Expert Systems with Applications 41 (2014) 2989–3004

Contents lists available at ScienceDirect

Expert Systems with Applications

journal homepage: www.elsevier.com/locate/eswa

Multi-label classification by exploiting label correlations

Ying Yu [a,b,c,e,*], Witold Pedrycz [b,d], Duoqian Miao [a,c]

[a] Department of Computer Science and Technology, Tongji University, Shanghai 201804, PR China
[b] Department of Electrical and Computer Engineering, University of Alberta, Edmonton, AB T6G 2G7, Canada
[c] Key Laboratory of Embedded System and Service Computing, Ministry of Education, Tongji University, Shanghai 201804, PR China
[d] System Research Institute, Polish Academy of Sciences, Warsaw, Poland
[e] School of Software, Jiangxi Agricultural University, Nanchang 330013, PR China

ARTICLE INFO

Keywords:
Multi-label classification
Rough sets
Uncertainty
Correlation

ABSTRACT

Nowadays, multi-label classification methods are of increasing interest in the areas such as text categorization, image annotation and protein function classification. Due to the correlation among the labels, traditional single-label classification methods are not directly applicable to the multi-label classification problem. This paper presents two novel multi-label classification algorithms based on the variable precision neighborhood rough sets, called multi-label classification using rough sets (MLRS) and MLRS using local correlation (MLRS-LC). The proposed algorithms consider two important factors that affect the accuracy of prediction, namely the correlation among the labels and the uncertainty that exists within the mapping between the feature space and the label space. MLRS provides a global view at the label correlation while MLRS-LC deals with the label correlation at the local level. Given a new instance, MLRS determines its location and then computes the probabilities of labels according to its location. The MLRS-LC first finds out its topic and then the probabilities of new instance belonging to each class is calculated in related topic. A series of experiments reported for seven multi-label datasets show that MLRS and MLRS-LC achieve promising performance when compared with some well-known multi-label learning algorithms.

1. Introduction

Nowadays, multi-label classification problem (Tsoumakas, Katakis, & Vlahavas, 2010) has received an increased attention finding applicability in various applications. For example, in text categorization, a document may belong to multiple classes simultaneously (Jiang, Tsai, & Lee, 2012). In the video indexing domain, each audio clip can have several different labels (Snoek et al., 2006). In functional genomics, a gene may have multiple functions (Vens, Struyf, Schietgat, et al., 2008). In automatic image annotation, a scene may be associated with several concepts as well (Yu, Pedrycz, & Miao, 2013). In all the cases identified above, each instance is associated with multiple labels and the classes encountered in the problem are not mutually exclusive but may overlap. This situation is different from the traditional single-label classification (i.e, multi-class) where an instance is only associated with a single label and by definition, the classes are mutually exclusive (see Fig. 1).

In what follows, we provide a formal definition of the multi-label classification problem.

Definition 1. Let $X \subset R^d$ denote a d-dimensions input domain of instances and let $Y = \{l_1, l_2, ..., l_m\}$ be an output domain of possible labels. Given a training set $T = \{(x_i, y_i) | 1 \leqslant i \leqslant n, x_i \in X, y_i \subset Y\}$, the goal of learning system is to form a multi-label classifier $f: X \to 2^Y$ which optimizes some specific evaluation metric. For a testing instance $x \in X$, its associated label set $y \subset Y$ is expressed with the use of f.

According to Definition 1, it is clear that a single-label classification is a particular case of the multi-label classification. When the number of labels of instances is equal to 1 ($|y_i| = 1$), the multi-label classification problem transforms into a single-label classification problem.

Due to the existence of relevance and co-occurrence among labels in multi-label classification, single-label classification methods cannot be used to directly address the multi-label classification problem (Streich & Buhmann, 2008; Tsoumakas et al., 2010). A large body of research has been carried out to explore effective and efficient multi-label classification approaches which are generally grouped into two main categories: problem transformation methods and algorithm adaptation methods (Tsoumakas et al., 2010). However, most of these methods neglect a fact that there exists some uncertainty during the process of classification. The uncertainty is caused by some reasons. First, due to the finite number of training instances, we cannot acquire an exact distribution of each

* Corresponding author at: Department of Computer Science and Technology, Tongji University, Shanghai 201804, PR China.
E-mail addresses: yuyingtongzhi@hotmail.com, yuyingjx@163.com (Y. Yu).

0957-4174/$ - see front matter © 2013 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.eswa.2013.10.030

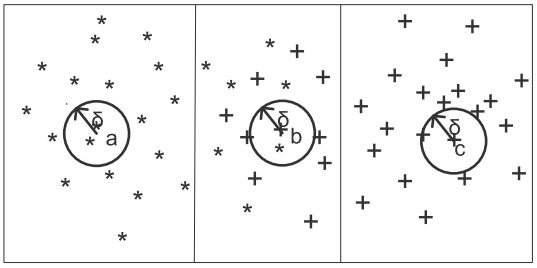


Fig. 1. Single-label example with two classes.

class. Second, because of the overlap existing among different classes, there exists some ambiguity in the feature space for a given instance. The uncertainty affects the precision of prediction. Rough sets form a conceptual vehicle to deal with ambiguous, vague, and uncertain knowledge while the neighborhood rough set model is an extension of traditional rough set model to deal with the uncertainty in the numerical data. The problems indicated above stimulate us to propose two multi-label classification algorithms based on neighborhood rough sets to cope with the uncertainty as well as the correlation among labels. The two proposed algorithms referred to as MLRS and MLRS-LC consider the global and local correlation among the labels. By introducing the concept of upper and lower approximations of neighborhood rough set model, MLRS and MLRS-LC firstly find out all the possibly related labels for a given instance and exclude all unrelated labels. Then they confirm the final labels according to the neighborhood of given instance. Experimental results concerning seven multi-label datasets show that the proposed approaches exhibit a promising performance when considering uncertainty and correlation aspects. They can not only improve the classification precision but also reduce the training time compared with other standard multi-label algorithms.

The paper is organized as follows. In Section 2, basic notation and evaluation metrics used in multi-label classification are briefly introduced. Section 3 provides some background material on multi-label classification and neighborhood rough sets respectively. Sections 4 and 5 respectively introduce the proposed approaches MLRS and MLRS-LC. Section 6 contains experimental results obtained by applying the proposed algorithms and other multi-label learning algorithms to multi-labeled datasets. Finally, Section 7 concludes the study and identifies some future research directions.

2. Preliminaries

In this section, we present the formal notation to be used throughout the paper.

We assume that $X \subset R^d$ denotes an input domain of instances and any instance is represented as a d-dimensional vector $x = [x^1, x^2, ..., x^d]$, $(x \in X)$. Let $Y = \{l_1, l_2, ..., l_m\}$ be a finite domain of possible labels. Each instance is associated with a subset of Y and this subset is described as an m-dimensional vector $y = [y^1, y^2, ..., y^m]$ where $y^j = 1$ only if instance x has label l_j and 0 otherwise.

Let $T = \{(x_i, y_i)|1 \leqslant i \leqslant n, x_i \in X, y_i \subset Y\}$ be a training set composed of n labeled instances and $D = \{(x_i, y_i)|1 \leqslant i \leqslant q, x_i \in X, y_i \subset Y\}$be a testing set composed of q labeled instances. The subscript in this description is used to avoid the confusion with the label dimension. Therefore, y_i^j corresponds to the binary relevance of the jth label belonging to the ith instance.

The performance evaluation of the multi-label classification system is different from that of the singe-label classification system. In multi-label classification, the evaluation is much more complicated. In experimental evaluation, we consider some measures proposed in literature Schapire & Singer (2000) and Godbole & Sarawagi (2004).

(1) *Hamming loss* (Schapire & Singer, 2000): this measure evaluates how many times an instance-label pair is misclassified considering the predicted set of labels y' and the ground-truth set of labels y.

$$hloss = 1 - \frac{1}{mq}\sum_{i=1}^{q}\sum_{j=1}^{m} 1_{y_i^j = y_i^j}$$

(2) *Average precision* (Schapire & Singer, 2000): this measure evaluates the average fraction of labels ranked above a particular label $\lambda \in y_i$ which actually are in y_i.

$$avgprec = \frac{1}{q}\sum_{i=1}^{q}\frac{1}{|y_i|}\sum_{\lambda \in y_i}\frac{|\{\lambda' \in y_i : r_i(\lambda') \leqslant r_i(\lambda)\}|}{r_i(\lambda)}$$

where $r_i(l)$ denotes the rank of label $l \in Y$ predicted by the algorithm for a given instance x_i.

(3) *Accuracy* (Godbole & Sarawagi, 2004): the measure gives an average degree of similarity between the predicted and the ground truth label sets of all testing examples.

$$accuracy = \frac{1}{q}\sum_{i=1}^{q}\frac{|y_i \cap y_i'|}{|y_i \cup y_i'|}$$

(4) *F1-measure* (Godbole & Sarawagi, 2004): for completeness of the analysis, we include the *F1-measure*. This is the harmonic mean between precision and recall, common to information retrieval. It can be calculated from true positives, true negatives, false positives and false negatives based on the predictions and the corresponding actual values.

$$F1 = \frac{1}{q}\sum_{i=1}^{q}\frac{|y_i \cap y_i'|}{|y_i| + |y_i'|}$$

Smaller values of *Hamming loss* correspond to higher classification quality, while larger values of *average precision*, *accuracy* and *F-measure* relate to higher classification quality.

3. Related work

Before embarking on an introduction of MLRS and MLRS-LC presented in this paper, let us review some existing works on multi-label learning and neighborhood rough sets.

3.1. Multi-label classification

As mentioned in Section 1, multi-label classification algorithms can be categorized into two different groups: (i) problem transformation methods and (ii) algorithm adaption methods. The first group includes methods that are algorithm independent. They transform the multi-label problem into one or more single-label problems. The representative problem transformation methods include binary relevance method (BR) Boutell, Luo, Shen, et al. (2004), binary pair wise classification approach (PW) Hüllermeier, Fürnkranz, Cheng, et al. (2008) and label combination or label power-set method (LC) Tsoumakas & Vlahavas (2007). The second group includes methods that extend specific learning algorithms in order to handle multi-label data directly. Well-known approaches include Adaboost (Schapire & Singer, 2000), BP-MLL (Zhang & Zhou, 2006), lazy methods (Denœux, Younes, & Abdallah, 2010; Spyromitros, Tsoumakas, & Vlahavas, 2008; Zhang & Zhou, 2007) and others.

BR (Boutell et al., 2004) is a popular problem transformation method that learns m binary classifiers for each different label in Y. Then each binary model is trained to predict the relevance of one of labels. Although BR is mentioned throughout the literature,

it has often been overlooked on the ground that it does not directly model correlations which exist between labels in the training data. The argument is that, due to this information, BR's predictive performance suffers. In order to address this problem, Read et al. proposed a new multi-label method Classifier Chain (CC) Read, Pfahringer, Holmes, et al. (2009) based on the BR method. CC can overcome the label-independent defect while maintaining the acceptable computational complexity of BR.

PW (Hüllermeier et al., 2008) can also be used to address multi-label problem as a problem transformation method, where a binary model is trained for each pair of labels. The prediction of these models result more naturally in a set of pairwise preferences than a multi-label prediction (thus becoming popular in ranking schemes), but this method has been adapted in Hüllermeier et al. (2008) to make multi-label predictions. Although PW performs well in several domains, it faces quadratic complexity in terms of the number of labels and for this reason its applicability could be limited.

Another well-known problem transform method is the LC (Tsoumakas & Vlahavas, 2007) which transforms a multi-label problem into a single-label problem by treating all label sets as atomic labels, i.e. each label set is treated as a single label in a single-label problem. Although being able to model correlations between labels, LC is challenged by application domains with large number of labels and training examples, due to high computational complexity (exponential with the number of labels) and tendency to over-fit the training data because it can only model label sets observed in the training set. In order to deal with the aforementioned problems of LP, Tsoumakas, Katakis, & Vlahavas (2011) proposed a new method called RA*k*ELd which randomly breaks the initial set of labels into a number of small-sized label sets, and employing LP to train a corresponding multi-label classifier. This way, the resulting single-label classification tasks are computationally simpler and the distribution of their class values is less skewed.

The Boostexter system (Schapire & Singer, 2000) is an important milestone in multi-label classification and ranking. The Boostexter provides two boosting algorithms, Adaboost.MH and Adaboost.MR, which are two extensions of Adaboost for multi-label classification. While Adaboost.MH is designed to minimize Hamming loss, AdaBoost.MR is design to find a hypothesis which places the correct labels at the top of the ranking. De Comité, Gilleron, and Tommasi (2003) extended the alternating decision tree learning algorithm for multi-label classification where the Adaboost.MH algorithm is explored to train the multi-label alternating decision trees. These Adaboost-based methods have been recognized as having computational complexity (Petrovskiy, 2006).

BP-MLL (Zhang & Zhou, 2006) is an adaptation of popular back-propagation algorithm for multi-label learning. The main modification to the algorithm is the introduction of a new error function that takes multiple labels into account.

A number of multi-label methods are based on the popular *k* Nearest Neighbors (*k*NN) lazy learning algorithm Denœux et al., (2010), Spyromitros et al. (2008) and Zhang & Zhou (2007). The first step in all these approaches is the same as in *k*NN, i.e. retrieving the k nearest examples. What differentiates them is the aggregation of the label set of these examples. Taking the ML*k*NN (Zhang & Zhou, 2007) for example, it uses the maximum a posteriori principle in order to determine the label set of the test instance, based on prior and posterior probabilities for the frequency of each label within the *k* nearest neighbors. ML-*k*NN uses the *k*NN algorithm independently for each label and has the capability of producing a ranking of the labels as an output.

