Chained Path Evaluation for Hierarchical Multi-label Classification

Mallinali Ramírez-Corona and **L. Enrique Sucar** and **Eduardo F. Morales**
Instituto Nacional de Astrofísica, Óptica y Electrónica
Luis Enrique Erro No. 1, Sta. Ma. Tonantzintla, Puebla, 72840, México
{mallinali.ramirez,esucar,emorales}@inaoep.mx

Abstract

In this paper we propose a novel hierarchical multi-label classification approach for tree and directed acyclic graph (DAG) hierarchies. The method predicts a single path (from the root to a leaf node) for tree hierarchies, and multiple paths for DAG hierarchies, by combining the predictions of every node in each possible path. In contrast with previous approaches, we evaluate all the paths, training local classifiers for each non-leaf node. The approach incorporates two contributions; (i) a cost is assigned to each node depending on the level it has in the hierarchy, giving more weight to correct predictions at the top levels; (ii) the relations between the nodes in the hierarchy are considered, by incorporating the parent label as in chained classifiers. The proposed approach was experimentally evaluated with 10 tree and 8 DAG hierarchical datasets in the domain of protein function prediction. It was contrasted with various state-of-the-art hierarchical classifiers using four common evaluation measures. The results show that our method is superior in almost all measures, and this difference is more significant in the case of DAG structures.

Introduction

The traditional classification task deals with problems where each example t is associated with a single label $y \in L$, where L is the set of classes. However, some classification problems are more complex and multiple labels are needed. For example, a news story can be classified as sports and entertainment at the same time; this is called multi-label classification. A multi-label dataset D is composed of n instances $(x_1, J_1), (x_2, J_2), ..., (x_n, J_n)$, where $J \subset L$. When the labels are ordered in a predefined structure, typically a tree or a DAG (Direct Acyclic Graph), the task is called Hierarchical Multi-label Classification (HMC). By taking into account the hierarchical organization of the classes, the classification performance can be boosted. In hierarchical classification, an example that belongs to certain class automatically belongs to all its superclasses (hierarchy constraint). Some major applications of HMC can be found in the fields of text categorization (Rousu et al. 2006), protein function prediction (Silla Jr. and Freitas 2009a), music genre classification (Silla Jr. and Freitas 2009b), phoneme classification (Dekel, Keshet, and Singer 2005), etc.

Copyright © 2014, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

There are two major trends to find a path in a hierarchy (Tsoumakas and Katakis 2007): train one global classifier or split the problem using various local classifiers. Methods based on local classifiers tend to produce inconsistent labels, while global classifiers become computationally too expensive as the number of classes increases.

We propose a novel HMC approach, Chained Path Evaluation (CPE), to predict single paths in tree and multiple paths in DAG hierarchies (paths from the root down to a leaf node). A local classifier per parent node is initially trained for each non-leaf node in the hierarchy. In the classification stage, the predictions of all the local classifiers are combined, using a logarithmic sum, to estimate the probability of all paths in the hierarchy. CPE incorporates two novel aspects: (i) a cost is assigned to each node depending on its level in the hierarchy, giving more weight to correct predictions at the top levels; (ii) the relations between the nodes in the hierarchy are considered, by incorporating the parent label as an additional attribute similarly to chained classifiers.

CPE was evaluated in ten tree structured hierarchies and in eight DAG structured hierarchies from protein function prediction. We compared our method against a number of state-of-the-art hierarchical classifiers and show that our method is competitive for tree-structured hierarchies, and clearly superior for DAG structures.

The document is organized as follows. Related Work reviews the relevant work in the area, Chained Path Evaluation describes the method in detail, Experimental Setup outlines the framework for the experiments, Results contrast our method against others and Conclusions and Future Work summarizes the paper and suggests possible future work.

