



**HAL**  
open science

## Editing training data for multi-label classification with the k-nearest neighbor rule

Sawsan Kanj, Fahed Abdallah, Thierry Denoeux, Kifah Tout

► **To cite this version:**

Sawsan Kanj, Fahed Abdallah, Thierry Denoeux, Kifah Tout. Editing training data for multi-label classification with the k-nearest neighbor rule. *Pattern Analysis and Applications*, 2016, 19 (1), pp.145-161. 10.1007/s10044-015-0452-8 . hal-01294269

**HAL Id: hal-01294269**

**<https://hal.science/hal-01294269>**

Submitted on 29 Mar 2016

**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

# Editing training data for multi-label classification with the $k$ -nearest neighbors rule

Sawsan Kanj<sup>a,b</sup>, Fahed Abdallah<sup>a</sup>, Thierry Dencœux<sup>a</sup>,  
Kifah Tout<sup>b</sup>

<sup>a</sup>*Heudiasyc, UMR CNRS 7253, Université de Technologie de Compiègne,  
Rue Roger Couvrolenc,  
CS 60319,  
60203 COMPIEGNE CEDEX,  
FRANCE,  
Tel : (33) 3 44 23 52 15 fax : (33) 3 44 23 44 77  
firstname.lastname@hds.utc.fr*

<sup>b</sup>*AZM center for biotechnology research, EDST, Lebanese University,  
Rue miten, face malaab baladi,  
Tripoli, Lebanon,  
ktout@ul.edu.lb*

---

## Abstract

Multi-label classification allows instances to belong to several classes at once. It has received significant attention in machine learning and has found many real world applications in recent years, such as text categorization, automatic video annotation and functional genomics, resulting in the development of many multi-label classification methods. Based on labelled examples in the training dataset, a multi-labelled method extracts inherent information in order to output a function that predicts the labels of unlabelled data. Due to several problems, like errors in the input vectors or in their labels, this information may be wrong and might lead the multi-label algorithm to fail. In this paper, we propose a simple algorithm for overcoming these problems by editing the existing training dataset, and adapting this edited set with different multi-label classification methods. Evaluation on benchmark datasets demonstrates the usefulness and effectiveness of our approach.

*Key words:* Classification; multi-label;  $k$ -nearest neighbors rule; prototype selection; edition.

---

# 1 Introduction

Multi-label classification is the supervised classification task where each instance can be associated with multiple classes simultaneously from a set of disjoint classes; the classes are then no longer mutually exclusive. Contrary to single-label classification, the multi-label problem is influenced by intrinsic latent correlations between labels, in the sense that the membership of an instance to a class can be helpful to predict its set of labels [44]. For example, a patient with a high blood pressure is more likely to develop heart disease than an other person, but less likely to develop a muscular dystrophy.

Multi-label classification methods have been applied with modern applications like text categorization, where each document can be associated with a set of predefined topics [30]. In bioinformatics, each protein may be labelled with multiple functional labels such as metabolism, energy and cellular biogenesis [17]. In video annotation, a film might be annotated with several labels or tags [25].

Multi-label methods learn usually a classifier function from the training dataset with known class labels. However, real world data often suffer from noisy or erroneous instances due to several problems, like errors in the input vectors or in their labels. To cope with this problem in the framework of single-label learning, several methods based on data reduction have been introduced. These techniques are usually based on prototype selection [4,13,27,39].

Prototype selection methods are usually applied to remove erroneous or redundant instances from the training dataset [13,20,24]. These methods are widely used with the traditional nearest neighbor rule due to their simplicity and effectiveness. In addition to improving classification accuracy for unseen instances, using prototypes dramatically decreases storage and classification-time costs.

However, despite extensive work in multi-label learning [5,21,26,37,42,44], there is a lack of methods for improving the quality of multi-labelled training instances. This fact motivated us to study this problem in the framework of multi-label learning. In this paper, we develop an original method based on a prototype selection using the nearest neighbor rule and a local evaluation criterion, in order to purify training dataset and improve the performance of multi-label classification algorithms. The evaluation criterion used in this paper is the very well known Hamming loss metric. Nevertheless, the proposed method may be straightforwardly adapted to any other criterion. Given one training instance with known set of labels, we consider the editing rule which, based on the Hamming Loss calculated by estimating to this instance a set of predicted labels from the neighborhood, either delete or remain the instance

40 unchanged. After applying this edited rule on all observations of the training  
41 dataset and eliminating the less relevant in the sense of the chosen criterion,  
42 a learning algorithm on the edited training set may be applied efficiently.

43 To show the effectiveness of this method, we apply existing multi-label clas-  
44 sification methods, which are the evidential multi-label  $k$ -nearest neighbor  
45 (EML $k$ NN) [10] and the Rank-SVM [12] methods, on the edited dataset. The  
46 proposed algorithm is applied to several multi-labelled data from different do-  
47 mains using several multi-label classification measures. Even if the Hamming  
48 loss is used as a criterion to edit the data, the performances are evaluated us-  
49 ing several multi-label classification measures. Note that, more than increasing  
50 classification performance, the new method has the advantage of needing less  
51 of storage requirements and decreasing the running time of the initial classi-  
52 fication algorithms.

53 Note that a short paper on the purification (or edition) of multi-labelled  
54 datasets was presented at the conference Fusion [18]. In this paper, the edi-  
55 tion method is more thoroughly interpreted and discussed. Furthermore, we  
56 add the effect of editing on the SVM techniques and we provide an illustra-  
57 tive example on a simulated dataset. In addition, extensive comparisons on  
58 several real world datasets are presented, and the effectiveness of the method  
59 compared to that before editing is shown using statistical tests (t-test and  
60 Friedman test).

61 This paper is structured as follows. Background notions on the nearest neigh-  
62 bor rule in the classical single-label framework and some related techniques for  
63 prototype selection will first be recalled in Section 2. Section 3 will introduce  
64 the principle of multi-label classification and review the EML $k$ NN and Rank-  
65 SVM methods. Our approach will then be exposed in Section 4. Section 5 will  
66 report the experimental evaluation of the presented methods on synthetic and  
67 real-world datasets. Finally, our contribution will be summarized in Section 6.

## 68 **2 Related work on prototype selection for single-labelled data**

69 The problem of noise handling has received considerable attention in the liter-  
70 ature on machine learning. Seeking to start with something relatively simple,  
71 scientists have focused on the nearest neighbor classifier considered as one of  
72 the most well-known technique in machine learning and data mining due to  
73 its simplicity and effectiveness. Given a training set of single-labelled data, the  
74 idea is to select an optimal set of training instances, known as prototypes, in  
75 order to maximize the performances of the Nearest Neighbor (NN) classifier  
76 and/or to minimize the computing time of this classifier [8]. Later, the idea  
77 of selecting "good" instances has also been applied to other types of classi-

78 fiers [4]. In this section, we will rapidly review the nearest neighbor rule, and  
79 give a definition and summary of work related to prototype selection methods  
80 for the NN rule.

## 81 2.1 Nearest Neighbor classification

82 The Nearest Neighbor rule [7] is a well-known and non-parametric decision  
83 procedure for machine learning and data mining tasks. It has been considered  
84 as one of the most effective algorithms in machine learning, and one of the top  
85 ten methods in data mining [13,41]. In traditional supervised learning, this  
86 rule assigns to an unseen sample  $\mathbf{x}$ , the class of the nearest training instance  
87 according to some distance metric. The voting  $k$ -nearest neighbor rule ( $k$ -NN),  
88 with  $k > 1$ , is a generalization of the NN approach where the predicted class of  
89  $\mathbf{x}$  is set as equal to the class represented a majority of its  $k$  nearest neighbors  
90 in the training set.

91 However, the  $k$ -NN rule suffers from several problems such as large storage re-  
92 quirements, high computational complexity in the operational phase, and low  
93 tolerance to noise due to considering all instances as relevant while the train-  
94 ing set may contain noisy or mislabelled examples. Different techniques have  
95 been proposed in the literature to alleviate these problems. One technique,  
96 known as prototype selection, consists of selecting an appropriate subset of  
97 the training data that yields a similar or even higher classification accuracy.  
98 Prototype selection methods can be categorized into three different families.  
99 First, *edition methods* eliminate noisy instances from the original training  
100 set in order to improve classification accuracy. Second, *condensation meth-*  
101 *ods* select a sufficiently small subset of training instances which lead to the  
102 same performance of the single nearest neighbor rule (1-NN), by removing  
103 instances that will not affect classification accuracy. Finally, *hybrid methods*  
104 select a small subset of training instances that incorporates the goals of these  
105 two previous methods [13,3]. In the following, we consider only the editing  
106 methods.

## 107 2.2 Editing methods

108 Editing methods process the training data by removing border and noisy in-  
109 stances or making other necessary cleaning, with the aim of improving classi-  
110 fication accuracy of learning algorithms on test data. Below we review some  
111 algorithms related to the editing approach for the nearest neighbor rule.

112 Wilson proposed the first editing rule [40], called Edited Nearest Neighbor  
113 (ENN), to improve the performance of the 1-NN rule. This method can be

114 described in the following manner. Each instance in the training set is classified  
115 using the  $k$ -NN rule, and it is marked for deletion if its predicted class does  
116 not agree with the true class. Edition is achieved by deleting all misclassified  
117 instances at once. After, any input sample is classified using the 1-NN rule  
118 with the remaining instances. Experiments with the editing rule were reported  
119 by Tomek who proposed two variants of the ENN rule: RENN and All  $k$ -  
120 NN [34]. The Repeated Edited Nearest Neighbor (RENN) rule repeats the  
121 ENN algorithm until a stable set is obtained where no more samples are edited  
122 out. The All  $k$ -NN applies iteratively the ENN algorithm with the  $i$ -NN rule  
123 where  $i$  is going from 1 to  $k$ .

124 In [19], the generalized editing procedure based on the  $kk'$ -NN rule was in-  
125 troduced. The purpose of this procedure was two-fold: improving the level of  
126 performance of the ENN algorithm and reducing the proportion of deleted  
127 samples. Based on the class of a majority of  $k'$  instances from a group of  $k$   
128 nearest samples to an instance  $\mathbf{x}$ , the group of  $k$  samples is either deleted  
129 or relabelled as belonging to the majority class. The 1-NN is then used on  
130 the edited set to classify an input instance. In [11], the authors proposed the  
131 well-known Multiedit algorithm, which randomly breaks the initial training  
132 set into different subsets. In each subset, every instance is classified using the  
133 1-NN rule with the instances in the next subset. Misclassified instances are  
134 discarded. The remaining instances constitute a new set and the algorithm is  
135 iteratively repeated until no more instances are edited out.

136 In [16], a Modified Edited  $k$ -NN rule (MEKNN) was proposed. According to  
137 this rule, a sample  $\mathbf{x}$  is deleted from the initial set if its class does agree with  
138 the class of its  $k$  nearest neighbors and their tying instances (tying instances  
139 are those in the training set that are at the same distance to  $\mathbf{x}$  as its furthest  
140 neighbor). In addition, this method introduces a fixed number of pairs  $(k, k')$ .  
141  $k$  is the number of neighbors to make the edition process and  $k'$  is employed  
142 to classify any new instance in the obtained edited set. The goal was to obtain  
143 the optimal pairs of  $k$  and  $k'$  to employ the final editing reference set.

144 Another method for nearest neighbor editing was proposed in [15]. This method  
145 uses the concept of semi-supervised learning and edits the training instances  
146 by using the whole dataset including: labelled and unlabelled instances. The  
147 proposed method, called NNEAUD (Nearest Neighbor Editing Aided by Un-  
148 labelled Data), consists of two steps: labels are first predicted for unlabelled in-  
149 stances, and the augmented dataset is then used in data editing. The NNEAUD  
150 method uses ENN, RENN, and All $k$ NN algorithms with unlabelled data to  
151 edit the training instances.

## 152 3 Multi-label learning

### 153 3.1 Problem

154 Let  $\mathbb{X}$  denote an instance space, and let  $\mathcal{Y} = \{\omega_1, \dots, \omega_Q\}$  be a finite set of  
155 labels. Let  $\mathcal{D} = \{(\mathbf{x}_1, Y_1), \dots, (\mathbf{x}_n, Y_n)\}$  denote a dataset composed of  $n$  multi-  
156 labeled objects  $(\mathbf{x}_i, Y_i)$ ,  $\mathbf{x}_i \in \mathbb{X}$  and  $Y_i \subseteq \mathcal{Y}$ , where each instance is independent  
157 and identically distributed (i.i.d.) drawn from an unknown distribution. The  
158 goal of multi-label learning is to build a multi-label classifier  $\mathcal{H}$  that maps  
159 an instance  $\mathbf{x}$  to its associated set of labels  $Y$  and optimizes some evaluation  
160 metrics. Here, the set of all subsets of  $\mathcal{Y}$  is the power set of  $\mathcal{Y}$  denoted by  $2^{\mathcal{Y}}$ .