3.2. Neighborhood rough sets

Rough set theory (Pawlak, Grzymala-Busse, Slowinski, et al., 1995), introduced by Pawlak, has been introduced as a tool to conceptualize, organize and analyze various types of data, in particular, to deal with inexact, uncertain or vague knowledge.

Pawlak's rough set model is built on equivalence relations and equivalence classes. The samples are said to be equivalent or indiscernible if their attribute values are identical to each other. However, some attributes in data are numerical in real-world applications. In order to extend the rough set model to support numerical attributes, Yao (1998) and Hu, Yu, & Xie (2008a) proposed the neighborhood rough set model based on the neighborhood relations. Formally speaking, the decision table $IS = \langle U, A\rangle$ embraces the samples for classification, where U is the nonempty set of samples $\{x_1, x_2, ..., x_n\}$, A is the nonempty set of attributes. To be more specific, $A = C \cup D$, where C is a set of condition attributes and D is a decision attribute.

Definition 2. Given arbitrary $x_i \in U$, $B \subseteq C$ and metric function Δ, the neighborhood $\delta_B(x_i)$ of x_i in the subspace B is defined as

$$\delta_B(x_i) = \{x_j | \Delta(x_i, x_j) \leqslant \delta, x_j \in U\}, \quad \text{where } \delta \geqslant 0$$

We also call $\delta_B(x_i)$ a neighborhood information granule induced by attribute B and object x_i. The family of neighborhood information granules $\{\delta_B(x) | x \in U\}$ forms a set of elemental concepts in the universe, which cover the universe, rather than partitioning it. In fact, neighborhood of x_i is a subset of samples close to x_i. There are several ways to define the neighborhoods δ of samples. One can define it using the fixed radius from the prototype sample or define the neighborhood with fixed k samples in the neighborhood, like the one considered in the k-nearest-neighbor method.

The neighborhood relation R over the universe can be written as a matrix $M(R) = (r_{ij})_{n\times n}$, where

$$r_{ij} = \begin{cases} 1, & x_j \in \delta_B(x_i) \\ 0, & otherwise \end{cases}$$

Definition 3. Given a neighborhood approximate space $\langle U, R\rangle$, for arbitrary subset $X \subseteq U$, $x_i \in U$ and a family of neighborhood information granules $\delta_B(x_i)$, $i = 1, 2, \ldots, n$, we define the lower and upper approximations of X in terms of the neighborhood relation R as

$$\underline{R_B}X = \{x_i | \delta_B(x_i) \subseteq X, \quad x_i \in U\}$$

$$\overline{R_B}X = \{x_i | \delta_B(x_i) \cap X \neq \varnothing, \quad x_i \in U\}$$

The lower approximation is the maximal union of the elements consistently contained in X, while the upper approximation is the minimal union of elements containing X. The difference between lower approximation and upper approximate is called approximation boundary of $X : BN(X) = \overline{R_B}X - \underline{R_B}X$. The element in the boundary region are inconsistent because only part of their neighbors belong to X.

Definition 4. Given a neighborhood decision table $NDT = \langle U, C \cup D\rangle$, $X_1, X_2, ..., X_N$ are the object subsets with decisions 1–N, $\delta_B(xi)$ is the neighborhood information granules including x_i and generated by attributes $B \subseteq C$, Then the lower and upper approximations of the decision D with respect to attributes B are defined as

$$\underline{R_B}D = \cup_{i=1}^{N} \underline{R_B}X_i$$

$$\overline{R_B}D = \cup_{i=1}^{N} \overline{R_B}X_i$$

where

$$\underline{R_B}X = \{x_i|\delta_B(x_i) \subseteq X, \quad x_i \in U\}$$

$$\overline{R_B}X = \{x_i|\delta_B(x_i) \cap X \neq \varnothing, \quad x_i \in U\}$$

The decision boundary region of D with respect to attributes B is defined as $BN(D) = \overline{R_B}D - \underline{R_B}D$.

The size of the decision boundary region reflects the degree of roughness of the decision. Usually we hope that the boundary region of the decision is as little as possible for decreasing uncertainty in decision. The lower approximation of the decision is called decision *positive region* which is denoted by $POS_B(D)$.

In practice, the above definition of the lower and the upper approximations are too strict to tolerate noise in the data. Hu, Liu, & Yu, (2008b) introduced the variable precision neighborhood rough sets to deal with the problem.

Definition 5. Given two sets A and B expressed in the universe U, the inclusion degree of A in B is defined as

$$I(A,B) = \frac{Card(A \cap B)}{Card(A)}$$

where $Card(\Phi)$ stands for the number of the elements of set Φ.

Definition 6. Given any subset $X \subseteq U$, then we define β lower and upper approximations of X as

$$\underline{R_B^\beta}X = \{x_i|I(\delta_B(x_i),X) \geqslant \beta, x_i \in U\}$$

$$\overline{R_B^\beta}X = \{x_i|I(\delta_B(x_i),X) \geqslant 1-\beta, x_i \in U\}$$

The upper definitions are given in the context of single-label classification. The neighbors of the object in the decision positive region consistently belong to one of the decision classes. The neighbors of the object in the decision boundary region come from more than one decision class. Namely, according to the information of the neighborhoods, the object in the positive region can be classified into one of the classes without uncertainty, while the object in the boundary region cannot be completely classified because of its various neighbors.

Fig. 2 shows a single-label example of binary classification in a 2-D numerical space, where class $X1$ is marked with '$*$' and class $X2$ is marked with '+'. We associate a neighborhood to every object in the sample space such as a, b, c. It is easy to find that the neighborhood of a is completely contained in class $X1$ ($\delta(a) \subseteq X1$) and neighborhood of c is completely contained in class $X2$ ($\delta(c) \subseteq X2$), while the objects in the neighborhood of b come from class X_1 and class $X2$ ($\delta(b) \cap X_1 \neq \varnothing$ and $\delta(b) \cap X_2 \neq \varnothing$). According to the above definition, a and c are located in the positive region of class $X1$ and class $X2$ and they respectively and constantly belong to one of the decision classes. On the other hand, b is positioned in the boundary region which is inconsistent in which misclassification could easily happen.

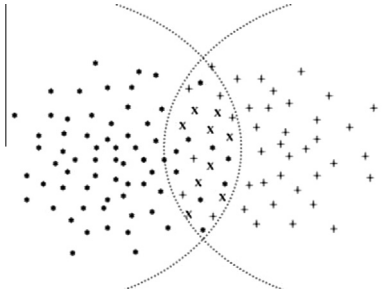


Fig. 2. Illustration of multi-label classification with two labels.

4. Multi-label classification using neighborhood rough sets (MLRS)

Contrary to single-label classification (consistent decision problem), the multi-label classification is regarded as inconsistent decision problem that two instances who have the same condition values may have different decision values. Namely, in multi-label classification, the base classes are non-mutually exclusive and may overlap by definition. A multi-label instance could be associated with a set of labels simultaneously. Generally speaking, the object with multiple labels is located in the overlapped region. Fig. 2 illustrates a binary multi-label classification task. Two classes X_1 and X_2 denoted by '$*$' and '+' respectively. Examples belonging to both '$*$' and '+' classes simultaneously are denoted by 'x'.

Because the classes in consistent decision system are mutually exclusive, it is easy to classify the instance according to some discriminative principle, such as maximum posterior probability $P(\omega_l|\delta(x_i)) = \max_j P(\omega_j|\delta(x_i))$. However, in inconsistent decision system, it is unreasonable to predict the labels with the same criteria as in consistent system. As shown in Fig. 2, there is co-occurrence among labels in multi-label classification. The co-occurrence states that the appearance of one label may improve the probability of appearance of other related labels. So it is indispensable to consider the correlation among labels during the process of classification. More specially, the probability $p(l \in y_i)$ of label l belonging to instance x_i can be formulated as follows.

$$p(l \in y_i) = \sum_{1 \leqslant j \leqslant m} p(l \in y_i|l_j \in y_i)p(l_j \in y_i) \tag{1}$$

$p(l_j \in y_i)$ denotes the probability of the label l_j belonging to instance x_i. $p(l \in y_i|l_j \in y_i)$ represents the conditional probability of the label l belonging to instance x_i when the label l_j belongs to instance x_i .

As mentioned above, there exists uncertainty during the process of classification. So we introduce the neighborhood rough set model to construct a new framework MLRS for the multi-label classification, which considers the uncertainty as well as the correlation among labels.

For each testing instance x_i, MLRS first finds its k nearest neighbors $\delta(x_i)$ from the training set. Then for each label l, let $|\delta_l(x_i)|$ denote the number of the neighbors with label l in the neighborhood $\delta(x_i)$. Let β ($0.5 \leqslant \beta \leqslant 1$) be the inclusion degree of the neighborhood rough sets and k is the number of neighbors, which represents the granularity level of the neighborhood granules. We observed that the complexity of the classification depends not only on the given feature space but also on the assumed granularity level. In order to get better results, we should specify a proper level of granularity.

According to the definition of neighborhood rough sets, if $|\delta_l(x_i)| \geqslant k \times \beta$, namely object x_i is located in the positive region of class l and it should be assigned label l, namely $p(l \in y_i) = 1$; If $|\delta_l(x_i)| < k \times (1-\beta)$, it can be confirmed that the object x_i is located in the negative region of class l and it should have no relation with label l, namely $p(l \in y_i) = 0$; If $k \times (1-\beta) \leqslant |\delta_l(x_i)| < k \times \beta$, the object x_i is located in the boundary region of class l and the algorithm will predict the ground truth labels for the instance x_i according to the neighborhood $\delta(x_i)$ and the correlation among different labels. Here probability formula (1) is employed for the prediction of probability of label l. $p(l_j \in y_i)$ is calculated following the formula $p(l_j \in y_i) = |\delta_{l_j}(x)|/\sum_{1 \leqslant f \leqslant m}|\delta_{l_f}(x)|$. The more the neighbors with label l_j, the greater the probability $p(l_j \in y_i)$ will be obtained and the higher the possibility of instance x_i related with label l_j will be gotten. $p(l \in y_i|l_j \in y_i)$ is denoted by the percentage of training

instances simultaneously with label l and l_j among the training instances only with label l_j.

Considering an example in Fig. 3, it illustrates how to predict the labels for some testing instances. As before, two classes X_1 and X_2 respectively marked by '$*$' and '+' in a 2-D space. Examples belonging to both X_1 and X_2 simultaneously are denoted by 'X'. For convenience, we assume that the distribution of two classes is circular and the number of the nearest neighbors $k = 5$. We associate a neighborhood to each object in example space such as a, b, c, d and assume the inclusion degree $\beta = 1$. For instance a, its each neighbor only has one label X_1, so $|\delta_{X_1}(a)| = 5$ and $|\delta_{X_2}(a)| = 0$. According to MLRS, the instance a only has label X_1. Similarly, the instance d is assigned label X_2. For instance b and c, both of them are in the overlapped region and their neighbors are not homogeneous. $|\delta_{X_1}(b)| = 5$ and $|\delta_{X_2}(b)| = 1$ indicates that we cannot decide the label for instance b directly and then formula $p(X_1 \in y_b) = \sum_{1 \leqslant j \leqslant 2} p(X_1 \in y_b | X_j \in y_b) p(X_j \in y_b)$ is used for the prediction of label X_1 of instance b. The analysis is the same for other situations.

The multi-label classification based on neighborhood rough sets algorithm MLRS can be formulated in the form of Algorithm 1. Here, we still use the formal notation introduced in Section 2. The input arguments include T, k, x, β and s. T is the training set and x denotes the testing instance. k denotes the number of the nearest neighbors, namely the granularity level of the neighborhood granules. β is the inclusion degree. Furthermore, s is the smoothing parameter controlling the value of conditional probability (s is set to be 1 which yields the Laplace smoothing). $y = [y^1, y^2, ...y^m]$ is the predicted result of testing instance and y^i is the value for label l_i.

Algorithm 1: Multi-label classification based on neighborhood rough sets (MLRS)

Input: T, k, x, β, s
Output: y
Train:
// Computing the conditional probability
1: for each $l_j \in Y$, $1 \leqslant j \leqslant m$
2: for each $l_i \in Y$, $1 \leqslant i \leqslant m$
3: $p(l_i \in y | l_j \in y) = \sum_{\phi=1}^{n} y_\phi^{i \cap j} / (\sum_{\alpha=1}^{n} y_\alpha^j + s)$
Test:
// Computing the prior probability
4: compute the neighborhood $\delta(x)$ of testing instance x
5: for each $l_j \in Y$, $1 \leqslant j \leqslant m$
6: $p(l_j \in y) = |\delta_{l_j}(t)| / \sum_{1 \leqslant i \leqslant m} |\delta_{l_i}(t)|$
// Computing the probability of the testing instance x
7:for each $l_i \in Y$, $1 \leqslant i \leqslant m$
8: if $|\delta_{l_i}(t)| \geqslant k \times \beta$
9: $p(y^i) = 1$
10: else if $|\delta_{l_i}(t)| \leqslant k \times (1 - \beta)$
11: $p(y^i) = 0$
12: else
13: $p(y^i) = \sum_{1 \leqslant j \leqslant m} p(l_i \in y | l_j \in y) p(l_j \in y)$
14: if $p(y^i) \geqslant$ *threshold* $y^i = 1$ else $y^i = 0$

As shown in Algorithm 1, based on the training set, steps from 1 to 3 are used to estimate the conditional probability $p(l \in y | l_j \in y)$ where $l_j \in y$ means the training instance with label l_j. In steps from 4 to 6, we estimate the prior probability $p(l_j \in y)$ from the neighborhood of instance x. Finally, steps from 7 to 13 count the probability of the testing instance x belonging to each class and step 14 produces the prediction of some label by comparing the values of probability with some thresholds.

