Hierarchy Decomposition Pipeline: A Toolbox for Comparison of Model Induction Algorithms on Hierarchical Multi-label Classification Problems

Vedrana Vidulin(✉) and Sašo Džeroski

Jožef Stefan Institute, Jamova cesta 39, 1000 Ljubljana, Slovenia
vedrana.vidulin@gmail.com, saso.dzeroski@ijs.si

Abstract. Hierarchical multi-label classification (HMC) is a supervised machine learning task, where each example can be assigned more than one label and the possible labels are organized in a hierarchy. HMC problems emerge in domains like functional genomics, habitat modelling, text and image categorization. They can be addressed with global model induction algorithms, which induce a single model that predicts the complete hierarchy, as well as with local algorithms, which induce multiple models that predict different segments of the hierarchy. However, there is no consensus about which of these approaches perform the best over different domains, especially in the setting of learning ensembles.

We introduce the hierarchy decomposition pipeline, a publicly available toolbox for comparison of model induction algorithms on HMC problems in an ensemble setting. The pipeline includes five algorithms, including the algorithm that predicts the complete hierarchy, and algorithms that perform partial and complete hierarchy decompositions. One of these algorithms is the novel "label specialization" algorithm that constructs a local multi-label classification model for each parent label in a hierarchy that simultaneously predicts the respective children labels.

We apply the pipeline on ten HMC data sets from four domains, which have both tree and directed acyclic graph label hierarchies, and confirm that there is no single best algorithm for all HMC problems. This finding shows that there exists a need for such a pipeline that enables a user to choose the best performing algorithm for his/her HMC data set. Finally, we show that the choice can be narrowed to a specific type of algorithm, based on the characteristics of the label hierarchy and the data set label cardinality.

Keywords: Hierarchical multi-label classification · Hierarchy decomposition · Structured prediction

1 Introduction

Hierarchical multi-label classification (HMC) is a supervised machine learning task, where each example can be assigned more than one label and the possible labels are

© Springer Nature Switzerland AG 2020
A. Appice et al. (Eds.): DS 2020, LNAI 12323, pp. 486–501, 2020.
https://doi.org/10.1007/978-3-030-61527-7_32

organized in a hierarchy [1]. The hierarchy can be in the shape of a tree, where each label has exactly one parent label, or in the form of a directed acyclic graph (DAG), where a label can have multiple parent labels. Label assignments follow the hierarchy constraint: When a label is assigned to an example, all labels on all possible paths from that label to the root of the hierarchy must be assigned too.

Many real life problems are best represented with HMC data sets [2–10]. An example of a HMC problem is gene function prediction, which aims to predict the biological functions of genes. Examples of gene function are the tree shaped hierarchy of FunCat [11] and the more comprehensive DAG shaped hierarchy of the Gene Ontology [12]. The latter is composed of three domains – molecular function, biological process and cellular component – and a single gene can be assigned with multiple functions from each of the three domains [13].

Model induction algorithms for HMC problems can be divided into two groups [14, 15]. Global algorithms induce a single model that predicts complete hierarchy. They can exploit dependencies among labels during a model induction phase to improve model's predictive performance. An example of the global algorithms is Clare and King [16] adaptation of the decision tree algorithm C4.5 [17], which predicts labels on different levels of a hierarchy by assigning a larger cost to misclassification high up in the hierarchy. Another example is the predictive clustering tree (PCT) algorithm, a generalization of the decision tree algorithm that predicts labels from both tree [18–20] and DAG hierarchies [1]. Local algorithms induce multiple models that predict a different part of a hierarchy, and then combine the predictions of those models. Some examples of a local algorithm construct an SVM model for each label and then combine predictions so as to satisfy the hierarchy constraint [21–24].

Levatić et al. [25] compare the predictive performance of four model induction algorithms over HMC problems from different domains. They compare two global and two local algorithms, where one in each group exploits the hierarchical dependencies among labels when constructing model(s) and the other does not. Both global algorithms construct one multi-label model, while both local algorithms construct many single-label classification models. All four approaches construct single PCT models of (random forest and bagging) ensembles of PCTs. When a single PCT models are constructed, the algorithms that exploit hierarchical dependencies outperform those that do not. However, when PCT ensembles are constructed, it is less clear what is the best performing algorithm.

We introduce the hierarchy decomposition pipeline, a publicly available toolbox for comparison of model induction algorithms for HMC problems in the ensemble setting (https://github.com/vedranav/hierarchy-decomposition-pipeline). The pipeline includes five algorithms, beginning with an algorithm that predicts the complete hierarchy in one shot, and following with four algorithms that perform partial and complete hierarchy decompositions. Partial decomposition algorithms construct models that predict the presence of one or more edges of the hierarchy, while complete decomposition algorithms construct model(s) that predict the presence of individual or all hierarchy nodes. We propose a novel partial decomposition algorithm, called the "label specialization", that constructs a multi-label classification model for each parent label in a hierarchy, which predicts the presence of its children labels. The algorithm is an extension of the hierarchical single label classification algorithm [1] that constructs a single-label classification model for each parent-child pair in a hierarchy, where collection of such models for a given parent can be viewed as a binary relevance classifier. Apart from the mentioned algorithms, the pipeline contains tools for performance-based comparison of the algorithms.

We applied the pipeline on ten HMC data sets from four domains. In the text categorization domain, we use the Enron data set that categorizes e-mails from the Enron corporation officials [4] and the Reuters data set that categorizes Reuters stories [5]. In the image categorization domain, we use two data sets from the 2007 CLEF image retrieval campaign that categorize medical X-ray images [7]. In the habitat modelling domain, we use the Danish farms data set that models the habitats of soil microarthropods [6] and the Slovenian rivers data set that models the habitats of aquatic organisms [2]. In the functional genomics domain, we use two data sets intended for predicting biological functions of genes in two model organisms: the plant *Arabidopsis thaliana* and the baker's or brewer's yeast *Saccharomyces cerevisiae* [3]. In addition, we use two data sets intended for predicting functions of genes in thousands of bacterial and archaeal organisms [8, 9]. In the first eight data sets, the labels are interconnected in tree shaped hierarchies, while in the last two the labels form a DAG.