161 Numerous methods have been proposed in the literature to deal with multi-  
162 label learning problems. Existing algorithms can be grouped into three cate-  
163 gories as proposed in [21]: problem transformation approaches, problem adap-  
164 tation algorithms and ensemble methods. The first category divides the multi-  
165 label problem into one or more conventional single-label problems. Binary  
166 Relevance and Label Powerset are two examples of such type of approaches.  
167 The second category generalizes single-label algorithms to cope with multi-  
168 labeled data directly. Examples include boosting [29], decision tree [2] and the  
169 Multi-label  $k$ -nearest neighbors methods [44,47]. Finally, the third category  
170 incorporates the merits of these two previous approaches. Several ensemble  
171 methods have been proposed, among them: ensemble of classifier chains [26],  
172 random  $k$ -label sets [38] and ensemble of multi-label classifiers [32].

### 173 3.2 Performance evaluation in multi-label learning

174 In the traditional single-label classification task, predictive performance is  
175 determined under the traditional *accuracy* measure, where each test instance  
176 can either be correctly or incorrectly classified, and performance is given by the  
177 proportion of correctly classified test instances. In the multi-label classification  
178 task, predictive performance is more complex than that of single-label systems,  
179 where the classification of each test instance can be fully correct, partially  
180 correct or fully wrong. Given a set  $\mathcal{S} = \{(\mathbf{x}_1, Y_1), \dots, (\mathbf{x}_m, Y_m)\}$  of  $m$  test  
181 examples, evaluation metrics can be divided into two groups: *prediction-based*  
182 and *ranking-based* metrics [37]. *Prediction-based* metrics are calculated based  
183 on the comparison between the predicted and the ground truth sets of labels,  
184 while *ranking-based* metrics evaluate the label ranking quality depending on a  
185 scoring function  $f(\cdot, \cdot)$ , ( $f : \mathbb{X} \times \mathcal{Y} \rightarrow \mathbb{R}$ , where  $\mathbb{X}$  is the domain of instances  
186 and  $\mathcal{Y}$  is the set of  $Q$  target classes) that attributes a score to each class in  
187  $\mathcal{Y}$  [44]. More details on evaluation metrics are given in appendix A.

188 We will focus in this paper on the Hamming loss and the Ranking loss metrics.  
 189 The Hamming loss is a prediction-based metric regarded as an average of the  
 190 error rate of the classifier on the  $Q$  binary problems where the decision is  
 191 performed separately [28]. It is defined by:

$$\mathcal{H}Loss = \frac{1}{m} \sum_{i=1}^m \frac{|Y_i \Delta \hat{Y}_i|}{Q}, \quad (1)$$

192 where  $Y_i$  is the ground truth label set for the pattern  $\mathbf{x}_i$ ,  $\hat{Y}_i$  is the predicted  
 193 label set for  $\mathbf{x}_i$  and  $\Delta$  denotes the symmetric difference between two sets.  
 194 In other words, the Hamming loss is based on counting prediction errors (an  
 195 incorrect label is predicted) and missing errors (a true label is not predicted).  
 196 The Ranking loss is a ranking-based metric which evaluates the average frac-  
 197 tion of crucial pairs of labels that are misordered for an instance [29]. The  
 198 *Ranking Loss* is:

$$\mathcal{R}Loss = \frac{1}{m} \sum_{i=1}^m \frac{1}{|Y_i| |\bar{Y}_i|} |R(\mathbf{x}_i)|, \quad (2)$$

199 where  $R(\mathbf{x}_i) = \{(\omega_q, \omega_r) \in Y_i \times \bar{Y}_i \mid f(\mathbf{x}_i, \omega_q) \leq f(\mathbf{x}_i, \omega_r)\}$ ,  $\bar{Y}_i$  denotes the  
 200 complement of  $Y_i$  in  $\mathcal{Y}$ . Smaller values of these metrics correspond to higher  
 201 classification quality. Note that the value of these evaluation criteria is in the  
 202 interval  $[0, 1]$ . We will present briefly in next sections the two multi-label  
 203 algorithms that will be used in this paper and which are the Evidential multi-  
 204 label  $k$ -NN and the Rank-SVM methods.

### 205 3.3 Evidential multi-label $k$ -NN classification

206 The evidential  $k$ -NN (EML $k$ NN) method introduced in [10] answers the multi-  
 207 label classification problems under the belief functions framework and can  
 208 be summarized as follows. Let  $\mathcal{D} = \{(\mathbf{x}_1, A_1, B_1), \dots, (\mathbf{x}_n, A_n, B_n)\}$  be the  
 209 learning set, where  $A_i \subseteq \mathcal{Y} = \{\omega_1, \dots, \omega_Q\}$  denotes a set of classes that surely  
 210 apply to the instance  $\mathbf{x}_i$ , and  $B_i$  is the complement of  $A_i$  in  $\mathcal{Y}$ , ( $\mathcal{Y}$  is known  
 211 as the *frame of discernment* of the problem).

212 To classify an unlabelled instance  $\mathbf{x}$ , we identify its  $k$ -nearest neighbors, deno-  
 213 ted as  $\mathcal{N}_{\mathbf{x}}$ , by computing the distance of  $\mathbf{x}$  to the labelled objects in  $\mathcal{D}$  based  
 214 on a certain distance function. Each element  $\mathbf{x}_i$  in  $\mathcal{N}_{\mathbf{x}}$  constitutes an item of  
 215 evidence regarding the label set of  $\mathbf{x}$ . This item of evidence can be described  
 216 by the following simple two-valued mass function:

$$\begin{aligned} m_i(A_i, B_i) &= \alpha \exp(-\gamma d(\mathbf{x}, \mathbf{x}_i)), \\ m_i(\emptyset, \emptyset) &= 1 - \alpha \exp(-\gamma d(\mathbf{x}, \mathbf{x}_i)), \end{aligned} \quad (3)$$



217 where  $d(\mathbf{x}, \mathbf{x}_i)$  is the distance between  $\mathbf{x}$  and  $\mathbf{x}_i$ ,  $\alpha$  and  $\gamma$  are two parameters,  
 218 such that  $0 < \alpha < 1$  and  $\gamma > 0$ . Parameter  $\alpha$  is usually fixed to a value close  
 219 to 1 such as 0.95 [9], whereas  $\gamma$  can be optimized or fixed heuristically [48].  
 220 If the number of neighbors of the  $\mathbf{x}$  is  $k$ , the resulting  $k$  mass functions are  
 221 combined using the conjunctive rule:

$$m = \bigodot_{i: \mathbf{x}_i \in \mathcal{N}_{\mathbf{x}}} m_i \quad (4)$$

222 where the  $\bigodot$  symbol denotes the unnormalized Dempster’s rule of combi-  
 223 nation [10]. This rule strongly emphasizes the agreement between multiple  
 224 sources, where no elementary item of evidence should be counted twice. The  
 225 predicted multi-label set for  $\mathbf{x}$  is then determined by computing separately for  
 226 each label  $\omega \in \Omega$  two quantities: the degree of belief  $bel(\{\omega\}, \emptyset)$  that the true  
 227 label set  $Y$  contains  $\omega$ , and the degree of belief  $bel(\emptyset, \{\omega\})$  that it does not  
 228 contain  $\omega$ . The multi-label classifier  $\mathcal{H}$  is defined finally as:

$$\mathcal{H}(\mathbf{x}) = \{\omega \in \mathcal{Y} | bel(\{\omega\}, \emptyset) \geq bel(\emptyset, \{\omega\})\}, \quad (5)$$

229 where  $\emptyset$  denotes the empty set of  $\mathcal{Y}$ .

### 230 3.4 Rank-SVM

231 Rank-SVM is a multi-label ranking approach introduced by Elisseeff and We-  
 232 ston in [12]. The ultimate goal was to minimize a criterion measure for multi-  
 233 label learning, called Ranking loss, and to maximize the margin. The authors  
 234 introduce a special multi-label margin defined on  $(\mathbf{x}, Y)$  as the signed distance  
 235 between the instance  $\mathbf{x}$  and the decision boundary. Note that the boundary of  
 236 class  $q$  is a grouping of several boundaries separating the class  $q$  and the other  
 237 classes. For Rank SVM method, which ranks the values of  $r_q(x) = \langle w_q, \mathbf{x} \rangle + b_q$ ,  
 238 the decision boundaries for  $\mathbf{x}$  are defined by the hyperplanes whose equations  
 239 are  $\langle w_q - w_l, \mathbf{x} \rangle + b_q - b_l = 0$ . Thus, the margin with respect to class  $q$  is equal  
 240 to:

$$\min_{(q,l) | (\omega_q, \omega_l) \in (Y \times \bar{Y})} y_q \frac{\langle w_q - w_l, \mathbf{x} \rangle + b_q - b_l}{\|w_q - w_l\|}$$

241 where  $w_q, w_l$  and  $b_q, b_l$  denote the weight vectors and bias terms, and  $y_q$  is a  
 242 binary element equal to +1 if label  $q$  is in  $Y$ , -1 otherwise. According to [12],  
 243  $q$  denotes a relevant label, and  $l$  the irrelevant one. For training instances, it is  
 244 desirable that any relevant label should be ranked higher than any irrelevant  
 245 one.

246 The Rank-SVM model is built from two different sub-systems. The first one,  
 247 named ranking system, orders the labels via a quadratic optimization problem,  
 248 according to their outputs,  $r_q(x) = \langle w_q, \mathbf{x} \rangle + b_q$  for  $q = 1, \dots, Q$ . The other  
 249 goal of this method is to predict a threshold  $t(\mathbf{x})$  and all integer  $q$  such that

250  $r_q(\mathbf{x}) > t(\mathbf{x})$  are considered to belong to the label set  $Y$  of  $\mathbf{x}$ . It is well-known  
251 that such an algorithm can be generalized to non-linear separating boundaries  
252 by just replacing the dot products  $\langle \mathbf{x}_i, \mathbf{x}_j \rangle$  by kernels  $k(\mathbf{x}_i, \mathbf{x}_j)$ .

## 253 4 Editing multi-labelled data using the $k$ -NN rule

### 254 4.1 Motivation

255 In multi-label learning, the goal is to generate a multi-label classifier that will  
256 generalize from a set of multi-labelled training instances in such a way that  
257 classification performances for labelling new data are optimized. However, er-  
258 rors in multi-labelled training datasets can occur for several reasons. One cause  
259 is the subjectivity, when the boundaries of each class are based on individ-  
260 ual perspectives. For example, in genre classification of musical signals, each  
261 musical genre may have its boundaries shifted from person to person [1]. A sec-  
262 ond cause of anomalies or noisy instances is ambiguity during data-entry. For  
263 example, in clinical text for multi-label classification (medical multi-labelled  
264 data collected from Cincinnati children’s hospital medical center), abbrevia-  
265 tions and acronyms used to anonymization of patients may lead to ambiguity  
266 when processing such data by taking more than one sense and having multi-  
267 purposes (in a clinical setting, *FT* can be an abbreviation for *full-term*, *foot*  
268 *test*, *field test*, *full-time* or *family therapy*) [23]. Other errors can arise from  
269 missing information and data transformation or storage. Furthermore, many  
270 examples may have an erroneous set of labels due to an experimental assign-  
271 ment problem or even a human annotation error. To the best of our knowledge,  
272 no algorithm addressing these problems under the multi-label framework has  
273 been proposed so far.

274 In the following, we propose an original method to edit multi-labelled data  
275 by identifying and eliminating erroneous or anomalous samples. The purpose  
276 of this method is three-fold: first, to increase the quality of training instances  
277 assumed to become more reliable; second, to improve the performances of  
278 the classifier built from the resulting training data; and third to increase the  
279 response time of the learning algorithm. This method is based on the  $k$ -nearest  
280 neighbor rule for multi-label classification, and on an evaluation criterion used  
281 locally in the set  $\mathcal{N}_{\mathbf{x}}$  of  $k$ -nearest neighbors of  $\mathbf{x}$  to evaluate the quality of  
282 an instance  $\mathbf{x}$ . Based on this evaluation criterion, we can delete the most  
283 irrelevant, or the *worst* samples from the initial training dataset. We will  
284 present hereafter a simple method using this metric conjointly with a  $k$ -NN  
285 rule for multi-label classification in order to edit the training dataset.

287 Let  $\mathbf{x}$  be an unseen instance for which we wish to estimate the set of labels.  
 288 In the following steps, we describe the proposed method to edit the training  
 289 dataset:

- 290 (1) For each training instance  $\mathbf{x}_i$  in  $\mathcal{D}$ , search for  $\mathcal{N}_{\mathbf{x}_i}$ , the set of its  $k$  nearest  
 291 neighbors;
- 292 (2) Apply a  $k$ -NN based multi-label classifier and calculate a predicted set  
 293 of labels  $\hat{Y}_i$  for  $\mathbf{x}_i$ ;
- 294 (3) For each training instance in  $\mathcal{D}$ , calculate the associated Hamming loss  
 295 given by:

$$\mathcal{H}Loss_i = \frac{|Y_i \Delta \hat{Y}_i|}{Q}; \quad (6)$$

- 296 (4) Estimate the Hamming loss  $\mathcal{H}Loss$ , which is the mean of the associated  
 297 Hamming loss for all instances in  $\mathcal{D}$ :
  - 298 • if  $\mathcal{H}Loss$  is less than a predefined threshold  $t$ , then stop the algorithm;
  - 299 • else,
    - 300 (a) Rank the training instances in  $\mathcal{D}$  with respect to their  $\mathcal{H}Loss_i$  and  
 301 select a subset  $\mathcal{E}^l$  containing  $l$  instances with the higher Hamming  
 302 loss  $\mathcal{H}Loss_i$ ;
    - 303 (b) Update the training set by deleting those in  $\mathcal{E}^l$  :  $\mathcal{D} \leftarrow \mathcal{D} \setminus \mathcal{E}^l$ ;
    - 304 (c) Return to step 1.

