# An Empirical Study of Lazy Multilabel Classification Algorithms

E. Spyromitros, G. Tsoumakas, and I. Vlahavas

Department of Informatics,
Aristotle University of Thessaloniki,
54124 Thessaloniki, Greece
{espyromi,greg,vlahavas}@csd.auth.gr

**Abstract.** Multilabel classification is a rapidly developing field of machine learning. Despite its short life, various methods for solving the task of multilabel classification have been proposed. In this paper we focus on a subset of these methods that adopt a lazy learning approach and are based on the traditional k-nearest neighbor ($k$NN) algorithm. Two are our main contributions. Firstly, we implement BR$k$NN, an adaptation of the $k$NN algorithm for multilabel classification that is conceptually equivalent to using the popular Binary Relevance problem transformation method in conjunction with the $k$NN algorithm, but much faster. We also identify two useful extensions of BR$k$NN that improve its overall predictive performance. Secondly, we compare this method against two other lazy multilabel classification methods, in order to determine the overall best performer. Experiments on different real-world multilabel datasets, using a variety of evaluation metrics, expose the advantages and limitations of each method with respect to specific dataset characteristics.

## 1 Introduction

Traditional *single-label* classification is concerned with learning from a set of examples that are associated with a single label $\lambda$ from a set of disjoint labels $L$, $|L| > 1$. If $|L| = 2$, then the learning task is called *binary* classification, while if $|L| > 2$, then it is called *multi-class* classification. In *multilabel* classification, each example is associated with a set of labels $Y \subseteq L$.

Multilabel classification methods can be categorized into two different groups [1]: i) *problem transformation* methods, and ii) *algorithm adaptation* methods. The first group of methods are algorithm independent. They transform the multilabel classification task into one or more single-label classification, regression or label ranking tasks. The second group of methods extend specific learning algorithms in order to handle multilabel data directly.

In this paper we focus on lazy multilabel classification methods of both categories that are based on the $k$ Nearest Neighbor ($k$NN) algorithm. Among the strong points of these methods is that their time complexity scales linearly with respect to $|L|$. Furthermore, their main computationally intensive operation is the calculation of nearest neighbors, which is actually independent of $|L|$.

J. Darzentas et al. (Eds.): SETN 2008, LNAI 5138, pp. 401–406, 2008.

Two are our main contributions in this work. Firstly, we implement BR$k$NN, an adaptation of the $k$NN algorithm for multilabel classification that is conceptually equivalent to using the popular Binary Relevance problem transformation method in conjunction with the $k$NN algorithm, but $|L|$ times faster. We also identify two useful extensions of BR$k$NN that improve its overall predictive performance. Secondly, we compare this method against two other lazy multilabel classification methods, in order to determine the overall best performer.

The rest of this paper is structured as follows. Section 2 presents the BR$k$NN method and its extensions. Section 3 presents the setup of the experimental work and Section 4 discusses the results. Finally, Section 5 concludes this work.

## 2 BR$k$NN and Extensions

Binary Relevance (BR) is the most widely-used problem transformation method for multilabel classification. It learns one binary classifier $h_\lambda : X \to \{\neg\lambda, \lambda\}$ for each different label $\lambda \in L$. BR transforms the original data set into $|L|$ data sets $D_\lambda$ that contain all examples of the original data set, labeled as $\lambda$ if the labels of the original example contained $\lambda$ and as $\neg\lambda$ otherwise. It is the same solution used in order to deal with a multi-class problem using a binary classifier, commonly referred to as one-against-all or one-versus-rest.

BR$k$NN is an adaptation of the $k$NN algorithm that is conceptually equivalent to using BR in conjunction with the $k$NN algorithm. Therefore, instead of implementing BR$k$NN, we could have utilized existing implementations of BR [2] and $k$NN [3]. However, the problem in pairing BR with $k$NN is that it will perform $|L|$ times the same process of calculating the $k$ nearest neighbors. To avoid these redundant time-intensive computations, BR$k$NN extends the $k$NN algorithm so that independent predictions are made for each label, following a single search of the $k$ nearest neighbors. This way BR$k$NN is $|L|$ times faster than BR plus $k$NN during testing, a fact that could be crucial in domains with a large set of labels and requirements for low response times. BR$k$NN was implemented within the MULAN multilabel classification software [2].