Related Work

In hierarchical classification, there are basically two types of classifiers: global classifiers and local classifiers. Global classifiers construct a global model and train it to predict all the classes of an instance at once. Vens et al. (Vens et al. 2008) present a global method that applies a Predicting Clustering Tree (PCT) to hierarchical multi-label classification, transforms the problem in a hierarchy of clusters with reduced intra-cluster variance. One problem of global classifiers is that the computational complexity grows exponentially with the number of labels in the hierarchy.

Local classifiers can be trained in three different ways: a

Local Classifier per hierarchy Level (LCL), that trains one multi-class classifier for each level of the class hierarchy; training a Local binary Classifier per Node (LCN), where each classifier decides wether a node is predicted or not; the third way is training a Local Classifier per Parent Node (LCPN), where a multi-class classifier is trained to predict its child nodes.

Cerri et al. (Cerri, Barros, and de Carvalho 2013) propose a method that incrementally trains a multilayer perceptron for each level of the classification hierarchy (LCL). Predictions made by a neural network at a given level are used as inputs to the network of the next level. The labels are predicted using a threshold value. Finally, a post processing phase is used to correct inconsistencies (when a subclass is predicted but its superclass is not). This phase removes those predicted classes that do not have predicted superclasses. Some difficulties of this approach are the selection of a correct threshold and the need of a post-processing phase.

Alaydie et al. (Alaydie, Reddy, and Fotouhi 2012) developed HiBLADE (Hierarchical multi-label Boosting with LAbel DEpendency), an LCN algorithm that takes advantage of not only the predefined hierarchical structure of the labels, but also exploits the hidden correlation among the classes that is not shown through the hierarchy. This algorithm attaches the predictions of the parent nodes as well as the related classes. However, appending multiple attributes can create models that over-fit the data.

Silla and Freitas (Silla Jr. and Freitas 2009b) propose an LCPN algorithm combined with two selective methods for training. The first method selects the best features to train the classifiers, the second selects both the best classifier and the best subset of features simultaneously, showing that selecting a classifier and features improves the classification performance. A drawback of this approach is that the selection of the best features and the best classifier for each node can be a time-consuming process.

Bi and Kwok (Bi and Kwok 2011; 2012) propose HIROM, a method that uses the local predictions (independently of the way they are trained) to search for the optimal consistent multi-label classification using a greedy strategy. Using Bayesian decision theory, they derive the optimal prediction rule by minimizing the conditional risk. The limitations of this approach is that it optimizes a function that does not necessarily maximizes the performance in other measures.

The approach of Hernandez et al. (Hernandez, Sucar, and Morales 2013), used for tree structured taxonomies, learns an LCPN. In the classification phase, it classifies a new instance with the local classifier at each node, and combines the results of all of them to obtain a score for each path from the root to a leaf-node. Two fusion rules were used to achieve this: product rule and sum rule. Finally it returns the path with the highest score. One limitation of this method is that it favors shorter (product rule) or longer paths (sum rule) depending on which combination rule is used. Another limitation is that it does not take into account the relations between nodes when classifying an instance.

Extending the work of Hernandez et al., our method (Chained Path Evaluation or CPE), changes the way the classifiers are trained to include the relations between the labels, specifically of the parent nodes of the labels, to boost the prediction. The score for each path is computed using a fusion rule that takes into account the level in the hierarchy, thus minimizing the effect that the length of the path has in the score. We also extended the method to work with DAG structured hierarchies.

To include the relations of the parent nodes we used the idea of chain classifiers proposed by Read et al. (Read et al. 2011) and further extended by Zaragoza et al. (Zaragoza et al. 2011). The chain classifiers proposed by Read et al., link the classifiers along a chain where each classifier deals with the binary classification problem associated with a label. The feature space of each link in the chain is extended with the 0/1 label of all the previous classifiers in the chain. The order of the labels is random, and a set of chains are combined using an ensemble. Zaragoza et al. proposed a Bayesian Chain Classifier where they obtain a dependency structure out of the data. This structure determines the order of the chain, so that the order of the class variables in the chain is consistent with the structure found in the first stage. Each intermediate node is a naive Bayes classifier. We adapt this idea to a hierarchical classifier, such that the *chain* structure is determined by the hierarchy.

