the dataset *Yeast* and *SA* as evaluation measure was an exception. Most of the examples from this dataset were predicted y the default case, in which no examples were selected by *MLMUT* and the nearest neighbor is considered as the one to be used as prediction.

In what follows, these best k values are used to compare the classifiers constructed by the four algorithms.

4.3 Results

Table 5 shows the experimental results for each dataset considering the best k value found by *F-Measure* — Table 4 — and where the classifiers have been evaluated using *Hamming-Loss* and *F-Measure*. Each line of this table presents the average value of the classifier evaluation measure among ten folds, as well as the corresponding algorithm rankings in parenthesis. The best results are highlighted in bold. The last column, $General_B$, shows the evaluation measure value of the baseline classifier — Section 2.3. As can be observed, some results are not better than the ones obtained by $General_B$. These cases are underlined.

Considering the average ranking, *MLkNN* was the best multi-label learning algorithm when the performance is measured by *Hamming-Loss*. This algorithm achieved the best results for 14 out of 15 datasets (though there is a tie for the *Genbase* dataset). This is an expected behavior, since *MLkNN* tries to optimize *Hamming Loss* as a loss function [18]. *BRkNN-b* was clearly the one with the worst behavior.

However, when the predictive performance is measured by *F-Measure* (which was the same measure used to find the best k value), *MLkNN* obtained the worst average ranking. On the other hand, *MLnotMUT* obtained the best average ranking, followed by *MLMUT* and *BRkNN-b*.

Comparing the results from the learning algorithms to $General_B$, it can be observed that some classifiers could not outperform this simple baseline. Considering *Hamming-Loss* as the evaluation measure of the classifiers, *MLMUT*, *MLnotMUT* and *BRkNN-b* were worse than or equal to $General_B$ for five datasets: *3-Corel16k*, *4-Corel5k*, *7-Fapesp*, *8-Llog-c* and *12-Ohsumed-f*. On the other hand, when considering *F-Measure* to evaluate the classifiers (the same measure which has been used to find the best k value), *MLkNN* could not outperform $General_B$ on five datasets, while *MLnotMUT* and *BRkNN-b* fail to outperform $General_B$ in only one dataset. *MLMUT* outperformed the baseline for all datasets considering *F-Measure*.

Table 6 shows similar results to Table 5, but considering *Hamming-Loss* to find the best k values. As can be observed, these results are quite similar to the ones in Table 5, keeping the same order in the average rankings. Moreover, there are few differences in terms of the best k values obtained using *F-Measure* and *Hamming-Loss* — Table 4.

However, as we have used the same evaluation measures to evaluate the classifiers and to find the best k values (four measures), it is important to analyze all possible cases to see if the results are biased to the measure used to find the best k values.

To analyze whether there is a difference among the algorithms, we ran the Friedmans test for each evaluation measure with the null hypotheses that the performance of all algorithms are comparable. When the null-hypothesis is rejected by the Friedmans test, at 95% of confidence level, we can proceed with a post-hoc test to detect which differences among the methods are significant [5]. We ran the Nemenyis multiple comparison test, comparing all algorithms, where the Nemenyi test points out whether there is a significant difference between the algorithms involved in the experimental evaluation, whenever their average rank differs by at least the critical difference[6]. In all cases, the null-hypothesis was rejected by the Friedman test and the Nemenyi test was used.

The results of the Nemenyi's test can be represented in a simple diagram illustrated in the following figures, where the average ranks of the methods is plotted on the X-axis (lower average rank is better). The critical difference (CD) is shown above the X-axis, and

[6] It has been observed that sometimes the Friedmans test reports a significant difference, but the post-hoc test fails to detect it. This is due to the lower power of the post-hoc test. In this case, the only conclusion that can be drawn is that some algorithms do differ significantly.

Table 5 Experimental results considering the best k value of each multi-label learning algorithm according to the *F-Measure* evaluation measure

