Hindawi Publishing Corporation
ISRN Bioinformatics
Volume 2014, Article ID 901419, 34 pages
http://dx.doi.org/10.1155/2014/901419

Review Article

Hierarchical Ensemble Methods for Protein Function Prediction

Giorgio Valentini

AnacletoLab-Dipartimento di Informatica, Università degli Studi di Milano, Via Comelico 39, 20135 Milano, Italy

Correspondence should be addressed to Giorgio Valentini; valentini@di.unimi.it

Received 2 February 2014; Accepted 25 February 2014; Published 5 May 2014

Academic Editors: T. Can and T. R. Hvidsten

Protein function prediction is a complex multiclass multilabel classification problem, characterized by multiple issues such as the incompleteness of the available annotations, the integration of multiple sources of high dimensional biomolecular data, the unbalance of several functional classes, and the difficulty of univocally determining negative examples. Moreover, the hierarchical relationships between functional classes that characterize both the Gene Ontology and FunCat taxonomies motivate the development of hierarchy-aware prediction methods that showed significantly better performances than hierarchical-unaware "flat" prediction methods. In this paper, we provide a comprehensive review of hierarchical methods for protein function prediction based on ensembles of learning machines. According to this general approach, a separate learning machine is trained to learn a specific functional term and then the resulting predictions are assembled in a "consensus" ensemble decision, taking into account the hierarchical relationships between classes. The main hierarchical ensemble methods proposed in the literature are discussed in the context of existing computational methods for protein function prediction, highlighting their characteristics, advantages, and limitations. Open problems of this exciting research area of computational biology are finally considered, outlining novel perspectives for future research.

1. Introduction

Exploiting the wealth of biomolecular data accumulated by novel high-throughput biotechnologies, "in silico" protein function prediction can generate hypotheses to drive the biological discovery and validation of protein functions [1]. Indeed, "in vitro" methods are costly in time and money, and automatic prediction methods can support the biologist in understanding the role of a protein or of a biological process or in annotating a new genome at high level of accuracy or more, in general, in solving problems of functional genomics [2].

The *Automated Function Prediction* (AFP) is a multiclass, multilabel classification problem characterized by hundreds or thousands of functional classes structured according to a predefined hierarchy. Even in principle, also unsupervised methods can be applied to AFP, due to the inherent difficulty of extracting functional classes without exploiting any available a priori information [3, 4]; usually supervised or semi-supervised learning methods are applied in order to exploit the available a priori information about gene annotations.

From a computational standpoint, AFP is a challenging problem for several reasons.

(i) *The number of functional classes is usually large*: hundreds for the Functional Catalogue (FunCat) [5] or thousands for the Gene Ontology (GO) [6].

(ii) *Proteins may be annotated for multiple functional classes*: since each protein may belong to more than one class at the same time, the classification problem is multilabel.

(iii) *Multiple sources of data are available for each protein*: high-throughput biotechnologies make an increasing number of sources of genomic and proteomic data available. Hence, in order to exploit all the information available for each protein, we need to learn methods that are able to integrate different data sources [7].

(iv) *Functional classes are hierarchically related*: annotations are not independent because functional classes are hierarchically organized; in general, known functional relationships (such as taxonomies) can be exploited to incorporate a priori knowledge in learning algorithms or to introduce explicit constraints between labels.

(v) *Small number of annotations for each class*: typically, functional classes are severely unbalanced, with a small number of available "positive" annotations.

(vi) *Multiple possible definitions of negative examples*: since we only have positive annotations (the total number of GO negative annotations is about 2500, considering all species (August 2013)), the notion of negative example is not uniquely determined, and different strategies of choosing negative examples can be applied in principle [8].

(vii) *Different reliability of functional labels*: functional annotations have different degrees of evidence; that is, each label is assigned to a gene with a specific level of reliability.

(viii) *Complex and noisy data*: data are usually complex (e.g., high-dimensional, large-scale, and graph-structured) and noisy.

Most of the computational methods for AFP have been applied to unicellular organisms (e.g., *S. cerevisiae*) [9–11], but recently several approaches have been applied to multicellular organisms (such as *M. musculus* or the *A. thaliana* plant model organisms [2, 12–16]).

Several computational approaches, and in particular machine learning methods, have been proposed to deal with the above issues, ranging from sequence-based methods [17] to network-based methods [18], structured output algorithm based on kernels [19], and hierarchical ensemble methods [20].

Other approaches focused primarily on the integration of multiple sources of data, since each type of genomic data captures only some aspects of the genes to be classified, and a specific source can be useful to learn a specific functional class while being irrelevant to others. In the literature, many approaches have been proposed to deal with this topic, for example, functional linkage networks integration [21], kernel fusion [11], vector space integration [22], and ensemble systems [23].

Extensive experimental studies showed that flat prediction, that is, predictions for each class made independently of the other classes, introduces significant inconsistencies in the classification, due to the violation of the *true path rule* that governs the functional annotations of genes both in the GO and in FunCat taxonomies [24]. According to this rule, positive predictions for a given term must be transferred to its "ancestor" terms and negative predictions to its descendants (see Appendix A and Section 7 for more details about the GO and the true path rule). Moreover flat predictions are difficult to interpret because they may be inconsistent with one another. A method that claims, for example, that a protein has homodimerization activity but does not have dimerization activity is clearly incorrect, and a biologist attempting to interpret these results would not likely trust either prediction [24].

It is worth noting that the results of the Critical Assessment of Functional Annotation (CAFA) challenge, a recent comprehensive critical assessment and comparison of different computational methods for AFP [16], showed that AFP is characterized by multiple complex issues, and one of the best performing CAFA methods corrected flat predictions taking into account the hierarchical relationships between functional terms, with an approach similar to that adopted by hierarchical ensemble methods [25]. Indeed, hierarchical ensemble methods embed in the learning process the relationships between functional classes. Usually, this is performed in a second "reconciliation" step, where the predictions are modified to make them consistent with the ontology [26–29]. More, in general, these methods exploit the relationships between ontology terms, structured according to a forest of trees [5] or a directed acyclic graph [6] to significantly improve prediction performances with respect to "flat" prediction methods [30–32].

Hierarchical classification and in particular ensemble methods for hierarchical classification have been applied in several domains different from protein function prediction, ranging from text categorization [33–35] to music genre classification [36–38], hierarchical image classification [39, 40] and video annotation [41], and automatic classification of worldwide web documents [42, 43]. The present review focuses on hierarchical ensemble methods for AFP. For a more general review on hierarchical classification methods and their applications in different domains, see [44].

The paper is structured as follows. In Section 2, we provide a synthetic picture of the main categories of protein function methods, to properly position hierarchical ensemble methods in the context of computational methods for AFP. In Section 3, the main common characteristics of hierarchical ensemble algorithms, as well as a general taxonomy of these methods, are proposed. The following five sections focus on the main families of hierarchical methods for AFP and discuss their main characteristics. Section 4 introduces hierarchical top-down methods, Section 5 Bayesian ensemble approaches, Section 6 reconciliation methods, Section 7 true path rule ensemble methods, and the last one (Section 8) ensembles based on decision trees. Section 9 critically discusses the main issues and limitations of hierarchical ensemble methods and shows that this approach, such as the other current approaches for AFP, cannot be successfully applied without considering the large set of complex learning issues that characterize the AFP problem. The last two sections discuss the open problems and future possible research lines in the context of hierarchical ensemble methods and summarize the main findings in this exciting research area. In the Appendix, some basic information about the FunCat and the GO, that is, the two main hierarchical ontologies that are widely used to annotate proteins in all organisms, are provided, as well as the characteristics of the hierarchical-aware performance measures proposed in the literature to assess the accuracy and the reliability of the predictions made by hierarchical computational methods.

2. A Taxonomy of Protein Function Prediction Methods

Several computational methods for the AFP problem have been proposed in the literature. Some methods provided predictions of a relatively small set of functional classes

[11, 45, 46], while others considered predictions extended to larger sets, using support vector machines and semidefinite programming [11], artificial neural networks [47], functional linkage networks [21, 48], Bayesian networks [45], or methods that combine functional linkage networks with learning machines using a logistic regression model [12] or simple algebraic operators [13].

Other research lines for AFP explicitly take into account the hierarchical nature of the multilabel classification problem. For instance, structured output methods are based on the joint kernelization of both input variables and output labels, using, for example, perceptron-like learning algorithms [49] or maximum-margin algorithms [50]. Other approaches improve the prediction of GO annotations by extracting implicit semantic relationships between genes and functions [51]. Finally, other methods adopted an ensemble approach [52] to take advantage of the intrinsic hierarchical nature of protein function prediction, explicitly considering the relationships between functional classes [24, 53–55].

Computational methods for AFP, mostly based on machine learning methods, can be schematically grouped in the following four families:

(1) sequence-based methods;

(2) network-based methods;

(3) kernel methods for structured output spaces;

(4) hierarchical ensemble methods.

This grouping is neither exhaustive nor strict, meaning that certain methods do not belong to any of these groups, and others belong to more than one.

2.1. Sequence-Based Methods. Algorithms based on alignment of sequences represent the first attempts to computationally predict the function of proteins [56, 57]: similar sequences are likely to share common functions, even if it is well known that secondary and tertiary structure conservation are usually more strictly related to protein functions. However, algorithms able to infer similarities between sequences are today standard methods of assigning functions to proteins in newly sequenced organisms [17, 58]. Of course, global or local structure comparison algorithms between proteins can be applied to detect functional properties [59], and, in this context, the integration of different sequence and structure-based prediction methods represents a major challenge [60].

Even if most of the research efforts for the design and development of AFP methods concentrated on machine learning methods, it is worth noting that in the AFP 2011 challenge [16] one of the best performing methods is represented by a sequence-based algorithm [61]. Indeed, when the only information available is represented by a raw sequence of amino acids or nucleotides, sequence-based methods can be competitive with state-of-the-art machine learning methods by exploiting homology-based inference [62].

2.2. Network-Based Methods. These methods usually represent each dataset through an undirected graph $G = (V, E)$, where nodes $v \in V$ correspond to gene/gene products and edges $e \in E$ are weighted according to the evidence of cofunctionality implied by data source [63, 64]. These algorithms are able to transfer annotations from previously annotated (labeled) nodes to unannotated (unlabeled) ones by exploiting "proximity relationships" between connected nodes. Basically, these methods are based on transductive label propagation algorithms that predict the labels of unannotated examples without using a global predictive model [14, 21, 45]. Several method exploited the semantic similarity between GO terms [65, 66] to derive functional similarity measures between genes to construct functional terms, using then supervised or semisupervised learning algorithm to infer GO annotations of genes [67–70].

Different strategies to learn the unlabeled nodes have been explored by "label propagation" algorithms, that is, methods able to "propagate" the labels of annotated proteins across the networks, by exploiting the topology of the underlying graph. For instance, methods based on the evaluation of the functional flow in graphs [64, 71], methods based on the Hopfield networks [48, 72, 73], methods based on the Markov [74, 75] and Gaussian random fields [14, 46], and also simple "guilt-by-association" methods [76, 77], based on the assumption that connected nodes/proteins in the functional networks are likely to share the same functions. Recently, methods based on kernelized score functions, able to exploit both local and global semisupervised learning strategies, have been successfully applied to AFP [78] as well as to disease gene prioritization [79] and drug repositioning problems [80, 81].

Reference [82] showed that different graph-based algorithms can be cast into a common framework where a quadratic cost objective function is minimized. In this framework, closed form solutions can be derived by solving a linear system of size equal to the cardinality of nodes (proteins) or using fast iterative procedures such as the Jacobi method [83]. A network-based approach, alternative to label propagation and exhibiting strong theoretical predictive guarantees in the so-called mistake bound model, has been recently proposed by [84].

2.3. Kernel Methods for Structured Output Spaces. By extending kernels to the output space, the multilabel hierarchical classification problem is solved globally: the multilabels are viewed as elements of a structured space modeled by suitable kernel functions [85–87], and structured predictions are viewed as a maximum a posteriori prediction problem [88].

Given a feature space $\mathcal{X}$ and a space of structured labels $\mathcal{Y}$, the task is to learn a mapping $f : \mathcal{X} \to \mathcal{Y}$ by an induced joint kernel function k that computes the "compatibility" of a given input-output pair (x, y): for each test example $x \in \mathcal{X}$, we need to determine the label $\overline{y} \in \mathcal{Y}$ such that $\overline{y} = \arg\max_{y \in \mathcal{Y}} k(x, y)$, for any $x \in \mathcal{X}$. By modeling probabilities by a log-linear model, and using a suitable feature map $\phi(x, y)$, we can define an induced joint kernel function that uses both inputs and outputs to compute the "compatibility" of a given input-output pair [88]

$$k : (\mathcal{X} \times \mathcal{Y}) \times (\mathcal{X} \times \mathcal{Y}) \longrightarrow \mathbb{R}. \quad (1)$$

Structured output methods infer a label $\hat{y}$ by finding the maximum of a function g that uses the previously defined joint kernel (1)

$$\hat{y} = \arg\max_{y \in \mathcal{Y}} g(x, y). \quad (2)$$

The *GOstruct* system implemented a structured perceptron and a variant of the structured support vector machine [85]. This approach has been successfully applied to the prediction of GO terms in mouse and other model organisms [19]. Structured output maximum-margin algorithms have been also applied to the tree-structured prediction of enzyme functions [50, 86].

2.4. Hierarchical Ensemble Methods. Other approaches take explicitly into account the hierarchical relationships between functional terms [26, 29, 53, 54, 89, 90]. Usually, they modify the "flat" predictions (i.e., predictions made independently of the hierarchical structure of the classes) and correct them improving accuracy and consistency of the multilabel annotations of proteins [24].

The flat approach makes predictions for each term independently and, consequently, the predictor may assign to a single protein a set of terms that are inconsistent with one another. A possible solution for this problem is to train a classifier for each term of the reference ontology to produce a set of prediction at each term and, finally, to reconcile the predictions by taking into account the relationships between the classes of the ontology. Different ensemble based algorithms have been proposed ranging from methods restricted to multilabels with single and no partial paths [91] to methods extended to multiple and also partial paths [92]. Many recent published works clearly demonstrated that this approach ensures an increment in precision, but this comes at expenses of the overall recall [2, 30].

In the next section, we discuss in detail hierarchical ensemble methods, since they constitute the main topic of this review.

3. Hierarchical Ensemble Methods: Exploiting the Hierarchy to Improve Protein Function Prediction

Ensemble methods are one of the main research areas of machine learning [52, 93–95]. From a general standpoint, ensembles of classifiers are sets of learning machines that work together to solve a classification problem (Figure 1). Empirical studies showed that in both classification and regression problems ensembles improve on single learning machines, and, moreover, large experimental studies compared the effectiveness of different ensemble methods on benchmark data sets [96–99], and they have been successfully applied to several computational biology problems [100–104]. Ensemble methods have been also successfully applied in an unsupervised setting [105, 106]. Several theories have been proposed to explain the characteristics and the successful application of ensembles to different application domains. For instance, Allwein, Schapire, and Singer interpreted the

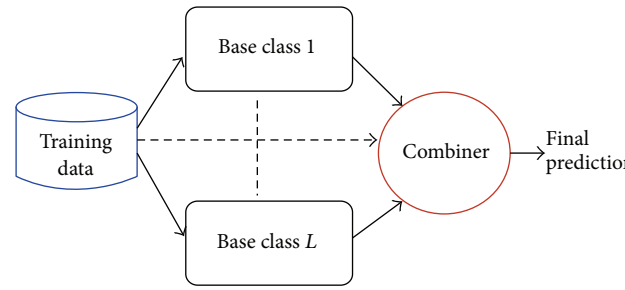


Figure 1: Ensemble of classifiers.

improved generalization capabilities of ensembles of learning machines in the framework of large margin classifiers [107, 108]; Kleinberg, in the context of Stochastic Discrimination Theory [109], and Breiman and Friedman in the light of the bias-variance analysis borrowed from classical statistics [110, 111]. The interest in this research area is motivated also by the availability of very fast computers and networks of workstations at a relatively low cost that allow the implementation and the experimentation of complex ensemble methods using off-the-shelf computer platforms.

Constraints between labels and, more in general, the issue of label dependence have been recognized to play a central role in multilabel learning [112]. Protein function prediction can be regarded as a paradigmatic multilabel classification problem, where the exploitation of a priori knowledge about the hierarchical relationships between the labels can dramatically improve classification performance [24, 27, 113].

In the context of AFP problems, ensemble methods reflect the hierarchy of functional terms in the structure of the ensemble itself: each base learner is associated with a node of the graph representing the functional hierarchy and learns a specific GO term or FunCat category. The predictions provided by the trained classifiers are then combined by exploiting the hierarchical relationships of the taxonomy.

In their more general form, hierarchical ensemble methods adopt a two-step learning strategy.

(1) In the first step, each base learner separately or interacting with connected base learners learns the protein functional category on a per term basis. In most cases, this yields a set of independent classification problems, where each base learning machine is trained to learn a specific functional term, independently of the other base learners.

(2) In the second step, the predictions provided by the trained classifiers are combined by considering the hierarchical relationships between the base classifiers modeled according to the hierarchy of the functional classes.

Figure 2 depicts the two learning steps of hierarchical ensemble methods. In the first step, a learning algorithm (a square object in Figure 2(a)) is applied to train the base classifiers associated with each class (represented with numbers from 1 to 9). Then, the resulting base classifiers

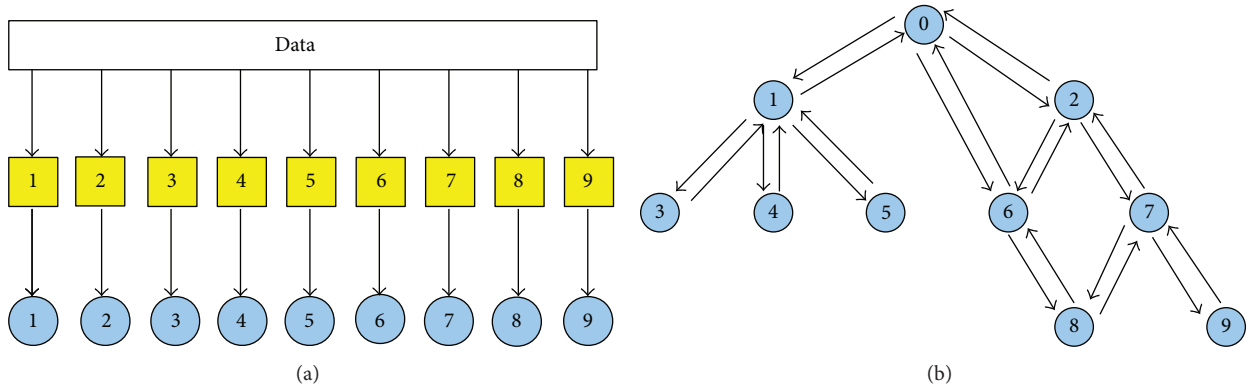


Figure 2: Schematic representation of the two main learning steps of hierarchical ensemble methods. (a) Training of base classifiers; (b) top-down and/or bottom-up propagation of the predictions.

(circles) in the prediction phase exploit the hierarchical relationships between classes to combine its predictions with those provided by the other base classifiers (Figure 2(b)). Note that the dummy 0 node is added to obtain a rooted hierarchy. Up and down arrows represent the possibility of combining predictions by exploiting those provided, respectively, by children and parents classifiers, according to a bottom-up or top-down learning strategy. Note that both "local" combinations are possible (e.g., the prediction of node 5 may depend only on the prediction of node 1), but also "global" combinations can be considered, by taking into account the predictions across the overall structure of the graph (e.g., predictions for node 9 can depend on all the predictions made by all the other base classifiers from 1 to 8). Moreover, both top-down propagation of the predictions (down arrows, Figure 2(b)) and bottom-up propagation (up arrows) can be considered, depending on the specific design of the hierarchical ensemble algorithm.

This ensemble approach is highly modular: in principle, any learning algorithm can be used to train the classifiers in the first step, and both annotation decisions, probabilities, or whatever scores provided by each base learner can be combined, depending on the characteristics of the specific hierarchical ensemble method.

In this section, we provide some basic notations and an ensemble taxonomy that will be used to introduce the different hierarchical ensemble methods for AFP.

3.1. Basic Notation. A gene/gene product g can be represented through a vector $\mathbf{x} \in \mathbb{R}^d$ having d different features (e.g., gene expression levels across d different conditions, sequence similarities with other genes/proteins, or presence or absence of a given domain in the corresponding protein or genetic or physical interaction with other proteins). Note that we, for the sake of simplicity and with a certain approximation, refer in the same way to genes and proteins, even if it is well known that a given gene may correspond to multiple proteins. A gene g is assigned to one or more functional classes in the set $C = \{c_1, c_2, \ldots, c_m\}$ structured according to a FunCat forest of trees T or a directed acyclic graph G of the Gene Ontology (usually a dummy root class c_0, which every gene belongs to, is added to T or G to facilitate the processing). The assignments are coded through a vector of multilabels $\mathbf{y} = (y_1, y_2, \ldots, y_m) \in \{0,1\}^m$, where g belongs to class c_i if and only if $y_i = 1$.

In both the Gene Ontology(GO) and FunCat taxonomies, the functional classes are structured according to a hierarchy and can be represented by a directed graph, where nodes correspond to classes and edges correspond to relationships between classes. Hence, the node corresponding to the class c_i can be simply denoted by i. We represent the set of children nodes of i by child(i), and the set of its parents by par(i). Moreover, $y_{\text{child}(i)}$ denotes the labels of the children classes of node i and analogously $y_{\text{par}(i)}$ denotes the labels of the parent classes of i. Note that in FunCat only one parent is permitted, since the overall hierarchy is a tree forest, while, in the GO, more parents are allowed, because the relationships are structured according to a directed acyclic graph.

Hierarchical ensemble methods train a set of calibrated classifiers, one for each node of the taxonomy T. These classifiers are used to derive estimates $\widehat{p}_i(g)$ of the probabilities $p_i(g) = \mathbb{P}(V_i = 1 \mid V_{\text{par}(i)} = 1, g)$ for all g and i, where $(V_1, \ldots, V_m) \in \{0,1\}^m$ is the vector random variable modeling the unknown multilabel of a gene g, and $V_{\text{par}(i)}$ denotes the random variables associated with the parents of node i. Note that $p_i(g)$ are probabilities conditioned to $V_{\text{par}(i)} = 1$, that is, the probability that a gene is annotated to a given term i, given that the gene is just annotated to its parent terms, thus respecting the true path rule. Ensemble methods infer a multilabel assignment $\widehat{\mathbf{y}} = (\widehat{y}_1, \ldots, \widehat{y}_m) \in \{0,1\}^m$ based on estimates $\widehat{p}_1(g), \ldots, \widehat{p}_m(g)$.

3.2. A Taxonomy of Hierarchical Ensemble Methods. Hierarchical ensemble methods for AFP share several characteristics, from the two-step learning approach to the exploitation of the hierarchical relationships between classes. For these reasons, it is quite difficult to clearly and univocally individuate taxonomy of hierarchical ensemble methods. Here, we show taxonomy useful mainly to describe and discuss existing methods for AFP. For a recent review and taxonomy of hierarchical ensemble methods, not specific for

AFP problems, we refer the reader to the comprehensive Silla and others' review [44].