Chained Path Evaluation

Let D be a training set with N examples, $e_e = (x_e, J_e)$, where x_e is a d-dimensional feature vector and $J \subset L$, $L = \{l_1, l_2, ..., l_M\}$ a finite set of M possible labels. These labels are represented as $Y \in \{0,1\}^M$, where $y_i = 1$ iff $y_i \in J_i$ else $y_i = 0$. The parent of label y_i in the hierarchy is represented as $pa(y_i)$.

Our method exploits the correlation of the labels with its ancestors in the hierarchy and evaluates each possible path from the root to a leaf node, taking into account the level of the predicted labels to give a score to each path and finally return the one with the best score. The method is composed of two phases: training and classification.

Training

The method trains local classifiers per parent node (LCPN) (see Figure 1). A multi-class classifier C_i is trained for each non leaf node y_i. The training set for C_i, is composed of the instances where $y_i = 1$ and a subset of the instances in the siblings of y_i ($sib(y_i)$), the siblings include all the children nodes of the parents of y_i ($pa(y_i)$) except y_i. The instances that belong to $sib(y_i)$ are under-sampled to create a balanced training set. The number of under-sampled instances is proportional to the mean of the training examples for each child of y_i. The possible classes in C_i are the labels of the children of y_i plus an "unknown" label that corresponds to the examples in $sib(y_i)$.

As in multidimensional classification, the class of each node in the hierarchy is not independent from the other nodes. To incorporate these relations, inspired by chain classifiers, we include the class predicted by the parent node(s) as an additional attribute in the LCPN classifier. That is, the feature space of each node in the hierarchy is extended

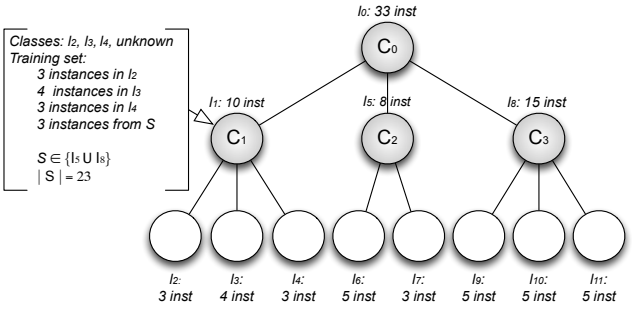

Figure 1: Example of a hierarchical structure. A classifier (C_i) is trained for each non-leaf node (grey) to predict its child labels (l_j).

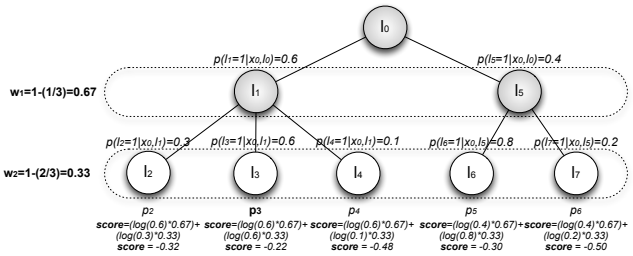

Figure 2: An example of the application of the merging rule is depicted. The left w_j represent the weights associated to each level of the hierarchy, each node has an associated probability $P(h_i|x_i, pa(h_i))$. Below the leaf nodes the score of the path is obtained using Equation (3).

with the 0/1 label association of the parent (tree structure) or parents (DAG structure) of the node, as in a Bayesian chain classifier.

Classification

The classification phase consists in calculating for each new instance with feature vector x_e, the probability of a node i to occur given the feature vector and the prediction of the parents of the current label $P(y_i = 1|x_e, pa(y_i))$. When the structure of the dataset is a DAG, it is possible that we obtain more than one prediction for one class, then the associated prediction is the average of all the predictions for that class. After computing a probability for each node, the predictions are merged using a rule to obtain a score for each path p_j, as explained below and shown in Figure 2.

