Feature Ranking for Hierarchical Multi-Label Classification with Tree Ensemble Methods

Matej Petković, Sašo Džeroski, Dragi Kocev

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

Abstract: In this work, we address the task of feature ranking for hierarchical multi-label classification (HMLC). The task of HMLC concerns problems with multiple binary variables, organized into a hierarchy of target attributes. The goal is to train a model to learn and accurately predict all of them, simultaneously. This task is receiving increasing attention from the research community, due to its wide application potential in text document classification and functional genomics. Here, we propose a group of feature ranking methods based on three established ensemble methods of predictive clustering trees: Bagging, Random Forests and Extra Trees. Predictive clustering trees are a generalization of decision trees, towards predicting structured outputs. Furthermore, we propose to use three scoring functions for calculating the feature importance values: Symbolic, Genie3 and Random Forest. We test the proposed methods on 30 benchmark HMLC datasets, show that Symbolic and Genie3 scores return relevant rankings, that all three scores outperform the HMLC-Relief ranking method and are computed in very time-efficient manner. For each scoring function, we find the most appropriate ensemble method and compare the scores to find the best one.

Keywords: hierarchical multi-label classification; feature ranking; ensemble methods; Relief

1 Introduction

Classification is a task in predictive modelling, where we develop a model that takes a vector $\boldsymbol{x}$ of descriptive variables (features) x_i as the input, and predicts the class value y, for a given example. 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 is sick, where $y \in \{yes, no\}$ (binary), or what is the blood type of a person where $y \in \{A, B, AB, 0\}$ (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 l from a predefined set $\mathcal{L}$ are *relevant* for a given example. For example, one can predict which of the genres from the set $\mathcal{L}$ = {*romance*, *drama*, *comedy*} are relevant for a given film. Clearly, a film can be *drama* and *comedy* at the same time.

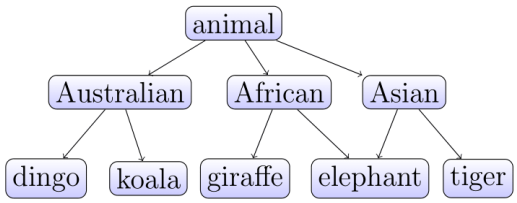

Figure 1
An exemplary hierarchy of animals

Hierarchical MLC (HMLC) is a generalization of MLC, where the labels are organized into a hierarchy that is given as a binary relation <, which partially orders the set $\mathcal{L}$. If $l_1 < l_2$, we say that l_1 is a predecessor of l_2. This relation imposes the *hierarchical constraint*: If l is relevant for a given example then all the predecessor labels are also relevant for the example. Fig. 1 shows a toy hierarchy of groups of animals. For example, *animal* < l for all labels $l \in \mathcal{L}$, *Asian* < *tiger* etc. The label *elephant* has two parents (*African* and *Asian*) and one additional predecessor (*animal*). Thus, if an example is an *elephant*, then it is also an *African*, *Asian* and an *animal*. If every label has, at most, one parent, the hierarchy is tree-shaped. Otherwise (as it is the case in the toy hierarchy), it is a general directed acyclic graph (DAG). In this paper, DAG refers only to the hierarchies that are not also tree-shaped.

HMLC is a practically relevant task with problems occurring in life sciences (e.g., gene function prediction, disease classification), environmental sciences (e.g., habitat modelling, remote sensing), multimedia (e.g., image classification and retrieval) and semantic web (classification and analysis of text and web pages). Furthermore, many of the datasets used in this study (the data with -GO in their names) are from functional genomics - the goal there is to assign to each gene the multiple functions it has. In turn, the gene functions are organized into an ontology that takes the form of a DAG known as Gene Ontology (GO) [7].

It has been shown in several studies that use of a hierarchy improves the performance as compared to MLC in a variety of domains. For example, [29] show that the task of HMLC is beneficial to exploit the interdependencies among the labels. In [18] it is shown that the use of hierarchy helps obtain better single tree models. Moreover, [20] shows that MLC can be approached as HMLC by constructing hierarchies of the labels by clustering the label co-occurrences.

However, it is still possible to approach HMLC problems by ignoring the hierarchy at the learning phase and use any of the MLC methods, such as binary relevance or power set approach [28]. Binary relevance is a simple method that

converts a MLC task to several binary classification tasks with y ∈ {*yes*, *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, the task of predicting a subset of $\mathcal{L}$ is converted to the task of predicting an element of the power set $2^{\mathcal{L}}$, and thus converting a MLC task to a multi-class classification task. However, the number of classes can be as high as $2^{|\mathcal{L}|}$ which results in a very sparse dataset. At prediction stage, predecessors of the labels predicted as relevant, must be added to the set of relevant labels, so that the hierarchical constraint is met. A similar two-step approach [1] learns support vector machines for each class separately, and then combines the predictions using a Bayesian network model so that the hierarchical constraint is met. However, method adaptation techniques where an existing method is adapted to a new problem may be more appropriate. This can be done with predictive clustering trees (PCTs), which were shown to outperform their basic versions that follow the binary relevance approach [29].

In this paper, we do not address the task of building predictive models for HMLC. Rather, we propose a feature ranking method that is useful in this context. Feature ranking is another important task in machine learning, where the goal is to assess 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 the 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 and domain experts collaborate. Predictive models, such as decision trees, are easier to interpret when a small number of the relevant features are used to learn them.

There is a plethora of feature ranking methods for the task of classification [27]. A possible approach to MLC feature ranking is to adapt the binary relevance approach from predictive modelling, where at the first stage, feature importance values are computed for every label $l \in \mathcal{L}$ separately as in the classification case. After that, the feature importance values are averaged over the different labels and a single ranking is returned. However, the landscape of methods for feature ranking for HMLC is not well populated, due to the complexity of the task.