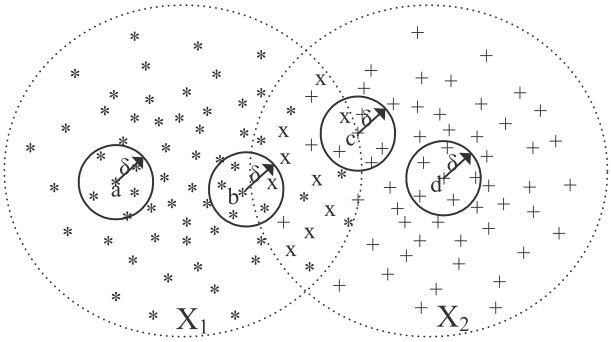


Fig. 3. Illustrationof the prediction in multi-label classification system.

5. The MLRS-LC approach

MLRS exploits label correlations in a global way by assuming that the correlations are shared by all instances. In real-word tasks, the label correlations are naturally local, where a label correlation may be shared by only a subset of instances rather than all instances. For example, as shown in Fig. 4, we consider the strong correlation between *water* and *boat*. For the image (b), *boat* are less prominent and thus could be difficult to predict; in this case, the correlation between *water* and *boat* can be helpful in learning label *boat* since the label *water* is relatively easier to predict in this image. For the image (c), with the *water* clearly presented, this correlation turns to be misleading since it suggests including the label *boat*, whereas this label is not proper for the image. Exploiting such correlations at the global level will enforce unnecessary or even misleading constraints on instances that do not contain such correlations, and therefore may hurt the performance by predicting some irrelevant labels.

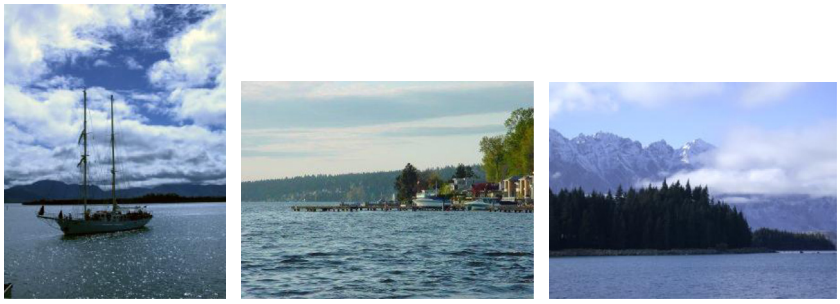
In this section, we present MLRS-LC (MLRS using local correlation) approach by exploiting label correlations locally, which tries to exploit label correlations in the data locally. We do not assume that there are external knowledge sources specifying the locality of label correlations. Instead, we assume that the instances can be divided into different topics according to the label correlations and each topic includes related instances.

Alluding to the previous discussion, we know that for a specific instance, only a subset of label correlations is helpful while the others are less informative or even harmful and the instances in the same topic share the same label correlations. We assume that the training data can be divided into ψ topics $T = \{T_1, T_2, ..., T_\psi\}$, where instances in the same topic share the same subset of label correlations. The topics can be discovered via clustering. The probability $p(l \in y_i)$ of label l belonging to instance x_i is expressed as follows.

$$p(l \in y_i) = \sum_{1 \leqslant \Delta \leqslant \psi} \sum_{1 \leqslant j \leqslant m} p(l \in y_i | l_j \in y_i, x_i \in T_\Delta) p(l_j \in y_i | x_i \in T_\Delta) p(x_i \in T_\Delta) \tag{2}$$

$p(x_i \in T_\Delta)$ is the probability of x_i belonging to topic T_Δ. $p(l_j \in y_i | x_i \in T_\Delta)$ denotes the probability of the label l_j belonging to instance x_i when instance x_i belongs to topic T_Δ. $p(l \in y_i | l_j \in y_i, x_i \in T_\Delta)$ represents the conditional probability of the label l belonging to instance x_i when the label l_j belongs to instance x_i in topic T_Δ.

During the training process, the training instances are clustered into several topics in the label space. Then in each topic, the label correlations are exploited with the same method as in MLRS and the conditional probabilities $p(l \in y_i | l_j \in y_i, x_i \in T_\Delta)$ is obtained in each topic. Given a testing instance, its k nearest neighbors $\delta(x_i)$ are found from the overall training set. The probability of instance x_i belonging to topic T_Δ is determined according to the number of neighbors belonging to topic T_Δ. Namely, $p(x_i \in T_\Delta) = |\delta_{T_\Delta}(x_i)| / k$,



(a) *water*, *sky*, *boat*, *mountain*

(b) *water*, *sky*, *tree*, *mountain*, *boat*

(c) *water*, *mountain*, *tree*, *sky*

Fig. 4. Illustration of local label correlations.

where $|\delta_{T_\Delta}(x_i)|$ denotes the number of the neighbors belonging to topic T_Δ. Then in each topic, the k nearest neighborhood of the testing instance are found and the probabilities $p(l_j \in y_i | x_i \in T_\Delta)$ are calculated for each topics according the number of neighbors coming with label l_j, which are the same as in MLRS.

The computing of MLRS-LC is presented in the form of Algorithm 2. Here a commonly used clustering algorithm of k-means is used to form topics. The number of clusters (the number of topics) is decided by the number of labels. The input and the output arguments are as same as those used in MLRS.

Algorithm 2: MLRS using Local Correlation (MLRS-LC)

Input: T, k, x, β, s
Output: y
Train:
// Obtain topics from training set and compute the conditional probability in each topic
1: cluster the training set into topics $T = \{T_1, T_2, ..., T_\psi\}$ using k-means
2: for each $T_\Delta \in T$, $1 \leqslant \Delta \leqslant \psi$
3: for each $l_j \in Y$, $1 \leqslant j \leqslant m$
4: for each $l_i \in Y$, $1 \leqslant i \leqslant m$
5: $p(l_i \in y | l_j \in y, x \in T_\Delta) = \sum_{\phi \in T_\Delta} y_\phi^{i \cap j} / (\sum_{\alpha \in T_\Delta} y_\alpha^j + s)$
Test:
// Compute the probability $p(l \in y)$ of the testing instance x
6: Determine the neighborhood $\delta(x)$ of testing instance x from training set
7: get the probabilities $p(x \in T_\Delta)$ for each topic T_Δ according to $\delta(x)$
8: for each $T_\Delta \in T$, $1 \leqslant \Delta \leqslant \psi$
9: compute the neighborhood $\delta_{T_\Delta}(x)$ from topic T_Δ
10: for each $l_i \in Y$, $1 \leqslant i \leqslant m$
11: if $|\delta_{l_i, T_\Delta}(x)| \geqslant k \times \beta$
12: $p(l \in y | x \in T_\Delta) = 1$
13: else if $|\delta_{l_i, T_\Delta}(x)| \leqslant k \times (1 - \beta)$
14: $p(l \in y | x \in T_\Delta) = 0$
15: else
16: $p(l \in y | x \in T_\Delta) = |\delta_{l, T_\Delta}(t)| / \sum_{1 \leqslant j \leqslant m} |\delta_{l_j, T_\Delta}(t)|$
17: for each $l_i \in Y$, $1 \leqslant i \leqslant m$
18: $p(y^i) = \sum_{1 \leqslant \Delta \leqslant \psi} \sum_{1 \leqslant j \leqslant m} p(l \in y | l_j \in y, x \in T_\Delta)$ $p(l_j \in y | x \in T_\Delta) p(x \in T_\Delta)$
19: if $p(y^i) \geqslant$ *threshold* $y^i = 1$ else $y^i = 0$

As shown in Algorithm 2, in steps from 1 to 5, we estimate the correlations among labels for each topic. Steps from 6 to 18 are used to calculate the probability of each label l belonging to testing instance x. Step 19 produces the final prediction for each label according to the probability compared with the threshold. The threshold is automatically produced.

6. Experiments

As mentioned above, there are many approaches to address multi-label problem. In this paper, we compare MLRS and MLRS-LC with the binary relevance method (BR) Tsoumakas et al., 2010, the kNN-based multi-label algorithms MLkNN (Zhang & Zhou, 2007) and BRkNN, the random k label-set method for multi-label classification RAkELd (Tsoumakas et al., 2011), the ensemble classifier chains algorithm ECC (Read, Pfahringer, Holmes, et al., 2011) and the back-propagation multi-label learning (BPMLL) Zhang & Zhou, 2006 learner, which are all well-known multi-label learning algorithms applicable to various multi-label problem. Furthermore, we show the influence of parameters k and β used in MLRS and MLRS-LC.

6.1. Datasets

We conducted comparative experiments using these algorithms on seven datasets containing multi-label instances coming from a variety of domains. Data statistics are listed in Table 1. In the table, "name" is the name of the dataset, "domain" represents the domain that the datasets come from. "instances" denotes the number of instances in the dataset. "attributes" is the number of attributes and "labels" means the number of labels. *Label Cardinality* is the average number of labels associated with each instance and showed by "cardinality". *Label density* is the average number of labels of instances dataset divided by the number of instances in the dataset and described as "density".

In Table 1, the first two datasets come from the music domain. *Emotions* (Tsoumakas et al., 2008) consists of 593 songs with six clusters of music emotions and *CAL500* (Turnbull, Barrington, Torres, et al., 2008) is composed of 500 popular western musical tracks. *Yeast* (Elisseeff & Weston, 2001) has 2,417 instances that each instance in the dataset represents a yeast gene and there

Table 1
Multi-label datasets used in experiments.

Name	Domain	Instances	Attributes	Labels	Cardinality	Density
Emotions	music	593	72	6	1.869	0.311
CAL500	music	502	68	174	26.044	0.150
Yeast	biology	2417	103	14	4.237	0.303
Medical	text	978	1449	45	1.245	0.028
Enron	text	1702	1001	53	3.378	0.064
Scene	image	2407	294	6	1.074	0.179
Corel5k	image	5000	499	374	3.522	0.009

Table 2
MLRS and MLRS-LC vs. other multi-label algorithms: *Hamming loss*.

algorithm	emotions	Yeast	Scene	Enron	Medical	CAL500	Corel5k
BR	0.2474 ± 0.0248	0.2454 ± 0.0091	0.1368 ± 0.0078	0.0508 ± 0.0022	0.0103 ± 0.0014	0.1615 ± 0.0049	0.0098 ± 0.0001
RAkELd	0.2642 ± 0.0177	0.2734 ± 0.0078	0.1371 ± 0.0082	0.0527 ± 0.0028	**0.0101 ± 0.0015**	0.1950 ± 0.0054	0.0098 ± 0.0001
BPMLL	0.2060 ± 0.0218	0.2225 ± 0.0099	0.2767 ± 0.0544	0.2589 ± 0.0594	0.7255 ± 0.2603	0.2478 ± 0.0155	0.5792 ± 0.0342
MLkNN	0.1951 ± 0.0243	**0.1933 ± 0.0123**	**0.0862 ± 0.0084**	0.0523 ± 0.0022	0.0151 ± 0.0018	0.1388 ± 0.0050	**0.0094 ± 0.0001**
BRkNN	0.1934 ± 0.0182	0.1952 ± 0.0119	0.0920 ± 0.0091	0.0580 ± 0.0023	0.0180 ± 0.0019	0.1425 ± 0.0036	**0.0094 ± 0.0001**
ECC	0.2030 ± 0.0243	0.2070 ± 0.0097	0.0926 ± 0.0111	**0.0481 ± 0.0024**	0.0102 ± 0.0013	0.1450 ± 0.0025	NaN
MLRS	**0.1807 ± 0.0200**	0.2032 ± 0.0093	0.0927 ± 0.0042	0.0568 ± 0.0025	0.0186 ± 0.0015	**0.1366 ± 0.0042**	**0.0094 ± 0.0001**
MLRS-LC	0.2054 ± 0.0379	0.2139 ± 0.0076	0.0990 ± 0.0099	0.0640 ± 0.0024	0.0212 ± 0.0038	0.1642 ± 0.0063	0.0117 ± 0.0003

Table 3
MLRS and MLRS-LC vs. other multi-label algorithms: *average precision*.

Algorithm	Emotions	Yeast	Scene	Enron	Medical	CAL500	Corel5k
BR	0.7014 ± 0.0316	0.6216 ± 0.0151	0.7109 ± 0.0283	0.5929 ± 0.0213	0.8341 ± 0.0278	0.3513 ± 0.0141	**0.2494 ± 0.0093**
RAkELd	0.6919 ± 0.0270	0.6138 ± 0.0160	0.7296 ± 0.0155	0.5272 ± 0.0227	0.8165 ± 0.0297	0.2800 ± 0.0100	0.1402 ± 0.0067
BPMLL	0.7911 ± 0.0318	0.7451 ± 0.0166	0.6743 ± 0.0198	0.3382 ± 0.0708	0.1037 ± 0.0118	**0.5095 ± 0.0143**	0.0550 ± 0.0050
MLkNN	0.7965 ± 0.0406	**0.7620 ± 0.0144**	**0.8662 ± 0.0174**	0.6322 ± 0.0239	0.8062 ± 0.0275	0.4942 ± 0.0168	0.2465 ± 0.0087
BRkNN	0.8037 ± 0.0337	0.7599 ± 0.0189	0.8496 ± 0.0218	0.5479 ± 0.0175	0.7686 ± 0.0310	0.4589 ± 0.0143	0.1857 ± 0.0065
ECC	0.8000 ± 0.0404	0.7476 ± 0.0210	0.8495 ± 0.0184	**0.6875 ± 0.0247**	**0.8842 ± 0.0163**	0.4665 ± 0.0140	NaN
MLRS	**0.8046 ± 0.0223**	0.7462 ± 0.0116	0.8520 ± 0.0094	0.5874 ± 0.0263	0.7599 ± 0.0221	0.5081 ± 0.0128	0.2388 ± 0.0098
MLRS-LC	0.8187 ± 0.0296	0.7331 ± 0.0147	0.8250 ± 0.0237	0.5069 ± 0.0286	0.7131 ± 0.0329	0.4541 ± 0.0117	0.1641 ± 0.0190

Table 4
MLRS and MLRS-LC vs. other multi-label algorithms: *F1-measure*.