The results of the performance comparison confirm that there is no single best model induction algorithm for all HMC data sets in the ensemble setting. There is no significant difference in the predictive performance of the five algorithms over the ten data sets. This finding shows that there exists a need for the proposed pipeline, which enables a user to find the best performing algorithm for his/her custom HMC data set. Finally, the results show that the search for the best performing algorithm can be narrowed to a specific type of an algorithm based on the characteristics of the hierarchy and the data set cardinality.

The remainder of the paper is organized as follows. In Sect. 2, we describe the hierarchy decomposition pipeline. Section 3 describes the experimental setup, including values of the pipeline's parameters and the HMC data sets. The results of the performance analysis are presented in Sect. 4. We conclude the paper with Sect. 5.

2 Hierarchy Decomposition Pipeline

The hierarchy decomposition pipeline is a toolbox for comparing model induction algorithms for HMC problems in the ensemble setting. The pipeline takes as input a HMC data set specified by a user and applies five model induction algorithms, beginning with the algorithm that induces a model predicting the complete hierarchy and continuing with partial and complete hierarchy decomposition algorithms. The performance-based evaluation tool computes the areas under the average precision-recall curves and performs a statistical test. The components of the pipeline are shown in Fig. 1.

The pipeline receives two **input** files: an HMC data set and a settings file (Fig. 1A). A description of the input file formats is available on the repository.

The **cross-validation module** takes the data set and divides its examples into folds (Fig. 1B). For each fold, it sends a training set to the hierarchy decomposition and the model induction modules, and a test set to the annotation module.

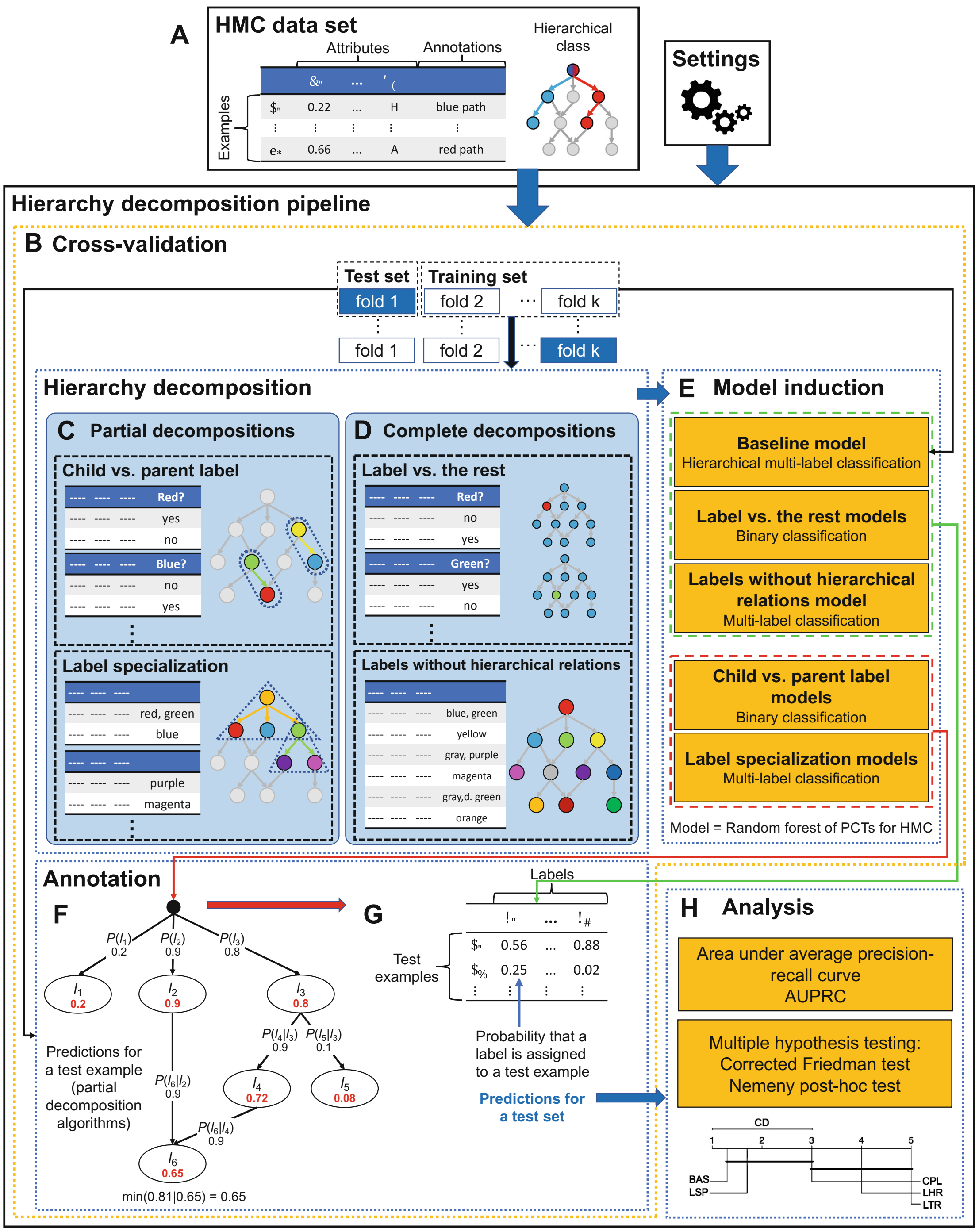
A
HMC data set
Attributes
Annotations
Hierarchical class
Examples
0.22
H
blue path
0.66
A
red path
Settings
Hierarchy decomposition pipeline
B Cross-validation
Test set
Training set
fold 1
fold 2
fold k
Hierarchy decomposition
C Partial decompositions
Child vs. parent label
Red?
yes
no
Blue?
no
yes
Label specialization
red, green
blue
purple
magenta
D Complete decompositions
Label vs. the rest
Red?
no
yes
Green?
yes
no
Labels without hierarchical relations
blue, green
yellow
gray, purple
magenta
gray,d. green
orange
E Model induction
Baseline model
Hierarchical multi-label classification
Label vs. the rest models
Binary classification
Labels without hierarchical relations model
Multi-label classification
Child vs. parent label models
Binary classification
Label specialization models
Multi-label classification
Model = Random forest of PCTs for HMC
Annotation
F
Predictions for a test example (partial decomposition algorithms)
min(0.81|0.65) = 0.65
G
Labels
Test examples
0.56
0.88
0.25
0.02
Probability that a label is assigned to a test example
Predictions for a test set
H Analysis
Area under average precision-recall curve AUPRC
Multiple hypothesis testing: Corrected Friedman test Nemeny post-hoc test
CD
BAS
LSP
CPL
LHR
LTR

Fig. 1. The hierarchy decomposition pipeline.