This work contains the following contributions:

a) We propose a group of novel feature ranking approaches for HMLC that base on the Symbolic, Genie3 [13] and Random Forest [4] scoring

functions, coupled with Bagging, Random Forests and Extra Trees ensembles of PCTs for HMLC [16, 25]

b) We develop the parameter-less version of the Symbolic score. The earlier version has been (as well as the other two scores) previously evaluated in the context of multi-target regression [22]

c) We evaluate the proposed approaches on 30 HMLC benchmark datasets by using kNN model that uses feature importance scores in the distance function: we compare them to two baselines and show that the proposed scoring functions outperform i) the non-informed ranking where all features have equal importance; ii) the adaptation of Relief algorithm to HMLC [26]

The rest of the paper is organized as follows. In part 2, we describe predictive clustering trees, ensembles thereof and the proposed feature ranking scores. Then, we proceed to the HMLC-Relief description. In part 3, the experimental design is given. In part 4, the results are presented and finally, we summarize in the Conclusion Section.

2 Methods

We first present the ensemble-based feature rankings and then proceed to the Relief ranking. Both PCT framework and the Relief family of algorithms is implemented in the CLUS system (http://source.ijs.si/ktclus/clus-public).

2.1 Predictive Clustering Trees and Ensembles Thereof

PCTs generalize decision trees and can be used for a variety of learning tasks, including clustering and different types of structured output prediction tasks, e.g., multi-target regression, multi-label classification, hierarchical multi-label classification, time series prediction etc. [3] [16]. PCTs are induced with the standard greedy top-down induction of decision trees algorithm [5]. The heuristic h that is used for selecting the tests guides the algorithm towards small trees with good predictive performance. If there are no candidate tests, a leaf is created and the prototype of the instances belonging to that leaf is computed.

In the HMLC case, the heuristic function is defined as follows. First, a label subset $S \subseteq \mathcal{L}$ is converted into 0/1-vector $\boldsymbol{s}$ of length $|\mathcal{L}|$, where $\boldsymbol{s_j} = 1 \Leftrightarrow l_j \in S$. We denote the variance of $\boldsymbol{s_j}$ over subset of examples $E \subseteq \mathcal{D}_{\text{TRAIN}}$ as $var_j(E)$. Additionally, each label l_j is assigned a weight w_j that is defined as $w_j = \alpha\ \overline{w}(Parents(l_j))$ if the set of parents $Parents(l_j)$ is not empty and $w_j = 1$ otherwise (in the root(s) of the hierarchy). The function $\overline{w}(P)$ returns the average weight of the

label set P, and the parameter $\alpha \in (0, 1)$ is user-defined. Then, the impurity function is defined as $impurity(E) = \sum_{j=1}^{|\mathcal{L}|} w_j var_j(E)$, hence the labels l_j that are closer to the root of the hierarchy have bigger influence on the heuristic function. Motivation for this is - considering the example in Fig. 1 - that one can only correctly predict leaf labels (*dingo* versus *koala*) if the correct predictions are made for their predecessors (*Australian* versus *African*). Heuristic h is defined as the decrease of the impurity after applying the test. In a leaf L, the prototype function returns a vector whose j-th component equals the average value of $\boldsymbol{s_j}$ of the examples belonging to L.

To calculate feature importance scores (i.e., feature rankings), we grow ensembles of PCTs instead of growing a single one. An ensemble is a set of base predictive models, whose prediction for each new example is made by combining the predictions of the models from the ensemble. In HMLC tasks, this is typically achieved by taking the average of the base-model predictions. In our experiments, we used the following three approaches.

Random Forests, Bagging. In the Random Forests ensemble, instead of being derived from the original dataset $\mathcal{D}_{\text{TRAIN}}$, each tree in the ensemble is learned from a different bootstrap replicate B of the dataset $\mathcal{D}_{\text{TRAIN}}$, called bag. Additionally, we choose a random subset S of features in every internal node of the tree, and consider only the tests that are yielded by the features in S when looking for the best test. Typical size of the set S is of the order *log F* or *root(F)*, where F is the number features in $\mathcal{D}_{\text{TRAIN}}$. If $|S| = F$, we obtain the Bagging procedure.

Extra Trees. Each tree is being developed directly from $\mathcal{D}_{\text{TRAIN}}$, but the candidate tests for each node are now extremely randomized. Again, we chose a random subset S of features in every internal node of the tree, and consider only one randomly chosen test per chosen feature, when looking for the best test.

2.2 Ensemble Scores

Once we build an ensemble of PCTs, we can exploit the ensemble structure to compute the feature ranking in three different ways. In the following, we denote a tree as *TR*, whereas $N \in TR$ denotes a node. Trees form a forest *FO*. Its size (the number of trees in the forest) is denoted as $|FO|$. The set of all internal nodes of a tree *TR* in which the feature x_i appears as part of a test is denoted as $\text{TR}(x_i)$.

Symbolic Ranking. In the simplest version of the score, we would count how many times a given feature occurs in the tests in the internal nodes of the trees. Since the features that appear closer to the root are intuitively more important than those that appear deeper in the trees, we weight these occurrences by the number of examples *e(N)* that reach a node N, and define the feature importance as

$$importance_{SYMB}(x_i) = \frac{1}{|FO|} \sum_{TR \in FO} \sum_{N \in TR(x_i)} e(N) \,/\, |\mathcal{D}_{\text{TRAIN}}| \qquad (1)$$

Genie3 Ranking. The main motivation for Genie3 ranking is that splitting the current subset $E \subseteq \mathcal{D}_{\text{TRAIN}}$, according to a test where an important attribute appears, should result in high impurity reduction. The Genie3 importance of the feature x_i is thus defined as:

$$importance_{GENIE3}(x_i) = \frac{1}{|FO|}\sum_{TR\in FO}\sum_{N\in TR(x_i)} h^* \quad (2)$$

where h^* is the heuristic value of the variance reduction function. Since h^* is proportional to $e(N)=|E|$, greater emphasis is again put on the attributes higher in the tree, where $|E|$ is larger.

Random Forest Ranking. (To avoid confusion, we use the plural form (Random Forests) to refer to the ensemble method, and singular form to refer to the feature ranking score (Random Forest)). This feature ranking method tests how much the noise in a given feature decreases the predictive performance of the trees in the forest. The greater the performance degradation, the more important the feature is. In contrast to the first two feature rankings which can be computed for all three ensemble methods, this score cannot be used with ensembles of Extra Trees, since it uses the internal out-of-bag estimates of the error.

