Bioinformatics
doi.10.1093/bioinformatics/xxxxxx
Advance Access Publication Date: Day Month Year
Manuscript Category

OXFORD

Subject Section

HEMDAG: a family of modular and scalable hierarchical ensemble methods to improve Gene Ontology term prediction

Marco Notaro [1], Marco Frasca [1], Alessandro Petrini [1], Jessica Gliozzo [1], Elena Casiraghi [1], Peter N. Robinson [2] and Giorgio Valentini [1,3,4]*

[1]AnacletoLab - Dipartimento di Informatica, Università degli Studi di Milano, Via Celoria 18, 20133 Milano, Italy
[2]The Jackson Laboratory for Genomic Medicine: 06032 Farmington, CT, US
[3]CINI, National Laboratory in Artificial Intelligence and Intelligent Systems—AIIS, Roma, Italy
[4] Data Science Research Center, Università degli Studi di Milano, 20133 Milano, Italy

*To whom correspondence should be addressed.

Associate Editor: XXXXXXX

Received on XXXXX; revised on XXXXX; accepted on XXXXX

Abstract

Motivation: Automated protein function prediction is a complex multi-class, multi-label, structured classification problem in which protein functions are organized in a controlled vocabulary, according to the Gene Ontology (GO). "Hierarchy-unaware" classifiers, also known as "flat" methods, predict GO terms without exploiting the inherent structure of the ontology, potentially violating the *True-Path-Rule* (TPR) that governs the GO, while "hierarchy-aware" approaches, even if they obey the TPR, do not always show clear improvements with respect to flat methods, or do not scale well when applied to the full GO.
Results: To overcome these limitations, we propose Hierarchical Ensemble Methods for Directed Acyclic Graphs (HEMDAG), a family of highly modular hierarchical ensembles of classifiers, able to build upon any flat method and to provide "TPR-safe" predictions, by leveraging a combination of isotonic regression and TPR learning strategies. Extensive experiments on synthetic and real data across several organisms firstly show that HEMDAG can be used as a general tool to improve the predictions of flat classifiers, and secondly that HEMDAG is competitive versus state-of-the-art hierarchy-aware learning methods proposed in the last CAFA international challenges.
Availability: Fully-tested R code freely available at `https://anaconda.org/bioconda/r-hemdag`. Tutorial and documentation at `https://hemdag.readthedocs.io`
Contact: marco.notaro@unimi.it
Supplementary information: Supplementary data are available at *Bioinformatics* online.

1 Introduction

The Automated Protein Function Prediction (AFP) is a central and challenging problem in Computational Biology (Zhou *et al.*, 2019). AFP can be modeled as a set of related binary classification or ranking tasks where each protein may be associated with multiple functional classes structured according to a predefined hierarchy, i.e. the Gene Ontology (Gene Ontology (GO) Consortium, 2018). Despite GO terms are interconnected according to a predefined hierarchy (i.e. a directed acyclic graph), most computational approaches proposed in literature, ranging from sequence-based methods (Juncker *et al.*, 2009; Törönen *et al.*, 2018) to network-based methods (Sharan *et al.*, 2007; Re *et al.*, 2012; Frasca *et al.*, 2020), learn functional terms independently each of the other, without considering the information coded in the hierarchical constraints, typically yielding to sub-optimal solutions. The GO is governed by the *True-Path-Rule* (also known as *annotation propagation rule*), which states that if a gene product is associated with a given functional term, it must be associated with all its parent terms and recursively with its ancestor terms. On the contrary, if a gene product is not associated with an ontology term, it cannot be associated with any of its offspring terms.

Theoretical studies (Armano, 2015) as well as applications in several domains (Silla and Freitas, 2011), showed the effectiveness of hierarchical approaches (Valentini, 2011; Cesa-Bianchi *et al.*, 2012; Kocev *et al.*, 2013). Furthermore, the results of three international challenges for the Critical Assessment of Functional Annotation, (CAFA (Radivojac *et al.*, 2013), CAFA2 (Jiang *et al.*, 2016) and CAFA3 (Zhou *et al.*, 2019)) emphasized the value of "hierarchy-aware" methods for GO term prediction. Indeed, in the last decade several structured output approaches have been proposed in literature, including methods based on genetic algorithms (Cerri *et al.*, 2019), multi-label learning (Wu *et al.*, 2014), neural (Cerri *et al.*, 2015; Kulmanov *et al.*, 2017) or deep neural networks (Kulmanov and Hoehndorf, 2019), active learning (Nakano *et al.*, 2020), and learning to rank framework (Liu *et al.*, 2020).

Structured output prediction methods can be schematically classified in two broad categories. The first one exploits joint input and output kernelization techniques, based on large margin methods for structured and interdependent output variables (Lampert and Blaschko, 2009; Sokolov and Ben-Hur, 2010; Kahanda *et al.*, 2015). The second one instead is based on ensembles of learning machines able to exploit the hierarchical relationship among classes (Guan *et al.*, 2008; Yu *et al.*, 2015; Cerri *et al.*, 2016; Wang *et al.*, 2018). For a broader overview on hierarchical classification methods we refer the reader to Valentini (2014) and Kulmanov *et al.* (2020). Although these two categories of structured output approaches have been successfully applied to bio-ontologies, hierarchical ensemble methods (HEMs), due to their high-modularity, scale better on large ontologies and dataset than kernel-based structured output approaches. Nevertheless, several studies showed that it is often hard to improve flat predictions using hierarchical ensemble methods (Obozinski *et al.*, 2008; Wang *et al.*, 2018).

To overcome these limitations we propose novel hierarchical ensemble methods that can improve flat learning method by correcting and making their flat predictions consistent, and that are competitive with state-of-the-art structured output methods. More precisely, the main contributions of this work are the following: a) novel hierarchical ensemble methods (ISO-TPR) based on the integration of the True-Path-Rule algorithm and isotonic regression, to find the "TPR-safe" closest solution (in a least square sense) to the flat predictions, assuring at the same time an improved sensitivity; b) a family of Hierarchical Ensemble Methods for Directed Acyclic Graphs (HEMDAG) that integrates ISO-TPR with our previously proposed approaches for the hierarchical prediction of bio-ontologies (Notaro *et al.*, 2017, 2019); c) extensive full GO ontology experiments on different organisms showing that HEMDAG can be used to systematically improve flat predictions; d) a systematic experimental comparison of the behavior of all the 20 HEMDAG methods on different artificial data sets scenarios, so as to estimate the probability they improve flat performance in different settings; e) time-lapse experiments to compare HEMDAG with state-of-the-art (SOTA) hierarchy-aware structured output methods; f) a fully-tested software library implementing HEMDAG methods, available at both `Bioconda` and `CRAN` repositories, alongside a comprehensive step-by-step tutorial showing how to use HEMDAG to predict GO terms (`https://hemdag.readthedocs.io`).