The **hierarchy decomposition module** transforms an input training set into multiple training sets by applying two types of decomposition:

Partial decompositions construct multiple training sets representing different edges of a hierarchy (Fig. 1C). The first partial decomposition "*child vs. parent label*" constructs a binary training set for each child-parent label pair in a hierarchy, composed of the training examples originally labeled with the parent label. In a newly created training set, the examples originally labeled with the child label are labeled as positive, while the rest of the examples are labeled as negative. The second partial decomposition, "*label specialization*", constructs a multi-label training set for each parent-children group of labels in a hierarchy, where the training set contains all examples originally labeled with the parent label, now only labeled with the applicable children labels.

Complete decompositions construct one or multiple training sets representing the nodes of the hierarchy, and ignoring the edges (Fig. 1D). The training set(s) contain the same examples as the input training set, but annotated with the labels that belong to the subset of most specific annotations. For example, if an example is originally labeled with two paths "root $>$ 1 $>$ 1.1" and "root $>$ 2 $>$ 2.1 $>$ 2.1.1", the example will be newly labeled with the most specific annotations 1.1 and 2.1.1. Accordingly, the subset of most specific annotations contains the labels that appear at least once as the most specific annotation in the input data set. The first complete decomposition "*label vs. the rest*" constructs a binary training set for each most specific annotation, where the examples originally annotated with the label are newly labeled as positive and the rest of the examples as negative. The second "*labels without hierarchical relations*" constructs a single multi-label training set that captures label cooccurrences by labeling examples with one or multiple labels that qualify as most specific annotations.

The **model induction module** constructs classification models from the input training set and the training sets created by the hierarchy decomposition module (Fig. 1E). The task of constructing the baseline model from the input training set is a HMC task, and the tasks of constructing models from the decomposed training sets are multi-label and binary classification tasks. Consequently, we choose PCTs as a base model, since the PCT algorithm covers all three modeling tasks in a unified framework. For each training set, a random forest of PCTs is constructed using CLUS [1].

Annotation module classifies an input test set by using the models created by the model induction module. It combines the predictions from multiple models and applies the hierarchy constraint (Fig. 1F, G). For each of the five model induction algorithms it outputs a table with predictions, where rows are test examples, columns labels and values probabilities that the labels are assigned to the examples. The tables are obtained in the following manner:

The *baseline* model is the global model that implicitly enforces the hierarchy constraint when annotating test examples. We use the outputted predictions as given by the model.

The *child vs. parent label* model collection is composed of multiple binary classification models, one for each non-root label l_j that outputs a conditional probability $P(l_j|\text{parent}(l_j))$. To make a prediction for a test example e_i and a label l_j, the product rule $P(l_j) = P(l_j|\text{parent}(l_j)) \cdot P(\text{parent}(l_j))$ is applied recursively, beginning with the model where $\text{parent}(l_j)$ is the root of a hierarchy. The procedure is illustrated with an example in Fig. 1F. To compute the probability that l_5 is assigned to e_i, we first use the model for the label l_3 to predict $P(l_3)$, which is 0.8. Then, we use the model for the label l_5 that predicts $P(l_5|l_3)$, which is 0.1. Finally, $P(l_5)$ is computed by applying the product rule $P(l_5|l_3) \cdot P(l_3)$, which is 0.08. By using the product rule we enforce the hierarchy constraint, ensuring that the probabilities decrease with increasing depth within the hierarchy. The presented example shows the case with a single path from a label node to root of a hierarchy. When a class is a DAG, there can be multiple paths from the label node to the root node. In this case, the probability is computed for each path, and the minimal observed probability is considered (P(l_6) in Fig. 1F).

The *label specialization* model collection is composed of multiple multi-label classification models, one for each parent node in the hierarchy, which output conditional probabilities P(l_j|parent(l_j)) for children labels. When the output space is a DAG, a label l_j can have multiple parents and, consequently, multiple multi-label models can output a conditional probability P(l_j|parent(l_j)). In this case, we consider as P(l_j|parent(l_j)) the maximal predicted conditional probability. To predict a label l_j for a test example e_i, we then use the product rule as in the case of "child vs. parent label" model.

The *label vs. the rest* model collection is composed of multiple binary classification models, one for each label l_j that qualifies as most specific annotation. For a test example e_i and a label l_j from the subset of most specific annotations, an l_j specific model outputs the probability P(l_j) that the label is assigned to the example. In the case of labels that do not belong to the subset of most specific annotations, the probability is zero.

The *labels without hierarchical relations* model is a single multi-label classification model that can output probabilities that labels from the subset of most specific annotations are assigned to an example.

The **analysis module** compares the performances of the five model induction algorithms, based on the predictions output by the annotation module. An algorithm's performance is measured as the area under average precision-recall curve (AUPRC) [1]. The statistical significance of AUPRC differences is assessed by using the corrected Friedman test and the Nemenyi post-hoc test [26] (Fig. 1H).

AUPRC is a threshold independent performance measure, where precision and recall points are obtained by changing the value of the threshold t from zero to one with the step of 0.01. For each value of t, precision and recall values are micro-averaged: $\overline{precision_t} = \frac{\sum_{i=1}^{p} TP_i}{\sum_{i=1}^{p} TP_i + \sum_{i=1}^{p} FP_i}$, $\overline{recall_t} = \frac{\sum_{i=1}^{p} TP_i}{\sum_{i=1}^{p} TP_i + \sum_{i=1}^{p} FN_i}$, where p is the number of labels that qualify as most specific annotations, TP are true positives, FP false positives and FN false negatives.

The statistical test is performed on a r by k matrix, where r is the number of model induction algorithms (five in our case), k is the number of cross-validation folds and the values in the matrix are AUPRCs. The corrected Friedman test determines if there is at least one algorithm with significantly different performance. For each fold, the test ranks the algorithms in decreasing order of AUPRC. In case of a tie, an average rank is assigned. Next, the test averages ranks over the k folds and calculates the Friedman statistic Q, distributed according to the χ^2 distribution with $r - 1$ degrees of freedom. The p-value is defined as $P\left(\chi^2_{r-1} \geq Q\right)$. If, according to the p-value, the difference is significant, the Nemenyi post-hoc test is used for pairwise comparisons among the algorithms. The performance of two algorithms is significantly different if their average ranks differ by more than a critical distance. The critical distance is computed from r, k and a critical value for a given significance level (a Studentized range statistic).