We propose two extensions to the basic BR$k$NN algorithm. Both are based on the calculation of *confidence* scores for each label $\lambda \in L$ from BR$k$NN. The confidence for a label can be easily obtained by considering the percentage of the $k$ nearest neighbors that include it. Formally, let $Y_j, j = 1 \ldots k$, be the label sets of the $k$ nearest neighbors of a new instance $x$. The confidence $c_\lambda$ of a label $\lambda \in L$ is equal to:

$$c_\lambda = \frac{1}{k} \sum_{j=1}^{k} I_{Y_j}(\lambda)$$

where $I_{Y_j} : L \to \{0, 1\}$ is a function that outputs 1 if its input label $\lambda$ belongs to set $Y_j$ and 0 otherwise, called *indicator function* in set theory.

The first extension of BR$k$NN, called BR$k$NN-a, checks whether BR$k$NN outputs the empty set, due to none of the labels $\lambda \in L$ being included in at least

half of the $k$ nearest neighbors. If this condition holds, then it outputs the label with the highest confidence. It so deals with a general disadvantage of BR, that has not been raised in the past: as each label is independently predicted in BR, there exists a possibility that the empty set is given as the overall output. We hypothesize that better results will be obtained through the proposed extension that outputs the most probable label when this phenomenon arises.

The second extension of BR$k$NN, called BR$k$NN-b calculates the average size $s$ of the label sets of the $k$ nearest neighbors at a first step, $s = \frac{1}{k}\sum_{j=1}^{k}|Y_j|$, and then outputs the $[s]$ (nearest integer of $s$) labels with the highest confidence.

## 3 Experimental Setup

### 3.1 Datasets

We experiment with 3 datasets from 3 different application domains: The biological dataset *yeast* [4] is concerned with protein function classification. The image dataset *scene* [5] is concerned with semantic indexing of still scenes. The music dataset *emotions* [6] is concerned with the classification of songs according to the emotions they evoke.

Table 1 shows certain standard statistics of these datasets, such as the number of examples in the train and test sets, the number of numeric and discrete attributes and the number of labels, along with multilabel data statistics, such as the number of distinct label subsets, the label cardinality and the label density [1]. Label cardinality is the average number of labels per example, while label density is the same number divided by $|L|$ .

**Table 1.** Standard and multilabel statistics for the data sets used in the experiments

| Dataset | Examples | Attributes | | Labels | Distinct | Label | Label |
|---|---|---|---|---|---|---|---|
| | | Numeric | Discrete | | Subsets | Cardinality | Density |
| scene | 2712 | 294 | 0 | 6 | 15 | 1.074 | 0.179 |
| emotions | 593 | 72 | 0 | 6 | 27 | 1.868 | 0.311 |
| yeast | 2417 | 103 | 0 | 14 | 198 | 4.327 | 0.302 |

### 3.2 Evaluation Methodology

We perform two sets of experiments. In the first one, we compare BR$k$NN to its extensions. In the second one, we compare the best version of BR$k$NN in each dataset to two other lazy multilabel classification methods, LP$k$NN and ML$k$NN, in order to make a final recommendation.

LP$k$NN is simply the pairing of the Label Powerset (LP) problem transformation method [2] with the $k$NN algorithm. LP considers each different subset of $L$ that appears in the training set as a different label of a single-label classification task. LP$k$NN has not been discussed in the related literature to the best of our knowledge.

ML$k$NN [7] is another adaptation of the $k$NN algorithm for multilabel data. What mainly differentiates this method from BR$k$NN is the use of prior and posterior probabilities which are directly estimated from the training set based on frequency counting. We implemented ML$k$NN in Java within the MULAN multilabel classification software [2] for the purposes of this study.

Each method was executed with a varying number of nearest neighbors. Specifically, the parameter $k$ ranged from 1 to 30. The performance of each method for each $k$ was evaluated using 10-fold cross-validation, in order to obtain an accurate performance estimate. In each fold, the following metrics were calculated [2], and eventually averaged over all folds:

- *Example-based*. Hamming loss, accuracy, F-measure and subset accuracy
- *Label-based*. Micro and macro version of F-measure

## 4 Experimental Results

### 4.1 Do the Proposed Extensions Improve BR$k$NN?

In this subsection we investigate whether BR$k$NN-a and BR$k$NN-b improve the performance of BR$k$NN. Table 2 reports the average performance of the three algorithms across all 30 values of the $k$ parameter for each dataset. It presents results for all evaluation metrics mentioned in Section 3.2. The best result on each metric and dataset is shown with bold typeface. The last line contains for each algorithm the number of metrics for which it achieves the best result, while within parentheses there is the number of metrics for which BR$k$NN-a and BR$k$NN-b are better than the base BR$k$NN algorithm.

