# MULTI-LABEL CLASSIFICATION ALGORITHM DERIVED FROM *K*-NEAREST NEIGHBOR RULE WITH LABEL DEPENDENCIES

*Zoulficar Younes, Fahed Abdallah, and Thierry Denœux*

Heudiasyc, UMR CNRS 6599/ Université de Technologie de Compiègne
Centre de recherche Royallieu, BP 20529 60205, Compiègne Cedex, France
phone: + (0033) 344234485, fax: + (0033) 344234477, email: firstname.lastname@hds.utc.fr

## ABSTRACT

*In multi-label learning, each instance in the training set is associated with a set of labels, and the task is to output a label set whose size is unknown a priori for each unseen instance. Common approaches to multi-label classification learn independent classifiers for each category, and perform ranking or thresholding schemes in order to obtain multi-label classification. In this paper, we describe an original method for multi-label classification problems derived from a Bayesian version of the $K$-nearest neighbor (KNN), and taking into account the dependencies between labels. Experiments on benchmark datasets show the usefulness and the efficiency of the proposed method compared to other existing methods.*

## 1. INTRODUCTION

Traditional *single-label classification* assigns an object to exactly one class, from a set of $Q$ disjoint classes. *Multi-label classification* is the task of assigning an instance simultaneously to one or multiple classes. In other words, the target classes are not exclusive: an object may belong to an unrestricted set of classes instead of exactly one. This task makes the multi-label classifier more difficult to learn than the traditional single-label classifier.

Recently, multi-label classification methods have been increasingly required by modern applications where it is quite natural that some instances belong to several classes at the same time. Typical examples for multi-label problems are *text categorization*, *functional genomics* or *scene classification*. In text categorization, each document may belong to multiple topics, such as arts and humanities [5] [8] [4]; in gene functional analysis, each gene may be associated with a set of functional classes, such as energy, metabolism and cellular biogenesis [3]; in natural scene classification, each image may belong to several image types at the same time, such as sea and sunset[1].

Few algorithms have been proposed for multi-label problems, and common approaches include the *binary approach* and the *direct multi-class approach*. In the binary approach, each binary classifier is trained to separate one class from the others. The output of each binary classifier is compared to a threshold value in order to decide if the class label will be included among those assigned to the instance [9]. Binary classifiers tacitly assume that labels can be assigned independently: when one label provides information about another, the binary classifier fails to capture this effect. For example, if a news article belongs to category *music*, it is very likely that the article belongs to category *entertainment*. Thus, the binary approach has been criticized for dealing with asymmetric problems and not considering the correlations between labels. In a different manner, the multi-class approach considers straightforwardly instances with multiple labels as a new separate classes and builds models for them. In [10], the authors present a Bayesian multi-label $K$-nearest neighbor ($ML-KNN$) approach where, in order to assign a set of labels to a new instance, a decision is made separately for each label by taking into account the number of neighbors containing the label to be assigned. Thus, this method also fails to take into account the possible dependencies between labels. In this paper, and in the same spirit, we present a generalization of the $ML-KNN$ based approach to multi-label classification problems where the dependencies between classes are considered. Thus, the proposed method is called $DML-KNN$ for dependent multi-label $K$-nearest neighbor. The principle of the method is as follows. For each unseen instance, we identify its $KNN$s in the training set. According to the class membership of neighboring instances, a *global* maximum a posteriori (MAP) principle is used in order to assign a set of labels to the new unseen instance. Note that, in a different manner from $ML-KNN$, and in order to decide if one should include a label into the set of labels of the instance, the *global* MAP rule takes into account the numbers of all labels in the neighborhood instead of taking only the number of neighbors of the label to be assigned.

The remainder of the paper is organized as follows. Section 2 introduces notations and some basic evaluation criteria for multi-label problems. Section 3 describes the principle of $DML-KNN$ and its implementation. Section 4 presents the comparative experiments and discusses the results, and finally Section 5 summarizes this work and makes concluding remarks.