Algorithm	Emotions	Yeast	Scene	Enron	Medical	CAL500	Corel5k
BR	0.5566 ± 0.0380	0.5635 ± 0.0194	0.5732 ± 0.0319	0.5257 ± 0.0348	0.7771 ± 0.0335	0.3375 ± 0.0170	0.0923 ± 0.0109
RAkELd	0.5260 ± 0.0354	0.5285 ± 0.0153	0.5757 ± 0.0253	0.5020 ± 0.0373	**0.7816 ± 0.0352**	0.3357 ± 0.0121	0.0914 ± 0.0099
BPMLL	0.6640 ± 0.0307	0.6346 ± 0.0215	0.4886 ± 0.0414	0.3106 ± 0.0350	0.0578 ± 0.0208	0.4586 ± 0.0105	0.0232 ± 0.0004
MLkNN	0.6138 ± 0.0527	0.6204 ± 0.0270	0.6811 ± 0.0330	0.4288 ± 0.0306	0.6065 ± 0.0428	0.3240 ± 0.0168	0.0194 ± 0.0060
BRkNN	0.5902 ± 0.0338	0.5984 ± 0.0246	0.6277 ± 0.0300	0.2532 ± 0.0150	0.4746 ± 0.0472	0.3059 ± 0.0094	0.0035 ± 0.0017
ECC	0.5855 ± 0.0400	0.5973 ± 0.0257	0.6533 ± 0.0381	**0.5652 ± 0.0292**	0.7806 ± 0.0286	0.3385 ± 0.0108	NaN
MLRS	**0.7818 ± 0.0287**	0.625 ± 0.0111	**0.7335 ± 0.0277**	0.4879 ± 0.0102	0.6077 ± 0.0289	**0.4756 ± 0.0115**	0.0491 ± 0.0078
MLRS-LC	0.7805 ± 0.0248	**0.6426 ± 0.0256**	0.7209 ± 0.0162	0.451 ± 0.0265	0.6329 ± 0.0435	0.452 ± 0.0140	**0.1223 ± 0.0173**

Table 5
MLRS and MLRS-LC vs. other multi-label algorithms: *Accuracy*.

Algorithm	Emotions	Yeast	Scene	Enron	Medical	CAL500	Corel5k
BR	0.4623 ± 0.0352	0.4395 ± 0.0186	0.5353 ± 0.0318	0.4129 ± 0.0313	0.7465 ± 0.0330	0.2067 ± 0.0125	0.0633 ± 0.0077
RAkELd	0.4388 ± 0.0346	0.4054 ± 0.0155	0.5418 ± 0.0254	0.3937 ± 0.0351	0.7520 ± 0.0346	0.2053 ± 0.0088	0.0622 ± 0.0073
BPMLL	0.5736 ± 0.0371	0.5218 ± 0.0221	0.3645 ± 0.0420	0.1985 ± 0.0270	0.0301 ± 0.0108	0.3043 ± 0.0085	0.0118 ± 0.0002
MLkNN	0.5326 ± 0.0515	0.5162 ± 0.0300	0.6670 ± 0.0307	0.3316 ± 0.0264	0.5813 ± 0.0421	0.1972 ± 0.0121	0.0147 ± 0.0047
BRkNN	0.5149 ± 0.0349	0.5002 ± 0.0268	0.6198 ± 0.0285	0.2027 ± 0.0152	0.4563 ± 0.0452	0.1856 ± 0.0061	0.0025 ± 0.0013
ECC	0.5112 ± 0.0343	0.4924 ± 0.0249	0.6417 ± 0.0382	**0.4558 ± 0.0297**	**0.7542 ± 0.0274**	0.2093 ± 0.0083	NaN
MLRS	0.6585 ± 0.0300	0.485 ± 0.0113	0.6738 ± 0.0302	0.3818 ± 0.0091	0.5325 ± 0.0318	**0.3218 ± 0.0097**	0.035 ± 0.0052
MLRS-LC	**0.6692 ± 0.0310**	**0.5271 ± 0.0120**	**0.6766 ± 0.0133**	0.3566 ± 0.0185	0.5815 ± 0.0472	0.3048 ± 0.0189	**0.0869 ± 0.0068**

are 14 labels indicating gene functional groups. *Enron* is used for UC Berkeley Enron Email Analysis Project and *medical* is used in the Computational Medicine Center's 2007 Medical Natural Language Processing Challenge. The last two datasets are composed of images. *Scene* (Boutell et al., 2004) is a benchmark for image classification containing 2407 natural scene images and *Corel5k* (Duygulu, Barnard, de Freitas, et al., 2006) is based on 5000 Corel images, which is used for the ECCV 2002 paper.

6.2. Threshold selection

Given a vector w of real-valued probability outputs, a multi-label prediction y' can be obtained by applying a threshold function $f_t(w)$ such that:

$$y'_j = \begin{cases} 1 & if\ w_j \geqslant t \\ 0 & if\ w_j < t \end{cases}$$

Some approaches, such as ML-kNN use 0.5 as the value of the threshold. Fan & Lin (2007) point out that it is better to use threshold implied by some threshold function than considering an arbitrary value. Our experience is that calibrating a single threshold t to be used across the entire evaluation is more effective and efficient. We calibrate the threshold t as follows:

$$t = \arg\max_t(sum(and(y, y'_{w_j \geqslant t})))$$

where function and $(y, y'_{w_j \geqslant t})$ forms a vector whose corresponding value is 1 when $y_i = y'_i$ and 0 otherwise.

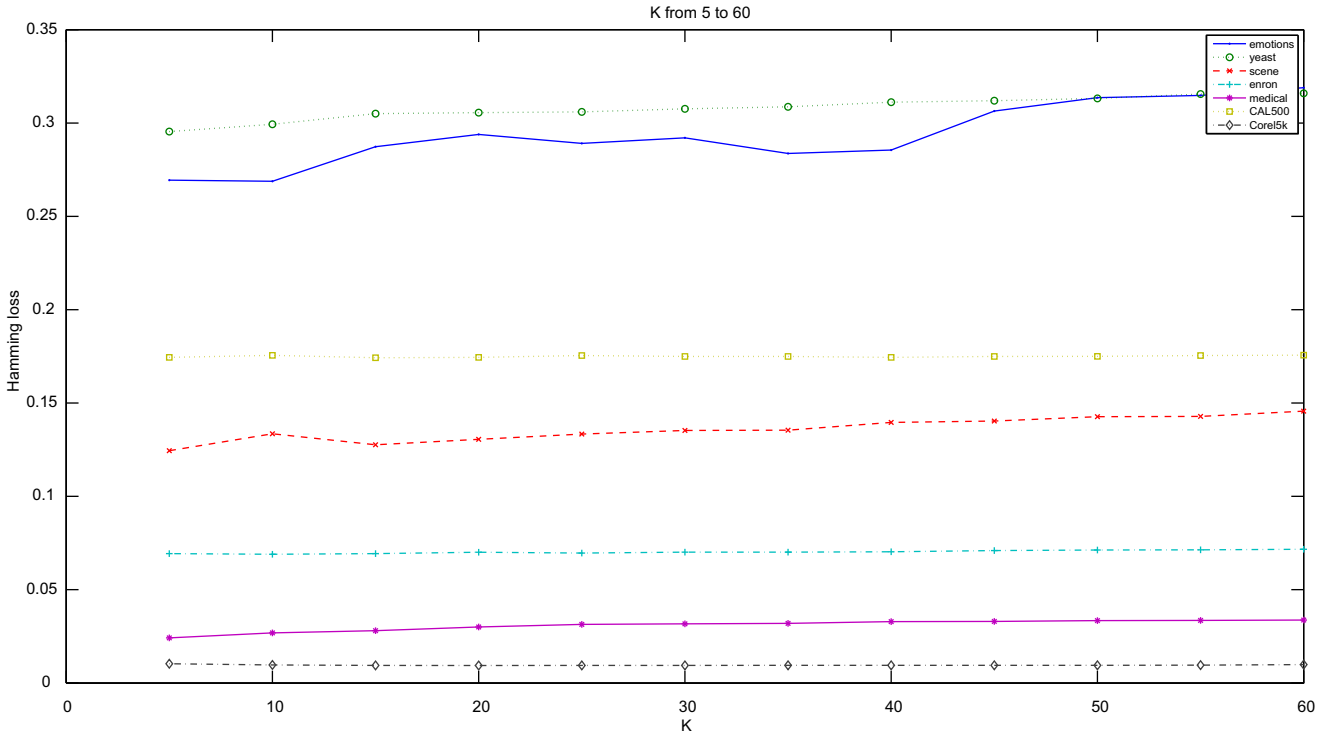


Fig. 5. Hamming-loss of MLRS for different levels of granularity *k*.

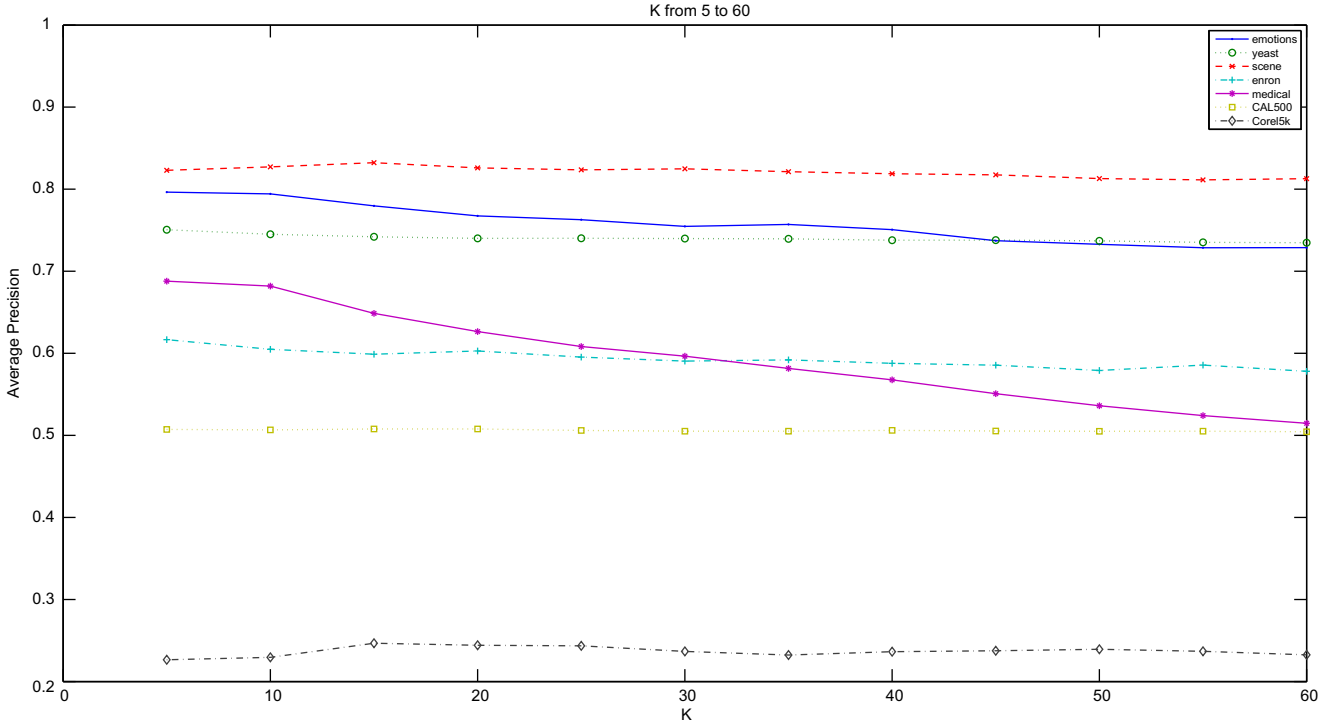


Fig. 6. Average precision of MLRS for different levels of granularity *k*.

6.3. Experimental setting

The implementation of the compared algorithms comes from the open source Mulan library (Tsoumakas, Xioufis, Vilcek, et al., 2011), which is based on the open source Weka library (Witten, Frank, & Hall, 2011). BR and RA*k*ELd use the C4.5 decision tree learning algorithm for training the underlying single-label classifier. As recommended in Zhang and Zhou (2007), MLkNN is run with 10 nearest neighbors and a smoothing factor equal to 1. As recommended in Zhang and Zhou (2006), BPMLL is run with 0.05 learning rate, 100 epochs and the number of the hidden units equal to 20% of the input units. The neighborhood is defined

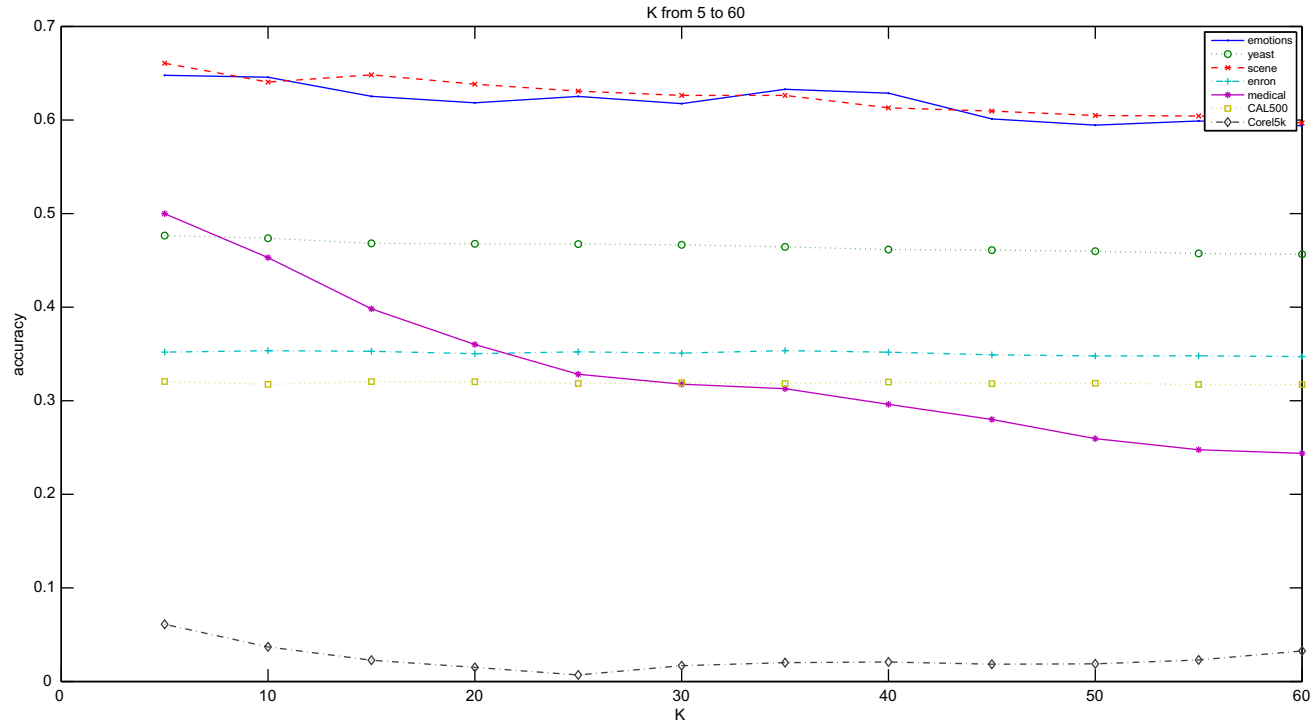


Fig. 7. Accuracy of MLRS for different levels of granularity *k*.