Once a tree *TR* is grown, the algorithm evaluates the performance of the tree by using the corresponding OOB_{TR} examples. This results in the predictive error $err(OOB_{TR})$, where lower error value corresponds to better predictions. To assess the importance of the feature x_i for the tree *TR*, we randomly permute its values in the set OOB_{TR} and obtain the set OOB_{TR}^{i}. Then, the error $err(OOB_{TR}^{i})$, is computed and the importance of the feature x_i for the tree *TR* is defined as the relative increase of error after noising. The final Random Forest score of the feature is the average of these values over all trees in the forest, namely:

$$importance_{RF}(x_i) = \frac{1}{|FO|}\sum_{TR\in FO}\frac{err(OOB_{TR}^{i}) - err(OOB_{TR})}{err(OOB_{TR})} \quad (3)$$

2.3 HMLC-Relief Ranking

The 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 neighbors). 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.

The values of the importance $importance_{Relief}(x_i)$ in the Relief can be written in a probabilistic fashion [17]: simplified to some extent, we have a relation:

$$importance_{Relief}(x_i) = \frac{P_{diffAttr,\ diffTarget}(i)}{P_{diffTarget}} - \frac{P_{diffAttr}(i) - P_{diffAttr,\ diffTarget}(i)}{1 - P_{diffTarget}} \quad (4)$$

where we define the probabilities $P_{ev} = P(ev)$ and $P_{ev1,ev2} = P(ev1 \wedge ev2)$ 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 in Eq. (4) 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, $P_{diffTarget}$ is modeled by the distance $d_{\mathcal{L}}$ between two label subsets of $\mathcal{L}$, and $P_{\text{diffAttr,diffTarget}}$ is modeled as their product $d_i \, d_{\mathcal{L}}$. This enables the generalization not only to numeric attributes and targets, but also to more complex target types, such as hierarchies as described in [26]. However, it must be assured that the upper bound of all distances is 1, which was overlooked in [26]. There, they proceed as follows:

First, the distances d_i on the feature domains X_i, and the distance d_X on whole descriptive domain X are defined as:

$$d_i(\boldsymbol{x}^1, \boldsymbol{x}^2) = \begin{cases} \mathbf{1}[x_i^1, x_i^2] & : X_i \nsubseteq \mathbb{R} \\ \frac{|x_i^1 - x_i^2|}{\max_{\boldsymbol{x}} x_i - \min_{\boldsymbol{x}} x_i} & : X_i \subseteq \mathbb{R} \end{cases} \qquad d_X = \frac{1}{F} \sum_{i=1}^{F} d_i(\boldsymbol{x}^1, \boldsymbol{x}^2) \tag{5}$$

where **1** is the indicator function defined as **1**[*true*] = 1 and **1**[*false*] = 0. The distance between two label sets S_1 and S_2 is defined as a weighted Euclidean distance between the corresponding 0/1-vectors $\boldsymbol{s}^1$ and $\boldsymbol{s}^2$ where $\boldsymbol{s}_j^{1,2}$ and w_j are defined as in Sec. 2.1. We correct this and define as:

$$\mu = 1 / \max_{S,\, S'} d_E(S, S') \quad \text{and} \quad d_{\mathcal{L}}(S_1, S_2) = \mu \, d_E(S_1, S_2) \tag{6}$$

The algorithm iteratively estimates the probabilities in Eq. (4) by randomly sampling the training dataset and comparing the chosen example r to its nearest neighbors $\boldsymbol{n}_j$. This is repeated m times and the estimate for $P_{diffTarget}(i)$ is thus,

$$P_{diffAttr}(i) = \frac{1}{mK} \sum_{r} \sum_{k=1}^{K} d_i(\boldsymbol{r}, \boldsymbol{n}_j).$$

The other probabilities are estimated analogously. The weight *1/mK* ensures that the computed importance values are between -1 and 1. The values of the two parameters are set as follows. Typically, we iterate over the whole dataset, i.e., $m = |\mathcal{D}_{\text{TRAIN}}|$. 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 [17]. Previous experiments [17] [21] have shown that $K = 10$ or $K = 15$ give the best performance.

The normalization factor in the numeric part of the definition of d_i in Eq. (5) is trivial to compute. However, this is not the case with the μ in Eq. (6), if we want to do that efficiently. For tree-shaped hierarchies, we developed an efficient algorithm for computing μ that recursively finds the distance-maximizing pair of the labels in $O(|\mathcal{L}|)$ time. If hierarchy is a general DAG we could not do considerably better than computing the maximizing pair by exhaustive search.

3 Experimental Design

3.1 Experimental Questions

The experiments were designed to answer the following experimental questions:

1) Given an ensemble feature ranking score, after which number of trees in the ensemble the quality of the ranking saturates?
2) Do the proposed ensemble feature ranking scores yield relevant rankings, i.e., can the additional information captured in the feature ranking boost the performance of the non-informed baseline classifier?
3) Do the proposed ensemble feature ranking scores outperform the improved version of HMLC-Relief algorithm?
4) Given an ensemble feature ranking score, which ensemble method is the most suitable?
5) Which ensemble feature ranking score yields the best rankings?

3.2 Datasets

We use 30 HMLC benchmark problems whose characteristics are summarized in Tab. 1. Most of the datasets have a few thousand of examples while the number of features could be as high as 74435. The label set typically contains a few hundred elements. Approximately 25% of the hierarchies are DAGs. Many of the datasets are microarray data and come from the field of functional genomics. They describe the connection between description of proteins and their functional classes that are taken from Gene Ontology [7] (the corresponding hierarchies are DAGs), or the MIPS functional hierarchy (http://mips.helmholtz-muenchen.de/funcatDB) (the corresponding hierarchies are tree-shaped). Some other datasets are about text categorization of the processed news (reuters), patent classification according to the World International Patent Organization (wipo) etc.