## 2. MULTI-LABEL CLASSIFICATION

### 2.1 Problem

In this paper we will generally adopt the same notations as in [10]. Let $\mathscr{X} = \mathbb{R}^d$ denote the domain of instances and let $\mathscr{Y} = \{1, 2, \ldots, Q\}$ be the finite set of labels. The multi-label classification problem can be formulated as follows. Given a set $T = \{(\mathbf{x}_1, \mathbf{Y}_1), (\mathbf{x}_2, \mathbf{Y}_2), \ldots, (\mathbf{x}_m, \mathbf{Y}_m)\}$ of m training examples, independently drawn from $\mathscr{X} \times 2^{\mathscr{Y}}$, and identically distributed, where $\mathbf{x}_i \in \mathscr{X}$ and $\mathbf{Y}_i \in 2^{\mathscr{Y}}$, the goal of the learning system is to output a multi-label classifier $h : \mathscr{X} \rightarrow 2^{\mathscr{Y}}$ which optimizes some pre-defined criterion or specific evaluation metric. As for classical classification problems, some algorithms used in multi-label learning are designed to output a real-valued function $f$ defined on $\mathscr{X} \times \mathscr{Y}$. Given an instance $\mathbf{x}_i$ and its associated label set $\mathbf{Y}_i$, a successful learning system produces a real-valued function $f$ that tend to assign larger values to labels in $\mathbf{Y}_i$ than those not in $\mathbf{Y}_i$. i.e., $f(\mathbf{x}_i, y_1) > f(\mathbf{x}_i, y_2)$ for any $y_1 \in \mathbf{Y}_i$ and $y_2 \notin \mathbf{Y}_i$. Note that the corresponding multi-label classifier $h(\cdot)$ can be derived from the function $f(\cdot,\cdot) : h(\mathbf{x}_i) = \{y \in \mathscr{Y} | f(\mathbf{x}_i, y) > t(\mathbf{x}_i)\}$, where $t(\cdot)$ is a threshold function that is usually set to be the zero constant function.

Some evaluation metrics evaluate the effectiveness of a multi-label algorithm using the ranking function constructed from the real-valued function $f(\cdot,\cdot)$. In ranking, the task is to order a set of labels $Q$, so that the topmost labels are likely to be more related with the new instance. Thus, a ranking function $rank_f(\cdot,\cdot)$ maps the outputs of $f(\mathbf{x}_i, y)$ for any $y \in \mathscr{Y}$ to $\{1, 2, \ldots, Q\}$ such that if $f(\mathbf{x}_i, y_1) > f(\mathbf{x}_i, y_2)$ then $rank_f(\mathbf{x}_i, y_1) < rank_f(\mathbf{x}_i, y_2)$.

In the next section we introduce some evaluation metrics based on the above definitions.

### 2.2 Evaluation metrics

The evaluation of multi-label learning system is different from that of single-label learning system. There exist a number of evaluation criteria that evaluate the performance of a multi-label learning system, given a set $S = \{(\mathbf{x}_1, \mathbf{Y}_1), \ldots, (\mathbf{x}_n, \mathbf{Y}_n)\}$ of $n$ test examples. We give hereafter some of the main evaluation measures used in the multi-label literature [10].

Let $\mathbf{Y}_i^* = h(\mathbf{x}_i)$ be the predicted label set for the test instance $\mathbf{x}_i$, while $\mathbf{Y}_i$ is the ground truth label set for $\mathbf{x}_i$.

**Hamming Loss**: Hamming Loss counts prediction errors (an incorrect label is predicted) and missing errors (a true label is not predicted).

$$hloss_S(h) = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{Q}\sum_{q=1}^{Q}(\delta(q \in \mathbf{Y}_i^* \wedge q \notin \mathbf{Y}_i) + \delta(q \notin \mathbf{Y}_i^* \wedge q \in \mathbf{Y}_i)). \quad (1)$$

Note that $\delta$ is a function that outputs 1 if its content is true and 0 otherwise. The smaller the value of $hloss_S(h)$, the better the performance. In the ideal case, $hloss_S(h) = 0$.