Name	*MLMUT*	*MLnotMUT*	*BRkNN-b*	*MLkNN*	$General_B$
Hamming-Loss					
1-*Bibtex*	0.023(3.0)	0.023(3.0)	0.023(3.0)	**0.014(1.0)**	0.025
2-*Cal500*	0.160(4.0)	0.158(2.0)	0.159(3.0)	**0.139(1.0)**	0.162
3-*Corel16k*	0.030(2.0)	0.032(3.5)	0.032(3.5)	**0.019(1.0)**	0.026
4-*Corel5k*	0.014(2.0)	0.015(3.5)	0.015(3.5)	**0.009(1.0)**	0.014
5-*Emotions*	0.208(2.5)	0.209(4.0)	0.208(2.5)	**0.192(1.0)**	0.330
6-*Enron*	0.064(4.0)	0.062(2.5)	0.062(2.5)	**0.052(1.0)**	0.066
7-*Fapesp*	0.040(2.0)	0.045(3.0)	0.046(4.0)	**0.026(1.0)**	0.039
8-*Genbase**	**0.001(2.5)**	**0.001(2.5)**	**0.001(2.5)**	**0.001(2.5)**	0.064
9-*Llog-c**	0.029(4.0)	0.027(2.0)	0.028(3.0)	**0.018(1.0)**	0.026
10-*Magtag5k*	0.044(3.0)	0.044(3.0)	0.044(3.0)	**0.033(1.0)**	0.053
11-*Medical*	0.018(3.5)	0.015(1.0)	0.018(3.5)	**0.016(2.0)**	0.038
12-*Ohsumed*	0.108(2.5)	0.108(2.5)	0.110(4.0)	**0.067(1.0)**	0.091
13-*Scene*	**0.083(1.0)**	0.086(3.0)	0.094(4.0)	0.084(2.0)	0.272
14-*Slashdot*	0.066(2.0)	0.072(3.0)	0.073(4.0)	**0.048(1.0)**	0.085
15-*Yeast*	0.200(3.0)	0.199(2.0)	0.202(4.0)	**0.193(1.0)**	0.255
average rank	2.733	2.700	3.333	1.233	
F-Measure					
1-*Bibtex*	0.253(2.0)	**0.261(1.0)**	0.242(3.0)	0.227(4.0)	0.099
2-*Cal500*	0.460(3.0)	**0.471(1.0)**	0.467(2.0)	0.327(4.0)	0.455
3-*Corel16k*	**0.197(1.0)**	0.167(3.0)	0.168(2.0)	0.021(4.0)	0.164
4-*Corel5k*	**0.242(1.0)**	0.179(2.0)	0.167(3.0)	0.037(4.0)	0.184
5-*Emotions*	0.657(2.5)	0.653(4.0)	0.657(2.5)	**0.669(1.0)**	0.296
6-*Enron*	0.462(2.0)	0.445(3.0)	0.433(4.0)	**0.485(1.0)**	0.417
7-*Fapesp*	**0.238(1.0)**	0.222(2.0)	0.201(3.0)	0.087(4.0)	0.072
8-*Genbase**	**0.992(1.5)**	0.990(3.0)	**0.992(1.5)**	0.987(4.0)	0.258
9-*Llog-c**	0.168(3.0)	**0.203(1.0)**	0.170(2.0)	0.052(4.0)	0.101
10-*Magtag5k*	0.384(2.0)	**0.388(1.0)**	0.377(3.0)	0.236(4.0)	0.189
11-*Medical*	0.657(4.0)	**0.711(1.0)**	0.658(3.0)	0.687(2.0)	0.232
12-*Ohsumed*	0.214(3.0)	**0.301(1.0)**	0.292(2.0)	0.159(4.0)	0.190
13-*Scene*	**0.768(1.0)**	0.757(2.0)	0.733(4.0)	0.749(3.0)	0.204
14-*Slashdot*	**0.360(1.0)**	0.299(2.0)	0.284(4.0)	0.297(3.0)	0.149
15-*Yeast*	0.654(2.5)	**0.655(1.0)**	0.654(2.5)	0.649(4.0)	0.550
average rank	2.033	1.867	2.767	3.333	

the secondary lines beneath the X-axis connect groups which are not significantly different.

Figure 3 shows these results when *F-Measure* was used to find the best k value. It can be seen that *MLkNN* outperformed the other algorithms with statistical significance when *Hamming-Loss* is considered to evaluate the classifiers, while it was outperformed with statistical significance by *MLMUT* and *MLnotMUT* when *F-Measure* is considered. No statistical differences were found by the Nemenyis test when *Accuracy* and *Subset-Accuracy* are used.