Table 1
Properties of the datasets: data size $|\mathcal{D}|$ given as the sum $|\mathcal{D}_{\text{TRAIN}}| + |\mathcal{D}_{\text{TEST}}|$ of training and test set sizes, number of features F, shape of the hierarchy, label set size, depth of the hierarchy (max d), and average leaf depth (average d) of the hierarchy

Dataset	$\|\mathcal{D}\|$	F	**shape**	$\|\mathcal{L}\|$	**max d**	**avg d**
cellcycle-yeast-FUN [6]	2482+1284	77	tree	751	4	4
church-yeast-FUN [6]	2480+1284	27	tree	751	4	4
clef07a-is [9]	10000+1006	80	tree	152	3	3
derisi-yeast-FUN [6]	2455+1278	63	tree	751	4	4

diatoms [11]	726+372	200	tree	81	2	2
eisen-yeast-FUN [6]	1588+837	79	tree	751	4	4
enron-corr [14]	988+660	1001	tree	67	3	2.2
expr-yeast-FUN [6]	2494+1294	552	tree	751	4	4
exprindiv-ara-FUN [6]	2314+1182	1251	tree	424	4	2.8
exprindiv-ara-GO [6]	7161+3679	1251	DAG	627	6.5	5.6
gasch1-yeast-FUN [6]	2486+1287	173	tree	751	4	4
hom-ara-FUN [6]	2260+1213	72869	tree	420	4	2.8
hom-ara-GO [6]	7119+4002	72869	DAG	623	6.5	5.6
hom-yeast-FUN [6]	2549+1318	47034	tree	751	4	4
icpr2010 [11]	4913+2999	4000	tree	76	3	2.5
interpro-ara-FUN [6]	2455+1264	2815	tree	427	4	2.8
interpro-ara-GO [6]	7778+3985	2815	DAG	630	6.5	5.6
pheno-yeast-FUN [6]	1010+582	69	tree	751	4	4
reuters [19]	3000+3000	47236	tree	143	4	2.5
scop-ara-FUN [6]	2055+1042	2003	tree	407	4	2.8
scop-ara-GO [6]	6507+3336	2003	DAG	572	6.5	5.7
seq-ara-FUN [6]	2455+1264	4450	tree	424	4	2.8
seq-ara-GO [6]	7778+3985	4450	DAG	630	6.5	5.6
seq-yeast-FUN [6]	2590+1342	478	tree	751	4	4
spo-yeast-FUN [6]	2442+1269	80	tree	751	4	4
struc-ara-FUN [29]	2455+1264	14804	tree	427	4	2.8
struc-ara-GO [29]	7778+3985	14804	DAG	630	6.5	5.6
struc-yeast-FUN [29]	2535+1316	19628	tree	751	4	4
wipo [24]	1352+358	74435	tree	528	4	4
yeast-GO [1]	2310+1155	5930	DAG	133	7.3	4.7

3.3 Evaluation Procedure

In our experiments, we use the same train/test splits of the datasets as the original authors, for all the data sets. First, a feature ranking is computed from the training set $\mathcal{D}_{\text{TRAIN}}$. Its quality is assessed by the k-nearest neighbor (kNN) algorithm in which the weighted version of Euclidean distance is used instead of the standard one, i.e., $d_E(\boldsymbol{x}^1, \boldsymbol{x}^2) = \sqrt{\sum_i w_i d_i^2(\boldsymbol{x}^1, \boldsymbol{x}^2)}$, where $\boldsymbol{x}^{1,2}$ are the input vectors of nominal/numeric feature values and d_i is defined by Eq. (5). Since the rankings that base on the Random Forest score and HMLC-Relief could contain negative relevance scores, which in both cases means that the feature is more irrelevant that a random feature would be, the weights are defined as $w_i = \max\{0, \mathit{importance}(x_i)\}$.

This evaluation procedure was chosen because kNN classifier is a distance-based model that can directly make use of feature importance values, learned in the first

phase of procedure. The second reason for our choice was kNN's simplicity: its only parameter is the number of neighbors, which we set to 10.

The rationale for using kNN as an evaluation model is as follows. If a feature ranking is meaningful, then when the feature importance values are used as weights in the calculation of the distances kNN should produce better predictions as compared to kNN without using these weights [30]. Once the kNN model is trained on $\mathcal{D}_{\text{TRAIN}}$, its performance on $\mathcal{D}_{\text{TEST}}$ is measured in terms of the area under the average precision-recall curve $AU\overline{PRC}$ [29] which is computed as follows. First, we define multi-set $P = \{(v_{i,l}, p_{i,l}) \mid l \in \mathcal{L}, (\boldsymbol{x}_i, S_i) \in \mathcal{D}_{\text{TEST}}\}$ where $v_{i,l} = \mathbf{1}[l \in S_i]$ and $p_{i,l} \in [0, 1]$ is the predicted probability of $l \in S_i$. After that, the numbers of true positives tp_θ, false positives fp_θ and false negatives fn_θ are computed, for all $\theta \in [0, 1]$, e.g., $tp_\theta = |\{(v, p) \in P \mid v = 1, p \geq \theta\}|$. From these, we compute recall $r_\theta = tp_\theta / (tp_\theta + fn_\theta)$ and precision $p_\theta = tp_\theta / (tp_\theta + fp_\theta)$, define the curve $PRC = \{(r_\theta, p_\theta) \mid \theta \in [0, 1]\}$, and compute the area under it.

It might seem that another possible approach to evaluation is extending a dataset with some randomly generated features and then see whether they are ranked at the bottom of the ranking. However, this approach is more suitable for synthetic data where the ground truth is known and all features can be made relevant. In the real world data, it may very well happen that some of the high-dimensional datasets indeed contain completely irrelevant features, hence, using this approach would yield incorrect performance estimates.

3.4 Statistical Analysis of the Results

We use the Wilcoxon's test for comparing two algorithms, and Friedman's test for comparing more than two. In both cases, the null hypothesis is that all considered algorithms have the same performance. If it is rejected by the Friedman's test, we additionally apply Nemenyi's post-hoc test to investigate where the statistically significant differences between the algorithms occur. A detailed description of all tests is available in [8]. When performing Wilcoxon's tests whose outcomes are not independent, we control the false discovery rate by the Benjamini-Hochberg procedure [2]: let p_i be the i-th smallest among the obtained p-values, and t the number of tests. Let i_0 be the largest i, such that $p_i \leq \alpha_i^* := (i / t)\, \alpha$. Then, we can reject the hypotheses belonging to p-values p_i, for $1 \leq i \leq i_0$.

The results of the Nemenyi's tests are presented on average ranks 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. If Friedman's test did not reject the null hypothesis, all algorithms on the average rank diagram are connected with the same line, and no critical distance is given.

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

3.5 Parameter Instantiation

First, we give the parameters used in the process of obtaining the ensemble-based rankings. Afterwards, we give those for HMLC-Relief.