2 Methods

The highly modular framework of ISO-TPR is based on a four-step learning strategy (Fig. 1). In the first step, a learning algorithm (LA, represented by ellipses in Fig. 1a) is applied to train a set of base classifiers (circles) associated with a specific GO term (denoted by $C1, \ldots, Cn$), using data sets $D1, \ldots, Dn$. In the second step, the base classifiers are hierarchically combined according to the GO topology (Fig. 1b). Then, for each protein, the "positive" predictions of the base learners are recursively propagated from the leaf nodes toward the root (Fig. 1c). Finally the isotonic regression algorithm is applied to assure that the predicted GO terms are consistent, i.e. obey the True-Path-Rule and are close to the predictions provided in the bottom-up step (Fig. 1d). More precisely ISO-TPR integrates an approximate version of the isotonic regression algorithm, i.e. the Generalized Pool-Adjacent-Violators (GPAV), having low computational complexity and high accuracy (Burdakov *et al.*, 2006), with the True-Path-Rule learning strategy (Valentini, 2011; Notaro *et al.*, 2017), to improve the overall sensitivity of the ensemble and to assure at the same time consistent predictions, i.e. predictions that obey the True-Path-Rule.

2.1 Basic notations and definitions

Let $G =< V, E >$ be a DAG with vertices $V = \{1, 2, \ldots, |V|\}$ and edges $e = (i, j) \in E \subset V \times V$. In our setting, nodes V represent the terms of the ontology and a directed edge $(i, j) \in E$ the hierarchical relationship between i and j: i is the parent term and j is the child term. The set of children of node i is denoted by $child(i)$, the set of its parents by $par(i)$, the set of its ancestors by $anc(i)$ and the set of its descendants by $desc(i)$. $\phi(i)$ and $\Delta(i)$ represent GO sets of terms annotated by the ensemble, that are respectively children or descendants of a node/GO term i. Given the gene product input space X and the multi-labeling space $\mathbb{Y} = [0, 1]^{|V|}$, a "flat multi-label scoring" predictor $f : X \to \mathbb{Y}$ provides a score vector $f(x) = \hat{\boldsymbol{y}}^{(x)} =< \hat{y}_1^{(x)}, \hat{y}_2^{(x)}, \ldots, \hat{y}_{|V|}^{(x)} >$ for a given protein $x \in X$, such that $\hat{y}_i^{(x)} \in [0, 1]$ represents the likelihood that x belongs to the term $i \in V$. To simplify the notation, when clear from the context, $\hat{\boldsymbol{y}}^{(x)}$ is simply denoted by $\hat{\boldsymbol{y}}$. We say that the multi-label scoring $\boldsymbol{y}$ is *consistent* if it obeys the *True-Path-Rule*:

$$\boldsymbol{y} \text{ is consistent } \iff \forall i, j \in V, i \in par(j) \Rightarrow y_i \geq y_j \ . \quad (1)$$

As a consequence of equation (1), in order to have consistent predictions, if a protein is predicted to belong to a given term i then the same protein must be predicted to belong to all the ancestor terms of i. HEMDAG algorithms compute a function $g(x) : \mathbb{Y} \to \mathbb{Y}$ which transforms a flat multi-label prediction $\hat{\boldsymbol{y}}$ into a consistent one $g(\hat{\boldsymbol{y}})$. Supplementary Fig. S1 shows an example of inconsistent flat scores and their HEMDAG correction for the mouse protein *C-C chemokine receptor type* 6 (`ENSMUSP00000095029`), whose high expression levels are associated with colon cancer metastasis (Kapur *et al.*, 2016).

2.2 Flat learning of the ontology terms

The first step of ISO-TPR (Fig. 1a) consists in independently learning a flat predictor $f_i : X \to [0, 1]$ for every term $i \in V$. The output of this step is thereby a flat multi-label predictor $f : X \to \mathbb{Y}$, such that $f(x) = \hat{\boldsymbol{y}} =< \hat{y}_1, \hat{y}_2, \ldots, \hat{y}_{|V|} >$, with $f_i(x) = \hat{y}_i$ for each $i \in \{1, \ldots, |V|\}$. f_i can be any supervised or semi-supervised base predictor able to provide probability scores or binary predictions. A real score $\hat{y}_i \in [0, 1]$ is interpreted as the likelihood that a gene product is annotated with the GO term i, whereas a binary prediction $\hat{y}_i \in \{0, 1\}$ as an association with the term i. The flat predictors $f_1, f_2, \ldots, f_{|V|}$ are finally hierarchically organized according to the GO ontology (Fig. 1b).

2.3 The ISOtonic True-Path-Rule algorithm and the HEMDAG family of hierarchical ensemble algorithms

In the third and fourth steps of the algorithm (Fig. 1c and d), ISO-TPR integrates the TPR-DAG (Notaro *et al.*, 2017) and GPAV (Burdakov *et al.*, 2006) algorithms to find a solution that: a) obey the True-Path-Rule; b) is the closest to the bottom-up modified flat predictions in a least square sense; c) improves the overall sensitivity of the hierarchical ensemble.

