Feature Ranking with Relief for Multi-label Classification: Does Distance Matter?

Matej Petković[1,2(✉)], Dragi Kocev[1,2], and Sašo Džeroski[1,2]

[1] Jožef Stefan Institute, Jamova 39, 1000 Ljubljana, Slovenia
[2] Jožef Stefan Postgraduate School, Jamova 39, 1000 Ljubljana, Slovenia
{matej.petkovic,dragi.kocev,saso.dzeroski}@ijs.si

Abstract. In this work, we address the task of feature ranking for multi-label classification (MLC). The task of MLC is to predict which labels from a maximal predefined label set are relevant for a given example. We focus on the Relief family of feature ranking algorithms and empirically show that the definition of the distances in the target space used within Relief should depend on the evaluation measure used to assess the performance of MLC algorithms. By considering different such measures, we improve over the currently available MLC Relief algorithm. We extensively evaluate the resulting MLC ranking approaches on 24 benchmark MLC datasets, using different evaluation measures of MLC performance. The results additionally identify the mechanisms of influence of the parameters of Relief on the quality of the rankings.

Keywords: Feature ranking · Multi-label classification · Relief

1 Introduction

Classification is a task in predictive modelling, where the goal is to learn a model that takes as the input a vector $\boldsymbol{x}$ of descriptive variables (features) x_i, and predicts the class value y that a given example belongs to. If y can take two different values, the task at hand is referred to as binary classification. Otherwise (y can take more than two values), the task at hand is multi-class classification. In both cases, every example is assigned precisely one value. For example, one can predict whether a person has survived a shipwreck where $y \in \{\texttt{yes}, \texttt{no}\}$ (binary), or what is the blood type of a person where $y \in \{\texttt{A}, \texttt{B}, \texttt{AB}, \texttt{0}\}$ (multi-class). In both cases, class values are mutually exclusive.

A related task is multi-label classification (MLC). As opposed to the standard classification, a MLC predictive model predicts which labels from a predefined

We acknowledge the financial supported of the Slovenian Research Agency via the grants P2-0103, J4-7362, L2-7509, N2-0056, and a young researcher grant to MP, the European Commission, through the grants HBP (The Human Brain Project), SGA2, and LANDMARK, and ARVALIS (project BIODIV).

L. Soldatova et al. (Eds.): DS 2018, LNAI 11198, pp. 51–65, 2018.
https://doi.org/10.1007/978-3-030-01771-2_4

set $\mathscr{L}$ are *relevant* for a given example. For example, one can predict which of the genres from the set $\mathscr{L} = \{\texttt{romance}, \texttt{drama}, \texttt{comedy}\}$ are relevant for a given film. Clearly, a film can be `drama` and `comedy` at the same time.

There are two main approaches to MLC: problem transformation and algorithm adaptation. From the problem transformation group of methods most widely known are binary relevance and label power set. Binary relevance is a simple method that converts a MLC task to several binary classification tasks with $y \in \{\texttt{yes}, \texttt{no}\}$ where we predict the relevance of each label separately. This approach is often criticized for it cannot make use of the interactions among the labels. In the label power set approach [24], the task of predicting a subset of $\mathscr{L}$ is converted to the task of predicting an element of the power set $2^{\mathscr{L}}$, and thus converting a MLC task to multi-class classification task. However, the number of classes can be as high as $2^{|\mathscr{L}|}$, which results in a very sparse dataset.

The second group of methods are method transformation techniques where an existing method is adapted to a new problem. A prominent member of this group are predictive clustering trees which generalize decision trees, so that they can handle MLC [15] and other structure output prediction tasks [12].

Another important task in machine learning is feature ranking, where the goal is to asses the importance of every descriptive attribute (feature) by using some scoring function. The output of a feature ranking algorithm is a list of features that is sorted with respect to the scores.

Feature ranking is typically considered a part of data preprocessing, since it can be used to reduce the dimensionality of the input space, so that only the features that contain the most information about labels (or target(s) in general) are kept in the dataset. By doing this, we decrease the computational cost of building a predictive model, while the performance of the model is not degraded. Another reason to compute a feature ranking is that dimensionality reduction typically results in models that are easier to understand, which is useful when a machine learning expert works in collaboration with a domain expert. Predictive models, such as decision trees, are easier to interpret when a small number of the most relevant features are used to learn them.

There is a plethora of feature ranking methods for the task of classification [22]. A possible approach to MLC feature ranking is to adapt the binary relevance approach from predictive modelling, where at the first stage, feature importances are computed for every label $\ell \in \mathscr{L}$ separately as in the classification case. After that, the feature importances are averaged over the different labels and a single ranking is returned. In this work, we focus on the Relief family of feature ranking algorithms, which are distance based approaches and thus widely applicable. They are part of the *filter methods* which compute the ranking without any additional predictive model [9]. The filters are typically fast, i.e., linear in the number of features, but myopic at the same time, i.e., cannot capture the feature interaction. Relief family of the feature ranking algorithms, however, overcomes this, and can successfully discover, e.g., XOR-relation [14].

The rest of the paper is organized as follows. In Sect. 2, the overview of related work is given. In Sect. 3, the proposed feature ranking algorithms are described and analyzed. In Sect. 4, the detailed description of the experimental is given. In Sect. 5, the results of the experiments are presented. In Sect. 6, conclusions and direction for further work are given.

2 Related Work

We start the overview of the related work with the extensions of the Relief family to MLC setting that are presented in [20]. There, the binary relevance and label power set approach were applied to the feature ranking scenario. More precisely, in the case of binary relevance approach, feature ranking was computed for every label $\ell \in \mathscr{L}$ separately. This was done by using the Relief algorithm for the standard binary classification [11]. After that, the feature importances were averaged to a single score. In the case of the label power set approach the multi-class extension of the Relief (ReliefF) was used [14].