305 Note that any  $k$ -NN based multi-label classifier [44,10,46] can be applied in  
 306 Step 2. In this paper, we chose the EML $k$ NN method introduced in Section 3.3.  
 307 According to this method, each element in  $\mathcal{N}_{\mathbf{x}_i}$  represents a piece of knowledge  
 308 about the labelling of  $\mathbf{x}_i$ . A two-valued mass function is then associated to each  
 309 of the  $k$  neighbors in  $\mathcal{N}_{\mathbf{x}_i}$  according to Equation 3. These items of evidence  
 310 are combined to produce a global mass function, using the conjunctive rule  
 311 of Equation 4. In order to estimate the label set for  $\mathbf{x}_i$  denoted by  $\hat{Y}_i$ , the  
 312 global mass function is used according to Equation 5. Intuitively,  $k$  should  
 313 be set to a small value because if  $k$  is high, undesirable instances elimination  
 314 will occur on the boundary between different classes. If  $k$  is equal to 1, the  
 315 EML $k$ NN algorithm boils down to the 1-NN algorithm, and the set of labels  
 316 to be assigned to an example is the same as that of his neighbor. In Step 3,  
 317 one can use other stopping criteria than the general  $\mathcal{H}Loss$ . For example, we  
 318 can stop editing if the Hamming loss associated to each instance is less than  
 319 a predefined threshold  $t$ . We can also substitute the Hamming loss by another  
 320 multi-label metric evaluation. In Steps 4a and 4b, we delete instances with  
 321 high value of  $\mathcal{H}Loss_i$ , which means deleting the *worst* instances with respect  
 322 to a local EML $k$ NN rule. One can add a condition to keep instances belonging  
 323 to classes with low occurrence.

324 **5 Experimental Evaluation**

325 In this section, we present experiment results with synthetic and real-world  
 326 datasets from different domains to demonstrate the effect of edition on the  
 327 performances of the two multi-label classification methods described below.

328 *5.1 Experiments with Synthetic Data*

329 In this section, we will illustrate the behavior of our editing algorithm on  
 330 synthetic datasets using the two methods of classification discussed above.  
 331 The goal of these experiments is to study the effects of edition on multi-label  
 332 learning algorithms.

333 A dataset with three-overlapping classes in two-dimension was first considered.  
 334 The dataset contains 600 instances belonging to three possible labels  $\Omega =$   
 335  $\{\omega_1, \omega_2, \omega_3\}$ . These instances were drawn from seven Gaussian distributions  
 336 with means  $(-5, -5)$ ,  $(5, -5)$ ,  $(0, 5)$ ,  $(0, -5)$ ,  $(-3, 1)$ ,  $(3, 1)$ , and  $(0, 0)$ . The  
 337 standard deviations was equal two for the first three distributions and one for  
 338 the others. We assigned the following classes, respectively, for samples drawn  
 339 from each of these distributions:  $\{\omega_1\}$ ,  $\{\omega_2\}$ ,  $\{\omega_3\}$ ,  $\{\omega_1, \omega_2\}$ ,  $\{\omega_1, \omega_3\}$ ,  $\{\omega_2, \omega_3\}$ ,  
 340  $\{\omega_1, \omega_2, \omega_3\}$ . This dataset was randomly divided into training and test datasets  
 341 with size 400 and 200, respectively. Table 1 gives the distribution of instances  
 342 over the different labels.

Table 1  
 Description of the synthetic data without the erroneous instances.

Label set	Training instances	Testing instances
$\{\omega_1\}$	85	41
$\{\omega_2\}$	84	41
$\{\omega_3\}$	82	46
$\{\omega_1, \omega_2\}$	30	20
$\{\omega_1, \omega_3\}$	42	18
$\{\omega_2, \omega_3\}$	46	23
$\{\omega_1, \omega_2, \omega_3\}$	31	11

343 To test our editing algorithm, 40 instances drawn in the region allocated to  
 344 classes  $\{\omega_1\}$ ,  $\{\omega_2\}$  and  $\{\omega_1, \omega_2\}$  were wrongly assigned to class  $\{\omega_3\}$ . These  
 345 noisy samples are generated randomly from two normal distributions with  
 346 means  $(-4, -6)$  and  $(4, -6)$ , respectively, and a standard deviation equal to  
 347 2. Figure 1 shows the dataset (initial + noisy instances) with their class as-

signments.

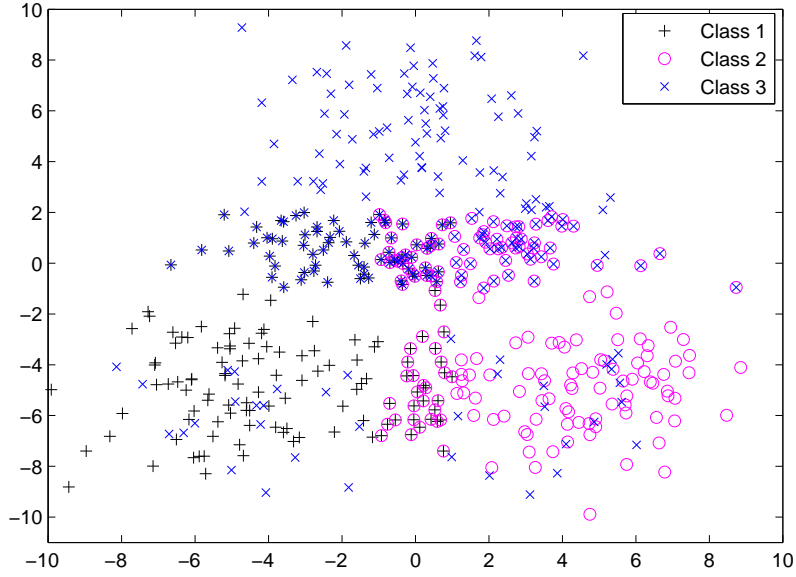
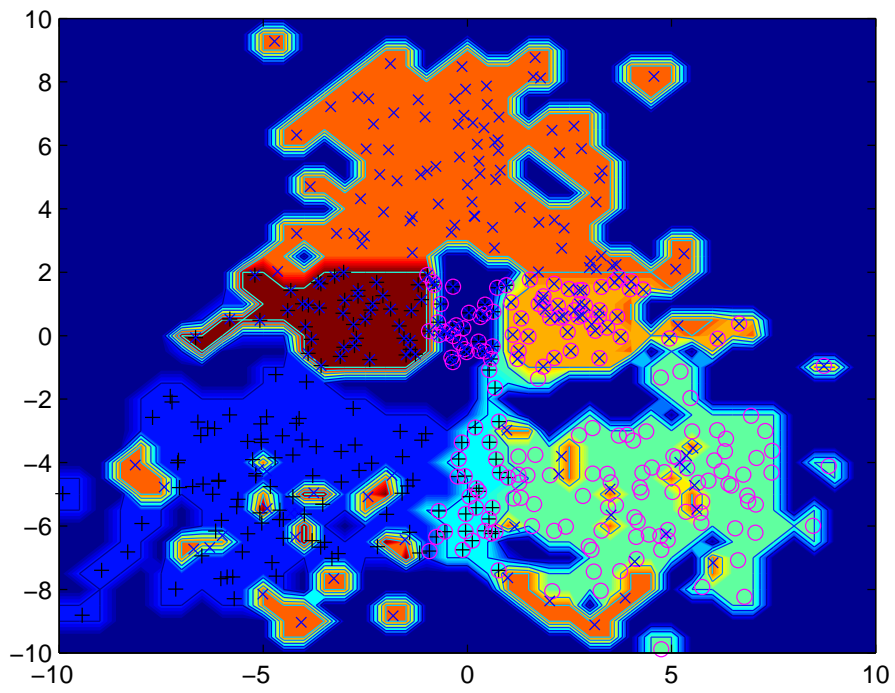


Fig. 1. Training instances of synthetic data

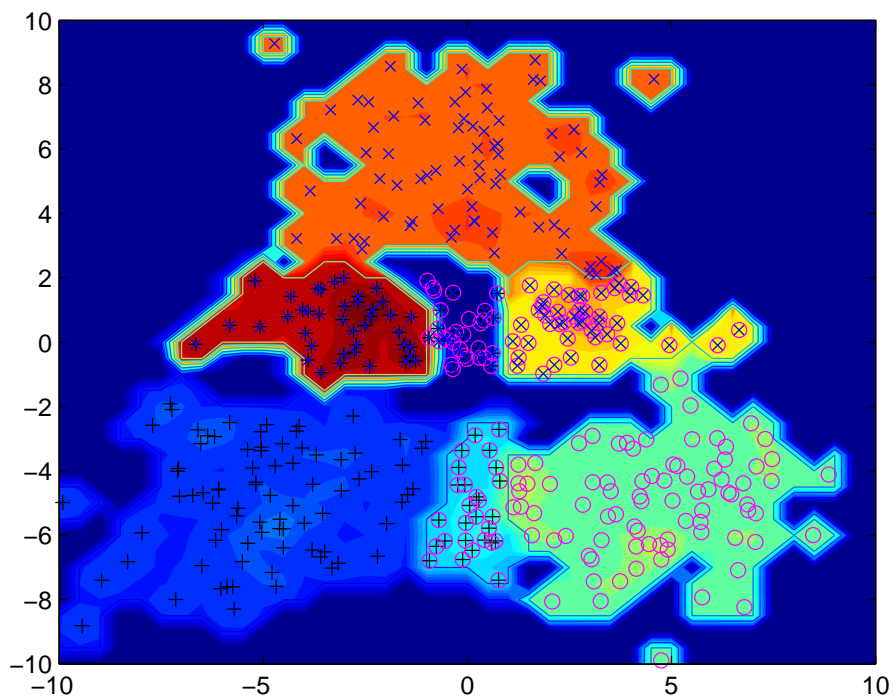
348

349 Figures 2(a) and 2(b) show the decision boundaries for our synthetic data with  
 350 a support vector domain using a Gaussian kernel. The boundary region for  
 351 each class label was drawn using the Rank-SVM method, with the Gaussian  
 352 kernel:  $k(\mathbf{x}, \mathbf{x}') = \exp(-\gamma_r \|\mathbf{x} - \mathbf{x}'\|^2)$ ,  $\gamma_r = 5$ . We used the same parameters  
 353 values for the Rank-SVM method with the training data before and after editing.  
 354 Figure 2(a) shows the decision boundaries for the initial training dataset.  
 355 As we can see, these decision boundaries are significantly influenced by noisy  
 356 instances and there is no clear separation between classes. In the area of class  
 357  $\{\omega_2\}$  (on the right of this figure), we can see several zones belonging to classes  
 358  $\{\omega_1, \omega_2, \omega_3\}$ . Also, in the area of class  $\{\omega_1\}$ , the erroneous instances create  
 359 many zones with instances belonging to class  $\{\omega_3\}$ .

360 In Figure 2(b), we can see the decision boundaries for the same dataset after  
 361 editing. In this figure, we can see that, with editing, noisy instances have been  
 362 removed, and, smoother decision boundaries are produced. The area is now  
 363 divided into seven zones. Instances belonging to class  $\{\omega_1\}$  are on the left of  
 364 Figure 2(b), instances assigned by label  $\{\omega_2\}$  are on the right, and instances  
 365 labelled with  $\{\omega_3\}$  are at the top. Using a geometrical interpretation, we can  
 366 easily distinguish the area belonging to each combination of these classes.  
 367 Instances annotated by three classes are in the middle of this figure. Note  
 368 that the number of training instances was reduced to 382 (initial number of  
 369 training data was 440), and the number of support vectors was decreased  
 370 from 409 to 352. Table 2 reports the experimental results on three evaluation



(a)



(b)

Fig. 2. The Rank-SVM decision boundaries between classes with the training instances, a) before editing, b) after editing.

371 criteria: Hamming loss, accuracy and the F1-measure.

Table 2

Some evaluation measures for the Rank-SVM method before and after the edition of the synthetic dataset.

	Before Editing	After Editing
Hamming loss <sup>-</sup>	0.1667	<b>0.1017</b>
Accuracy <sup>+</sup>	0.7283	<b>0.8358</b>
F1 <sup>+</sup>	0.7422	<b>0.8423</b>

+(-): the higher (smaller) the value, the better the performance.

372 From these two figures, we can observe that training the Rank-SVM method  
373 with a purified dataset leads to smoother separating boundaries, creates ho-  
374 mogeneous clusters and reduces the number of support vectors.