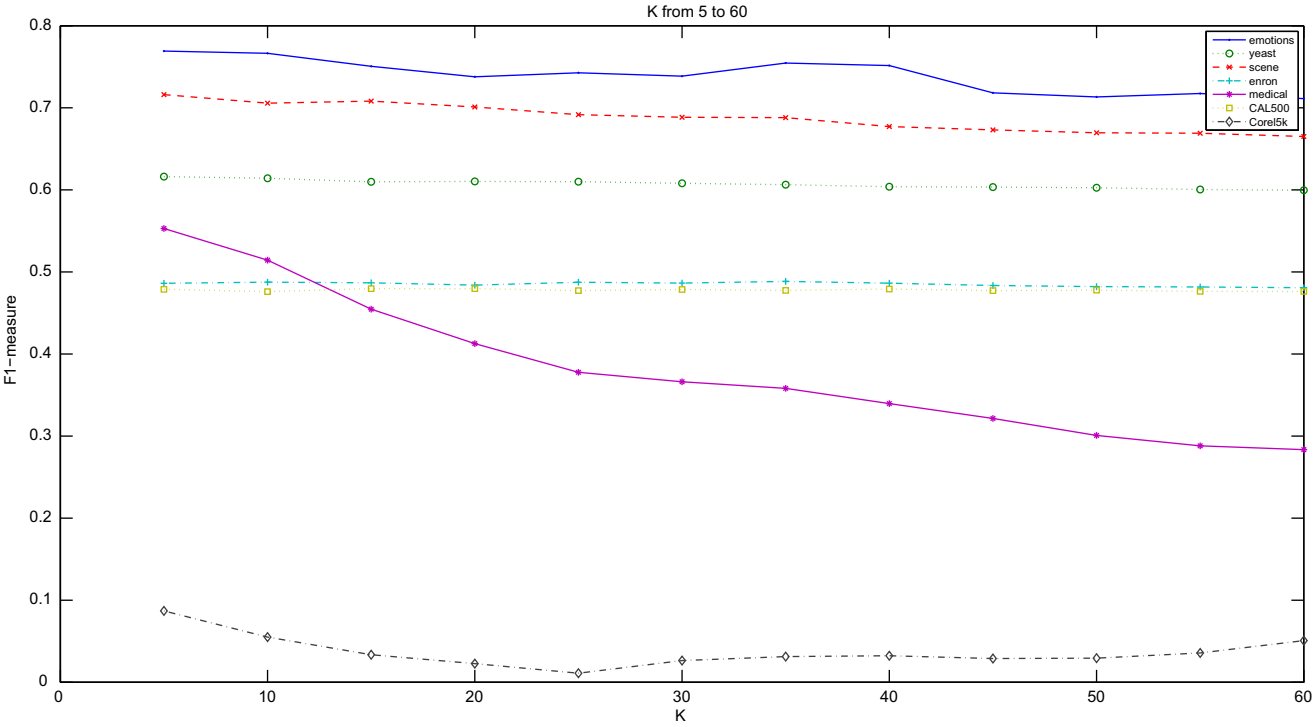


Fig. 8. F1-measure of MLRS for different levels of granularity *k*.

by the number of neighbors *k*. We evaluate all learning algorithms using 10 fold cross evaluation on the datasets mentioned above and use the average results to display the significance of results.All experiments were run on a workstation equipped with a 3.0GHZ processor and 8.0G memory.

6.4. Results and discussion

The results of ten cross validation in terms of predictive performances: *Hamming loss*, *average precision*, *accuracy* and *F1-measure* are shown in Tables 2–5 where $k = 10$, namely the

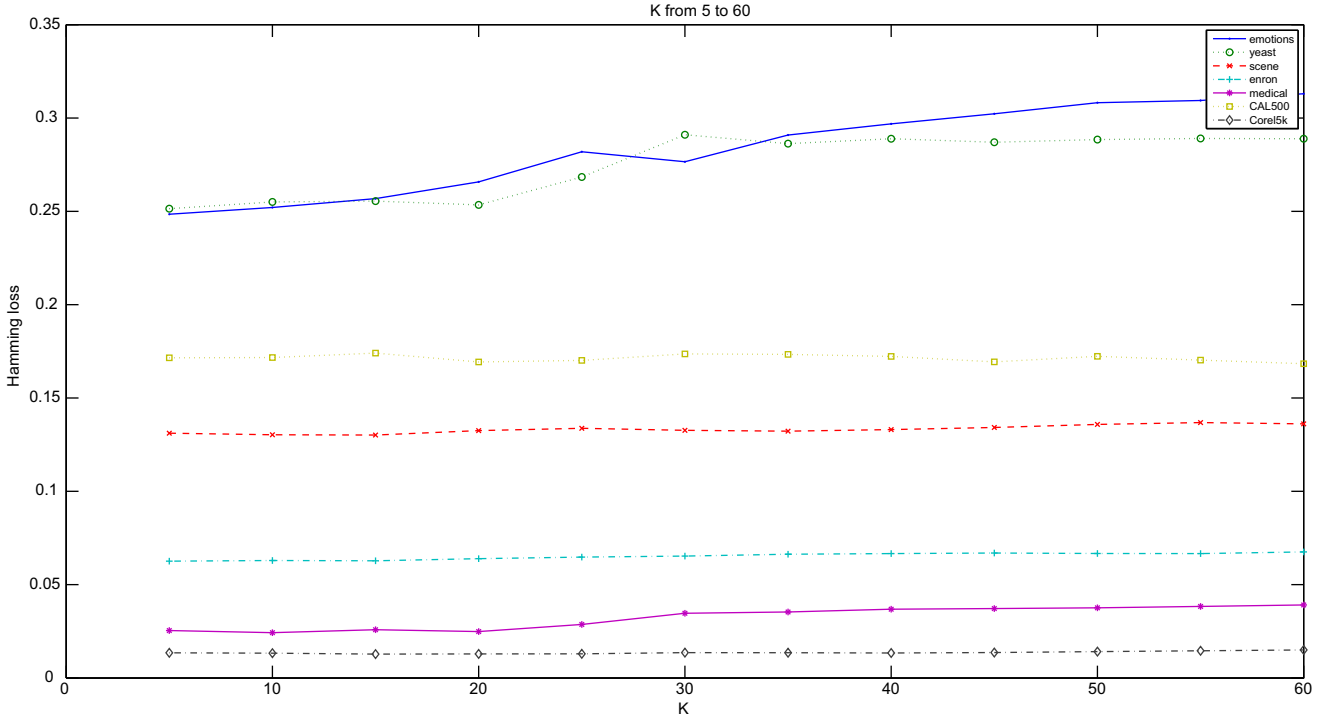


Fig. 9. Hamming-loss of MLRS-LC for different levels of granularity k.

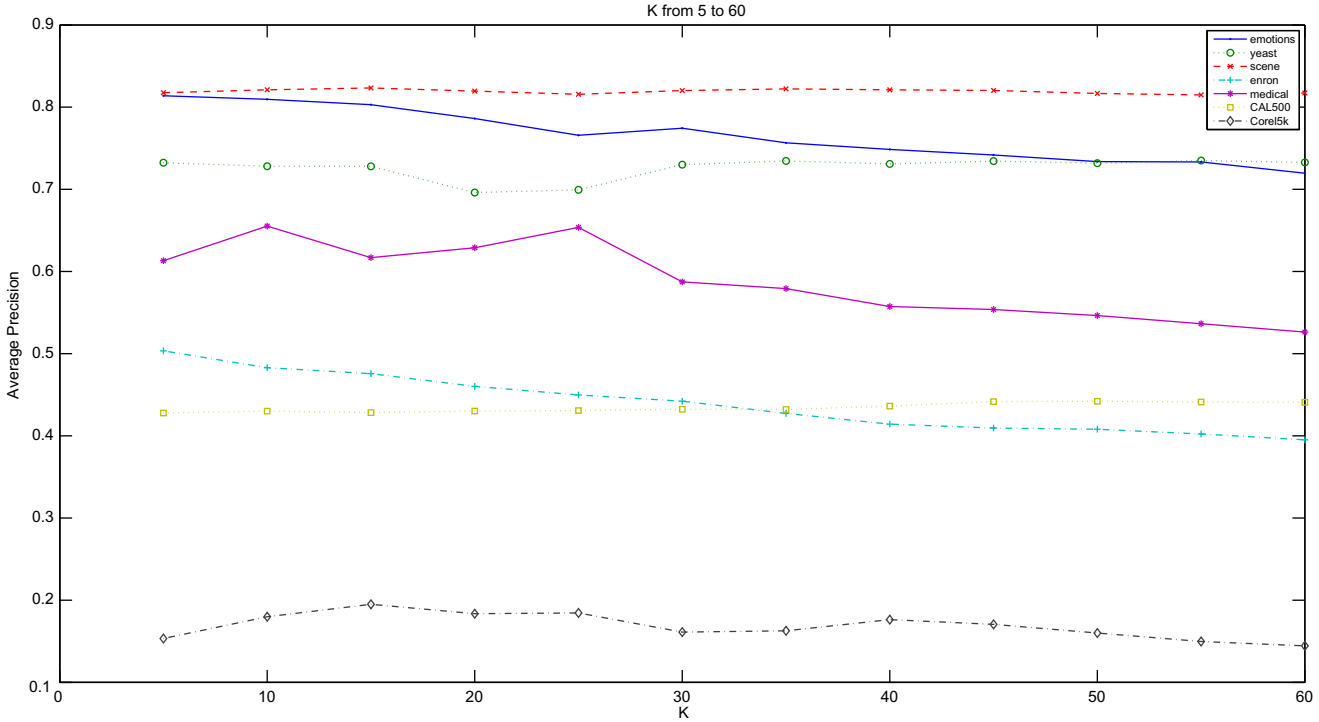


Fig. 10. Average precision of MLRS-LC for different levels of granularity k.

size of the granularity is 10 and the inclusion degree β is 1. *NaN* means the training process failed. The entries in boldface indicate the best performance. We study the results produced by the MLRS and MLRS-LC algorithms separately for each dataset.

In comparison with other methods known in the literature, MLRS and MLRS-LC yield superior predictive performance (see Tables 2–5) than any other methods on most data sets. The exception is *enron* and *medical* whose attributes are high-dimensional.

In multi-label classification, it cannot be expected that a method performs best over all types of evaluation measures, except in cases where the method is tailored to each and run

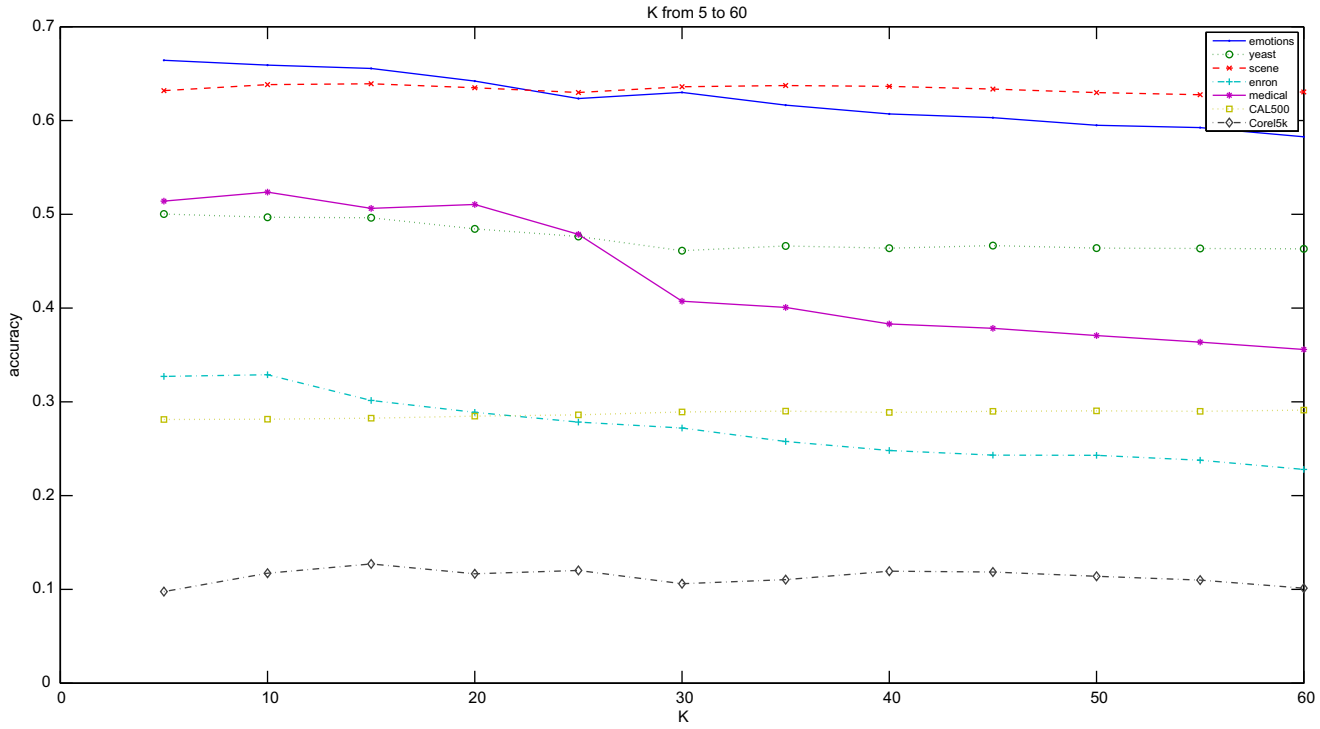


Fig. 11. Accuracy of MLRS-LC for different levels of granularity *k*.