**One-error**: One-error evaluates how many times the top-ranked label is not in the true set of labels of the instance:

$$one\_err_S(f) = \frac{1}{n}\sum_{i=1}^{n}\delta([\arg\max_{y \in \mathscr{Y}} f(\mathbf{x}_i, y)] \notin \mathbf{Y}_i). \quad (2)$$

The smaller the value of $one - err_S(f)$, the better the performance. Note that, for single-label classification problems, the one-error is identical to ordinary classification error.

**Coverage**: The goal of the coverage measure is to assess the performance of a system for all the possible labels of a sample, not only the top-ranked label as in the case of the one-error. Coverage is defined as the average distance to cover all the proper labels assigned to a test example:

$$coverage_S(f) = \frac{1}{n}\sum_{i=1}^{n}\max_{y \in \mathbf{Y}_i} rank_f(\mathbf{x}_i, y) - 1. \quad (3)$$

Note that $coverage_S(f) = 0$ if the system does not make any classification errors.

**Ranking Loss**: It calculates the average fraction of label pairs that are reversely ordered for the instance:

$$rloss_S(f) = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{|\mathbf{Y}_i||\overline{\mathbf{Y}}_i|}|\{(y_1, y_2) \in \mathbf{Y}_i \times \overline{\mathbf{Y}}_i \mid f(\mathbf{x}_i, y_1) \leq f(\mathbf{x}_i, y_2)\}| \quad (4)$$

where $\overline{\mathbf{Y}}$ denotes the complement of $\mathbf{Y}$ in $\mathscr{Y}$. The smaller the value of $rloss_S(f)$, the better the performance.

**Average Precision**: Coverage and one-error are not complete measures for the multi-label case. We can achieve low coverage but, at the same time, suffer high one-error rates, and vice versa. To assess both aspects, non-interpolated average precision may be used. Usually this performance evaluation is used in information retrieval systems to measure the document ranking performance of a query. Adapted to multi-label classification problems, this metric evaluates the effectiveness of the label rankings. In other words, it measures the average fraction of labels ranked above a particular label $y \in \mathbf{Y}_i$ which actually are in $\mathbf{Y}_i$:

$$avgprec_S(f) = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{|\mathbf{Y}_i|}\sum_{y \in \mathbf{Y}_i}\frac{|\{y' \in \mathbf{Y}_i \mid rank_f(\mathbf{x}_i, y') \leq rank_f(\mathbf{x}_i, y)\}|}{rank_f(\mathbf{x}_i, y)}. \quad (5)$$

Note that $avgprec_S(f) = 1$ for any real-valued function $f$ that ranks perfectly the labels for all the test samples.

## 3. METHOD DESCRIPTION

In order to present the method, we will introduce suitable notations as in [10]. Given an instance $\mathbf{x}$ and its associated label set $\mathbf{Y} \in 2^{\mathscr{Y}}$, let $N(\mathbf{x})$ denote the set of the $K$ closest training examples of $\mathbf{x}$, $\mathbf{y_x}$ the $Q$-dimensional *category* vector of $\mathbf{x}$ whose $l$th component ($l \in \mathscr{Y}$) takes the value 1 if $l \in \mathbf{Y}$ and 0 otherwise, and $\mathbf{c_x}$ the $Q$-dimensional *membership counting* vector of $\mathbf{x}$ whose $l$th component indicates how many samples amongst the *KNN*s of $\mathbf{x}$ belong to the class $l$:

$$\mathbf{c_x}(l) = \sum_{\mathbf{z} \in N(\mathbf{x})} \mathbf{y_z}(l), \quad l \in \mathscr{Y}. \tag{6}$$

Note that the Euclidean metrics can be used in order to measure the distances between instances.

