True Path Rule hierarchical ensembles for genome-wide gene function prediction

Giorgio Valentini

Abstract

Gene function prediction is a complex computational problem, characterized by several items: the number of functional classes is large, and a gene may belong to multiple classes; functional classes are structured according to a hierarchy; classes are usually unbalanced, with more negative than positive examples; class labels can be uncertain and the annotations largely incomplete; to improve the predictions, multiple sources of data need to be properly integrated. In this contribution we focus on the first three items, and in particular on the development of a new method for the hierarchical genome-wide and ontology-wide gene function prediction. The proposed algorithm is inspired by the "true path rule" that governs both the Gene Ontology and FunCat taxonomies. According to this rule, the proposed *True Path Rule (TPR)* ensemble method is characterized by a two-way asymmetric flow of information that traverses the graph-structured ensemble: positive predictions for a node influence in a recursive way its ancestors, while negative predictions influence its offsprings. Cross-validated results with the model organism *S. cerevisiae*, using 7 different sources of biomolecular data, and a theoretical analysis of the the *TPR* algorithm show the effectiveness and the drawbacks of the proposed approach.

Index Terms

Gene function prediction, ensemble methods, hierarchical classification, Functional Catalogue.

1 INTRODUCTION

Exploiting the wealth of biomolecular data accumulated by novel high-throughput biotechnologies, "in silico" gene function prediction methods [1] provide hypothetical annotations that can drive the biological validation and discovery of novel functions of genes and gene products, thus resulting in a relevant saving of experimental resources [2].

Gene function prediction is a multiclass, multilabel classification problem characterized by hundreds or thousands of functional classes structured according to a predefined hierarchy.

• *Giorgio Valentini is with DSI, Dipartimento di Scienze dell'Informazione, Università degli Studi di Milano, Via Comelico 39, Milano, Italy, E-mail: valentini@dsi.unimi.it*

From a general standpoint several approaches have been proposed for multilabel classification, with applications ranging from gene function prediction, to music categorization and semantic scene classification [3], [4], [5]. Classifications of hierarchically structured classes have been applied in several contexts, such as the automatic classification of World Wide Web documents [6] or the prediction of enzyme classes according to the Enzyme Classification scheme [7].

In the framework of gene function prediction, the two main taxonomies of gene functional classes are represented by the Gene Ontology (GO) [8], and The Functional Catalogue (FunCat) [9]. The GO is composed by thousands of functional classes structured according to a directed acyclic graph, 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, 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). The FunCat represents a more 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.

Several gene function prediction methods considered a relatively small set of functional classes [10], [11], [12], while others provided predictions extended to larger sets, using graph based or machine learning methods such as functional linkage networks [13], [14], Bayesian Networks [11], Support Vector Machines and semidefinite programming [10], artificial neural networks [15], or methods that combine functional linkage networks with learning machines using a logistic regression model [16] or simple algebraic operators [17]. Other promising approaches are represented by structured-output methods, based on the joint kernelization of both input variables and output labels, using e.g. perceptron-like learning algorithms [18] or maximum-margin algorithms [19], and by methods that improve the prediction of GO annotations by extracting implicit semantic relationships between genes and functions [20]. Several methods tried to take advantage of the intrinsic hierarchical nature of gene function prediction, explicitly considering the relationships between functional classes [21], [22], [23], [24], [25].

Our proposed approach not only explicitly takes into account the hierarchical relationships between functional classes, but it is also directly inspired by the *true path rule* that can be summarized as follows [26]: “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”. According to this rule, that governs the annotations of both GO and FunCat taxonomies, the proposed ensemble method is

characterized by a two-way asymmetric flow of information that traverses the graph-structured ensemble: positive predictions for a node influence in a recursive way its ancestors, while negative predictions influence its offsprings. The resulting ensemble embeds the functional relationships between functional classes that characterize the hierarchical taxonomy. Our approach is related to two recently proposed methods [27], [28]. In [27] 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. Nevertheless, this approach propagates information only towards the bottom of the graph, differing from our proposed method that propagates information also from bottom to top of the overall hierarchy.

The proposed *True Path Rule (TPR)* ensemble method is also related to the approach proposed by [28], by which predictions of an ensemble of SVMs, each specialized to identify genes belonging to a specific GO class, are hierarchically combined, extending a previous approach based on Bayesian networks [29]. More precisely in [28] two local strategies are proposed to take into account the relationships between GO nodes: the first one is based on the Markov blanket associated to each node (that is the subgraph involving its parents, children and children's parents), and the other one on a breadth first search to recover all descendants up to a maximum of 30 GO nodes. Differently from [28], in our approach the flow of information between nodes is not limited for each node to a specific subgraph of the taxonomy, but it traverses the graph involving a large set of classes of the hierarchy. This global strategy, even if in principle applicable with slight modifications to the GO, has been applied to FunCat, where graphs are simpler and structured according to a tree forest.

The *TPR* method can be applied to predict the annotations of genes at the level of the entire taxonomy or considering specific subsets of the hierarchical functional classes, and provides probabilistic and structured predictions of gene annotations. Moreover, by tuning a single global parameter, it allows to regulate the trade-off between precision and recall that characterizes gene function prediction problems. We applied the *True Path Rule* hierarchical ensemble method to the prediction of gene functions in yeast, using probabilistic SVMs as base learners [30], but the algorithm is general enough to be used with any probabilistic base learner and with other model organisms. Considering that data integration is crucial to improve prediction performances [31], *TPR* ensembles can be easily integrated with state-of-the-art biomolecular data integration methods [32], such as vector-space integration [33], kernel fusion [10] or ensembles of learning machines [34], without any modification of the algorithmic scheme.

A basic version of the *TPR* ensemble method has been recently presented in two conferences on ensemble methods and multilabel learning [35], [36]. In this paper we present an extended and enhanced version in order to discuss in detail the characteristics and the theoretical properties of *TPR* ensembles, their application to the genome-wide gene function prediction in model organisms, and new possible research lines in the context of the hierarchical classification of gene functions. More precisely this paper is organized as follows: in Sect. 2 the ensemble method inspired by the *true path rule* is presented, and its

properties are analyzed and discussed. Sect. 3 summarizes the experimental set-up, and introduces the types of bio-molecular data used in the experiments and the performance measures applied to properly evaluate the results in a multi-label hierarchical context. Sect. 4 shows genome-wide gene function prediction results obtained with the model organism *S. cerevisiae* using the proposed methods compared with hierarchical top-down and *Flat* ensemble approaches, and discusses the pros and cons of the true path rule-based ensembles. The conclusions summarizes the main contributions and depicts possible future developments of this work. Detailed experimental results are available as Supplementary Information at: `http://homes.dsi.unimi.it/~valenti/SupplInfo/TPR`. Source R code implementing the *TPR* algorithm is available upon request from the author.

2 METHODS

2.1 Notation and basic definitions

Genome-wide gene function prediction can be modeled as a hierarchical, multiclass and multilabel classification problem. Indeed a gene/gene product x can be assigned to one or more functional classes of the set $\mathcal{C} = \{c_1, c_2, \ldots, c_m\}$. The assignments can be coded through a vector of multilabels $\mathbf{y} =< y_1, y_2, \ldots, y_m >\in \{0,1\}^m$, by which if x belongs to class c_i, then $y_i = 1$, otherwise $y_i = 0$, where the variable $i, 1 \leq i \leq m$, refers to the indices corresponding to the m classes belonging to the set $\mathcal{C}$.

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 arcs 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 $\mathrm{child}(i)$, and the set of its parents by $\mathrm{par}(i)$. Moreover $y_{child(i)}$ denotes the labels of the children classes of node i and analogously $y_{par(i)}$ 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. A classifier $D : X \rightarrow \{0,1\}^m$ computes the multilabel associated to each gene $x \in X$, and $d_i(x) \in \{0,1\}$ is the label predicted by the classifier for class c_i. For the sake of simplicity if there is no ambiguity we represent $d_i(x)$ simply by d_i.

2.2 A gene function prediction method inspired by the True Path Rule

The True Path Rule enforces the consistency of gene function annotations in both GO and FunCat taxonomies:

> "If the child term describes the gene product, then all its parent terms must also apply to that gene product"

In other words, if a gene is annotated with a specific functional term (functional class), then it is annotated with all the "parent" classes, and with all its ancestors in a recursive way. If a gene x is not annotated to a

class c, the situation is slightly different in the GO and FunCat. Indeed in FunCat x cannot belong to none of the offsprings classes of c, because each node can have only one parent; in the GO, being structured according to a direct acyclic graph, x can be annotated to some of the offsprings of c, if there are one or more ancestors of the offsprings of c for which the gene x is annotated. Fig. 1 shows an example of the application of the True Path Rule with the FunCat taxonomy.