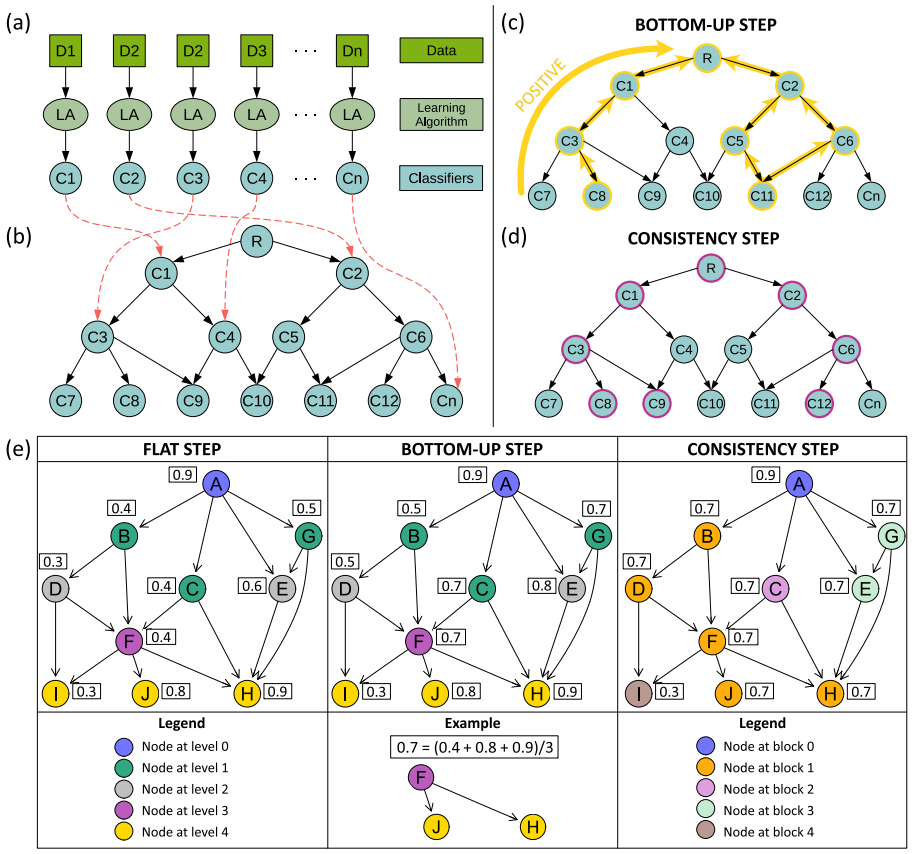

Fig. 1. High level picture of ISO-TPR methods (a-d) and a representation of the ISO-TPR computation on a specific toy example (e). (a) Training of the base classifier: each classifier is trained using a specific learning algorithm on each term of the GO; (b) Hierarchical combination of the base classifiers: base classifiers are hierarchically organized according to the GO topology; (c) Bottom-up step: only the nodes considered to be "positive" are bottom-up propagated (circles with yellow rim); bottom-up yellow arrows represent positive predictions up-propagated and combined with those of their parents. This step boosts the sensitivity of the predictions, but it does not guarantee that they are consistent with the hierarchy. (d) Consistency step: it provides "TPR-safe" predictions. Circles with purple rim represent nodes whose predictions are corrected according to the hierarchy. (e) Computation of ISO-TPR on a specific example. "FLAT STEP" panel: nodes and their scores represent respectively GO terms and their flat predictions. The different colors represent the levels, defined according to the maximum distance from the root node. "BOTTOM-UP STEP" panel: flat scores are modified by recursively using only "positive" predictions from the leaves to the root through a per-level visit of the DAG. The "Example" box shows a close-up view of how the correction of node *F* is performed, i.e. by averaging the flat score of node *F* with those of its "positive" children (nodes *J* and *H*) – selected through the threshold-free strategy (see Section 2.3.1). "CONSISTENCY STEP" panel: in the last step the GPAV algorithm is applied to guarantee that the ensemble predictions satisfy the True-Path-Rule. The graph nodes in this panel are colored according to the "absorption" operation of the GPAV algorithm (Burdakov et al., 2006).

2.3.1 The True-Path-Rule algorithm

In the bottom-up step of the algorithm, TPR-DAG propagates "positive" predictions from the leaves toward the root of the GO-DAG, by recursively applying the following rule according to a "per-level" visit of the DAG starting from the deepest nodes:

$$\bar{y}_i := \frac{1}{1+|\phi_i|}(\hat{y}_i + \sum_{j \in \phi_i} \bar{y}_j) \tag{2}$$

where ϕ_i are "positive" children of i, i.e. the children nodes annotated as positive by the ensemble, $\hat{y}_i$ is the flat prediction, and $\bar{y}_i$ the updated prediction of the parent node. This step guarantees an improvement of the sensitivity of the ensemble (Notaro *et al.*, 2017). According to the definition of positive children, different variants of ISO-TPR are available:

Threshold Free (TF): positive children are those that can increment the parent score, thus $\phi_i := \{j \in child(i) | \bar{y}_j > \hat{y}_i\}$. Indeed, since the bottom-up step is recursive, children scores are already "consensus" of predictions coming from lower levels.

Fixed Threshold (FT): a unique threshold $\bar{t}$ is a-priori set for all nodes to determine the set of positives $\phi_i := \{j \in child(i) | \bar{y}_j > \bar{t}\}$, for every $i \in V$. For instance, if the flat predictions are probabilities, it could be meaningful to a-priori set $\bar{t} = 0.5$;

Learned Threshold (LT): a threshold is selected to maximize some performance metric $\mathcal{M}$ estimated on the training data, e.g. the F-measure or the Area Under the Precision-Recall curve (AUPRC). Namely, the threshold is selected to maximize the measure $\mathcal{M}(j,t)$ on the training data for the term j with respect to the threshold t, thus resulting in $\phi_i := \{j \in child(i) | \bar{y}_j > t^*_j, t^*_j = \arg\max_t \mathcal{M}(j,t)\}$. Internal cross-validation is used to select t^*_j within a set of possible thresholds $t \in (0,1)$.

As an example, in Fig. 1e we show a pictorial example of the action mode of the threshold-free variant of the ISO-TPR algorithm. We also propose a weighted version algorithm (ISOtprW), where a weight $w \in [0,1]$ is introduced to balance the contribution of the parent node i and its "positive" children:

$$\bar{y}_i := w\hat{y}_i + \frac{(1-w)}{|\phi_i|} \sum_{j \in \phi_i} \bar{y}_j \tag{3}$$

If $w = 1$, no weight is attributed to the children and the ISO-TPR reduces to the GPAV algorithm. If $w = 0$, only the predictors associated to the positive children nodes "vote" to predict node i.

2.3.2 The Generalized Pool Adjacent Violators (GPAV) algorithm

GPAV is based on the Isotonic Regression (IR) algorithm proposed by Barlow and Brunk (1972). The IR problem involves finding a weighted least-squares fit $\bar{\boldsymbol{y}} \in \mathbb{R}^n$ to a vector $\hat{\boldsymbol{y}} \in \mathbb{R}^n$, with weight vector $\boldsymbol{w} \in \mathbb{R}^n$, subject to a set of given constraints $\bar{y}_i \geq \bar{y}_j \; \forall (i,j) \in E$. Such constraints define a partial or a total order and in our context are encoded by the DAG $G(V, E)$. In particular, any pair $(i, j) \in E$ represents the constraint $\bar{y}_i \geq \bar{y}_j$. Each node is associated with an observed value and each edge is associated with a monotonicity relationship. Formally, given a vector of observed values $\hat{\boldsymbol{y}} \in \mathbb{R}^n$, a strictly positive vector of weights $\boldsymbol{w} \in \mathbb{R}^n$ and a DAG $G(V, E)$, GPAV finds the vector of fitted values $\bar{\boldsymbol{y}} \in \mathbb{R}^n$ that solves the following convex quadratic program:

$$\begin{aligned} &\min_{\bar{y}} \sum_{i \in V} w_i (\bar{y}_i - \hat{y}_i)^2 \\ &s.t. \quad \bar{y}_i \geq \bar{y}_j \quad \forall (i,j) \in E \end{aligned} \tag{4}$$

In this sense the vector $\bar{\boldsymbol{y}}$ is the "constraints-satisfying" solution closest to $\hat{\boldsymbol{y}}$ in a least square sense. It is easy to solve the problem (4) when the constraints are in the simple form: $\bar{y}_1 \leq \bar{y}_2 \leq \ldots \bar{y}_{|V|}$, i.e., when the associated graph $G(V, E)$ has a linear structure. For this special case of complete order, the most efficient and widely used algorithm is the Pool-Adjacent-Violators algorithm (*PAV*) (Ayer *et al.*, 1955; Barlow, 1972), whose computational complexity is $\mathcal{O}(|V|)$ (Grotzinger and Witzgall, 1984). On the contrary, the optimization-based algorithms to solve the general IR problem (4) can be used only when the number of observations is quite small (up to few hundred), because they are characterized by a high computational complexity $\mathcal{O}(|V|^4)$ (Maxwell and Muckstadt, 1985). Nevertheless, Burdakov and coworkers proposed an approximate algorithm, named Generalized Pool-Adjacent-Violators (GPAV), that achieves both low computational complexity ($\mathcal{O}(|V|^2)$) and high accuracy (Burdakov *et al.*, 2006). By combining GPAV and ISO-TPR algorithms we obtain ISO-TPR: its pseudocode is available in Supplementary Fig. S2, while Supplementary Section S2 provides an analysis of the computational complexity of the algorithm.