Let $\mathbf{t}$ be a test instance. Like in all *KNN* based methods, for each test instance $\mathbf{t}$, $N(\mathbf{t})$ should be firstly identified. Under the multi-label assumption, the counting vector $\mathbf{c_t}$ is computed. Let $H_1^l$ denote the event that $\mathbf{t}$ has label $l$, $H_0^l$ denote the event that $\mathbf{t}$ has not label $l$ and $E_j^q$ ($j \in \{0,1,\ldots,K\}$) denote the event that there are exactly $j$ instances in $N(\mathbf{t})$ which have label $q$. The goal is to determine the category vector $\mathbf{y_t}$ of the instance $\mathbf{t}$ using the following maximum a posteriori principle (MAP):

$$\begin{aligned} \mathbf{y_t}(l) &= \arg\max_{b\in\{0,1\}} P(H_b^l | \{E^q_{\mathbf{c_t}(q)}\}_{q\in\mathscr{Y}}) \\ &= \arg\max_{b\in\{0,1\}} P(H_b^l | E^l_{\mathbf{c_t}(l)}, \{E^q_{\mathbf{c_t}(q)}\}_{q\in\mathscr{Y}\setminus\{l\}}). \end{aligned} \tag{7}$$

In contrast to the MAP used in [10], we can see in the Equation (7) that the assignment of label $l$ to test instance $\mathbf{t}$ depends not only on the event that there are exactly $\mathbf{c_t}(l)$ instances having the label $l$ in $N(\mathbf{t})$, i.e. $E^l_{\mathbf{c_t}(l)}$, but also on $\{E^q_{\mathbf{c_t}(q)}\}_{q\in\mathscr{Y}\setminus\{l\}}$ which is the set of events stating that there are exactly $\mathbf{c_t}(q)$ instances having the label $q$ in $N(\mathbf{t})$, where $q \in \mathscr{Y}\setminus\{l\}$. Thus, it is obvious that the dependency between labels is taken into account in (7) since all the components of the counting vector $\mathbf{c_t}$ are involved in the assignment of the label $l$ which is not the case in [10]. As a consequence we call the method dependent multi-label $K$-nearest neighbor ($DML-KNN$).

Note that, depending on the counting vector $\mathbf{c_t}$, there can be at most $K^Q$ possible set of events $\{E^q_{\mathbf{c_t}(q)}\}_{q\in\mathscr{Y}}$. This in fact means that, in addition to the problem complexity, the estimation of (7) from a relatively small training set will not be accurate. To overcome this difficulty, we will adopt a fuzzy approximation for (7). This approximation is based on the event $F_j^q$, $j \in \{0,1,\ldots,K\}$, which is the event that there are *approximately* $j$ instances in $N(\mathbf{t})$ which have label $q$, i.e., $F_j^q$ denote the event that the number of instances in $N(\mathbf{t})$ that have label $q$ will be in the interval $[j-\delta_q; j+\delta_q]$, where $\delta_q \in \{0,\ldots,K\}$. As a consequence, we can derive a fuzzy MAP rule :

$$\mathbf{y_t}(l) = \arg\max_{b\in\{0,1\}} P(H_b^l | E^l_{\mathbf{c_t}(l)}, \{F^q_{\mathbf{c_t}(q)}\}_{q\in\mathscr{Y}\setminus\{l\}}). \tag{8}$$

The *margin* vector $\delta = (\delta_1,\ldots,\delta_Q)$ characterizes the dependency level between labels, e.g., taking $\delta_q = K$ where $q \in \mathscr{Y}\setminus\{l\}$ indicates that we suppose that the assignment of the label $l$ to the test instance $\mathbf{t}$ does not depend on label $q$. The smaller the value of $\delta_q$, the more dependency between $l$ and $q$. Furthermore, the $ML-KNN$ algorithm introduced in [10] is a particular case of the $DML-KNN$ where $\delta_q = K$ for all $q \in \mathscr{Y}\setminus\{l\}$.