Merging Rule. The rule that merges the predictions of each local classifier into one score considers the level in the hierarchy of the node to determine the weight that this node will have in the overall score. Misclassifications at the upper hierarchy levels (which correspond to more generic concepts) are more *expensive* than those at the lower levels (which correspond to more specific concepts). To achieve this task, the weight of a node ($w(y_i)$) is defined in Equation (2), where $level(y_i)$ is the level at which the node y_i is placed in the hierarchy (Equation (1)). For a tree structure it is simply the weight of its parent plus one, and for DAG structures it is computed as the mean of the levels of the m parents ($pa(y_i)$) of the node (y_i) plus one. Finally, $maxLevel$ is the length of the longest path in the hierarchy. This way of computing the weight of each node assures that the weights are well distributed along the hierarchy; so that the weights of the lower levels do not tend rapidly to zero, as in other approaches (Vens et al. 2008; Bi and Kwok 2011).

$$level(y_i) = 1 + \frac{1}{|pa(y_i)|} \sum_{j=1}^{m} level(pa(y_i)_j) \qquad (1)$$

$$w(y_i) = 1 - \frac{level(y_i)}{maxLevel + 1} \qquad (2)$$

Equation (3) describes the merging rule which is the sum of the logarithms of the probabilities on the nodes along the path, where n is the number of nodes in the path, h_i is the i^{th} node in the path and $P(h_i = 1|x_e, pa(h_i))$ is the probability of the node h_i to occur obtained by the local classifier. Taking the sum of logarithms is used to ensure numerical stability when computing the probability for long paths. This scheme assumes independence between the labels, although in an indirect way the dependencies with the parent nodes are considered by incorporating them as additional attributes. As in chain classifiers, this scheme looks for a balance between classification accuracy and computational complexity.

$$score = \sum_{i=0}^{n} w_{h_i} \times \log(P(h_i|x_i, pa(h_i))) \qquad (3)$$

For DAG structures there might be numerous paths from the root to one leaf node. In that case, all the paths that end in that leaf node are returned. Figure 2 shows a classification example.

Experimental Setup

The proposed method, Chained Path Evaluation (CPE), was evaluated experimentally with a number of tree and DAG structured hierarchies, using four different evaluation measures and compared with various state-of-the-art hierarchical classification techniques. Next we describe the experimental setup and then present the results.

For tree structured hierarchies, we used ten hierarchical datasets and compared CPE against three HMC methods:

1. Multidimensional Hierarchical Classifier (MHC). Proposed by Hernandez et al. (Hernandez 2012; Hernandez, Sucar, and Morales 2013).
2. Top-Down LCPN (TD). Training a LCPN and selecting at each level the most probable node. Only the children of this node are explored to preserve the consistency of the prediction.
3. HIROM. Proposed by Wei and Kwok (2011).

In the case of DAG structures, CPE was evaluated in eight hierarchical datasets and compared against two HMC methods:

1. Top-Down LCPN (TD). Training a LCPN and selecting at each level the most probable node. Only the children of this node are explored to preserve the consistency of the prediction.
2. HIROM. Proposed by Bi and Kwok (2011), the variant for DAG structures.

These methods were selected because they are all based on local classifiers and thus are the most clearly related with our method.

Datasets

Eighteen datasets were used in the tests (see Table 1), these datasets are from the field of functional genomics (http://dtai.cs.kuleuven.be/clus/hmcdatasets/). Ten of them (tree structured) are labeled using the FunCat annotation scheme (Ruepp et al. 2004). The remaining eight datasets (DAG structured) are labeled using the Gene Ontology vocabulary (Ashburner, Ball, and Blake 2000) to describe the roles of genes and gene products in any organism.

All the datasets were pruned to get instances with no more than one path in the hierarchy from the root to a leaf node. Only the leaf nodes with enough instances to train were considered, more than 70 for tree hierarchies and more than 50 for DAG hierarchies thus having a hierarchical tree like structure with up to four levels of increasing specificity and a DAG structure of maximum 11 levels.