As mentioned before, these two approaches have some drawbacks. Binary relevance approach does not take the label interactions into account and can be expensive to run when the number of labels is high: we have to solve $|\mathscr{L}|$ feature ranking problems which results in high time or space complexity. High space complexity is also a drawback of label power set approach if the number of different relevant label subsets is high. In that case, the data may also become too sparse for the ranking to be relevant.

Both procedures were evaluated on a rather small subset of ten datasets presented in this study (see Sect. 4.2), in a manner similar to our evaluation procedure, which uses k nearest neighbours classifier. No statistical tests were done and the feature rankings were not compared to any baseline.

Another data transformation approach was presented in [13] where the MLC problem is transformed into $|\mathscr{L}|(|\mathscr{L}|-1)/2$ binary classification problems - one for each of the label pairs (ℓ_1, ℓ_2) where $\ell_1 \neq \ell_2$. For each binary problem, only the examples for which either ℓ_1 or ℓ_2 is relevant (but not both) are retained in the corresponding dataset. The exclusion of the examples for which both labels are relevant is necessary to avoid ill-defined terms in the equations for importance update. The authors motivate this by claiming that the number of the examples for which both labels are relevant, is small in comparison to the number of examples for which precisely one of the two labels is relevant. However, this may not be the case in some data sets, as observed in [18]. The main drawback of this approach is the computational complexity, since the number of feature ranking problems to solve grows quadratically with the number of labels.

A member of the Relief family ReliefF-ML [18] does solve the multi-label ranking problem directly, yet its space complexity is still considerable. The algorithm RReliefF-ML [18] overcomes this issue since it is an extension of the RReliefF version that is suitable for regression tasks [14]. In contrast to the extensions, RReliefF(-ML) computes only one group of the nearest neighbors per example which results in significantly smaller space complexity.

The method was empirically shown to yield relevant feature rankings [18] since it statistically significantly outperformed the baseline. For showing statistical significance, Friedman test was used. However, we need to point out that the very basic assumption of the independence of *data samples* (datasets in this case) was not met, since 10 out of 34 are basically different versions of the same data (Corel16k datasets). In our experiments, we also show that the seemingly ad-hoc choice of the target distance may not lead to the best rankings if we want to optimize for a particular evaluation measure.

Regarding pure predictive modelling setting, the authors in [4] show that in general, different evaluation measures result in different optimal classifiers. However, the authors also show that, e.g., *Hamming Loss* and *Subset Accuracy* have the same optimal classifier under some rather strict conditions.

3 MLC-Relief

Relief family of feature ranking algorithms calculates the feature importance scores by considering differences in the feature values between pairs of examples (an example and its nearest neighbor). More specifically, if the values of features of a pair of examples from the same class are different then the features' importance decreases. Conversely, if the feature values are different for examples from different classes then the features' importance increases.

In the following, we first introduce the distance measures used within the algorithm. Then, the algorithm is described and its computational complexity (including the complexity of computing different distances) is analyzed. Throughout the paper, F and L always denote the number of features and labels respectively.

3.1 Distances: Why and Which

All methods of the Relief family assign feature x_i a weight w_i that is a measure of feature importance in these algorithms. The expected value of the w_i has a nice probability interpretation in the case when both the target and x_i are nominal [14]: simplified to some extent, we have a relation

$$\mathbb{E}[w_i] = \frac{P_{\text{diffAttr, diffTarget}}}{P_{\text{diffTarget}}} - \frac{P_{\text{diffAttr}} - P_{\text{diffAttr, diffTarget}}}{1 - P_{\text{diffTarget}}}, \tag{1}$$

where we define the probabilities $P_{\texttt{ev}} = P(\texttt{ev})$ and $P_{\texttt{ev1, ev2}} = P(\texttt{ev1} \wedge \texttt{ev2})$ that base on the events `diff/sameAttr` (two instances have different/same value of x_i) and `diff/sameTarget` (two instances have different/same target value). The probabilities from the right hand side of Eq. (1) are modeled as the distances in the corresponding spaces: P_{diffAttr} is modeled by the distance d_i on the domain of feature x_i, $P_{\text{diffTarget}}$ is modeled by the distance $d_{\mathscr{L}}$ on the label set $\mathscr{L}$, and $P_{\text{diffAttr, diffTarget}}$ is modeled as their product $d_i d_{\mathscr{L}}$.

First, the distance on the whole descriptive domain $\mathcal{X}$ is defined via the distances d_i on the domains $\mathcal{X}_i$ of features x_i as

$$d_i(\boldsymbol{x}^1, \boldsymbol{x}^2) = \begin{cases} \mathbf{1}[\boldsymbol{x}_i^1 \neq \boldsymbol{x}_i^2] & : \mathcal{X}_i \not\subseteq \mathbb{R} \\ \frac{|\boldsymbol{x}_i^1 - \boldsymbol{x}_i^2|}{\max_{\boldsymbol{x}} \boldsymbol{x}_i - \min_{\boldsymbol{x}} \boldsymbol{x}_i} & : \mathcal{X}_i \subseteq \mathbb{R} \end{cases} \qquad d_{\mathcal{X}}(\boldsymbol{x}^1, \boldsymbol{x}^2) = \frac{1}{F} \sum_{i=1}^{F} d_i(\boldsymbol{x}^1, \boldsymbol{x}^2) \tag{2}$$

where $\mathbf{1}$ is the indicator function with the values $\mathbf{1}[\texttt{true}] = 1$ and $\mathbf{1}[\texttt{false}] = 0$, and max and min go over the examples $\boldsymbol{x}$ in the training set.

For the distance $d_{\mathscr{L}}$ between two sets of labels S^1 and S^2, we consider four options. The use of the first (Hamming Loss) was proposed in [18].