Using the Bayes'rule, Equation (8) can be rewritten as:

$$\begin{aligned} \mathbf{y_t}(l) &= \arg\max_{b\in\{0,1\}} \frac{P(H_b^l)P(E^l_{\mathbf{c_t}(l)}, \{F^q_{\mathbf{c_t}(q)}\}_{q\in\mathscr{Y}\setminus\{l\}} | H_b^l)}{P(E^l_{\mathbf{c_t}(l)}, \{F^q_{\mathbf{c_t}(q)}\}_{q\in\mathscr{Y}\setminus\{l\}})} \\ &= \arg\max_{b\in\{0,1\}} P(H_b^l)P(E^l_{\mathbf{c_t}(l)}, \{F^q_{\mathbf{c_t}(q)}\}_{q\in\mathscr{Y}\setminus\{l\}} | H_b^l). \end{aligned} \tag{9}$$

To rank labels in $\mathscr{Y}$, a real-valued vector $\mathbf{r_t}$ can be calculated. Component $l$ of $\mathbf{r_t}$ is defined as the posterior probability $P(H_1^l | E^l_{\mathbf{c_t}(l)}, \{F^q_{\mathbf{c_t}(q)}\}_{q\in\mathscr{Y}\setminus\{l\}})$.

$$\begin{aligned} \mathbf{r_t}(l) &= P(H_1^l | E^l_{\mathbf{c_t}(l)}, \{F^q_{\mathbf{c_t}(q)}\}_{q\in\mathscr{Y}\setminus\{l\}}) \\ &= \frac{P(H_1^l)P(E^l_{\mathbf{c_t}(l)}, \{F^q_{\mathbf{c_t}(q)}\}_{q\in\mathscr{Y}\setminus\{l\}} | H_1^l)}{P(E^l_{\mathbf{c_t}(l)}, \{F^q_{\mathbf{c_t}(q)}\}_{q\in\mathscr{Y}\setminus\{l\}})} \\ &= \frac{P(H_1^l)P(E^l_{\mathbf{c_t}(l)}, \{F^q_{\mathbf{c_t}(q)}\}_{q\in\mathscr{Y}\setminus\{l\}} | H_1^l)}{\sum_{b\in\{0,1\}} P(H_b^l)P(E^l_{\mathbf{c_t}(l)}, \{F^q_{\mathbf{c_t}(q)}\}_{q\in\mathscr{Y}\setminus\{l\}} | H_b^l)}. \end{aligned} \tag{10}$$

In order to determine the category vector $\mathbf{y_t}$ and the real-valuated vector $\mathbf{r_t}$ of the instance $\mathbf{t}$, we must compute the prior probabilities $P(H_b^l)$ and the posterior probabilities $P(E^l_{\mathbf{c_t}(l)}, \{F^q_{\mathbf{c_t}(q)}\}_{q\in\mathscr{Y}\setminus\{l\}} | H_b^l)$, where $l \in \{1\cdots Q\}$ and $b \in \{0,1\}$. These probabilities can be estimated using the training set.