2.3.3 The ISO-DESCENS algorithm

Valentini (2011) showed that in tree-based hierarchies the contribution of the descendants of a given node decays exponentially with their distance from the node itself. It is straightforward to see that this property also holds for DAG structured taxonomies. To overcome this limitation, which penalizes the predictions of the most specific terms of the GO, we designed the ISO-DESCENS algorithm:

$$\bar{y}_i := \frac{1}{1 + |\Delta_i|} (\hat{y}_i + \sum_{j \in \Delta_i} \bar{y}_j) \tag{5}$$

where Δ_i is the set of "positive" descendants of node i. The main feature of this strategy resides in emphasizing the contribution of the positive descendants of each node, to weigh more the information embedded in the leaf nodes, that are the most specific and informative from a biological standpoint. The weighted ISO-DESCENS variant can be designed by simply replacing ϕ_i with Δ_i in eq. (3). In addition, we designed a novel ISO-DESCENS variant, named ISOdescensTAU, by adding a weight $\tau \in [0, 1]$ to modulate the contribution of positive children and descendants (excluding children):

$$\bar{y}_i := \frac{\tau}{1 + |\phi_i|} (\hat{y}_i + \sum_{j \in \phi_i} \bar{y}_j) + \frac{1 - \tau}{1 + |\delta_i|} (\hat{y}_i + \sum_{j \in \delta_i} \bar{y}_j) \tag{6}$$

where $\delta_i = \Delta_i \setminus \phi_i$ are the descendants of i without its children. If $\tau = 1$ we consider only the contribution of the "positive" children of i, if $\tau = 0$ only the descendants that are not children contribute to the score, while for intermediate values of τ we can balance the contribution of ϕ_i and δ_i positive nodes.

2.3.4 The family of HEMDAG algorithms

Table 1 summarizes the hierarchical ensemble methods implemented in the HEMDAG library: those highlighted in bold represent novel algorithms presented for the first time in this work, while the others are previously proposed algorithms. These methods may differ in the parametric or parametric-free strategy chosen for the selection of the "positive" children or descendant nodes in the bottom-up step, and for the algorithm adopted in the consistency step (ISO-TPR or HTD). Two ensemble methods only execute the consistency step, 14 are parametric and 6 are parameter-free. Among the parametric variants, 6 boost the sensitivity of the flat predictions by propagating the children predictions, whereas the remaining 8 take directly into account also the descendant predictions. Each HEMDAG algorithm is detailed in Supplementary Section S3.

Table 1. HEMDAG family of hierarchical ensemble algorithms. In bold are highlighted the novel algorithms presented for the first time in this manuscript. The acronyms TF (Threshold-Free), T (Threshold, that includes both Fixed Threshold – FT and learned Threshold – LT strategies), W (Weight) and WT (Weight-Threshold) at the end of the HEMs names denote the bottom-up strategy by which the corresponding algorithm chooses the "positive" nodes. "None" in the column "Bottom-up step", points out that HTD and GPAV algorithms include only the consistency step.

HEMs	Subfamily	Bottom-up step	Consistency step	Type
HTD	HTD	None	HTD	Parameter-free
GPAV	GPAV		GPAV	
tprTF	TPR-DAG	Children	HTD	
ISOtprTF	ISO-TPR		GPAV	
descensTF	DESCENS	Descendants	HTD	
ISOdescensTF	ISO-DESCENS		GPAV	
tprT	TPR-DAG	Children	HTD	Parametric
tprW				
tprWT				
ISOtprT	ISO-TPR		GPAV	
ISOtprW				
ISOtprWT				
descensT	DESCENS	Descendants	HTD	
descensW				
descensWT				
descensTAU				
ISOdescensT	ISO-DESCENS		GPAV	
ISOdescensW				
ISOdescensWT				
ISOdescensTAU				

3 Results and discussion

To assess the soundness and robustness of HEMDAG methods, three different sets of experiments were performed. The goal of the first set (Section 3.1) is to empirically show that HEMDAG can be used to systematically improve flat predictions. We only selected parameter-free algorithms to avoid that the improvement over flat methods was simply due to the hyper-parameters tuning. In Section 3.2, a comparison and a characterization of the different HEMDAG methods is performed using synthetic data, whereas the last set of experiments (Section 3.3) aims at comparing the top performing algorithms resulting from Section 3.2 with SOTA structured output methods (Zhou *et al.*, 2019).

3.1 HEMDAG boosts predictions of flat classifiers

We compared the performance of 11 flat ML methods (Decision Tree, Generalized Linear Models, Linear Discriminant Analysis, Logit Boost, Multi Layer Perceptron, Naive Bayes, Random Forest, Support Vector Machine, Bagged Ensemble of Decision Trees, Extreme Gradient Boosting) with HEMDAG threshold free counterparts, by using the same methods as base learners.