Evaluation measures

Measures for conventional classification are not adequate for hierarchical multi-label problems, for that reason specific measures for HMC have been proposed. In our work we used four of the more common evaluation measures.

Table 1: Datasets used in the experiments. L=Labels, A=Attributes, I=Instances and D=Maximum depth.

Dataset	L	A	I	D
Tree hierarchies				
Cellcycle_FUN	36	77	2339	4
Church_FUN	36	29	2340	4
Derisi_FUN	37	65	2381	4
Eisen_FUN	25	81	1681	3
Expr_FUN	36	553	2356	4
Gasch1_FUN	36	175	2346	4
Gasch2_FUN	36	54	2356	4
Pheno_FUN	17	71	1162	3
Seq_FUN	39	480	2466	4
Spo_FUN	36	82	2302	4
DAG hierarchies				
Cellcycle_GO	53	78	1708	11
Church_GO	53	28	1711	11
Derisi_GO	54	64	1746	11
Expr_GO	53	552	1720	11
Gasch1_GO	53	174	1716	11
Gasch2_GO	53	53	1720	11
Seq_GO	52	479	1711	11
Spo_GO	53	81	1685	11

The notation in the formulas include $\hat{y}_i$ as the predicted labels and y_i for the real set of labels.

Accuracy. A multi-label measure of accuracy introduced in (Godbole and Sarawagi 2004) (see Equation (4)). This is the ratio of the size of the union and intersection of the predicted and actual label sets (represented by the logical AND and OR operations in bit-vector notation, respectively), taken for each example, and averaged over the number of examples.

$$Accuracy = \frac{1}{N}\sum_{i=1}^{N}\frac{|y_i \wedge \hat{y}_i|}{|y_i \vee \hat{y}_i|} \tag{4}$$

Hamming Loss / Hamming Accuracy. Depicted in Equation (5), where $y_i \oplus \hat{y}_i$ is the symmetrical difference between y_i and $\hat{y}_i$ (the logical XOR operation). It represents a label-based accuracy.

$$Hloss = \frac{1}{NL}\sum_{i=1}^{N}|y_i \oplus \hat{y}_i| \tag{5}$$

Hamming accuracy is defined as $Haccuracy = 1 - Hloss$.

Exact Match. Represents the proportion of the real labels that were predicted.

$$ExactMatch = \frac{1}{N}\sum_{i=1}^{N}1_{y_i=\hat{y}_i} \tag{6}$$

F1-measure. The F1-measure, commonly used in information retrieval, has also been popular for multi-label classifications. For any vector of label associations $y \in \{0,1\}^T$, a label is relevant if $y_i = 1$ and predicted if $\hat{y}_i = 1$ (in a corresponding vector of predicted label associations), and from this we can define: *precision* as the fraction of predicted relevances which are actually relevant $\frac{|y \wedge \hat{y}|}{|\hat{y}|}$; and *recall* as the fraction of actual relevances which are also predicted $\frac{|y \wedge \hat{y}|}{|y|}$. F1-measure (F1) is calculated as in Equation (7).

$$F1 = \frac{2 \times precision \times recall}{precision + recall} \tag{7}$$

When it is averaged by the number of examples, like in this case, it is called F1-macro D.

$$F1macro^{\times D}(D) = \frac{1}{N}\sum_{j=0}^{N}F1(y_j, \hat{y}_j) \tag{8}$$

Results

The results were obtained by a stratified 10-fold cross-validation. The four evaluation measures are reported in different tables for each dataset. We used Naive Bayes as base classifier for all the methods. The best results are marked in bold. A paired two-tailed t-test was carried out to find statistical significance in the results with a confidence degree of 95%. Statistically inferior results against CPE are marked with $\downarrow$ and statistically superior results are marked with $\uparrow$.