Hamming Loss. This distance is defined as

$$d_{Hamming}(S^1, S^2) = \left|S^1 \setminus S^2 \cup S^2 \setminus S^1\right| \ / \ L. \tag{3}$$

We observe that this is an analogue of $d_{\mathcal{X}}$ from Eq. (2). Encoding a subset $S \subseteq \mathscr{L}$ as a 0/1 vector $\boldsymbol{s}$, where $\boldsymbol{s}_j = 1 \Leftrightarrow \ell_j \in S$, we have $d_{Hamming}(S^1, S^2) = \frac{1}{L}\sum_{j=1}^{L} d_j(\boldsymbol{s}^1, \boldsymbol{s}^2)$, where the numeric part of d_i in Eq. (2) applies in d_j. We believe that there are more suitable choices for the distance $d_{\mathscr{L}}$ that take into account the set structure.

Accuracy. The similarity between two sets can be also measured by their Jaccard index $|S^1 \cap S^2|/|S^1 \cup S^2|$ which is well defined when at least one of the subsets $S^{1,2}$ is not empty (this is the case in our datasets). We then define

$$d_{Accuracy}(S^1, S^2) = 1 - |S^1 \cap S^2| \ / \ |S^1 \cup S^2|. \tag{4}$$

***F*$_1$ distance.** This distance is defined as

$$d_{F1}(S^1, S^2) = 1 - 2|S^1 \cap S^2| \ / \ (|S^1| + |S^2|), \tag{5}$$

where the second term can be seen as the harmonic mean of the precision and recall [15]. However, these two measures are not symmetric, thus inappropriate as the distance measures.

Subset Accuracy. This distance is defined as

$$d_{SubsetAcc}(S^1, S^2) = \mathbf{1}\left[S^1 \neq S^2\right]. \tag{6}$$

It is the strictest, since it does not differentiate between, e.g., almost the same and disjunctive pairs of subsets. This allows for a faster computation of the distance as compared to the other options (Lemma 2).

Except for the d_{F1}, all distances are also metrics. We named them after the measures that they are expected to optimize (defined in Sect. 4.4), and believe that no other standard measures (see [15,27]) allow for a direct derivation of distance definitions.

3.2 Algorithm Description

The calculation of the weights $w_i = \mathit{importance}(x_i)$ using the MLC extension of RReliefF is outlined in Algorithm 1. RReliefF is an iterative procedure. For each of the m iterations, we randomly select an example $\boldsymbol{r}$ from $\mathscr{D}_{\text{TRAIN}}$ (line 4) and find its K nearest neighbors (line 5) using the distance $d_{\mathcal{X}}$ from Eq. (2). After that, we use the neighbors to update the estimates of probabilities that appear in the definition of the weights (1) for all attributes (lines 8–10). The estimates of probabilities are updated with the weighted average of the distances between

Algorithm 1 MLC-RReliefF($\mathscr{D}_{\text{TRAIN}}$, m, K, $d_{\mathscr{L}}$)

```
 1: $\boldsymbol{P}_{\text{diffAttr, diffTarget}}, \boldsymbol{P}_{\text{diffAttr}}$ = zero lists of length $F$
 2: $P_{\text{diffTarget}} = 0.0$
 3: for $\iota = 1, 2, \ldots, m$ do
 4:     $\boldsymbol{r}$ = random example from $\mathscr{D}$
 5:     $\boldsymbol{n}_1, \boldsymbol{n}_2, \ldots, \boldsymbol{n}_K = K$ nearest neighbors of $\boldsymbol{r}$
 6:     for $k = 1, 2, \ldots, K$ do
 7:         $P_{\text{diffTarget}} \mathrel{+}= \delta(\ell) d_{\mathscr{L}}\big(\boldsymbol{r}, \boldsymbol{n}_k\big)$
 8:         for $i = 1, 2, \ldots, F$ do
 9:             $\boldsymbol{P}_{\text{diffAttr}}[i] \mathrel{+}= \delta(\ell) d_i\big(\boldsymbol{r}, \boldsymbol{n}_k\big)$
10:             $\boldsymbol{P}_{\text{diffAttr, diffTarget}}[i] \mathrel{+}= \delta(\ell) d_i\big(\boldsymbol{r}, \boldsymbol{n}_k\big) d_{\mathscr{L}}\big(\boldsymbol{r}, \boldsymbol{n}_k\big)$
11: for $i = 1, 2, \ldots, F$ do
12:     $w_i = \frac{\boldsymbol{P}_{\text{diffAttr, diffTarget}}[i]}{P_{\text{diffTarget}}} - \frac{\boldsymbol{P}_{\text{diffAttr}}[i] - \boldsymbol{P}_{\text{diffAttr, diffTarget}}[i]}{1 - P_{\text{diffTarget}}}$
```

$\boldsymbol{r}$ and its neighbors. Here, the distance $d_{\mathscr{L}}$ from the algorithm input is used. The weight $\delta(k) = 1/(mK)$ ensures that $w_i \in [-1, 1]$ when the algorithms finishes. At the end, the weights w_i are computed (line 12) by using the relation (1).

The default values of the parameters are set as follows. Typically, we iterate over the whole dataset, i.e., $m = |\mathscr{D}_{\text{TRAIN}}|$. By doing this, the estimates of probabilities are expected to be more accurate. The value of K is typically set small enough to capture the local structure in the data. In that way, we implicitly capture the interactions between features [14].

3.3 Computational Complexity

We first analyze the time complexity of a single iteration. Since the space-partitioning data structures, such as kD trees do not perform well when the number of features F is high, we use a brute-force method for finding the nearest neighbors. Hence, the computation of the distances between $\boldsymbol{r}$ and the neighbour candidates takes $\mathcal{O}(MF)$ steps, where $M = |\mathscr{D}_{\text{TRAIN}}|$. In addition to this, the current group of the nearest neighbors must be updated from time to time.