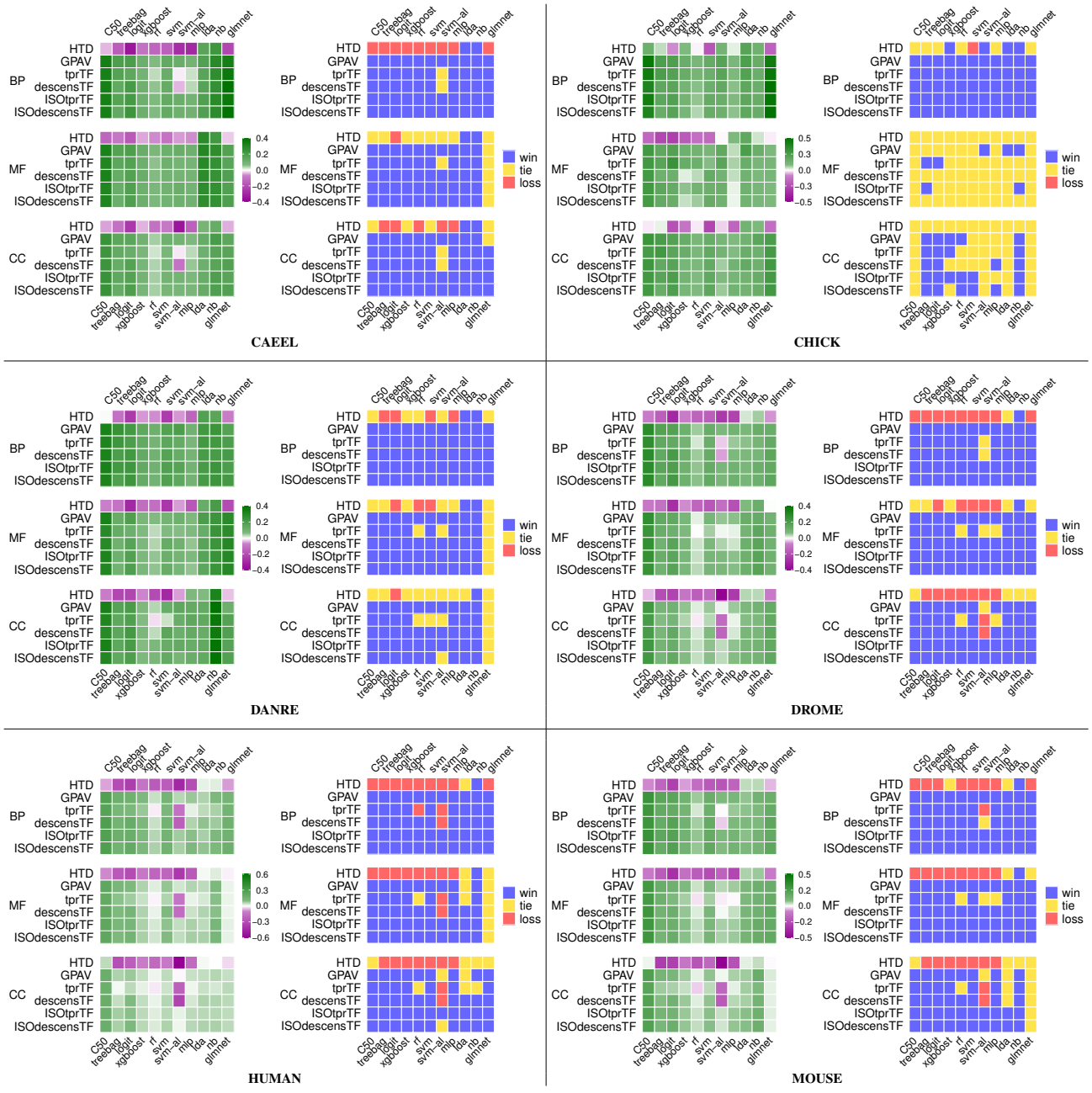

Fig. 2. Synoptic comparison of hierarchical ensemble methods versus flat approaches across GO ontologies and organisms. Heatmaps compare a hierarchical ensemble method (row) against a flat classifier (column) separately for each ontology and organism using the AUPRCmetric. For each species, the first heatmap shows the comparison for each pair hierarchical ensemble-flat classifier. The value of each cell is computed as follows: $HeatmapCell[i,j] = \frac{\text{AUPRC}_{hier_i} - \text{AUPRC}_{flat_j}}{max(\text{AUPRC}_{hier_i}, \text{AUPRC}_{flat_j})}$, where AUPRC_{hier_i} and AUPRC_{flat_j} are the average AUPRC values of the i^{th} hierarchical ensemble method and of the j^{th} flat classifier. Green cells denote better average results across GO terms for HEMDAG and magenta for flat classifiers. The second heatmap shows whether the performance difference of each pair of flat and HEMDAG classifiers is statistically significant according to a two-sided paired Wilcoxon rank sum test ($\alpha = 10^{-4}$). Blue cells indicate that a HEMDAG algorithm performs significantly better than a flat classifier, red cells that flat methods prevail, while gold cells no statistically significant difference observed.

We predict protein function of several animal species, including both invertebrates and vertebrates: *C. elegans* (CAEEL), *G. Gallus* (CHICK), *D. rerio* (DANRE), *D. melanogaster* (DROME), *H. sapiens* (HUMAN) and *M. musculus* (MOUSE), using GO annotations from the Gene Ontology Annotation (GOA) database (December 2017 release).

Experimental set-up. We used the rows of the STRING weighted adjacency matrix (Szklarczyk *et al.*, 2015) as features to train the learning machines. Our experiments involved more than 75K annotated proteins and about 7K unique GO terms. The data preparation pipeline is detailed in Supplementary Section S4. The generalization capabilities of the compared methods were assessed through a 5-fold cross-validation. Folds were stratified, i.e. positive instances were equally distributed among folds. To train a given model, we chose as positives all the proteins annotated with a GO term, whereas we retained as negatives all the remaining proteins not annotated with that term. To reduce the high

dimension of the feature space, we applied univariate feature selection methods. The source code adopted to build the datasets is available at `https://github.com/AnacletoLAB/godata-pipe`. Full details on the data and experimental set-up are available in Supplementary Section S6.4.

Experimental results By observing the heatmaps in Fig. 2 across organisms and GO domains, we can note that all the hierarchical methods (except for the HTD algorithm) significantly outperform flat predictors independently of the choice of the base learner, showing that hierarchical ensembles are able to enhance their flat counterparts. Notably, the improvement occurs across the three GO domains and the six considered organisms, except for CHICK where the flat and hierarchical approaches tie in the majority of cases. This is probably due to the scarcity of the available data (see Supplementary Table S8). Indeed, when flat classifiers provide random or very noisy predictions, we verified that only in about half cases the hierarchical correction is able to improve the flat predictions (see Section 3.2). Detailed experimental results that show the improvements achieved by HEMDAG with respect to flat methods for each organism and ontology domain are available in Supplementary Section S6.

We also performed a random search of model parameters (Bergstra and Bengio, 2012) to evaluate whether model selection can have an impact on the boosted performance of HEMDAG vs flat classifiers. To reduce the computational burden of the model selection implemented through a double (external and internal) cross-validation setting, we limited the experiments to two model organisms (*C. elegans* and *D. rerio*). Both the performance of flat classifiers and HEMDAG are slightly improved, and our proposed hierarchical ensembles confirmed their boosted predictions with respect to flat classifiers also when model selection is performed (see Supplementary Section S6.8).

Furthermore, we performed tests with the same two model organisms (*C. elegans* and *D. rerio*) by removing `IPI` annotations and functional interactions to assess the impact of a possible circularity in the data. Results confirmed the ability of HEMDAG of improving flat predictions, while overall prediction performance only decreased slightly with respect to the experimental setting enriched with IPI annotations. (Supplementary Section S6.9).

3.2 Characterization of the HEMDAG family of hierarchical ensemble methods

This section analyzes the behavior of all the HEMDAG methods to estimate their probability of improving the performance of flat predictors, using artificially generated data and the DAG scheme of the MF ontology. To this end and in order not to depend on a specific organism, we randomly generated GO annotations obeying the TPR for a set of proteins, and then 1000 repetitions of flat predictions, considering different levels of inconsistency with respect to the correct GO annotations. Finally, we computed how many times HEMDAG improves inconsistent flat predictions, i.e. we estimated the probability p that HEMDAG can improve flat predictions. Supplementary section S7 provides the details of the experimental design.