In the following sections, we discuss the following groups of hierarchical ensemble methods:

(i) top-down ensemble methods. These methods are characterized by a simple top-down approach in the second step: only the output of the parent node/base classifier influences the output of the children, thus resulting in a top-down propagation of the decisions;

(ii) Bayesian ensemble methods. These are a class of methods theoretically well founded and in some cases they are optimal from a Bayesian standpoint;

(iii) Reconciliation methods. This is a heterogeneous class of heuristics by which we can combine the predictions of the base learners, by adopting different "local" or "global" combination strategies;

(iv) true path rule ensembles. These methods adopt a heuristic approach based on the "true path rule" that governs both the GO and FunCat ontologies;

(v) decision tree-based ensembles. These methods are characterized by the application of decision trees as base learners or by adopting decision tree-like learning strategies to combine predictions of the base learners.

Despite this general characterization, several methods could be assigned to different groups, and for several hierarchical ensemble methods it is difficult to assign them to any the introduced classes of methods.

For instance, in [114–116] the authors used the hierarchy only to construct training sets different for each term of the Gene Ontology, by determining positive and negative examples on the basis of the relationships between functional terms. In [89] for each classifier associated with a node, a gene is labeled as positive (i.e., belonging to the term associated with that node) if it actually belongs to that node or as negative if it does not belong to that node or to the ancestors or descendants of the node.

Other approaches exploited the correlation between nearby classes [32, 53, 117]. Shahbaba and Neal [53] take into account the hierarchy to introduce correlation between functional classes, using a multinomial logit model with Bayesian priors in the context of *E. coli* functional classification with Riley's hierarchies [118]. Bogdanov and Singh incorporated functional interrelationships between terms during the extraction of features based on annotations of neighboring genes and then applied a nearest-neighbor classifier to predict protein functions [117]. The HiBLADE method (hierarchical multilabel boosting with label dependency) [32] not only takes advantage of the preestablished hierarchical taxonomy of the classes but also effectively exploits the hidden correlation among the classes that is not shown through the class hierarchy, thereby improving the quality of the predictions. In particular, the dependencies of the children for each label in the hierarchy are captured and analyzed using the Bayes method and instance-based similarity. Experiments using the FunCat taxonomy and the yeast model organism show that the proposed method is competitive with TPR-W (Section 7.2) and HABYES-CS (Section 5.3) hierarchical ensemble methods.

An adaptation of a classical multiclass boosting algorithm [119] has been adapted to fit the hierarchical structure of the FunCat taxonomy [120]: the method is relatively simple and straightforward to be implemented and achieves competitive results for the AFP in the yeast model organism.

Finally, other hierarchical approaches have been proposed in the context of competitive networks learning framework. Competitive networks are well-known unsupervised and supervised methods able to map the input space into a structured output space where clusters or classes are usually arranged according to a grid topology and where learning adopts at the same way a competition, cooperation, and adaptation strategy [121]. Interestingly enough, in [122], the authors adopted this approach to predict the hierarchy of gene annotations in the yeast model organism, by using a tree-topology according to the FunCat taxonomy: each neuron is connected with its parent or with its children. Moreover, each neuron in tree-structured output layer is connected to all neurons of the input layer, representing the instances, that is, the set of genomic features associated with each gene to be classified. Results obtained with the hierarchy of enzyme commission codes showed that this approach is competitive with those obtained with hierarchical decision trees ensembles [29] (Section 8).

To provide a general picture of the methods discussed in the following sections, Table 1 summarizes their main characteristics. The first two columns report the name and a reference to the method, the third whether multiple or single paths across the taxonomy are allowed, and the next whether partial paths are considered (i.e., paths that do not end with a leaf). The successive columns refer to the class structure (a tree or a DAG), to the adoption or not of cost-sensitive (i.e., unbalance-aware) classification approaches, and to the adoption of strategies to properly select negative examples in the training phase. Finally, the last three columns summarize the type of the base learner used ("spec" means that only a specific type of base learner is allowed and "any" means that any type of learner can be used within the method), whether the method improves or not with respect to the flat approach, and the mode of processing of the nodes ("TD": top-down approach, and "TD&BUP": adopting both top-down and bottom-up strategies). Of course methods having more checkmarks are more flexible and in general methods that can process a DAG can also process tree-structured ontologies, but the opposite is not guaranteed, while the type of node processing relies on the way the information is propagated across the ontology. It is worth noting that all the considered methods improve on baseline "flat" classification methods.

4. Hierarchical Top-Down (HTD) Ensembles

These ensemble methods exploit the hierarchical relationships between functional terms in a top-to-bottom fashion, that is, considering only the relationships denoted by the down arrows in Figure 2(b). The basic hierarchical top-down

Table 1: Characteristics of some of the main hierarchical ensemble methods for AFP.

Methods	References	Multipath	Partial path	Class structure	Cost sens.	Sel neg.	Base learn	Improves on flat	Node process
HMC-LMLP	[124, 125]	√	√	TREE			any	√	TD
HTD-CS	[27]	√	√	TREE	√		any	√	TD
HTD-MULTI	[127]		√	TREE			any	√	TD
HTD-PERLEV	[128]			TREE			spec	√	TD
HTD-NET	[26]	√	√	DAG			any	√	TD
BAYES NET-ENS	[20]	√	√	DAG		√	spec	√	TD & BUP
HIER-MB and BFS	[30]	√	√	DAG			any	√	TD & BUP
HBAYES	[92, 135]	√	√	TREE		√	any	√	TD & BUP
HBAYES-CS	[123]	√	√	TREE	√	√	any	√	TD & BUP
Reconc-heuristic	[24]	√	√	DAG			any	√	TD
Cascaded log	[24]	√	√	DAG			any	√	TD
Projection-based	[24]	√	√	DAG			any	√	TD & BUP
TPR	[31, 142]	√	√	TREE		√	any	√	TD & BUP
TPR-W	[31]	√	√	TREE	√	√	any	√	TD & BUP
TPR-W weighted	[145]	√	√	TREE	√		any	√	TD & BUP
Decision-tree-ens	[29]	√	√	DAG			spec	√	TD & BUP

ensemble method (HTD) algorithm is straightforward: for each gene g, starting from the set of nodes at the first level of the graph G (denoted by root(G)), the classifier associated with the node $i \in G$ computes whether the gene belongs to the class c_i. If yes, the classification process continues recursively on the nodes $j \in \text{child}(i)$; otherwise, it stops at node i, and the nodes belonging to the descendants rooted at i are all set to 0. To introduce the method, we use probabilistic classifiers as base learners trained to predict class c_i associated with the node i of the hierarchical taxonomy. Their estimates $\widehat{p}_i(g)$ of $\mathbb{P}(V_i = 1 \mid V_{\text{par}(i)} = 1, g)$ are used by the HTD ensemble to classify a gene g as follows:

$$\widehat{y}_i = \begin{cases} \left\{\widehat{p}_i(g) > \frac{1}{2}\right\} & \text{if } i \in \text{root}(G) \\ \left\{\widehat{p}_i(g) > \frac{1}{2}\right\} & \text{if } i \notin \text{root}(G) \ \wedge \left\{\widehat{p}_{\text{par}(i)}(g) > \frac{1}{2}\right\} \\ 0 & \text{if } i \notin \text{root}(G) \ \wedge \left\{\widehat{p}_{\text{par}(i)}(g) \leq \frac{1}{2}\right\}, \end{cases} \tag{3}$$

where $\{x\} = 1$ if $x > 0$; otherwise, $\{x\} = 0$ and $\widehat{p}_{\text{par}(i)}$ is the probability predicted for the parent of the term i. It is easy to see that this procedure ensures that the predicted multilabels $\widehat{\mathbf{y}} = (\widehat{y}_1, \ldots, \widehat{y}_m)$ are consistent with the hierarchy. We can apply the same top-down procedure also using nonprobabilistic classifiers, that is, base learners generating continuous scores, or also discrete decisions, by slightly modifying (3).

In [123], a cost-sensitive version of the basic top-down hierarchical ensemble method HTD has been proposed: by assigning $\widehat{y}_i$ before the label of any j in the subtree rooted at i, the following rule is used:

$$\widehat{y}_i = \left\{\widehat{p}_i \geq \frac{1}{2}\right\} \times \left\{\widehat{y}_{\text{par}(i)} = 1\right\} \tag{4}$$

for $i = 1, \ldots, m$ (note that the guessed label $\widehat{y}_0$ of the root of G is always 1). Then, the cost-sensitive variant HTD-CS introduces a single cost-sensitive parameter $\tau > 0$ which replaces the threshold 1/2. The resulting rule for HTD-CS is then

$$\widehat{y}_i = \left\{\widehat{p}_i \geq \tau\right\} \times \left\{\widehat{y}_{\text{par}(i)} = 1\right\}. \tag{5}$$

By tuning τ, we may obtain ensembles with different precision/recall characteristics. Despite the simplicity of the hierarchical top-down methods, several works showed their effectiveness for AFP problems [28, 31].

For instance, Cerri and De Carvalho experimented different variants of top-down hierarchical ensemble methods for AFP [28, 124, 125]. The HMC-LMLP (hierarchical multilabel classification with local multilayer perceptron) successively trains a local MLP network for each hierarchical level, using the classical backpropagation algorithm [126]. Then, the output of the MLP for the first layer is used as input to train the MLP that learns the classes of the second level and so on (Figure 3). A gene is annotated to a class if its corresponding output in the MLP is larger than a predefined threshold; then, in a postprocessing phase (second-step of the hierarchical classification), inconsistent predictions are removed (i.e., classes predicted without the prediction of their superclasses) [125]. In practice, instead of using a dichotomic classifier for each node, the HMC-LMLP algorithm applies a single multiclass multilayer perceptron for each level of the hierarchy.

A related approach adopts multiclass classifiers (HTD-MULTI) for each node, instead of a simple binary classifier, and tries to find the most likely path from the root to the leaves of the hierarchy, considering simple techniques, such as the multiplication or the sum of the probabilities estimated at each node along the path [127]. The method has been applied to the cell cycle branch of the FunCat hierarchy with the yeast model organism, showing improvements with respect to classical hierarchical top-down methods, even if the proposed approach can only predict classes along a single

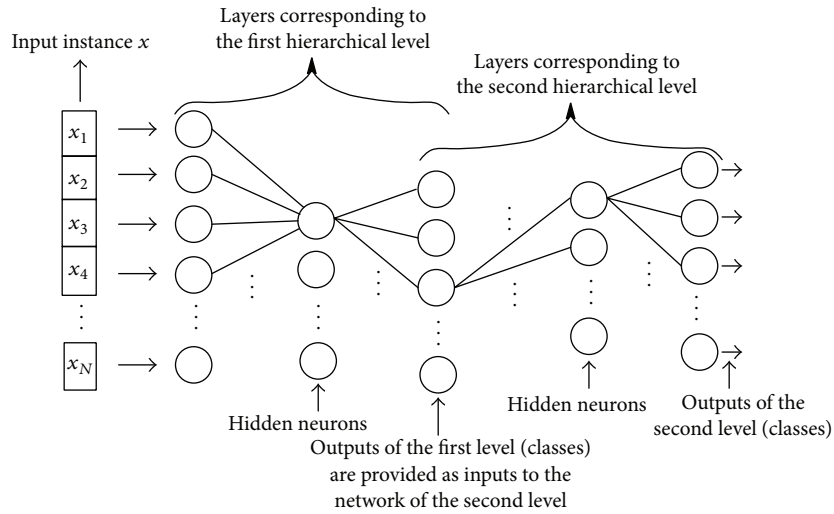


Figure 3: HMC-LMLP: outputs of the MLP responsible for the predictions in the first level are used as input to another MLP for the predictions in the second level (adapted from [125]).

"most likely path," thus not considering that in AFP we may have annotations involving multiple and partial paths.

Another method that introduces multiclass classifiers instead of simple dichotomic classifiers has been proposed by Paes et al. [128]: local per level multiclass classifiers (HTD-PERLEV) are trained to distinguish between the classes of a specific level of the hierarchy, and two different strategies to remove inconsistencies are introduced. The method has been applied to the hierarchical classification of enzymes using the EC taxonomy for the hierarchical classification of enzymes, but unfortunately this algorithm is not well suited to AFP, since leaf nodes are mandatory (that is partial path annotations are not allowed) and multilabel annotations along multiple paths are not allowed.

Another interesting top-down hierarchical approach proposed by the same authors is HMC-LP (hierarchical multilabel classification label-powerset), a hierarchical variation of the *label-powerset* nonhierarchical multilabel method [129], that has been applied to the prediction of gene function of the yeast model organism using 10 different data sets and the FunCat taxonomy [124]. According to the *label-powerset* approach, the method is based on a first label-combination step by which, for each example (gene), all classes assigned to the example are combined into a new and unique class, and this process is repeated for each level of the hierarchy. In this way, the original problem is transformed into a hierarchical single-label problem. In both the training and test phases, the top-down approach is applied, and at the end of the classification phase the original classes can be easily reconstructed [124]. In an experimental comparison using the FunCat taxonomy for *S. cerevisiae,* results showed that hierarchical top-down ensemble methods significantly outperform decision trees-based hierarchical methods, but no significant difference between different flavors of top-down hierarchical ensembles has been detected [28].

Top-down algorithms can be conceived also in the context of network-based methods (HTD-NET). For instance, in [26], a probabilistic model that combines relational protein-protein interaction data and the hierarchical structure of GO to predict true-path consistent function labels obeys the true path rule by setting the descendants of a node as negative whenever that node is set to negative. More precisely, the authors at first compute a local hierarchical conditional probability, in the sense that, for any nonroot GO term, only the parents affect its labeling. This probability is computed within a network-based framework assuming that the labeling of a gene is independent of any other genes given that of its neighbors (a sort of the Markov property with respect to gene functional interaction networks) and assuming also a binomial distribution for the number of neighbors labeled with child terms with respect to those labeled with the parent term. These assumptions are quite stringent but are necessary to make the model tractable. Then, a global hierarchical conditional probability is computed by recursively applying the previously computed local hierarchical conditional probability by considering all the ancestors. More precisely, by assuming that $\mathbb{P}(\widehat{y}_i = 1 \mid g, N(g))$, that is, the probability that a gene g is annotated for a a node i, given the status of the annotations of its neighborhood $N(g)$ in the functional networks, the global hierarchical conditional probability factorizes according to the GO graph as follows:

$$\mathbb{P}\left(\widehat{y}_i = 1 \mid g, N\left(g\right)\right) = \prod_{j \in \mathrm{anc}(i)} \mathbb{P}\left(\widehat{y}_j = 1 \mid \widehat{y}_{\mathrm{par}(j)} = 1, N_{\mathrm{loc}}\left(g\right)\right), \quad (6)$$

where $N_{\mathrm{loc}}(g)$ represents the local hierarchical neighborhood information on the parent-child GO term pair par(j) and j [26]. This approach guarantees to produce GO term label assignments consistent with the hierarchy, without the need of a postprocessing step.

Finally in [130], the author applied a hierarchical method to the classification of yeast FunCat categories. Despite its well-founded theoretical properties based on large margin

methods, this approach is conceived for one path hierarchical classification, and hence it results to be unsuited for hierarchical AFP, where usually multiple paths in the hierarchy should be considered, since in most cases genes can play different functional roles in the cell.

5. Ensemble Based Bayesian Approaches for Hierarchical Classification

These methods introduce a Bayesian approach to the hierarchical classification of proteins, by using the classical Bayes theorem or Bayesian networks to obtain tractable factorizations of the joint conditional probabilities from the original "full Bayesian" setting of the hierarchical AFP problem [20, 30] or to achieve "Bayes-optimal" solutions with respect to loss functions well suited to hierarchical problems [27, 92].

5.1. The Solution Based on Bayesian Networks. One of the first approaches addressing the issue of inconsistent predictions in the Gene Ontology is represented by the Bayesian approach proposed in [20] (BAYES NET-ENS). According to the general scheme of hierarchical ensemble methods, two main steps characterize the algorithm:

(1) flat prediction of each term/class (possibly inconsistent);

(2) Bayesian hierarchical combination scheme to allow collaborative error-correction over all nodes.

After training a set of base classifiers on each of the considered GO terms (in their work, the authors applied the method to 105 selected GO terms), we may have a set of (possibly inconsistent) $\hat{\mathbf{y}}$ predictions. The goal consists in finding a set of consistent $\mathbf{y}$ predictions, by maximizing the following equation derived from the Bayes theorem:

$$\mathbb{P}(y_1, \ldots, y_n \mid \hat{y}_1, \ldots, \hat{y}_n) = \frac{\mathbb{P}(\hat{y}_1, \ldots, \hat{y}_n \mid y_1, \ldots, y_n)\,\mathbb{P}(y_1, \ldots, y_n)}{Z}, \tag{7}$$

where n is the number of GO nodes/terms and Z is a constant normalization factor.

Since the direct solution of (7) is too hard, that is, exponential in time with respect to the number of nodes, the authors proposed a Bayesian network structure to solve this difficult problem, in order to exploit the relationships between the GO terms. More precisely, to reduce the complexity of the problem, the authors imposed the following constraints:

(1) y_i nodes conditioned to their children (GO structure constraints);

(2) $\hat{y}_i$ nodes conditioned on their label y_i (the Bayes rule);

(3) $\hat{y}_i$ are independent from both $\hat{y}_j$, $i \neq j$, and y_j, $i \neq j$, given y_i.

In other words, we can ensure that a label is 1 (positive) when any one of its children is 1 and the edges from y_i to $\hat{y}_i$ assure that a classifier output $\hat{y}_i$ is a random variable independent of all other classifier outputs $\hat{y}_j$ and labels y_j, given its true label y_i (Figure 4).

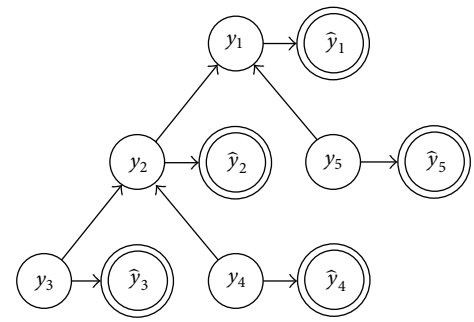


Figure 4: Bayesian network involved in the hierarchical classification (adapted from [20]).

More precisely, from the previous constraints we can derive the following equations:

from the first constraint:

$$\mathbb{P}(y_1, \ldots, y_n) = \prod_{i=1}^{n} \mathbb{P}(y_i \mid \text{child}(y_i)) \tag{8}$$

from the last two constraints:

$$\mathbb{P}(\hat{y}_1, \ldots, \hat{y}_n \mid y_1, \ldots, y_n) = \prod_{i=1}^{n} \mathbb{P}(\hat{y}_i \mid y_i). \tag{9}$$

Note that (8) can be inferred from training labels simply by counting, while (9) can be inferred by validation during training, by modeling the distribution of $\hat{y}_i$ outputs over positive and negative examples, by assuming a parametric model (e.g., Gaussian distribution; see Figure 5).

For the implementation of their method, the authors adopted bagged ensemble of SVMs [131] to make their predictions more robust and reliable at each node of the GO hierarchy, and median values of their outputs on out-of-bag examples have been used to estimate means and variances for each class. Finally, means and variances have been used as parameters of the Gaussian models used to estimate the conditional probabilities of (9).

Results with the 105 terms/nodes of the GO BP (model organism *S. cerevisiae*) showed substantial improvements with respect to nonhierarchical "flat" predictions: the hierarchical approach improves AUC results on 93 of the 105 GO terms (Figure 6).

5.2. The Markov Blanket and Approximated Breadth First Solution. In [30], the authors proposed an alternative approximated solution to the complex equation (7) by introducing the following two variants of the Bayesian integration:

(1) HIER-MB: hierarchical Bayesian combination involving nodes in the Markov blanket.

(2) HIER-BFS: hierarchical Bayesian combination involving the 30 first nodes visited through a breadth-first-search (BFS) in the GO graph.

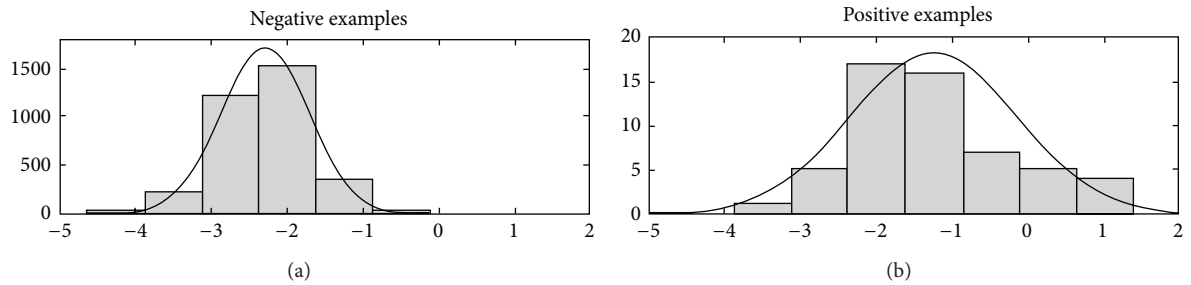


FIGURE 5: Distribution of positive and negative validation examples (a Gaussian distribution is assumed). Adapted from [20].

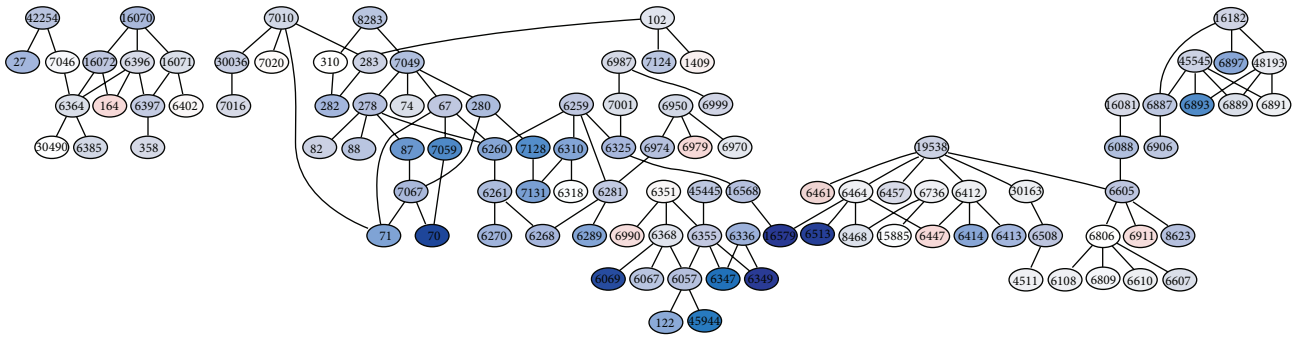


FIGURE 6: Improvements induced by the hierarchical prediction of the GO terms. Darker shades of blue indicate largest improvements, and darker shades of red indicate largest deterioration; white means no change (adapted from [20]).

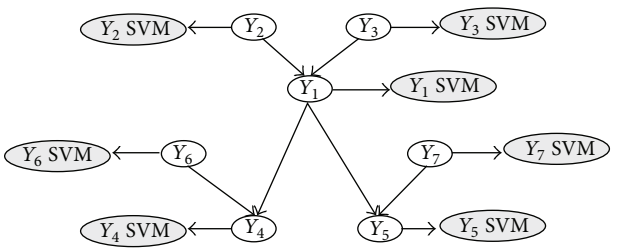


FIGURE 7: Markov blanket surrounding the GO term Y_1. Each GO term is represented as a blank node, while the SVM classifier output is represented as a gray node (adapted from [30]).

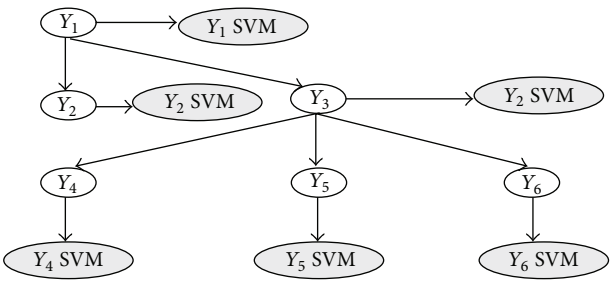


FIGURE 8: The breadth-first subnetwork stemming from Y_1. Each GO term is represented through a blank node and the SVM outputs are represented as gray nodes (adapted from [30]).