Figure 4 and 5 show the results when *Hamming-Loss* and *Accuracy* were used to find the best k value. In both cases, *MLkNN* also outperformed the

Table 6 Experimental results considering the best k value for each multi-label learning algorithm according to the *Hamming-Loss* evaluation measure

Name	*MLMUT*	*MLnotMUT*	*BRkNN-b*	*MLkNN*	$General_B$
Hamming-Loss					
1-*Bibtex*	0.023(3.0)	0.023(3.0)	0.023(3.0)	**0.014(1.0)**	0.025
2-*Cal500*	0.160(4.0)	0.158(2.0)	0.159(3.0)	**0.137(1.0)**	0.162
3-*Corel16k*	0.030(2.0)	0.031(3.5)	0.031(3.5)	**0.019(1.0)**	0.026
4-*Corel5k*	0.014(2.0)	0.015(3.5)	0.015(3.5)	**0.009(1.0)**	0.014
5-*Emotions*	0.203(2.0)	0.208(3.5)	0.208(3.5)	**0.192(1.0)**	0.330
6-*Enron*	0.064(4.0)	0.062(2.5)	0.062(2.5)	**0.052(1.0)**	0.066
7-*Fapesp*	0.040(2.0)	0.041(3.0)	0.042(4.0)	**0.026(1.0)**	0.039
8-*Genbase**	**0.001(2.5)**	**0.001(2.5)**	**0.001(2.5)**	**0.001(2.5)**	0.064
9-*Llog-c**	0.029(4.0)	0.027(2.5)	0.027(2.5)	**0.018(1.0)**	0.026
10-*Magtag5k*	0.044(3.0)	0.044(3.0)	0.044(3.0)	**0.033(1.0)**	0.053
11-*Medical*	0.018(3.5)	**0.015(1.5)**	0.018(3.5)	**0.015(1.5)**	0.038
12-*Ohsumed*	0.106(4.0)	0.101(2.0)	0.103(3.0)	**0.067(1.0)**	0.091
13-*Scene*	**0.083(1.0)**	0.086(3.0)	0.094(4.0)	0.084(2.0)	0.272
14-*Slashdot*	0.066(2.0)	0.072(3.0)	0.073(4.0)	**0.047(1.0)**	0.085
15-*Yeast*	0.200(3.0)	0.199(2.0)	0.202(4.0)	**0.192(1.0)**	0.255
average rank	2.800	2.700	3.300	1.200	
F-Measure					
1-*Bibtex*	0.250(2.0)	**0.261(1.0)**	0.242(3.0)	0.227(4.0)	0.099
2-*Cal500*	0.460(3.0)	**0.471(1.0)**	0.467(2.0)	0.307(4.0)	0.455
3-*Corel16k*	**0.197(1.0)**	0.157(2.0)	0.149(3.0)	0.013(4.0)	0.164
4-*Corel5k*	**0.242(1.0)**	0.179(2.0)	0.167(3.0)	0.034(4.0)	0.184
5-*Emotions*	0.652(2.0)	0.650(3.5)	0.650(3.5)	**0.669(1.0)**	0.296
6-*Enron*	0.462(2.0)	0.445(3.0)	0.433(4.0)	**0.485(1.0)**	0.417
7-*Fapesp*	**0.238(1.0)**	0.108(2.0)	0.086(4.0)	0.087(3.0)	0.072
8-*Genbase**	**0.992(1.5)**	0.990(3.0)	**0.992(1.5)**	0.987(4.0)	0.258
9-*Llog-c**	0.167(2.0)	**0.192(1.0)**	0.166(3.0)	0.052(4.0)	0.101
10-*Magtag5k*	0.384(2.0)	**0.388(1.0)**	0.377(3.0)	0.236(4.0)	0.189
11-*Medical*	0.657(4.0)	**0.711(1.0)**	0.658(3.0)	0.673(2.0)	0.232
12-*Ohsumed*	0.214(3.0)	**0.281(1.0)**	0.261(2.0)	0.159(4.0)	0.190
13-*Scene*	**0.768(1.0)**	0.757(2.0)	0.733(4.0)	0.749(3.0)	0.204
14-*Slashdot*	**0.360(1.0)**	0.299(2.0)	0.284(3.0)	0.213(4.0)	0.149
15-*Yeast*	0.654(2.5)	**0.655(1.0)**	0.654(2.5)	0.647(4.0)	0.550
average rank	1.933	1.767	2.967	3.333	

other three algorithms with statistical significance for *Hamming-Loss* and *MLMUT* and *MLnotMUT* are significantly better than *MLkNN* when *F-Measure* is used. Again, no statistical differences were found when *Accuracy* and *Subset-Accuracy* are used to evaluate the classifiers.

Figure 6 shows the results when *Subset-Accuracy* was used to find the best k values. As previously, considering statistical differences, results are similar, although in this case only *MLMUT* is significantly better than the other three algorithms when *F-Measure* is used.