375 Figure 3 shows the performance of our editing approach on the synthetic data  
376 using the EML $k$ NN method. We used from the library of multi-label measures  
377 three evaluation criteria: Hamming loss, accuracy and the F1-measure. The  
378 values of these metrics are shown as a function of the number of neighbors  $k$ .  
379 From this figure, we can observe that when  $k$  takes small values, the EML $k$ NN  
380 algorithm tested on the edited dataset performs better than EML $k$ NN tested  
381 on noisy dataset. As  $k$  increases, the EML $k$ NN method tends to have the same  
382 performance on these two datasets . This can be explained by the fact that,  
383 when increasing the number of neighbors, the effect of randomly erroneous  
384 instances decreases giving that we use more information (coming from more  
385 instances), and also the applied method (EML $k$ NN) is based on an evidential  
386 distance-weighted  $k$ -nearest neighbor rule.

## 387 5.2 Experiments on Real-World Data

388 In this section, we apply the two multi-label classification methods discussed  
389 above (EML $k$ NN and Rank-SVM) to our datasets and we evaluate their per-  
390 formances before and after editing. In the following, we will report the bench-  
391 mark datasets, the evaluation metrics used in our experiments and parameter  
392 settings for edition. Finally, we will provide a discussion of experimental re-  
393 sults.

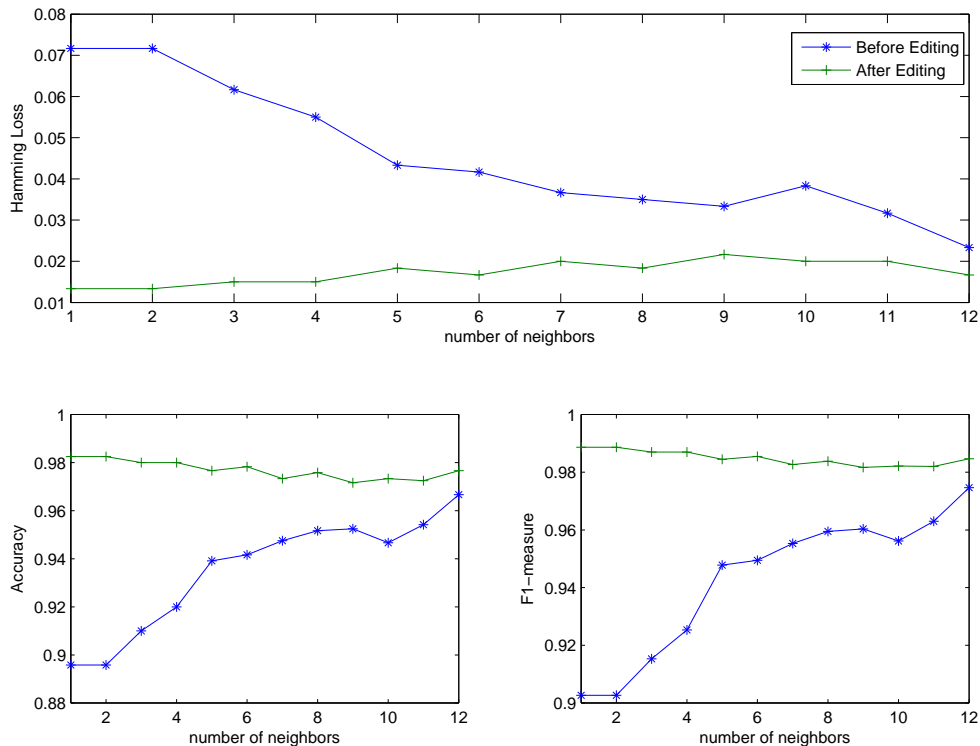


Fig. 3. Some evaluation measures for the EMLkNN method before and after the edition of the synthetic dataset.

### 394 5.2.1 Datasets

395 The datasets<sup>1</sup> that were included in our experiments cover different appli-  
 396 cation domains: multimedia classification (Emotions), bioinformatics (Yeast)  
 397 and text categorization (Medical, Enron and Webpage).

- 398 • *Emotions dataset*. This dataset consists of 593 songs labeled by experts ac-  
 399 cording to the emotions they generated. Each piece of music is described by 8  
 400 rhythmic features and 64 timbre features, and can be annotated with the fol-  
 401 lowing emotions: *amazed-surprised*, *happy-pleased*, *relaxed-calm*, *quiet-still*,  
 402 *sad-lonely* and *angry-fearful*. The average number of labels for each song is  
 403 1.869, and the number of distinct label sets is equal to 27 [35].
- 404 • *Yeast dataset*. The *yeast Saccharomyces cerevisiae* is one of the best stud-  
 405 ied organisms. Each gene is described by the concatenation of micro-array  
 406 expression data and phylogenetic profile and it is associated with a subset  
 407 of 14 functional classes from the Comprehensive Yeast Genome Database

<sup>1</sup> Datasets available at <http://mulan.sourceforge.net/datasets.html>, and <http://cse.seu.edu.cn/people/zhangml/>.



408 of the Munich Information Center for Protein Sequences<sup>2</sup>. This dataset  
 409 contains 2417 genes and 14 possible labels [22].

410 • *Medical dataset*. This dataset contains 978 documents for patient symptom  
 411 histories collected from the *Computational Medicine Center* concerning a  
 412 challenge task on the automated processing of clinical free text. Each doc-  
 413 ument is represented by a vector of 1449 features [23].

414 • *Enron dataset*. The *Enron email*<sup>3</sup> dataset was made public by the Federal  
 415 Energy Regulatory Commission during its investigation. It contains around  
 416 517.431 emails (without attachments) from 151 users distributed in 3500  
 417 folders. Each message includes the senders and the receiver email address,  
 418 date and time, subject, body, text and some other email specific technical  
 419 details. After preprocessing and careful selection of these documents, 53  
 420 different labels are obtained with 753 combinations of distinct label sets [31].

421 • *Webpage categorization dataset*. This dataset were collected from the "ya-  
 422 hoo.com" domain [33]. Eleven different webpage categorization subproblems  
 423 are considered, corresponding to 11 independent multi-label categories: *Arts*  
 424 *and Humanities*, *Business and Economy*, *Computers and Internet*, *Edu-*  
 425 *cation*, *Entertainment*, *Health*, *Recreation and Sports*, *Reference*, *Science*,  
 426 *Social and Science*, and *Society and Culture*. Each subproblem consists of  
 427 5000 documents (2000 as training dataset and 3000 as testing dataset).  
 428 Each webpage was represented as a bag of words and normalized to the  
 429 unit length.

430 Tables 3 and 4 provide an overview of the different characteristics of all ex-  
 431 perimental datasets. These characteristics are explained in the appendix B at  
 432 the end of the article.

Table 3

Characteristics of the Emotions, Yeast, Medical and Enron datasets.

	Domain	Number of instances	Feature vector dimension	Number of labels	Label cardinality	Label density	Distinct label sets
Emotions	music	593	72	6	1.868	0.311	27
Yeast	biology	2417	103	14	4.237	0.303	198
Medical	text	978	1449	45	1.245	0.028	94
Enron	text	1702	1001	53	3.378	0.064	753

### 433 5.3 Parameter Tuning

434 In this section, we comment how to tune different parameters to apply the  
 435 different algorithms described in this paper. We call *editing* parameters those  
 436 applied on the initial dataset with the editing algorithm in order to obtain an  
 437 edited training dataset. We call *testing* parameters those used in a multi-label

<sup>2</sup> <http://mips.gsf.de/genre/proj/yeast/>

<sup>3</sup> <http://enrondata.org/content/research/>

Table 4  
 Characteristics of the Webpage categorization dataset.

	Number of instances	Feature vector dimension	Number of labels	Label cardinality	Label density	Distinct label sets
Arts and Humanities	5000	462	26	1.627	0.063	462
Business and Economy	5000	438	30	1.590	0.053	161
Computers and Internet	5000	681	33	1.487	0.046	253
Education	5000	550	33	1.465	0.044	308
Entertainment	5000	640	21	1.426	0.068	232
Health	5000	612	32	1.667	0.052	257
Recreation and Sports	5000	606	22	1.414	0.065	322
Reference	5000	793	33	1.159	0.035	217
Science	5000	743	40	1.489	0.036	398
Social and Science	5000	1047	39	1.274	0.033	226
Society and Culture	5000	636	27	1.705	0.063	582

438 classification algorithm learnt from initial or edited learning dataset. Note that  
 439 the number  $k$  of neighbors to be used is not necessarily the same as that used  
 440 in the editing algorithm. To avoid confusion, the number of neighbors used in  
 441 the editing algorithm will be noted by  $k'$ . Hereafter, we will show the influence  
 442 of these parameters by using the *Emotions* dataset.

### 443 5.3.1 Editing parameters

444 For the editing algorithm presented in Section 4.2, there are three tunable  
 445 parameters:

- 446 •  $\gamma$ : Parameter used in Equation 3 to scale the distance to each neighbor. It  
 447 was fixed at the best value obtained by cross validation using the EML $k$ NN  
 448 method on the initial training dataset.
- 449 •  $k'$ : Number of neighbors used in the editing algorithm.
- 450 •  $t$ : Threshold used to determine the number  $l$  of instances to delete. We use  
 451 in the simulation a Hamming loss calculated on each instance as in Equation  
 452 (6). This Hamming loss calculated on only one instance will have a value  
 453 equals  $q/Q$ , where  $q \in \{0, \dots, Q\}$ . Note that the value of the parameter  $t$   
 454 to be taken should depend on the global Hamming loss calculated on the  
 455 training dataset.

456 Figure 4 shows the box plot for the Hamming loss metric obtained by the  
 457 EML $k$ NN method on the initial dataset before editing the data for different  
 458 values of  $\gamma$ , where  $k$  was varied from 1 to 12 (thus each box plot corresponds  
 459 to 12 values of the Hamming loss obtained for a given  $\gamma$ ). Figure ?? shows the  
 460 Hamming loss measure obtained as a function of  $t$ , where  $k'$  was varied from 1  
 461 to 12,  $\gamma$  was fixed to 0.1, and  $k$  was fixed to 3 (we can get the same results for  
 462 any value of  $k$ , for that we chose for it a small value). The box plot in Figure

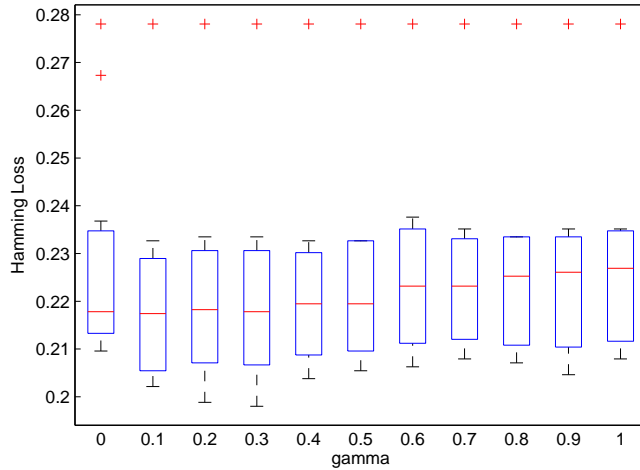


Fig. 4. Hamming loss measure for EML $k$ NN on the initial Emotions training set for different values of  $\gamma$ .

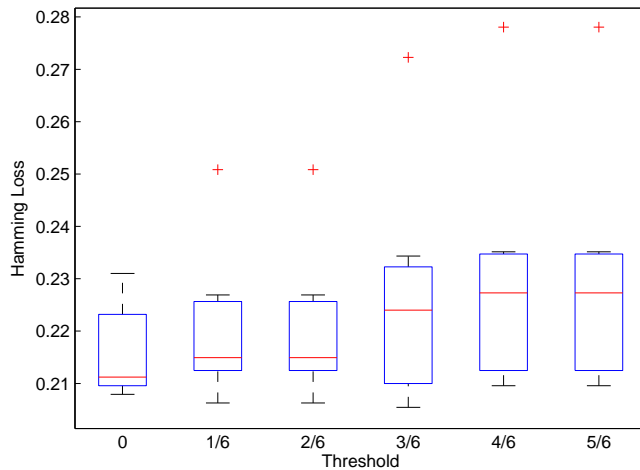


Fig. 5. Hamming loss measure for EML $k$ NN after editing the Emotions training set as a function of  $t$ .

463 6 shows the Hamming loss criterion with respect to the number of neighbors  
 464  $k'$ .  $k$  was varied from 1 to 12, and  $\gamma$  was fixed to 0.1.

### 465 5.3.2 Testing parameters

466 In the testing phase, the EML $k$ NN and the Rank-SVM methods are tested  
 467 with the edited data. EML $k$ NN has two parameters: the number of neighbors  
 468  $k$ , and the discounting parameter  $\gamma$ . These parameters were determined using  
 469 grid search and by focusing on the Hamming loss measure:  $k$  was varied from

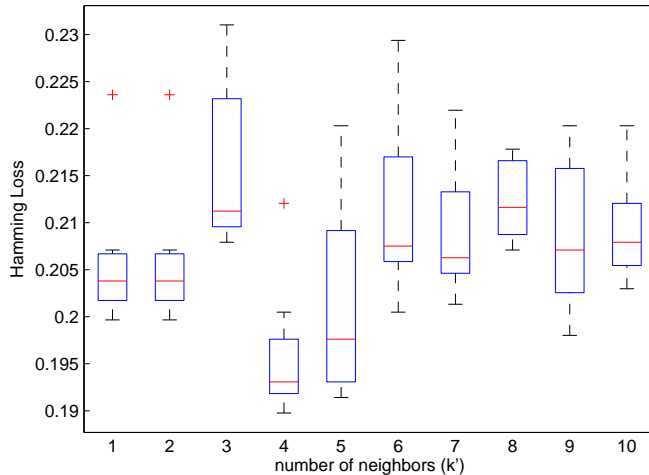


Fig. 6. Hamming loss measure for EML $k$ NN on the edited Emotions training set as a function of  $k'$ .