The method has been applied to the prediction of more than 2000 GO terms for the mouse model organism and performed among the top methods in the *MouseFunc* challenge [2].

The first approach (HIER-MB) modifies the output of the base learners (SVMs in the Guan et al. paper) taking into account the Bayesian network constructed using the Markov blanket surrounding the GO term of interest (Figure 7). In a Bayesian network, the Markov blanket of a node i is represented by its parents (par(i)), its children (child(i)), and its children's other parents. The Bayesian network involving the Markov blanket of node i is used to provide the prediction $\hat{y}_i$ of the ensemble, thus leveraging the local relationships of node i and the predictions for the nodes included in its Markov blanket.

To enlarge the size of the Bayesian subnetwork involved in the prediction of the node of interest, a variant based on the Bayesian networks constructed by applying a classical breadth-first search is the basis of the HIER-BFS algorithm. To reduce the complexity at most, 30 terms are included (i.e., the first 30 nodes reached by the breadth-first algorithm; see Figure 8). In the implementation, ensembles of 25 SVMs have been trained for each node, using vector space integration techniques [132] to integrate multiple sources of data.

Note that with both HIER-MB and HIER-BFS methods we do not take into account the overall topology of the GO network but only the terms related to the node for which we perform the prediction. Even if this general approach is reasonable and achieves good results, its main drawback is represented by the locality of the hierarchical integration (limited to the Markov blanket and the first 30 BFS nodes). Moreover, in previous works, it has been shown that the adopted integration strategy (vector space integration) is

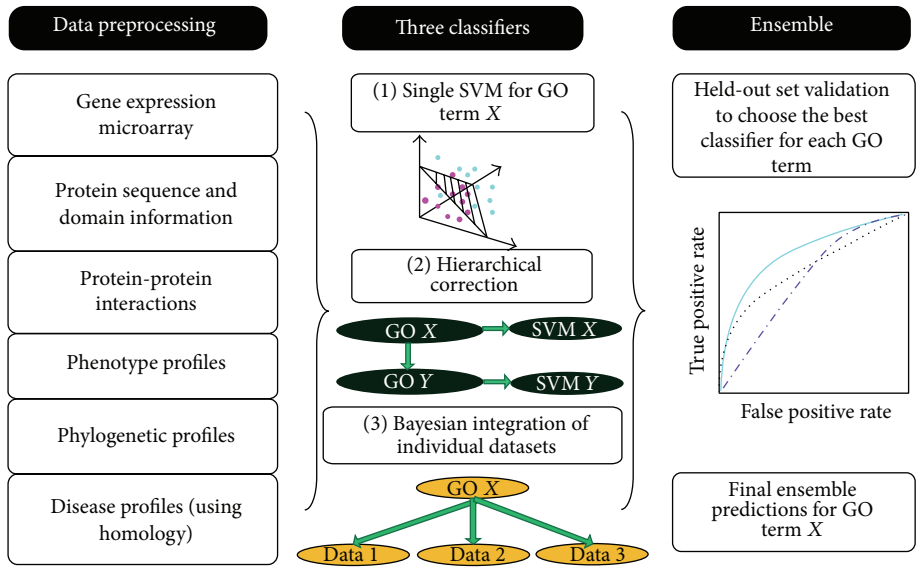


Figure 9: Integration of diverse methods and diverse sources of data in an ensemble framework for AFP prediction. The best classifier for each GO term is selected through held-out set validation (adapted from [30]).

in most cases worse than kernel fusion [11] and ensemble methods for data integration [23].

In the same work [30], the authors propose also a sort of "test and select" method [133], by which three different classification approaches (a) single flat SVMs, (b) Bayesian hierarchical correction, and (c) Naive Bayes combination are applied, and for each GO term the best one is selected by internal cross-validation (Figure 9).

It is worth noting that other approaches adopted Bayesian networks to resolve the hierarchical constraints underlying the GO taxonomy. For instance, in the *FALCON* algorithm the GO is modeled as a Bayesian network and for any given input the algorithm returns the most probable GO term assignment in accordance with the GO structure, by using an evolutionary-based optimization algorithm [134].

5.3. HBAYES: An "Optimal" Hierarchical Bayesian Ensemble Approach. The HBAYES ensemble method [92, 135] is a general technique for solving hierarchical classification problems on generic taxonomies G structured according to forest of trees. The method consists in training a calibrated classifier at each node of the taxonomy. In principle, any algorithm (e.g., support vector machines or artificial neural networks) whose classifications are obtained by thresholding a real prediction $\widehat{p}$, for example, $\widehat{y} = \mathrm{SGN}(\widehat{p})$, can be used as base learner. The real-valued outputs $\widehat{p}_i(g)$ of the calibrated classifier for node i on the gene g are viewed as estimates of the probabilities $p_i(g) = \mathbb{P}(y_i = 1 \mid y_{\mathrm{par}(i)} = 1, g)$. The distribution of the random Boolean vector $\mathbf{Y}$ is assumed to be

$$\mathbb{P}(\mathbf{Y} = \mathbf{y}) = \prod_{i=1}^{m} \mathbb{P}\left(Y_i = y_i \mid Y_{\mathrm{par}(i)} = 1, g\right) \quad \forall \mathbf{y} \in \{0,1\}^m, \tag{10}$$

where, in order to enforce that only multilabels $\mathbf{Y}$ that respect the hierarchy have nonzero probability, it is imposed that $\mathbb{P}(Y_i = 1 \mid Y_{\mathrm{par}(i)} = 0, g) = 0$ for all nodes $i = 1, \ldots, m$ and all g. This implies that the base learner at node i is only trained on the subset of the training set including all examples $(g, \mathbf{y})$ such that $y_{\mathrm{par}(i)} = 1$.

5.3.1. HBAYES Ensembles for Protein Function Prediction. H-loss is a measure of discrepancy between multilabels based on a simple intuition: if a parent class has been predicted wrongly, then errors in its descendants should not be taken into account. Given fixed cost coefficients $\theta_1, \ldots, \theta_m > 0$, the H-loss $\ell_H(\widehat{\mathbf{y}}, \mathbf{v})$ between multilabels $\widehat{\mathbf{y}}$ and $\mathbf{v}$ is computed as follows: all paths in the taxonomy T from the root down to each leaf are examined and whenever a node $i \in \{1, \ldots, m\}$ is encountered such that $\widehat{y}_i \neq v_i$, θ_i is added to the loss, while all the other loss contributions from the subtree rooted at i are discarded. This method assumes that, given a gene g, the distribution of the labels $\mathbf{V} = (V_1, \ldots, V_m)$ is $\mathbb{P}(\mathbf{V} = \mathbf{v}) = \prod_{i=1}^{m} p_i(g)$ for all $\mathbf{v} \in \{0,1\}^m$, where $p_i(g) = \mathbb{P}(V_i = v_i \mid V_{\mathrm{par}(i)} = 1, g)$. According to the true path rule, it is imposed that $\mathbb{P}(V_i = 1 \mid V_{\mathrm{par}(i)} = 0, g) = 0$ for all nodes i and all genes g.

In the evaluation phase, HBAYES predicts the Bayes-optimal multilabel $\widehat{\mathbf{y}} \in \{0,1\}^m$ for a gene g based on the estimates $\widehat{p}_i(g)$ for $i = 1, \ldots, m$. By definition of Bayes-optimality, the optimal multilabel for g is the one that minimizes the loss when the true multilabel $\mathbf{V}$ is drawn from the joint distribution computed from the estimated conditionals $\widehat{p}_i(g)$. That is,

$$\widehat{\mathbf{y}} = \operatorname*{argmin}_{\mathbf{y} \in \{0,1\}^m} \mathbb{E}\left[\ell_H(\mathbf{y}, \mathbf{V}) \mid g\right]. \tag{11}$$

In other words, the ensemble method HBAYES provides an approximation of the optimal Bayesian classifier with respect to the H-loss [135]. More precisely, as shown in [27] the following theorem holds.

Theorem 1. *For any tree T and gene g the multilabel generated according to the HBAYES prediction rule is the Bayes-optimal classification of g for the H-loss.*

In the evaluation phase, the uniform cost coefficients $\theta_i = 1$, for $i = 1, \ldots, m$, are used. However, since with uniform coefficients the H-loss can be made small simply by predicting sparse multilabels (i.e., multilabels $\hat{\mathbf{y}}$ such that $\sum_i \hat{y}_i$ is small), in the training phase the cost coefficients are set to $\theta_i = 1/|\text{root}(G)|$, if $i \in \text{root}(G)$, and to $\theta_i = \theta_j/|\text{child}(j)|$ with $j = \text{par}(i)$, if otherwise. This normalizes the H-loss, in the sense that the maximal H-loss contribution of all nodes in a subtree excluding its root equals that of its root.

Let $\{E\}$ be the indicator function of event E. Given g and the estimates $\hat{p}_i = \hat{p}_i(g)$ for $i = 1, \ldots, m$, the HBAYES prediction rule can be formulated as follows.

HBAYES Prediction Rule. Initially, set the labels of each node i to

$$\hat{y}_i = \underset{y \in \{0,1\}}{\text{argmin}} \left(\theta_i \hat{p}_i (1 - y) + \theta_i (1 - \hat{p}_i)\, y + \hat{p}_i \{y = 1\} \sum_{j \in \text{child}(i)} H_j(\hat{\mathbf{y}}) \right), \tag{12}$$

where

$$H_j(\hat{\mathbf{y}}) = \theta_j \hat{p}_j \left(1 - \hat{y}_j\right) + \theta_j \left(1 - \hat{p}_j\right) \hat{y}_j + \hat{p}_j \left\{\hat{y}_j = 1\right\} \sum_{k \in \text{child}(j)} H_k(\hat{\mathbf{y}}) \tag{13}$$

is recursively defined over the nodes j in the subtree rooted at i with each $\hat{y}_j$ set according to (12).

Then, if $\hat{y}_i$ is set to zero, set all nodes in the subtree rooted at i to zero as well.

It is worth noting that $\hat{\mathbf{y}}$ can be computed for a given g via a simple bottom-up message-passing procedure. It can be shown that if all child nodes k of i have $\hat{p}_k$ close to a half, then the Bayes-optimal label of i tends to be 0 irrespective of the value of $\hat{p}_i$. On the contrary, if i's children all have $\hat{p}_k$ close to either 0 or 1, then the Bayes-optimal label of i is based on $\hat{p}_i$ only, ignoring the children. This behaviour can be intuitively explained in the following way: the estimate $\hat{p}_k$ is built based only on the examples on which the parent i of k is positive; hence, a "neutral" estimate $\hat{p}_k = 1/2$ signals that the current instance is a negative example for the parent i. Experimental results show that this approach achieves comparable results with the TPR method (Section 7), an ensemble approach based on the "true path rule" [136].

5.3.2. HBAYES-CS: The Cost-Sensitive Version. The HBAYES-CS is the cost-sensitive version of HBAYES proposed in [27]. By this approach, the misclassification cost coefficient θ_i for node i is split into two terms θ_i^+ and θ_i^- for taking into account misclassifications, respectively, for positive and negative examples. By considering separately these two terms, (12) can be rewritten as

$$\hat{y}_i = \underset{y \in \{0,1\}}{\text{argmin}} \left(\theta_i^- \hat{p}_i (1 - y) + \theta_i^+ (1 - \hat{p}_i)\, y + \hat{p}_i \{y = 1\} \sum_{j \in \text{child}(i)} H_j(\hat{\mathbf{y}}) \right), \tag{14}$$

where the expression of $H_j(\hat{y})$ gets changed correspondingly. By introducing a factor $\alpha \geq 0$ such that $\theta_i^- = \alpha \theta_i^+$ while keeping $\theta_i^+ + \theta_i^- = 2\theta_i$, the relative costs of false positives and false negatives can be parameterized, thus allowing us to further rewrite the hierarchical Bayesian rule (Section 5.3.1) as follows:

$$\hat{y}_i = 1 \Longleftrightarrow \hat{p}_i \left(2\theta_i - \sum_{j \in \text{child}(i)} H_j \right) \geq \frac{2\theta_i}{1 + \alpha}. \tag{15}$$

By setting $\alpha = 1$, we obtain the original version of the hierarchical Bayesian ensemble and by incrementing α we introduce progressively lower costs for positive predictions. In this way, we can obtain that the recall of the ensemble tends to increase, eventually at the expenses of the precision, and by tuning the α parameter we can obtain different combinations of precision/recall values.

In principle, a cost factor α_i can be set for each node i to explicitly take into account the unbalance between the number of positive n_i^+ and negative n_i^- examples, estimated from the training data

$$\alpha_i = \frac{n_i^-}{n_i^+} \Longrightarrow \theta_i^+ = \frac{2}{n_i^-/n_i^+ + 1} \theta_i = \frac{2n_i^+}{n_i^- + n_i^+} \theta_i. \tag{16}$$

The decision rule (15) at each node then becomes

$$\hat{y}_i = 1 \Longleftrightarrow p_i \left(2\theta_i - \sum_{j \in \text{child}(i)} H_j \right) \geq \frac{2\theta_i}{1 + \alpha_i} = \frac{2\theta_i n_i^+}{n_i^- + n_i^+}. \tag{17}$$

Results obtained with the yeast model organism showed that HBAYES-CS significantly outperform HTD methods [27, 136].

6. Reconciliation Methods

Hierarchical ensemble methods are basically two-step methods, since at first provide predictions for the single classes and then arrange these predictions to take into account the functional relationships between GO terms. Noble and colleagues name this general approach *reconciliation methods* [24]: they proposed methods for calibrating and combining independent predictions to obtain a set of probabilistic predictions that are consistent with the topology of the ontology. They applied their ensemble methods to the genome-wide and ontology-wide function prediction with *M. musculus*, involving about 3000 GO terms.

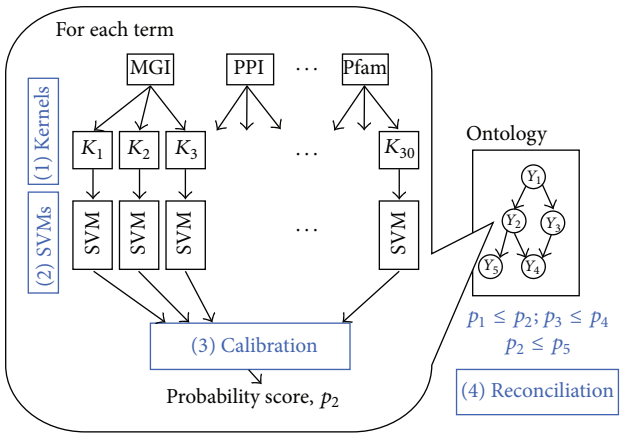


Figure 10: The overall scheme of reconciliation methods (adapted from [24]).

Their goal consists in providing consistent predictions, that is, predictions whose confidence (e.g., posterior probability) increases as we ascend from more specific to more general terms in the GO. Moreover, another important issue of these methods is the availability of confidence values associated with the predictions that can be interpreted as probabilities that a protein has a certain function given the information provided by the data.

The overall reconciliation approach can be summarized in the following four basic steps (Figure 10):

(1) *Kernel computation*: at first a set of kernels is computed from the available data. We may choose kernel specific for each source of data (e.g., diffusion kernels for protein-protein interaction data [137], linear or Gaussian kernel for expression data, and string kernel for sequence data [138]). Multiple kernels for the same type of data can also be constructed [24].

(2) *SVM learning*: SVMs are used as base learners using the kernels selected at the previous step; the training is performed by internal cross-validation to avoid overfitting, and a local cost-sensitive strategy is applied, by tuning separately the C regularization factor for positive and negative examples. Note that the authors in their experiments used SVMs as base learners but any meaningful classifier could be used at this step.

(3) *Calibration*: to produce individual probabilistic outputs from the set of SVM outputs corresponding to one GO term, a logistic regression approach is applied. In this way, a calibration of the individual SVM outputs is obtained, resulting in a probabilistic prediction of the random variable Y_i, for each node/term i of the hierarchy, given the outputs $\widehat{y}_i$ of the SVM classifiers.

(4) *Reconciliation*: the first three steps generate unreconciled outputs; that is, in practice a "flat" ensemble is applied that may generate inconsistent predictions with respect to the given taxonomy. In this step, the outputs of step three are processed by a "reconciliation method." The goal of this stage is to combine predictions for each term to produce predictions that are consistent with the ontology, meaning that all the probabilities assigned to the ancestors of a GO term are larger than the probability assigned to that term.

The first three steps are basically the same for (or very similar to) each reconciliation ensemble method. The crucial step is represented by the fourth, that is, the reconciliation step, and different ensemble algorithms can be designed to implement it. The authors proposed 11 different ensemble methods for the reconciliation of the base classifier outputs. Schematically, they can be subdivided into the following four main classes of ensembles:

(1) heuristic methods;

(2) Bayesian network-based methods;

(3) cascaded logistic regression;

(4) projection-based methods.

6.1. Heuristic Methods. These approaches preserve the "reconciliation property"

$$\forall i, j \in G, \qquad (i, j) \in G \Longrightarrow \widehat{p}_i \geq \widehat{p}_j \tag{18}$$

through simple heuristic modifications of the probabilities computed at step 3 of the overall reconciliation scheme.

(i) The MAX method simply chooses the largest logistic regression value for the node i and all its descendants desc

$$p_i = \max_{j \in \mathrm{desc}(i)} \widehat{p}_j. \tag{19}$$

(ii) The AND method implements the notion that the probability of all ancestral GO terms anc(i) of a given term/node i is large, assuming that, conditional on the data, all predictions are independent

$$p_i = \prod_{j \in \mathrm{anc}(i)} \widehat{p}_j. \tag{20}$$

(iii) OR estimates the probability that the node i is annotated at least for one of the descendant GO terms, assuming again that, conditional on the data, all predictions are independent

$$1 - p_i = \prod_{j \in \mathrm{desc}(i)} \left(1 - \widehat{p}_j\right). \tag{21}$$

6.2. Cascaded Logistic Regression. Instead of modeling class-conditional probabilities, as required by the Bayesian approach, logistic regression can be used instead to directly model posterior probabilities. Considering that modeling conditional densities are in most cases difficult (also using strong independence assumptions as shown in Section 5.1),

the choice of logistic regression could be a reasonable one. In [24], the authors embedded in the logistic regression setting the hierarchical dependencies between terms. By assuming that a random variable $\mathbf{X}$ whose values represent the features of the gene g of interest is associated to a given gene g and assuming that $\mathbb{P}(\mathbf{Y} = \mathbf{y} \mid \mathbf{X} = \mathbf{x})$ factorizes according to the GO graph, then it follows

$$\begin{aligned}&\mathbb{P}\left(\mathbf{Y} = \mathbf{y} \mid \mathbf{X} = \mathbf{x}\right)\\&\quad = \prod_i \mathbb{P}\left(Y_i = y_i \mid \forall j \in \operatorname{par}(i)\, Y_j = y_j, X_i = x_i\right)\end{aligned} \tag{22}$$

with $\mathbb{P}(Y_i = 1 \mid \forall j \in \operatorname{par}(i)\ Y_j = 0, X_i = x_i) = 0$. The authors estimated $\mathbb{P}(Y_i = 1 \mid \forall j \in \operatorname{par}(i) Y_j = 1, X_i = x_i)$ with logistic regression. This approach is quite similar to fitting independent logistic regressions, but note that in this case only examples of proteins having all parents GO terms are used to fit the model, thus implicitly taking into account the hierarchical relationships between GO terms.

6.3. Bayesian Network-Based Methods. These methods are variants of the Bayesian network approach proposed in [20] (Section 5.1): the GO is viewed as a graphical model where a joint Bayesian prior is put on the binary GO term variables y_i. The authors proposed four variants that can be summarized as follows:

(i) the BPAL is a belief propagation approach with asymmetric Laplace likelihoods. The graphical model has edges directed from more general terms to more specific terms. Differently from [20], the distribution of each SVM output is modeled as an asymmetric Laplace distribution, and a variational inference algorithm that solves an optimization problem whose minimizer is the set of marginal probabilities of the distribution is used to estimate the posterior probabilities of the ensemble [139];

(ii) the BPALF approach is similar to BPAL but with edges inverted and directed from more specific terms to more general terms;

(iii) the BPLR is a heuristic variant of BPAL, where, in the inference algorithm, the Bayesian log posterior ratio for Y_i is replaced by the marginal log posterior ratio obtained from the logistic regression (LR);

(iv) The BPLRF is equal to BPLR but with reversed edges.

6.4. Projection-Based Methods. A different approach is represented by methods that directly use the calibrated values obtained from logistic regression (step 3 of the overall scheme of the reconciliation methods) to find the closest set of values that are consistent with the ontology. This approach leads to a constrained optimization problem. The main contribution of the Obozinski et al. work [24] is represented by the introduction of projection reconciliation techniques based on isotonic regression [140] and the Kullback-Leibler divergence.

The *isotonic regression* method tries to find a set of marginal probabilities p_i that are close to the set of calibrated values $\widehat{p}_i$ obtained from the logistic regression. The Euclidean distance is used as a measure of closeness. Hence, considering that the "reconciliation property" requires that $p_i \geq p_j$ when $(i, j) \in E$, this approach yields the following quadratic program:

$$\begin{aligned}&\min_{p_i, i \in I} \quad \sum_{i \in I} \left(p_i - \widehat{p}_i\right)^2\\&\text{s.t.} \quad p_j \leq p_i, \quad (i, j) \in E.\end{aligned} \tag{23}$$

This problem is the classical isotonic regression problems that can be solved using an interior point solver or also approximated algorithm when the number of edges of the graph is too large [141].

Considering that we deal with probabilities, a natural measure of distance between probability density functions $f(\mathbf{x})$ and $g(\mathbf{x})$ defined with respect to a random variable $\mathbf{x}$ is represented by the Kullback-Leibler divergence $D_{f_{\mathbf{x}} \| g_{\mathbf{x}}}$ as follows:

$$D_{f_{\mathbf{x}} \| g_{\mathbf{x}}} = \int_{-\infty}^{\infty} f(\mathbf{x}) \log\left(\frac{f(\mathbf{x})}{g(\mathbf{x})}\right) d\mathbf{x}. \tag{24}$$

In the context of reconciliation methods, we need to consider a discrete version of the Kullback-Leibler divergence, yielding the following optimization problem:

$$\begin{aligned}&\min_{\mathbf{p}} \quad D_{\widehat{\mathbf{p}} \| \mathbf{p}} = \min_{p_i, i \in I} \sum_{i \in I} \widehat{p}_i \log\left(\frac{\widehat{p}_i}{p_i}\right)\\&\text{s.t.} \quad p_j \leq p_i, \quad (i, j) \in E.\end{aligned} \tag{25}$$

The algorithm finds the probabilities closest to the probabilities $\widehat{\mathbf{p}}$ obtained from logistic regression according to the Kullback-Leibler divergence and obeying the constraints that probabilities cannot increase while descending on the hierarchy underlying the ontology.

The extensive experiments exploited in [24] show that, among the reconciliation methods, isotonic regression is the most generally useful. Across a range of evaluation modes, term sizes, ontologies, and recall levels, isotonic regression yields consistently high precision. On the other hand, isotonic regression is not always the "best method," and a biologist with a particular goal in mind may apply other reconciliation methods. For instance, with small terms usually Kullback-Leibler projections achieve the best results, but considering average "per term" results heuristic methods yield precision at a given recall comparable with projection methods and better than that achieved with Bayes-net methods.