3 Experimental Setup

We applied the hierarchy decomposition pipeline on ten data sets, using a unified experimental setup. The unified setup means that all models are random forests of 500 PCTs.

The size of random subspaces considered at each node is equal to the square root of the number of attributes, and the five model induction algorithms are evaluated by performing 10-fold cross-validation.

The ten data sets used are described below. They represent four domains: text categorization, image categorization, habitat modelling and functional genomics. Eight of them have a tree-shaped label hierarchy and two a DAG shaped label hierarchy. Data set statistics are presented in Table 1.

Table 1. Data set statistics. Columns: n = number of examples; a_d = discrete attributes; a_n = numeric attributes; h_n = hierarchy nodes; h_l = hierarchy leaves; $c\text{-}h_l$ = cardinality accounting for labels that are hierarchy leaves (cardinality is an average number of labels per example); p = labels that qualify as most specific annotations; $c\text{-}p_c$ = cardinality accounting for most specific annotations available to the complete decomposition algorithms; $c\text{-}p_h$ = cardinality accounting for most specific annotations available to the hierarchical algorithms; d = maximal depth of the hierarchy; type = tree or DAG hierarchy.

Data set	n	a_d/a_n	h_n	h_l	$c\text{-}h_l$	p	$c\text{-}p_c$	$c\text{-}p_h$	d	Type
Enron	1,648	1,001/0	56	52	2.85	53	2.87	3.37	3	Tree
Reuters	6,000	0/47,236	100	79	1.19	99	1.46	3.13	4	Tree
ImCLEF07A	11,006	0/80	96	63	1.00	63	1.00	1.00	3	Tree
ImCLEF07D	11,006	0/80	46	26	1.00	26	1.00	1.00	3	Tree
Danish farms	1,893	132/5	70	35	6.27	39	6.74	7.08	3	Tree
Slo. rivers	1,060	0/16	724	492	24.56	637	33.04	50.67	4	Tree
ExprYeast	3,788	4/547	417	161	2.28	194	2.29	4.00	4	Tree
SeqAra	3,718	2/4,448	196	148	0.94	194	1.30	3.32	4	Tree
PP	15,313	2,071/0	1,260	377	0.89	947	2.59	16.67	14	DAG
MPP-I	3,531	0/4,777	826	220	1.32	620	3.30	20.49	13	DAG

Text categorization is a problem of automatic annotation of textual documents with one or several categories. The Enron data set contains bag-of-words descriptions of e-mails from the labeled subset of the Enron corpus [4]. Hierarchically organized categories define genre, emotional tone and topic. Reuters data set contains tf-idf descriptions of stories from the “Topics” category of the Reuters Corpus Volume I (RCV1) [5]. Hierarchically organized categories are topic-based, e.g., economics, industrial or government.

Image categorization annotates images with categories that represent visual concepts the images contain. ImCLEF07A and ImCLEF07D represent medical X-ray images annotated with parts of the human anatomy and orientations of body parts [7]. The images are described with edge histograms indicating a frequency and a directionality of brightness changes in an image.

Habitat modelling studies relationships between environmental variables and the presence of plants and animals in the environment. The Danish farms data set represents habitats of soil microarthropods on Danish experimental and organic farms [6]. The Slovenian rivers data set represents habitats of aquatic organisms in Slovenian rivers [2]. The tree-shaped hierarchies in both data sets represent parts of the taxonomic hierarchy that contain habitat-specific species.

Functional genomics annotates genes with their biological functions. The ExprYeast data set represents *Saccharomyces cerevisiae* (baker's yeast) microarray gene expression levels measured under various experimental conditions, such as heat shock or nitrogen depletion [3]. The SeqAra data set contains attributes derived from amino acid sequences of the *Arabidopsis thaliana* plant genes, such as amino acid ratios, molecular weight and sequence length [3]. The PP data set represents phyletic profiles, i.e., presence and absence patterns of gene families (clusters of genes that share function) in 2,071 bacterial and archaeal genomes [8]. The MPP-I data set represents metagenome phyletic profiles, i.e., relative abundances of gene families in metagenomes obtained from the IMG database [9]. The first two data sets are annotated with functions from the tree shaped hierarchy of FunCat [11] and the last two with functions from the DAG of the Gene Ontology [12].

4 Results

The analysis has three goals. First, to clarify how model induction algorithms for HMC problems can be compared. Second, to examine whether there exists a single best performing algorithm for all ten HMC problems. Third, to investigate whether specific properties of HMC data sets can be related to a type of the best performing algorithm.

4.1 How Model Induction Algorithms for HMC Problems Can Be Compared?

The performance-based comparison of model induction algorithms should be ideally based on annotations available to all of the algorithms. The pipeline contains two types of algorithms. The complete decomposition algorithms use only the most specific annotations, while the hierarchical algorithms use additional annotations obtained by enforcing the hierarchy constraint. To compare the two types of algorithms we aggregate AUPRC over the subset of labels common to both types, that is, over the labels that qualify as most specific annotations. However, this step does not guarantee that the comparison is performed on the common set of annotations.

Distributions of common labels are not necessarily the same in the training sets created by the complete decomposition algorithms and the hierarchical algorithms. This property is best illustrated with an example. Suppose that we have a data set annotated with labels from the hierarchy in Fig. 2A and a derived training set composed of five examples annotated with most specific annotations as presented in Fig. 2B. The training set, which illustrates the annotations available to complete decomposition algorithms, shows that six labels qualify as most specific annotations: "1", "1.1", "1.1.1", "1.1.2", "1.2" and "2.2". When the hierarchical algorithms apply the hierarchy constraint on those six labels, the training set will look like the one in Fig. 2C. The difference in distribution of common labels is considerable: the cardinality doubled.