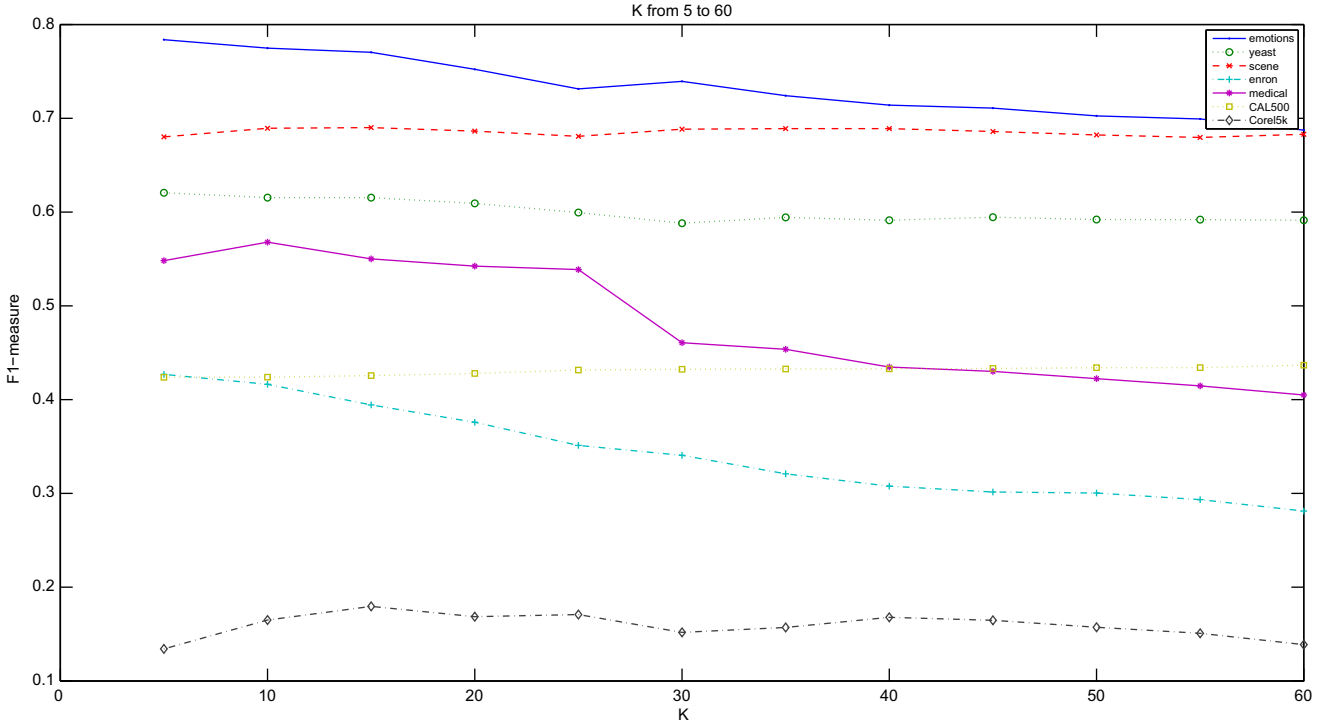


Fig. 12. F1-measure of MLRS-LC for different levels of granularity *k*.

separately. Therefore it comes as no surprise that different methods performed best under different measures. The nearest-neighbors-based methods ML*k*NN, RA*k*EL, MLRS and MLRS-LC perform well on *Hamming-loss* compared with other measures: it votes relatively more conservatively than other methods and is rewarded for it. However, nearest-neighbors-based methods give poor performance on high-dimensional datasets. This is because that the calculation of distances suffers from the sparse training instances caused by high-dimensional feature space. ECC perform well in several situations but its complexity prohibits its completion on anything but small datasets. In average precision, BPMLL perform best on *CAL500* while its hamming-loss is highest.

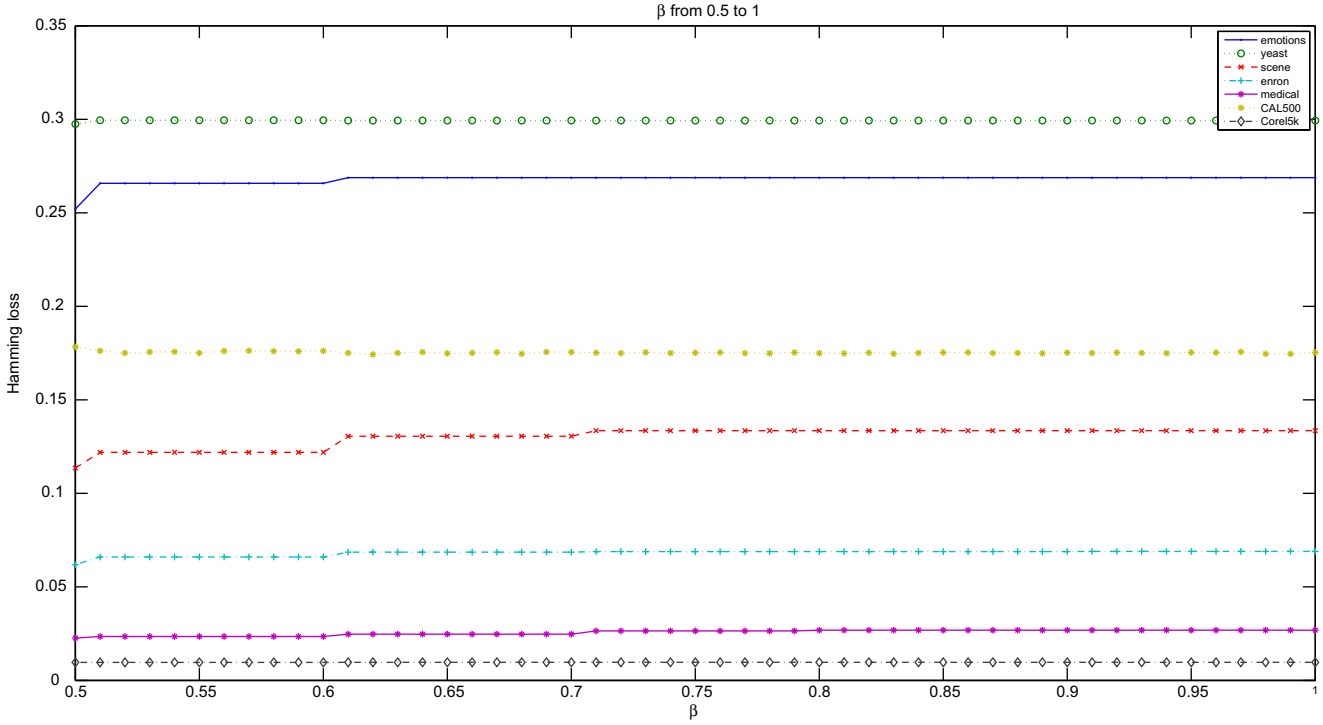


Fig. 13. Hamming-loss of MLRS for different levels of granularity *t*.

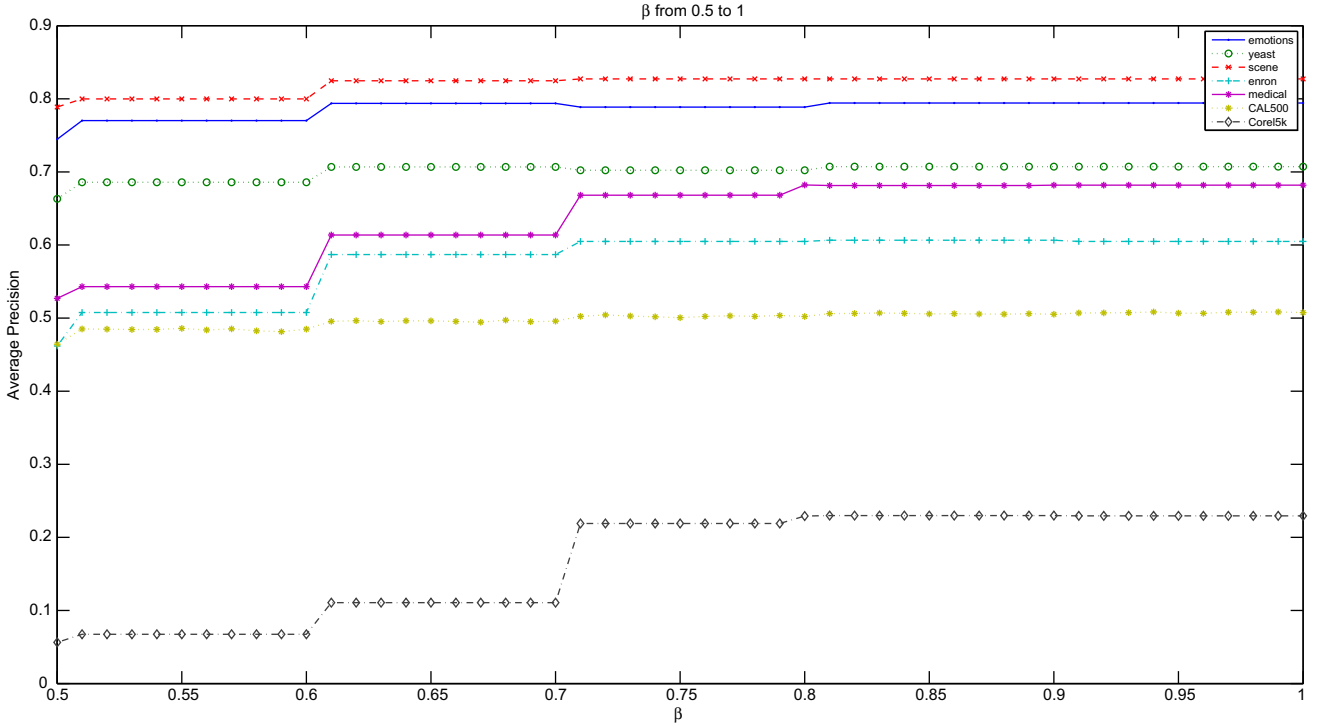


Fig. 14. Average precision of MLRS for different levels of granularity *t*.

In conclusion, the overall advantage of MLRS and MLRS-LC is clear: they prove competitive not only under label set-based like *accuracy* measures but also label-based measures like *Hamming-loss*. On the other hand, the simple baseline BR often performs surprisingly well against other methods on some datasets, such as *Corel5k* and *medical*.

The comparison between MLRS and MLRS-LC illustrates that none of them always perform significantly better than other under different measures. The performance depends on the character of datasets. In addition, we find that the performance of MLRS-LC is affected by the initial centroids which are used in the clustering of topics.

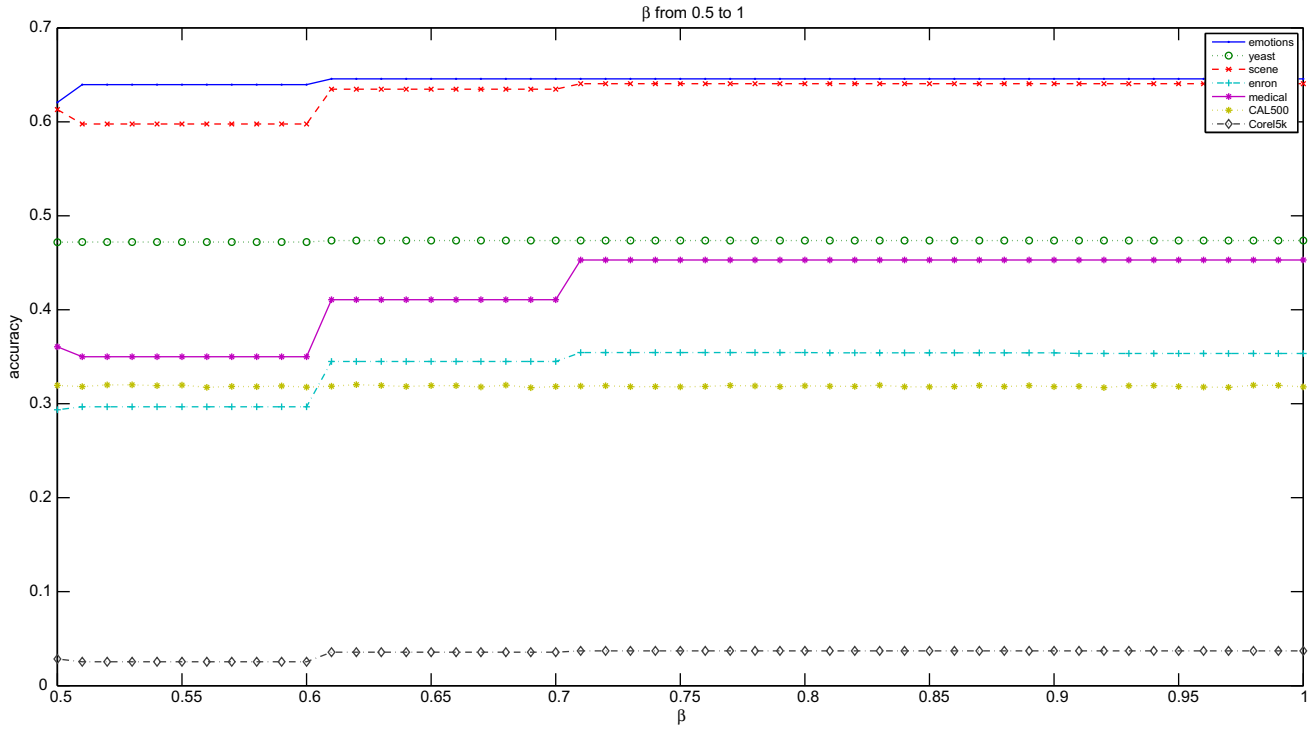


Fig. 15. Accuracy of MLRS for different levels of granularity t.