This ensemble approach achieved excellent results in the prediction of protein function in the mouse model organism, demonstrating that hierarchical multilabel methods can play a crucial role for the improvement of protein function prediction performances [24]. Nevertheless, the approach suffers from some drawbacks. Indeed, the paper focuses on the comparison of hierarchical multilabel methods, but it does not analyze impact of the concurrent use of data integration and hierarchical multilabel methods on the overall classification performances. Moreover, potential improvements could

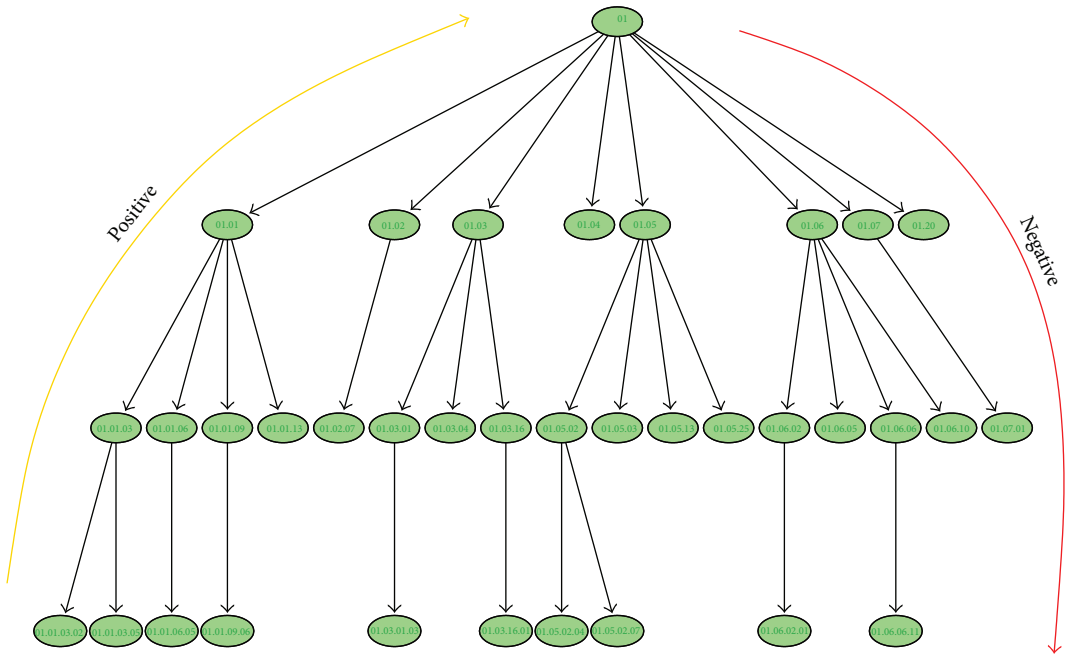


Figure 11: The asymmetric flow of information suggested by the true path rule.

be introduced by applying cost-sensitive variants of hierarchical multilabel predictors, able to effectively calibrate the precision/recall trade-off at different levels of the functional ontology.

7. True Path Rule Hierarchical Ensembles

These ensemble methods exploit at the same time the downward and upward relationships between classes, thus considering both the parent-to-child and child-to-parent functional links (Figure 2(b)).

The true path rule (TPR) ensemble method [31, 142] is directly inspired by the *true path rule* that governs both GO and FunCat taxonomies. Citing the curators of the Gene Ontology is as follows [143]: "An annotation for a class in the hierarchy is automatically transferred to its ancestors, while genes unannotated for a class cannot be annotated for its descendants." Considering the parents of a given node i, a classifier that respects the true path rule needs to obey the following rules:

$$\begin{aligned} y_i = 1 &\Longrightarrow y_{\mathrm{par}(i)} = 1 \\ y_i = 0 &\nRightarrow y_{\mathrm{par}(i)} = 0. \end{aligned} \tag{26}$$

On the other hand, considering the children of a given node i, a classifier that respects the true path rule needs to obey the following rules:

$$\begin{aligned} y_i = 1 &\nRightarrow y_{\mathrm{child}(i)} = 1 \\ y_i = 0 &\Longrightarrow y_{\mathrm{child}(i)} = 0. \end{aligned} \tag{27}$$

From (26) and (27), we observe an asymmetry in the rules that govern the assignments of positive and negative labels. Indeed, we have a propagation of positive predictions from bottom to top of the hierarchy in (26) and a propagation of negative labels from top to bottom in (27). Conversely, negative labels cannot propagate from bottom to top, and positive predictions cannot propagate from top to bottom.

The "true path rule" suggests algorithms able to propagate "positive" decisions from bottom to top of the hierarchy and negative decisions from top to bottom (Figure 11).

7.1. The True Path Rule Ensemble Algorithm. The TPR algorithm puts together the predictions made at each node by local "base" classifiers to realize an ensemble that obeys the "true path rule."

The basic ideas behind the method can be summarized as follows:

(1) training of the base learners: for each node of the hierarchy, a suitable learning algorithm (e.g., a multilayer perceptron or a support vector machine) provides a classifier for the associated functional class;

(2) in the evaluation phase, the trained classifiers associated with each class/node of the graph provide a local decision about the assignment of a given example to a given node;

(3) positive decisions, that is, annotations to a specific functional class, may propagate from bottom to top across the graph: they influence the decisions of the parent nodes and of their ancestors in a recursive way, by traversing the graph towards higher level nodes/classes. Conversely, negative decisions do not affect decisions of the parent node;that is, they do not propagate from bottom to top (26);

(4) negative predictions for a given node (taking into account the local decision of its descendants) are propagated to the descendants, to preserve the consistency of the hierarchy according to the true path rule, while positive decisions do not influence decisions of child nodes (27).

The ensemble combines the local predictions of the base learners associated with each node with the positive decisions that come from the bottom of the hierarchy, and with the negative decisions that spring from the higher level nodes. More precisely, base classifiers estimate local probabilities $\widehat{p}_i(g)$ that a given example g belongs to class θ_i, but the core of the algorithm is represented by the evaluation phase, where the ensemble provides an estimate of the "consensus" global probability $\overline{p}_i(g)$.

It is worth noting that instead of a probability, $\widehat{p}_i(g)$ may represent a score associated with the likelihood that a given gene/gene product belongs to the functional class i.

Let us consider the set $\phi_i(g)$ of the children of node i for which we have a positive prediction for a given gene g

$$\phi_i(g) = \left\{ j : j \in \text{child}(i), \widehat{y}_j = 1 \right\}. \tag{28}$$

The global consensus probability $\overline{p}_i(g)$ of the ensemble depends both on the local prediction $\widehat{p}_i(g)$ and on the prediction of the nodes belonging to $\phi_i(g)$

$$\overline{p}_i(g) = \frac{1}{1 + |\phi_i(g)|} \left(\widehat{p}_i(g) + \sum_{j \in \phi_i(g)} \overline{p}_j(g) \right). \tag{29}$$

The decision $\widehat{y}_i(g)$ at node/class i is set to 1 if $\overline{p}_i(g) > t$ and to 0, if otherwise (a natural choice for t is 0.5), and only children nodes for which we have a positive prediction can influence their parent. In the leaf nodes, the sum of (29) disappears and (29) becomes $\overline{p}_i(g) = \widehat{p}_i(g)$. In this way, positive predictions propagate from bottom to top, and negative decisions are propagated to their descendants when for a given node $\widehat{y}_i(g) = 0$.

The bottom-up per level traversal of the tree assures that all the offsprings of a given node i are taken into account for the ensemble prediction. For the same reason, we can safely set the classes belonging to the subtree rooted at i to negative, when $\widehat{y}_i$ is set to 0. It is worth noting that we have a two-way asymmetric flow of information across the tree: positive predictions for a node influence its ancestors, while negative predictions influence its offsprings.

The algorithm provides both the multilabels $\widehat{y}_i$ and an estimate of the probabilities $\overline{p}_i$ that a given example g belongs to the class $i = 1, \ldots, m$.

7.2. The Cost-Sensitive Variant. Note that in the TPR algorithm there is no way to explicitly balance the local prediction $\widehat{p}_i(g)$ at node i with the positive predictions coming from its offsprings (29). By balancing the local predictions with the positive predictions coming from the ensemble, we can explicitly modulate the interplay between local and descendant predictors. To this end, a *weight* w, $0 \leq w \leq 1$, is introduced, such that if $w = 1$ the decision at node i depends only by the local predictor; otherwise, the prediction is shared proportionally to w and $1 - w$ between, respectively, the local parent predictor and the set of its children

$$\overline{p}_i = w\widehat{p}_i + \frac{1-w}{|\phi_i|} \sum_{j \in \phi_i} \overline{p}_j. \tag{30}$$

This variant of the TPR algorithm is the *weighted true path rule* (TPR-W) hierarchical ensemble algorithm. By tuning the w parameter, we can modulate the precision/recall characteristics of the resulting ensemble. More precisely, for $w \to 0$, the weight of the parent local predictor is small, and the ensemble decision mainly depends on the positive predictions of the offsprings nodes (classifiers). Conversely, $w \to 1$ corresponds to a higher weight of the parent predictor; then, less weight is given to possible positive predictions of the children, and the decision depends mainly on the local/parent base classifier. In case of a negative decision, all the subtree is set to 0, causing the precision to increase. Note that for $w \to 1$ the behaviour of TPR-W becomes similar to that of HTD (Section 4).

A specific advantage of the TPR-W ensembles is the capability of tuning precision and recall rates, through the parameter w (30). For small values of w, the weight of the decision of the parent local predictor is small, and the ensemble decision depends mainly by the positive predictions of the offsprings nodes (classifiers), and higher values of w correspond to a higher weight of the "parent" local predictor, with a resulting higher precision. In [31], the author shows that the w parameter highly influences the precision/recall characteristics of the ensemble: low w values yield a higher recall, while high values improve the precision of the TPR-W ensemble.

Recently, Chen and Hu proposed a method that applies the TPR-W hierarchical strategy, but using composite kernel SVMs as base classifiers, and a supervised clustering with oversampling strategy to solve the imbalance data set learning problem, showed that the proper selection of base learners, and unbalance-aware learning strategies can further improve the results in terms of hierarchical precision and recall [144].

The same authors proposed also an enhanced version of the TPR-W strategy to overcome a limitation of this bottom-up hierarchical method for AFP. Indeed, for some classes at the lower levels of the hierarchy, the classifier performances are sometimes quite poor, due to both noisy data and the relatively low number of available annotations. More precisely, in the basic TPR ensemble, the probabilities $\overline{p}_j$ computed by the children of the node i (30) contribute in equal way to the probability $\overline{p}_i$ computed by the ensemble at node i, independently of the accuracy of the predictions made by its children classifiers. This "unweighted" mechanism may generate error propagation of the errors across the hierarchy: a poor performance child classifier may, for instance, with high probability, predict a negative example as positive and this error may propagate to its parent node and recursively to its ancestor nodes. To try to alleviate this possible bottom-up error propagation in [145], Chen and Hu proposed an improved TPR ensemble (TPR-W *weighted*), based on classifier performance. To this end, they weighted the contribution

of each child classifier on the basis of their performance evaluated on a validation data set, by adding to (30) another weight ν_j

$$\overline{p}_i = w\widehat{p}_i + \frac{1-w}{|\phi_i|}\sum_{j\in\phi_i}\nu_j \cdot \overline{p}_j, \tag{31}$$

where ν_j is computed on the basis of some accuracy metric A_j (e.g., the F-score) estimated for the child classifiers associated with node j as follows:

$$\nu_j = \frac{A_j}{\sum_{k\in\phi_i} A_k}. \tag{32}$$

In this way, the contribution of "poor" classifier is reduced, while "good" classifiers weight more in the final computation of $\overline{p}_i$ (31). Experiments with the "Protein Fate" subtree of the FunCat taxonomy with the yeast model organism show that this approach improves prediction with respect to the "vanilla" TPR-W hierarchical strategy [145].

7.3. Advantages and Drawbacks of TPR Methods. While the propagation of negative decisions from top to bottom nodes is quite straightforward and common to the hierarchical *top-down* algorithm, the propagation of positive decisions from bottom to top nodes of the hierarchy is specific to the TPR algorithm. For a discussion of this item, see Appendix C.

Experimental results show that TPR-W achieves equal or better results than the TPR and *top-down* hierarchical strategy, and both hierarchical strategies achieve significantly better results than *flat* classification methods [55, 123]. The analysis of the per level classification performances shows that TPR-W, by exploiting a global strategy of classification, is able to achieve a good compromise between precision and recall, enhancing the F-measure at each level of the taxonomy [31].

Another advantage of TPR-W consists in the possibility of tuning precision and recall by using a global strategy: large values of the w parameter improve the precision, and small values improve the recall.

Moreover, TPR and TPR-W ensembles provide also a probabilistic estimate of the prediction reliability for each functional class of the overall taxonomy.

The decisions performed at each node of the hierarchical ensemble are influenced by the positive decisions of its descendants. More precisely, the analyses performed in [31] showed the following:

(i) weights of descendants decrease exponentially with respect to their depth. As a consequence the influence of descendant, nodes decay quickly with their depth;

(ii) the parameter w plays a central role in balancing the weight of the parent classifier associated with a given node with the weights of its positive offsprings: small values of w increase the weight of descendant nodes, and large values increase the weight of the local parent predictor associated with that node;

(iii) the effect on the overall probability predicted by the ensemble is the result of the choice of the w parameter, the strength of the prediction of the local learners and of its descendants.

These characteristics of TPR-W ensembles are well suited for the hierarchical classification of protein functions, considering that annotations of deeper nodes are likely to have less experimental evidence than higher nodes. Moreover, by enforcing the strength of the descendant nodes through low w values, we can improve the recall characteristics of the overall system (at the expense of a possible reduction in precision).

Unfortunately, the method has been conceived and applied only to the FunCat taxonomy, structured according to a tree forest (Section 11), while no applications have been performed using the GO, structured according to a directed acyclic graph (Section 11).

8. Ensembles Based on Decision Trees

Another interesting research line is represented by hierarchical methods base on inductive decision trees [146]. The first attempts to exploit the hierarchical structure of functional ontologies for AFP simply used different decision tree models for each level of the hierarchy [147] or investigated a modified decision tree model, in which the assignment to a node is propagated toward the parent nodes [9], by extending the classical C4.5 decision tree algorithm for multiclass classification.

In the context of the predictive clustering tree framework [148], Blockeel et al. proposed an improved version which they applied to the prediction of gene function in the yeast [149].

More recent approaches, always based on modified decision trees, used distance measure derived from the hierarchy and significantly improved previous methods [54]. The authors showed that separate decision tree models are less accurate than a single decision tree trained to predict all classes at once, even when they are built taking into account the hierarchy.

Nevertheless, the previously proposed decision tree-base methods often achieve results not comparable with state-of-the-art hierarchical ensemble methods. To overcome this limitation, Schietgat et al. showed that ensembles of hierarchical multilabel decision trees are competitive with state-of-the-art statistical learning methods for DAG-structured prediction of protein function in *S. cerevisiae*, *A. thaliana,* and *M. musculus* model organisms [29]. A further work explored the suitability of different ensemble methods based on predictive clustering trees, ranging from global ensembles that learn ensembles of predictive models, each able to predict the entire structure of the hierarchy (i.e., all the GO terms for a given gene), to local ensembles that train an entire ensemble as a classifier for each branch of the taxonomy. Recently, a novel approach used PPI network autocorrelation in hierarchical multilabel classification trees to improve gene function prediction [150].

In [151], methods related to decision trees, in the sense that interpretable classification rules to predict all functions at all levels of the GO hierarchy, have been proposed, using an

ant colony optimization classification algorithm to discover classification rules.

Finally, bagging and random forest ensembles [152] have been applied to the AFP in yeast, showing that both local and global hierarchical ensemble approaches perform better than the single model counterparts in terms of predictive power [153].

9. The Hierarchical Classification Alone Is Not Enough

Several works showed that in protein function prediction problems we need to consider several learning issues [1, 16, 18]. In particular, in [80], the authors showed that even if hierarchical ensemble methods are fundamental to improve the accuracy of the predictions, their mere application is not enough to assure state-of-the-art results if we at the same time do not consider other important learning issues related to AFP. Indeed, in [123], it has been shown a significant synergy between hierarchical classification, data integration methods, and cost-sensitive techniques, highlighting that hierarchical ensemble methods should be designed taking into account different learning issues essential for the AFP problem.

9.1. Hierarchical Methods and Data Integration. Several works and the recently published results of the CAFA 2011 (Critical Assessment of Functional Annotation) challenge showed that data integration plays a central role to improve the predictions of protein functions [16, 25, 154–156].

Indeed, high-throughput biotechnologies make increasing quantities of biomolecular data of different types available, and several works pointed out that data integration is fundamental to improve the accuracy in AFP [1].

According to [154], we may subdivide the main approaches to data integration for AFP in four groups as follows:

(1) vector subspace integration;

(2) functional association networks integration;

(3) kernel fusion;

(4) ensemble methods.

Vector Space Integration. This approach consists in concatenating vectorial data to combine different sources of biomolecular data [132]. For instance, [22] concatenates different vectors, each one corresponding to a different source of genomic data, in order to obtain a larger vector that is used to train a standard SVM. A similar approach has been proposed by [30], but each data source is separately normalized in order to take into account the data distribution in each individual vector space.

Functional Association Networks Integration. In functional association networks, different graphs are combined to obtain the composite resulting network [21, 48]. The simplest approaches adopt conjunctive/disjunctive techniques [63], that is, respectively, adding an edge when in all the networks two genes are linked together or when a link between the two genes is present in at least one functional network or probabilistic evidence integration schemes [45].

Other methods differentially weight each data source using techniques ranging from Gaussian random fields [46] to the naive-Bayes integration [157] and constrained linear regression [14], or by merging data taking into account the GO hierarchy [158], or by applying XML-based techniques [159].

Kernel Fusion. These techniques at first construct a separated Gram matrix for each available data source using appropriate kernels representing similarities between genes/gene products. Then, by exploiting the closure property with respect to the sum and other algebraic operators, the Gram matrices are combined to obtain a "consensus" global integrated matrix.

Besides combining kernels linearly with fixed coefficients [22], one may also use semidefinite programming to learn the coefficients [11]. As methods based on semidefinite programming do not scale well to multiple data sources, more efficient methods for multiple kernel learning have been recently proposed [160, 161]. Kernel fusion methods, both with and without weighting the data sources, have been successfully applied to the classification of protein functions [162–165]. Recently, a novel method proposed an enhanced kernel integration approach by which the weights are iteratively optimized by reducing the empirical loss of a multilabel classifier for each of the labels simultaneously, using a combined objective function [165].

Ensemble Methods. Genomic data fusion can be realized by means of an ensemble system composed by learners trained on different "views" of the data and then combining the outputs of the component learners. Each type of data may capture different and complementary characteristics of the objects to be classified and the resulting ensemble may obtain better prediction capabilities through the diversity and the anticorrelation of the base learner responses.

Some examples of ensemble methods for data combination include "late integration" of kernels trained on different sources [22], the naive-Bayes integration [166] of the outputs of SVMs trained with multiple sources [30], and logistic regression for combining the output of several SVMs trained with different biomolecular data and kernels [24].

Recently, in [23], the authors showed that simple ensemble methods, such as weighted voting [167, 168] or decision templates [169], give results comparable to state-of-the-art data integration methods, exploiting at the same time the modularity and scalability that characterize most ensemble algorithms. Another work showed that ensemble methods are also resistant to noise [170].

Using an ensemble approach, biomolecular data differing in their structural characteristics (e.g., sequences, vectors, and graphs) can be easily integrated, because with ensemble methods the integration is performed at the decision level, combining the outputs produced by classifiers trained on different datasets [171–173].

As an example of the effectiveness of the integration of hierarchical ensemble methods with data fusion techniques,

Table 2: Comparison of the results (average per class F-scores) achieved with single sources and multisource (data fusion) techniques. FLAT, HTD, HTD-CS, HB (HBAYES), HB-CS (HBAYES-CS), TPR, and TPR-W ensemble methods are compared with and without data integration. In the last row, the number in parentheses refers to the percentage relative increment in F-score performance achieved with data fusion techniques with respect to the best single source of evidence (BIOGRID).

Methods	FLAT	HTD	HTD-CS	HB	HB-CS	TPR	TPR-W
				Single-source			
BIOGRID	0.2643	0.3759	0.4160	0.3385	0.4183	0.3902	0.4367
String	0.2203	0.2677	0.3135	0.2138	0.3007	0.2801	0.3048
PFAM BINARY	0.1756	0.2003	0.2482	0.1468	0.2407	0.2532	0.2738
PFAM LOGE	0.2044	0.1567	0.2541	0.0997	0.2847	0.3005	0.3160
Expr.	0.1884	0.2506	0.2889	0.2006	0.2781	0.2723	0.3053
Seq. sim.	0.1870	0.2532	0.2899	0.2017	0.2825	0.2742	0.3088
				Multisource (data fusion)			
Kernel fusion	0.3220(22)	0.5401(44)	0.5492(32)	0.5181(53)	0.5505(32)	0.5034(29)	0.5592(28)

in [123], six different sources of yeast biomolecular data have been integrated, ranging from protein domain data (PFAM BINARY and PFAM LOGE) [174], gene expression measures (EXPR) [175], predicted and experimentally supported protein-protein interaction data (STRING and BIOGRID) [176, 177] to pairwise sequence similarity data (SEQ. SIM.). Kernel fusion integration (sum of the Gram matrices) has been applied, and preprocessing has been performed using the the *HCGene* R package [178].

Table 2 summarizes the results of the comparison across about two hundreds of FunCat classes, including single-source and data integration approaches together with both flat and hierarchical ensembles.

Data fusion techniques improve average per class F-score across classes in flat ensembles (first column of Table 2) and significantly boost multilabel hierarchical methods (columns HTD, HTD-CS, HB, HB-CS, TPR, and TPR-W of Table 2).

Figure 12 depicts the classes (black nodes) where kernel fusion achieves better results than the best single-source data set (BIOGRID). It is worth noting that the number of black nodes is significantly larger in TPR-W (Figure 12(b)) with respect to FLAT methods (Figure 12(a)).

Hierarchical multilabel ensembles largely outperform FLAT approaches [24, 30], but Table 2 and Figure 12 also reveal a synergy between hierarchical ensemble methods and data fusion techniques.

9.2. Hierarchical Methods and Cost-Sensitive Techniques. According to [27, 142], cost-sensitive approaches boost predictions of hierarchical methods when single-sources of data are used to train the base learners. These results are confirmed when cost-sensitive methods (HBAYES-CS, Section 5.3.2; HTD-CS, Section 4; and TPR-W, Section 7.2) are integrated with data fusion techniques, showing a synergy between multilabel hierarchical, data fusion (in particular, kernel fusion), and cost-sensitive approaches (Figure 13) [123].

Perlevel analysis of the F-score in HBAYES-CS, HTD-CS, and TPR-W ensembles shows a certain degradation of performance with respect to the depth of nodes, but this degradation is significantly lower when data fusion is applied. Indeed, the per-level F-score achieved by HBAYES-CS and HTD-CS when a single source is used consistently decreases from the top to the bottom level, and it is halved at level 5 with respect to the first level. On the other hand, in the experiments with Kernel Fusion the average F-score at level 2, 3 and 4 is comparable, and the decrement at level 5 with respect to level 1 is only about 15% (Figure 14). Similar results are reported also with TPR-W ensembles.

In conclusion, the synergic effects of hierarchical multilabel ensembles, cost-sensitive, and data fusion techniques significantly improve the performance of AFP. Moreover, these enhancements allow obtaining better and more homogeneous results at each level of the hierarchy. This is of paramount importance, because more specific annotations are more informative and can get more biological insights into the functions of genes.