Lemma 1. *The expected number of updates of the group of current nearest neighbors of the instance* $\boldsymbol{r}$ *is approximately* $K \log M$.

Proof. When we iterate over the neighbors, the group of currently K nearest neighbors is updated if, and only if, at most $K - 1$ better candidates have been found so far. Let $\boldsymbol{n}_k$ be the instances from $\mathscr{D}_{\text{TRAIN}} \setminus \{\boldsymbol{r}\}$, sorted increasingly by the distance to $\boldsymbol{r}$, i.e., $\boldsymbol{n}_1$ is the nearest neighbor and $\boldsymbol{n}_{M-1}$ is the farthest neighbor. Let E_k be the expected number of updates when we find the candidate $\boldsymbol{n}_k$. Then, E_k equals the probability p_k of discovering at most $K - 1$ of the instances $\boldsymbol{n}_1, \ldots, \boldsymbol{n}_{k-1}$ before $\boldsymbol{n}_k$. Probability $p_{k,s}$ of discovering precisely s of them equals the probability that $\boldsymbol{n}_k$ appears in the $(s + 1)$-th position in the random permutation of the instances $\boldsymbol{n}_1, \ldots, \boldsymbol{n}_k$, hence $p_{k,s} = 1/k$, for all $s < k$, and $p_{k,s} = 0$ otherwise. It follows that $p_k = \sum_{s=0}^{K-1} p_{k,s} = \min\{k, K\}/k$.

The total number of expected updates E then equals $E = \sum_{k=1}^{M-1} E_k$, hence $E = \sum_{k=1}^{M-1} p_k = K + K\sum_{k=K+1}^{M-1} \frac{1}{k} = K(1 + H_{M-1} - H_K)$, where k-th harmonic number H_k is defined as $H_k = \sum_{s=1}^{k} 1/s$. Since $\log k < H_k < 1 + \log k$, the leading term in E is indeed $K \log M$.

The overall cost of updating the current nearest neighbours is thus $\mathcal{O}(K \log M \log K)$ if we are using, e.g., the heap structure.

When the neighbours $\boldsymbol{n}_k$, $1 \leq k \leq K$, are computed, the distance between their label set and the label set of $\boldsymbol{r}$ are computed. Considering that we store the label sets as 0/1-lists of length L, this takes $\mathcal{O}(KL)$ steps for the distances $d_{Hamming}$, $d_{Accuracy}$ and d_{F1}, since we have to iterate over all labels. In the case of $d_{SubsetAcc}$, we can do much better, knowing that the labels are typically sparse. To be able to obtain a closed form expression, we will assume that all labels have the same probability to be relevant and that they are independent.

Lemma 2. *The expected value of the labels considered in one computation of* $d_{SubsetAcc}$ *is* $\frac{1-p^L(2-p)^L}{(1-p)^2}$*, where* p *is the probability of a label being relevant.*

Proof. We know that $d_{SubsetAcc}(S^1, S^2) = 1$ as soon as we encounter the label $\ell_l \notin S^1 \cap S^2$. Let X be the number of labels considered. The key observation is that we can easily compute $P(X \geq k) = P(\ell_1, \ldots, \ell_{k-1} \in S^1 \cap S^2) = (1 - p)^{2(k-1)}$. This is useful since $\mathbb{E}[X] = \sum_{x=1}^{L} P(X \geq k)$. We obtained geometric series whose sum equals $\mathbb{E}[X] = \frac{1-p^L(2-p)^L}{(1-p)^2}$.

Table 1 reveals that the dataset `Delicious` has $L = 983$ labels and label cardinality (average number of labels per example) $\ell_c \doteq 19$. Thus, $p \doteq 0.019$ and $\mathbb{E}[X] \doteq 1.04$, which is considerably smaller than L.

After the distances $d_{\mathscr{L}}$ are computed, the probability estimates are updated in $\mathcal{O}(KF)$ steps. After all iterations, the weights are computed in $\mathcal{O}(F)$ steps, thus the final time complexity is $\mathcal{O}(m[MF + K \log M \log K + KL + KF] + F) = \mathcal{O}(m[MF + KL])$ (in the case of $d_{SubsetAcc}$, L the term KL is replaced by $\mathbb{E}[X]$). If the number of labels is high, then the term KL may not be negligible, which was overlooked in [18].

4 Experimental Design

Here, we give the detailed experimental design for evaluating the performance of the proposed distances. We begin by stating the experimental questions and summarizing the MLC datasets used in this study. Then, we present the evaluation procedure and give the specific parameters instantiations of the methods.

4.1 Experimental Questions

The main experimental question is: *Does the choice of the distance* $d_{\mathscr{L}}$ *matter?*

Furthermore, we investigate (i) whether the knowledge encapsulated in the feature importances leads to better predictive performance of a model, i.e, are the obtained feature rankings relevant, and (ii) how the quality of ranking is influenced by the number of neighbors K and the number of iterations m.

4.2 Datasets

We use 24 MLC benchmark problems. Table 1 presents the basic statistics of the datasets. The number of features ranges from 72 to 52350. The features are numeric and nominal. The label set size L ranges from 6 to 983, while the number of training examples ranges from 322 up to 70000. The average number of labels per example (in $\mathscr{D}_{\text{TRAIN}} \cup \mathscr{D}_{\text{TEST}}$), i.e., *label cardinality* is also given. With the exception of *Delicious* dataset, it ranges between 1.0 and 4.38.