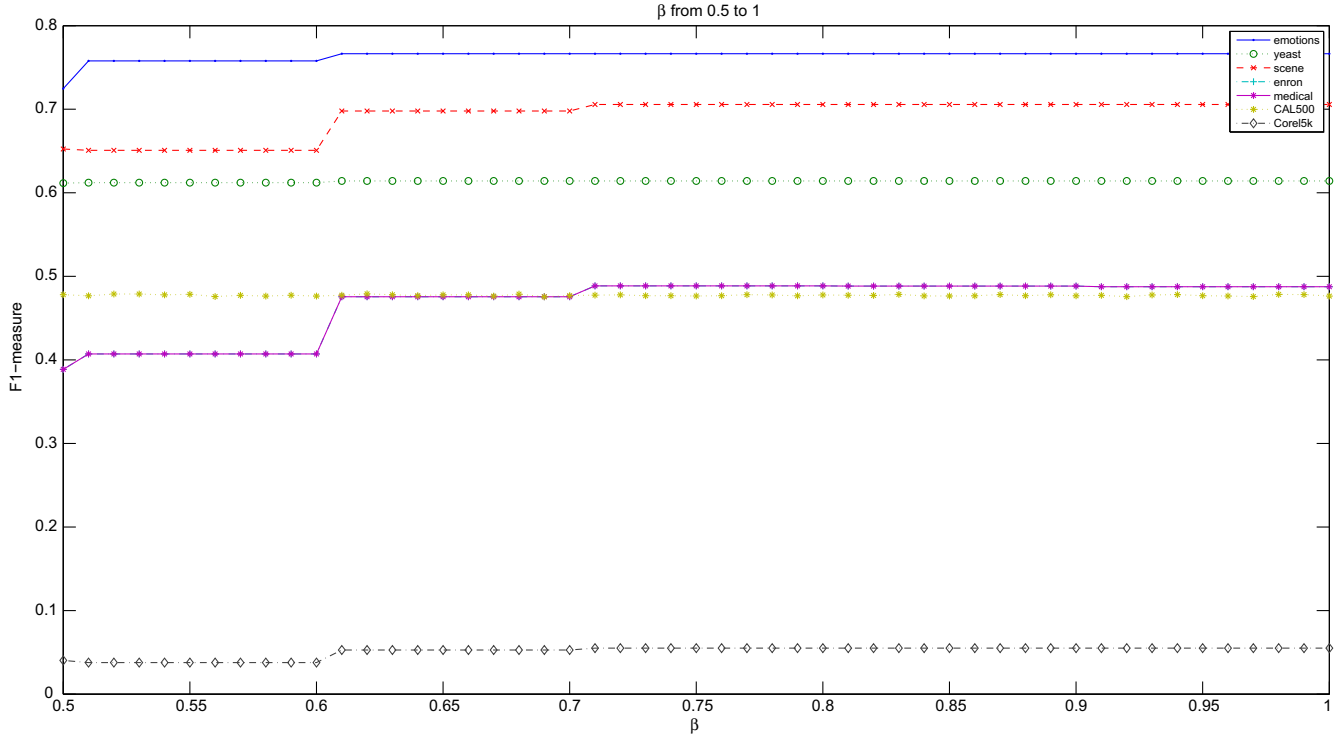


Fig. 16. F1-measure of MLRS for different levels of granularity t.

In order to consider the impact of parameters k and β, we experimented with different parameter values. For convenience, all datasets are split into training sets and testing sets which are same as those presented in *mulan*. Firstly, the number of nearest neighbors k varies from 5 to 60 with step 5 and the inclusion degree β is fixed at 1. From Figs. 5 and 9, we can see that with the increasing of parameter k, both MLRS and MLRS-LC show slowly growing tendency on most datasets in *Hamming-loss* while on Corel5k they perform stably due to the abundant instances of datasets. From Figs. 6–8 and Figs. 10–12, it can be seen that with the increasing

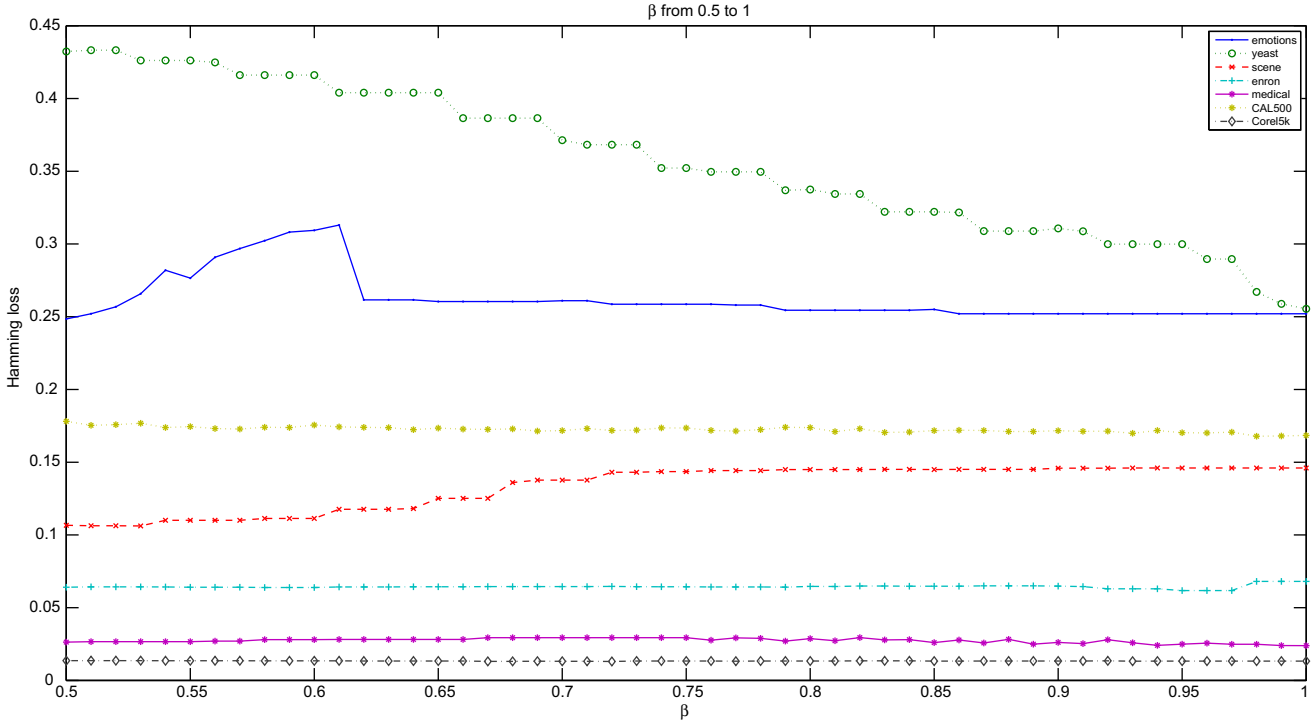


Fig. 17. Hamming-loss of MLRS-LC for different levels of granularity *t*.

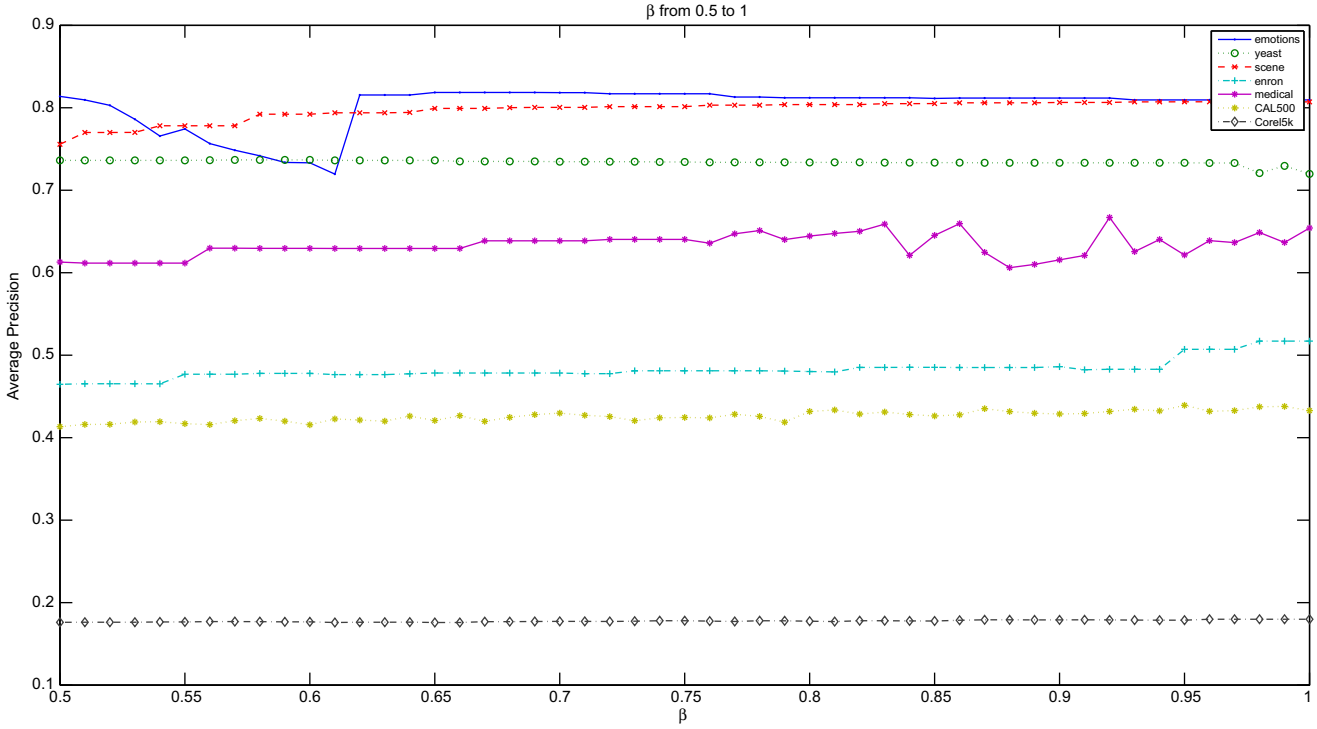


Fig. 18. Average precision of MLRS-LC for different levels of granularity *t*.

of parameter *k*, both MLRS and MLRS-LC generally experience downward trend in *average precision*, *F1-measure* and *accuracy* on all datasets, although different datasets show different fluctuation. The different datasets arriving at optimum performance at different value of parameter *k* states that different datasets have different best levels of granularity *k*. We can select an appropriate value as the level of granularity of for all datasets, such as 10, which can not only reduce the amount of calculation but also obtain a nice performance.

Secondly, the inclusion degree β varies from 0.5 to 1 with step 0.01 and the number of nearest neighbors *k* is fixed at 10. From Figs. 13–16, it can be seen that MLRS presents the

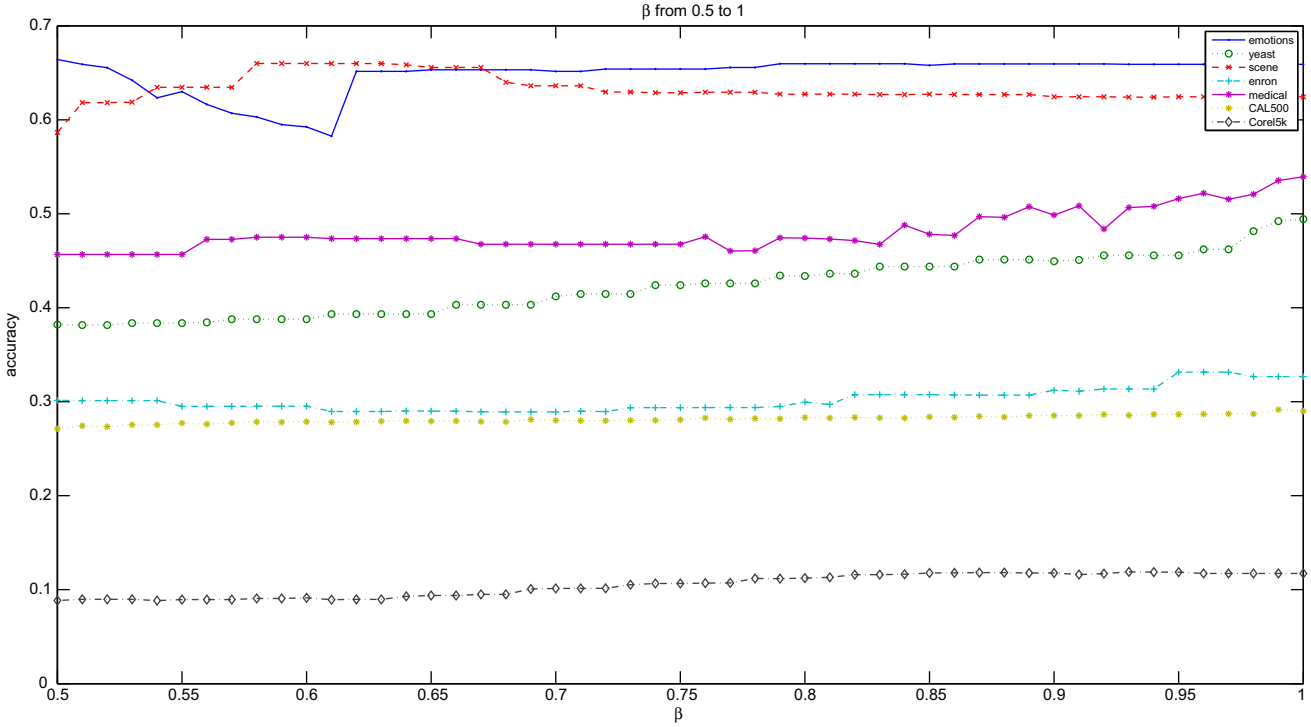


Fig. 19. Accuracy of MLRS-LC for different levels of granularity *t*.