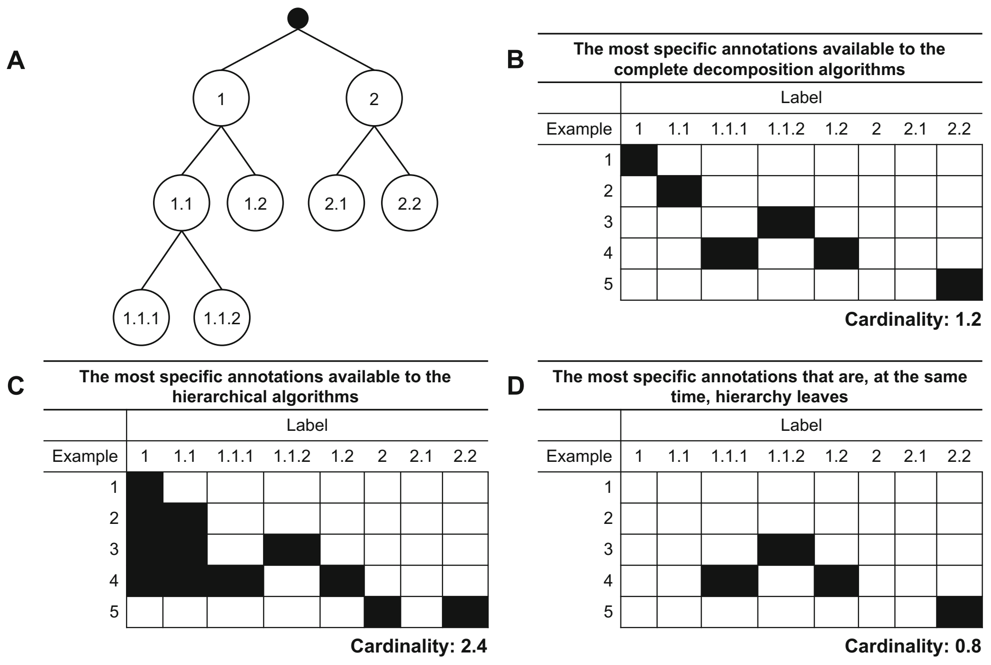
A
1
2
1.1
1.2
2.1
2.2
1.1.1
1.1.2
B
The most specific annotations available to the complete decomposition algorithms
Label
Example
Cardinality: 1.2
C
The most specific annotations available to the hierarchical algorithms
Cardinality: 2.4
D
The most specific annotations that are, at the same time, hierarchy leaves
Cardinality: 0.8

Fig. 2. Example illustrating differences in distribution of labels common to the complete decomposition and hierarchical algorithms.

The difference in distribution of common labels can give an advantage to the hierarchical algorithms due to additional information from the hierarchy. This property affects eight data sets with an exception of the two data sets from image categorization domain (see columns $c\text{-}p_c$ and $c\text{-}p_h$ in Table 1) and tends to have higher impact on the data sets with larger hierarchies.

The presented problem can be addressed by considering only those annotations that are, at the same time, hierarchy leaves. This approach would, however, ignore many annotations (Fig. 2D, see columns $c\text{-}h_l$ and $c\text{-}p_c$ in Table 1). For example, the cardinality of the SeqAra and PP data sets would fall below one, although both data sets have at least one label per example (column $c\text{-}h_l$ in Table 1).

4.2 Is There a Single Best Model Induction Algorithm Across All HMC Data Sets?

To answer this question we: (1) measure AUPRC_m aggregated over the labels that qualify as most specific annotations (Table 2); (2) measure AUPRC_l aggregated over the labels that qualify as most specific annotations and are, at the same time, hierarchy leaves (Table 2); and (3) examine whether the differences in AUPRC_m and AUPRC_l among the five model inductions algorithms are statistically significant at the significance threshold of 0.05.

As a statistical significance test, we use the corrected Friedman test on the matrix where rows are the ten data sets, columns are the five algorithms and values are AUPRCs. We apply the test separately for each type of AUPRC, using the two matrices in Table 2. The p-value for AUPRC_m is 0.029 and for AUPRC_l 0.458. At a significance threshold of 0.05, there are no significant differences in performance considering AUPRC_l.

Table 2. Area under average precision-recall curve aggregated over the labels that qualify as most specific annotations ($AUPRC_m$) and labels that are hierarchy leaves ($AUPRC_l$). AUPRCs of the best performing algorithms are shown in bold. Abbreviations: BAS = baseline, CPL = child vs. parent label, LSP = label specialization, LHR = labels without hierarchical relations, LTR = label vs. the rest, alg. = algorithms, d. = decomposition algorithms.

Data set	$AUPRC_m$					$AUPRC_l$				
	Hierarchical alg.			Complete d.		Hierarchical alg.			Complete d.	
	BAS	CPL	LSP	LHR	LTR	BAS	CPL	LSP	LHR	LTR
Enron	0.646	0.648	**0.657**	0.533	0.532	0.596	**0.601**	0.600	0.594	0.595
Reuters	0.798	**0.816**	0.797	0.446	0.462	0.668	**0.692**	0.661	0.632	0.692
ImCLEF07A	0.886	0.891	0.889	0.888	**0.898**	0.886	0.891	0.889	0.888	**0.898**
ImCLEF07D	0.872	0.871	0.870	0.872	**0.882**	0.872	0.871	0.870	0.872	**0.882**
Danish farms	0.824	0.815	0.825	0.816	**0.827**	0.828	0.819	0.828	**0.830**	0.828
Slo. rivers	**0.658**	0.642	0.657	0.456	0.432	0.504	0.495	**0.510**	0.509	0.486
ExprYeast	0.465	0.449	**0.489**	0.407	0.320	0.372	0.381	**0.415**	0.401	0.347
SeqAra	0.498	**0.524**	0.488	0.238	0.230	0.381	0.395	0.385	0.381	**0.410**
PP	0.341	**0.349**	0.345	0.095	0.097	0.127	0.130	0.129	**0.157**	0.151
MPP-I	0.497	**0.511**	0.507	0.348	0.342	0.301	**0.472**	0.427	0.392	0.398

For $AUPRC_m$, we proceed to the post hoc test (Fig. 3). At the significance level of 0.05, the post hoc test shows that none of the algorithms perform significantly better that the rest. Given the results of the statistical test, we confirm the hypothesis that there is no single best model induction algorithm across all HMC data sets.