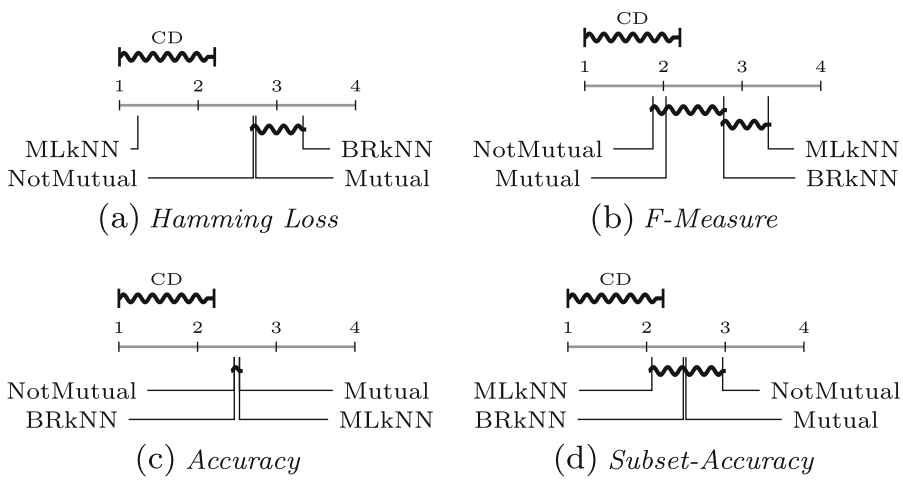

Fig. 3 Results considering the best k value according to *F-Measure*

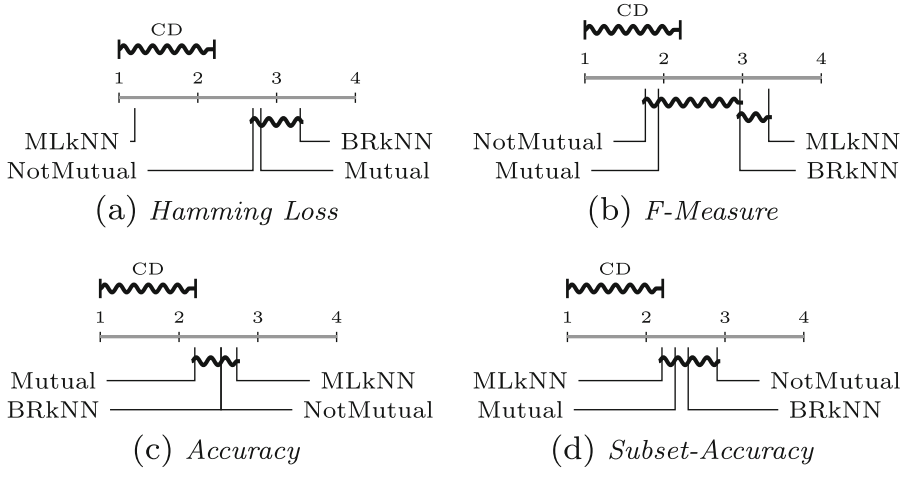

Fig. 4 Results considering the best k value according to *Hamming-Loss*

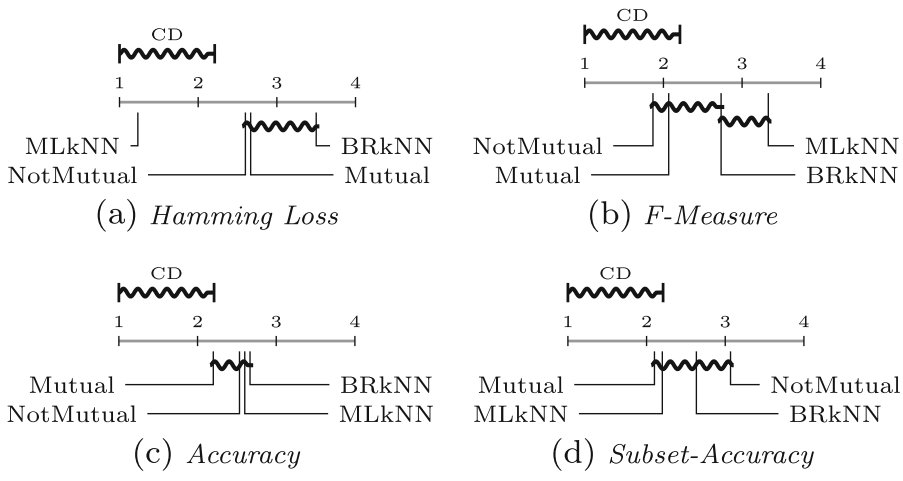

Fig. 5 Results considering the best k value according to *Accuracy*

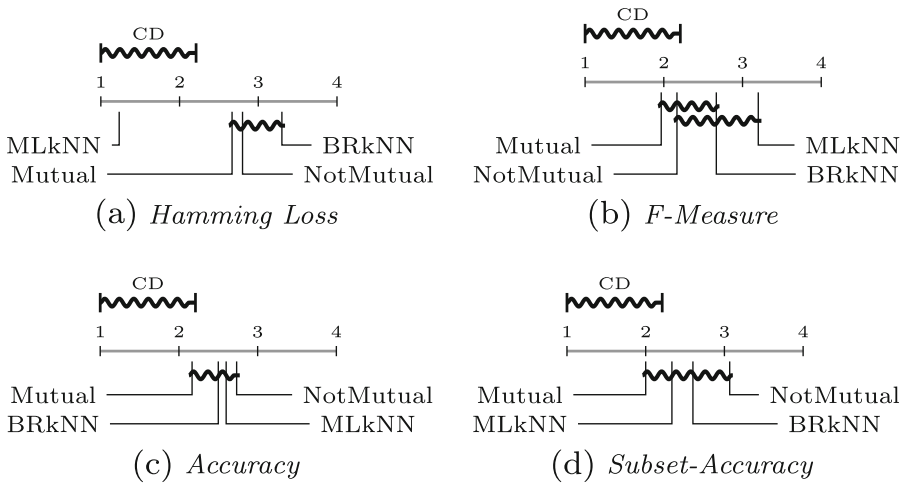

Fig. 6 Results considering the best k value according to *Subset-Accuracy*

Nevertheless, the results with statistical differences give strong evidence about the bias of the algorithms. Theoretical analysis have shown that *MLkNN* is biased to minimize *Hamming-Loss* [18]. Although without theoretical analysis, the experimental results show that *MLMUT* and *MLnotMUT* seems to be biased to *F-Measure* as they have obtained the best results for this measure. As with other multi-label measures, *Hamming-Loss* and *F-Measure* have different bias, *i.e.*, they take into account different characteristics for evaluating multi-label predictions. In many cases, classifiers that are well evaluated by *Hamming-Loss* are not well evaluated by *F-Measure*, and vice-versa.

The behavior on four datasets (1-*Bibtex*, 3-*Corel16k*, 4-*Corel5k* and 9-*Llog-c*) can be highlighted to show the previous mentioned characteristic. First of all, they are four of the most difficult multi-label datasets among the 15 considered in this work. All of them have high number of single labels ($|L|$) with low label density (LD) (see Table 2) and very unbalanced/infrequent single labels (see Table 3).

For these datasets the results obtained may be considered unsatisfactory as even *MLkNN* presents results not better than a specific baseline. As shown in Table 3, all single labels in these datasets are very skewed. In fact, no more than 2% of all examples are associated to 3/4 of the