The datasets come from different domains. *Arts*, *Business*, *Computers*, *Education*, *Entertainment*, *Health*, *Recreation*, *Reference*, *Science*, *Social* and *Society* describe the problems of finding relevant subtopics of the given main topic of a web page. *Bibtex* and *Bookmarks* are automatic tag suggestion problems, *Birds* deals with predictions of multiple bird species in a noisy environment. *Corel5k* contains Corel images. *Delicious* contains contextual data about web pages along with their tags. *Emotions* deals with emotions in music. *Enron* contains data about emails. *Genbase* and *Yeast* come from biological domain. *Mediamill* was introduced in a video annotation challenge. *Medical* comes from Medical Natural Language Processing Challenge. *Scene* deals with labelling of natural scenes. *TMC2007-500* is about discovering anomalies in text reports.

4.3 Evaluation Methodology

We adopted the evaluation methodology that has been previously used in MLC context [18] and in the other types of structured output prediction [17].

We use the same train-test split of the datasets as in the Mulan repository http://mulan.sourceforge.net/datasets-mlc.html. A ranking is computed from the training part $\mathscr{D}_{\text{TRAIN}}$ only, and evaluated on the testing part $\mathscr{D}_{\text{TEST}}$.

The quality of the ranking is assessed by using the kNN algorithm where instead of the standard Euclidean distance, its weighted version was used. For two input vectors $\boldsymbol{x}^1$ and $\boldsymbol{x}^2$, the distance between them is defined as

$$d(\boldsymbol{x}^1, \boldsymbol{x}^2) = \sqrt{\sum_{i=1}^{F} w_i d_i^2(\boldsymbol{x}_i^1, \boldsymbol{x}_i^2)}, \tag{7}$$

where d_i is defined by Eq. (2). The weights are set to $w_i = \max\{\mathit{importance}(x_i), 0\}$, since they need to be made non-negative to ensure that d is well defined, and also to ignore the attributes that have smaller values for importance than a randomly generated attribute would have.

The evaluation through a kNN predictive model was chosen because of two main reasons. First, this is a distance based model, hence, it can easily make use of the information contained in the feature importances in the learning phase. The second reason is kNN's simplicity: its only parameter is the number of neighbors, which we set to 15. In the prediction stage, the neighbors' contributions to the predicted value are equally weighted, so we do not introduce additional parameters that would influence the performance.

Table 1. Data characteristics: sizes of train and test part of the dataset, number of features F, labelset size L and label cardinality ℓ_c.

Dataset	$\lvert\mathscr{D}_{\text{TRAIN}}\rvert$	$\lvert\mathscr{D}_{\text{TEST}}\rvert$	F	L	ℓ_c
Arts [26]	3712	3772	23146	26	1.65
Bibtex [10]	4880	2515	1836	159	2.40
Birds [3]	322	323	260	19	1.01
Bookmarks [10]	70000	17856	2150	208	2.04
Business [26]	5710	5504	21924	30	1.60
Computers [26]	6270	6174	34096	33	1.51
Corel5k [7]	4500	500	499	374	3.52
Delicious[25]	12920	3185	500	983	19.02
Education [26]	6030	6000	27534	33	1.46
Emotions[23]	391	202	72	6	1.87
Enron [1]	1123	579	1001	53	3.38
Entertainment [26]	6356	6374	32001	21	1.41
Genbase [6]	463	199	1185	27	1.25
Health [26]	4557	4648	30605	32	1.64
Mediamill [19]	30993	12914	120	101	4.38
Medical [16]	645	333	1449	45	1.25
Recreation [26]	6471	6357	30324	22	1.43
Reference [26]	4027	4000	39679	33	1.17
Scene [2]	1211	1196	294	6	1.07
Science [26]	3214	3214	37187	40	1.45
Social [26]	6037	6074	52350	39	1.28
Society [26]	7273	7239	31802	27	1.67
TMC2007-500 [21]	21519	7077	500	22	2.22
Yeast [8]	1500	917	103	14	4.24

The second rationale for using kNN as an evaluation model is as follows. If a feature ranking is meaningful, then when the feature importances are used as weights in the calculation of the distances kNN should produce better predictions as compared to kNN without using these weights [28].

4.4 Evaluation Measures

In the following, we denote the sets of true and predicted labels for an example $\boldsymbol{x}$ respectively by $y(\boldsymbol{x})$ and $\hat{y}(\boldsymbol{x})$. The measures *Hamming Loss*, *Accuracy*, *F1 Score* and *Subset Accuracy* can be defined in terms of the distances (3)–(6). They are respectively the means (over $\mathscr{D}_{\text{TEST}}$) of the values $d_{Hamming}(y(\boldsymbol{x}), \hat{y}(\boldsymbol{x}))$, $1 - d_{Accuracy}(y(\boldsymbol{x}), \hat{y}(\boldsymbol{x}))$, $1 - d_{F1}(y(\boldsymbol{x}), \hat{y}(\boldsymbol{x}))$ and $1 - d_{SubsetAcc}(y(\boldsymbol{x}), \hat{y}(\boldsymbol{x}))$.

Thus, *Hamming Loss* should be minimized while the remaining three should be maximized. We use another four well known measures: *One Error*, *Precision*, *Recall* and area under the pooled precision-recall curve (*pooledAUPRC*). The definitions can be found in [15,27].

4.5 Statistical Analysis of the Results

For comparing the algorithms, we use the Friedman test. The null hypothesis H_0 is that all considered algorithms have the same performance. If H_0 is rejected by the Friedman's test, we additionally apply Nemenyi or Bonferroni-Dunn post-hoc test. The first is used when we investigate where the statistically significant differences between *any* two algorithms occur, while the second is used when we are interested in the differences between one particular algorithm and the others. A detailed description of all tests is available in [5].

The results of the Nemenyi and Bonferroni-Dunn tests are presented on critical distance diagrams. Each diagram shows the average rank of the algorithm over the considered datasets, and the critical distance, i.e., the distance for which average ranks of two considered algorithms must differ to be considered statistically significantly different. Additionally, the groups of algorithms among which no statistically significant differences occur are connected with a line.