9.3. Different Strategies to Select "Negative" Genes. In both GO and FunCat, only positive annotations are usually available, while negative annotations are much reduced. More precisely, in the GO, only about 2500 negative annotations are available, and surely this amount does not allow a sufficient coverage of negative examples.

Moreover, some seminal works in functional genomics pointed out that the strategy of choosing negative training examples does affect the classifier performance [8, 163, 179, 180].

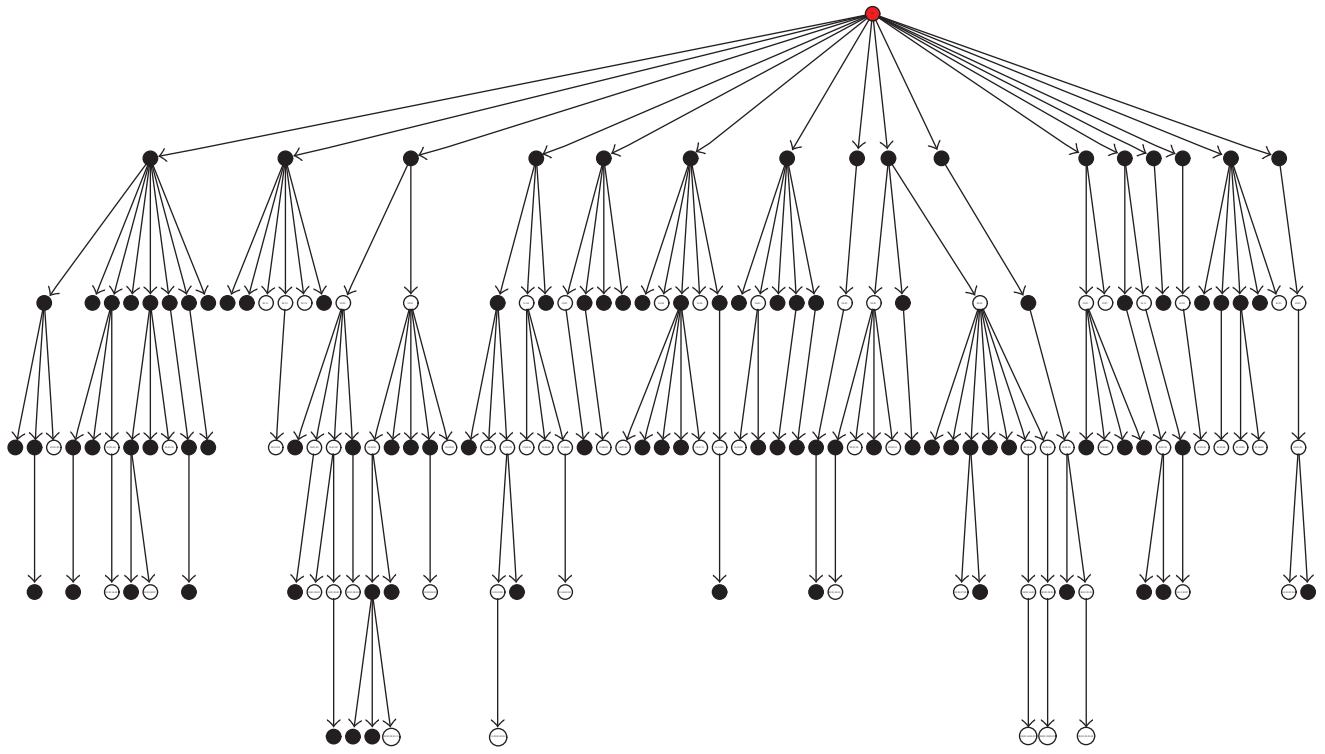
In [123], two strategies for choosing negative examples have been compared: the *basic* (B) and the *parent only* (PO) strategy.

According to the B strategy, the set of negative examples are simply those genes g that are not annotated for class c_i; that is,

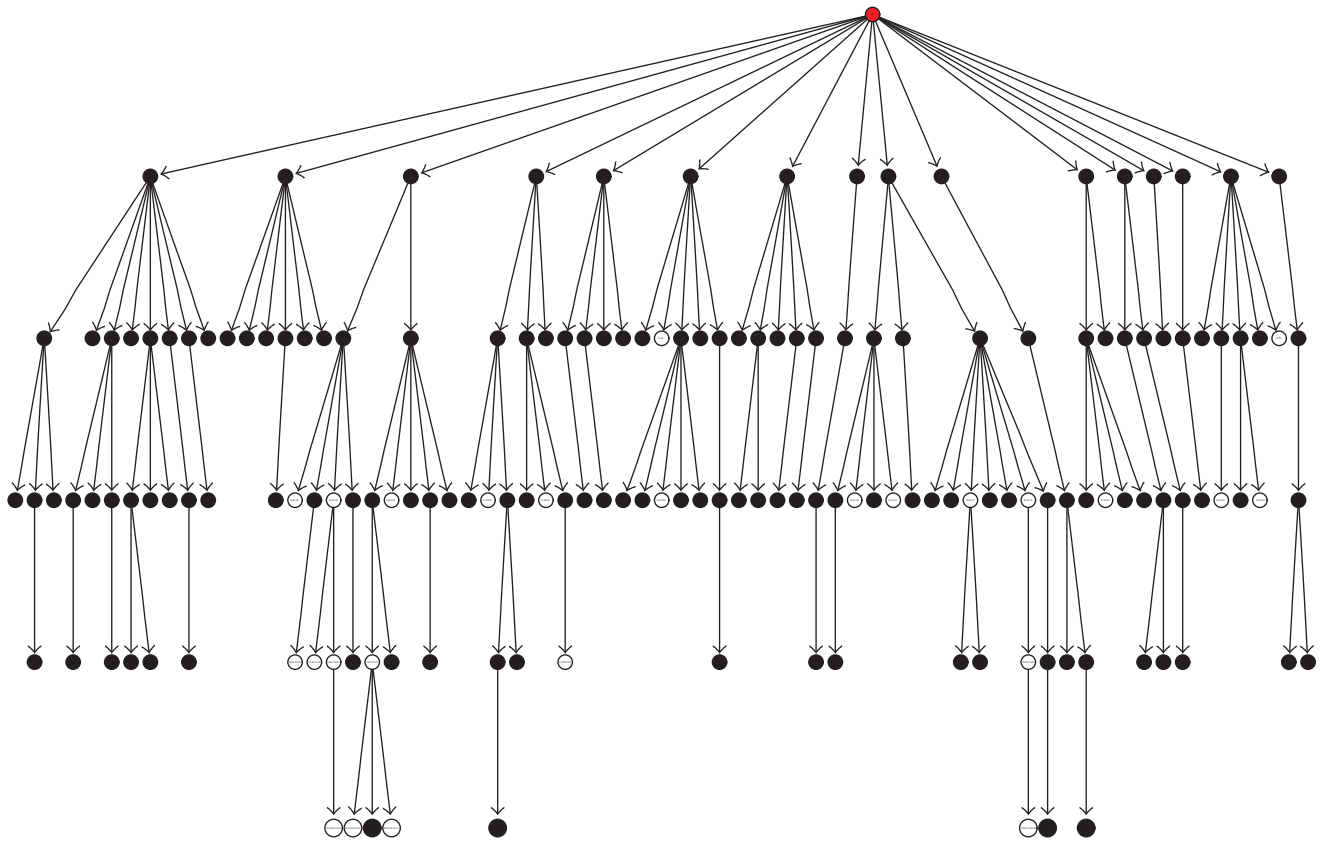
$$N_B = \{g : g \notin c_i\}. \tag{33}$$

The PO selection strategy chooses as negatives for the class c_i only those examples that are nonannotated to c_i but are annotated for a parent class. More precisely, for a given class c_i corresponding to node i in the taxonomy, the set of negative examples is

$$N_{\mathrm{PO}} = \{g : g \notin c_i, g \in \mathrm{par}(i)\}. \tag{34}$$



(a)



(b)

Figure 12: FunCat trees to compare F-scores achieved with data integration (KF) to the best single-source classifiers trained on BIOGRID data. Black nodes depict functional classes for which KF achieves better F-scores. (a) FLAT and (b) TPR-W ensembles.

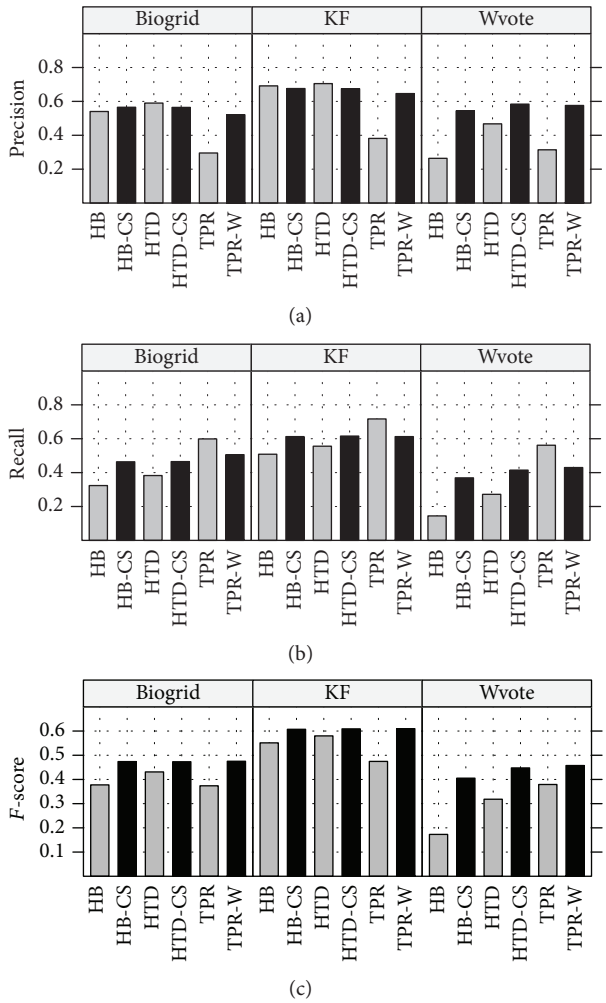


Figure 13: Comparison of hierarchical precision, recall, and F-score among different hierarchical ensemble methods using the best source of biomolecular data (BIOGRID), kernel fusion (KF), and weighted voting (WVOTE) data integration techniques. HB stands for HBAYES.

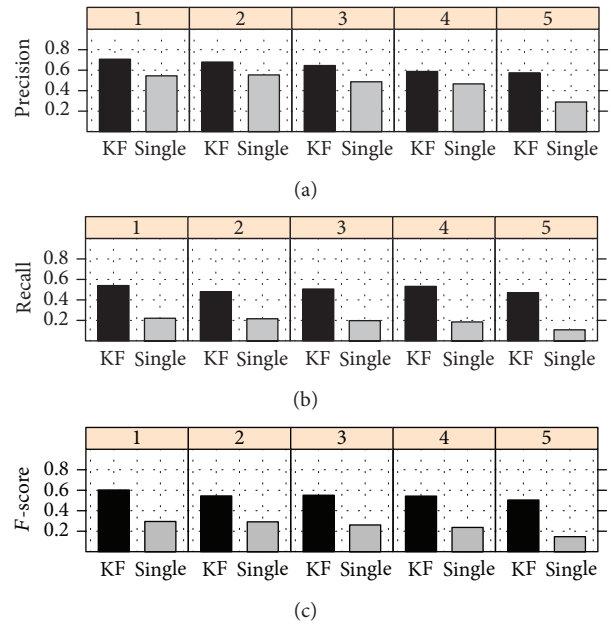


Figure 14: Comparison of per level average precision, recall, and F-score across the five levels of the FunCat taxonomy in HBAYES-CS using single data sets (single) and kernel fusion techniques (KF). Performance of "single" is computed by averaging across all the single data sources.

Hence, this strategy selects negative examples for training that are in a certain sense "close" to positives. It is easy to see that $N_{PO} \subseteq N_B$; hence, this strategy selects for training a large set of generic negative examples, possibly annotated with classes that are associated with faraway nodes in the taxonomy. Of course, the set of positive examples is the same for both strategies.

The B strategy worsens the performance of hierarchical multilabel methods, while for FLAT ensembles there is no clear trend. Indeed, in Figure 15, we compare the F-scores obtained with B to those obtained with PO, using both hierarchical cost-sensitive (Figure 15(a)) and FLAT (Figure 15(b)) methods. Each point represents the F-score for a specific FunCat class achieved by a specific method with B (abscissa) and PO (ordinate) strategy for the selection of negative examples. In Figure 15(a), most points lie above the bisector independently of the hierarchical cost-sensitive method being used. This shows that hierarchical methods gain in performance when using the PO strategy as opposed to the B strategy (P-value = 2.2×10^{-16} according to the Wilcoxon signed-ranks test). This is not the case for FLAT methods (Figure 15(b)).

These results can be explained by considering that the PO strategy takes into account the hierarchy to select negatives, while the B strategy does not. More precisely, FLAT methods having no information about the hierarchical structure of classes may fail to distinguish negative examples belonging to very distant classes, thus resulting in a high false positive rate, while hierarchical methods, which know the taxonomy, can use the information coming from other base classifiers to prevent a local base learner from incorrectly classifying "distant" negative examples.

In conclusion, these seminal works show that the strategy to choose negative examples exerts a significant impact on the accuracy of the predictions of hierarchical ensemble methods, and more research work is needed to explore this topic.

10. Open Problems and Future Trends

In the previous section, we showed that different learning issues should be considered to improve the effectiveness and the reliability of hierarchical ensemble methods. Most of these issues and others related to hierarchical ensemble methods and to AFP represent challenging problems that have been only partially considered by previous work. For these reasons, we try to delineate some of the open problem and research trends in the context of this research area.

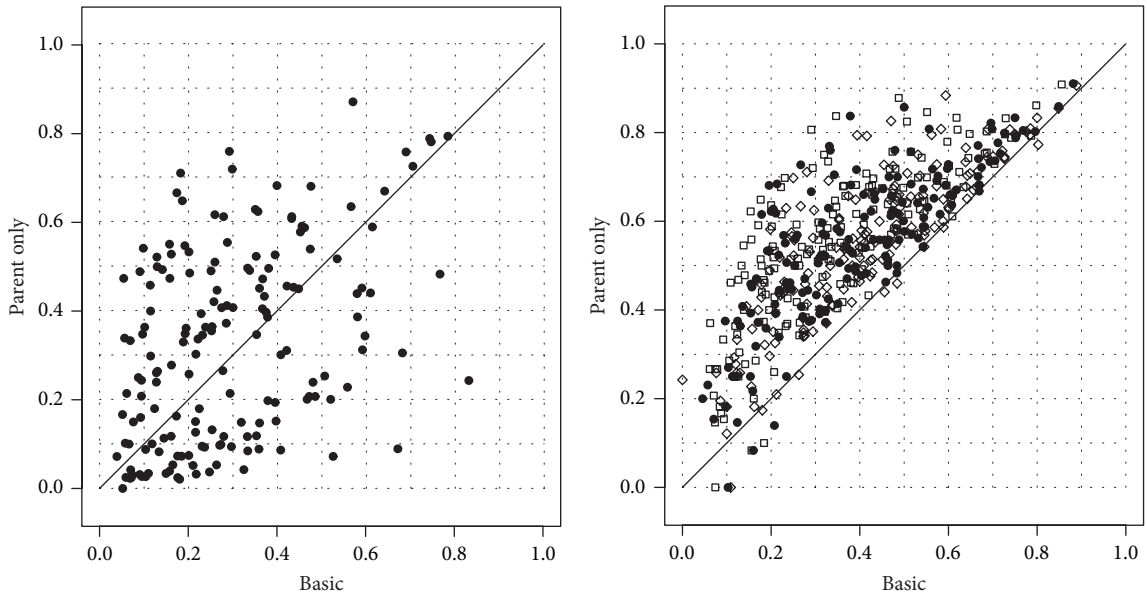


Figure 15: Comparison of average per class F-score between *basic* and PO strategies. (a) FLAT ensembles; (b) hierarchical cost-sensitive strategies: HTD-CS (squares), TPR-W (triangles), and HBAYES-CS (filled circles). Abscissa: per class F-score with base learners trained according to the *basic* strategy; ordinate: per class F-score with base learners trained according to the PO strategy.

For instance, the selection strategies for negative examples have been only partially explored, even if some seminal works show that this item exerts a significant impact on the accuracy of the predictions [123, 163, 179, 180]. Theoretical and experimental comparison of different strategies should be performed in a systematic way, to assess the impact of the different strategies on different hierarchical methods, considering also the characteristics of the learning machines used as base learners.

Some works showed also that the cost-sensitive strategies are needed to significantly improve predictions, especially in a hierarchical context [123], but new research could be considered for both applying and designing cost-sensitive base learners and to develop novel hierarchical ensemble unbalance-aware. Cost-sensitive methods have been applied to both the single base learners and also to the overall hierarchical ensemble strategy [31, 123], and recently a hierarchical variant of *SMOTE* (synthetic minority oversampling technique) [181] has been applied to hierarchical protein function prediction, showing very promising results [145]. In principle classical "balancing" strategies should be explored to improve the accuracy and the reliability of the base learners and hence of the overall hierarchical classification process. For instance, random undersampling or oversampling techniques could be applied: the former augments the annotations by exactly duplicating the annotated proteins, whereas the latter randomly takes away some unannotated examples [182]. Other approaches could be considered such as heuristic resampling methods [183] or embedding resampling methods into data mining algorithms [184] or ensemble methods tailored to imbalanced classification problems [185–187].

Since functional classes are unbalanced, precision/recall analysis plays a central role in AFP problems and often drives "in vitro" experiments that provide biological insights into specific functional genomics problems [1]. Only a few hierarchical ensemble methods, such as HBAYES-CS [27] and TPR-W [31], can tune their precision/recall characteristics through a single global parameter. In HBAYES-CS, by incrementing the cost factor $\alpha = \theta_i^- / \theta_i^+$, we introduce progressively lower costs for positive predictions, thus resulting in an increment of the recall (at the expenses of a possibly lower precision). In TPR-W, by incrementing w, we can reduce the recall and enhance the precision. Parametric versions of other hierarchical ensemble methods could be developed, in order to design ensemble methods with "tunable" precision/recall characteristics.

Another important issue that should be considered in the design of novel hierarchical ensemble methods is the incompleteness of the available annotations and its impact on the performance of computational methods for AFP. Indeed, the successful application of supervised and semisupervised machine learning methods to these tasks requires a gold standard for protein function, that is, a trusted set of correct examples, but unfortunately the annotations is incomplete and undergoes frequent updates, and also the GO is frequently updated. Some seminal works showed that, on the one hand, current machine learning approaches are able to generalize and predict novel biology from an incomplete gold standard and, on the other hand, incomplete functional annotations adversely affect the evaluation of machine learning performance [188]. A very recent work addressed these items by proposing methods based on *weak-label learning* specifically designed to replenish the functions of proteins under the assumption that proteins are partially annotated. More precisely, two new algorithms have been proposed: *ProWL*, protein function prediction with weak-label learning, which

can recover missing annotations by using the available *relevant* annotations, that is, a set of trusted annotations for a given protein, and *ProWL-IF*, protein function prediction with weak-label learning and knowledge of irrelevant function, by which also *irrelevant* functions, that is, functions that cannot be associated with the protein of interest, are exploited to replenish the missing functions [189, 190]. The results show that these items should be considered in future works for hierarchical multilabel predictions of protein functions in model organisms.

Another issue is represented by the reliability of the annotations. Usually, only experimental evidence is used to annotate the proteins for training AFP methods, but most of the available annotations are computationally predicted annotations without any experimental validation [191]. To at least partially exploit this huge amount of information, computational methods able to take into account the different reliability of the available annotations should be developed and integrated into hierarchical ensemble algorithms.

A quite neglected item is the interpretability of the hierarchical models. Nevertheless, the generation of comprehensible classification models is of paramount importance for biologists in order to provide new insights into the correlation of protein features and their functions [192]. A first step in this direction is represented by the work of Cerri et al. that exploits the advantages of grammar-based evolutionary algorithms to incorporate prior knowledge with the simplicity of genetic algorithms for optimization problems in order to produce interpretable rules for hierarchical multilabel classification [193].

Other issues depend on "strength" or the general rule that relates the predictions made by the base learner at a given term/node of the hierarchy with the predictions made by the other base learners of the hierarchical ensemble. For instance, the TPR algorithm (Section 7) weights the positive predictions of deeper nodes with an exponential decrement with respect to their depth (Section 11), but other rules (e.g., linear or polynomial) could be considered as the basis for the development of new algorithms that put more weight on the decisions of deep nodes of the hierarchy. Other enhancements could be introduced with the TPR-W algorithm (Section 7.2); indeed, we can note that positive children of a node at level i of the hierarchy have the same weight, independently of the size of their hanging subtree. In some cases, this could be useful, but in other cases it could be desirable to directly take into account the fact that a positive prediction is maintained along a path of the tree; indeed, this witnesses for a positive annotation of the node at level i.

More in general, in the spirit of a work recently proposed [123], the analysis of the synergy between the issues introduced above could be of great interest to better understand the behaviour of hierarchical ensemble methods.

Finally, we introduce some problems that could open new and interesting research lines in the context of hierarchical ensemble methods.

At first, an important issue could be represented by the design and development of multitask learning strategies [194] able to exploit the relationships between functional classes just during the learning phase, in order to establish a functional connection between learning processes associated with hierarchically related classes of the functional taxonomy. In this way, just during the training of the base learners, the learning processes will be dependent on each other (at least for nearby nodes/classes), enabling "mutual learning" of related classes in the taxonomy.

A second, to my knowledge, not explored learning issue is represented by the metaintegration of hierarchical predictions. Considering that there is no "killer" hierarchical ensemble method, a metacombination of the hierarchical predictions could be explored to enhance the overall performances.

A last issue is represented by multispecies predictions in a hierarchical context. By exploiting homology relationships between proteins of different species, we could enhance the prediction for a particular species by using predictions or data available for other species. This is a common practice with, for example, sequence-based methods, but novel research is needed to extend this homology-based approach in the context of hierarchical ensemble methods for multispecies prediction. It is worth noting that this multispecies approach yields to big-data analysis with the associated problems of scalability of existing algorithms. A possible solution to this last problem could be represented by distributed parallel computation [195] or by the adoption of secondary memory-based computational techniques [196].

11. Conclusions

Hierarchical ensemble methods represent one of the main research lines for AFP. Their two-step learning strategy introduces a high modularity in the prediction system: in the first step, different base learners can be trained to individually learn the functional classes, and in the second step different algorithms can be chosen to hierarchically combine the predictions provided by the base classifiers. The best results can be obtained when the global topology of the ontology is exploited and when both top-down and bottom-up learning strategies are applied [24, 27, 30, 31].

Nevertheless, a hierarchical learning strategy alone is not enough to achieve state-of-the-art results for AFP. Indeed, we need to design hierarchical ensemble methods in the context of the learning issues strictly related to the AFP problem.

The first one is represented by data fusion, since each source of biomolecular data may provide different and often complementary information about a protein and an integration of data fusion methods with hierarchical ensembles is mandatory to improve AFP results.

The second one is represented by the cost-sensitive techniques needed to take into account the usually small number of positive annotations: data unbalance-aware methods should be embedded in hierarchical methods to avoid solutions biased toward low sensitivity predictions.