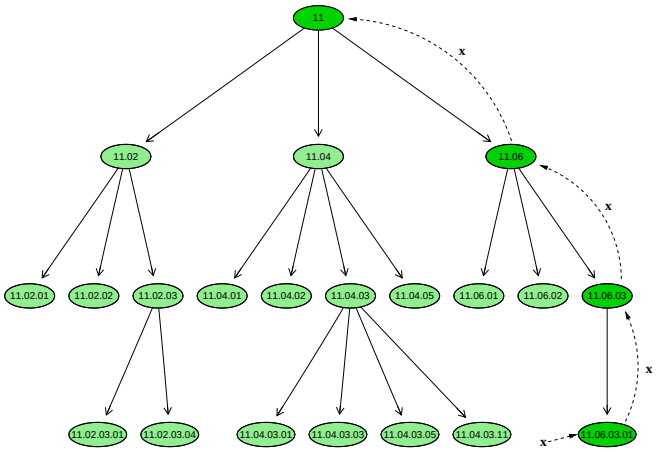

Fig. 1. FunCat tree rooted at class 11 (Transcription): if example x belongs to class 11.06.03.01 then it belongs also to class 11.06.03, 11.06 and 11. On the contrary, if a gene x does not belong to class 11.02 it cannot belong, e.g., to class 11.02.03 or 11.02.03.01.

For a given example x, considering the parents of a given node i, a classifier that respects the true path rule needs to obey the following rules:

$$\begin{cases} d_i = 1 & \Rightarrow \quad d_{par(i)} = 1 \\ d_i = 0 & \not\Rightarrow \quad d_{par(i)} = 0 \end{cases} \tag{1}$$

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:

$$\begin{cases} d_i = 1 & \not\Rightarrow \quad d_{child(i)} = 1 \\ d_i = 0 & \Rightarrow \quad d_{child(i)} = 0 \end{cases} \tag{2}$$

From eq. 1 and 2 we can 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 (eq. 1), and a propagation of negative labels from top to bottom (eq. 2). On the contrary negative labels cannot propagate from bottom to top, and positive predictions cannot propagate from top to bottom.

2.3 The *True Path Rule* (*TPR*) ensemble algorithm

The proposed hierarchical ensemble 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 *true path rule ensemble algorithm* can be summarized as follows:

a. Training of the base learners: for each node of the hierarchy a suitable learning algorithm (e.g. a multi-layer perceptron or a support vector machine) provides a classifier for the associated functional class.
b. In the evaluation phase the trained classifiers associated to each class/node of the graph provide a local decision about the assignment of a given example to a given node.
c. Positive decisions 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. On the contrary negative decisions do no affect decisions of the parent node (that is they do not propagate from bottom to top, eq. 1).
d. 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. On the contrary positive decisions do not influence decisions of child nodes (eq. 2).

The ensemble combines the local predictions of the base learners associated to 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 $\hat{p}_i(x)$ that a given example x belongs to class c_i, but the core of the algorithm is represented by the evaluation phase, where the ensemble provides an estimate of the "consensus" global probability $p_i(x)$.

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

$$\phi_i(x) = \{j | j \in \text{child}(i), d_j(x) = 1\} \tag{3}$$

The global consensus probability $p_i(x)$ of the ensemble depends both on the local prediction $\hat{p}_i(x)$ and on the prediction of the nodes belonging to $\phi_i(x)$:

$$p_i(x) = \frac{1}{1 + |\phi_i(x)|} \left(\hat{p}_i(x) + \sum_{j \in \phi_i(x)} p_j(x) \right) \tag{4}$$

The decision $d_i(x)$ at node/class i is set to 1 if $p_i(x) > t$, and to 0 otherwise (a natural choice for t is 0.5). Note that the restriction to nodes belonging to $\phi_i(x)$ in the summation of eq. 4 depends on the true path rule: indeed only children nodes for which we have a positive prediction can influence their parent. In the leaf nodes the sum of eq. 4 disappears and eq. 4 becomes $p_i(x) = \hat{p}_i(x)$. In this way positive predictions propagate from bottom to top. On the contrary if for a given node $d_i(x) = 0$, then this decision is propagated to its subtree.

A high-level representation of the evaluation phase of the algorithm for a tree-structured graph is given in Fig. 2. The pseudocode of the algorithm is characterized by two main for loops: the external for (from row 1 to 18) handles a per level bottom-up traversal of the tree, while the internal (from row 2 to 17) scans the nodes at each level. If a node is a leaf (row 3), then the consensus probability p_i is equal to the

Fig. 2. **True Path Rule bottom-up hierarchical algorithm**

`Input`:

- gene x whose classes need to be predicted
- tree T of the m hierarchical classes
- set of m classifiers (one for each node) each predicting $\hat{p}_i(x),\ 1 \leq i \leq m$

`begin algorithm`

01: `for each` level k of the tree T from bottom to top `do`
02: `for each` node i at level k `do`
03: `if` i is a leaf
04: $p_i(x) \leftarrow \hat{p}_i(x)$
05: `if` $(p_i(x) > t)$ $d_i(x) \leftarrow 1$
06: `else` $d_i(x) \leftarrow 0$
07: `else`
08: $\phi_i(x) \leftarrow \{j | j \in \text{child}(i), d_j(x) = 1\}$
09: $p_i(x) \leftarrow \frac{1}{1+|\phi_i(x)|}\left(\hat{p}_i(x) + \sum_{j \in \phi_i(x)} p_j(x)\right)$
10: `if` $(p_i(x) > t)$ $d_i(x) \leftarrow 1$
11: `else`
12: $d_i(x) \leftarrow 0$
13: `for each` $j \in$ subtree(i) `do`
14: $d_j(x) \leftarrow 0$
15: `if` $(p_j(x) > p_i(x))$ $p_j(x) \leftarrow p_i(x)$
16: `end for`
17: `end for`
18: `end for`

`end algorithm`.

`Output`: for each node i

- the ensemble decisions $d_i(x) = \begin{cases} 1 & \text{if } x \text{ belongs to node } i \\ 0 & \text{otherwise} \end{cases}$
- the probabilities $p_i(x)$ that $x \in X$ belongs to the node $i \in T$

local probability $\hat{p}_i(x)$. Note that a positive decision is taken if $p_i(x)$ is larger than a threshold t (row 5). If a node is not a leaf (rows 7-16), at first the set $\phi_i(x)$ collects all the children nodes for which we have a positive prediction, and the consensus probability p_i of the ensemble is computed by considering both the local estimate of the probability $\hat{p}_i$ and the probabilities computed by the children nodes for which

a positive decision has been taken (row 9). In case of a negative decision for a node i, all the classes belonging to the subtree rooted at i are set to negative, and their probabilities are set to p_i if larger than p_i (rows 13-16). Indeed, according to the hierarchical structure of GO and FunCat taxonomies, if a gene does not belong to a class, it cannot belong to the descendants of that class. The algorithm provides both the multilabels associated to the example x and the probabilities p_i that a given example belongs to the class i, $1 \leq i \leq m$.

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 $d_i(x)$ 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. This comes from the fact that the ensemble respects the *true path rule*.

Note that in the *TPR* algorithm there is no way to explicitly balance the local prediction $\hat{p}_i(x)$ at node i with the positive predictions coming from its offsprings (eq. 4). 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 we introduce a *parent weight* w, $0 \leq w \leq 1$, 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:

$$p_i(x) = w \cdot \hat{p}_i(x) + \frac{1-w}{|\phi_i(x)|} \sum_{j \in \phi_i(x)} p_j(x) \tag{5}$$

Hence we can obtain a variant of the *TPR* algorithm, that we name *weighted True Path Rule (TPR-w)* hierarchical ensemble algorithm by substituting rows 8 and 9 of the basic algorithm with the following pseudocode:

$\phi_i(x) \leftarrow \{j | j \in \text{child}(i), d_j(x) = 1\}$
`if` $(|\phi_i(x)| > 0)$
  $p_i(x) \leftarrow w \cdot \hat{p}_i(x) + \frac{1-w}{|\phi_i(x)|} \sum_{j \in \phi_i(x)} p_j(x)$
`else`
  $p_i(x) \leftarrow \hat{p}_i(x)$

2.4 Analysis of the propagation of positive decisions

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. In this section we analyze the contribution of the descendants nodes/classifiers to the decision of the upper nodes, that is we study the influence of

the positive decisions of the descendants on the decision of their ancestors. This analysis aims at a better understanding of the characteristics of the *TPR* algorithm, and at suggesting new research lines based on variants of the proposed true path rule-based algorithms.

We define nodes at level k as nodes with distance k from the root, that is nodes with a path to the root of length k. The root is at level 0, its children at level 1, its grandchildren at level 2 and so on. 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. Note that q_k differs from p_i defined in Sect. 2.3, since p_i refers to the probability computed for the node i of the hierarchy, while q_k to the probability computed for a generic node at level k of the hierarchy. More precisely we denote by q_k the probability computed by the ensemble and by $\hat{q}_k$ 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. Finally, define a node at level k positive when it is assigned to a node at level k by the ensemble, that is whenever $q_k \geq t$.

We start by considering a simplified setting, where for a node at level k we have exactly one positive descendant for each of the m lower levels below. Considering that in this case we have exactly 1 child for each level, to simplify the notation we set $q_k^j \equiv q_k$.

Proposition 1 (Influence of positive descendant nodes with one child): 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 exactly one positive descendant for each of the m lower levels below, that is such that $q_{k+j} > t, 1 \leq j \leq m$, the following equality holds for each $m \geq 1$:

$$q_k = \sum_{j=0}^{m-1} w(1-w)^j \hat{q}_{k+j} + (1-w)^m q_{k+m} \tag{6}$$

Proof: By induction, see Appendix A.