Generating random flat predictions. Let L be the matrix of the generated TPR-safe GO annotations and M the matrix of the randomly generated flat predictions (rows represent proteins and columns GO terms), then probability p was estimated according to four different experimental setups:

Unif. The matrix M is generated according to a continuous uniform distribution in $[0, 1]$;

Leaves. M is initially generated to fully respect the TPR rule, and having AUPRC $= 1$ for all GO terms; then, an inconsistency for each protein i is introduced at leaf level by uniformly picking up a leaf j (without replacement) and swapping the corresponding score with that of one of its parents $q \in par(j)$ (that is by swapping M_{ij} and M_{iq});

Root. The same as *Leaves* experiment, except for the inconsistency, which is placed at root level: for each protein i, a child j of the root r is uniformly selected, and its score M_{ij} exchanged with the parent one M_{ir};

Mixed. After generating TPR consistent matrices M as in *Leaves* and *Root* experiments, each entry M_{ij} was mixed with a uniform noise in $[0, s]$, with $s = 0.2, 0.4, 0.6, 0.8, 1$, to simulate realistic scenarios with both poor ($s = 1$) and good ($s = 0.1$) quality predictions. The levels of noise injected in the 1000 repetitions of M have been evenly partitioned among the different values of s.

Table 2 shows the 95% confidence intervals of the estimated p values for each HEMDAG method. Model selection has been performed through internal cross-validation and grid search over the hyperparameters of the HEMDAG methods. Supplementary Section S7.2 provides details about the tuning of the hyperparameters and the experimental set-up.

Table 2. Confidence interval for the probability p that the ensemble algorithms of the HEMDAG family improve flat predictions. Only the best HEMDAG performing methods are reported. Full results are available in Supplementary Table S11

Method	Unif	Leaves	Root	Mixed
GPAV	[0.469, 0.531]	[0.997, 1.000]	[0.997, 1.000]	[0.679, 0.736]
ISOtprTF	[0.462, 0.524]	[0.997, 1.000]	[0.997, 1.000]	[0.681, 0.737]
ISOdescensTF	[0.465, 0.527]	[0.997, 1.000]	[0.997, 1.000]	[0.681, 0.737]
ISOtprW	[0.488, 0.550]	[0.997, 1.000]	[0.997, 1.000]	[0.681, 0.737]
ISOdescensW	[0.481, 0.543]	[0.997, 1.000]	[0.997, 1.000]	[0.681, 0.737]
ISOdescensTAU	[0.476, 0.538]	[0.997, 1.000]	[0.997, 1.000]	[0.683, 0.739]

Analysis of results. The best performing HEMDAG methods, able to outperform flat methods in different experimental conditions, belong to the ISO-TPR subfamily newly proposed in this work (Table 2). Indeed ISO-TPR finds the "TPR-safe" solution closest (in a least square sense) to the "sensitivity enhanced" predictions performed in the bottom-up step. Among the bottom-up strategies, ISO-DESCENS methods, explicitly considering the positive descendants, achieve better results than those that consider the positive children only (see also Supplementary Table S11 for full details). This confirms our intuition that the information embedded in the leaf nodes should be propagated most. HTD performs worse than all the other hierarchical ensemble approaches in the Leaves, Root and Mixed scenarios (Supplementary Table S11). This is quite expected, since HTD removes the constraints violations by reducing the scores of the descendants, and thus also reducing the sensitivity. Quite surprisingly the ensemble parameter-free variants achieve comparable and sometimes even better results than the parametric ones. Nevertheless, this can be due to the limited grid of values adopted to adjust the parameters, which could not be expanded further due to the increase in computational time and to the massive set of experiments to be simulated. Overall these results confirm that HEMDAG and in particular ISO-TPR-based methods can enhance flat predictions, especially when flat predictions are not completely random. Indeed when flat predictions are completely random (Unif set-up), there is no advantage in using HEMDAG methods ($p = 0.5$). When flat classifiers provide relatively consistent predictions (Leaves and Root setting) we obtain $p \simeq 1$. In most intermediate cases (Mixed setup) the probability is significantly larger than 0.5, especially with ISO-TPR and ISO-DESCENS methods.

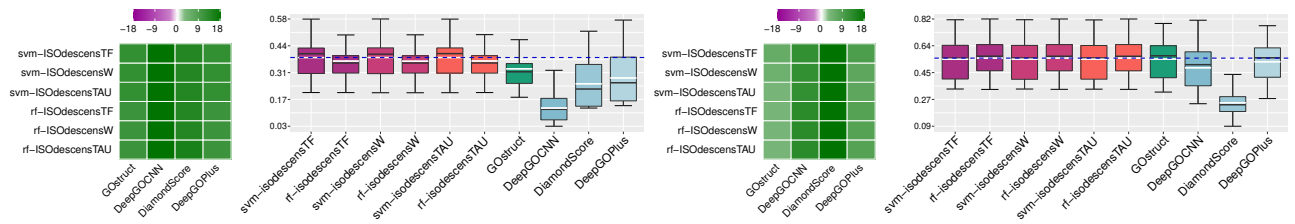

Fig. 3. Win-tie-loss heatmap and boxplot results for the time-lapse experiments comparing HEMDAG with SOTA structured output methods. Results are summarized across organisms and GO domains. Left: AUPRC; Right: F_{max} results. The win-tie-loss heatmaps count how many times HEMDAG algorithms win, tie or loose against a CAFA3 approach according to the two-sided paired Wilcoxon rank sum test ($\alpha = 0.05$). A heatmap cell can range from a maximum of +18 to a minimum of -18, due to the number of organisms (6) and GO domains (3). Boxplots show the distribution of average results across organisms and ontologies. The black and white line in the boxplots refer respectively to the median and the mean of a method computed across organisms and ontologies, whereas the blue dashed line represent the mean of the best HEMDAG methods.

3.3 Experimental comparison of HEMDAG with SOTA structured output methods

We evaluated the generalization performance of three of the best performing HEMDAG algorithms from Section 3.2, i.e. ISOdescensTF, ISOdescensW, ISOdescensTAU, versus the following SOTA structured output methods: GOstruct, a kernel-based output-structured approach (Sokolov and Ben-Hur, 2010), DeepGOCNN, a deep convolutional neural network, DiamondScore, a sequence similarity-based prediction method, and DeepGOPlus, an ensemble approach that combines DeepGOCNN and DiamondScore (Kulmanov and Hoehndorf, 2019). We selected the competing structured output methods according to the following criteria: a) they have been among the top ranked in the recent CAFA2 and CAFA3 international challenges and b) their standalone software is freely available for research purposes. We considered the same six organisms used in the experiments presented in Section 3.1, by covering more than 80K annotated proteins and about 7K of unique GO terms. We adopted a time-lapse hold-out procedure, i.e. predicting the new GO annotations of the June 2020 GO release using the annotations available in the GO release of December 2017.