Before proceeding with the statistical analysis, we round the performances to three decimal points. In the analysis, the significance level was set to $\alpha = 0.05$.

4.6 Parameter Instantiation

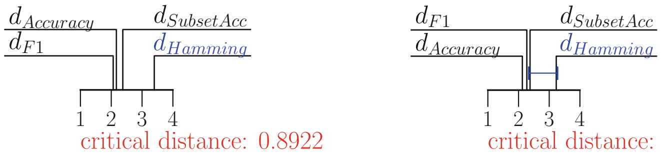
Since the sizes of datasets range over different orders of magnitude, the number of iterations m is given as the proportion of the size of $\mathscr{D}_{\text{TRAIN}}$. The considered values are $m \in \{1\%, 5\%, 10\%, 25\%, 50\%, 100\%\}$. On the other hand, since the number of neighbors K controls the level of locality, it is better given in absolute values. Our choice is to consider the following values $K \in \{1, 5, 10, 15, 20, 25, 30, 40\}$.

5 Results

5.1 Does the Distance Matter?

To give every distance as good chance as possible, we compute and evaluate feature rankings for all combinations of the parameters m and K and for every dataset and distance version, the best pair (with respect to the evaluation measure at hand) is chosen.

Friedman test rejected the null hypothesis for three of the four evaluation measures that the distance definitions are part of: *Accuracy* ($p = 5.2 \cdot 10^{-4}$), *F₁ Score* ($p = 3.5 \cdot 10^{-4}$) and *Subset Accuracy* ($p = 0.011$). In the case of *Hamming Loss*, the performances are not statistically significantly different ($p = 0.28$). The Bonferroni-Dunn test reveals that $d_{Hamming}$ performs statistically significantly worse than the other three distances, for the evaluation measures *Accuracy* (Fig. 1a) and *F_1 Score* (with qualitatively the same diagram). In the case of

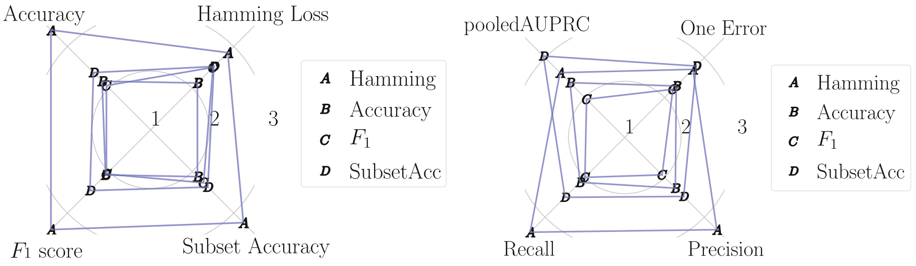


(a) Evaluation measure: *Accuracy* (b) Evaluation measure: *Subset Accuracy*

Fig. 1. Comparison of the four distance functions in terms of (a) *Accuracy*, and (b) *Subset Accuracy*: Critical distance diagrams from Bonferroni-Dunn test with the baseline $d_{Hamming}$.

Subset Accuracy, it has still the worst performance, but it is not statistically significantly worse than $d_{SubsetAcc}$ (Fig. 1b). Interestingly enough, the hypotheses was not rejected for the *Hamming Loss* evaluation measure. Also in this case, the rankings with $d_{Hamming}$ have the worst average rank of 2.9 (as compared to the best average rank of 2.1 that belongs to $d_{Accuracy}$), which leads us to a conclusion that the rankings with $d_{Hamming}$ are indeed to some extent optimized for *Hamming Loss*, but not sufficiently. Average ranks for this four measures are shown on the radar plot in Fig. 2a.

The average ranks of the feature rankings with respect to the other four measures are shown in Fig. 2b. Here, the null hypothesis H_0 is rejected in the case of *Precision* ($p = 0.0011$) and *Recall* ($p = 3.8 \cdot 10^{-4}$). This is not that surprising, since optimizing for *F_1 Score* should directly result in optimized *Precision* and/or *Recall*, as noted after the definition of d_{F1} (Eq. (5)). The results of the follow-up Bonferroni-Dunn tests are similar to those for *Subset Accuracy*: rankings obtained with $d_{Hamming}$ have the worst rank, but are not statistically significantly worse than those obtained with $d_{SubsetAcc}$. Additionally, H_0 is also rejected in the case of *pooledAUPRC* ($p = 0.027$), but in this case, no ranking is statistically significantly different from the one that corresponds to $d_{Hamming}$.



(a) Distance related measures (b) Other measures

Fig. 2. Average ranks of the rankings computed with the four distance functions (denoted by A, B, C and D), in terms of measures that (a) are, and (b) are not directly related to any of distances.

Since we have rejected the null hypotheses (all algorithms perform equally well) in 6 of 8 cases, we can already claim that choosing an appropriate distance measure does matter. Moreover, both diagrams in Fig. 2 show that our newly proposed distance definitions result in rankings that outperform those computed with $d_{Hamming}$. A reason for this may be that the latter cannot really capture the possible interactions between the labels since it can be decomposed to the per-label distances, as noted in Sect. 3.1. This may also be the reason why the rankings computed with the newly proposed distances are typically closer to each other than to the rankings computed with $d_{Hamming}$.

To detect the differences among the rankings, we also apply Nemenyi post-hoc test. In addition to the relations discovered with Bonferroni-Dunn test, we now know that there is statistically significant difference between d_{F1} and $d_{SubsetAcc}$, when the quality is measured in terms of $pooledAUPRC$.