Remark 1: From Proposition 1 we can observe that considering the node at level $k+m$, if we have exactly m positive descendants of the node at level k, then $q_{k+m} = \hat{q}_{k+m}$. As a consequence the influence due to the node at level $k+m$ is $(1-w)^m \hat{q}_{k+m}$. To get insights into the influence of the m^{th} positive node on the prediction of node k, we can study the function $f(w) = (1-w)^m$, with $w \in [0,1]$ (last term of eq. 6). It is easy to see that for each $m \geq 1$:

$$\arg\max f(w) = 0, \qquad \arg\min f(w) = 1$$

Looking at the graph of $f(w)$ for different values of $m \geq 1$ (Fig. 3), we can observe that the deeper the node is (the larger the value of m), the lesser is the influence of the node at level $k+m$ on the ancestor node at level k.

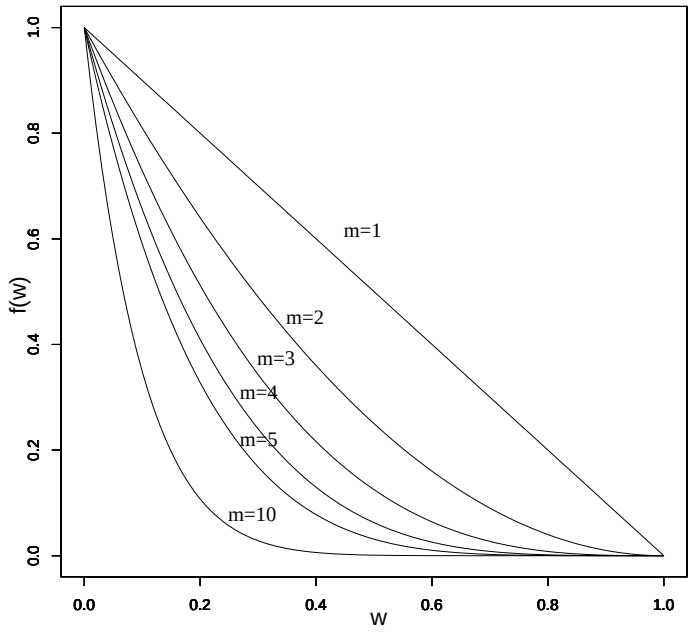

Fig. 3. Plot of $f(w) = (1-w)^m$, while varying m from 1 to 10.

Remark 2: From Proposition 1, positive nodes above the deepest one, that is nodes at level $k+j$, $j < m$, affect q_k by the quantity $w(1-w)^j \hat{q}_{k+j}$ (eq. 6). Considering the function $g(w) = w(1-w)^j$, with $w \in [0,1]$, we have that $\arg\max g(w) = \frac{1}{j+1}$ (see Appendix B for details). Fig. 4 shows the plot of $g(w)$ for different values of j: the nodes closer to the main node at level k have always larger influence independently of the value of w. Nevertheless, for large values of w (e.g. $w = 0.8$), nodes below 3 levels ($j \geq 3$) in practice are uninfluential on the node k. On the contrary for small w values (e.g. $w = 0.1$) also nodes at deep levels can play a certain role. The vertical lines in Fig. 4 highlight the differences between the relative influence of the nodes at different levels j from k for respectively small, medium and large values of the parameter w.

Remark 3: Proposition 1 and remarks 1 and 2 show that the contribution of the positive descendant nodes decays exponentially with their level. The intensity of the decay depends on the parameter w, and the parameter w affects the contribution of the different levels (Fig. 3 and 4).

Proposition 1 can be easily extended to the more general case when we have a variable number of positive children for each descendant node. To this end, recalling that ϕ_i is the set of children of i for which we have a positive prediction (eq. 3), we can extend its definition for a generic node at level k: ϕ_k is the set of children of the generic node at level k for which we have a positive prediction, and eq. 5 for a generic node at level k becomes:

$$q_k(x) = w \cdot \hat{q}_k(x) + \frac{1-w}{|\phi_k(x)|} \sum_{j \in \phi_k(x)} q^j_{k+1}(x) \tag{7}$$

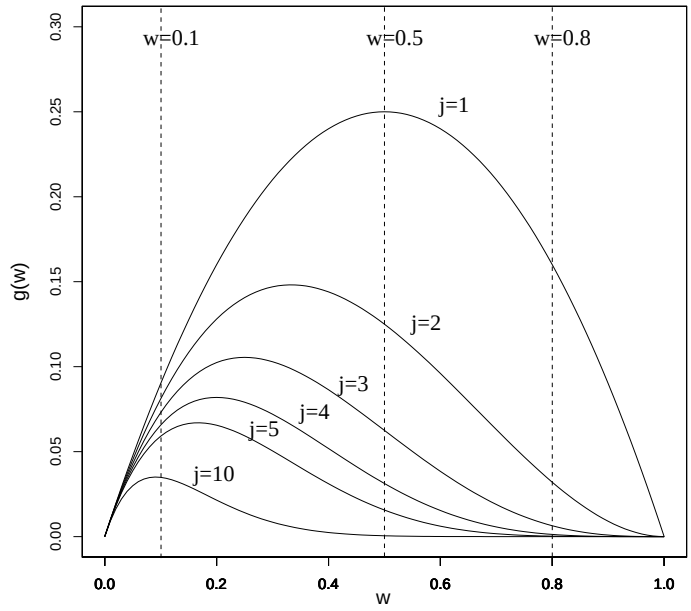

Fig. 4. Plot of $g(w) = w(1-w)^j$, while varying j from 1 to 10. The integers j refer to internal nodes at distance j from the reference node at level k

We can observe that the quantity $\frac{1}{|\phi_k(x)|}\sum_{j\in\phi_k(x)} q^j_{k+1}(x)$ in eq. 7 is the average of the probabilities computed by the positive children nodes of a generic node at level k. For brevity, we define this average as a_{k+1}:

$$a_{k+1} = \frac{1}{|\phi_k|}\sum_{j\in\phi_k} \hat{q}^j_{k+1} \tag{8}$$

The average of the probability averages of the positive grandchildren of a node at level k is:

$$a_{k+2} = \frac{1}{|\phi_k|}\sum_{j\in\phi_k} \frac{1}{|\phi^j_{k+1}|}\sum_{l\in\phi^j_{k+1}} \hat{q}^l_{k+2} \tag{9}$$

At a next level, the average of the averages of the probability averages of a node at level K is:

$$a_{k+3} = \frac{1}{|\phi_k|}\sum_{j\in\phi_k} \frac{1}{|\phi^j_{k+1}|}\sum_{l\in\phi^j_{k+1}} \frac{1}{|\phi^l_{k+2}|}\sum_{r\in\phi^l_{k+2}} \hat{q}^r_{k+3} \tag{10}$$

By extending these definition across levels, we can obtain the following proposition that generalizes Proposition 1:

Proposition 2 (Influence of positive descendant nodes with a variable number of children): 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 or equal than 1 of positive descendants for each of the m lower levels below, the following equality holds for each $m \geq 1$:

$$q_k = w\hat{q}_k + \sum_{j=1}^{m-1} w(1-w)^j a_{k+j} + (1-w)^m a_{k+m} \tag{11}$$

Proof: See Appendix C

Remark 4: Note that the form of Proposition 2 is very similar to that of Proposition 1, by substituting q_k with a_k. For this reason, the remarks 1, 2 and 3 can be extended to the general case when positive nodes have a variable number of positive children. Indeed, also in the general case the contribution of the descendant nodes decays exponentially with their depth and depends critically on the choice of the w parameter.

To get a quantitative insight into the influence on a given node at level k of its positive descendants, using the results of Proposition 2, we can study the function:

$$h(\hat{q_k}, a, w, m) = w\hat{q}_k + \sum_{j=1}^{m-1} w(1-w)^j a_{k+j} + (1-w)^m a_{k+m} \tag{12}$$

by varying $\hat{q_k}$, and w. We numerically simulated the results of the predictions of a generic node at level k in *TPR-w* by varying $\hat{q_k}$ and w between 0 and 1; m is fixed to 5 (5 levels of positive descendants) and we considered the following cases for the averages a_{k+j}, $1 \leq j \leq m$:

Local rule : No influence of descendant nodes.

Constant rule : $a_{k+j} = 0.75$.

Decreasing rule : $a_{k+j} = 1.1 - 0.1 \cdot j$.

Increasing rule : $a_{k+j} = 0.5 + 0.1 \cdot j$.

Of course these rules represent an oversimplification of real cases, because a_{k+j} may vary with different and non monotone rules across levels. Nevertheless this setting can be useful to shed light on the general behaviour of the algorithm in simplified and well characterized situations.