Experimental set-up and performance metrics. As protein-protein interaction network, we used the most recent release of STRING – v11 (Szklarczyk *et al.*, 2018), and we downloaded the protein-GO term associations from GOA database (June 2020 release). The source code adopted to build the datasets is available at `https://github.com/AnacletoLAB/godata-pipe`. Full details of the experimental set-up are available in Supplementary Section S8.3.

As HEMDAG base learners, we adopted the ones that achieved the top performances (averaged across organisms and ontologies) in the experiments presented in Section 3.1, namely the random forest (*rf*) and the support vector machine (*svm*).

While ISOdescensTF is a non parametric method, for ISOdescensTAU and ISOdescensW we run grid search through internal 5-fold cross-validation to tune hyper-parameters w and τ (both in the range from 0.1 to 0.9 at 0.1 steps) by optimizing the AUPRC value. For what regards DeepGOPlus, we used the optimized CNN architecture as provided by the authors (Kulmanov and Hoehndorf, 2019), while parameter α was tuned by using the authors' implementation. We did not optimize the hyper-parameters of GOstruct, due to its impractical computational costs and we used the default parameter setting suggested by the GOstruct authors themselves (Sokolov and Ben-Hur, 2010).

Experimental results. The heatmaps and the boxplots depicted in Fig. 3 comparatively show HEMDAG versus SOTA structured output methods performances, by merging the results coming from all organisms and GO domains. The heatmaps show that HEMDAG significantly outperforms the structured output methods (across organisms and GO domains) for both the AUPRC and F_{max} metrics (Fig. 3). According to the two-sided paired Wilcoxon rank sum test ($\alpha = 0.05$) and considering AUPRC (resp. F_{max}), HEMDAG wins in the 78% (62%) of total cases; ties in the 19% (36%) of the cases and looses in the remaining 3% (2%) of cases. Detailed experimental results available in Supplementary Section S8 reports win-tie-loss results and compared boxplots for each organism, GO domain and performance metric, the compared precision at different recall levels, showing that HEMDAG is competitive with SOTA structured output prediction methods for AFP. Supplementary Table S15 shows that HEMDAG is also significantly faster than the competing methods.

3.4 Experiments summary

Among HEMDAG methods, the experimental results show that the newly proposed ISO-TPR algorithms should be chosen to improve flat predictions. In particular, as a "thumb rule", we suggest to use ISOdescensTAU as default HEMDAG method, since it obtained the best results with both synthetic and actual data for GO term prediction. Indeed this algorithm integrates isotonic regression that assures consistent predictions close to those obtained by the TPR algorithm in the bottom-up step (that improves sensitivity); moreover predictions at each node directly use all the "positive" predictions of the descendant nodes, and finally the contribution of the children nodes can be differentially weighted with respect of the rest of the descendant nodes. Interestingly, a relatively similar algorithm, i.e. descensTAU (that applies HTD instead of GPAV in the consistency step), was the algorithm that achieved the best results in the context of the the hierarchical prediction of HPO terms (Notaro *et al.*, 2019). A user can also experimentally compare the different HEMDAG methods to choose the one best suited for specific data, organism and GO domain. To this end the online tutorial (available at `https://hemdag.rtfd.io/en/latest/call.html`) offers specific functions and scripts to facilitate this task using the freely downloadable HEMDAG software library.

4 Conclusion

We proposed a suite of highly modular and scalable hierarchical ensemble methods to predict GO protein functions and able to satisfy the GO hierarchical constraints. Extensive experiments carried-out on several organisms, tens of thousands of proteins and thousands of GO terms showed that our methods systematically boost the predictions of "hierarchy-unaware" classifiers, and that they achieve competitive results, while exhibiting a lower computational complexity than SOTA structured output approaches. From a more general standpoint, HEMDAG is a flexible tool that can be applied upon any off-the-shelf classifier neglecting hierarchical constraints, to correct and improve its predictions. A critical

aspect of HEMDAG, and in general of all hierarchical ensemble methods, is that their performance depend on the ability of the underlying flat learner to provide accurate predictions, making the choice of the base learners crucial to attain good quality solutions. Our experimental results suggest that Random Forests and Support Vector Machines constitute in general a good choice, with Random Forests preferable when the goal is to predict the function of a given protein (protein-centric evaluation), and Support Vector Machines preferable when the aim is to prioritize the gene products with respect to a specific GO term (term-centric evaluation). More generally, considering that our experimental results support the hypothesis that HEMDAG can be applied to improve any flat prediction, especially when they are accurate, a possible future collaborative work with the AFP community could be the application of SOTA AFP prediction methods (e.g. the best CAFA3 methods) in conjunction with HEMDAG to verify whether our hierarchical ensemble approach can further improve their performance. Besides, the HEMDAG framework is general enough to be safely applied also to any DAG-structured ontology, such as HPO (Köhler *et al.*, 2018) or the Disease Ontology (Schriml *et al.*, 2018), but also to any tree-structured taxonomy, such as for instance FunCat (Ruepp *et al.*, 2004), being trees a particular case of DAGs. The highly modular algorithmic schema of HEMDAG allows also to easily introduce novel algorithmic variants, or bottom-up strategies to select "positive" descendant nodes. In addition, to further enhance the overall performance, multi-task learning algorithms (Widmer and Rätsch, 2012; Frasca and Cesa-Bianchi, 2016) can be in perspective employed as base learners of the HEMDAG framework, so as to embed the relationships among GO terms already during the classifier training phase.

Acknowledgments

We would like to thank the editor and the reviewers for their comments and suggestions.

Funding

This work has been supported by the Transition grant "UNIMI Partneriat H2020" [PSR2015-1720GVALE_01] and by the PSR 2019 project "Machine Learning and Big Data Analysis for Bioinformatics" [PSR2019_DIP_010_GVALE].

References

Ayer, M., *et al.* (1955). An empirical distribution function for sampling with incomplete information. *Ann. Math. Statist.*, **26**(4), 641–647.

Armano, G. (2015). Modelling Progressive Filtering. *Fundam. Inform.*, **138**(3), 285–320.

Barlow, R. (1972). *Statistical Inference Under Order Restrictions: The Theory and Application of Isotonic Regression*. J. Wiley.

Barlow, R. E. and Brunk, H. D. (1972). The isotonic regression problem and its dual. *J. Am. Stat. Assoc.*, **67**(337), 140–147.

Bergstra, J. and Bengio, Y. (2012). Random Search for Hyper-Parameter Optimization. *Journal of Machine Learning Research*, **12**, 281—305.

Burdakov, O., *et al.* (2006). *An O(n2) Algorithm for Isotonic Regression*, chapter 83, pages 25–33. Springer US, Boston, MA.

Cerri, R., *et al.* (2015). Hierarchical classification of gene ontology-based protein functions with neural networks. In *IJCNN*, pages 1–8.