Figure 1 shows the pseudo code of the $DML-KNN$ algorithm. The values in $\delta$ are selected through cross-validation methods and given as input to the algorithm. The first time, the prior probabilities $P(H_b^q)$, $b = \{0,1\}$, for each class are calculated and the number of instances belonging to each label is counted (steps from 1 to 3). Note that $s$ is a smoothing parameter that controls the strength of uniform prior. Throughout the experiments, the Laplace smoothing is used which means that $s$ is set to 1. Given a test instance, the *KNN*s are identified and the membership counting vector $\mathbf{c_t}$ is determined

```
$[\mathbf{y_t}, \mathbf{r_t}] = DML-KNN(T, t, K, s, \delta)$

    %Computing the prior probabilities and the number of instances belonging to each class
1.  For $q = 1 \cdots Q$
2.    $P(H_1^q) = (s + \sum_{i=1}^m \mathbf{y}_{\mathbf{x}_i}(q))/(s \times 2 + m);\ P(H_0^q) = 1 - P(H_1^q);$
3.    $C[q] = \sum_{i=1}^m \mathbf{y}_{\mathbf{x}_i}(q);\ C'[q] = m - C[q];$
    EndFor
    %For each test instance t
4.  Identify $N(\mathbf{t})$ and $\mathbf{c_t}$
    %Counting the training instances whose membership counting vectors satisfy the constraints (11)
5.  For $q = 1 \cdots Q$
6.    $V[q] = 0;\ V'[q] = 0$
    EndFor
7.  For $i = 1 \cdots m$
8.    Identify $N(\mathbf{x}_i)$ and $\mathbf{c}_{\mathbf{x}_i}$
9.    If $\mathbf{c_t}(q) - \delta_q \leq \mathbf{c}_{\mathbf{x}_i}(q) \leq \mathbf{c_t}(q) + \delta_q, \forall\, q \in \mathscr{Y}$ Then
10.     For $q = 1 \cdots Q$
11.       If $\mathbf{c}_{\mathbf{x}_i}(q) == \mathbf{c_t}(q)$ Then
12.         If $\mathbf{y}_{\mathbf{x}_i}(q) == 1$ Then $V[q] = V[q] + 1;$
                                  Else $V'[q] = V'[q] + 1;$
      EndFor
    EndFor
    %Computing $\mathbf{y_t}$ and $\mathbf{r_t}$
13. For $l = 1 \cdots Q$
14.   $P(E^l_{\mathbf{c_t}(l)}, \{F^q_{\mathbf{c_t}(q)}\}_{q \in \mathscr{Y} \setminus \{l\}} | H_1^l) = (s + V[l])/(s \times Q + C[l]);$
15.   $P(E^l_{\mathbf{c_t}(l)}, \{F^q_{\mathbf{c_t}(q)}\}_{q \in \mathscr{Y} \setminus \{l\}} | H_0^l) = (s + V'[l])/(s \times Q + C'[l]);$
16.   $\mathbf{y_t}(l) = \arg\max_{b \in \{0,1\}} P(H_b^l) P(E^l_{\mathbf{c_t}(l)}, \{F^q_{\mathbf{c_t}(q)}\}_{q \in \mathscr{Y} \setminus \{l\}} | H_b^l)$
17.   $\mathbf{r_t}(l) = \frac{P(H_1^l) P(E^l_{\mathbf{c_t}(l)}, \{F^q_{\mathbf{c_t}(q)}\}_{q \in \mathscr{Y} \setminus \{l\}} | H_1^l)}{\sum_{b \in \{0,1\}} P(H_b^l) P(E^l_{\mathbf{c_t}(l)}, \{F^q_{\mathbf{c_t}(q)}\}_{q \in \mathscr{Y} \setminus \{l\}} | H_b^l)}$
    EndFor
```

Figure 1: *DML − KNN* algorithm.

(step 4). In order to assign or not the label $l$ to the test instance $\mathbf{t}$, we should estimate also the posterior probability $P(E^l_{\mathbf{c_t}(l)}, \{F^q_{\mathbf{c_t}(q)}\}_{q \in \mathscr{Y} \setminus \{l\}} | H_b^l)$ using training samples whose membership counting vectors satisfy the constraints

$$\begin{cases} \mathbf{c}_{\mathbf{x}_i}(l) = \mathbf{c_t}(l) \\ \mathbf{c_t}(q) - \delta_q \leq \mathbf{c}_{\mathbf{x}_i}(q) \leq \mathbf{c_t}(q) + \delta_q,\ q \in \mathscr{Y} \setminus \{l\}. \end{cases} \tag{11}$$

This is illustrated in steps from 5 to 12. The number of instances from the training set which verify these constraints, and containing the label $l$ in their sets of labels is stored in $V[l]$. The number of remaining instances which verify the previous constraints and not having class $l$ in their sets of labels is stored in $V'[l]$. Finally the category vector $\mathbf{y_t}$ and the real-valued vector $\mathbf{r_t}$ to rank labels in $\mathscr{Y}$ are calculated (steps from 13 to 17).