Fig. 5 summarizes the results. Fig. 5 (a) reports results when there is no influence of positive descendants (Local rule). The decision depends only on the probability computed by the local predictor: a positive prediction is given when the probability of the local classifier is larger than 0.5. This situation happens when no positive descendants are available (that is, all the children of the node give a negative prediction). Note that this is equivalent to the situation of *Flat* ensembles that do no take into account the decisions of the other nodes. Fig. 5 b, c and d show that the decision of a node in a *TPR-w* ensemble is influenced by the positive decisions of its offsprings. Indeed, the pink and red areas (probability of the ensemble larger than 0.5) cross on the left side the vertical dotted line representing the 0.5 probability of the local predictor, thus resulting in a modification of the negative decision of the local classifier to a positive one of the overall ensemble. It is worth noting that the value of the ensemble parameter w (abscissa of Fig. 5) affects significantly the prediction of the ensembles: high values enforce the prediction of the parent classifiers, while low values increase the influence of the descendant nodes. The results depend of course also on the "strength" (i.e. the probability) of the predictions of the offsprings and by their level. Indeed, when the probabilities computed by the deeper nodes (decreasing rule, Fig. 5 c) are lower, even for low values of w (that is, giving more weight to the positive descendant nodes) the intensity

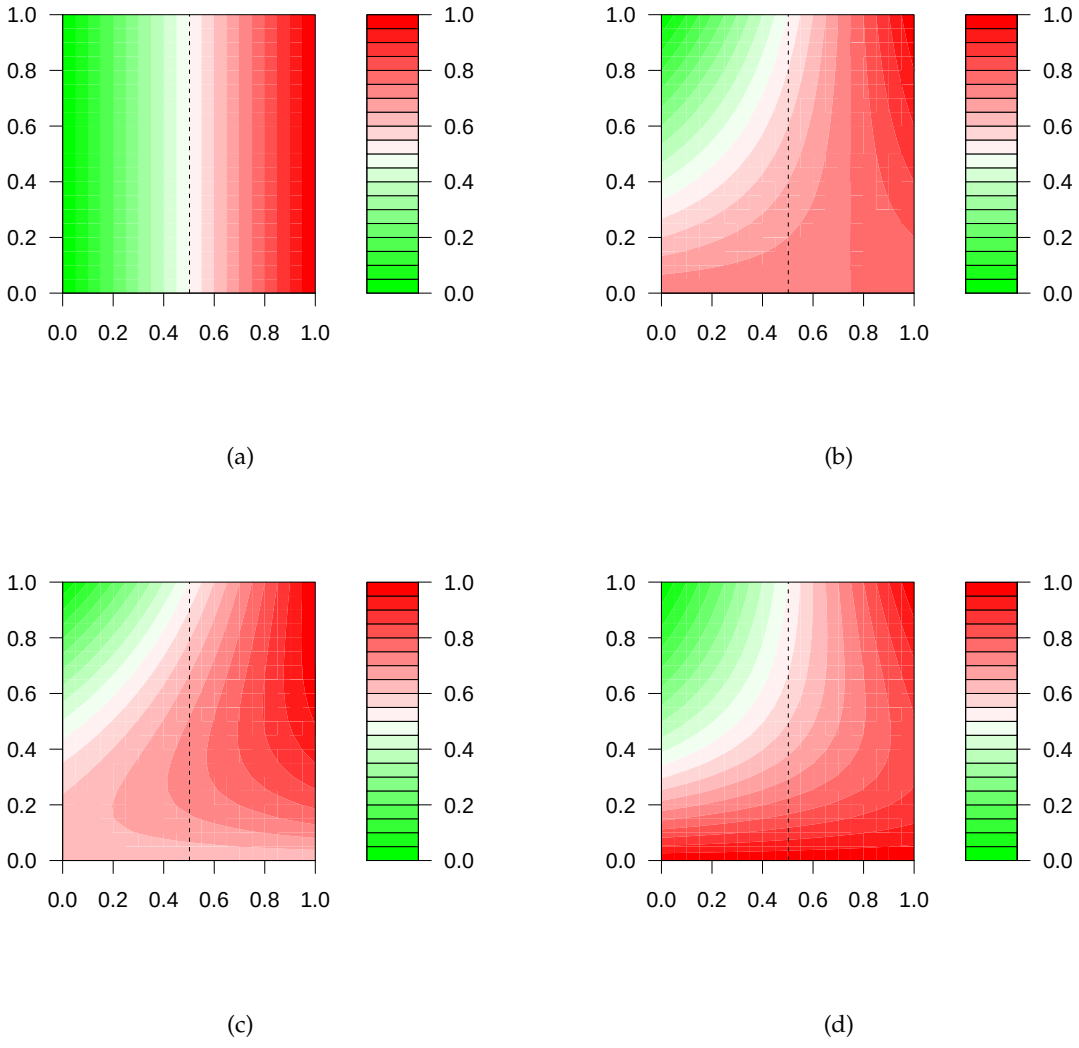

Fig. 5. Influence of positive descendants on the prediction of a node in *TPR-w* ensembles by varying the values of a_{k+j} according to the Local, Constant, Decreasing and Increasing rules. Abscissa: probability $\hat{q}_k$ predicted by the parent local classifier; ordinate: value of the w parameter. Colors represent the probability computed by the ensemble (red: high probability; green: low probability). The vertical dotted line highlights the 0.5 probability computed by the local predictor for a node at level k. (a) Local rule; (b) Constant rule; (c) Decreasing rule; (d) Increasing rule.

of the modification toward a positive decision of the ensemble is significant, but lower with respect to the increasing rule case, where the probabilities computed by the deeper nodes are larger than those computed by high level nodes (Fig. 5 d). In Fig. 5 b (constant rule) we are in an intermediate case. These results are explained by Proposition 2, since the impact of lower level nodes on the decision of a node

at level k decreases exponentially with their depth, but depends also on the average of the probabilities computed by the positive descendants at each level, and on the global parameter w.

Summarizing, this section provides a quantitative analysis of the influence of the positive predictions of descendant nodes on a generic node at level k. This analysis shows that the posterior probability computed by a generic node of the ensemble depends non linearly on the weight parameter and on the nested average of the probabilities computed by the positive descendants nodes at each level. These results can also suggest some modifications of the *TPR-w* algorithm, as briefly discussed in Sect. 4.8.

3 EXPERIMENTAL SET-UP

We considered the functional classification of yeast genes at genome-wide level for a large number of classes structured according to the *FunCat* (Functional Catalogue), a hierarchically tree-structured, controlled classification system enabling the functional description of proteins from any organism [9].

For each data set we evaluated the performance of four different ensembles: the *Flat* ensemble, the Hierarchical *Top-down* [37], [38] and the proposed *True Path Rule* (*TPR*) hierarchical ensemble and its weighted variant (*TPR-w*, Sect. 2.3).

The *Flat* ensemble does not take into account the hierarchical structure of the data and simply returns the predictions of each base classifier trained to recognize the genes belonging to a specific functional class. In other words its output is the set of predictions made separately by the base learners without any correction due to the hierarchical relationships between the classes. The hierarchical *Top-down* algorithm classifies an example x, where $d_i(x)$ is the classifier decision at node i and $root(T)$ denotes the set of nodes at the first level of the tree T, in the following way:

$$y_i = \begin{cases} d_i(x) & \text{if } i \in root(T) \\ d_i(x) & \text{if } i \notin root(T)\ AND\ y_{par(i)} = 1 \\ 0 & \text{if } i \notin root(T)\ AND\ y_{par(i)} = 0 \end{cases}$$

For each ensemble we used two different types of SVM base learners (linear and gaussian). The probabilistic output of the SVMs composing *TPR* ensembles has been computed using the sigmoid fitting proposed in [30]. The performance of the ensembles have been compared using 5-fold cross-validation techniques. The selection of the w parameter in *TPR-w* ensembles has been performed by internal cross-validation. The threshold t of *TPR* ensembles has been set to 0.5 in all the experiments.

3.1 Data sets

For the prediction of gene function in the yeast we used 7 bio-molecular data sets. For each data set we selected only the genes annotated to FunCat [1], using the *HCgene* R package [39]. We also removed the

1. We used the funcat-2.1 scheme, and funcat-2.1_data_20070316 data, available from the MIPS web site (http://mips.gsf.de/projects/funcat).

genes annotated only with the "99" FunCat class ("Unclassified proteins") and selected classes with at least 20 positive examples, in order to get a not too small set of positive examples for training. From the data sets we removed also uninformative features (e.g. features with the same value for all the available examples). At the end of these pre-processing steps we obtained data whose characteristics are summarized in Tab. 1.

TABLE 1
Biomolecular data sets used in the experiments; n.feat stands for the number of features.

Data set	Description	n.samples	n. feat.	n.class
Pfam-1	protein domain binary data from *Pfam* [40]	3529	4950	211
Pfam-2	protein domain log E data from *Pfam* [41]	3529	5724	211
Phylo	phylogenetic data [33]	2445	24	187
Expr	gene expression data [42], [43]	4532	250	230
PPI-BG	PPI data from *BioGRID* [44]	4531	5367	232
PPI-VM	PPI data from von Mering experiments [45]	2338	2559	177
SP-sim	Sequence pairwise similarity data [10]	3527	6349	211

Pfam-1 data have been originally analyzed by Deng et al. [40]: for each gene product the presence or absence of 4950 protein domains obtained from the *Pfam* (Protein families) database [41] is stored as a binary vector. Moreover we used also an enriched representation of Pfam domains (Pfam-2), by replacing the binary scoring with log E-values obtained with the HMMER software toolkit [46].