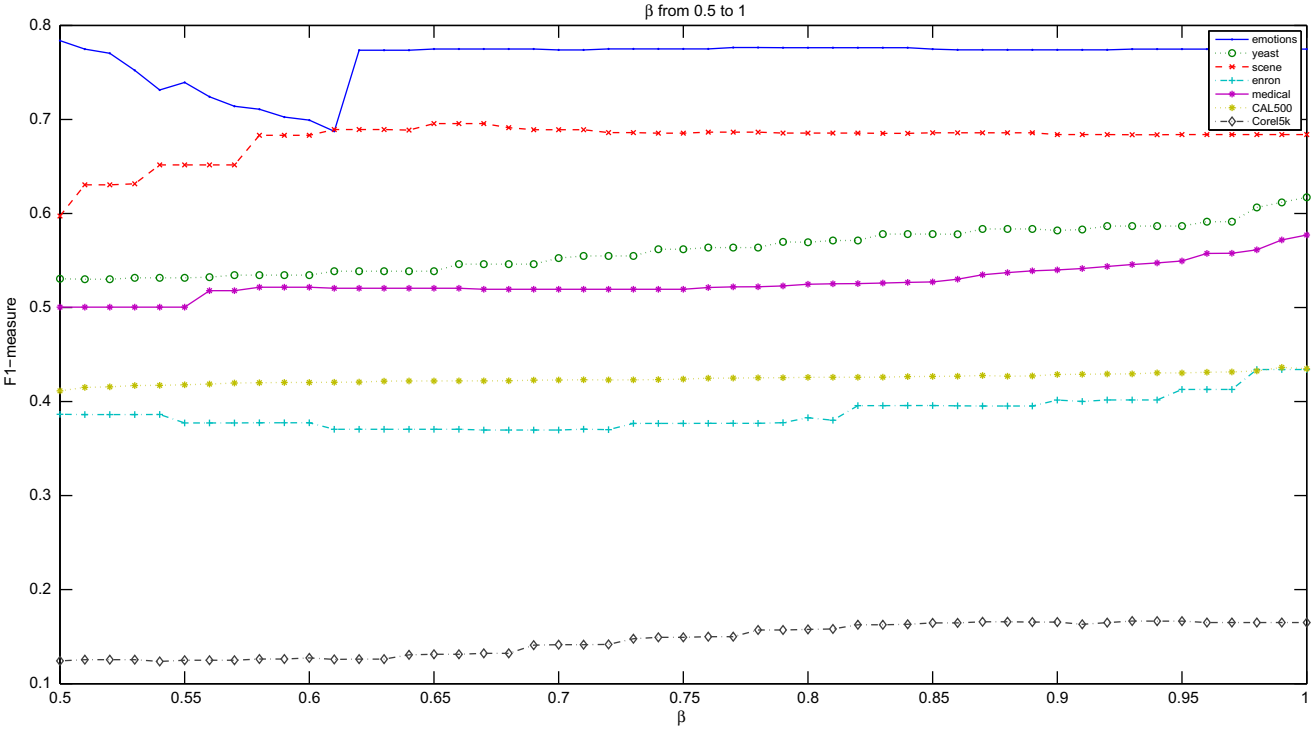


Fig. 20. F1-measure of MLRS-LC for different levels of granularity *t*.

trend of growth overall in *average precision*, *accuracy* and *F1-measure* while performs stably in *Hamming-loss* (Fig. 12). It also can be seen that the performance numbers almost remain unchanged when the inclusion degree β arrives at some value. So we can select an appropriate value, such as 0.9, for inclusion degree β; for MLRS, which can maintain high performance as well as low computational cost. From Fig. 17, it can be seen that MLRS-LC presents a downward trend on most datasets, while from Figs. 18–20, it can be seen that MLRS-LC shows a rising trend on most datasets except for *medical* on which MLRS-LC fluctuated significantly in *average precision* and *accuracy*. We can select an appropriate parameter value for MLRS-LC, such as 0.95, which can reduce the calculation amount while obtain an acceptable

performance. In short, the inclusion degree β has great impact on the performance. That the performances can maintain stability when the inclusion degree β arrives at some value illustrates that both MLRS and MLRS-LC can tolerate the existence of uncertainty.

6.5. Time complexity

As for the time complexity for MLRS and MLRS-LC, we analyze them in two phases. During the process of training, MLRS only need to go through each instance once to obtain the conditional probability $p(l \in y_i | l_j \in y_i)$ and the time complexity is $o(n)$ while MLRS-LC need time to cluster firstly and the time complexity is $o(nkt)$, where n is the number of instances; k is the number of clusters and t is the iterative time. Generally speaking, $k \ll n$ and $t \ll n$. So the time complexity can be regarded as $o(n)$. Then MLRS-LC obtains the correlation $p(l \in y_i | l_j \in y_i)$ in each topic, which need to go through each instance and the total time complexity for MLRS-LC is $o(n)$. During the process of testing, both MLRS and MLRS-LC calculate the neighborhood of testing instance firstly and the time complexity is $o(n \times k)$ where k is the number of nearest neighbors and n is the number of training instances. Then MLRS-LC need time to compute neighborhood in some topic again and the time complexity is $o(n \times k)$ in worst case. So the total time complexity for MLRS-LC is $o(n \times k)$.

7. Conclusions and future work

In this paper, we developed two approaches named MLRS and MLRS-LC for multi-label classification, which we argued have many advantages over more sophisticated methods currently in use. MLRS and MLRS-LC consider the uncertainty existing in the process of classification, which are neglected by existing algorithms. In addition, MLRS and MLRS-LC respectively make use of global and local correlation among labels to improve the precision of prediction, which more comprehensively reflect the influence of correlation. In other words, MLRS and MLRS-LC provide new ideas for the problem of uncertainty and correlation among labels.

Through the empirical multi-label classification evaluation, we compared the proposed methods versus a variety of existing multi-label classification methods. MLRS and MLRS-LC come with high predictive performance while maintaining low computational complexity compared to other more complex methods. The experimental results indicate that it is beneficial for performance to consider the uncertainty and correlation among labels. As for the global and local correlations, it can be seen that none of them always produce better performance better than the other one under different measures. Namely, the performance depends on the nature of the data. However, it should be noted that both MLRS and MLRS-LC do not perform well for high-dimensional datasets. In light of this finding, the future work will focus on the dimensionality reduction for high-dimensional multi-label datasets.

Acknowledgments

The authors would like to thank the Editors for their kindly help and the anonymous referees for their valuable comments and helpful suggestions. The work is partially supported by the National Natural Science Foundation of China (Serial No. 61075056, 61273304, 61075056, 61103067), the Fundamental Research Funds for the Central Universities and the State Scholarship Fund of China (File No. 201206260047).

References

Boutell, M. R., Luo, J., Shen, X., et al. (2004). Learning multi-label scene classification. *Pattern Recognition, 37*(9), 1757–1771.

De Comité, F., Gilleron, R., & Tommasi, M. (2003). Learning multi-label alternating decision trees from texts and data. In *Machine learning and data mining in pattern recognition* (pp. 35–49). Heidelberg: Springer Berlin.

Denœux, T., Younes, Z., & Abdallah, F. (2010). Representing uncertainty on set-valued variables using belief functions. *Artificial Intelligence, 174*(7), 479–499.

Duygulu, P., Barnard, K., de Freitas, J. F. G., et al. (2006). Object recognition as chine translation: Learning a lexicon for a fixed image vocabulary. In *Computer vision-ECCV 2002* (pp. 97–112). Heidelberg: Springer Berlin.

Elisseeff, A., & Weston, J. (2001). A kernel method for multi-labelled classification. In *Advances in neural information processing systems* (pp. 681-687).

Fan, R. E., Lin, C. J. (2007). A study on threshold selection for multi-label classification. Department of Computer Science, National Taiwan University.

Godbole, S., & Sarawagi, S. (2004). Discriminative methods for multi-labeled classification. In *Advances in knowledge discovery and data mining* (pp. 22–30). Heidelberg: Springer Berlin.

Hu, Q., Liu, J., & Yu, D. (2008b). Mixed feature selection based on granulation and approximation. *Knowledge-Based Systems, 21*(4), 294–304.

Hu, Q., Yu, D., & Xie, Z. (2008a). Neighborhood classifiers. *Expert Systems with Applications, 34*(2), 866–876.

Hüllermeier, E., Fürnkranz, J., Cheng, W., et al. (2008). Label ranking by learning pairwise preferences. *Artificial Intelligence, 172*(16), 1897–1916.

Jiang, J. Y., Tsai, S. C., & Lee, S. J. (2012). FSKNN: Multi-label text categorization based on fuzzy similarity and k nearest neighbors. *Expert Systems with Applications, 39*(3), 2813–2821.

Pawlak, Z., Grzymala-Busse, J., Slowinski, R., et al. (1995). Rough sets. *Communications of the ACM, 38*(11), 88–95.

Petrovskiy, M. (2006). Paired comparisons method for solving multi-label learning problem. In *Hybrid intelligent systems, 2006. HIS'06. sixth international conference on. IEEE* (pp. 42-42).

Read, J., Pfahringer, B., Holmes, G., et al. (2009). Classifier chains for multi-label classification. In *Machine learning and knowledge discovery in databases* (pp. 254–269). Heidelberg: Springer Berlin.

Read, J., Pfahringer, B., Holmes, G., et al. (2011). Classifier chains for multi-label classification. *Machine Learning, 85*(3), 333–359.

Schapire, R. E., & Singer, Y. (2000). BoosTexter: A boosting-based system for text categorization. *Machine learning, 39*(2–3), 135–168.

Snoek, C. G. M., Worring, M., Van Gemert, J. C., et al. (2006). he challenge problem for automated detection of 101 semantic concepts in multimedia. In *Proceedings of the 14th annual ACM international conference on multimedia* (pp. 421–430). ACM.

Spyromitros, E., Tsoumakas, G., & Vlahavas, I. (2008). An empirical study of lazy multilabel classification algorithms. In *Artificial Intelligence. Theories, Models and Applications* (pp. 401–406). Heidelberg: Springer Berlin.

Streich, A. P., & Buhmann, J. M. (2008). Classification of multi-labeled data: A generative approach. In *Machine learning and knowledge discovery in databases* (pp. 390–405). Heidelberg: Springer Berlin.

Tsoumakas, K. T. G., Kalliris, G., & Vlahavas, I. (2008). Multi-label classification of music into emotions. In *ISMIR 2008: Proceedings of the 9th international conference of music information retrieval* (pp. 325). Lulu. com.

Tsoumakas, G., Katakis, I., & Vlahavas, I. (2010). Mining multi-label data. In *Data mining and knowledge discovery handbook* (pp. 667–685). US: Springer.

Tsoumakas, G., Katakis, I., & Vlahavas, I. (2011). Random k-labelsets for multilabel classification. *Knowledge and Data Engineering, IEEE Transactions on, 23*(7), 1079–1089.

Tsoumakas, G., & Vlahavas, I. (2007). Random k-labelsets: An ensemble method for multilabel classification. In *Machine learning: ECML 2007* (pp. 406–417). Heidelberg: Springer Berlin.

Tsoumakas, G., Xioufis, E. S., Vilcek, J., et al. (2011). MULAN: A java library for multi-label learning. *Journal of Machine Learning Research, 12*(7), 2411–2414.

Turnbull, D., Barrington, L., Torres, D., et al. (2008). Semantic annotation and retrieval of music and sound effects. *Audio, Speech, and Language Processing, IEEE Transactions on, 16*(2), 467–476.

Vens, C., Struyf, J., Schietgat, L., et al. (2008). Decision trees for hierarchical multi-label classification. *Machine Learning, 73*(2), 185–214.

Witten, I. H., Frank, E., & Hall, M. A. (2011). *Data mining: Practical machine learning tools and techniques*. Elsevier.

Yao, Y. Y. (1998). Relational interpretations of neighborhood operators and rough set approximation operators. *Information Sciences, 111*(1), 239–259.

Yu, Y., Pedrycz, W., & Miao, D. (2013). Neighborhood rough sets based multi-label classification for automatic image annotation. *International Journal of Approximate Reasoning*.

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

Zhang, M. L., & Zhou, Z. H. (2007). ML-KNN: A lazy learning approach to multi-label learning. *Pattern Recognition, 40*(7), 2038–2048.