## 4. EXPERIMENTS

We report in this section the results on two datasets collected from real world applications. These datasets concern the problem of *yeast gene functional analysis*, and *natural scene classification*. Using these datasets, authors in [10] show the superiority of the *ML − KNN* over other existing methods, namely *BoosTEXTER* [8], *ADTBOOST.MH* [2] [7] and *RANK − SVM* [3]. In the following, we report a comparative study between the *DML − KNN* and the *ML − KNN* methods.

### 4.1 Yeast data set

The gene dataset contains 2417 yeast genes, each represented by a 103-dimensional feature vector [3] [6]. Functional classes of many genes have been already determined and classified into a hierarchy of functions. Each gene may have several functions at the same time. Thus, the problem of Yeast classification is a multi-label problem with 14 labels. In the available training set, the maximum number of labels assigned to an instance is 11 and the average number of labels is $4.25 \pm 1.57$. As in [10], ten-fold cross-validation was performed on the Yeast dataset.

Table 1 reports the experimental results of *DML − KNN* and *ML − KNN* methods for $K$ = 8, 9, and 10. For the *DML − KNN* method, the margin values in $\delta$ was determined via cross-validation methods. As can be seen in Table 1, for the different values of $K$, *DML − KNN* outperform *ML − KNN* according to all criteria . The values following " ± " gives the standard deviations and the best result for each criterion is shown in bold face. It can be seen that taking into account the dependencies between labels has improved the performance of the *KNN* based multi-label learning rule.

### 4.2 Natural scene classification

This dataset contains 2000 natural scene images belonging to the classes *sea, sunset, trees, desert and mountains*. For each image, spatial color moments are used as features. We divide the images into 49 blocks using $7 \times 7$ grid and we compute the mean and the variance of each band, corresponding to a low-resolution image and to computationally inexpensive texture features, respectively [1]. Each image is then transformed into a $49 \times 3 \times 2 = 294$-dimensional feature vector. In the available training set, the maximum number of labels assigned to an instance is 3 and the average number of labels is 1.24. As in [10], ten-fold cross-validation was performed on the natural scene dataset.

Table 2 reports the experimental results of *DML − KNN* and *ML − KNN* methods. $\delta$ was determined via cross-validation methods. As can be seen in Table 1, *DML − KNN* outperform *ML − KNN*, but the difference between the performances is not very significant. This is due to the fact that the average number of labels for the instances in the available set is 1.24 (80% of the data have a single label). In other words, most natural scene images in the training set belong to only one class making it more dif-

|  | $DML-KNN$ |  |  | $ML-KNN$ |  |  |
|---|---|---|---|---|---|---|
| $K$ | 8 | 9 | 10 | 8 | 9 | 10 |
| Hamming loss | **0.193 ± 0.006** | **0.195 ± 0.007** | **0.193 ± 0.008** | 0.196 ± 0.010 | 0.196 ± 0.009 | 0.195 ± 0.009 |
| One-error | **0.224 ± 0.017** | **0.226 ± 0.018** | **0.225 ± 0.022** | 0.235 ± 0.033 | 0.231 ± 0.040 | 0.231 ± 0.033 |
| Coverage | **6.280 ± 0.130** | **6.285 ± 0.201** | **6.269 ± 0.148** | 6.294 ± 0.240 | 6.298 ± 0.242 | 6.282 ± 0.248 |
| Ranking loss | **0.168 ± 0.005** | **0.167 ± 0.011** | **0.167 ± 0.008** | 0.172 ± 0.017 | 0.169 ± 0.016 | 0.168 ± 0.017 |
| Average precision | **0.765 ± 0.007** | **0.765 ± 0.013** | **0.766 ± 0.011** | 0.756 ± 0.022 | 0.759 ± 0.023 | 0.761 ± 0.022 |

Table 1: Experimental results (mean±std) on the Yeast dataset for $DML-KNN$ and $ML-KNN$ methods.