Phylogenetic data (Phylo) are obtained through BLAST searches: each feature corresponds to the negative logarithm of the lowest E-value reported by BLAST version 2.0 in a search against a complete genome, with negative values (corresponding to E-values greater than 1) truncated to 0 [33].

We merged the experiments of Spellman et al. (gene expression measures relative to 77 conditions) [42] with the transcriptional responses of yeast to environmental stress (173 conditions) by Gasch et al. [43] to obtain the gene expression (Expr) data set.

Protein-protein interaction data (PPI-BG) have been downloaded from the *BioGRID* database, that collects PPI data from both high-throughput studies and conventional focused studies [44]. BioGRID houses high-throughput two-hybrid [47], mass spectrometric protein interaction data [48] and synthetic lethal genetic interactions obtained through synthetic genetic array and molecular barcode methods [49], as well as a vast collection of well-validated physical and genetic interactions from literature. Data are binary: they represent the presence or absence of protein-protein interactions.

We used also another data set of protein-protein interactions (PPI-VM) that collects binary protein-protein interaction data from yeast two-hybrid assay, mass-spectrometry of purified complexes, correlated

mRNA expression and genetic interactions [45]. These data are binary too.

Finally we considered pairwise similarities between yeast genes (SP-sim), by using data collected by William Noble and colleagues [50]. They computed the Smith and Waterman log-E values between all pairs of yeast sequences, obtaining a symmetric matrix that expresses the pairwise similarities between yeast genes.

Different strategies can be chosen to select negative examples for each functional class [51], [39]. In this work negative examples for each class have been selected in such a way that they are not annotated for the class, but belong to the parent class (i.e. positive for the parent class). In this way only negative examples that are not too dissimilar to the positive ones are selected.

3.2 Performance metrics

Considering the unbalance between positive and negative examples, we adopted the classical F-score to jointly take into account the precision and recall of the ensemble for each class of the hierarchy.

Nevertheless, the classical precision and recall measures, conceived for unstructured classification problems, appear to be inadequate to fully address the hierarchical nature of functional annotation. To this end we used the *Hierarchical F-measure* that represents a generalization of the classical F-measure [52].

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. In other words we need specific measures to estimate how far a predicted structured annotation is from a correct one. 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. To capture these characteristics of functional annotations, we should consider how much the entire path from the most specific upward to the more general annotation is correctly predicted or not. 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 to x, and let be $l(P)$ the set of the leaves (nodes without children) of the graph P. Given a leaf $p \in P(x)$, let be $\uparrow p$ the set of ancestors of the node p that belong to $P(x)$, and given a leaf $c \in C(x)$, let be $\uparrow c$ the set of ancestors of the node c that belong to $C(x)$, we can define the following definitions of Hierarchical Precision (HP), Hierarchical Recall (HR) and Hierarchical F-score (HF) [52]:

$$HP = \frac{1}{|l(P(x))|} \sum_{p \in l(P(x))} \max_{c \in l(C(x))} \frac{|\uparrow c \cap \uparrow p|}{|\uparrow p|}$$

$$HR = \frac{1}{|l(C(x))|} \sum_{c \in l(C(x))} \max_{p \in l(P(x))} \frac{|\uparrow c \cap \uparrow p|}{|\uparrow c|}$$

$$HF = \frac{2 \cdot HP \cdot HR}{HP + HR} \tag{13}$$

It is easy to see that in the case of the FunCat taxonomy, since it is structured as a tree, we can simplify HP, HR and HF as follows:

$$HP = \frac{1}{|l(P(x))|} \sum_{p \in l(P(x))} \frac{|C(x) \cap \uparrow p|}{|\uparrow p|}$$
$$HR = \frac{1}{|l(C(x))|} \sum_{c \in l(C(x))} \frac{|\uparrow c \cap P(x)|}{|\uparrow c|}$$
$$HF = \frac{2 \cdot HP \cdot HR}{HP + HR} \tag{14}$$

An overall high hierarchical precision is indicative of most predictions being ancestors of the correct predictions, or in other words 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 most predictions are successors of the actual, or 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.

4 RESULTS AND DISCUSSION

The performances of *Flat*, *HTD*, *TPR* and *TPR-w* ensembles have been compared using 5-fold cross-validation techniques, as explained in Sect. 3. We performed two nested cross-validation procedures with *TPR-w* ensembles . The "external" cross-validation has been applied to estimate the generalization capabilities of the ensemble, while the "internal" one to select the best w parameter. In this way the selection of the best value for w was independent of the test data used in the external cross-validation.

We tested the ensembles using the "per-class" and hierarchical F-measures (Sect. 4.1), then we analyzed the performance of the ensembles at each level of the FunCat hierarchy (Sect. 4.2) and we also considered separated subtrees of the overall hierarchy related to the main functional categories of the taxonomy (Sect. 4.3). In Sect. 4.4 the precision/recall characteristics of *TPR-w* ensembles have been studied as a function of the global parameter w. We analyzed also whether the proper choice of the threshold parameter t (Sect. 2.3) can influence the performance of *TPR-w* ensembles (Sect. 4.5) and we investigated also the effectiveness of the *TPR* algorithms using different probabilistic base learners (Sect. 4.6). In Sect. 4.7 we evaluated whether local optimization techniques are able to improve the overall classification results, and finally we discuss the main advantages and limitations of the proposed methods, considering both the theoretical analysis of the proposed algorithms (Sect. 2.4) and the experimental results presented in this section.

4.1 "Per class" and hierarchical F-measure results

At first we compared the performance of ensemble methods considering the "per class" F-measure averaged across all FunCat classes for each data set. The results show that hierarchical methods largely

outperform *Flat* ensembles (Fig. 6). The first seven groups of barplots refer to the 7 data sets used in the experiments (Tab. 1), while the last one reports the results averaged across all the data. Using linear SVM as base learners, *Flat* ensembles obtain an average F-measure across the 7 data sets used in the experiments of 0.15 against respectively 0.22, 0.18 and 0.24 with *HTD*, *TPR* and *TPR-w* ensembles. With gaussian SVMs *Flat* ensembles achieve an average F-measure of 0.14 against respectively 0.18, 0.19 and 0.23 with *HTD*, *TPR* and *TPR-w* ensembles. The comparison of the algorithms using the Wilcoxon signed-ranks test [53] shows that *TPR-w* ensembles outperform all other methods at 0.01 significance level with gaussian base learners, and at 0.02 significance level with linear SVMs, while no significant difference can be detected between *HTD* and *TPR* ensembles. The difference between hierarchical and *Flat* methods is always significant (Tab. 2).

TABLE 2

Comparison between F-scores of ensemble methods across the data sets; each cell shows the p-values computed according to the Wilcoxon signed-ranks test; differences at 0.02 significance level are highlighted in bold.

Linear SVMs			
	Flat	*HTD*	*TPR*
HTD	**0.0195**		
TPR	**0.0195**	0.9453	
TPR-w	**0.0039**	**0.0195**	**0.0039**
Gaussian SVMs			
	Flat	*HTD*	*TPR*
HTD	0.0546		
TPR	**0.0039**	0.3203	
TPR-w	**0.0039**	**0.0078**	**0.0078**

As explained in the experimental set-up (Sect. 3), the F hierarchical measure is a more appropriate performance metric for the hierarchical classification of gene functions. Note that we did not report the results obtained with *Flat* ensembles because in all cases they were very significantly worse than those achieved with hierarchical ensemble methods. Fig. 7 shows that on the average *TPR-w* achieves the best results using both linear and gaussian SVMs as base learners: 0.34 versus 0.25 and 0.29 with respect to *TPR* and *HTD* ensembles using linear SVMs, and 0.32 versus 0.27 and 0.25 with gaussian SVMs [2]. Note that *TPR-w* obtains equal or better results than *HTD* ensembles with respect to all the data sets. More

2. We used a polynomial kernel with SP-sim data, since the gaussian kernel showed convergence problems on this classification task.

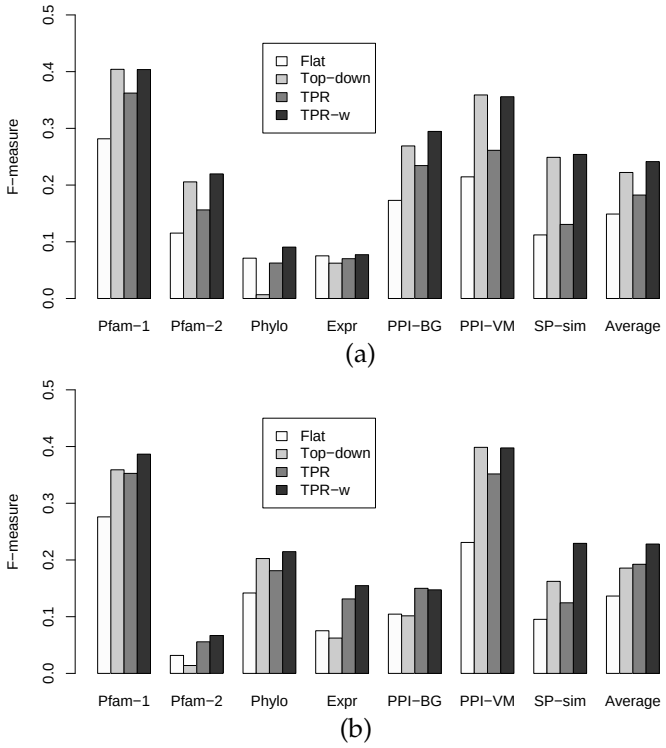

Fig. 6. Per-class F-measure comparison between *Flat*, *Top-down* (*HTD*), *True Path Rule* (*TPR*), and TPR weighted (*TPR-w*) ensembles. Top: linear SVMs; bottom: gaussian SVMs

precisely considering the three repetitions of 5-fold cross validation results for each the 7 considered data sets (21 hierarchical classification tasks), *TPR-w* reported better results than *HTD* at 0.05 significance level on 14 tasks with linear SVMs and on 18 tasks with gaussian SVMs, according the 5-fold cross-validated paired t-test [54]. Comparison between algorithms using the Wilcoxon signed-ranks test show that *TPR-w* ensembles significantly outperform all the other hierarchical methods, while the difference between *Top-down* and *TPR* is not significant (Tab. 3). These results show that we need the weighted version of *TPR* ensembles to enhance *HTD* predictions.