Other issues, ranging from the proper choice of negative examples to the reliability and the incompleteness of the available annotation, the balance between local and global learning strategies, and the metaintegration of hierarchical predictions have been only partially addressed in previous

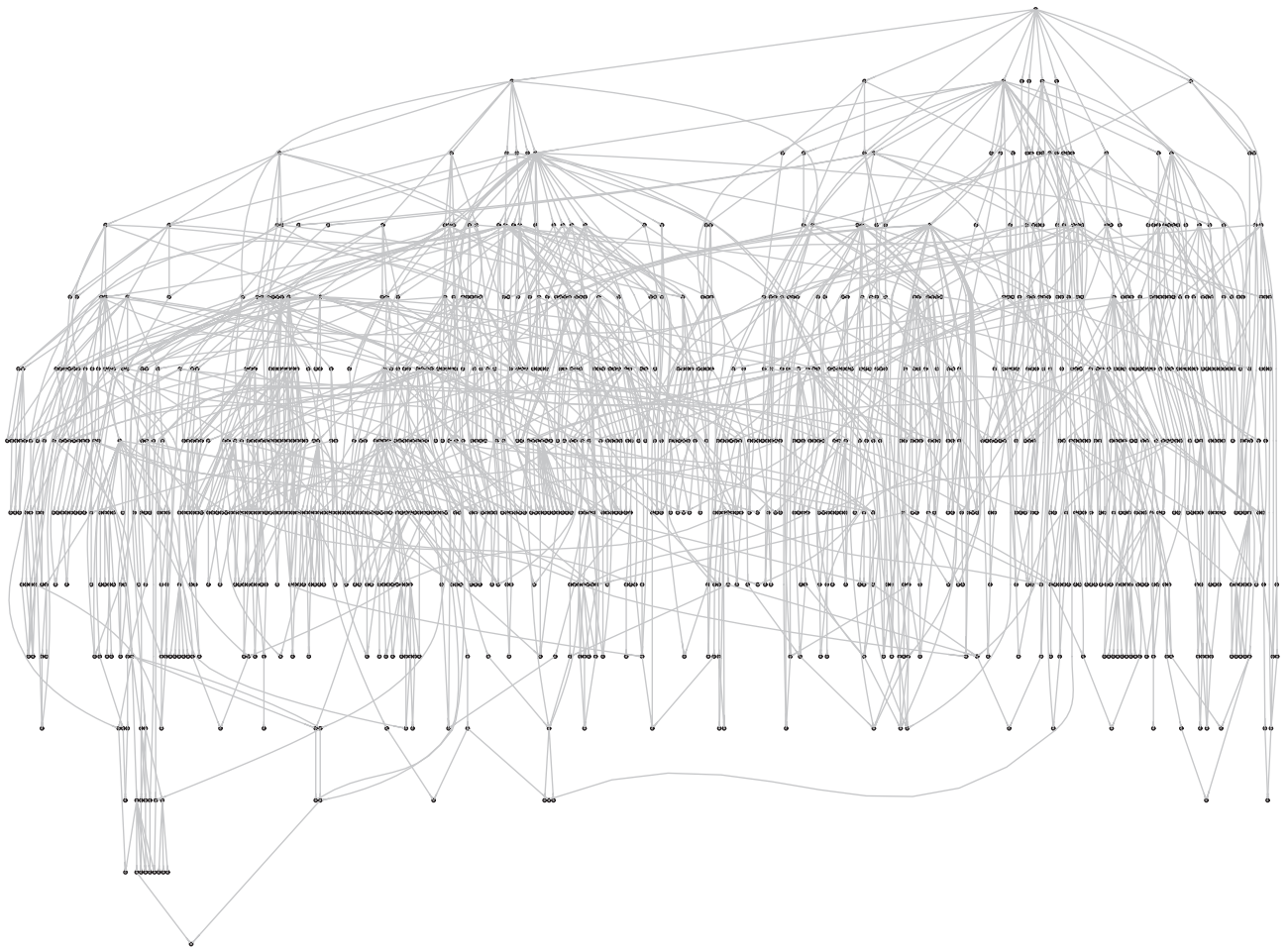


Figure 16: GO BP DAG for the yeast model organism (realized through the *HCGene* software [178]), involving more than 1000 terms and more than 2000 edges.

work. More in general, the synergy between hierarchical ensemble methods, data integration algorithms, cost-sensitive techniques, and other related issues is the key to improve AFP methods and to drive experiments aimed at discovering previously unannotated or partially annotated protein functions [123].

Indeed, despite their successful application to protein function prediction in different model organisms, as outlined in Section 9, there is large room for future research in this challenging area of computational biology.

In particular, the development of multitask learning methods to jointly learn related GO terms in a hierarchical context and the design of multispecies hierarchical algorithms, able to scale with millions of proteins, represent a compelling challenge for the computational biology and bioinformatics community.

Appendices

A. Gene Ontology and FunCat

The two main taxonomies of gene functional classes are represented by the Gene Ontology (GO) [6] and the functional catalogue (FunCat) [5]. In the former, the functional classes are structured according to a directed acyclic graph, that is, a DAG (Figure 16), while in the latter, the functional classes are structured through a forest of trees (Figure 17). The GO is composed of thousands of functional classes, and it is set out in three separated ontologies: "biological processes," "molecular function," and "cellular component". Indeed, a gene can participate to specific biological processes (e.g., cell cycle, metabolism, and nucleotide biosynthesis) and at the same time can perform specific molecular functions (e.g., catalytic or binding activities that occur at the molecular level) in specific cellular components (e.g., mitochondrion or rough endoplasmic reticulum).

A.1. GO: The Gene Ontology. The Gene Ontology (GO) project began as collaboration between three model organism databases, FlyBase (*Drosophila*), the *Saccharomyces* genome database (SGD), and the mouse genome database (MGD), in 1998. Now, it includes several of the world's major repositories for plant, animal, and microbial genomes. The GO project has developed three structured controlled vocabularies (ontologies) that describe gene products in terms of their associated biological processes, cellular components,

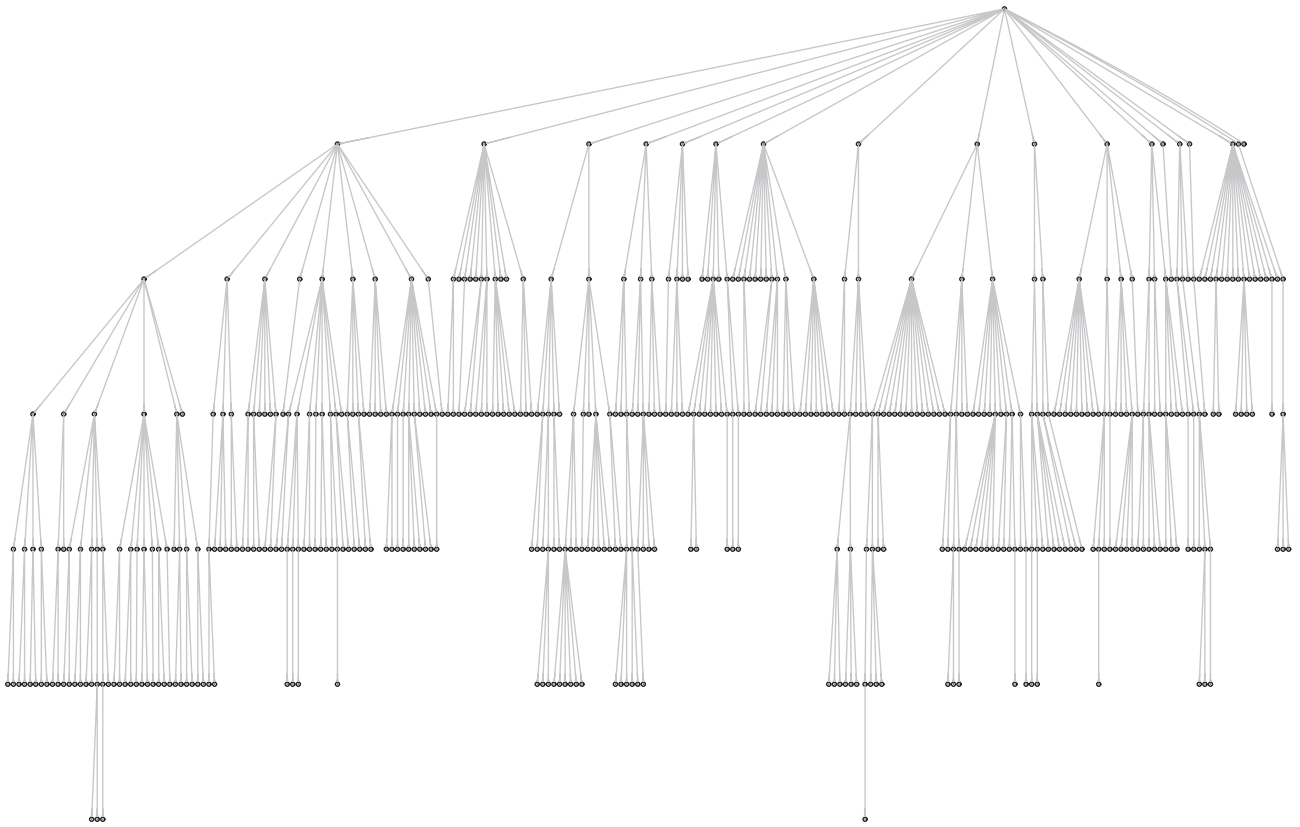


Figure 17: FunCat tree for the yeast model organism (realized through the *HCGene* software) [178]. A "dummy" root node has been added to obtain a single tree from the tree forest.

and molecular functions in a species-independent manner (Figure 16). Biological process (BP) represents series of events accomplished by one or more ordered assemblies of molecular functions that exploit a specific biological function. for instance, lipid metabolic process or tricarboxylic acid cycle. Molecular function (MF) describes activities that occur at the molecular level, such as catalytic or binding activities. An example of MF is glycine dehydrogenase activity or glucose transporter activity. Cellular component (CC) represents just parts or components of a cell, such as organelles or physical places or compartments in which a specific gene product is located. An example is the endoplasmic reticulum or the ribosome.

The ontologies of the GO are structured as a directed acyclic graph (DAG) $G = \langle V, E \rangle$, where $V = \{t \mid \text{terms of the GO}\}$ and $E = \{(t, u) \mid t, u \in V\}$. Relations between GO terms are also categorized in the following three main groups:

(i) *is-a* (subtype relations): if we say the term/node A is a B, we mean that node A is a subtype of node B. For example, mitotic cell cycle is a cell cycle, or lyase activity is a catalytic activity;

(ii) *part-of* (part-whole relations): A is part of B which means that whenever A exists, it is part of B, and the presence of A implies the presence of B. For instance, mitochondrion is part of cytoplasm;

(iii) *regulates* (control relations): if we say that A regulates B we mean that A directly affects the manifestation of B; that is, the former regulates the latter.

While *is-a* and *part-of* are transitive, "*regulates*" is not. Moreover, in some cases, regulatory proteins are not expected to have the same properties as the proteins they regulate and hence predicting regulation may require other data and assumptions than predicting function similarity. For these reasons, usually in AFP, *regulates* relations (that however are a minority of the existing relations) are usually not used.

Each annotation is labeled with an *evidence code* that indicates how the annotation to a particular term is supported. They are subdivided in several categories ranging from experimental evidence codes, used when experimental assays have been applied for the annotation, for example, inferred from physical interaction (IPI) or inferred from mutant phenotype (IMP) or inferred from genetic interaction (IGI), to author statement codes, such as traceable author statement (TAS), that indicate that the annotation was made on the basis of a statement made by the author(s) in the cited reference, to computational analysis evidence codes, based on an in silico analyses manually reviewed (e.g., inferred from sequence or structural similarity (ISS)). For the full set of available evidence codes, please see the GO website (http://www.geneontology.org/).

A GO graph for the yeast model organism is represented in Figure 16. It is worth noting that, despite the complexity of the represented graph, Figure 16 does not show all the available terms and the relationships involved in the GO BP ontology with the yeast model organism.

A.2. FunCat: The Functional Catalogue. The FunCat taxonomy started with *Saccharomyces cerevisiae* genome project at MIPS (http://mips.gsf.de/): at the beginning, FunCat contained only those categories required to describe yeast biology [197, 198], but successively its content has been extended to plants to annotate genes from the *Arabidopsis thaliana* genome project and furthermore to cover prokaryotic organisms and finally animals too [5].

The *FunCat* represents a relatively simple and concise set of gene functional classes: it consists of 28 main functional categories (or branches) that cover general fields like cellular transport, metabolism, and cellular communication/signal transduction. These main functional classes are divided into a set of subclasses with up to six levels of increasing specificity, according to a tree-like structure that accounts for different functional characteristics of genes and gene products. Genes may belong at the same time to multiple functional classes, since several classes are subclasses of more general ones, and because a gene may participate in different biological processes and may perform different biological functions.

Taking into account the broad and highly diverse spectrum of known protein functions, the FunCat annotation scheme covers general features, like cellular transport, metabolism, and protein activity regulation. Each of its main 28 functional branches is organized as a hierarchical, tree-like structure, thus leading to a tree forest with hundreds of functional categories.

Differently from the GO, the FunCat is more compact and does not intend to classify protein functions down to the most specific level. From a general standpoint, it can be compared to parts of the molecular function and biological process terms of the GO system.

One of the main advantages of FunCat is its intuitive category structure. For instance, the annotation of yeast uses only 18 of the main categories and less than 300 distinct categories (Figure 17), while the *Saccharomyces* genome database (SGD) [199] uses more than 1500 GO terms in its yeast annotation. Indeed, FunCat focuses on the functional process and in part on the molecular function, while GO aims at representing a fine granular description of proteins that provides annotations with a wealth of detailed information. However, to achieve this goal, the detailed description offered by GO leads to a large number of terms (e.g., the ontology for biological processes alone contains more than 10000 terms), and such a huge amount of terms is very difficult to be handled for annotators. Moreover, we may have a very large number of possible assignments that may lead to erroneous or inconsistent annotations. FunCat is simpler: its tree structure, compared with the DAG structure of the GO, leads to both simple procedures for annotation and less difficult computational-based classification tasks. In other words, it represents a well-balanced compromise between extensive depth, breadth, and resolution but without being too granular and specific.

It is worth noting that both FunCat and GO ontologies undergo modifications between different releases, and at the same time the annotations are also subjected to changes, since they represent the results of the knowledge of the scientific community at a given time. As a consequence, predictions resulting, for example, in false positives for a given release of the GO, may become true positive in future releases, and more in general we should keep in mind that the available annotations are always partial and incomplete and depend on the knowledge available for the species under study. Nevertheless, even if some works pointed out the inconsistency of current GO taxonomies through the analysis of violations of terms univocality [200], GO and FunCat are considered the ground truth to evaluate AFP methods, since they represent the main effort of the scientific community to organize commonly accepted taxonomy of protein functions [191].

B. AFP Performance Assessment in a Hierarchical Context

In the context of ontology-wide protein function prediction problems, where negative examples are usually a lot more than positive ones, accuracy is not a reliable measure to assess the classification performance. For this reason, the classical F-score is used instead, to take into account the unbalance of functional classes. If TP represents the positive examples correctly predicted as positive, FN, the positive examples incorrectly predicted as negative, and, FP, the negatives incorrectly predicted as positives, then the precision P, and the recall R are

$$P = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}} \qquad R = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}}. \tag{B.1}$$

The F-score F is the harmonic mean between precision and recall

$$F = \frac{2 \cdot P \cdot R}{P + R}. \tag{B.2}$$

If we need to evaluate the correct ranking of annotated proteins with respect to a specific functional class, a valuable measure is represented by the area under the receiving operating characteristic curve (AUC). A random ranking corresponds to AUC $\simeq 0.5$, while values close to 1 correspond to near optimal ranking; that is, AUC $= 1$, if all the annotated genes are ranked before the unannotated ones.

In order to better capture the hierarchical and sparse nature of the protein function prediction problem, we also need specific measures that estimate how far a predicted structured annotation is from the correct one. Indeed, functional classes are structured according to a direct acyclic graph (Gene Ontology) or to a tree (FunCat), and we need measures to accommodate not just "exact matches" but also "near misses" of different sorts.

For instance, correctly predicting a parent or ancestor annotation, while failing to predict the most specific available

annotation, should be "partially correct," in the sense that we can gain information about the more general functional characteristics of a gene, missing only its most specific functions.

More precisely, given a general taxonomy G representing the graph of the functional classes, for a given gene/gene product x, consider the graph $P(x) \subset G$ of the predicted classes and the graph $C(x)$ of the correct classes associated with x, and let $l(P)$ be the set of the leaves (nodes without children) of the graph P. Given a leaf $p \in P(x)$, let $\uparrow p$ be the set of ancestors of the node p that belong to $P(x)$, and, given a leaf $c \in C(x)$, let $\uparrow c$ be the set of ancestors of the node c that belong to $C(x)$; the hierarchical precision (HP), hierarchical recall (HR), and hierarchical F-score (HF) are defined as follows [201]:

$$\begin{aligned} \mathrm{HP} &= \frac{1}{|l(P(x))|} \sum_{p \in l(P(x))} \max_{c \in l(C(x))} \frac{|\uparrow c \cap \uparrow p|}{|\uparrow p|} \\ \mathrm{HR} &= \frac{1}{|l(C(x))|} \sum_{c \in l(C(x))} \max_{p \in l(P(x))} \frac{|\uparrow c \cap \uparrow p|}{|\uparrow c|} \\ \mathrm{HF} &= \frac{2 \cdot \mathrm{HP} \cdot \mathrm{HR}}{\mathrm{HP} + \mathrm{HR}}. \end{aligned} \tag{B.3}$$

In the case of the FunCat taxonomy, since it is structured as a tree, we can simplify HP, HR, and HF as follows:

$$\begin{aligned} \mathrm{HP} &= \frac{1}{|l(P(x))|} \sum_{p \in l(P(x))} \frac{|C(x) \cap \uparrow p|}{|\uparrow p|} \\ \mathrm{HR} &= \frac{1}{|l(C(x))|} \sum_{c \in l(C(x))} \frac{|\uparrow c \cap P(x)|}{|\uparrow c|} \\ \mathrm{HF} &= \frac{2 \cdot \mathrm{HP} \cdot \mathrm{HR}}{\mathrm{HP} + \mathrm{HR}}. \end{aligned} \tag{B.4}$$

An overall high hierarchical precision is indicating that the predictor is able to detect the most general functions of genes/gene products. On the other hand, a high average hierarchical recall indicates that the predictors are able to detect the most specific functions of the genes. The hierarchical F-measure expresses the correctness of the structured prediction of the functional classes, taking into account also partially correct paths in the overall hierarchical taxonomy, thus providing in a synthetic way the effectiveness of the structured hierarchical prediction.

Another variant of hierarchical classification measure is represented by the hierarchical F-measure proposed by Kiritchenko et al. [202]. Let $P(x)$ be the set of classes predicted in the overall hierarchy for a given gene/gene product x, and let $C(x)$ be the corresponding set of "true" classes. Then, the hierarchical precision HP_K and the hierarchical recall HR_K according to Kiritchenko are defined as

$$\mathrm{HP}_K = \sum_x \frac{|P(x) \cap C(x)|}{|P(x)|} \qquad \mathrm{HR}_K = \sum_x \frac{|P(x) \cap C(x)|}{|C(x)|}. \tag{B.5}$$

Note that these definitions do not explicitly consider the paths included in the predicted subgraphs but simply the ratio between the number of common classes and, respectively, the predicted (HP_K) and the true classes (HR_K). The hierarchical F-measure HF_K is the harmonic mean between HP_K and HR_K

$$\mathrm{HF}_K = \frac{2 \cdot \mathrm{HP}_K \cdot \mathrm{HR}_K}{\mathrm{HP}_K + \mathrm{HR}_K}. \tag{B.6}$$

C. Effect of the Propagation of the Positive Decisions in TPR Ensembles

In TPR ensembles, a *generic* node at level k is any node whose distance from the root is equal to k. The posterior probability computed by the ensemble for a generic node at level k is denoted by q_k. More precisely, q_k denotes the probability computed by the ensemble and $\widehat{q}_k$ denotes the probability computed by the base learner local to a node at level k. Moreover, we define q_{k+1}^j as the probability of a child j of a node at level k, where the index $j \geq 1$ refers to different children of a node at level k. From (30), we can derive the following expression for the probability q_k computed for a generic node at level k of the hierarchy [31]:

$$q_k(g) = w \cdot \widehat{q}_k(g) + \frac{1-w}{|\phi_k(g)|} \sum_{j \in \phi_k(g)} q_{k+1}^j(g). \tag{C.1}$$

To simplify the notation, we can introduce the following expression to indicate the average of the probabilities computed by the positive children nodes of a generic node at level k:

$$a_{k+1} = \frac{1}{|\phi_k|} \sum_{j \in \phi_k} \widehat{q}_{k+1}^j \tag{C.2}$$

and we can introduce similar notations for a_{k+2} (average of the probabilities of the grandchildren) and more in general for a_{k+j} (descendants at level j of a generic node). By extending these definitions across levels, we can obtain the following theorem.

Theorem 2 (influence of positive descendant nodes). *In a TPR-W ensemble, for a generic node at level k, with a given parameter w, $0 \leq w \leq 1$, balancing the weight between parent and children predictors, and having a variable number larger than or equal to 1 of positive descendants for each of the m lower levels below, the following equality holds for each $m \geq 1$:*

$$q_k = w\widehat{q}_k + \sum_{j=1}^{m-1} w(1-w)^j a_{k+j} + (1-w)^m a_{k+m}. \tag{C.3}$$

For the full proof, see [31].

Theorem 2 shows that the contribution of the descendant nodes decays exponentially with their depth and depends critically on the choice of the w parameter. To get more insights into the relationships between w and its effect on the influence of positive decisions on a generic node at level k

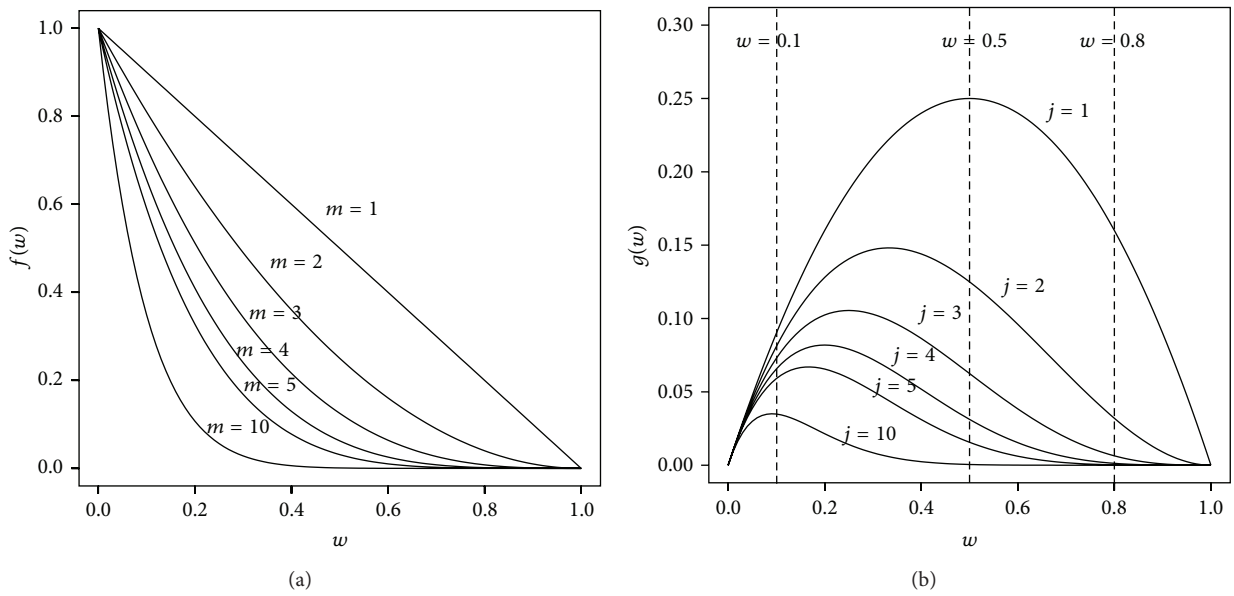


FIGURE 18: (a) Plot of $f(w) = (1 - w)^m$, while varying m from 1 to 10. (b) Plot of $g(w) = w(1 - w)^j$, while varying j from 1 to 10. The integers j refer to internal nodes at distance j from the reference node at level k.