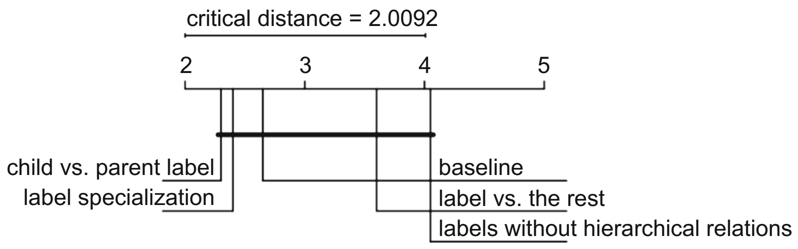

Fig. 3. Average ranks diagram comparing predictive performance, measured as $AUPRC_m$, of the five model induction algorithms over the ten HMC data sets. The numbers on the line represent ranks of the algorithms averaged over the data sets. Better performing algorithms are on the left-hand side. The algorithms with average ranks that differ by less than the critical distance for a p-value of 0.05 are connected with a line.

4.3 Can We Relate Properties of HMC Data Sets to a Type of the Best Performing Model Induction Algorithm?

The best performing algorithm for a data set is the one that receives the highest rank in the statistical test (described in Sect. 2, analysis module), and is significantly better than at least one other algorithm at the significance level of 0.05 (Fig. 4). The second criterion is not satisfied by any of the five algorithms on the PP and MPP-I data sets (Fig. 4I, J). Since we cannot determine the best performing algorithm on the two data sets, they are not going to be used in the analysis. We characterize the best performing algorithm along each of the two dimensions: single- or multi-label classification algorithm, and hierarchical or complete decomposition algorithm. The former dimension indicates whether the model(s) constructed by the algorithm perform(s) single or multi-label classification. The latter indicates whether hierarchical constraint is applied.

Data sets are characterized with two groups of properties, the first describing the hierarchy of labels and the second describing the density of annotations. The hierarchy is described in terms of the number of nodes and leaves, and a branching factor. Annotations are described through cardinality computed both for annotations available to the hierarchical and complete decomposition algorithms. The number of annotations available to the hierarchical, but not to the complete decomposition algorithms is measured as a difference between the two cardinalities. Finally, we measure a share of incomplete annotations in most specific annotations. The most specific annotation is incomplete if it is not a leaf label and can, consequently, be further specialized.

Multi-label classification algorithms perform best on the data sets with large hierarchies: they perform the best on the two data sets with the largest hierarchies, ExprYeast and Slovenian rivers (417 and 724 nodes, Table 3). An average branching factor, which is an indicator of complexity, is, however, not related in the same way. We expected that multi-label classification algorithms would perform best on data sets with high cardinality, but this is not the case. The ExprYeast and Enron data sets have moderate cardinality (from 2.29 to 4), and a multi-label classification algorithm performs best on the former and a single-label classification algorithm performs best on the latter. Similarly, the Danish farms and Slovenian rivers data sets have high cardinality (from 6.74 to 50.67), and a single-label classification algorithm performs best on the former and a multi-label classification algorithm on the latter. An exception are the data sets with low cardinality (less than two) where a single-label classification algorithm always performs best.

Hierarchical algorithms perform better on data sets where they can obtain additional annotations (by applying the hierarchy constraint), as compared to the complete decomposition algorithms. They profit even when only half of an annotation on average is added to examples (Table 4). Interestingly, when the best performing algorithm is a hierarchical algorithm, it performs significantly better than both complete decomposition algorithms (Fig. 4A, B, F–H). Furthermore, the presence of incomplete annotations in a data set is related to the emergence of a hierarchical algorithm as the best performing: When at least 1% of the annotations in a data set are incomplete, a hierarchical algorithm is the best choice. Hierarchical algorithms also perform the best on class hierarchies with more than 100 nodes and an average branching factor higher than three.