The results show that both extensions outperform the base BR$k$NN method in more than half of the 6 metrics on all datasets. BR$k$NN-a outperforms BR$k$NN in 6, 5 and 5 out of the 6 metrics in the scene, emotions and yeast datasets respectively. BR$k$NN-b outperforms BR$k$NN in 6, 4 and 4 out of the 6 metrics in the scene, emotions and yeast datasets respectively. These two pieces of evidence strongly support that both BR$k$NN-a and BR$k$NN-b are beneficial extensions.

Studying the performance of the algorithms at each individual dataset, we notice that BR$k$NN-a dominates in scene and emotions, while BR$k$NN-b dominates

**Table 2.** Experimental results of BR$k$NN, BR$k$NN-a and BR$k$NN-b on all datasets, averaged for all $k$

| | scene | | | emotions | | | yeast | | |
|---|---|---|---|---|---|---|---|---|---|
| metric | base | ext-a | ext-b | base | ext-a | ext-b | base | ext-a | ext-b |
| Hamming loss | 0.0950 | **0.0938** | 0.0941 | **0.1976** | 0.1982 | 0.2175 | **0.1974** | 0.1975 | 0.2082 |
| accuracy | 0.6256 | **0.7226** | 0.7218 | 0.5215 | **0.5441** | 0.5430 | 0.5062 | 0.5080 | **0.5346** |
| F-measure | 0.6386 | **0.7392** | 0.7381 | 0.6275 | 0.6576 | **0.6590** | 0.5777 | 0.5795 | **0.6652** |
| subset accuracy | 0.5993 | **0.6889** | 0.6886 | 0.2895 | **0.2971** | 0.2759 | 0.1958 | **0.1959** | 0.1766 |
| micro F-measure | 0.6964 | **0.7296** | 0.7284 | 0.6499 | **0.6577** | 0.6509 | 0.6374 | 0.6380 | **0.6567** |
| macro F-measure | 0.6955 | **0.7363** | 0.7349 | 0.6224 | **0.6303** | 0.6294 | 0.3926 | 0.3931 | **0.4261** |
| #wins (#better) | 0 | 6 (6) | 0 (6) | 1 | 4 (5) | 1 (4) | 1 | 1 (5) | 4 (4) |

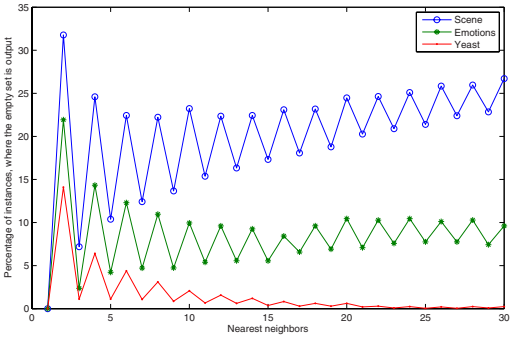


**Fig. 1.** Percentage of new instances, where BR$k$NN outputs the empty set (y axis), with respect to the number of nearest neighbors ($k$) (x axis) for all datasets

in yeast. This performance pattern correlates with the cardinality of the datasets, which is 1.074, 1.868 and 4.327 for the scene, emotions and yeast dataset respectively (see Table 3.1). Actually, it is natural for datasets of low cardinality, such as scene and emotions, to favor BR$k$NN-a over BR$k$NN, because the probability that the latter outputs the empty set increases in such datasets. This is clearly shown in Figure 1, which plots the percentage of the instances, where BR$k$NN outputs the empty set, for various values of the $k$ parameter. BR$k$NN-a deals with exactly this problem of BR$k$NN. On the other hand, BR$k$NN-b works better in datasets with larger cardinality, as it includes a mechanism to predict the number of true labels associated with a new instance.