|  | $DML-KNN$ |  |  | $ML-KNN$ |  |  |
|---|---|---|---|---|---|---|
| $K$ | 8 | 9 | 10 | 8 | 9 | 10 |
| Hamming loss | **0.191 ± 0.011** | **0.193 ± 0.007** | **0.192 ± 0.010** | 0.198 ± 0.012 | 0.199 ± 0.010 | 0.199 ± 0.011 |
| One-error | **0.393 ± 0.043** | **0.411 ± 0.022** | **0.408 ± 0.021** | 0.393 ± 0.045 | 0.412 ± 0.024 | 0.408 ± 0.029 |
| Coverage | **1.140 ± 0.123** | **1.153 ± 0.060** | **1.159 ± 0.071** | 1.148 ± 0.124 | 1.157 ± 0.061 | 1.166 ± 0.074 |
| Ranking loss | **0.217 ± 0.028** | **0.222 ± 0.010** | **0.223 ± 0.014** | 0.219 ± 0.028 | 0.224 ± 0.011 | 0.224 ± 0.016 |
| Average precision | **0.744 ± 0.027** | **0.736 ± 0.009** | **0.737 ± 0.013** | 0.743 ± 0.028 | 0.735 ± 0.013 | 0.734 ± 0.016 |

Table 2: Experimental results (mean±std) on the scene dataset for $DML-KNN$ and $ML-KNN$ methods.

ficult to capture the label dependencies from the training set.

*Remark*: The results of the $ML-KNN$ method on the two datasets shown in this paper are not identical to the results shown in [10] because, when performing ten-fold cross-validation, the outputs of a classifier will depend on the arrangement of the elements of the dataset.

## 5. CONCLUSION

In this paper, we proposed an original multi-label learning algorithm derived from the $KNN$ rule, where the dependencies between labels are taken into account. This method generalizes the $ML-KNN$ algorithm developed for multi-label problems given in [10]. The proposed extension is particularly useful in practical situations where the data are *significantly* multi-labelled. The experiment results demonstrate this fact and prove the usefulness and the effectiveness of the proposed method as compared to the $ML-KNN$ method.

## REFERENCES

[1] M. R. Boutell, J. Luo, X. Shen, and C. M. Brown, "Learning multi-label scene classification," *Pattern Recognition*, vol. 37, no. 9, pp. 1757-1771, 2004.

[2] F. D. Comité, R. Gilleron, and M. Tommasi, "learning multi-label alternating decison tree from texts and data," in *Proc. of the 3rd International Conference on Machine Learning and Data Mining in Pattern Recognition*, Leipzig, Germany, July 5-7. 2003, pp. 35-49.

[3] A. Elisseeff and J. Weston, "Kernel methods for multi-labelled classification and categorical regression problems," *Advances in Neural Information Processing Systems*, vol. 14, pp. 681-687, 2002.

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

[5] A. McCallum, "Multi-Label Text Classification with a Mixture Model Trained by EM," in *Proc. of the AAAI'99 Workshop on Text Learning*, Orlando, Florida, July 18-22 1999.

[6] P. Pavlidis, J. Weston, J. Cai, and W. N. Grundy, "Combining microarray expression data and phylogenetic profiles to learn functional categories using support vector machines," in *Proc. of the Fifth Annual International Conference on Computational Biology*, Montréal, Canada, 2001, pp. 242-248.

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

[8] R. E. Schapire and Y. Singer, "BoosTexter: A Boosting-based System for Text Categorization," *Machine Learning*, vol. 39, no. 2-3, pp. 135-168, 2000.

[9] Y. Yang, "An evaluation of statistical approaches to text categorisation," *Information Retrieval*, vol. 1, no. 1-2, pp. 69-90, 1999.

[10] M.-L. Zhang and Z.-H. Zhou, "ML-KNN:A lazy learning approach to multi-label learning," *Pattern Recognition*, vol. 40, no.7, pp. 2038-3048, 2007.