We consider the following ensemble sizes: $|FO| \in \{10, 25, 50, 75, 100, 150, 250\}$. Since 100 trees is a typical recommended value [15], this should be enough. This is the only parameter for Bagging, while Random Forests and Extra Trees method need another one: the number of features F considered in each internal node as described in Sec. 2.1. The recommended value for Random Forests is $F' = root(F)$ [15] and $F' = F$ for Extra Trees [15]. However, if we carefully examine the data, we see that are some datasets, e.g., enron-corr and struc-ara-GO, for which the diversifying mechanism of the Extra Trees algorithm does not work if we set $F' = F$, since every single one of the attributes takes, at most, two values, which results in only one possible split per attribute. Thus, we rather choose $F' = root(F)$ for Extra Trees also, since a necessary condition for an ensemble to be more accurate than any of its individual members, is that the members are diverse models [12].

If we fix the ensemble size and feature ranking score and compare the quality of the rankings from the ensembles of Extra Trees that use *root(F)*- and *F*-feature subsets via Wilcoxon's test, the results show that *root(F)*-version of the ensemble statistically significantly outperforms the *F*-version on the problematic datasets, and that there are no statistically significant differences on the other datasets.

As for Relief algorithm, it has been shown in the previous experiments that the Relief algorithm is quite robust regarding the value of the number of neighbors K and that no other value outperforms $K = 15$ [17] [21], hence we will adhere to this value. Since the datasets are not all of equal size, the number of iterations m in HMLC-Relief algorithm is given as the proportion of the size of $\mathcal{D}_{\text{TRAIN}}$. The considered values are $m \in \{1\%, 5\%, 10\%, 25\%, 50\%, 100\%\}$.

4 Results

Feature rankings, and the extended results (for every dataset separately) are available at http://source.ijs.si/mpetkovic/hmlc-ranking.

4.1 Saturation of the Ranking Quality

Influence of the Ensemble Size. We analyze each ranking score and ensemble separately. While these two are fixed, we let the number of the trees in the ensemble vary, and apply Friedman's test to discover whether some differences among them occur.

The resulting *p*-values are all bigger than 0.05, which means that the rankings can be computed very efficiently since it suffices to grow only 10 trees. Therefore, the ensemble size in the subsequent experiments is fixed to 10. Fig. 2a shows the resulting average rank diagram for Genie3 score, computed from a Bagging ensemble. Similar conclusions can be made using the other ensembles and scores.

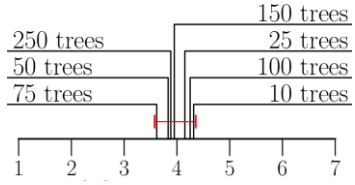

Figure 2a

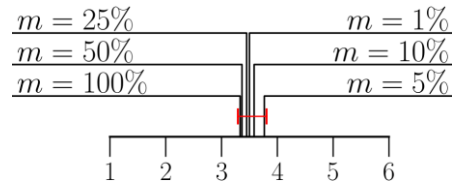

Figure 2b

Saturation of the rankings: Friedman's test discovered no statistically significant differences among different (a) ensemble sizes in the ensemble rankings (Genie3 score, coupled with the Bagging ensemble is shown), (b) considered proportions of the dataset by HMLC-Relief

Influence of the HMLC-Relief's Number of Iterations. Similarly, to previous setting, we let the value of the parameter *m* vary and compare the quality of the corresponding HMLC-Relief rankings by applying Friedman's test. Again, there are no statistically significant differences among the algorithms ($p = 0.96$) and the differences among quality of the rankings are now even smaller. As shown in Fig. 2b, no two average ranks differ by more than 0.43. Since the most time-efficient setting is $m = 1\%$, this is the considered number of iterations in the subsequent HMLC-Relief experiments.

4.2 Are the Ensemble-based Rankings Relevant?

To answer this question, we compare the predictive performance of the kNN classifier which uses the importance values from a particular feature ranking, to a non-weighted kNN baseline by using Wilcoxon's test. This pair-wise comparison is made for every admissible pair of feature ranking score and ensemble method, which results in 8 (not independent) comparisons. After we compute the *p*-values, the Benjamini-Hochberg correction is applied.

It turns out that all weighted kNN classifiers perform better than the baseline. However, the differences are statistically significant in 5 out of 8 cases, as assessed after applying the Benjamini-Hochberg correction after the Wilcoxon's test results: (Symbolic, RandomForests), (Genie3, RandomForests), (Symbolic, Bagging), (Genie3, ExtraTrees), and (Genie3, Bagging). Here, the *p*-values range from $2.14 \cdot 10^{-4}$ to $2.63 \cdot 10^{-2}$. The remaining three cases (with *p*-values at least $4.17 \cdot 10^{-2}$) are both feature rankings that are computed using Random Forest score and the feature ranking computed from Symbolic score and Extra Trees ensemble. Thus, using Genie3 score always results in relevant rankings, while Symbolic score fails to yield relevant rankings when used in combination with Extra Trees. The fact that Random Forest ranking fails to yield relevant rankings in all cases, may be at least partially explained by the sparsity of the data, since in that case, the differences of the error estimates on the out of bag examples and out of bag examples with permuted values of a feature, may not be that significant.

4.3 The most Appropriate Ensemble for a Given Score

Here, we fix the remaining parameter of the ensemble-based rankings, i.e., we find the most appropriate ensemble method for each feature ranking score. This is done by first fixing a feature ranking score, and then comparing the quality of the rankings obtained using this score and one of the possible ensemble methods. In the case of Symbolic and Genie3 score, Friedman's test is applied, since they can be used in combination with three ensemble methods. In the case of Random Forest score which cannot be paired with Extra Trees, Wilcoxon's test is used.

In the case of Symbolic ranking, the differences are not statistically significant ($p = 0.106$), as shown in Fig. 3a. Following the rationale from the previous section, Random Forests ensemble is proclaimed as the optimal one, since this method is considerably more time-efficient than Bagging. Since the majority of attributes is numeric (or can be considered numeric, because they are nominal and binary), Extra Trees i) have the same O time complexity of inducing one node as Random Forests, ii) typically result in bigger trees than Random Forests, the Random Forests ensemble is the most time efficient.