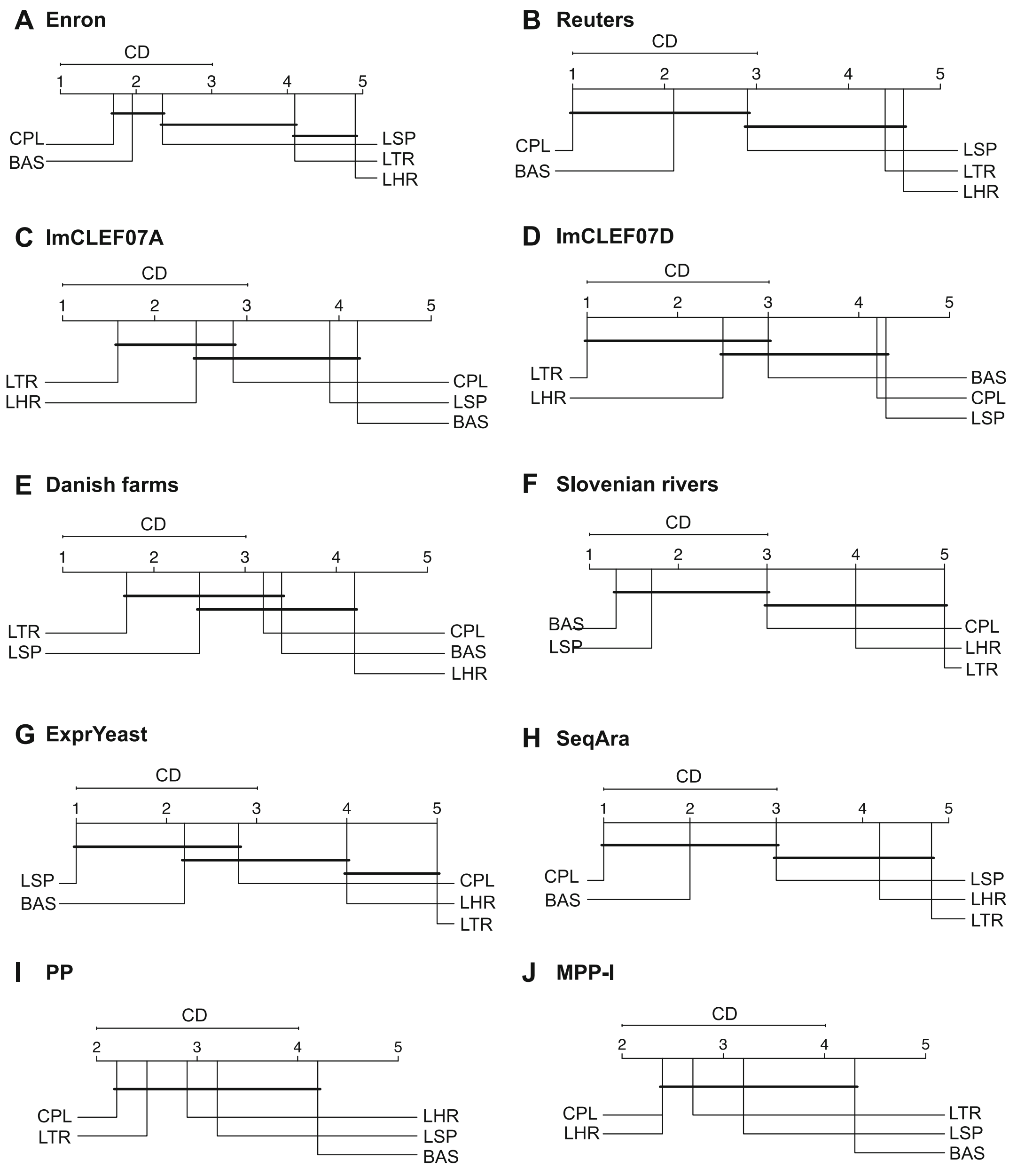

Fig. 4. Average ranks diagrams comparing predictive performance, measured as AUPRC$_{\mathrm{m}}$, of the five model induction algorithms for each of the ten HMC data sets. The numbers on the line represent ranks of the algorithms averaged over the ten cross-validation folds. Better performing algorithms are on the left-hand side. The algorithms with average ranks that differ by less than the critical distance for a p-value of 0.05 are connected with a line. CD = critical distance = 2.0092. Abbreviations: BAS = baseline, CPL = child vs. parent label, LSP = label specialization, LHR = labels without hierarchical relations, LTR = label vs. the rest.

Table 3. The relation between the size of a hierarchy and emergence of a single- or a multi-label classification algorithm as the best performing. Columns: h_n = hierarchy nodes; h_l = hierarchy leaves; $\bar{b}$ = average branching factor; $c\text{-}p_c$ = cardinality accounting for most specific annotations available to the complete decomposition algorithms; $c\text{-}p_h$ = cardinality accounting for most specific annotations available to the hierarchical algorithms; Alg. = algorithm (for the abbreviations of algorithm names, we refer to Fig. 4).

Data set	Hierarchy			Annotations		Best performing algorithm		
	h_n	h_l	$\bar{b}$	$c\text{-}p_c$	$c\text{-}p_h$	Alg.	Multi-label?	Hierarchical?
ImCLEF07D	**46**	26	2.19	1.00	1.00	LTR	**No**	No
Enron	**56**	52	11.20	2.87	3.37	CPL	**No**	Yes
Danish farms	**70**	35	1.94	6.74	7.08	LTR	**No**	No
ImCLEF07A	**96**	63	2.82	1.00	1.00	LTR	**No**	No
Reuters	**100**	79	4.55	1.46	3.13	CPL	**No**	Yes
SeqAra	**196**	148	4.00	1.30	3.32	CPL	**No**	Yes
ExprYeast	**417**	161	1.62	2.29	4.00	LSP	**Yes**	Yes
Slo. rivers	**724**	492	3.11	33.04	50.67	BAS	**Yes**	Yes

Table 4. The relation between the amount of additional annotations available to the hierarchical algorithms and the emergence of a hierarchical or a complete decomposition algorithm as the best performing. Columns: diff = $c\text{-}p_h$ - $c\text{-}p_c$; ia = percentage of incomplete annotations. For a description of the rest of the abbreviations, we refer to Table 3.

Data set	Annotations		Hierarchy			Best performing algorithm		
	diff	ia	h_n	h_l	$\bar{b}$	Alg.	Multi-label?	Hierarchical?
ImCLEF07A	**0**	0%	96	63	2.82	LTR	No	**No**
ImCLEF07D	**0**	0%	46	26	2.19	LTR	No	**No**
Danish farms	**0.34**	0%	70	35	1.94	LTR	No	**No**
Enron	**0.50**	1%	56	52	11.20	CPL	No	**Yes**
Reuters	**1.67**	18%	100	79	4.55	CPL	No	**Yes**
ExprYeast	**1.71**	1%	417	161	1.62	LSP	Yes	**Yes**
SeqAra	**2.02**	27%	196	148	4.00	CPL	No	**Yes**
Slo. rivers	**17.63**	26%	724	492	3.11	BAS	Yes	**Yes**

5 Conclusions and Discussion

We introduced the hierarchy decomposition pipeline, a publicly available software toolbox for comparison of model induction algorithms for hierarchical multi-label classification (HMC) problems in the ensemble setting. The pipeline contains five algorithms: the algorithm that constructs a global model, which predicts the complete hierarchy, two partial decomposition algorithms that construct local models, which predict different edges of a hierarchy, and two complete decomposition algorithms that construct one or multiple models, which predict subset(s) of hierarchy nodes. The pipeline also contains tools for performance-based comparison of the algorithms, which compute the area under the average precision-recall curve and perform a statistical test of the differences in performance.

We applied the pipeline on ten HMC data sets and draw the following conclusions:

First, by comparing the algorithms on a set of common labels, we cannot guarantee that they will be compared on a set of common annotations. The set of labels common to all algorithms is composed of those labels that are assigned to at least one example as the most specific annotation. While the complete decomposition algorithms use only the most specific annotations, the hierarchical algorithms may assign additional annotations for the common labels, simply by applying the hierarchy constraint. This issue can be addressed by comparing the algorithms on a set of common labels that are, at the same time, hierarchy leaves. However, we should have in mind that by making this choice we may omit many annotations. The middle ground is to perform both types of comparisons considering their advantages and disadvantages.

Second, there exists a need for the proposed pipeline, since there is no single best algorithm for all HMC problems.

Third, the properties of a HMC data set can be related to the type of best performing algorithm on that data set. Multi-label classification algorithms perform best on data sets with large hierarchies. Interestingly, high cardinality is not strongly related to the advantage of multi-label classification algorithms. Hierarchical algorithms perform best on data sets from which they can obtain additional annotations compared to the complete decomposition algorithms, simply by applying the hierarchy constraint. They also perform best on data sets with large and complex hierarchies.