single labels from these datasets. In these cases, we could have good results if each single majority label is predicted for all examples. This simple classifier is a baseline which was used to compare results for *Hamming-Loss* in [8]. The value of this baseline is equal to LD (Table 2) when all labels have frequency lower than 50% of the total number of examples from the dataset.

This way, a baseline classifier that always predicts negative single labels (empty multi-labels) for all examples, would obtain results similar to the ones obtained by *MLkNN* for *Hamming-Loss*, although the other measures would be penalized (see Table 2). This might explain the best results for *Hamming-Loss* and the poor results for *F-Measure* obtained by *MLkNN*.

Table 7 shows, for each dataset, the best overall result obtained for each classifier considering the four evaluation measures, as well as the algorithm which has obtained the best result. The last line (Total) shows the number of times that an algorithm has the best overall result for the corresponding evaluation measure.

Observe that *BRkNN-b* does not appear in Table 7. As the difference among *BRkNN-b* and both, *MLMUT* and *MLnotMUT*, is only related to the strategy for finding the set of examples B which will be used by the voting criteria (the three algorithms use the same voting criteria) to predict the multi-label of a new example, we can conclude that both strategies are quite good for this voting criteria. *MLMUT* is winner in 19 cases and *MLnotMUT* 12 out of 60 cases. *MLkNN* also shows very good results, as is the winner in 28 out of 60 cases. As *MLkNN* uses the same strategy as *BRkNN-b* to find the set of examples B, differing only in the voting criteria, it would be interesting to analyze its behavior using the mutual and not-mutual nearest neighbor strategies.