We observe a similar situation when comparing different ensemble methods when the feature ranking score is fixed to Genie3, as shown in Fig. 3b. Again, no statistically significant differences are found ($p = 0.705$), hence Random Forests ensemble is again chosen as the most appropriate one.

As for the Random Forest feature ranking score, Wilcoxon's test detects that Bagging ensemble statistically significantly ($p = 0.030$) outperforms Random Forests ensemble, hence Bagging is the most appropriate one for this feature ranking score.

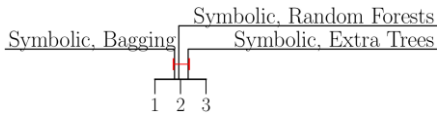

Figure 3a

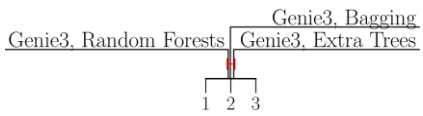

Figure 3b

The quality of different feature rankings as assessed by Friedman's test, when ensemble method varies, and feature ranking score is fixed to (a) Symbolic and (b) Genie3. No statistically significant differences were found.

4.4 Comparison of the Scores

In Sec. 4.1, we have shown that we obtain as good as it gets ensemble-based feature rankings when we grow 10 trees, and as good as it gets HMLC-Relief feature rankings when we set then number of iterations to m = 1% of the training set size $|\mathcal{D}_{\text{TRAIN}}|$. In the previous section, we additionally found the most appropriate ensemble method for a given feature ranking score. Now, we first check whether the three ensemble scores computed with the optimal parameters outperform the HMLC-Relief score, computed with the optimal parameters. This is done in a similar fashion to the Sec. 4.2 by applying three pairwise comparisons via Wilcoxon's test and Benjamini-Hochberg correction.

The differences reported here are always in favor of the ensemble-based scores. The obtained p-values are (sorted in the increasing order): $p_1 = 4.86 \cdot 10^{-5}$ (Symbolic, Random Forests), $p_2 = 1.74 \cdot 10^{-4}$ (Genie3, Random Forests), $p_3 = 1.59 \cdot 10^{-3}$ (Random Forest, Bagging). After applying the correction, all three differences are statistically significant, hence all three ensemble-based rankings outperform the HMLC-Relief ranking.

Next, we investigate which of the ensemble-based scores performs best. To this end, we apply Friedman's test. The obtained p-value equals $2.33 \cdot 10^{-3}$ and we can proceed to the Nemenyi's post-hoc test to discover where the differences occur. The results are shown in the Fig. 4. There are two groups of scores that do not perform statistically significantly different. The first group consist of Symbolic and Genie3 score, and the second one consists of Genie3 and Random Forest score. The graph also reveals Symbolic score (with the average rank of 1.55) performs statistically significantly better than Random Forest score (with the average rank of 2.42). Since the Random Forest score has the worst time complexity, we prefer the other two over it.

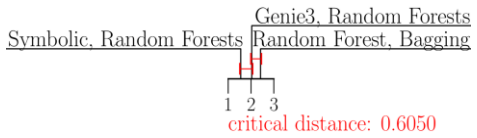

Figure 4
Comparison of the three ensemble-based feature ranking scores

4.5 A Closer Look to Some Other Rankings' Characteristics

A high number of benchmark problems allows for a statistical analysis performed in the previous sections. However, when averaging the performances, some information is always lost, therefore we now take a closer look at two characteristics of the obtained feature rankings in addition to the one already mentioned: the quality of the ranking stabilizes quite quickly (after growing ten trees). We will use the graphs in Figs. 5a and 5b which show the results for hom-ara-FUN dataset, as a running example. One of the reasons for choosing this dataset is that it is high dimensional and is also one the datasets where considering all features when inducing the trees in the Extra Trees ensemble does not work. This is visible from Fig. 5a, which shows the qualities of the Symbolic ranking, computed from different ensembles. Considering only a subset of features in each node (Extra Trees, SQRT) as compared to considering all the features (Extra Trees, all), pushes the quality of the rankings over the baseline.

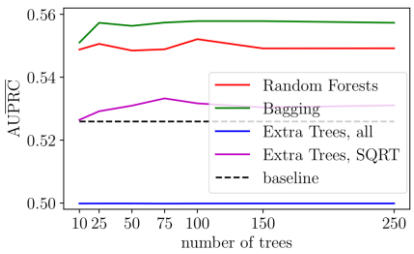

Figure 5a

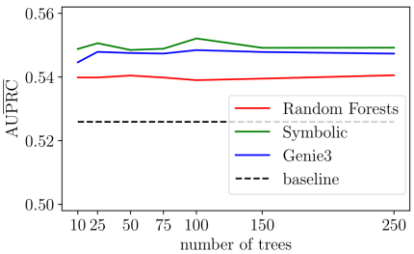

Figure 5b

Quality of the feature rankings on the hom-ara-FUN dataset when number of trees varies and (a) feature ranking score is fixed to Symbolic, (b) ensemble method is fixed to Random Forests. In the figure (a), all and SQRT denote the number of features considered in the Extra Trees algorithm.

The order of the rankings. Fig. 5b depicts typical situation with respect to the order of the feature ranking scores when an ensemble method is fixed. More precisely, only in 4 out of 30 cases, the ranking that belongs to the Random Forest score is placed in between those that belong to Symbolic and Genie3 score. This can be explained by the fact that the mechanisms for computing feature relevance values in the latter two scores, are more similar to each other than to the mechanism of Random Forest score: Symbolic score (number of examples) and Genie3 score (variance reduction) both consider statistics that are somewhat related since number of examples is also part of the variance reduction statistic. Random Forest score, on the other hand, takes a look at the error reduction values.

Efficiency of the rankings. Under efficiency of a ranking, we mean the (relative) number of the features that have a positive feature importance. The lower the number, the more efficient the ranking. In Fig. 5b, all three scores result in

relevant feature rankings, since all three curves lie above the baseline. The dataset at hand has the second highest number of features (72869) and it is surprising that the weighted kNN algorithms which make use of the weights from the rankings, wipo ignores more than 90% of the features - those that were proclaimed irrelevant and have weight 0 (or negative): In the case of Genie3 and Symbolic score, approximately, 8% of the features are being used, whereas in the in the case of the Random Forest score, this number is even lower: 3%. Similar situation was observed for the other extremely high dimensional datasets, e.g., wipo where Random Forest rankings proclaim 99% of the features irrelevant, and reuters. On the other hand, the rankings seem to be less efficient on lower-dimensional datasets. For example, in the case of cellcycle, clef07a-is and gasch1-yeast-FUN datasets, all features have positive importance.