(Theorem 1), Figure 18(a) shows the function that governs the decay of the influence of leaf nodes at different depths m, and Figure 18(b) shows the function responsible of the influence of nodes above the leaves.

Conflict of Interests

The author has no direct financial relations that might lead to a conflict of interests associated with this paper.

Acknowledgments

The author thanks the reviewers for their comments and suggestions and acknowledges partial support from the PRIN project "Automi e Linguaggi Formali: aspetti matematici e applicative," funded by the Italian Ministry of University.

References

[1] I. Friedberg, "Automated protein function prediction—the genomic challenge," *Briefings in Bioinformatics*, vol. 7, no. 3, pp. 225–242, 2006.

[2] L. Pena-Castillo, M. Tasan, C. L. Myers et al., "A critical assessment of Mus musculus gene function prediction using integrated genomic evidence," *Genome Biology*, vol. 9, supplement 1, article S2, 2008.

[3] G. Valentini, "Mosclust: a software library for discovering significant structures in bio-molecular data," *Bioinformatics*, vol. 23, no. 3, pp. 387–389, 2007.

[4] A. Bertoni and G. Valentini, "Discovering multi-level structures in bio-molecular data through the Bernstein inequality," *BMC Bioinformatics*, vol. 9, no. 2, article S4, 2008.

[5] A. Ruepp, A. Zollner, D. Maier et al., "The FunCat, a functional annotation scheme for systematic classification of proteins from whole genomes," *Nucleic Acids Research*, vol. 32, no. 18, pp. 5539–5545, 2004.

[6] M. Ashburner, C. A. Ball, J. A. Blake et al., "Gene ontology: tool for the unification of biology," *Nature Genetics*, vol. 25, no. 1, pp. 25–29, 2000.

[7] A. Valencia, "Automatic annotation of protein function," *Current Opinion in Structural Biology*, vol. 15, no. 3, pp. 267–274, 2005.

[8] N. Youngs, D. Penfold-Brown, K. Drew, D. Shasha, and R. Bonneau, "Parametric bayesian priors and better choice of negative examples improve protein function prediction," *Bioinformatics*, vol. 29, no. 9, pp. 1190–1198, 2013.

[9] A. Clare and R. D. King, "Predicting gene function in *Saccharomyces cerevisiae*," *Bioinformatics*, vol. 19, no. 2, pp. II42–II49, 2003.

[10] Y. Bilu and M. Linial, "The advantage of functional prediction based on clustering of yeast genes and its correlation with non-sequence based classifications," *Journal of Computational Biology*, vol. 9, no. 2, pp. 193–210, 2002.

[11] G. R. G. Lanckriet, T. De Bie, N. Cristianini, M. I. Jordan, and S. Noble, "A statistical framework for genomic data fusion," *Bioinformatics*, vol. 20, no. 16, pp. 2626–2635, 2004.

[12] W. Tian, L. V. Zhang, M. Taşan et al., "Combining guilt-by-association and guilt-by-profiling to predict Saccharomyces cerevisiae gene function," *Genome Biology*, vol. 9, no. 1, article S7, 2008.

[13] W. K. Kim, C. Krumpelman, and E. M. Marcotte, "Inferring mouse gene functions from genomic-scale data using a combined functional network/classification strategy," *Genome Biology*, vol. 9, no. 1, article S5, 2008.

[14] S. Mostafavi, D. Ray, D. Warde-Farley, C. Grouios, and Q. Morris, "GeneMANIA: a real-time multiple association network integration algorithm for predicting gene function," *Genome Biology*, vol. 9, no. 1, article S4, 2008.

[15] I. Lee, B. Ambaru, P. Thakkar, E. M. Marcotte, and S. Y. Rhee, "Rational association of genes with traits using a genome-scale

gene network for Arabidopsis thaliana," *Nature Biotechnology*, vol. 28, no. 2, pp. 149–156, 2010.

[16] P. Radivojac, W. T. Clark, T. R. Oron et al., "A large-scale evaluation of computational protein function prediction," *Nature Methods*, vol. 10, no. 3, pp. 221–227, 2013.

[17] A. S. Juncker, L. J. Jensen, A. Pierleoni et al., "Sequence-based feature prediction and annotation of proteins," *Genome Biology*, vol. 10, no. 2, p. 206, 2009.

[18] R. Sharan, I. Ulitsky, and R. Shamir, "Network-based prediction of protein function," *Molecular Systems Biology*, vol. 3, p. 88, 2007.

[19] A. Sokolov and A. Ben-Hur, "Hierarchical classification of gene ontology terms using the GOstruct method," *Journal of Bioinformatics and Computational Biology*, vol. 8, no. 2, pp. 357–376, 2010.

[20] Z. Barutcuoglu, R. E. Schapire, and O. G. Troyanskaya, "Hierarchical multi-label prediction of gene function," *Bioinformatics*, vol. 22, no. 7, pp. 830–836, 2006.

[21] H. N. Chua, W.-K. Sung, and L. Wong, "An efficient strategy for extensive integration of diverse biological data for protein function prediction," *Bioinformatics*, vol. 23, no. 24, pp. 3364–3373, 2007.

[22] P. Pavlidis, J. Weston, J. Cai, and W. S. Noble, "Learning gene functional classifications from multiple data types," *Journal of Computational Biology*, vol. 9, no. 2, pp. 401–411, 2002.

[23] M. Re and G. Valentini, "Simple ensemble methods are competitive with stateof- the-art data integration methods for gene function prediction," *Journal of Machine Learning Research, W&C Proceedings, Machine Learning in Systems Biology*, vol. 8, pp. 98–111, 2010.

[24] G. Obozinski, G. Lanckriet, C. Grant, M. I. Jordan, and W. S. Noble, "Consistent probabilistic outputs for protein function prediction," *Genome Biology*, vol. 9, no. 1, article S6, 2008.

[25] D. Cozzetto, D. Buchan, K. Bryson, and D. Jones, "Protein function prediction by massive integration of evolutionary analyses and multiple data sources," *BMC Bioinformatics*, vol. 14, supplement 3, article S1, 2013.

[26] X. Jiang, N. Nariai, M. Steffen, S. Kasif, and E. D. Kolaczyk, "Integration of relational and hierarchical network information for protein function prediction," *BMC Bioinformatics*, vol. 9, article 350, 2008.

[27] N. Cesa-Bianchi and G. Valentini, "Hierarchical cost-sensitive algorithms for genome-wide gene function prediction," *Journal of Machine Learning Research, W&C Proceedings, Machine Learning in Systems Biology*, vol. 8, pp. 14–29, 2010.

[28] R. Cerri and A. C. P. L. F. De Carvalho, "New top-down methods using SVMs for hierarchical multilabel classification problems," in *Proceedings of the International Joint Conference on Neural Networks (IJCNN '10)*, pp. 1–8, IEEE Computer Society, July 2010.

[29] L. Schietgat, C. Vens, J. Struyf, H. Blockeel, D. Kocev, and S. Džeroski, "Predicting gene function using hierarchical multi-label decision tree ensembles," *BMC Bioinformatics*, vol. 11, article 2, 2010.

[30] Y. Guan, C. L. Myers, D. C. Hess, Z. Barutcuoglu, A. A. Caudy, and O. G. Troyanskaya, "Predicting gene function in a hierarchical context with an ensemble of classifiers," *Genome Biology*, vol. 9, no. 1, article S3, 2008.

[31] G. Valentini, "True path rule hierarchical ensembles for genome-wide gene function prediction," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 8, no. 3, pp. 832–847, 2011.

[32] N. Alaydie, C. K. Reddy, and F. Fotouhi, "Exploiting label dependency for hierarchical multi-label classification," in *Advances in Knowledge Discovery and Data Mining*, vol. 7301 of *Lecture Notes in Computer Science*, pp. 294–305, Springer, 2012.

[33] D. Koller and M. Sahami, "Hierarchically classifying documents using very few words," in *Proceedings of the 14th International Conference on Machine Learning*, pp. 170–178, 1997.

[34] S. Chakrabarti, B. Dom, R. Agrawal, and P. Raghavan, "Scalable feature selection, classification and signature generation for organizing large text databases into hierarchical topic taxonomies," *VLDB Journal*, vol. 7, no. 3, pp. 163–178, 1998.

[35] M.-L. Zhang and Z.-H. Zhou, "Multilabel neural networks with applications to functional genomics and text categorization," *IEEE Transactions on Knowledge and Data Engineering*, vol. 18, no. 10, pp. 1338–1351, 2006.

[36] A. Burred and J. Lerch, "A hierarchical approach to automatic musical genre classification," in *Proceedings of the 6th International Conference on Digital Audio Effects*, pp. 8–11, 2003.

[37] C. DeCoro, Z. Barutcuoglu, and R. Fiebrink, "Bayesian aggregation for hierarchical genre classification," in *Proceedings of the 8th International Conference on Music Information Retrieval*, pp. 77–80, 2007.

[38] K. Trohidis, G. Tsoumahas, G. Kalliris, and I. P. Vlahavas, "Multilabel classification of music into emotions," in *Proceedings of the 9th International Conference on Music Information Retrieval*, pp. 325–330, 2008.

[39] A. Binder, M. Kawanabe, and U. Brefeld, "Bayesian aggregation for hierarchical genre classification," in *Proceedings of the 9th Asian Conference on Computer Vision*, 2009.

[40] Z. Barutcuoglu and C. DeCoro, "Hierarchical shape classification using Bayesian aggregation," in *Proceedings of the IEEE International Conference on Shape Modeling and Applications (SMI '06)*, p. 44, June 2006.

[41] A. Dimou, G. Tsoumakas, V. Mezaris, I. Kompatsiaris, and I. Vlahavas, "An empirical study of multi-label learning methods for video annotation," in *Proceedings of the 7th International Workshop on Content-Based Multimedia Indexing (CBMI '09)*, pp. 19–24, Chania, Greece, June 2009.

[42] K. Punera and J. Ghosh, "Enhanced hierarchical classification via isotonic smoothing," in *Proceedings of the 17th International Conference on World Wide Web (WWW '08)*, pp. 151–160, ACM, Beijing, China, April 2008.

[43] M. Ceci and D. Malerba, "Classifying web documents in a hierarchy of categories: a comprehensive study," *Journal of Intelligent Information Systems*, vol. 28, no. 1, pp. 37–78, 2007.

[44] C. N. Silla Jr. and A. A. Freitas, "A survey of hierarchical classification across different application domains," *Data Mining and Knowledge Discovery*, vol. 22, no. 1-2, pp. 31–72, 2011.

[45] O. G. Troyanskaya, K. Dolinski, A. B. Owen, R. B. Altman, and D. Botstein, "A Bayesian framework for combining heterogeneous data sources for gene function prediction (in *Saccharomyces cerevisiae*)," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 100, no. 14, pp. 8348–8353, 2003.

[46] K. Tsuda, H. Shin, and B. Schölkopf, "Fast protein classification with multiple networks," *Bioinformatics*, vol. 21, supplement 2, pp. ii59–ii65, 2005.

[47] J. Xiong, S. Rayner, K. Luo, Y. Li, and S. Chen, "Genome wide prediction of protein function via a generic knowledge discovery approach based on evidence integration," *BMC Bioinformatics*, vol. 7, article 268, 2006.

[48] U. Karaoz, T. M. Murali, S. Letovsky et al., "Whole-genome annotation by using evidence integration in functional-linkage networks," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 101, no. 9, pp. 2888–2893, 2004.

[49] A. Sokolov and A. Ben-Hur, "A structured-outputs method for prediction of protein function," in *Proceedings of the 2nd International Workshop on Machine Learning in Systems Biology (MLSB '08)*, 2008.

[50] K. Astikainen, L. Holm, E. Pitkanen, S. Szedmak, and J. Rousu, "Towards structured output prediction of enzyme function," *BMC Proceedings*, vol. 2, supplement 4, article S2, 2008.

[51] B. Done, P. Khatri, A. Done, and S. Drăghici, "Predicting novel human gene ontology annotations using semantic analysis," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 7, no. 1, pp. 91–99, 2010.

[52] G. Valentini and F. Masulli, "Ensembles of learning machines," in *Neural Nets WIRN-02*, vol. 2486 of *Lecture Notes in Computer Science*, pp. 3–19, Springer, 2002.

[53] B. Shahbaba and R. M. Neal, "Gene function classification using Bayesian models with hierarchy-based priors," *BMC Bioinformatics*, vol. 7, article 448, 2006.

[54] C. Vens, J. Struyf, L. Schietgat, S. Džeroski, and H. Blockeel, "Decision trees for hierarchical multi-label classification," *Machine Learning*, vol. 73, no. 2, pp. 185–214, 2008.

[55] G. Valentini, "True path rule hierarchical ensembles," in *Multiple Classifier Systems. Eighth InternationalWorkshop, MCS, 2009, Reykjavik, Iceland*, J. Kittler, J. Benediktsson, and F. Roli, Eds., vol. 5519 of *Lecture Notes in Computer Science*, pp. 232–241, Springer, 2009.

[56] S. F. Altschul, W. Gish, W. Miller, E. W. Myers, and D. J. Lipman, "Basic local alignment search tool," *Journal of Molecular Biology*, vol. 215, no. 3, pp. 403–410, 1990.

[57] S. F. Altschul, T. L. Madden, A. A. Schäffer et al., "Gapped blast and psi-blast: a new generation of protein database search programs," *Nucleic Acids Research*, vol. 25, no. 17, pp. 3389–3402, 1997.

[58] A. Conesa, S. Götz, J. M. García-Gómez, J. Terol, M. Talón, and M. Robles, "Blast2GO: a universal tool for annotation, visualization and analysis in functional genomics research," *Bioinformatics*, vol. 21, no. 18, pp. 3674–3676, 2005.

[59] Y. Loewenstein, D. Raimondo, O. C. Redfern et al., "Protein function annotation by homology-based inference," *Genome biology*, vol. 10, no. 2, p. 207, 2009.

[60] A. Prlić, T. A. Down, E. Kulesha, R. D. Finn, A. Kähäri, and T. J. P. Hubbard, "Integrating sequence and structural biology with DAS," *BMC Bioinformatics*, vol. 8, article 333, 2007.

[61] M. Falda, S. Toppo, A. Pescarolo et al., "Argot2: a large scale function prediction tool relying on semantic similarity of weighted Gene Ontology terms," *BMC Bioinformatics*, vol. 13, supplement 4, article S14, 2012.

[62] T. Hamp, R. Kassner, S. Seemayer et al., "Homology-based inference sets the bar high for protein function prediction," *BMC Bioinformatics*, vol. 14, supplement 3, article S7, 2013.

[63] E. M. Marcotte, M. Pellegrini, M. J. Thompson, T. O. Yeates, and D. Eisenberg, "A combined algorithm for genome-wide prediction of protein function," *Nature*, vol. 402, no. 6757, pp. 83–86, 1999.

[64] A. Vazquez, A. Flammini, A. Maritan, and A. Vespignani, "Global protein function prediction from protein-protein interaction networks," *Nature Biotechnology*, vol. 21, no. 6, pp. 697–700, 2003.

[65] J. Z. Wang, R. Du, R. S. Payattakil, P. S. Yu, and C. F. Chen, "Improving GO semantic similarity measures by exploring the ontology beneath the terms and modelling uncertainty," *Bioinformatics*, vol. 23, no. 10, pp. 1274–1281, 2007.

[66] H. Yang, T. Nepusz, and A. Paccanaro, "Improving GO semantic similarity measures by exploring the ontology beneath the terms and modelling uncertainty," *Bioinformatics*, vol. 28, no. 10, pp. 1383–1389, 2012.

[67] H. Yu, L. Gao, K. Tu, and Z. Guo, "Broadly predicting specific gene functions with expression similarity and taxonomy similarity," *Gene*, vol. 352, no. 1-2, pp. 75–81, 2005.

[68] Y. Tao, L. Sam, J. Li, C. Friedman, and Y. A. Lussier, "Information theory applied to the sparse gene ontology annotation network to predict novel gene function," *Bioinformatics*, vol. 23, no. 13, pp. i529–i538, 2007.

[69] G. Pandey, C. L. Myers, and V. Kumar, "Incorporating functional inter-relationships into protein function prediction algorithms," *BMC Bioinformatics*, vol. 10, article 142, 2009.

[70] X.-F. Zhang and D.-Q. Dai, "A framework for incorporating functional interrelationships into protein function prediction algorithms," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 9, no. 3, pp. 740–753, 2012.

[71] E. Nabieva, K. Jim, A. Agarwal, B. Chazelle, and M. Singh, "Whole-proteome prediction of protein function via graph-theoretic analysis of interaction maps," *Bioinformatics*, vol. 21, no. 1, pp. 302–310, 2005.

[72] A. Bertoni, M. Frasca, and G. Valentini, "COSNet: a cost sensitive neural network for semi-supervised learning in graphs," in *European Conference on Machine Learning, ECML PKDD 2011*, vol. 6911 of *Lecture Notes on Artificial Intelligence*, pp. 219–234, Springer, 2011.

[73] M. Frasca, A. Bertoni, M. Re, and G. Valentini, "A neural network algorithm for semi-supervised node label learning from unbalanced data," *Neural Networks*, vol. 43, pp. 84–98, 2013.

[74] M. Deng, T. Chen, and F. Sun, "An integrated probabilistic model for functional prediction of proteins," *Journal of Computational Biology*, vol. 11, no. 2-3, pp. 463–475, 2004.

[75] Y. A. I. Kourmpetis, A. D. J. Van Dijk, M. C. A. M. Bink, R. C. H. J. Van Ham, and C. J. F. Ter Braak, "Bayesian markov random field analysis for protein function prediction based on network data," *PLoS ONE*, vol. 5, no. 2, Article ID e9293, 2010.

[76] S. Oliver, "Guilt-by-association goes global," *Nature*, vol. 403, no. 6770, pp. 601–603, 2000.

[77] J. McDermott, R. Bumgarner, and R. Samudrala, "Functional annotation from predicted protein interaction networks," *Bioinformatics*, vol. 21, no. 15, pp. 3217–3226, 2005.

[78] M. Re, M. Mesiti, and G. Valentini, "A fast ranking algorithm for predicting gene functions in biomolecular networks," *IEEE ACM Transactions on Computational Biology and Bioinformatics*, vol. 9, no. 6, pp. 1812–1818, 2012.

[79] M. Re and G. Valentini, "Cancer module genes ranking using kernelized score functions," *BMC Bioinformatics*, vol. 13, supplement 14, article S3, 2012.

[80] M. Re and G. Valentini, "Large scale ranking and repositioning of drugs with respect to DrugBank therapeutic categories," in *International Symposium on Bioinformatics Research and Applications (ISBRA 2012)*, vol. 7292 of *Lecture Notes in Computer Science*, pp. 225–236, Springer, 2012.

[81] M. Re and G. Valentini, "Network-based drug ranking and repositioning with respect to drugbank therapeutic categories,"

IEEE/ACM Transactions on Computational Biology and Bioinformatics, vol. 10, no. 6, pp. 1359–1371, 2014.

[82] Y. Bengio, O. Delalleau, and N. Le Roux, "Label propagation and quadratic criterion," in *Semi-Supervised Learning*, O. Chapelle, B. Scholkopf, and A. Zien, Eds., pp. 193–216, MIT Press, 2006.

[83] Y. Saad, *Iterative Methods For Sparse Linear Systems*, PWS Publishing Company, Boston, Mass, USA, 1996.

[84] N. Cesa-Bianchi, C. Gentile, F. Vitale, and G. Zappella, "Random spanning trees and the prediction of weighted graphs," in *Proceedings of the 27th International Conference on Machine Learning (ICML '10)*, pp. 175–182, Haifa, Israel, June 2010.

[85] I. Tsochantaridis, T. Joachims, T. Hofmann, and Y. Altun, "Large margin methods for structured and interdependent output variables," *Journal of Machine Learning Research*, vol. 6, pp. 1453–1484, 2005.

[86] J. Rousu, C. Saunders, S. Szedmak, and J. Shawe-Taylor, "Kernel-based learning of hierarchical multilabel classification models," *Journal of Machine Learning Research*, vol. 7, pp. 1601–1626, 2006.

[87] C. H. Lampert and M. B. Blaschko, "Structured prediction by joint kernel support estimation," *Machine Learning*, vol. 77, no. 2-3, pp. 249–269, 2009.

[88] G. Bakir, T. Hoffman, B. Scholkopf, B. Smola, A. J. Taskar, and S. V. N. Vishwanathan, *Predicting Structured Data*, MIT Press, Cambridge, Mass, USA, 2007.

[89] R. Eisner, B. Poulin, D. Szafron, P. Lu, and R. Greiner, "Improving protein function prediction using the hierarchical structure of the gene ontology," in *Proceedings of the IEEE Symposium on Computational Intelligence in Bioinformatics and Computational Biology (CIBCB '05)*, November 2005.

[90] H. Blockeel, L. Schietgat, and A. Clare, "Hierarchical multilabel classification trees for gene function prediction," in *Probabilistic Modeling and Machine Learning in Structural and Systems Biology*, J. Rousu, S. Kaski, and E. Ukkonen, Eds., Helsinki University Printing House, Tuusula, Finland, 2006.

[91] O. Dekel, J. Keshet, and Y. Singer, "Large margin hierarchical classification," in *Proceedings of the 21th International Conference on Machine Learning (ICML '04)*, pp. 209–216, Omnipress, July 2004.

[92] N. Cesa-Bianchi, C. Gentile, and L. Zaniboni, "Hierarchical classifications combining bayes with SVM," in *Proceedings of the 23rd International Conference on Machine Learning (ICML '06)*, pp. 177–184, ACM Press, June 2006.

[93] T. G. Dietterich, "Ensemble methods in machine learning," in *Multiple Classifier Systems. First International Workshop, MCS, 2000, Cagliari, Italy*, J. Kittler and F. Roli, Eds., vol. 1857 of *Lecture Notes in Computer Science*, pp. 1–15, Springer, 2000.

[94] O. Okun, G. Valentini, and M. Re, *Ensembles in Machine Learning Applications*, vol. 373 of *Studies in Computational Intelligence*, Springer, Berlin, Germany, 2011.

[95] M. Re and G. Valentini, "Ensemble methods: a review," in *Advances in Machine Learning and Data Mining for Astronomy*, Data Mining and Knowledge Discovery, pp. 563–594, Chapman & Hall, 2012.

[96] E. Bauer and R. Kohavi, "Empirical comparison of voting classification algorithms: bagging, boosting, and variants," *Machine Learning*, vol. 36, no. 1, pp. 105–139, 1999.

[97] T. G. Dietterich, "Experimental comparison of three methods for constructing ensembles of decision trees: bagging, boosting, and randomization," *Machine Learning*, vol. 40, no. 2, pp. 139–157, 2000.

[98] R. E. Banfield, L. O. Hall, O. Lawrence, K. W. Bowyer, W. Kevin, and W. P. Kegelmeyer, "A comparison of decision tree ensemble creation techniques," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 1, pp. 173–180, 2007.

[99] S. Y. Sohn and H. W. Shin, "Experimental study for the comparison of classifier combination methods," *Pattern Recognition*, vol. 40, no. 1, pp. 33–40, 2007.