5.2 Are the Obtained Rankings Relevant?

To answer this question, we partially repeat the analysis from the previous section: in addition to the evaluation of the four ranking types, also the non-weighted 15NN algorithm is evaluated. If we reject the null hypothesis H_0 with Friedman test, the four rankings are compared to the non-weighted 15NN classifier with Bonferroni-Dunn post-hoc test. If there is a statistically significant difference between the weighted 15NN classifier and non-weighted 15NN classifier (in favour of the weighted one), we proclaim the ranking relevant.

H_0 is rejected for all evaluation measures. The corresponding Bonferroni-Dunn tests identifies the following. The distances $d_{Accuracy}$ and d_{F1} always result in relevant rankings. The distance $d_{SubsetAcc}$ fails to result in relevant rankings in the case of *One Error*. The distance $d_{Hamming}$ results in relevant rankings when the quality is measured in terms of *Subset Accuracy* and $pooledAUPRC$.

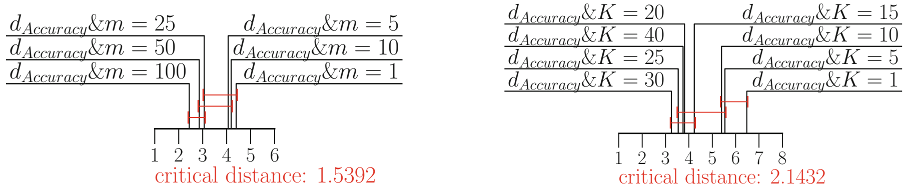
5.3 Influence of the Parameters m and K

To assess how does the number of iterations m influence the quality of ranking, we choose one of the distance functions and a value for the number of neighbors K. When m varies over the values specified in Sect. 4.6, six different rankings are obtained. We compare their quality in terms of the chosen evaluation measure, by applying the Friedman test.

H_0 is rejected for all values of m and for all versions of target distance in the case of *Accuracy*, *F_1 Score*, *Precision*, *Recall* and *Subset Accuracy*. In the case of *Hamming Loss*, it is never rejected. In the case of *One Error*, it is rejected for d_{F1} when $K \geq 25$ and for $d_{SubsetAcc}$ when $K = 40$. In the case of $pooledAUPRC$, the hypothesis is only rejected for $d_{SubsetAcc}$ when $m = 40$.

The only values of m which are always in the top performing group of algorithms, are 25%, 50% and 100%. A typical critical distance diagram (for $d_{Accuracy}$ and $K = 20$) is shown in Fig. 3a.

To assess the influence of the number of Relief neighbours K, a similar analysis is performed, now with the interchanged roles of m and K: the former is



(a) Influence of the number of Relief iterations, for $d_{Accuracy}$ when $K = 20$.

(b) Influence of the number of Relief neighbors, for $d_{Accuracy}$ when $m = 25\%$.

Fig. 3. Critical distance diagrams from Nemenyi tests that show the influence of the number of (a) iterations, and (b) neighbors, on the quality of the $d_{Accuracy}$ rankings, measured in terms of *Precision*.

fixed and the latter varies. The summary of the results is as follows. Number of neighbors seems to have a lesser influence on the quality, since we do not reject all hypotheses for any of the evaluation measures. However, this is mostly due to the fact that K almost never statistically significantly influence the quality of the $d_{Hamming}$ rankings. For the other distances, the hypothesis is always rejected when the quality is measured in terms of *Accuracy*, *F1 Score*, *Precision*, *Recall*. This also holds for *Subset Accuracy* with two exceptions for $d_{SubsetAcc}$: $m \in \{1\%, 100\%\}$. Again, no hypothesis is rejected in the case of *Hamming Loss*.

Typically, more is better regarding the number of neighbors and the highest values of K, i.e., $K \in \{30, 40\}$ have often the best average rank. This can be explained by the sparsity of the labels. To properly asses the average label space distance in $\mathscr{D}_{\text{TRAIN}}$, one has to consider larger neighborhoods. However, the differences among the algorithms for which $K \geq 15$ are not statistically significant. A typical situation (for $d_{Accuracy}$ and $m = 25\%$) is shown in Fig. 3b.

6 Conclusions and Future Work

In this paper, we propose the use of three distance measures on the target space within an extension of RReliefF approach to feature ranking for MLC tasks. These are the distances that are used within the evaluation measures *Accuracy*, *F1 Score* and *Subset Accuracy* for predictive performance on MLC tasks. We have shown that using any of these distances always results in rankings of higher quality than the rankings computed with the distance used in the evaluation measure *Hamming Loss* [18]. Additionally, the newly proposed measures outperform the old one in terms of *Precision* and *Recall*, since these two are directly connected to the *F1 Score*. For more independent measures, such as *pooledAUPRC* and *One Error* we did not observe any differences, so we can conclude that the use of the proposed distance within RReliefF optimizes the corresponding MLC evaluation measures.

We have also shown that all proposed rankings are relevant by comparing the nearest neighbor classifier that uses feature relevance information, to the standard nearest neighbor classifier. Additionally, we measure the influence of the parameters m (number of Relief iterations) and K (number of Relief neighbors) and show that rankings computed from $m = 25\%$ of the training dataset cannot be statistically significantly outperformed on average. The same goes for rankings that were computed by examining the neighborhoods of size $K = 15$.

There are several directions for future work. We plan to find appropriate distance measures for the hierarchical version of the MLC task: hierarchical multi-label classification. Incorporating probabilities in the distances, the Relief family can be also extended in the direction of data with missing labels and semi-supervised problems. Once these are solved, we also plan to develop an extension of Relief for seemingly much harder context of unsupervised learning, where there are no target variables and the analogous approach cannot be taken.