Table 2: Results obtained using the four different evaluation measures for ten tree structured hierarchies. Statistically inferior results against CPE are marked with ↓ and statistically superior results are marked with ↑.

Accuracy				
Dataset	CPE	MHC	TD	HIROM
Cellcycle_FUN	**20.80**	19.68	19.58	9.33↓
Church_FUN	**13.23**	12.98	13.00	4.01↓
Derisi_FUN	**13.85**	13.44	13.40	4.17↓
Eisen_FUN	**28.62**	28.57	27.82	17.06↓
Expr_FUN	**23.49**	23.28	23.08	17.90↓
Gasch1_FUN	**20.05**	19.83	19.61	13.28↓
Gasch2_FUN	20.03	**20.28**	20.16	8.38↓
Pheno_FUN	**23.25**	22.04	22.65	13.30↓
Seq_FUN	31.75	31.73	**31.86**	21.32↓
Spo_FUN	**15.32**	15.20	14.79	6.86↓

Hamming Accuracy				
Dataset	CPE	MHC	TD	HIROM
Cellcycle_FUN	88.87	89.01	88.65	**91.22**↑
Church_FUN	86.00	86.04	85.94	**90.61**↑
Derisi_FUN	87.43	87.59	87.20	**90.87**↑
Eisen_FUN	87.18	87.30	87.07	**89.48**↑
Expr_FUN	88.80	88.82	88.69	**91.67**↑
Gasch1_FUN	88.74	88.90	88.64	**91.55**↑
Gasch2_FUN	88.65	89.00↑	88.59	**91.12**↑
Pheno_FUN	83.85	83.47	83.56	**85.53**↑
Seq_FUN	91.32	91.36	91.33	**91.95**↑
Spo_FUN	87.95	88.30	87.90	**90.95**↑

Exact Match				
Dataset	CPE	MHC	TD	HIROM
Cellcycle_FUN	**17.66**	16.42	16.33	3.81↓
Church_FUN	**11.37**	10.98	10.98	2.91↓
Derisi_FUN	**11.17**	10.53	10.67	3.11↓
Eisen_FUN	24.80	**24.87**	23.85	4.52↓
Expr_FUN	**20.25**	20.08	20.07	9.29↓
Gasch1_FUN	**17.18**	16.92	16.88	6.91↓
Gasch2_FUN	**16.43**	16.38	16.26	4.16↓
Pheno_FUN	**12.12**	10.93	11.01	2.15↓
Seq_FUN	27.25	27.25	**27.41**	13.26↓
Spo_FUN	**12.51**	12.42	12.12	3.52↓

F1-macro D				
Dataset	CPE	MHC	TD	HIROM
Cellcycle_FUN	**22.41**	21.35	21.29	11.42↓
Church_FUN	**14.29**	14.20	14.15	4.49↓
Derisi_FUN	**15.40**	15.03	14.99	4.70↓
Eisen_FUN	**30.62**	30.52	29.92	21.64↓
Expr_FUN	**25.14**	24.92	24.64	21.25↓
Gasch1_FUN	**21.52**	21.31	21.02	15.80↓
Gasch2_FUN	21.89	**22.19**	22.11	10.06↓
Pheno_FUN	**28.82**	27.61	28.49	17.09↓
Seq_FUN	34.02	33.99	**34.12**	24.43↓
Spo_FUN	**16.72**	16.53	16.12	8.23↓

Tree structured hierarchies

The results for tree-structured hierarchies are summarized in Table 2. CPE outperforms the other ones in most datasets in the measures of accuracy, exact-match and F1-macroD. It is inferior in hamming accuracy to the HIROM algorithm; however, HIROM obtains such good results due to the fact that this method optimizes a function which can be reduced to the hamming loss (Bi and Kwok 2012). CPE, MHC and TD achieve, in general, a balance between precision and recall. HIROM tends to sacrifice precision for recall or vice versa.