[100] M. Dettling and P. Bühlmann, "Boosting for tumor classification with gene expression data," *Bioinformatics*, vol. 19, no. 9, pp. 1061–1069, 2003.

[101] V. Robles, P. Larrañaga, J. M. Peña et al., "Bayesian network multi-classifiers for protein secondary structure prediction," *Artificial Intelligence in Medicine*, vol. 31, no. 2, pp. 117–136, 2004.

[102] G. Valentini, M. Muselli, and F. Ruffino, "Cancer recognition with bagged ensembles of support vector machines," *Neurocomputing*, vol. 56, no. 1–4, pp. 461–466, 2004.

[103] T. Abeel, T. Helleputte, Y. Van de Peer, P. Dupont, and Y. Saeys, "Robust biomarker identification for cancer diagnosis with ensemble feature selection methods," *Bioinformatics*, vol. 26, no. 3, Article ID btp630, pp. 392–398, 2009.

[104] A. Rozza, G. Lombardi, M. Re, E. Casiraghi, G. Valentini, and P. Campadelli, "A novel ensemble technique for protein subcellular location prediction," in *Ensembles in Machine Learning Applications*, vol. 373 of *Studies in Computational Intelligence*, pp. 151–177, Springer, Berlin, Germany, 2011.

[105] A. Topchy, A. K. Jain, and W. Punch, "Clustering ensembles: models of consensus and weak partitions," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 12, pp. 1866–1881, 2005.

[106] A. Bertoni and G. Valentini, "Ensembles based on random projections to improve the accuracy of clustering algorithms," in *Neural Nets, WIRN 2005*, vol. 3931 of *Lecture Notes in Computer Science*, pp. 31–37, Springer, 2006.

[107] R. E. Schapire, Y. Freund, P. Bartlett, and W. S. Lee, "Boosting the margin: a new explanation for the effectiveness of voting methods," *Annals of Statistics*, vol. 26, no. 5, pp. 1651–1686, 1998.

[108] E. L. Allwein, R. E. Schapire, and Y. Singer, "Reducing multiclass to binary: a unifying approach for margin classifiers," *Journal of Machine Learning Research*, vol. 1, no. 2, pp. 113–141, 2001.

[109] E. M. Kleinberg, "On the algorithmic implementation of stochastic discrimination," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 5, pp. 473–490, 2000.

[110] L. Breiman, "Bias, variance and arcing classifiers," Tech. Rep. TR 460, Statistics Department, University of California, Berkeley, Calif, USA, 1996.

[111] J. Friedman and P. Hall, "On bagging and nonlinear estimation," Tech. Rep., Statistics Department, University of Stanford, Stanford, Calif, USA, 2000.

[112] K. Dembczynski, W. Waegeman, W. Cheng, and E. Hullermeier, "On label dependence in multi-label classification," in *Proceedings of the 2nd International Workshop on learning from Multi-Label Data (ICML-MLD '10)*, pp. 5–12, Haifa, Israel, 2010.

[113] S. Mostafavi and Q. Morris, "Using the Gene Ontology hierarchy when predicting gene function," in *Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence (UAI '09)*, pp. 419–427, AUAI Press, Corvallis, Ore, USA, June 2009.

[114] L. J. Jensen, R. Gupta, H.-H. Stærfeldt, and S. Brunak, "Prediction of human protein function according to Gene Ontology categories," *Bioinformatics*, vol. 19, no. 5, pp. 635–642, 2003.

[115] A. Lægreid, T. R. Hvidsten, H. Midelfart, J. Komorowski, and A. K. Sandvik, "Predicting gene ontology biological process from

temporal gene expression patterns," *Genome Research*, vol. 13, no. 5, pp. 965–979, 2003.

[116] R. Bi, Y. Zhou, F. Lu, and W. Wang, "Predicting Gene Ontology functions based on support vector machines and statistical significance estimation," *Neurocomputing*, vol. 70, no. 4–6, pp. 718–725, 2007.

[117] P. Bogdanov and A. K. Singh, "Molecular function prediction using neighborhood features," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 7, no. 2, pp. 208–217, 2010.

[118] M. Riley, "Functions of the gene products of *Escherichia coli*," *Microbiological Reviews*, vol. 57, no. 4, pp. 862–952, 1993.

[119] R. E. Schapire and Y. Singer, "BoosTexter: a boosting-based system for text categorization," *Machine Learning*, vol. 39, no. 2, pp. 135–168, 2000.

[120] N. Alaydie, C. K. Reddy, and F. Fotouhi, "Hierarchical multilabel boosting for gene function prediction," in *Proceedings of the International Conference on Computational Systems Bioinformatics (CSB '10)*, Stanford, Calif, USA, 2010.

[121] T. Kohonen, "The self-organizing map," *Neurocomputing*, vol. 21, no. 1–3, pp. 1–6, 1998.

[122] H. Borges and J. Nievola, "Hierarchical classification using a competitive neural network," in *Proceedings of the International Conference on Computing, Networking and Communications (ICNC '12)*, pp. 172–177, 2012.

[123] N. Cesa-Bianchi, M. Re, and G. Valentini, "Synergy of multilabel hierarchical ensembles, data fusion, and cost-sensitive methods for gene functional inference," *Machine Learning*, vol. 88, no. 1, pp. 209–241, 2012.

[124] R. Cerri and A. C. P. L. F. De Carvalho, "Hierarchical multilabel classification using Top-Down label combination and Artificial Neural Networks," in *Proceedings of the 11th Brazilian Symposium on Neural Networks (SBRN '10)*, pp. 253–258, IEEE Computer Society, October 2010.

[125] R. Cerri and A. de Carvalho, "Hierarchical multilabel protein function prediction using local neural networks," in *Advances in Bioinformatics and Computational Biology*, vol. 6832 of *Lecture Notes in Computer Science*, pp. 10–17, Springer, 2011.

[126] D. E. Rumelhart, R. Durbin, and Y. Chauvin, "Backpropagation: the basic theory," in *Backpropagation: Theory, Architectures and Applications*, D. E. Rumelhart and Y. Chauvin, Eds., pp. 1–34, Lawrence Erlbaum, Hillsdale, NJ, USA, 1995.

[127] J. Hernandez, L. E. Sucar, and E. Morales, "A hybrid global-local approach forhierarchcial classification," in *Proceedings of the 26th International Florida Artificial Intelligence Research Society Conference*, pp. 432–437, 2013.

[128] B. Paes, A. Plastion, and A. Freitas, "Improving loacal per level hierarchical classification," *Journal of Information and Data Management*, vol. 3, no. 3, pp. 394–409, 2012.

[129] G. Tsoumakas, I. Katakis, and I. Vlahavas, "Random k-labelsets for multilabel classification," *IEEE Transactions on Knowledge and Data Engineering*, vol. 23, no. 7, pp. 1079–1089, 2011.

[130] H. Wang, X. Shen, and W. Pan, "Large margin hierarchical classification with mutually exclusive class membership," *Journal of Machine Learning Research*, vol. 12, pp. 2721–2748, 2011.

[131] L. Breiman, "Bagging predictors," *Machine Learning*, vol. 24, no. 2, pp. 123–140, 1996.

[132] M. des Jardins, P. Karp, M. Krummenacker, T. J. Lee, and C. A. Ouzounis, "Prediction of enzyme classification from protein sequence without the use of sequence similarity," in *Proceedings of the 5th International Conference on Intelligent Systems for Molecular Biology (ISMB '97)*, pp. 92–99, AAAI Press, 1997.

[133] A. Sharkey, N. Sharkey, U. Gerecke, and G. Chandroth, "The test and select approach to ensemble combination," in *Multiple Classifier Systems. First International Workshop, MCS 2000, Cagliari, Italy*, J. Kittler and F. Roli, Eds., vol. 1857 of *Lecture Notes in Computer Science*, pp. 30–44, Springer, 2000.

[134] Y. Kourmpetis, A. van Dijk, and C. ter Braak, "Gene Ontology consistent protein function prediction: the FALCON algorithm applied to six eukaryotic genomes," *Algorithms for Molecular Biology*, vol. 8, article 10, 2013.

[135] N. Cesa-Bianchi, C. Gentile, A. Tironi, and L. Zaniboni, "Incremental algorithms for hierarchical classification," in *Advances in Neural Information Processing Systems*, vol. 17, pp. 233–240, MIT Press, 2005.

[136] M. Re and G. Valentini, "An experimental comparison of Hierarchical Bayes and True Path Rule ensembles for protein function prediction," in *Multiple Classifier Systems. Nineth International Workshop, MCS, 2010, Cairo, Egypt*, N. El-Gayar, Ed., vol. 5997 of *Lecture Notes in Computer Science*, pp. 294–303, Springer, 2010.

[137] A. J. Smola and R. Kondor, "Kernels and regularization on graphs," in *Proceedings of the 16th Annual Conference on Learning Theory*, B. Scholkopf and M. K. Warmuth, Eds., Lecture Notes in Computer Science, pp. 144–158, Springer, August 2003.

[138] C. Leslie, E. Eskin, A. Cohen, J. Weston, and W. S. Noble, "Mismatch string kernels for svm protein classification," in *Advances in Neural Information Processing Systems*, pp. 1441–1448, MIT Press, Cambridge, Mass, USA, 2003.

[139] M. J. Wainwright and M. I. Jordan, "Graphical models, exponential families, and variational inference," *Foundations and Trends in Machine Learning*, vol. 1, no. 1-2, pp. 1–305, 2008.

[140] R. E. Barlow and H. D. Brunk, "The isotonic regression problem and its dual," *Journal of the American Statistical Association*, vol. 67, no. 337, pp. 140–147, 1972.

[141] O. Burdakov, O. Sysoev, A. Grimvall, and M. Hussian, "An $O(n^2)$ algorithm for isotonic regression," in *Large-Scale Nonlinear Optimization*, vol. 83 of *Nonconvex Optimization and Its Applications*, pp. 25–33, Springer, 2006.

[142] G. Valentini and M. Re, "Weighted True Path Rule: a multilabel hierarchical algorithm for gene function prediction," in *Proceedings of the 1st International Workshop on learning from Multi-Label Data (MLD-ECML '09)*, pp. 133–146, Bled, Slovenia, 2009.

[143] Gene Ontology Consortium. True path rule, 2010, http://www.geneontology.org/GO.usage.shtml#truePathRule.

[144] B. Chen, L. Duan, and J. Hu, "Composite kernel based svm for hierarchical multilabel gene function classification," in *Proceedings of the 26th International Florida Artificial Intelligence Research Society Conference*, pp. 1380–1385, 2012.

[145] B. Chen and J. Hu, "Hierarchical multi-label classification based on over-sampling and hierarchy constraint for gene function prediction," *IEEJ Transactions on Electrical and Electronic Engineering*, vol. 7, no. 2, pp. 183–189, 2012.

[146] J. R. Quinlan, "Induction of decision trees," *Machine Learning*, vol. 1, no. 1, pp. 81–106, 1986.

[147] R. D. King, A. Karwath, A. Clare, and L. Dehaspe, "The utility of different representations of protein sequence for predicting functional class," *Bioinformatics*, vol. 17, no. 5, pp. 445–454, 2001.

[148] H. Blockeel, M. Bruynooghe, S. Dzeroski, J. Ramon, and J. Struyf, "Top-down induction of clustering trees," in *Proceedings of the 15th International Conference on Machine Learning*, pp. 55–63, 1998.

[149] H. Blockeel, L. Schietgat, S. Dzeroski, and A. Clare, "Decision trees for hierarchical multilabel classification: a case study in functional genomics," in *Proc. of the 10th European Conference on Principles and Practice of Knowledge Discovery in Databases*, vol. 4213 of *Lecture Notes in Artificial Intelligence*, pp. 18–29, Springer, 2006.

[150] D. Stojanova, M. Ceci, D. Malerba, and S. Deroski, "Using PPI network autocorrelation in hierarchical multi-label classification trees for gene function prediction," *BMC Bioinformatics*, vol. 14, article 285, 2013.

[151] F. Otero, A. Freitas, and C. Johnson, "A hierarchical classification ant colony algorithm for predicting gene ontology terms," in *Evolutionary Computation, Machine Learning and Data Mining in Bioinformatics*, vol. 5483 of *Lecture Notes in Computer Science*, pp. 68–79, Springer, 2009.

[152] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.

[153] D. Kocev, C. Vens, J. Struyf, and S. Dzeroski, "Tree ensembles for predicting structured outputs," *Pattern Recognition*, vol. 46, no. 3, pp. 817–833, 2013.

[154] W. S. Noble and A. Ben-Hur, "Integrating information for protein function prediction," in *Bioinformatics—from Genomes To Therapies*, T. Lengauer, Ed., vol. 3, pp. 1297–1314, Wiley-VCH, 2007.

[155] L. Lan, N. Djuric, Y. Guo, and S. Vucetic, "MS-kNN: protein function prediction by integrating multiple data sources," *BMC Bioinformatics*, vol. 14, supplement 3, article S8, 2013.

[156] A. Sokolov, C. Funk, K. Graim, K. Verspoor, and A. Ben-Hur, "Combining heterogeneous data sources for accurate functional annotation of proteins," *BMC Bioinformatics*, vol. 14, supplement 3, article S10, 2013.

[157] C. L. Myers and O. G. Troyanskaya, "Context-sensitive data integration and prediction of biological networks," *Bioinformatics*, vol. 23, no. 17, pp. 2322–2330, 2007.

[158] S. Mostafavi and Q. Morris, "Fast integration of heterogeneous data sources for predicting gene function with limited annotation," *Bioinformatics*, vol. 26, no. 14, Article ID btq262, pp. 1759–1765, 2010.

[159] M. Mesiti, E. Jiménez-Ruiz, I. Sanz et al., "XML-based approaches for the integration of heterogeneous bio-molecular data," *BMC Bioinformatics*, vol. 10, no. 12, article 1471, p. S7, 2009.

[160] S. Sonnenburg, G. Rätsch, C. Schäfer, and B. Schölkopf, "Large scale multiple kernel learning," *Journal of Machine Learning Research*, vol. 7, pp. 1531–1565, 2006.

[161] A. Rakotomamonjy, F. Bach, S. Canu, and Y. Grandvalet, "More efficiency in multiple kernel learning," in *Proceedings of the 24th International Conference on Machine Learning (ICML '07)*, pp. 775–782, ACM, New York, NY, USA, June 2007.

[162] G. R. Lanckriet, M. Deng, N. Cristianini, M. I. Jordan, and W. S. Noble, "Kernel-based data fusion and its application to protein function prediction in yeast," *Proceedings of the Pacific Symposium on Biocomputing*, pp. 300–311, 2004.

[163] D. P. Lewis, T. Jebara, and W. S. Noble, "Support vector machine learning from heterogeneous data: an empirical analysis using protein sequence and structure," *Bioinformatics*, vol. 22, no. 22, pp. 2753–2760, 2006.

[164] N. Cesa-Bianchi, M. Re, and G. Valentini, "Functional inference in FunCat through the combination of hierarchical ensembles with data fusion methods," in *Proceedings of the 2nd International Workshop on learning from Multi-Label Data (MLD '10)*, pp. 13–20, Haifa, Israel, 2010.

[165] G. Yu, H. Rangwala, C. Domeniconi, G. Zhang, and Z. Zhang, "Protein function prediction by integrating multiple kernels," in *Proceedings of the 23rd International Joint Conference on Artificial Intelligence (IJCAI '13)*, Beijing, China, 2013.

[166] D. M. Titterington, G. D. Murray, L. S. Murray et al., "Comparison of discrimination techniques applied to a complex data set of head injured patients," *Journal of the Royal Statistical Society A*, vol. 144, no. 2, pp. 145–175, 1981.

[167] N. C. de Condorcet, *Essai sur l'Application de l'Analyse a la Probabilité des Décisions Rendues à la Pluralité des Voix*, Imprimerie Royale, Paris, France, 1785.

[168] J. Kittler, M. Hatef, R. P. W. Duin, and J. Matas, "On combining classifiers," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, no. 3, pp. 226–239, 1998.

[169] L. I. Kuncheva, J. C. Bezdek, and R. P. W. Duin, "Decision templates for multiple classifier fusion: an experimental comparison," *Pattern Recognition*, vol. 34, no. 2, pp. 299–314, 2001.

[170] M. Rè and G. Valentini, "Noise tolerance of multiple classifier systems in data integration-based gene function prediction," *Journal of Integrative Bioinformatics*, vol. 7, no. 3, article 139, 2010.

[171] M. Re and G. Valentini, "Ensemble based data fusion for gene function prediction," in *Multiple Classifier Systems. Eighth International Workshop, MCS, 2009, Reykjavik, Iceland*, J. Kittler, J. Benediktsson, and F. Roli, Eds., vol. 5519 of *Lecture Notes in Computer Science*, pp. 448–457, Springer, 2009.

[172] M. Re and G. Valentini, "Prediction of gene function using ensembles of SVMs and heterogeneous data sources," in *Applications of Supervised and Unsupervised Ensemble Methods*, vol. 245 of *Computational Intelligence Series*, pp. 79–91, Springer, 2009.

[173] M. Re and G. Valentini, "Integration of heterogeneous data sources for gene function prediction using decision templates and ensembles of learning machines," *Neurocomputing*, vol. 73, no. 7-9, pp. 1533–1537, 2010.

[174] R. D. Finn, J. Tate, J. Mistry et al., "The Pfam protein families database," *Nucleic Acids Research*, vol. 36, no. 1, pp. D281–D288, 2008.

[175] A. P. Gasch, P. T. Spellman, C. M. Kao et al., "Genomic expression programs in the response of yeast cells to environmental changes," *Molecular Biology of the Cell*, vol. 11, no. 12, pp. 4241–4257, 2000.

[176] C. Stark, B.-J. Breitkreutz, T. Reguly, L. Boucher, A. Breitkreutz, and M. Tyers, "BioGRID: a general repository for interaction datasets," *Nucleic Acids Research*, vol. 34, pp. D535–D539, 2006.

[177] C. Von Mering, R. Krause, B. Snel et al., "Comparative assessment of large-scale data sets of protein-protein interactions," *Nature*, vol. 417, no. 6887, pp. 399–403, 2002.

[178] G. Valentini and N. Cesa-Bianchi, "HCGene: a software tool to support the hierarchical classification of genes," *Bioinformatics*, vol. 24, no. 5, pp. 729–731, 2008.

[179] A. Ben-Hur and W. S. Noble, "Choosing negative examples for the prediction of protein-protein interactions," *BMC Bioinformatics*, vol. 7, supplement 1, article S2, 2006.

[180] N. Skunca, A. Altenhoff, and C. Dessimoz, "Quality of computationally inferred gene ontology annotations," *PLoS Computational Biology*, vol. 8, no. 5, Article ID e1002533, 2012.

[181] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: synthetic minority over-sampling technique," *Journal of Artificial Intelligence Research*, vol. 16, pp. 321–357, 2002.

[182] G. Batista, R. Prati, and M. Monard, "A study of the behavior of several methods for balancing machine learning training data," *ACM SIGKDD Explorations Newsletter*, vol. 6, no. 1, pp. 20–29, 2004.

[183] M. Kubat and S. Matwin, "Addressing the curse of imbalanced training sets: one-sided selection," in *Proceedings of the 14th International Conference on Machine Learning (ICML '97)*, pp. 179–187, Nashville, Tenn, USA, 1997.

[184] H. Guo and H. Viktor, "Learning from imbalanced data sets with boosting and data generation: the Databoost-IM approach," *ACM SIGKDD Explorations Newsletter*, vol. 6, no. 1, pp. 30–39, 2004.

[185] Y. Sun, M. S. Kamel, A. K. C. Wong, and Y. Wang, "Cost-sensitive boosting for classification of imbalanced data," *Pattern Recognition*, vol. 40, no. 12, pp. 3358–3378, 2007.

[186] Y. Zhang and D. Wang, "Cost-sensitive ensemble method for class-imbalanced datasets," *Abstract and Applied Analysis*, vol. 2013, Article ID 196256, 6 pages, 2013.

[187] M. Galar, A. Fernandez, E. Barrenechea, H. Bustince, and F. Herrera, "A review on ensembles for the class imbalance problem: bagging-. boosting-, and hybrid-based approaches," *IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews*, vol. 42, no. 4, pp. 463–484, 2012.

[188] C. Huttenhower, M. A. Hibbs, C. L. Myers, A. A. Caudy, D. C. Hess, and O. G. Troyanskaya, "The impact of incomplete knowledge on evaluation: an experimental benchmark for protein function prediction," *Bioinformatics*, vol. 25, no. 18, pp. 2404–2410, 2009.

[189] G. Yu, C. Domeniconi, H. Rangwala, G. Zhang, and Z. Yu, "Transductive multilabel ensemble classification for protein function prediction," in *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1077–1085, 2012.

[190] G. Yu, H. Rangwala, C. Domeniconi, G. Zhang, and Z. Yu, "Protein function prediction with incomplete annotations," *IEEE Transactions on Computational Biology and Bioinformatics*, 2013.

[191] Gene Ontology Consortium, "Gene Ontology annotations and resources," *Nucleic Acids Research*, vol. 41, pp. D530–D535, 2013.

[192] A. A. Freitas, D. C. Wieser, and R. Apweiler, "On the importance of comprehensible classification models for protein function prediction," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 7, no. 1, pp. 172–182, 2010.

[193] R. Cerri, R. Barros, A. de Carvalho, and A. Freitas, "A grammatical evolution algorithm for generation of hierarchical multilabel classification rules," in *Proceedings of the IEEE Congress on Evolutionary Computation*, 2013.

[194] C. Widmer and G. Ratsch, "Multitask learning in computational biology," *Journal of Machine Learning Research, W&P*, vol. 27, pp. 207–216, 2012.

[195] J. Gonzalez, Y. Low, H. Gu, D. Bickson, and C. Guestrin, "PowerGraph: distributed graph-parallel computation on natural graphs," in *Proceedings of the 10th USENIX conference on Operating Systems Design and Implementation (OSDI '12)*, pp. 17–30, Hollywood, Calif, USA, 2012.

[196] M. Mesiti, M. Re, and G. Valentini, "Scalable network-based learning methods for automated function prediction based on the neo4j graph-database," in *Proceedings of the Automated Function Prediction SIG 2013—ISMB Special Interest Group Meeting*, pp. 6–7, Berlin, Germany, 2013.

[197] H. W. Mewes, K. Albermann, M. Bähr et al., "Overview of the yeast genome," *Nature*, vol. 387, no. 6632, pp. 7–8, 1997.

[198] H. W. Mewes, D. Frishman, C. Gruber et al., "MIPS: a database for genomes and protein sequences," *Nucleic Acids Research*, vol. 28, no. 1, pp. 37–40, 2000.

[199] K. R. Christie, S. Weng, R. Balakrishnan et al., "Saccharomyces Genome Database (SGD) provides tools to identify and analyze sequences from Saccharomyces cerevisiae and related sequences from other organisms," *Nucleic Acids Research*, vol. 32, pp. D311–D314, 2004.

[200] K. Verspoor, D. Dvorkin, K. B. Cohen, and L. Hunter, "Ontology quality assurance through analysis of term transformations," *Bioinformatics*, vol. 25, no. 12, pp. i77–i84, 2009.

[201] K. Verspoor, J. Cohn, S. Mniszewski, and C. Joslyn, "A categorization approach to automated ontological function annotation," *Protein Science*, vol. 15, no. 6, pp. 1544–1549, 2006.

[202] S. Kiritchenko, S. Matwin, and A. Famili, "Hierarchical text categorization as a tool of associating genes with Gene Ontology codes," in *Proceedings of the 2nd European Workshop on Data Mining and Text Mining for Bioinformatics*, pp. 26–30, 2004.