Cerri, R., *et al.* (2016). Reduction strategies for hierarchical multi-label classification in protein function prediction. *BMC Bioinform.*, **17**, 373.

Cerri, R., *et al.* (2019). Inducing hierarchical multi-label classification rules with genetic algorithms. *Appl. Soft Comput.*, **77**, 584–604.

Cesa-Bianchi, N., *et al.* (2012). Synergy of multi-label hierarchical ensembles, data fusion, and cost-sensitive methods for gene functional inference. *Mach. Learn.*, **88**(1-2), 209–241.

Frasca, M. and Cesa-Bianchi, N. (2016). Multitask protein function prediction through task dissimilarity. *IEEE/ACM Trans. Comput. Biol. Bioinform.*

Frasca, M., *et al.* (2020). Multitask Hopfield Networks. In *ECML PKDD 2019*, volume 11907 of *Lecture Notes in Computer Science*. Springer.

Gene Ontology (GO) Consortium (2018). The Gene Ontology Resource: 20 years and still GOing strong. *NAR*, **47**(D1), D330–D338.

Grotzinger, S. J. and Witzgall, C. (1984). Projections onto order simplexes. *Appl. Math. Optim.*, **12**(1), 247–270.

Guan, Y., *et al.* (2008). Predicting gene function in a hierarchical context with an ensemble of classifiers. *Genome Biol.*, **9**, S3.

Jiang, Y., *et al.* (2016). An expanded evaluation of protein function prediction methods shows an improvement in accuracy. *Genome Biol.*, **17**(1), 184.

Juncker, A. S., *et al.* (2009). Sequence-based feature prediction and annotation of proteins. *Genome biol.*, **10**(2), 206.

Kahanda, I., *et al.* (2015). PHENOstruct: Prediction of human phenotype ontology terms using heterogeneous data sources. *F1000Res*, **4**, 259.

Kapur, N., *et al.* (2016). Ccr6 expression in colon cancer is associated with advanced disease and supports epithelial-to-mesenchymal transition. *Br. J. Cancer*, **114**(12), 1343.

Kocev, D., *et al.* (2013). Tree ensembles for predicting structured outputs. *Pattern Recognit.*, **46**(3), 817–833.

Köhler, S., *et al.* (2018). Expansion of the Human Phenotype Ontology (HPO) knowledge base and resources. *NAR*, **47**(D1), D1018–D1027.

Kulmanov, M. and Hoehndorf, R. (2019). DeepGOPlus: improved protein function prediction from sequence. *Bioinformatics*, **36**(2), 422–429.

Kulmanov, M., *et al.* (2017). DeepGO: predicting protein functions from sequence and interactions using a deep ontology-aware classifier. *Bioinformatics*, **34**(4), 660–668.

Kulmanov, M., *et al.* (2020). Semantic similarity and machine learning with ontologies. *Brief. Bioinform.*

Lampert, C. H. and Blaschko, M. B. (2009). Structured prediction by joint kernel support estimation. *Mach. Learn.*, **77**(2-3), 249–269.

Liu, L., *et al.* (2020). HPOLabeler: improving prediction of human protein–phenotype associations by learning to rank. *Bioinformatics*, **36**(14), 4180–4188.

Maxwell, W. L. and Muckstadt, J. A. (1985). Establishing consistent and realistic reorder intervals in production-distribution systems. *Oper. Res.*, **33**(6), 1316–1341.

Nakano, F. K., *et al.* (2020). Active learning for hierarchical multi-label classification. *Data Min. Knowl. Discov.*, **34**(5), 1496–1530.

Notaro, M., *et al.* (2017). Prediction of human phenotype ontology terms by means of hierarchical ensemble methods. *BMC Bioinform.*, **18**(1), 449:1–449:18.

Notaro, M., *et al.* (2019). Ensembling descendant term classifiers to improve gene - abnormal phenotype predictions. In *Computational Intelligence Methods for Bioinformatics and Biostatistics*, pages 70–80, Cham. Springer International Publishing.

Obozinski, G., *et al.* (2008). Consistent probabilistic output for protein function prediction. *Genome Biol.*, **9**, 135–142.

Radivojac, P., *et al.* (2013). A large-scale evaluation of computational protein function prediction. *Nat. methods*, **10**(3), 221–227.

Re, M., *et al.* (2012). A Fast Ranking Algorithm for Predicting Gene Functions in Biomolecular Networks. *IEEE/ACM Trans. Comput. Biol. Bioinform.*, **9**(6), 1812–1818.

Ruepp, A., *et al.* (2004). The FunCat, a functional annotation scheme for systematic classification of proteins from whole genomes. *NAR*, **32**(18), 5539–5545.

Schriml, L. M., *et al.* (2018). Human Disease Ontology 2018 update: classification, content and workflow expansion. *NAR*, **47**(D1), D955–D962.

Sharan, R., *et al.* (2007). Network-based prediction of protein function. *Mol. Sys. Biol.*, **8**(88).

Silla, C. and Freitas, A. (2011). A survey of hierarchical classification across different application domains. *Data Min. Knowl. Discov.*, **22**(1).

Sokolov, A. and Ben-Hur, A. (2010). Hierarchical classification of gene ontology terms using the gostruct method. *J. Bioinf. Comp. Biol.*, **8**(2), 357–376.

Szklarczyk, D., *et al.* (2015). STRING v10: protein-protein interaction networks, integrated over the tree of life. *NAR*, **43**, 447–452.

Szklarczyk, D., *et al.* (2018). STRING v11: protein–protein association networks with increased coverage, supporting functional discovery in genome-wide experimental datasets. *NAR*, **47**(D1), D607–D613.

Törönen, P., *et al.* (2018). Pannzer2: a rapid functional annotation web server. *NAR*, **46**(W1), W84–W88.

Valentini, G. (2011). True path rule hierarchical ensembles for genome-wide gene function prediction. *IEEE/ACM Trans. Comput. Biol. Bioinform.*, **8**(3), 832–847.

Valentini, G. (2014). Hierarchical ensemble methods for protein function prediction. *ISRN Bioinform*, **2014**, 34 pages.

Wang, L., *et al.* (2018). Large-scale protein function prediction using heterogeneous ensembles. *F1000Res.*, **7**(1577).

Widmer, C. and Rätsch, G. (2012). Multitask learning in computational biology. In *Proceedings of ICML Workshop on Unsupervised and Transfer Learning*, volume 27, pages 207–216. PMLR.

Wu, J., *et al.* (2014). Genome-wide protein function prediction through multi-instance multi-label learning. *IEEE/ACM Trans. Comput. Biology Bioinform.*, **11**(5), 891–902.

Yu, G., *et al.* (2015). Predicting protein functions using incomplete hierarchical labels. *BMC Bioinform.*, **16**, 1:1–1:12.

Zhou, N., *et al.* (2019). The CAFA challenge reports improved protein function prediction and new functional annotations for hundreds of genes through experimental screens. *Genome Biol.*, **20**(1), 244.