By carefully inspecting the results, we make three main observations. First, Symbolic and Genie3 score columns are equal, but this can be explained by the fact that they are computed from the same ensemble (Random Forests) and the terms in Eq. (1) and Eq. (2) are always positive which is obvious for the Symbolic score, and can be proven with simple algebra for the Genie3 score.

A more interesting observation is that Random Forest ranking is consistently more efficient that the other two ensemble rankings (on 28 of 30 datasets). The reason for this is most likely the fact that Random Forest rankings are computed from the Bagging ensemble which always considers all features when inducing a new node of a tree. If the relevant features can be told apart from the irrelevant ones, then, always one of the relevant features would be chosen in a test split. This does not hold for the Random Forests ensembles which Genie3 and Symbolic scores are computed from, since they consider only a subspace of features, so all (or most of the) relevant ones can be skipped by chance. In addition to that, bootstrapping may also play an important role in this process, especially when the data is sparse which is true for many of the datasets, e.g., yeast-GO, it can happen, that different features are important, for different bootstrap replicates.

The last observation is that HMLC-Relief feature rankings are typically more efficient than the ensemble-based feature rankings. This is another proof that data is sparse, since the second term in Eq. (4) - which can make the relevance negative - should converge to zero when the domain is populated with more and more examples (a sketch of a proof can be found in [17]). However, the efficiency is not correlated with the ranking quality as shown in Sec. 4.4.

The other view on efficiency relates to time efficiency. We can estimate the time complexities $O(F\, m^2 + n\, |\mathcal{L}|)$ for HMLC-Relief and $O(F\, m\, log\, m\, (log\, m + |\mathcal{L}|))$ in the worst case of ensemble-based rankings (using bagging ensemble), where $m = |\mathcal{D}_{\mathrm{TRAIN}}|$. This reveals that ensemble-based rankings are typically more time-efficient than HMLC-Relief rankings (unless the hierarchy size $|\mathcal{L}|$ is sufficiently larger than the other quantities).

Ranking Similarity. The similarity of the two feature rankings is measured in terms of their Fuzzy Jaccard Index (FUJI) score [23] which is defined as follows. Given two rankings $r_i = (x_{(1)}^{\ i}, \ldots, x_{(F)}^{\ i})$, i = 1, 2, where $x_{(j)}^{\ i}$ denotes the *j*-th top-ranked feature in ranking r_i, accompanied by the feature importance score f_j^i, we define the sets $F_j^i = \{x_{(1)}^{\ i}, \ldots, x_{(j)}^{\ i}\}$ as the sets of top-ranked features of ranking r_i. Finally, the fuzzy membership function μ of the feature $x_{(k)}^{\ i}$ for the set F_j^i is defined as $\mu(F_j^i, x_{(k)}^{\ i}) = min\{1, f_k^i / f_j^i\}$, and $FUJI(F_j^1, F_j^2)$ is defined as:

$$FUJI\left(F_j^1, F_j^2\right) = \left(\sum_{x \in F_j^1 \cup F_j^2} \min_i \mu(F_j^i, x)\right) / \left(\sum_{x \in F_j^1 \cup F_j^2} \max_i \mu(F_j^i, x)\right)$$

This score is computed for $1 \leq i \leq F$ and the final similarity measure is the area under the FUJI curve consisting of the points (j, $FUJI(F_j^1, F_j^2)$). We compute this for all 6 pairs of the feature ranking scores and for all datasets. In Fig. 6, we present the distance $d = 1 - FUJI$ between the scores, averaged over the datasets.

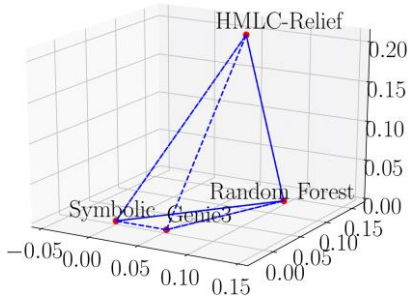

Figure 6

Tetrahedron whose vertices correspond to the feature ranking scores. The length of a side between two vertices equals the average distance $d = 1 - FUJI$ between the corresponding two feature ranking scores.

We can see that the Symbolic and Genie3 score produce the most similar rankings which is explained by the fact that they both consider tree node statistics. The closest to these two scores is the Random Forest score which reflects the fact that HMLC-Relief is the only non-ensemble-based score among the analyzed scores.

Conclusions

In this work we proposed three feature ranking scores, Symbolic, Genie3 and the Random Forest score, for the task of HMLC. The proposed feature ranking methods can be computed very efficiently, since it suffices to grow only 10 trees in the ensemble. The first two scores yield relevant feature ranking, while Random Forest score, fails. For the Symbolic and Genie3 score, the most suitable ensemble method is Random Forests, whereas bagging is the most suitable for Random Forest score. When coupled with the suitable ensemble method, all three scores outperform the HMLC-Relief feature ranking. Moreover, the Random Forest score is statistically and significantly, outperformed by the Symbolic score. Therefore, we recommend using either the latter or Genie3 score, since there are no

statistically significant differences among these two (but the Symbolic score has the lowest rank on average). We have also shown that Symbolic and Genie3 score are more closely related to each other than to Random Forest score. Especially on the extremely high-dimensional datasets, all three feature ranking scores successfully filtered out a majority of the features and still outperform the baseline that uses all of them. The HMLC-Relief feature rankings are even more efficient in that sense, but they are of lower quality.

This work can be extended in at least two directions. First, we could improve the HMLC-Relief, so that its performance would be comparable to the ensemble-based rankings. Second, we could extend the ensemble-based scores to the gradient boosting ensemble technique, which is inherently different from those presented herein, since, the trees are not independent of each other, which also allows for the analysis of the development of the ranking through the iterations.