For applications in which obtaining an exact match in all the classes is important, our method seems to be the best option as in this measure (exact match) it obtains the best results. It also obtains the best results for the F1 measure that provides a balance between precision and recall.

In the case of training and classification time, CPE, MHC and TD spend approximately the same amount, on the order of 1 second for training and 0.1 milliseconds for classifying an instance. HIROM spends twice as much time as the other approaches for both training and classification.

In summary, our method is competitive with respect to other state-of-the-art methods. It shows a slightly superior performance in three of the four measures, and a difference with respect to HIROM. Our method is also competitive in terms of running time.

DAG structured hierarchies

The results for DAG-structured hierarchies are summarized in Table 3. In this case our method is superior in all cases to the other two approaches, and the difference is statistically significant in practically all the measures and datasets.

There are different factors that could explain this difference for DAG structures. One could be that our method was developed considering this type of hierarchies, while TD was not. However, HIROM also considers DAG structures, and our method is superior even for the Hamming accuracy, the measure that HIROM optimizes. Other possible explanation is that our approach is better for deeper hierarchies, as the tree datasets have as maximum depth of 3/4, while the DAG datasets have a maximum depth of 11. The weighting scheme per level in our method could be one of the reasons for this difference. A more thorough investigation of these issues will be a topic of future research.

Conclusions and Future Work

We presented a novel approach for hierarchical multi-label classification for tree and DAG structures. The method estimates the probability of each path by combing LCPNs, incorporating two additional features: (i) a weighting scheme that gives more importance to correct predictions at the top levels; (ii) an extension of the chain idea for hierarchical classification, incorporating the label of the parent nodes as additional attributes. Experiments with 18 tree and DAG hierarchies show that the proposed method is competitive with other techniques for tree structures, and superior for DAGs.

As future work we plan to extend the proposed method for non-mandatory leaf node prediction.

Table 3: Results obtained using different evaluation measures for DAG structured hierarchies. Statistically inferior results against CPE are marked with ↓ and statistically superior results are marked with ↑.

Accuracy			
Dataset	CPE	TD	HIROM
Cellcycle_GO	**35.12**	33.38↓	16.55↓
Church_GO	**26.35**	26.01	10.58↓
Derisi_GO	**27.18**	25.54↓	10.69↓
Expr_GO	**36.45**	31.89↓	12.78↓
Gasch1_GO	**34.52**	31.66↓	16.28↓
Gasch2_GO	**33.83**	32.16↓	17.34↓
Seq_GO	**46.30**	41.89↓	14.17↓
Spo_GO	**27.80**	26.13↓	16.09↓

Hamming Accuracy			
Dataset	CPE	TD	HIROM
Cellcycle_GO	**91.56**	90.37↓	86.68↓
Church_GO	**90.58**	90.29↓	31.77↓
Derisi_GO	**90.68**	90.19↓	31.54↓
Expr_GO	**91.65**	90.56↓	51.23↓
Gasch1_GO	**91.49**	90.76↓	70.84↓
Gasch2_GO	**91.43**	90.56↓	85.13↓
Seq_GO	**92.70**	91.79↓	77.67↓
Spo_GO	**90.72**	89.64↓	64.81↓

Exact Match			
Dataset	CPE	TD	HIROM
Cellcycle_GO	**20.25**	14.64↓	0.17↓
Church_GO	**11.69**	11.40	0.06↓
Derisi_GO	**12.26**	9.80↓	0.06↓
Expr_GO	**22.03**	15.52↓	0.35↓
Gasch1_GO	**19.99**	15.79↓	0.52↓
Gasch2_GO	**18.89**	14.71↓	0.81↓
Seq_GO	**31.32**	22.62↓	0.00↓
Spo_GO	**12.58**	9.85↓	1.01↓