4.2 Per-level F-measure analysis

To get more insights into the reasons why *TPR-w* achieve better results, we performed a per-level analysis of the F-measure of the FunCat trees. Tab. 4 shows the per level F-measure results with Pfam-1 protein domain data and Pairwise sequence similarity data (SP-sim). Level 1 refers to the root nodes of the FunCat hierarchy, level i, to nodes at depth i, $2 \leq i \leq 5$.

Looking at the results, we can observe that *Flat* ensembles tend to have the highest recall, *Top-down* the highest precision, while *TPR-w* tends to stay in the middle with respect to both the recall and precision,

TABLE 3

Comparison between hierarchical F-scores of ensemble methods across the data sets; each cell shows the p-values computed according to the Wilcoxon signed-ranks test; differences at 0.05 significance level are highlighted in bold.

Linear SVMs		
	HTD	*TPR*
TPR	0.9023	
TPR-w	**0.0039**	**0.0039**

Gaussian SVMs		
	HTD	*TPR*
TPR	0.3203	
TPR-w	**0.0039**	**0.0390**

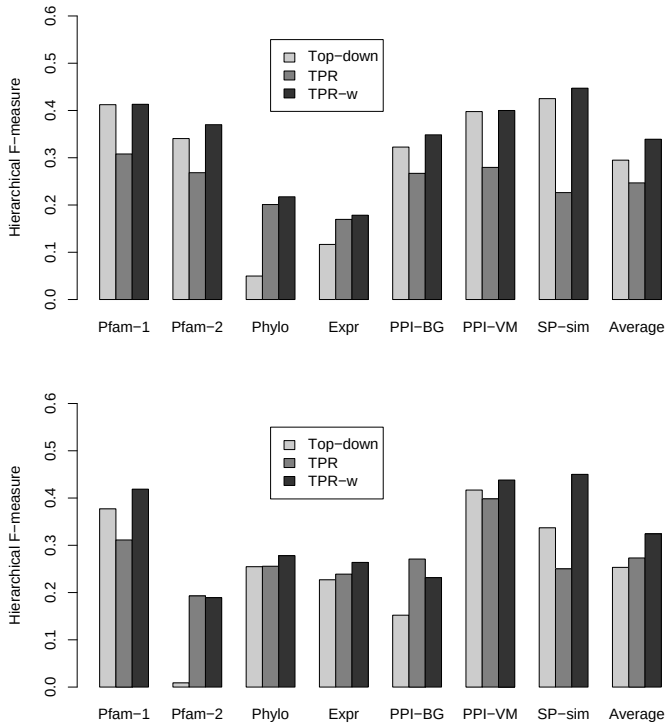

Fig. 7. Hierarchical F-measures comparison between *Top-down* (*HTD*), *True Path Rule* (*TPR*), and TPR weighted (*TPR-w*) ensembles. Top: linear SVMs; bottom: gaussian SVMs

thus achieving the best F-measure at each level.

These results can be explained analyzing the behaviour of *HTD* and *TPR-w* algorithms. Indeed for

TABLE 4

Per level precision, recall, F-measure and accuracy comparison between *Flat*, *Top-down*, *True Path Rule weighted* (*TPR-w*) ensembles. Left: Pfam-1 protein domain data with gaussian SVMs as base learners. Right: SP-sim Pairwise sequence similarity data with polynomial SVMs as base learners.

Protein domain (Pfam-1)					**Sequence similarity (SP-sim)**				
Flat					**Flat**				
Level	Prec.	Rec.	F	Acc.	Level	Prec.	Rec.	F	Acc.
1	**0.7633**	0.3110	0.4337	0.8888	1	**0.5507**	0.4143	0.4703	0.8722
2	0.4027	**0.4753**	0.3505	0.8098	2	0.0822	0.3401	0.1189	0.7454
3	0.3187	**0.4683**	0.2725	0.7711	3	0.0305	**0.2940**	0.0502	0.7380
4	0.1500	**0.6370**	0.1561	0.5400	4	0.0205	**0.4962**	0.0371	0.5251
5	0.1517	**0.3819**	0.1707	0.8567	5	0.0058	**0.2989**	0.0110	0.6857
HTD					**HTD**				
Level	Prec.	Rec.	F	Acc.	Level	Prec.	Rec.	F	Acc.
1	**0.7633**	0.3110	0.4337	0.8888	1	**0.5507**	0.4143	0.4703	0.8722
2	**0.6945**	0.2926	0.3978	0.9597	2	**0.3091**	0.1765	0.2148	0.9440
3	**0.6289**	0.2580	0.3519	0.9792	3	**0.2305**	0.0945	0.1221	0.9718
4	**0.5627**	0.2372	0.3193	0.9811	4	**0.2141**	0.0777	0.0956	0.9746
5	**0.4774**	0.2013	**0.2736**	0.9906	5	0.0498	0.0394	0.0316	0.9871
TPR-w					**TPR-w**				
Level	Prec.	Rec.	F	Acc.	Level	Prec.	Rec.	F	Acc.
1	0.7244	**0.3618**	**0.4727**	0.8911	1	0.4432	**0.5573**	**0.4898**	0.8489
2	0.6445	0.3400	**0.4319**	0.9597	2	0.2723	**0.4044**	**0.3156**	0.9273
3	0.5656	0.3052	**0.3812**	0.9787	3	0.1949	0.2384	**0.1995**	0.9621
4	0.5245	0.2764	**0.3430**	0.9804	4	0.1579	0.1973	**0.1532**	0.9673
5	0.4547	0.2143	0.2623	0.9901	5	**0.0503**	0.1254	**0.0695**	0.9846

both hierarchical algorithms we have a "negative" flow of information from top to bottom: when a node answers "no", its prediction is propagated to the subtree below, and the specificity and the precision are increased. Moreover *TPR-w* shows also a positive flow of information in the opposite direction: "yes" answers of the descendants nodes influence the decisions of the parent nodes, thus improving the recall of the overall system. Indeed *TPR-w* recall tends to be larger than the corresponding *Top-down* recall at each level, and sometimes even larger than the recall of *Flat* ensembles. In other words the two fluxes of information tend to balance precision and recall, thus resulting in an improved F-measure at each level of the FunCat hierarchy (Tab. 4).

Note that the accuracy is high at each level (at least with hierarchical ensemble methods), but these results are not significant, considering the large unbalance between positive and negative genes for each