Acknowledgement

We acknowledge the financial support of the Slovenian Research Agency (grant P2-0103 and a young researcher grant to MP), the European Commission (the grants MAESTRA (Learning from Massive, incompletely annotated, and Structured Data) and HBP (The Human Brain Project), SGA1 and SGA2. SD also acknowledges support by Slovenian Research Agency (via grants J4-7362, L2-7509, and N2-0056), the European Commission (project LANDMARK) and ARVALIS (project BIODIV). The experiments presented here were executed on the computing infrastructure from the Slovenian Grid (SLING) initiative.

References

[1] Barutcuoglu, Z., Schapire, R. E., & Troyanskaya, O. G. (2006) Hierarchical multi-label prediction of gene function. Bioinformatics, 22, 830-836

[2] Benjamini, Y., & Hochberg, Y. (1995) Controlling the False Discovery Rate: A Practical and Powerful Approach to Multiple Testing. Journal of the Royal Statistical Society, 57, 289-300, doi:10.2307/2346101

[3] Blockeel, H. (1998) Top-down Induction of First Order Logical Decision Trees. Ph.D. dissertation, Katholieke Universiteit Leuven, Leuven

[4] Breiman, L. (2001) Random Forests. Machine Learning, 45, 5-32

[5] Breiman, L., Friedman, J., Olshen, R. A., & Stone, C. J. (1984) Classification and Regression Trees. Chapman & Hall/CRC

[6] Clare, A. (2003) Machine learning and data mining for yeast functional genomics. Ph.D. dissertation, University of Wales Aberystwyth, Aberystwyth

[7] Consortium, T. G. (2000) Gene Ontology: tool for the unification of biology. Natural Genetics, 25, 25-29

[8] Demšar, J. (2006) Statistical comparisons of classifiers over multiple data sets. Journal of Machine Learning Research, 7, 1-30

[9] Dimitrovski, I., Kocev, D., Loskovska, S., & Džeroski, S. (2008) Hierchical annotation of medical images. Proceedings of the 11th International Multiconference - Information Society IS 2008 (str. 174-181) IJS, Ljubljana

[10] Dimitrovski, I., Kocev, D., Loskovska, S., & Džeroski, S. (2010) Detection of Visual Concepts and Annotation of Images Using Ensembles of Trees for Hierarchical Multi-Label Classification. Recognizing Patterns in Signals, Speech, Images and Videos (str. 152-161) Springer

[11] Dimitrovski, I., Kocev, D., Loskovska, S., & Džeroski, S. (2011) Hierarchical Classification of Diatom Images using Predictive Clustering Trees. Ecological Informatics

[12] Hansen, L. K., & Salamon, P. (1990) Neural network ensembles. IEEE Transactions on Pattern Analysis and Machine Intelligence, 12, 993-1001

[13] Huynh-Thu, V. A., Irrthum, Wehenkel, L., & Geurts, P. (2010) Inferring Regulatory Networks from Expression Data Using Tree-Based Methods. PLoS One, 5, 1-10

[14] Klimt, B., & Yang, Y. (2004) The Enron Corpus: A New Dataset for Email Classification Research. ECML '04: Proceedings of the 18th European Conference on Machine Learning -- LNCS 3201 (str. 217-226) Springer

[15] Kocev, D. (2011) Ensembles for predicting structured outputs. Ph.D. dissertation, IPS Jožef Stefan, Ljubljana

[16] Kocev, D., Vens, C., Struyf, J., & Džeroski, S. (2013) Tree ensembles for predicting structured outputs. Pattern Recognition, 46, 817-833

[17] Kononenko, I., & Robnik-Šikonja, M. (2003) Theoretical and Empirical Analysis of ReliefF and RReliefF. Machine Learning Journal, 55, 23-69

[18] Levatić, J., Kocev, D., & Džeroski, S. (2015) The Importance of the Label Hierarchy in Hierarchical Multi-label Classification. Journal of Intelligent Information Systems, 45, 247-271

[19] Lewis, D. D., Yang, Y., Rose, T. G., & Li, F. (2004) RCV1: A New Benchmark Collection for Text Categorization Research. Journal of Machine Learning Research, 5, 361-397

[20] Madjarov, G., Gjorgjevikj, D., Dimitrovski, I., & Džeroski, S. (2016) The use of data-derived label hierarchies in multi-label classification. Journal of Intelligent Information Systems, 47, 57-90

[21] Petković, M., Džeroski, S., & Kocev, D. (2018) Feature Ranking with Relief for Multi-label Classification: Does Distance Matter? V L.

Soldatova, J. Vanschoren, G. Papadopoulos, & M. Ceci (Ured.), Discovery Science (str. 51-65) Springer

[22] Petković, M., Kocev, D., & Džeroski, S. (2019) Feature ranking for multi-target regression. Machine Learning

[23] Petković, M., Lucas, L., Kocev, D., Džeroski, S., Boumghar, R., & Simidjievski, N. (2019) Quantifying the effects of gyroless flying of the Mars Express Spacecraft with machine learning. 2019 7th International Conference on Space Mission Challenges for Information Technology (SMC-IT)

[24] Rousu, J., Saunders, C., Szedmak, S., & Shawe-Taylor, J. (2006) Kernel-Based Learning of Hierarchical Multilabel Classification Models. Journal of Machine Learning Research, 7, 1601-1626

[25] Schietgat, L., Vens, C., Struyf, J., Blockeel, H., Kocev, D., & Džeroski, S. (2010) Predicting gene function using hierarchical multi-label decision tree ensembles. BMC Bioinformatics, 11

[26] Slavkov, I., Karcheska, J., Kocev, D., & Džeroski, S. (2018) HMC-ReliefF: Feature Ranking for Hierarchical Multi-label Classification. Computer Science and Information Systems, 15, 187-209

[27] Stańczyk, U., & Jain, L. C. (Ured.) (2015) Feature Selection for Data and Pattern Recognition. Springer

[28] Tsoumakas, G., & Katakis, I. (2007) Multi-label classification: An overview. International Journal of Data Warehousing and Mining, 1-13

[29] Vens, C., Struyf, J., Schietgat, L., Džeroski, S., & Blockeel, H. (2008) Decision trees for hierarchical multi-label classification. Machine Learning, 73, 185-214

[30] Wettschereck, D. (1994) A study of distance based algorithms. Ph.D. dissertation, Oregon State University, USA