### 4.2 Comparison of BR$k$NN, LP$k$NN and ML$k$NN

Table 3 reports the average performance of the three algorithms across all 30 values of the $k$ parameter for each dataset. It presents results for all evaluation metrics mentioned in Section 3.2. The best result on each metric and dataset is shown with bold typeface. The last line contains for each algorithm the number of metrics for which it achieves the best result.

**Table 3.** Experimental results of best version of BR$k$NN, LP$k$NN and ML$k$NN with normalization on all datasets, averaged for all $k$

| | scene | | | emotions | | | yeast | | |
|---|---|---|---|---|---|---|---|---|---|
| metric | BR-a | LP | ML | BR-a | LP | ML | BR-b | LP | ML |
| Hamming loss | 0.0938 | 0.0955 | **0.0884** | **0.1982** | 0.2094 | 0.2003 | 0.2082 | 0.2143 | **0.1950** |
| accuracy | **0.7226** | 0.7181 | 0.6720 | 0.5441 | **0.5600** | 0.5233 | **0.5346** | 0.5280 | 0.5105 |
| F-measure | **0.7392** | 0.7343 | 0.6944 | 0.6576 | **0.6662** | 0.6352 | **0.6652** | 0.6375 | 0.5823 |
| subset accuracy | **0.6889** | 0.6854 | 0.6272 | 0.2971 | **0.3287** | 0.2780 | 0.1766 | **0.2452** | 0.1780 |
| micro F-measure | 0.7296 | 0.7249 | **0.7316** | 0.6577 | **0.6649** | 0.6509 | **0.6567** | 0.6415 | 0.6422 |
| macro F-measure | **0.7363** | 0.7323 | 0.7341 | 0.6303 | **0.6505** | 0.6110 | 0.4261 | **0.4322** | 0.3701 |
| #wins | 4 | 0 | 2 | 1 | 5 | 0 | 3 | 2 | 1 |

We notice that BR$k$NN-a and LP$k$NN dominate in the scene and emotions datasets respectively, while in the yeast dataset there is no clear winner. However BR$k$NN-b performs better in most measures, followed by LP$k$NN and finally ML$k$NN. There is no apparent explanation on why LP$k$NN performs better in the emotions dataset. We notice in Table 3.1 that this dataset has the highest label density, while the scene dataset where LP$k$NN has the worst performance has the lowest label density. However we cannot safely argue that high density datasets lead to improved performance of the LP$k$NN algorithm.

## 5 Conclusions

This paper has studied how the $k$ Nearest Neighbor ($k$NN) algorithm is used for the classification of multilabel data. It presented BR$k$NN, an efficient implementation of the pairing of BR with $k$NN, along with two interesting extensions. Experimental results indicated that the proposed extensions are in the right direction. In addition, the paper compared experimentally BR$k$NN with two other methods (LP$k$NN and ML$k$NN) and reached to some interesting conclusions as to what kind of evaluation metrics and what kind of datasets are well-suited to the different methods.

## References

1. Tsoumakas, G., Katakis, I.: Multi-label classification: An overview. International Journal of Data Warehousing and Mining 3, 1–13 (2007)
2. Tsoumakas, G., Vlahavas, I.: Random k-labelsets: An ensemble method for multilabel classification. In: Kok, J.N., Koronacki, J., Lopez de Mantaras, R., Matwin, S., Mladenič, D., Skowron, A. (eds.) ECML 2007. LNCS (LNAI), vol. 4701, pp. 406–417. Springer, Heidelberg (2007)
3. Witten, I., Frank, E.: Data Mining: Practical Machine Learning Tools and Techniques, 2nd edn. Morgan Kaufmann, San Francisco (2005)
4. Elisseeff, A., Weston, J.: A kernel method for multi-labelled classification. In: Advances in Neural Information Processing Systems 14 (2002)
5. Boutell, M., Luo, J., Shen, X., Brown, C.: Learning multi-label scene classification. Pattern Recognition 37, 1757–1771 (2004)
6. Trohidis, K., Tsoumakas, G., Kalliris, G., Vlahavas, I.: Multilabel classification of music into emotions. In: Proc. 9th International Conference on Music Information Retrieval (ISMIR 2008), Philadelphia, PA, USA (2008)
7. Zhang, M.L., Zhou, Z.H.: A k-nearest neighbor based algorithm for multi-label classification. In: Proceedings of the 1st IEEE International Conference on Granular Computing, pp. 718–721 (2005)