References

1. UC Berkeley Enron Email Analysis Project. http://bailando.sims.berkeley.edu/enron_email.html (2018). Accessed 28 June 2018
2. Boutell, M.R., Luo, J., Shen, X., Brown, C.M.: Learning multi-label scene classification. Pattern Recognit. **37**(9), 1757–1771 (2004)
3. Briggs, F., et al.: The 9th annual mlsp competition: new methods for acoustic classification of multiple simultaneous bird species in a noisy environment. In: IEEE International Workshop on Machine Learning for Signal Processing, MLSP 2013, pp. 1–8 (2013)
4. Dembczyński, K., Waegeman, W., Cheng, W., Hüllermeier, E.: On label dependence and loss minimization in multi-label classification. Mach. Learn. **88**(1), 5–45 (2012)
5. Demšar, J.: Statistical comparisons of classifiers over multiple data sets. J. Mach. Learn. Res. **7**, 1–30 (2006)
6. Diplaris, S., Tsoumakas, G., Mitkas, P.A., Vlahavas, I.: Protein classification with multiple algorithms. In: Bozanis, P., Houstis, E.N. (eds.) PCI 2005. LNCS, vol. 3746, pp. 448–456. Springer, Heidelberg (2005). https://doi.org/10.1007/11573036_42
7. Duygulu, P., Barnard, K., de Freitas, J.F.G., Forsyth, D.A.: Object recognition as machine translation: learning a lexicon for a fixed image vocabulary. In: Heyden, A., Sparr, G., Nielsen, M., Johansen, P. (eds.) ECCV 2002. LNCS, vol. 2353, pp. 97–112. Springer, Heidelberg (2002). https://doi.org/10.1007/3-540-47979-1_7
8. Elisseeff, A., Weston, J.: A kernel method for multi-labelled classification. In: Dietterich, T.G., Becker, S., Ghahramani, Z. (eds.) Advances in Neural Information Processing Systems 14. Springer International Publishing (2001)
9. Guyon, I., Elisseeff, A.: An introduction to variable and feature selection. J. Mach. Learn. Res. **3**, 1157–1182 (2003)
10. Katakis, I., Tsoumakas, G., Vlahavas, I.: Multilabel text classification for automated tag suggestion. In: Proceedings of the ECML/PKDD 2008 Discovery Challenge (2008)
11. Kira, K., Rendell, L.A.: The feature selection problem: traditional methods and a new algorithm. In: Proceedings of the Tenth National Conference on Artificial Intelligence, pp. 129–134. AAAI'92, AAAI Press (1992)
12. Kocev, D., Vens, C., Struyf, J., Džeroski, S.: Tree ensembles for predicting structured outputs. Pattern Recognit. **46**(3), 817–833 (2013)

13. Kong, D., Ding, C., Huang, H., Zhao, H.: Multi-label ReliefF and F-statistic feature selections for image annotation. In: 2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 2352–2359 (2012)
14. Kononenko, I., Robnik-Šikonja, M.: Theoretical and empirical analysis of ReliefF and RReliefF. Mach. Learn. J. **55**, 23–69 (2003)
15. Madjarov, G., Kocev, D., Gjorgjevikj, D., Džeroski, S.: An extensive experimental comparison of methods for multi-label learning. Pattern Recognit. **45**, 3084–3104 (2012)
16. Pestian, J.P., et al.: A shared task involving multi-label classification of clinical free text. In: Proceedings of the Workshop on BioNLP 2007: Biological, Translational, and Clinical Language Processing (BioNLP '07), pp. 97–104 (2007)
17. Petković, M., Džeroski, S., Kocev, D.: Feature ranking for multi-target regression with tree ensemble methods. In: Yamamoto, A., Kida, T., Uno, T., Kuboyama, T. (eds.) DS 2017. LNCS (LNAI), vol. 10558, pp. 171–185. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-67786-6_13
18. Reyes, O., Morell, C., Ventura, S.: Scalable extensions of the ReliefF algorithm for weighting and selecting features on the multi-label learning context. Neurocomputing **161**, 168–182 (2015)
19. Snoek, C.G.M., Worring, M., van Gemert, J.C., Geusebroek, J.M., Smeulders, A.W.M.: The challenge problem for automated detection of 101 semantic concepts in multimedia. In: Proceedings of the 14th ACM International Conference on Multimedia, pp. 421–430. ACM, New York (2006)
20. Spolaôr, N., Cherman, E.A., Monard, M.C., Lee, H.D.: A comparison of multi-label feature selection methods using the problem transformation approach. Electron. Notes Theor. Comput. Sci. **292**, 135–151 (2013)
21. Srivastava, A.N., Zane-Ulman, B.: Discovering recurring anomalies in text reports regarding complex space systems. In: 2005 IEEE Aerospace Conference (2005)
22. Stańczyk, U., Jain, L.C. (eds.): Feature selection for data and pattern recognition. Studies in Computational Intelligence. Springer, Berlin (2015)
23. Trochidis, K., Tsoumakas, G., Kalliris, G., Vlahavas, I.: Multilabel classification of music into emotions. In: 2008 International Conference on Music Information Retrieval (ISMIR 2008), pp. 325–330 (2008)
24. Tsoumakas, G., Katakis, I.: Multi-label classification: An overview. Int. J. Data Warehous. Min. pp. 1–13 (2007)
25. Tsoumakas, G., Katakis, I., Vlahavas, I.: Effective and efficient multilabel classification in domains with large number of labels. In: ECML/PKDD 2008 Workshop on Mining Multidimensional Data (MMD'08) (2008)
26. Ueda, N., Saito, K.: Parametric mixture models for multi-labeled text. In: Advances in Neural Information Processing Systems 15, pp. 721–728. MIT Press (2003)
27. Vens, C., Struyf, J., Schietgat, L., Džeroski, S., Blockeel, H.: Decision trees for hierarchical multi-label classification. Mach. Learn. **73**(2), 185–214 (2008)
28. Wettschereck, D.: A study of distance based algorithms. Ph.D. thesis, Oregon State University, USA (1994)