470 1 to 12, and  $\gamma$  from 0 to 1 with 0.01 steps. Note that the algorithm presented  
 471 in Section 4.2 was repeated only once by taking the best value of  $t$ ; i.e., the  
 472 one that eliminates an important number of erroneous instances at once.

473 For the Rank-SVM method, we used the Gaussian kernel with three tunable  
 474 parameters: kernel scale parameter  $\gamma_r$ , penalty constant  $C$ , and maximal iter-  
 475 ations  $M$ . By focusing on the Hamming loss measure, cross-validation via grid  
 476 search was applied for parameter tuning as explained in [43]. The  $\gamma_R$  and  $C$   
 477 parameters took values from  $2^7, 2^6, \dots$ , to  $2^{-7}$  respectively.  $M$  was set to 50,  
 478 100, 150 and 200.

#### 479 5.4 Results and Discussion

480 In this section, we evaluate the performance of the editing algorithm by com-  
 481 paring the results achieved by the EML $k$ NN and Rank-SVM methods before  
 482 and after editing. Using the optimal parameter values obtained on the train-  
 483 ing datasets, we studied the performance using independent test datasets.  
 484 The experimental results on datasets are given in Tables 5-8. For the webpage  
 485 dataset, the average performance out of the 11 different categorization prob-  
 486 lems is reported in Table 9. The rank of each method is given, and the best  
 487 value on each evaluation criterion is highlighted in bold letters. These results  
 488 can be summarized as follows:

- 489 • From the results in Tables 5-9, we can observe that EML $k$ NN applied on  
 490 edited data improves the performance of the same method on the initial  
 491 dataset for all prediction-based metrics except the Hamming loss measure.

492 The Hamming loss criterion is similar before and after edition.  
 493 • Regarding the Rank-SVM method, results on editing datasets are better  
 494 than those on initial datasets for all measures (prediction-based and ranking-  
 495 based metrics).  
 496 • The Rank-SVM applied on editing datasets gives the best performance ac-  
 497 cording to the majority of evaluation measures for the Yeast, Medical, En-  
 498 ron, and Webpage datasets. For the Emotions dataset, the best performance  
 499 on the ranking-based measures was obtained by the Rank-SVM method  
 500 applied to the edited dataset, while the best results according to predicted-  
 501 based measures were obtained by the EML $k$ NN algorithm applied to the  
 502 edited dataset.

503 In order to show the effect of edition on data storage, Table 10 reports infor-  
 504 mation about the used datasets: the number of instances of full and edited  
 505 training datasets, and the time necessary to editing each of these datasets.  
 506 The results indicate that the edited datasets require less storage space than  
 507 do the initial datasets.

508 In order to study the impact of edition on classification time, we compared  
 509 the time used by each method (programmed in Matlab) applied to the ini-  
 510 tial and edited datasets. Table 11 presents the total running time (learning +  
 511 testing time) using the initial training and edited datasets. We can see that  
 512 the running time of the two classifiers (EML $k$ NN and Rank-SVM) are signif-  
 513 icantly reduced in our experiments, except for the Enron dataset. In general,  
 514 EML $k$ NN is faster than Rank-SVM, due to the space complexity of the Rank-  
 515 SVM method which is proportional to  $n * Q^2$ . The machine used was Intel(R)  
 516 Xeon(R) CPU at 2.67 GHz, 12 GB RAM with Matlab2012a.

517 To statistically measure the significance of performance difference between  
 518 results on initial datasets and those on edited datasets, two statistical in-  
 519 dicators are carried out using ten-fold cross validation: the *pairwise t-tests*  
 520 and the *Friedman test* [14]. The average results of different evaluation crite-  
 521 ria using ten-fold cross validation are reported in Tables 12-16 in which we  
 522 use the *pairwise t-test*. The significance is usually determined under a signif-  
 523 icance level of  $\alpha = 0.05$ . To be able to see *the power* of our conclusions, the  
 524 *p-values* of *Friedman test* and *t-test* on the different datasets are indicated in  
 525 Tables 17-21. Note that small values for the *p-values* indicate strong presump-  
 526 tion against null hypothesis, which is the hypothesis saying that the results  
 527 of the two methods are not different. For the chosen significance level, we can  
 528 consider that the results of the two methods (before and after editing) are  
 529 different if *p-value*  $< 0.05$ . We can see also from the tables that the results  
 530 of the *pairwise t-test* and the *Friedman test* are correlated leading almost to  
 531 similar conclusion about the significance of the difference between methods.

532 The results presented in this section show the advantage of editing multi-label  
533 datasets to improve the performance of multi-label classifiers. By comparing  
534 the performance of multi-label classifiers (EML $k$ NN and Rank-SVM) before  
535 and after edition, we can conclude that editing initial multi-label datasets  
536 improve the performance evaluation of some classifiers. Furthermore, we may  
537 reduce the complexity of classifiers since we need to train less instances, which  
538 are distributed into more homogeneous clusters. We might deduce also that  
539 editing training datasets is a way to reduce the running time complexity of  
540 some multi-label classification methods. Even if we use the Hamming loss  
541 criterion to edit the training datasets, we can get better performance on other  
542 metrics.

543 Note that we tested the use of the edited data set on other multi-label clas-  
544 sifiers, namely the C4.5 based on decision trees [6], and the *MLMLP* based  
545 on neural network [45]. The results we obtained show better performances of  
546 these methods on all the datasets, which follows the behavior of the previ-  
547 ous presented methods (EML $k$ NN and Rank-SVM). We are not showing the  
548 numerical results for a better readability of the paper.

## 549 6 Conclusion

550 In this paper, we have addressed the problem of prototype selection in the  
551 framework of multi-label learning. Although the extensive work in multi-label  
552 classification, to the best of our knowledge, the topic of prototype selection has  
553 not received any attention so far. The goal is not only to optimize performance  
554 of some classifiers, but also the size of the training dataset must be reduced  
555 as well as the computational time of learning algorithms.

556 Edited Nearest Neighbor for Multi-Labelled data is an efficient editing method.  
557 The idea is, first, to classify all training instances using a  $k$ -NN rule, and, sec-  
558 ond, to eliminate erroneous instances based on a local criterion induced from  
559 the Hamming loss measure. The reduced set of instances is then used to classify  
560 unseen instances. We have demonstrated the effect of editing dataset on two  
561 learning algorithms: the EML $k$ NN and the Rank-SVM. This was illustrated  
562 through an example on synthetic data.

563 We applied our algorithm of editing to five real-world datasets from different  
564 domains of application: multimedia classification, bioinformatics and text cat-  
565 egorization. Experimenting with these datasets, we observed that the learning  
566 algorithms (EML $k$ NN and Rank-SVM) with the editing datasets significantly  
567 outperformed the same algorithms on the initial datasets in terms of classi-  
568 fication performance and computational costs. The explanation is that the  
569 editing datasets are distributed in more homogeneous clusters by reducing the

570 number of irrelevant instances. Learning from these new instances is faster  
 571 with better generalization ability.

572 Note that the scalability of the presented approach depends on that of the  
 573  $k$ -NN algorithm. Several approaches exist to adapt the use of the  $k$ -NN on  
 574 large datasets. For example, a simple approach is to use the idea of clustering  
 575 in order to not to compute distances of a testing instance with respect to all  
 576 training instances. It is clear that if the training data is *very noisy*, our method  
 577 of edition may be a solution to reduce the number of training data by using  
 578 only clean instances with a relatively small number with respect to the initial  
 579 dataset.

580 Future research should consider applying the existing algorithm to other appli-  
 581 cation domains, e.g., audio, video and images and showing extensive results on  
 582 several classifiers, including for example decision trees [6], neural networks [45]  
 583 and classifier chains [26], to investigate better the merit of editing in these set-  
 584 tings. A very interesting idea will be to apply the edition on the training set  
 585 using another method than the  $k$ -NN based one. Surely, this idea will not be  
 586 straightforward. For example, if we decide to use the Rank SVM, we have to  
 587 edit instances that are misclassified (since these instances increase the Ham-  
 588 ming Loss), but it is well known that these instances (with non zero Lagrange  
 589 multipliers) are involved in the construction of decision boundaries.

Table 5  
 Experimental results on the Emotions dataset.

	EML $k$ NN		Rank-SVM	
	Before Editing	After Editing	Before Editing	After Editing
Hamming loss <sup>-</sup>	0.209(4)	0.204(2)	0.206(3)	<b>0.194(1)</b>
One-Error <sup>-</sup>	0.287(2)	0.297(3)	0.302(4)	<b>0.277(1)</b>
Coverage <sup>-</sup>	1.881(2)	2.010(4)	1.896(3)	<b>1.847(1)</b>
Ranking loss <sup>-</sup>	0.168(3)	0.220(4)	0.166(2)	<b>0.158(1)</b>
Average Precision <sup>+</sup>	0.7994(2)	0.7959(4)	0.7993(3)	<b>0.8080(1)</b>
Accuracy <sup>+</sup>	0.519(4)	<b>0.569(1)</b>	0.546(3)	0.561(2)
Precision <sup>+</sup>	0.656(3)	<b>0.705(1)</b>	0.651(4)	0.690(2)
Recall <sup>+</sup>	0.592(3)	<b>0.657(1)</b>	0.642(2)	0.642(2)
F1 <sup>+</sup>	0.596(4)	<b>0.648(1)</b>	0.621(3)	0.637(2)

+(-): the higher (smaller) the value, the better the performance.

Table 6  
Experimental results on the Yeast dataset.

	EMLkNN		Rank-SVM	
	Before Editing	After Editing	Before Editing	After Editing
Hamming loss <sup>-</sup>	0.205(4)	0.202(3)	0.197(2)	<b>0.193(1)</b>
One-Error <sup>-</sup>	0.261(4)	0.249(3)	<b>0.221(1)</b>	0.238(2)
Coverage <sup>-</sup>	6.494(3)	6.577(4)	6.424(2)	<b>6.269(1)</b>
Ranking loss <sup>-</sup>	0.188(3)	0.201(4)	0.167(2)	<b>0.165(1)</b>
Average precision <sup>+</sup>	0.751(3)	0.751(4)	0.767(2)	<b>0.768(1)</b>
Accuracy <sup>+</sup>	0.515(4)	0.529(2)	0.522(3)	<b>0.539(1)</b>
Precision <sup>+</sup>	0.685(4)	0.689(3)	0.697(2)	<b>0.703(1)</b>
Recall <sup>+</sup>	0.599(4)	0.618(3)	0.625(3)	<b>0.635(1)</b>
F1 <sup>+</sup>	0.613(4)	0.627(3)	0.628(3)	<b>0.641(1)</b>

+(-): the higher (smaller) the value, the better the performance.

Table 7  
Experimental results on the Medical dataset.

	EMLkNN		Rank-SVM	
	Before Editing	After Editing	Before Editing	After Editing
Hamming loss <sup>-</sup>	0.018(3)	0.018(4)	0.012(2)	<b>0.011(1)</b>
One-Error <sup>-</sup>	0.285(2)	0.291(3)	<b>0.141(1)</b>	<b>0.141(1)</b>
Coverage <sup>-</sup>	3.541(3)	3.661(4)	<b>1.135(1)</b>	1.255(2)
Ranking loss <sup>-</sup>	0.124(3)	0.126(4)	<b>0.015(1)</b>	0.018(2)
Average precision <sup>+</sup>	0.779(3)	0.776(4)	0.897(2)	<b>0.898(1)</b>
Accuracy <sup>+</sup>	0.559(4)	0.585(3)	0.688(2)	<b>0.726(1)</b>
Precision <sup>+</sup>	0.617(4)	0.647(3)	0.744(2)	<b>0.781(1)</b>
Recall <sup>+</sup>	0.569(4)	0.594(3)	0.718(2)	<b>0.754(1)</b>
F1 <sup>+</sup>	0.581(4)	0.608(3)	0.716(2)	<b>0.754(1)</b>

+(-): the higher (smaller) the value, the better the performance.

## 590 A Evaluation measures

591 As discussed in Section 3.2, Performance evaluation for multi-label learning  
592 systems differs from that of single-label classification. Let  $\mathcal{H} : \mathbb{X} \rightarrow 2^{\mathcal{Y}}$  be a  
593 multi-label classifier that assigns a predicted label subset of  $\mathcal{Y} = \{\omega_1, \dots, \omega_Q\}$   
594 to each instance  $\mathbf{x} \in \mathbb{X}$ , and let  $f : \mathbb{X} \times \mathcal{Y} \rightarrow [0, 1]$  be the corresponding scoring  
595 function which gives a score for each label  $\omega_q$  which in turn is interpreted as



Table 8  
Experimental results on the Enron dataset.