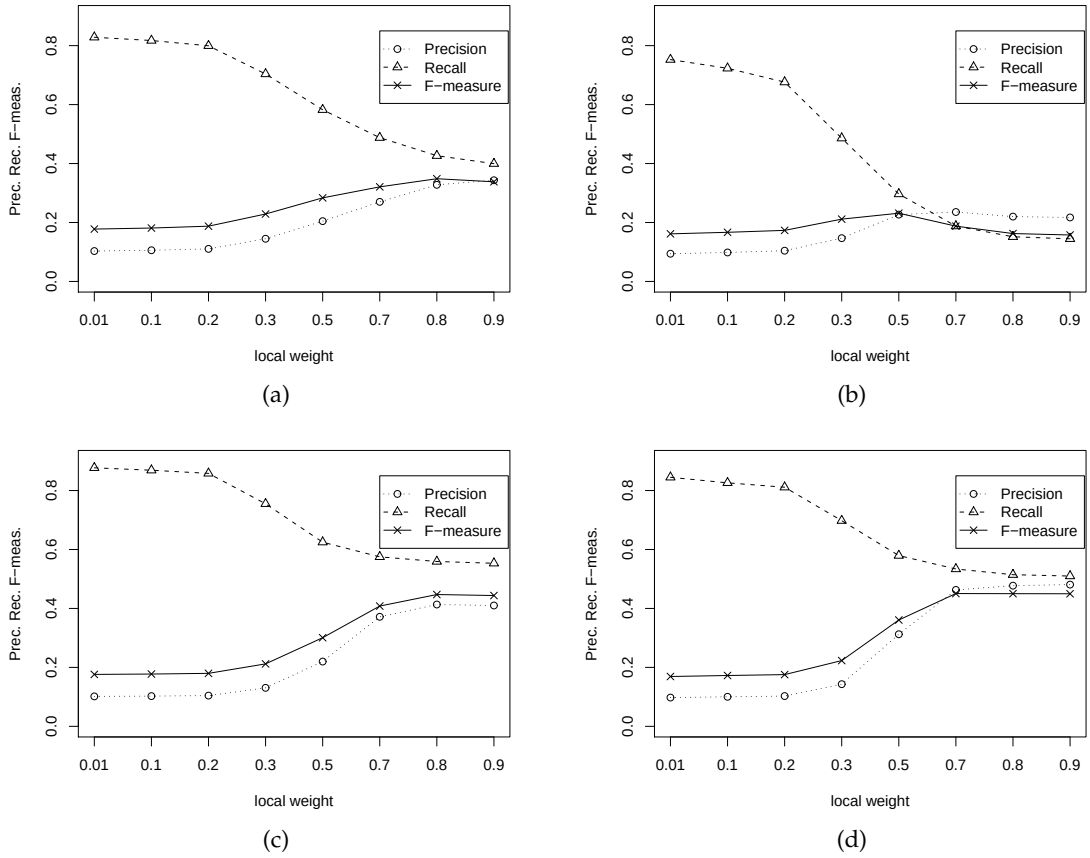

Fig. 8. Precision, Recall and F-measure as a function of the parent weight in *TPR-w* ensembles. PPI BioGRID data: (a) linear (b) gaussian kernel; Pairwise sequence similarity data: (c) linear (d) polynomial kernel.

functional class (Tab. 4).

4.3 Classification of specific subtrees

Even if the main goal of this work consists in the development of a hierarchical algorithm that can be applied to the prediction of the overall taxonomy of a gene, we can restrict the analysis to specific subtrees of the taxonomy. For instance, with the tree rooted at the "protein fate" FunCat class (FunCat ID = 14), composed by 15 nodes (Fig. 9), using Pfam-1 data we obtain an average precision equal to 0.79, an average recall of 0.48, and an average F-measure of 0.58 (Tab. 5), and for several classes we obtain an F-measure larger than 0.70, even if only 1 source of data is used for the hierarchical classification. Other results relative to specific subtrees of the FunCat taxonomy are available in the Supplementary Information. It is worth noting that for each subtree and more in general for each functional class the results largely depend on the choice of the data set. Indeed each type of biomolecular data captures

specific characteristics of genes that may correspond to different functional features. For instance, gene expression data obtained from time series experiments can be informative to classify cell-cycle related classes, while are less informative or unuseful to classify several other classes (e.g. classes related to signal transduction and interaction with the environment).

Fig. 9. Tree of the FunCat classes rooted at Funcat ID=14 (Protein fate).

4.4 Tuning precision and recall in *TPR-w* ensembles

Another advantage of the *TPR-w* ensembles is the capability of tuning precision and recall rates, through the parameter parent weight w (eq. 5). Fig. 8 shows, the hierarchical precision, recall and F-measure as functions of the parameter w. For small values of w (w can vary from 0 to 1) 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): as a consequence we obtain a higher hierarchical recall for the *TPR-w* ensemble. On the contrary higher values of w correspond to a higher weight of the parent predictor, with a resulting higher precision. The opposite trends of precision and recall are quite clear in all graphs of Fig. 8. The best F-score is in "middle" values of the parameter parent weight: in practice in most of the analyzed data sets the best F-measure is achieved for w between 0.5 and 0.8, but if we need higher recall rates (at the expense of the precision) we can choose lower w values, and higher values of w are needed if precision is our first aim. Other results and graphs are available on-line in the Supplementary Information.

TABLE 5

Classification results on the FunCat tree rooted at "Protein fate". Each row represents a functional class of the FunCat taxonomy. Prec. stands for precision, Rec. recall, Sp. specificity, F F-measure, Acc. accuracy.

FunCat ID	Description	Prec.	Rec.	Sp.	F	Acc.
14	Protein fate (folding, modification, destination)	0.8342	0.5182	0.9645	0.6393	0.8503
14.01	protein folding and stabilization	0.8437	0.6585	0.9970	0.7397	0.9892
14.04	protein targeting, sorting and translocation	0.6224	0.2760	0.9888	0.3824	0.9441
14.07	protein modification	0.8287	0.5753	0.9816	0.6791	0.9274
14.07.01	modification with fatty acids	0.8333	0.2083	0.9997	0.3333	0.9943
14.07.02	modification with sugar residues	0.9090	0.4615	0.9991	0.6122	0.9892
14.07.02.02	N-directed glycosylation, deglycosylation	0.7826	0.4390	0.9985	0.5625	0.9920
14.07.03	modification by phosphorylation	0.8717	0.7445	0.9955	0.8031	0.9858
14.07.04	modification by acetylation	0.8823	0.3061	0.9994	0.4545	0.9897
14.07.05	modification by ubiquitination	0.8846	0.6865	0.9982	0.7731	0.9923
14.07.11	protein processing (proteolytic)	0.7142	0.3846	0.9971	0.5000	0.9858
14.10	assembly of protein complexes	0.7555	0.2048	0.9967	0.3222	0.9594
14.13	protein/peptide degradation	0.7986	0.5609	0.9912	0.6590	0.9662
14.13.01	cytoplasmic and nuclear protein degradation	0.6940	0.5886	0.9878	0.6369	0.9699
14.13.01.01	proteasomal degradation	0.6629	0.5619	0.9912	0.6082	0.9784

4.5 Choice of the decision threshold and its impact on the performance of *TPR-w* ensembles

The output of *TPR* ensembles is probabilistic, but the decision $d_i(x)$ of the ensemble at node i for a given gene x depends on the threshold t (see lines 05 or 10 of the pseudocode of the algorithm, Fig. 2). A reasonable choice for t is 0.5, but in this section we analyze the behaviour of the *TPR* algorithm by varying the value of the threshold t between 0 and 1, in order to understand whether a proper choice of t could improve the performance of the ensemble. To this end, we studied the hierarchical precision, recall and F-score of two *TPR-w* ensembles trained respectively with linear and gaussian SVMs on the data sets Pfam-1 and PPI-VMs (Sect.3.1), by varying the decision threshold between 0.05 and 0.95. Fig. 10 shows that the best results in terms of the F hierarchical score (solid lines with crosses) are achieved when the threshold is close to 0.5: more precisely 0.45 with both Pfam-1 and PPI-VM data. It is worth noting that with values in the range $[0.4, 0.6]$ we obtain in any case comparable results. As expected, the hierarchical recall decreases monotonically by increasing the threshold, while the precision at first increases, then reaches a quite "smooth" maximum and then decreases for the largest values of the threshold. The "smooth" maximum is due to the joint effect of the reduction of both false and true positives with the increment of the threshold, with opposite effects on the precision. Of course, when we have large values of the threshold, both precision and recall undergo a substantial reduction that leads to small values of the F-score. Summarizing, choosing a threshold close to 0.5 seems to be an acceptable choice, even if

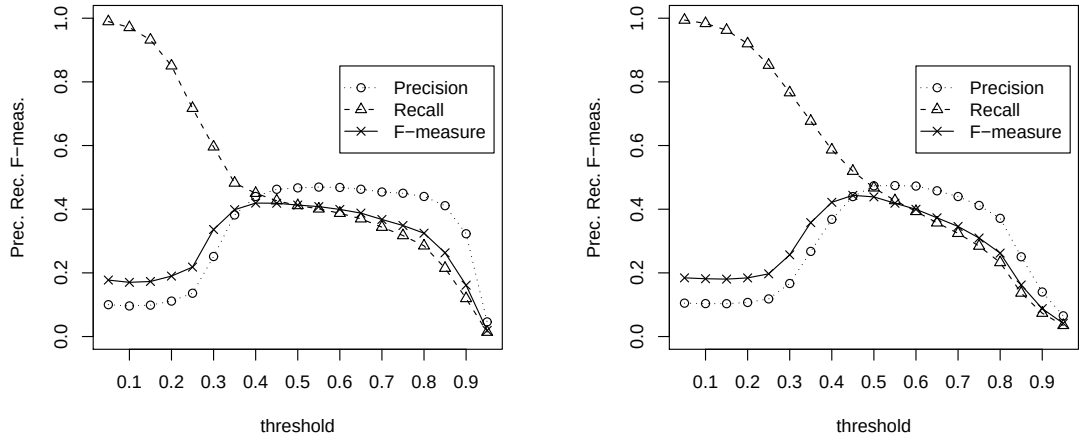

Fig. 10. Precision, Recall and F-measure as a function of the decision threshold t in *TPR-w* ensembles. (a) Pfam-1 data (b) PPI-VM data

optimizing this parameter can lead to a slight improvement of the performance of the ensemble.

4.6 Using logistic regression classifiers as base learners

The proposed hierarchical ensembles can use different probabilistic classifiers as base learners. In this section we compare the results obtained with *Flat*, *Top-down*, *TPR* and *TPR-w* ensembles using probabilistic SVMs and logistic regression models as base classifiers. Logistic regression classifiers directly estimate the posterior probabilities of the classes [55], while probabilistic SVMs model posterior probabilities by fitting a sigmoid on the output of the discriminant function computed by the SVM algorithm [30]. The 5-fold cross-validation results with Expr and PPI-VM data show that *TPR-w* ensembles achieve equal or better results than those obtained with the other methods, independently of the base learner used (Tab. 6). In particular, on these two data sets, logistic regression base classifiers behave better than probabilistic linear SVMs, and sometimes also better than probabilistic radial SVMs (e.g. with PPI-VM data, Tab. 6). Summarizing, these results show that the proposed *TPR* ensemble methods can be applied using different probabilistic base learners.