The limitation of the analysis that relates the properties of a HMC data set to the type of best performing algorithm is the small number of data sets in the study. The limitation can be addressed by performing a simulation that: (1) generates hundreds of artificial HMC data sets with predefined properties; (2) applies the proposed pipeline on the data sets to determine the best performing algorithm; and (3) uses the collected data for meta learning to produce a classifier relating dataset properties to the type of best performing algorithm. The simulation should consider data sets with both a tree shaped and a DAG shaped label hierarchy. In this study, we had only two data sets with a DAG shaped label hierarchy and none of the algorithms performed significantly better than the rest on those two data sets. They were, consequently, left out of the analysis.

The pipeline can be improved in several directions. While it has been developed in the ensemble setting to maximize predictive performance on complex HMC data sets (e.g., functional genomics data sets), it can be adapted to construct single models. Furthermore, it can be modified to construct ensembles other than random forests, e.g., bagging or boosting. Finally, additional research need to be performed to understand whether it is possible to add stratified cross-validation as an option.

References

1. Vens, C., Struyf, J., Schietgat, L., Džeroski, S., Blockeel, H.: Decision trees for hierarchical multi-label classification. Mach. Learn. **73**, 185–214 (2008)
2. Džeroski, S., Demšar, D., Grbović, J.: Predicting chemical parameters of river water quality from bioindicator data. Appl. Intell. **13**(1), 7–17 (2000)
3. Clare, A.: Machine learning and data mining for yeast functional genomics. Ph.D. thesis, University of Wales Aberystwyth, Aberystwyth, UK (2003)
4. Klimt, B., Yang, Y.: The enron corpus: a new dataset for email classification research. In: Boulicaut, J.-F., Esposito, F., Giannotti, F., Pedreschi, D. (eds.) ECML 2004. LNCS (LNAI), vol. 3201, pp. 217–226. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-30115-8_22
5. Lewis, D.D., Yang, Y., Rose, T.G., Li, F.: RCV1: a new benchmark collection for text categorization research. J. Mach. Learn. Res. **5**, 361–397 (2004)
6. Demšar, D., et al.: Using multi-objective classification to model communities of soil. Ecol. Modell. **191**(1), 131–143 (2006)
7. Dimitrovski, I., Kocev, D., Loskovska, S., Džeroski, S.: Hierchical annotation of medical images. In: Proceedings of the 11th International Multiconference - Information Society, pp. 174–181. JSI, Ljubljana (2008)
8. Vidulin, V., Šmuc, T., Supek, F.: Extensive complementarity between gene function prediction methods. Bioinformatics **32**(23), 3645–3653 (2016)
9. Vidulin, V., Šmuc, T., Džeroski, S., Supek, F.: The evolutionary signal in metagenome phyletic profiles predicts many gene functions. Microbiome **6**(1), 129 (2018)
10. Madjarov, G., Vidulin, V., Dimitrovski, I., Kocev, D.: Web genre classification with methods for structured output prediction. Inf. Sci. **503**, 551–573 (2019)
11. Ruepp, A., et al.: The FunCat, a functional annotation scheme for systematic classification of proteins from whole genomes. Nucleic Acids Res. **32**(18), 5539–5545 (2004)
12. Ashburner, M., et al.: Gene ontology: tool for the unification of biology. Nat. Genet. **25**(1), 25 (2000)
13. Zhou, N., et al.: The CAFA challenge reports improved protein function prediction and new functional annotations for hundreds of genes through experimental screens. Genome Biol. **20**(1), 1–23 (2019)
14. Bakır, G.H., Hofmann, T., Schölkopf, B., Smola, A.J., Taskar, B., Vishwanathan, S.V.N. (eds.): Predicting Structured Data. The MIT Press, Cambridge (2007)
15. Silla, C., Freitas, A.: A survey of hierarchical classification across different application domains. Data Min. Knowl. Disc. **22**(1–2), 31–72 (2011)
16. Clare, A., King, R.D.: Predicting gene function in Saccharomyces cerevisiae. Bioinformatics **19**(S2), ii42–ii49 (2003)
17. Quinlan, J.R.: C4.5: Programs for Machine Learning. Morgan Kaufmann, San Francisco (1993)
18. Blockeel, H.: Top-down induction of first order logical decision trees. Ph.D. thesis, Katholieke Universiteit Leuven, Leuven, Belgium (1998)
19. Blockeel, H., Bruynooghe, M., Džeroski, S., Ramon, J., Struyf, J.: Hierarchical multi-classification. In: Proceedings of the ACM SIGKDD Workshop on Multi-Relational Data Mining, pp. 21–35 (2002)
20. Blockeel, H., Schietgat, L., Struyf, J., Džeroski, S., Clare, A.: Decision trees for hierarchical multilabel classification: a case study in functional genomics. In: Fürnkranz, J., Scheffer, T., Spiliopoulou, M. (eds.) PKDD 2006. LNCS (LNAI), vol. 4213, pp. 18–29. Springer, Heidelberg (2006). https://doi.org/10.1007/11871637_7

21. Obozinski, G., Lanckriet, G., Grant, C., Jordan, M.I., Noble, W.S.: Consistent probabilistic outputs for protein function prediction. Genome Biol. **9**(S1), S6+ (2008)
22. Barutcuoglu, Z., Schapire, R.E., Troyanskaya, O.G.: Hierarchical multi-label prediction of gene function. Bioinformatics **22**(7), 830–836 (2006)
23. Guan, Y., Myers, C.L., Hess, D.C., Barutcuoglu, Z., Caudy, A., Troyanskaya, O.: Predicting gene function in a hierarchical context with an ensemble of classifiers. Genome Biol. **9**(S1), S3+ (2008)
24. Valentini, G.: True path rule hierarchical ensembles for genome-wide gene function prediction. IEEE ACM Trans. Comput. Biol. **8**(3), 832–847 (2011)
25. Levatić, J., Kocev, D., Džeroski, S.: The importance of the label hierarchy in hierarchical multi-label classification. J. Intell. Inf. Syst. **45**(2), 247–271 (2014). https://doi.org/10.1007/s10844-014-0347-y
26. Demšar, J.: Statistical comparisons of classifiers over multiple data sets. J. Mach. Learn. Res. **7**, 1–30 (2006)