	EMLkNN		Rank-SVM	
	Before Editing	After Editing	Before Editing	After Editing
Hamming loss <sup>-</sup>	0.057(3)	0.059(4)	0.055(2)	<b>0.053(1)</b>
One-Error <sup>-</sup>	0.437(3)	0.478(4)	0.287(2)	<b>0.275(1)</b>
Coverage <sup>-</sup>	21.959(3)	26.226(4)	13.758(2)	<b>12.772(1)</b>
Ranking loss <sup>-</sup>	0.261(3)	0.395(4)	0.099(2)	<b>0.090(1)</b>
Average precision <sup>+</sup>	0.568(3)	0.509(4)	0.619(2)	<b>0.647(1)</b>
Accuracy <sup>+</sup>	0.303(4)	0.318(3)	0.398(2)	<b>0.436(1)</b>
Precision <sup>+</sup>	0.473(4)	0.484(3)	0.574(2)	<b>0.587(1)</b>
Recall <sup>+</sup>	0.340(4)	0.359(3)	0.495(2)	<b>0.556(1)</b>
F1 <sup>+</sup>	0.372(4)	0.390(3)	0.511(2)	<b>0.550(1)</b>

+(-): the higher (smaller) the value, the better the performance.

Table 9  
Experimental results on the Webpage dataset.

	EMLkNN		Rank-SVM	
	Before Editing	After Editing	Before Editing	After Editing
Hamming loss <sup>-</sup>	0.065(4)	0.058(3)	0.043(2)	<b>0.042(1)</b>
One-Error <sup>-</sup>	0.589(4)	0.558(3)	0.417(2)	<b>0.402(1)</b>
Coverage <sup>-</sup>	12.164(3)	12.906(4)	5.080(2)	<b>4.169(1)</b>
Ranking loss <sup>-</sup>	0.506(3)	0.545(4)	0.128(2)	<b>0.1000(1)</b>
Average precision <sup>+</sup>	0.471(3)	0.470(4)	0.651(2)	<b>0.673(1)</b>
Accuracy <sup>+</sup>	0.338(4)	0.364(3)	0.402(2)	<b>0.437(1)</b>
Precision <sup>+</sup>	0.393(4)	0.426(3)	0.465(2)	<b>0.508(1)</b>
Recall <sup>+</sup>	0.387(4)	0.391(3)	0.435(2)	<b>0.467(1)</b>
F1 <sup>+</sup>	0.371(4)	0.392(3)	0.432(2)	<b>0.469(1)</b>

+(-): the higher (smaller) the value, the better the performance.

596 the probability that  $\omega_q$  is relevant. The function  $f(., .)$  can be transformed to a  
597 ranking function  $rank_f(., .)$  which maps the outputs of  $f(\mathbf{x}, \omega)$  for any  $\omega \in \mathcal{Y}$   
598 to  $\{\omega_1, \omega_2, \dots, \omega_Q\}$  so that  $f(\mathbf{x}_i, \omega_q) > f(\mathbf{x}_i, \omega_r)$  implies that  $rank_f(\mathbf{x}_i, \omega_q) <$   
599  $rank_f(\mathbf{x}_i, \omega_r)$ .

600 Given a set  $\mathcal{S} = \{(\mathbf{x}_1, Y_1), \dots, (\mathbf{x}_m, Y_m)\}$  of  $m$  test examples, the evaluation  
601 metrics of multi-label learning systems are divided into two groups: *prediction-*  
602 *based* and *ranking-based* metrics. *Prediction-based* measures are calculated

Table 10

Information about the used datasets.

	Number of instances in initial training data	Number of instances in edited training data	Editing Time (seconds)
Emotions	391	113	2.4
Yeast	1500	832	30.8
Medical	645	624	6.2
Enron	1123	861	9.2
Webpage	22000	13693	435.3

Table 11

Running Time (in Seconds) for learning and testing for the two methods.

	EML $k$ NN		Rank-SVM	
	Before Editing	After Editing	Before Editing	After Editing
Emotions	1.4	0.4	162.9	9.9
Yeast	12.4	10.1	$1.1 * 10^4$	$0.2 * 10^4$
Medical	7.3	3.7	$1.6 * 10^4$	$1.4 * 10^4$
Enron	18.8	8.2	$1.0 * 10^4$	$1.6 * 10^4$
Webpage	61.7	47.7	$2.1 * 10^4$ s $\simeq$ 14 h	$8.6 * 10^3$ s $\simeq$ 5.9 h

Table 12

Experimental results (mean $\pm$ std) on the Emotions dataset.

	EML $k$ NN		Rank-SVM	
	Before Editing	After Editing	Before Editing	After Editing
Hamming loss <sup>-</sup>	0.191 $\pm$ 0.019●	0.146 $\pm$ 0.064	0.192 $\pm$ 0.022●	0.148 $\pm$ 0.0474
One-Error <sup>-</sup>	0.266 $\pm$ 0.044●	0.188 $\pm$ 0.100	0.256 $\pm$ 0.081●	0.158 $\pm$ 0.0691
Coverage <sup>-</sup>	1.816 $\pm$ 0.198○	1.524 $\pm$ 0.402	1.702 $\pm$ 0.293○	1.505 $\pm$ 0.4016
Ranking loss <sup>-</sup>	0.173 $\pm$ 0.028○	0.131 $\pm$ 0.061	0.156 $\pm$ 0.039●	0.100 $\pm$ 0.0495
Average precision <sup>+</sup>	0.799 $\pm$ 0.030●	0.864 $\pm$ 0.054	0.807 $\pm$ 0.044●	0.874 $\pm$ 0.0477
Accuracy <sup>+</sup>	0.558 $\pm$ 0.045●	0.681 $\pm$ 0.119	0.541 $\pm$ 0.049●	0.658 $\pm$ 0.0946
Precision <sup>+</sup>	0.688 $\pm$ 0.052●	0.774 $\pm$ 0.095	0.663 $\pm$ 0.065●	0.757 $\pm$ 0.0805
Recall <sup>+</sup>	0.641 $\pm$ 0.051●	0.772 $\pm$ 0.086	0.654 $\pm$ 0.062●	0.768 $\pm$ 0.1017
F1 <sup>+</sup>	0.635 $\pm$ 0.045●	0.748 $\pm$ 0.095	0.626 $\pm$ 0.052●	0.735 $\pm$ 0.0850

●(○): statistically significant (non-significant) difference of performance of the classification algorithm applied on the initial and the edited dataset, based on two-tailed paired t-test at 5% significance.

603 based on the average difference of the actual and the predicted set of la-  
604 bels over all test examples. *Ranking-based* metrics evaluate the label ranking  
605 quality depending on the scoring function  $f(., .)$ .

Table 13

Experimental results (mean±std) on the Yeast dataset.

	EMLkNN		Rank-SVM	
	Before Editing	After Editing	Before Editing	After Editing
Hamming loss <sup>-</sup>	0.201 ± 0.010●	0.168 ± 0.041	0.195 ± 0.006●	0.157 ± 0.0384
One-Error <sup>-</sup>	0.242 ± 0.027●	0.170 ± 0.074	0.216 ± 0.032○	0.165 ± 0.0724
Coverage <sup>-</sup>	6.481 ± 0.263●	5.727 ± 0.800	6.336 ± 0.236●	5.496 ± 0.7416
Ranking loss <sup>-</sup>	0.186 ± 0.015●	0.139 ± 0.053	0.165 ± 0.007●	0.116 ± 0.0451
Average precision <sup>+</sup>	0.757 ± 0.018●	0.813 ± 0.059	0.773 ± 0.013●	0.829 ± 0.0587
Accuracy <sup>+</sup>	0.524 ± 0.021●	0.604 ± 0.074	0.529 ± 0.016●	0.611 ± 0.0763
Precision <sup>+</sup>	0.682 ± 0.024●	0.734 ± 0.060	0.695 ± 0.016●	0.755 ± 0.0571
Recall <sup>+</sup>	0.614 ± 0.023●	0.705 ± 0.076	0.633 ± 0.022●	0.713 ± 0.0795
F1 <sup>+</sup>	0.621 ± 0.021●	0.697 ± 0.068	0.634 ± 0.011●	0.707 ± 0.0692

●(○): statistically significant (non-significant) difference of performance of the classification algorithm applied on the initial and the edited dataset, based on two-tailed paired t-test at 5% significance.

Table 14

Experimental results (mean±std) on the Medical dataset.

	EMLkNN		Rank-SVM	
	Before Editing	After Editing	Before Editing	After Editing
Hamming loss <sup>-</sup>	0.017 ± 0.003○	0.015 ± 0.002	0.011 ± 0.002○	0.010 ± 0.001
One-Error <sup>-</sup>	0.277 ± 0.068○	0.242 ± 0.028	0.137 ± 0.033○	0.118 ± 0.034
Coverage <sup>-</sup>	3.356 ± 1.164○	4.129 ± 0.881	1.070 ± 0.370○	0.864 ± 0.284
Ranking loss <sup>-</sup>	0.108 ± 0.034●	0.151 ± 0.027	0.014 ± 0.007○	0.011 ± 0.005
Average precision <sup>+</sup>	0.784 ± 0.047○	0.796 ± 0.019	0.906 ± 0.022○	0.915 ± 0.021
Accuracy <sup>+</sup>	0.592 ± 0.064●	0.642 ± 0.034	0.719 ± 0.042●	0.769 ± 0.030
Precision <sup>+</sup>	0.654 ± 0.065●	0.705 ± 0.033	0.761 ± 0.049●	0.810 ± 0.034
Recall <sup>+</sup>	0.611 ± 0.061●	0.668 ± 0.038	0.757 ± 0.041●	0.819 ± 0.032
F1 <sup>+</sup>	0.619 ± 0.063●	0.672 ± 0.034	0.746 ± 0.044●	0.799 ± 0.030

●(○): statistically significant (non-significant) difference of performance of the classification algorithm applied on the initial and the edited dataset, based on two-tailed paired t-test at 5% significance.

### 606 A.1 Prediction-based measures

607 **Hamming loss:** The hamming loss metric for the set of labels is defined as  
608 the fraction of labels whose relevance is incorrectly predicted:

$$\mathcal{H}Loss(\mathcal{H}, \mathcal{S}) = \frac{1}{m} \sum_{i=1}^m \frac{|Y_i \Delta \hat{Y}_i|}{Q}, \quad (\text{A.1})$$

609 where  $\Delta$  denotes the symmetric difference between two sets.

610 **Accuracy:** The accuracy metric gives an average degree of similarity between  
611 the predicted and the ground truth label sets:

$$Accuracy(\mathcal{H}, \mathcal{S}) = \frac{1}{m} \sum_{i=1}^m \frac{|Y_i \cap \hat{Y}_i|}{|Y_i \cup \hat{Y}_i|}. \quad (\text{A.2})$$

Table 15

Experimental results (mean $\pm$ std) on the Enron dataset.

	EMLkNN		Rank-SVM	
	Before Editing	After Editing	Before Editing	After Editing
Hamming loss <sup>-</sup>	0.062 $\pm$ 0.010 $\circ$	0.057 $\pm$ 0.004	0.056 $\pm$ 0.014 $\circ$	0.049 $\pm$ 0.003
One-Error <sup>-</sup>	0.571 $\pm$ 0.098 $\bullet$	0.390 $\pm$ 0.075	0.294 $\pm$ 0.111 $\circ$	0.215 $\pm$ 0.049
Coverage <sup>-</sup>	25.645 $\pm$ 6.323 $\circ$	23.242 $\pm$ 2.423	13.937 $\pm$ 3.234 $\bullet$	11.476 $\pm$ 1.167
Ranking loss <sup>-</sup>	0.334 $\pm$ 0.086 $\circ$	0.294 $\pm$ 0.058	0.098 $\pm$ 0.024 $\bullet$	0.072 $\pm$ 0.012
Average precision <sup>+</sup>	0.467 $\pm$ 0.056 $\circ$	0.577 $\pm$ 0.052	0.614 $\pm$ 0.087 $\bullet$	0.702 $\pm$ 0.035
Accuracy <sup>+</sup>	0.165 $\pm$ 0.058 $\bullet$	0.366 $\pm$ 0.048	0.380 $\pm$ 0.125 $\bullet$	0.486 $\pm$ 0.040
Precision <sup>+</sup>	0.334 $\pm$ 0.106 $\bullet$	0.542 $\pm$ 0.059	0.575 $\pm$ 0.097 $\bullet$	0.660 $\pm$ 0.038
Recall <sup>+</sup>	0.184 $\pm$ 0.066 $\bullet$	0.421 $\pm$ 0.054	0.466 $\pm$ 0.110 $\bullet$	0.599 $\pm$ 0.044
F1 <sup>+</sup>	0.219 $\pm$ 0.075 $\bullet$	0.448 $\pm$ 0.054	0.494 $\pm$ 0.105 $\bullet$	0.602 $\pm$ 0.040

$\bullet(\circ)$ : statistically significant (non-significant) difference of performance of the classification algorithm applied on the initial and the edited dataset, based on two-tailed paired t-test at 5% significance.