Table 7 Best overall classifiers for each evaluation measure

Dataset	*HL*	*F*1	*AC*	*SA*
1-*Bibtex*	0.014/*MLkNN*	0.261/*MLnotMUT*	0.278/*MLMUT*	0.203/*MLMUT*
2-*Cal500*	0.137/*MLkNN*	0.471/*MLnotMUT*	0.632/*MLkNN*	0.311/*MLnotMUT*
3-*Corel16k*	0.019/*MLkNN*	0.197/*MLMUT*	0.183/*MLMUT*	0.124/*MLMUT*
4-*Corel5k*	0.009/*MLkNN*	0.242/*MLMUT*	0.170/*MLMUT*	0.027/*MLMUT*
5-*Emotions*	0.192/*MLkNN*	0.669/*MLkNN*	0.696/*MLkNN*	0.544/*MLkNN*
6-*Enron*	0.052/*MLkNN*	0.485/*MLkNN*	0.564/*MLkNN*	0.335/*MLkNN*
7-*Fapesp*	0.026/*MLkNN*	0.238/*MLMUT*	0.239/*MLMUT*	0.182/*MLMUT*
8-*Genbase**	0.001/*DRAW*	0.992/*MLMUT*	0.995/*MLkNN*	0.987/*MLkNN*
9-*Llog-c**	0.018/*MLkNN*	0.203/*MLnotMUT*	0.184/*MLnotMUT*	0.150/*MLnotMUT*
10-*Magtag5k*	0.033/*MLkNN*	0.388/*MLnotMUT*	0.417/*MLkNN*	0.154/*MLkNN*
11-*Medical*	0.015/*MLkNN*	0.711/*MLnotMUT*	0.679/*MLnotMUT*	0.603/*MLkNN*
12-*Ohsumed*	0.067/*MLkNN*	0.301/*MLnotMUT*	0.228/*MLnotMUT*	0.088/*MLkNN*
13-*Scene*	0.083/*MLMUT*	0.768/*MLMUT*	0.756/*MLMUT*	0.722/*MLMUT*
14-*Slashdot*	0.047/*MLkNN*	0.360/*MLMUT*	0.347/*MLMUT*	0.308/*MLMUT*
15-*Yeast*	0.192/*MLkNN*	0.655/*MLnotMUT*	0.725/*MLkNN*	0.519/*MLkNN*
Total	13/*MLkNN*	07/*MLnotMUT*	06/*MLMUT*	07/*MLkNN*
	01/*MLMUT*	06/*MLMUT*	06/*MLkNN*	06/*MLMUT*
	01/*DRAW*	02/*MLkNN*	03/*MLnotMUT*	02/*MLnotMUT*

Finally, experiments using the Binary Relevance (*BR*) approach were performed to compare its results to the ones obtained by the lazy algorithms. Table 8 shows results for *BR* using three different base learning algorithms: Naive Bayes (NB), J48 and SMO. All theses base learning algorithms are implemented in

Table 8 Results for the Binary Relevance multi-label classifier using three different base learning algorithms: NB, J48 and SMO

	HL			*F*1			*AC*			*SA*		
Dataset	*J*48	*NB*	*SMO*	*J*48	*NB*	*SMO*	*J*48	*NB*	*SMO*	*J*48	*NB*	*SMO*
1-*Bibtex*	**0,015**	0,087	0,016	0,371	0,262	**0,408**	0,305	0,187	**0,334**	0,142	0,059	**0,147**
2-*Cal500*	0,162	0,319	**0,137**	**0,338**	0,328	0,334	**0,207**	0,203	0,204	0,000	0,000	0,000
3-*Corel16k*	0,020	0,035	**0,019**	0,092	**0,210**	0,031	0,067	**0,141**	0,023	**0,011**	0,006	0,004
4-*Corel5k*	**0,010**	0,013	0,011	0,092	**0,218**	0,106	0,063	**0,149**	0,075	0,003	0,005	**0,007**
5-*Emotions*	0,247	0,252	**0,193**	0,557	**0,633**	0,597	0,462	**0,529**	0,517	0,184	0,206	**0,270**
6-*Enron*	**0,052**	0,218	0,060	**0,519**	0,305	0,518	0,406	0,193	**0,408**	0,098	0,000	**0,116**
7-*Fapesp*	0,032	0,027	**0,025**	**0,224**	0,005	0,165	**0,182**	0,004	0,147	0,078	0,003	**0,097**
8-*Genbase**	**0,001**	0,034	**0,001**	0,990	0,299	**0,993**	0,986	0,289	**0,990**	0,971	0,267	**0,980**
9-*Llog-c**	**0,021**	0,077	0,022	0,157	0,152	**0,187**	0,138	0,111	**0,165**	0,091	0,037	**0,105**
10-*Magtag5k*	0,042	0,237	**0,034**	**0,261**	0,178	0,108	**0,173**	0,101	0,070	**0,004**	0,000	**0,004**
11-*Medical*	**0,010**	0,026	**0,010**	0,773	0,403	**0,779**	0,744	0,367	**0,748**	0,654	0,265	**0,657**
12-*Ohsumed*	**0,063**	0,142	0,064	0,445	0,401	**0,462**	0,379	0,294	**0,394**	0,197	0,055	**0,209**
13-*Scene*	0,137	0,242	**0,106**	0,573	0,567	**0,620**	0,535	0,453	**0,596**	0,427	0,169	**0,526**
14-*Slashdot*	**0,042**	0,068	0,046	0,379	0,435	**0,494**	0,361	0,377	**0,460**	0,310	0,229	**0,363**
15-*Yeast*	0,245	0,303	**0,199**	0,564	0,538	**0,611**	0,440	0,420	**0,501**	0,068	0,095	**0,148**