F1-Macro D			
Dataset	CPE	TD	HIROM
Cellcycle_GO	**44.83**	43.73↓	27.23↓
Church_GO	**36.63**	36.22↓	18.51↓
Derisi_GO	**37.47**	36.01↓	18.65↓
Expr_GO	**45.88**	41.98↓	21.63↓
Gasch1_GO	**44.16**	41.71↓	26.68↓
Gasch2_GO	**43.68**	42.54↓	28.19↓
Seq_GO	**55.17**	51.96↓	24.06↓
Spo_GO	**38.18**	36.48↓	26.02↓

References

Alaydie, N.; Reddy, C. K.; and Fotouhi, F. 2012. Exploiting Label Dependency for Hierarchical Multi-label Classification. *Advances in Knowledge Discovery and Data Mining* 7301:294–305.

Ashburner, M.; Ball, C. A.; and Blake, J. A. 2000. Gene Ontology: tool for the unification of biology. *Nature Genetics* 25:25–29.

Bi, W., and Kwok, J. T. 2011. Multi-label classification on tree-and dag-structured hierarchies. *Proc. of the 28th Inter. Conf. on Machine Learning*.

Bi, W., and Kwok, J. T. 2012. Hierarchical Multilabel Classification with Minimum Bayes Risk. *2012 IEEE 12th Inter. Conf. on Data Mining* 101–110.

Cerri, R.; Barros, R. C.; and de Carvalho, A. C. 2013. Hierarchical multi-label classification using local neural networks. *Journal of Computer and System Sciences* 1:1–18.

Dekel, O.; Keshet, J.; and Singer, Y. 2005. An online algorithm for hierarchical phoneme classification. *Machine Learning for Multimodal Interaction* 146–158.

Godbole, S., and Sarawagi, S. 2004. Discriminative methods for multi-labeled classification. In *Advances in Knowledge Discovery and Data Mining*. Springer. 22–30.

Hernandez, J.; Sucar, L. E.; and Morales, E. F. 2013. A Hybrid Global-Local Approach for Hierarchical Classification. *Twenty-Sixth Inter. Florida Artificial Intelligence Research Society Conference*.

Hernandez, J. 2012. Multidimensional Hierarchical Classification. Master's thesis, Instituto Nacional de Astrofísica, Óptica y Electrónica.

Read, J.; Pfahringer, B.; Holmes, G.; and Frank, E. 2011. Classifier chains for multi-label classification. *Machine Learning* 254–269.

Rousu, J.; Saunders, C.; Szedmak, S.; and Shawe-Taylor, J. 2006. Kernel-based learning of hierarchical multilabel classification models. *Journal of Machine Learning Research* 7:1601–1626.

Ruepp, A.; Zollner, A.; Maier, D.; Albermann, K.; Hani, J.; Mokrejs, M.; Tetko, I.; Güldener, U.; Mannhaupt, G.; Münsterkötter, M.; and Mewes, H. W. 2004. The FunCat, a functional annotation scheme for systematic classification of proteins from whole genomes. *Nucleic acids research* 32(18):5539–45.

Silla Jr., C. N., and Freitas, A. A. 2009a. A Global-Model Naive Bayes Approach to the Hierarchical Prediction of Protein Functions. *IEEE Inter. Conf. on Data Mining* 992–997.

Silla Jr., C. N., and Freitas, A. A. 2009b. Novel top-down approaches for hierarchical classification and their application to automatic music genre classification. In *Systems, Man and Cybernetics, 2009. SMC 2009. IEEE International Conference on*, 3499–3504.

Tsoumakas, G., and Katakis, I. 2007. Multi-Label Classification: An Overview. *Inter. Journal of Data Warehousing and Mining* 3(September):1–13.

Vens, C.; Struyf, J.; Schietgat, L.; Džeroski, S.; and Blockeel, H. 2008. Decision trees for hierarchical multi-label classification. *Machine Learning* 73(2):185–214.

Zaragoza, J. H.; Sucar, L. E.; Morales, E. F.; Bielza, C.; and Larra, P. 2011. Bayesian Chain Classifiers for Multidimensional Classification. *Computational Intelligence* 2192–2197.