Table 16

Experimental results (mean $\pm$ std) on the Webpage dataset.

	EMLkNN		Rank-SVM	
	Before Editing	After Editing	Before Editing	After Editing
Hamming loss <sup>-</sup>	0.063 $\pm$ 0.001 $\bullet$	0.050 $\pm$ 0.007	0.042 $\pm$ 0.005	<b>0.041 <math>\pm</math> 0.004<math>\circ</math></b>
One-Error <sup>-</sup>	0.564 $\pm$ 0.007 $\bullet$	0.523 $\pm$ 0.059	0.399 $\pm$ 0.036	<b>0.394 <math>\pm</math> 0.035<math>\circ</math></b>
Coverage <sup>-</sup>	12.124 $\pm$ 0.149 $\bullet$	10.313 $\pm$ 1.531	5.253 $\pm$ 0.397	<b>4.321 <math>\pm</math> 0.687<math>\bullet</math></b>
Ranking loss <sup>-</sup>	0.494 $\pm$ 0.006 $\bullet$	0.415 $\pm$ 0.064	0.132 $\pm$ 0.009	<b>0.108 <math>\pm</math> 0.015<math>\bullet</math></b>
Average precision <sup>+</sup>	0.485 $\pm$ 0.004 $\bullet$	0.532 $\pm$ 0.057	0.631 $\pm$ 0.031	<b>0.678 <math>\pm</math> 0.028<math>\bullet</math></b>
Accuracy <sup>+</sup>	0.359 $\pm$ 0.004 $\circ$	0.372 $\pm$ 0.063	0.402 $\pm$ 0.028	<b>0.440 <math>\pm</math> 0.039<math>\bullet</math></b>
Precision <sup>+</sup>	0.417 $\pm$ 0.004 $\circ$	0.425 $\pm$ 0.061	0.467 $\pm$ 0.032	<b>0.500 <math>\pm</math> 0.037<math>\bullet</math></b>
Recall <sup>+</sup>	0.402 $\pm$ 0.006 $\circ$	0.396 $\pm$ 0.062	0.439 $\pm$ 0.030	<b>0.486 <math>\pm</math> 0.040<math>\bullet</math></b>
F1 <sup>+</sup>	0.391 $\pm$ 0.005 $\circ$	0.397 $\pm$ 0.062	0.434 $\pm$ 0.030	<b>0.474 <math>\pm</math> 0.039<math>\bullet</math></b>

+(-): the higher (smaller) the value, the better the performance.

$\bullet(\circ)$ : statistically significant (non-significant) difference of performance of the classification algorithm applied on the initial and the edited dataset, based on two-tailed paired t-test at 5% significance.

612 **Precision:** The precision metric computes the proportion of true positive  
613 predictions:

$$Precision(\mathcal{H}, \mathcal{S}) = \frac{1}{m} \sum_{i=1}^m \frac{|Y_i \cap \hat{Y}_i|}{|\hat{Y}_i|}. \quad (\text{A.3})$$

614 **Recall:** This metric estimates the proportion of true labels that have been  
615 predicted as positives:

$$Recall(\mathcal{H}, \mathcal{S}) = \frac{1}{m} \sum_{i=1}^m \frac{|Y_i \cap \hat{Y}_i|}{|Y_i|}. \quad (\text{A.4})$$

616 **F1-measure:** F1 measure is defined as the harmonic mean of precision and

Table 17  
*P-values* on the Emotions dataset.

	t-test		Friedman test	
	EML <i>k</i> NN	Rank-SVM	EML <i>k</i> NN	Rank-SVM
Hamming loss <sup>-</sup>	0.048	0.016	0.058	0.206
One-Error <sup>-</sup>	0.037	0.009	0.527	0.011
Coverage <sup>-</sup>	0.054	0.225	0.206	1
Ranking loss <sup>-</sup>	0.061	0.011	0.058	0.206
Average precision <sup>+</sup>	0.004	0.004	0.011	0.011
Accuracy <sup>+</sup>	0.007	0.003	0.011	0.058
Precision <sup>+</sup>	0.021	0.010	0.011	0.011
Recall <sup>+</sup>	0.001	0.007	0.011	0.206
F1 <sup>+</sup>	0.003	0.003	0.011	0.058

+(-): the higher (smaller) the value, the better the performance.

Table 18  
*P-values* on the Yeast dataset.

	t-test		Friedman test	
	EML <i>k</i> NN	Rank-SVM	EML <i>k</i> NN	Rank-SVM
Hamming loss <sup>-</sup>	0.023	0.007	0.206	0.058
One-Error <sup>-</sup>	0.011	0.059	0.206	0.058
Coverage <sup>-</sup>	0.011	0.003	0.058	0.058
Ranking loss <sup>-</sup>	0.015	0.003	0.058	0.051
Average precision <sup>+</sup>	0.010	0.008	0.206	0.058
Accuracy <sup>+</sup>	0.004	0.004	0.058	0.058
Precision <sup>+</sup>	0.020	0.005	0.206	0.011
Recall <sup>+</sup>	0.002	0.007	0.508	0.058
F1 <sup>+</sup>	0.004	0.004	0.058	0.058

+(-): the higher (smaller) the value, the better the performance.

617 recall. It is calculated as:

$$\mathcal{F1}(\mathcal{H}, \mathcal{S}) = \frac{1}{m} \sum_{i=1}^m \frac{2|Y_i \cap \hat{Y}_i|}{|Y_i| + |\hat{Y}_i|}. \quad (\text{A.5})$$

618 Note that the smaller the value of the Hamming loss, the better the perfor-  
619 mance. For the other metrics, higher values correspond to better classification  
620 quality.

Table 19  
*P-values* on the Medical dataset.

	t-test		Friedman test	
	EML <i>k</i> NN	Rank-SVM	EML <i>k</i> NN	Rank-SVM
Hamming loss <sup>-</sup>	0.105	0.136	0.527	0.527
One-Error <sup>-</sup>	0.144	0.224	0.058	1
Coverage <sup>-</sup>	0.111	0.178	0.206	0.058
Ranking loss <sup>-</sup>	0.006	0.325	0.058	0.058
Average precision <sup>+</sup>	0.476	0.336	1	1
Accuracy <sup>+</sup>	0.043	0.007	0.206	0.058
Precision <sup>+</sup>	0.04	0.021	0.058	0.527
Recall <sup>+</sup>	0.022	0.002	0.206	0.011
F1 <sup>+</sup>	0.032	0.005	0.058	0.058

+(-): the higher (smaller) the value, the better the performance.

Table 20  
*P-values* on the Enron dataset.

	t-test		Friedman test	
	EML <i>k</i> NN	Rank-SVM	EML <i>k</i> NN	Rank-SVM
Hamming loss <sup>-</sup>	0.154	0.112	0.058	0.011
One-Error <sup>-</sup>	0	0.054	0.011	0.058
Coverage <sup>-</sup>	0.276	0.036	0.206	0.011
Ranking loss <sup>-</sup>	0.232	0.007	0.206	0.011
Average precision <sup>+</sup>	0	0.008	0.011	0.011
Accuracy <sup>+</sup>	0	0.019	0.002	0.011
Precision <sup>+</sup>	0	0.018	0.011	0.011
Recall <sup>+</sup>	0	0.002	0.002	0.011
F1 <sup>+</sup>	0	0.007	0.001	0.011

+(-): the higher (smaller) the value, the better the performance.

621 *A.2 Ranking-based measures*

622 **One-error:** This metric computes how many times the top-ranked label is  
623 not in the true set of labels of the instance, and it ignores the relevancy of all  
624 other labels.

$$OErr(f, \mathcal{S}) = \frac{1}{m} \sum_{i=1}^m \langle [\arg \max_{\omega \in Y} f(\mathbf{x}_i, \omega)] \notin Y_i \rangle, \quad (\text{A.6})$$

Table 21

*P-values* on the Web dataset.

	t-test		Friedman test	
	EMLkNN	Rank-SVM	EMLkNN	Rank-SVM
Hamming loss <sup>-</sup>	0	0.537	0.002	0.206
One-Error <sup>-</sup>	0.042	0.569	0.206	0.527
Coverage <sup>-</sup>	0.002	0	0.058	0.002
Ranking loss <sup>-</sup>	0.001	0	0.011	0.001
Average precision <sup>+</sup>	0.018	0.028	0.206	0.058
Accuracy <sup>+</sup>	0.500	0.117	0.058	0.058
Precision <sup>+</sup>	0.675	0.281	0.206	0.206
Recall <sup>+</sup>	0.744	0.054	0.011	0.058
F1 <sup>+</sup>	0.757	0.117	0.058	0.058

+(-): the higher (smaller) the value, the better the performance.

625 where for any proposition  $H$ ,  $\langle H \rangle$  equals to 1 if  $H$  holds and 0 otherwise.  
626 Note that, for single-label classification problems, the One Error is identical  
627 to ordinary classification error.

628 **Coverage:** Coverage computes the average of how far we need to move down  
629 the ranked label list in order to cover all the labels assigned to a test instance.

$$Cov(f, \mathcal{S}) = \frac{1}{m} \sum_{i=1}^m \max_{\omega \in Y_i} rank_f(\mathbf{x}_i, \omega) - 1. \quad (\text{A.7})$$

630 **Ranking loss:** This metric computes the number of times that an incorrect  
631 label is ranked higher than a correct label.

$$\mathcal{R}Loss(f, \mathcal{S}) = \frac{1}{m} \sum_{i=1}^m \frac{1}{|Y_i| |\bar{Y}_i|} |(\omega_q, \omega_r) \in Y_i \times \bar{Y}_i \setminus f(\mathbf{x}_i, \omega_q) \leq f(\mathbf{x}_i, \omega_r)| \quad (\text{A.8})$$

632 where  $\bar{Y}_i$  is the complementary set of  $Y_i$  in  $\mathcal{Y}$ .

633 **Average precision:** This metric evaluates the average fraction of labels  
634 ranked above a particular label  $\omega \in Y_i$  which are actually in  $Y_i$ .

$$AvPrec(f, \mathcal{S}) = \frac{1}{m} \sum_{i=1}^m \frac{1}{|Y_i|} \sum_{\omega_q \in Y_i} \frac{|\{\omega_r \in Y_i\} \setminus rank_f(\mathbf{x}_i, \omega_r) \leq rank_f(\mathbf{x}_i, \omega_q)|}{rank_f(\mathbf{x}_i, \omega_q)}. \quad (\text{A.9})$$

635 Note that  $AvPrec(f, \mathcal{S}) = 1$  means that the labels are perfectly ranked. For  
636 the other metrics, smaller values correspond to a better label ranking quality.

## 637 **B Multi-labeled dataset statistics**

638 Given a multi-labeled dataset  $\mathcal{D} = \{(\mathbf{x}_i, Y_i), i = 1, \dots, n\}$  with  $\mathbf{x}_i \in \mathbb{X}$  and  
639  $Y_i \subseteq \mathcal{Y}$ , this dataset can be measured by the number of instances ( $n$ ), the  
640 number of attributes in the input space, and the number of labels ( $Q$ ). In the  
641 following, we review some statistics about the multi-labeled dataset  $\mathcal{D}$  [36].

642 **Label Cardinality:** The Label Cardinality (LCard) of  $\mathcal{D}$  is the average num-  
643 ber of labels per instance. Label cardinality is calculated as

$$LCard(\mathcal{D}) = \frac{1}{n} \sum_{i=1}^n |Y_i| \quad (\text{B.1})$$

644 **Label Density:** The Label Density (LDen) of  $\mathcal{D}$  is defined as the average  
645 number of labels per instance divided by the total number of labels  $Q$ . Label  
646 density is calculated as:

$$LDen(\mathcal{D}) = \frac{1}{n} \sum_{i=1}^n \frac{|Y_i|}{Q} \quad (\text{B.2})$$

647 Both metrics indicate the number of alternative labels that characterize the  
648 examples of a multi-labeled dataset. Label cardinality is independent of the  
649 total number of labels in the classification problem, while label density takes  
650 into consideration the total number of labels. Two datasets with the same label  
651 cardinality but with different label densities may present different properties  
652 that influence the performance of the multi-label classification methods.