Table 9 Number of datasets in which the best lazy classifier is equal or better than any other Binary Relevance classifier

Eval. Measure	*MLMUT*	*MLnotMUT*	*BRkNN-b*	*MLkNN*	*Lazy*
HL	1/15	2/15	1/15	9/15	11/15
$F1$	3/15	4/15	0/15	1/15	8/15
AC	4/15	1/15	0/15	6/15	11/15
SA	5/15	2/15	0/15	5/15	12/15

the Weka framework and were executed with their default values. The best result for each dataset and each evaluation measure are in bold. Results worse than the baseline $General_B$ are underlined.

A comparison among the best results of each lazy classifier and the best results of the *BR* classifiers are presented in Table 9. This table shows the number of datasets in which the performance of each lazy algorithm outperformed the best *BR* classifier, *i.e.*, it compares Table 7 to Table 8. The first column of Table 9 shows the evaluation measure considered in the comparison. The other columns represent each lazy algorithm, except the last one that represents all lazy algorithms together against the BR ones.

Considering HL as evaluation measure, *MLkNN* was equal or better than any other lazy and BR algorithm for 9 out of 15 datasets followed by *MLnotMUT* (2 out of 15 victories).

MLnotMUT was the best lazy algorithm when $F1$ is considered (4 out of 15), followed by *MLnotMUT* (3 out of 15).

Considering AC as evaluation measure, *MLkNN* was also the best option (6 out of 15), followed by *MLMUT* (4 out of 15).

Finally, considering SA as evaluation measure, *MLMUT* and *MLkNN* are also highlighted. Both where the best option for 5 out of 15 datasets. Thus, overall, *MLkNN* is the winner, followed by *MLMUT* and *MLnotMUT*, while *BRkNN* shows the worst results.

The last column shows that the lazy algorithms are more competitive than the BR ones. There is a clear advantage for the lazy algorithms for HL, AC and SA evaluation measures. Even for $F1$, the lazy algorithms are better than the BR ones for most of the cases.

Moreover, it can be observed that the great majority of the *BR* victories were obtained using text datasets. This behavior can be explained by the distance metric used in this work by the lazy algorithms, which is the Euclidean distance. However, the best choice for text dataset is the Cosseno distance [9], which would probably improve even more the results obtained by the Lazy approaches. Nevertheless, it is worth observing that the evaluation of the best k vales for the lazy algorithms might be overfitted. Thus, these results are biased in favor of the lazy algorithms.

5 Conclusions and Future Work

Lazy multi-label learning methods have become an important research topic within the multi-label community. These methods usually take the standard k-Nearest Neighbors of a new instance as input and predict its multi-label by following a voting criteria within the multi-labels of the examples in the set of k-Nearest Neighbors. In this work, we experimentally analyze two strategies to find the set of examples which will be taken into account by the voting criteria to predict the multi-label of a new instance: the Mutual and Not Mutual Nearest Neighbors strategies. Although these strategies have already been used and evaluated in graph single-label learning, it is not of our knowledge that they have been analyzed in the context of multi-label learning. Both strategies were experimentally evaluated on 15 datasets and the experimental results were compared with the ones obtained by two other lazy multi-label algorithms: *BRkNN* and *MLkNN*, where the latter is considered a state-of-the-art algorithm. These algorithms were also compared to the Binary Relevance approach.

The lazy algorithms presented better performance than the ones which use the Binary Relevance approach for most of cases. Compared to the other algorithms, *BRkNN* did not present good predictive performance for almost all the datasets. On the other hand, *MLkNN* presented the best predictive performance for *Hamming-Loss*, although it was outperformed by *MLMUT* and *MLnotMUT* when *F-Measure* is considered. As *Hamming-Loss* considers

equally negative (absence) and positive single label in the multi-labels, *MLkNN* would be more appropriate for applications which require this condition. Otherwise, when positive single labels (the label itself) are more important, *F-Measure* is a more suitable evaluation measure and the mutuality strategies are recommended.

As future work, we plan to implement and evaluate the mutuality strategies to choose the neighborhood of the *MLkNN*, as well as *IBLR* [1] algorithm, which also shows good results.

Acknowledgments This research was supported by the São Paulo Research Foundation (FAPESP), grants 2010/15992-0, 2011/02393-4, 2011/22749-8 and 2013/12191-5, as well as by the National Council for Scientific and Technological Development (CNPq), grant 151836/2013-2.

References

1. Cheng, W., Hüllermeier, E.: Combining instance-based learning and logistic regression for multilabel classification. Mach. Learn. **76**(2-3), 211–225 (2009)
2. Cherman, E.A., Spolaôr, N., Valverde-Rebaza, J., Monard, M.C.: Graph based algorithms for multi-label classification (in portuguesse). In: Encontro Nacional de Inteligńcia Computacional, pp. 1–12. SBC (2013)
3. Clare, A., King, R.D.: Knowledge discovery in multi-label phenotype data. In: European Conference on Principles of Data Mining and Knowledge Discovery, pp. 42–53. Springer-Verlag (2001)
4. Dembczynski, K., Waegeman, W., Cheng, W., Hüllermeier, E.: On label dependence and loss minimization in multi-label classification. Mach. Learn. **88**(1-2), 5–45 (2012)
5. Demsar, J.: Statistical comparisons of classifiers over multiple data sets. J. Mach. Learn. Res. **7**, 1–30 (2006)
6. Liu, H., Zhang, S., Zhao, J., Zhao, X., Mo, Y.: A new classification algorithm using mutual nearest neighbors. In: 9th International Conference on Grid and Cooperative Computing (GCC), pp. 52–57 (2010)
7. Marques, G., Domingues, M.A., Langlois, T., Gouyon, F.: Three current issues in music autotagging. In: Klapuri, A., Leider, C. (eds.): ISMIR, pp. 795–800. University of Miami (2011)
8. Metz, J., de Abreu, L.F.D., Cherman, E.A., Monard, M.C.: On the estimation of predictive evaluation measure baselines for multi-label learning. In: IBERAMIA, pp. 189–198 (2012)
9. Pang-Ning, T., Steinbach, M., Kumar, V., et al.: Introduction to Data Mining (2006)
10. Rossi, R.G., Rezende, S.O.: Building a topic hierarchy using the bag-of-related-words representation. In: Symposium on Document Engineering, pp. 195–204. ACM (2011)
11. Spyromitros, E., Tsoumakas, G., Vlahavas, I.: An empirical study of lazy multilabel classification algorithms. In: Hellenic conference on Artificial Intelligence, pp. 401–406. Springer-Verlag, Berlin, Heidelberg (2008)
12. Tsoumakas, G., Katakis, I., Vlahavas, I.P.: Mining multilabel data. In: Maimon, O., Rokach, L. (eds.): Data Mining and Knowledge Discovery Handbook, pp. 667–685, Springer (2010)
13. Tsoumakas, G., Spyromitros, E., Vilcek, J., Vlahavas, I.: Mulan: A java library for multi-label learning. J. Mach. Learn. Res. **12**, 2411–2414 (2011)
14. Wilson, D.R., Martinez, T.R.: Reduction techniques for exemplar-based learning algorithms. Mach. Learn. **38**(3), 257–286 (2000)
15. Younes, Z., Abdallah, F., Denoeux, T., Snoussi, H.: A dependent multilabel classification method derived from the k-nearest neighbor rule. EURASIP J. Adv. Signal Process. **2011**, 1–14 (2011)
16. Zhang, M.L.: Multilabel neural networks with applications to functional genomics and text categorization. IEEE Trans. Knowl. Data Eng. **18**(10), 1338–1351 (2006)
17. Zhang, M.L., Zhou, Z.H.: A k-nearest neighbor based algorithm for multi-label classification. IEEE International Conference on Granular Computing **2**, 718–721 (2005)
18. Zhang, M.L., Zhou, Z.H.: A review on multi-label learning algorithms. IEEE Trans. Knowl. Data Eng. **99**, 1 (2013)