653 **Distinct Label sets:** The Distinct Label sets (DL) counts the number of  
654 label sets that are unique across the total number of examples. Distinct label  
655 sets is given by:

$$DL(\mathcal{D}) = |\{Y_i \subseteq \mathcal{Y} | \exists \mathbf{x}_i \in \mathbb{X} : (\mathbf{x}_i, Y_i) \in \mathcal{D}\}| \quad (\text{B.3})$$

656 This measure gives an idea of the regularity of the labeling scheme.

---

<sup>3</sup> [http://www.wormbook.org/chapters/www\\_genomclassprot/genomclassprot.html](http://www.wormbook.org/chapters/www_genomclassprot/genomclassprot.html)



657 **References**

- 658 [1] Barbedo, J.G.A., Lopes, A.: Automatic Genre Classification of Musical Signals.  
659 EURASIP Journal on Advances in Signal Processing **2007**(1), 064,960 (2007).  
660 DOI 10.1155/2007/64960
- 661 [2] Blockeel, H., De Raedt, L., Ramon, J.: Top-down induction of clustering trees.  
662 In: the 15th International Conference on Machine Learning, pp. 55–63. Morgan  
663 Kaufmann (1998)
- 664 [3] Brighton, H., Mellish, C.: Advances in Instance Selection for Instance-Based  
665 Learning Algorithms. Data Mining and Knowledge Discovery **6**(2), 153–172  
666 (2002). DOI 10.1023/A:1014043630878
- 667 [4] Brodley, C.E., Friedl, M.A.: Identifying mislabeled training data. Journal Of  
668 Artificial Intelligence Research **11**, 131–167 (1999). DOI 10.1613/jair.606
- 669 [5] de Carvalho, A., Freitas, A.A.: A Tutorial on Multi-label Classification  
670 Techniques. In: Foundations of Computational Intelligence Volume 5, Studies  
671 in Computational Intelligence, pp. 177–195. Springer Berlin Heidelberg, Berlin,  
672 Heidelberg (2009). DOI 10.1007/978-3-642-01536-6\\_8
- 673 [6] Clare, A., King, R.D.: Knowledge Discovery in Multi-Label Phenotype Data.  
674 In: Proceedings of the 5th European Conference on Principles of Data Mining  
675 and Knowledge Discovery (PKDD '01), vol. 2168, pp. 42–53. Springer-Verlag,  
676 London, UK, Baden-Wurttemberg, Germany (2001)
- 677 [7] Cover, T., Hart, P.: Nearest neighbor pattern classification. IEEE Transactions  
678 on Information Theory **13**(1), 21–27 (1967). DOI 10.1109/TIT.1967.1053964
- 679 [8] Dasarathy, B.V.: Nearest neighbor (NN) norms: NN Pattern Classification  
680 Techniques. IEEE Computer Society Press (1991). DOI ieeecomputersociety.  
681 org/10.1109/2.84880
- 682 [9] Denoeux, T.: A k-nearest neighbor classification rule based on Dempster-Shafer  
683 Theory. IEEE Transactions on Systems, Man, and Cybernetics **25**(05), 804–813  
684 (1995)
- 685 [10] Denoeux, T., Younes, Z., Abdallah, F.: Representing uncertainty on set-valued  
686 variables using belief functions. Artificial Intelligence **174**(7-8), 479–499 (2010).  
687 DOI 10.1016/j.artint.2010.02.002
- 688 [11] Devijver, P.A.: On the editing rate of the Multiedit algorithm. Pattern  
689 Recognition Letters **4**(1), 9–12 (1986). DOI 10.1016/0167-8655(86)90066-8
- 690 [12] Elisseeff, A., Weston, J.: Kernel methods for Multi-labelled classification and  
691 Categorical regression problems. In: Advances in Neural Information Processing  
692 Systems 14, vol. 14, pp. 681–687. Biowulf Technologies, MIT Press (2001)
- 693 [13] García, S., Derrac, J., Cano, J.R., Herrera, F.: Prototype selection for nearest  
694 neighbor classification: taxonomy and empirical study. IEEE transactions on

- 695 pattern analysis and machine intelligence **34**(3), 417–35 (2012). DOI 10.1109/  
696 TPAMI.2011.142
- 697 [14] Garcia, S., Herrera, F.: An Extension on "Statistical Comparisons of Classifiers  
698 over Multiple Data Sets" for all Pairwise Comparisons. *Journal of Machine*  
699 *Learning Research* **9**, 2677–2694 (2008)
- 700 [15] Guan, D., Yuan, W., Lee, Y.K., Lee, S.: Nearest neighbor editing aided by  
701 unlabeled data. *Information Sciences* **179**(13), 2273–2282 (2009). DOI 10.  
702 1016/j.ins.2009.02.011
- 703 [16] Hattori, K., Takahashi, M.: A new edited k-nearest neighbor rule in the pattern  
704 classification problem. *Pattern Recognition* **33**(3), 521–528 (2000). DOI 10.  
705 1016/S0031-3203(99)00068-0
- 706 [17] Jin, B., Muller, B., Zhai, C., Lu, X.: Multi-label literature classification based  
707 on the Gene Ontology graph. *BMC bioinformatics* **9**, 525 (2008). DOI 10.1186/  
708 1471-2105-9-525
- 709 [18] Kanj, S., Abdallah, F., Denoeux, T.: Purifying training data to improve  
710 performance of multi-label classification algorithms. In: *Proceedings of the*  
711 *15th Int. Conf. on Information Fusion (FUSION 2012)*, pp. 1784–1792. IEEE,  
712 Singapore (2012)
- 713 [19] Koplowitz, J., Brown, T.A.: On the relation of performance to editing in nearest  
714 neighbor rules. *Pattern Recognition* **13**(3), 251–255 (1981). DOI 10.1016/  
715 0031-3203(81)90102-3
- 716 [20] Li, Y., Hu, Z., Cai, Y., Zhang, W.: Support vector based prototype selection  
717 method for nearest neighbor rules. In: *Proceedings of the first international*  
718 *conference on Advances in Natural Computation*, pp. 528–535. Springer Berlin  
719 Heidelberg, Changsha, China (2005). DOI 10.1007/11539087\\_68
- 720 [21] Madjarov, G., Kocev, D., Gjorgjevikj, D., Džeroski, S.: An extensive  
721 experimental comparison of methods for multi-label learning. *Pattern*  
722 *Recognition* **45**(9), 3084–3104 (2012). DOI 10.1016/j.patcog.2012.03.004
- 723 [22] Pavlidis, P., Grundy, W.N.: Combining microarray expression data and  
724 phylogenetic profiles to learn gene functional categories using support vector  
725 machines. Tech. rep., Department of Computer Science, Columbia University,  
726 New York (2000)
- 727 [23] Pestian, J.P., Brew, C., Matykiewicz, P., Hovermale, D.J., Johnson, N., Cohen,  
728 K.B., Duch, W.: A Shared Task Involving Multi-label Classification of Clinical  
729 Free Text. In: *Proceedings of the Workshop on BioNLP 2007: Biological,*  
730 *Translational, and Clinical Language Processing (BioNLP '07)*, vol. 1, pp. 97–  
731 104. Association for Computational Linguistics, Prague, Czech Republic (2007)
- 732 [24] Pkalska, E., Duin, R.P., Paclík, P.: Prototype selection for dissimilarity-based  
733 classifiers. *Pattern Recognition* **39**(2), 189–208 (2006). DOI 10.1016/j.patcog.  
734 2005.06.012

- 735 [25] Qi, G.J., Hua, X.S., Rui, Y., Tang, J., Mei, T., Zhang, H.J.: Correlative multi-  
736 label video annotation. In: Proceedings of the 15th international conference  
737 on Multimedia - MULTIMEDIA '07, p. 17. ACM Press, Augsburg, Germany  
738 (2007). DOI 10.1145/1291233.1291245
- 739 [26] Read, J., Pfahringer, B., Holmes, G., Frank, E.: Classifier chains for multi-  
740 label classification. *Machine Learning* **85**(3), 333–359 (2011). DOI 10.1007/  
741 s10994-011-5256-5
- 742 [27] Sánchez, J., Pla, F., Ferri, F.: Prototype selection for the nearest neighbour rule  
743 through proximity graphs. *Pattern Recognition Letters* **18**(6), 507–513 (1997).  
744 DOI 10.1016/S0167-8655(97)00035-4
- 745 [28] Schapire, R.E., Singer, Y.: Improved Boosting Algorithms Using Confidence-  
746 rated Predictions. *Machine learning* **37**(3), 297–336 (1999). DOI 10.1023/A:  
747 1007614523901
- 748 [29] Schapire, R.E., Singer, Y.: BoosTexter : A Boosting-based System for Text  
749 Categorization. *Machine Learning* **39**(2-3), 135–168 (2000). DOI 10.1023/A:  
750 1007649029923
- 751 [30] Sebastiani, F.: Machine learning in automated text categorization. *ACM*  
752 *Computing Surveys* **34**(1), 1–47 (2002). DOI 10.1145/505282.505283
- 753 [31] Shetty, J., Adibi, J.: The Enron Email Dataset Database Schema and Brief  
754 Statistical Report. Tech. rep., Information Sciences Institute Technical Report,  
755 University of Southern California (2004)
- 756 [32] Tahir, M.A., Kittler, J., Bouridane, A.: Multilabel classification using  
757 heterogeneous ensemble of multi-label classifiers. *Pattern Recognition Letters*  
758 **33**(5), 513–523 (2012). DOI 10.1016/j.patrec.2011.10.019
- 759 [33] Tang, L., Rajan, S., Narayanan, V.K.: Large scale multi-label classification via  
760 metalabeler. In: Proceedings of the 18th international conference on World  
761 wide web (WWW '09), p. 211. ACM Press, Madrid, Spain (2009). DOI 10.  
762 1145/1526709.1526738
- 763 [34] Tomek, I.: Two Modifications of CNN. *IEEE Transactions on Systems, Man,*  
764 *and Cybernetics* **6**(11), 769–772 (1976). DOI 10.1109/TSMC.1976.4309452
- 765 [35] Trohidis, K., Tsoumakas, G., Kalliris, G., Vlahavas, I.: Multi-label classification  
766 of music into emotions. In: Proceedings of the 9th International Conference on  
767 Music Information Retrieval (ISMIR '08), pp. 325–330. Philadelphia, PA, USA  
768 (2008)
- 769 [36] Tsoumakas, G., Katakis, I.: Multi-Label Classification : An Overview.  
770 *International Journal of data warehousing & mining* **3**(3), 1–13 (2007)
- 771 [37] Tsoumakas, G., Katakis, I., Vlahavas, I.: Mining Multi-label Data. In:  
772 O. Maimon, L. Rokach (eds.) *Data Mining and Knowledge Discovery Handbook*,  
773 pp. 667–685. Springer US, Thessaloniki, Greece (2010). DOI 10.1007/  
774 978-0-387-09823-4\_34

- 775 [38] Tsoumakas, G., Katakis, I., Vlahavas, I.: Random k-Labelsets for Multilabel  
776 Classification. *IEEE Transactions on Knowledge and Data Engineering* **23**(7),  
777 1079–1089 (2011). DOI 10.1109/TKDE.2010.164
- 778 [39] Van Hulse, J., Khoshgoftaar, T.: Knowledge discovery from imbalanced and  
779 noisy data. *Data & Knowledge Engineering* **68**(12), 1513–1542 (2009). DOI  
780 10.1016/j.datak.2009.08.005
- 781 [40] Wilson, D.L.: Asymptotic Properties of Nearest Neighbor Rules Using Edited  
782 Data. *IEEE Transactions on Systems, Man, and Cybernetics* **2**(3), 408–421  
783 (1972). DOI 10.1109/TSMC.1972.4309137
- 784 [41] Wu, X., Kumar, V., Ross Quinlan, J., Ghosh, J., Yang, Q., Motoda, H.,  
785 McLachlan, G.J., Ng, A., Liu, B., Yu, P.S., Zhou, Z.H., Steinbach, M., Hand,  
786 D.J., Steinberg, D.: Top 10 algorithms in data mining. *Knowledge and  
787 Information Systems* **14**(1), 1–37 (2007). DOI 10.1007/s10115-007-0114-2
- 788 [42] Xu, J.: An extended one-versus-rest support vector machine for multi-label  
789 classification. *Neurocomputing* **74**(17), 3114–3124 (2011). DOI 10.1016/j.  
790 neucom.2011.04.024
- 791 [43] Xu, J.: An efficient multi-label support vector machine with a zero label. *Expert  
792 Systems with Applications* **39**(5), 4796–4804 (2012). DOI 10.1016/j.eswa.2011.  
793 09.138
- 794 [44] Younes, Z., Abdallah, F., Denoeux, T., Snoussi, H.: A Dependent Multilabel  
795 Classification Method Derived from the -Nearest Neighbor Rule. *EURASIP  
796 Journal on Advances in Signal Processing* **2011**(1), 645,964 (2011). DOI 10.  
797 1155/2011/645964
- 798 [45] Zhang, M.L., Zhou, Z.H.: Multilabel neural networks with applications to  
799 functional genomics and text categorization. *IEEE Trans. on Knowl. and Data  
800 Eng.* **18**(10), 1338–1351 (2006). DOI 10.1109/TKDE.2006.162
- 801 [46] Zhang, M.L., Zhou, Z.h.: ML-KNN: A lazy learning approach to multi-label  
802 learning. *Pattern Recognition* **40**(7), 2038–2048 (2007). DOI 10.1016/j.patcog.  
803 2006.12.019
- 804 [47] Zhou, Z.H., Zhang, M.L.: Multi-instance multi-label learning with application to  
805 scene classification. In: the Twenty-Second Conference on Artificial Intelligence  
806 (AAAI ), vol. 40, pp. 1609–1616. MIT Press, Vancouver, Canada (2007). DOI  
807 10.1016/j.patcog.2006.12.019
- 808 [48] Zouhal, L., Denoeux, T.: An evidence-theoretic k-NN rule with parameter  
809 optimization. *IEEE Transactions on Systems, Man and Cybernetics, Part C  
810 (Applications and Reviews)* **28**(2), 263–271 (1998). DOI 10.1109/5326.669565