4.7 Testing local optimization techniques

Finally we tested whether local optimization techniques can improve the overall performance of the *TPR* ensembles. We analyzed the results obtained through: a) computationally intensive "per node" model selection techniques b) local cost-sensitive strategies. In both cases we used linear SVMs as base learners

TABLE 6

Comparison of hierarchical F-measure results using different probabilistic base learners with *Flat*, *Top-down*, *TPR* and *TPR-w* ensembles.

Gene Expression data (Expr)				
	Flat	*HTD*	*TPR*	*TPR-w*
Radial SVM	0.1153	0.2270	0.2390	0.2638
Linear SVM	0.1246	0.1166	0.1696	0.1784
Log. regression	0.1384	0.1672	0.1859	0.2453
Protein-protein interaction data (PPI-VM)				
	Flat	*HTD*	*TPR*	*TPR-w*
Radial SVM	0.1720	0.4169	0.3983	0.4381
Linear SVM	0.1658	0.3977	0.2796	0.4000
Log. regression	0.1833	0.4382	0.3392	0.4553

and only 2 data sets: PPI-VM and Phylo. Results obtained with 5-fold cross validation and internal 3-fold cross validation for model selection did not result in an improvement of the overall performance of the hierarchical ensembles. This could seem quite surprising, but in this context, where examples can be at least partially mislabeled and data can be noisy (at least for certain data sets relatively to specific functional classes) a too intensive model selection can lead to overfitting. On the contrary, using local cost-sensitive strategies, we can achieve a certain improvement of the overall performances. More precisely we imposed different costs for misclassifications of positive and negative examples for training linear SVMs, simply by setting the cost for misclassification of negative examples to 1 and the cost of misclassification of positive examples to the ratio between negative and positive examples in the training set. With PPI-VM data we observed a slight increment from 0.40 to 0.41 of the F-hierarchical measure with *TPR-w* ensembles and a slight decrement from 0.28 to 0.27 with *TPR* ensembles. With phylogenetic data the increment is from 0.22 to 0.24 with *TPR-w* and from 0.20 to 0.21 with *TPR* ensembles.

These results (no influence of the C regularization factor and a certain impact of local cost-sensitive strategies) confirm previous findings observed with the classification of GO hierarchies [28].

4.8 Advantages and limitations of the *TPR-based* algorithms

F hierarchical measures 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, using the classical "per-class" F-measure, while no significant difference, according to the Wilcoxon signed-ranks test can be registered between *Top-down* and the basic *TPR* algorithm.

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.

We can note for both hierarchical algorithms a degradation of both precision and recall (and as a consequence of the F-measure) by descending the levels of the trees. This fact could be at least in part due to the lack of annotations at the lowest levels of the hierarchy, where we may have several genes with unannotated specific functions. We also conjecture that experimenting with different strategies to assign negative examples in the training phase could improve the performance at the lower levels of the hierarchy [51].

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 the recall. The choice to favor precision or recall depends on the researcher's experimental objectives. In most data sets the best compromise between precision and recall is achieved for weights in the range between 0.5 and 0.8, that is giving a weight equal or larger to the parent predictor with respect to the predictions taken by its offsprings. In other words we may obtain different precision-recall curves for different values of w: the weight is a global parameter that affects the general precision/recall characteristics of the ensemble.

It is worth noting that we may vary the threshold t to obtain precision recall curves for a fixed value of w: this can slightly enhance the overall performance of *TPR-w* ensembles. Moreover *TPR* and *TPR-w* ensembles provide also a probabilistic estimate of the prediction reliability for each functional class of the overall taxonomy.

In Sect. 2.4 we discussed how the decisions performed at each node of the hierarchical ensemble are influenced by the positive decisions of its descendants. The results of this analysis can be summarized as follows:

- Weights of descendants decrease exponentially w.r.t their depth. As a consequence the influence of descendant nodes decays quickly with their depth.
- The parameter w plays a central role in balancing the weight of the parent classifier associated to a given node with the weights of its positive offsprings: small values of w increase the weight of descendant nodes, large values the weight of the local parent predictor associated to that node.
- 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. Different behaviors of the ensemble have been characterized according to some general rules by which probabilities are assigned to the positive offsprings (Sect. 2.4, Fig. 5).

These characteristics of *TPR-w* ensembles are well-suited for the hierarchical classification of gene 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).

Nevertheless 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, for the reasons discussed above, 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. The proposed algorithm weights the positive predictions of deeper nodes with an exponentially decrement with respect to their depth (Sect. 2.4), 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.

Another possible research line could consist in the integration of the recently proposed hierarchical method based on isotonic regression [6] with the true path rule that governs both the GO and FunCat taxonomies. Indeed, this more general and principled approach to hierarchical classification could overcome the heuristic nature of the *TPR* algorithm, by adapting isotonic regression to the biological characteristics of gene function taxonomies.

5 Conclusions

In this paper we proposed a new hierarchical strategy, inspired by the true path rule, for gene function prediction extended to the overall functional taxonomy of genes.

TPR-w ensembles significantly outperform both the basic *TPR* and *Top-down* ensembles in the genome and ontology-wide prediction of gene functions in *S. cerevisiae*. The analysis of the experimental results and a theoretical investigation of the flow of information that traverses the hierarchical ensemble show the reasons why *TPR-w* are well-suited to the prediction of gene functions, and suggest new research lines for the development of new hierarchy-aware gene function prediction methods.

The overall results show that using a single source of evidence we can obtain a high precision and recall for specific trees of the FunCat forest. Nevertheless, we need to integrate multiple data sources to obtain methods to predict functions of hypothetical genes, or to discover or complete the functional annotation of genes whose function is incomplete or unknown. To this end, the proposed approach can be easily integrated with at least three different general strategies for biomolecular data integration: vector space integration [33], kernel fusion [10] and ensemble methods [56]. Indeed for each node/class of the tree we may substitute a classifier trained on a specific type of biomolecular data with a classifier trained on concatenated vectors of different data, or trained on a (weighted) sum of kernels, or with an ensemble of learners each trained on a different type of data. This is the object of our planned future research.

Acknowledgments

The author would like to thank the anonymous reviewers for their comments and suggestions, and gratefully acknowledges partial support by the PASCAL2 Network of Excellence under EC grant no. 216886. This publication only reflects the author's views.

APPENDIX A

PROOF OF PROPOSITION 1: INFLUENCE OF POSITIVE DESCENDANT NODES WITH ONE CHILD

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 exactly one positive descendant for each of the m lower levels below, that is such that $q_{k+j} \geq 0.5, 1 \leq j \leq m$, the following equality holds for each $m \geq 1$:

$$q_k = \sum_{j=0}^{m-1} w(1-w)^j \hat{q}_{k+j} + (1-w)^m q_{k+m} \tag{15}$$

Proof: The proof is by induction. Considering that in *TPR-w* ensembles for a given node at level k we have (eq.7):

$$q_k(x) = w \cdot \hat{q}_k(x) + \frac{1-w}{|\phi_k(x)|} \sum_{j \in \phi_k(x)} q^j_{k+1}(x)$$

for $m = 1$, having 1 positive child for a node at level k, we obtain:

$$q_k = w\hat{q}_k + (1-w)q_{k+1} \tag{16}$$

and hence eq. 15 holds for $m = 1$.

We suppose that eq. 15 is true for $m = n$:

$$q_k = \sum_{j=0}^{n-1} w(1-w)^j \hat{q}_{k+j} + (1-w)^n q_{k+n} \tag{17}$$

We show now that the equality holds also for $m = n+1$. Indeed for the node at level $k+n$ we have:

$$q_{k+n} = w\hat{q}_{k+n} + (1-w)q_{k+n+1} \tag{18}$$

By substituting eq. 18 in eq. 17 we obtain:

$$\begin{aligned} q_k &= \sum_{j=0}^{n-1} w(1-w)^j \hat{q}_{k+j} + (1-w)^n (w\hat{q}_{k+n} + (1-w)q_{k+n+1}) \\ &= \sum_{j=0}^{n-1} w(1-w)^j \hat{q}_{k+j} + w(1-w)^n \hat{q}_{k+n} + (1-w)^{n+1} q_{k+n+1} \\ &= \sum_{j=0}^{n} w(1-w)^j \hat{q}_{k+j} + (1-w)^{n+1} q_{k+n+1} \end{aligned} \tag{19}$$

□.

APPENDIX B

COMPUTATION OF THE MAXIMUM OF $g(w)$

Considering the function $g(w) = w(1-w)^j$, $w \in [0,1], j \leq 1$, we have that $\arg\max g(w) = \frac{1}{j+1}$. Indeed:

$$\frac{dg}{dw} = (1-w)^j - jw(1-w)^{j-1} = (1-w)^{j-1}(1-w-jw)$$
$$\frac{dg}{dw} = 0 \Longleftrightarrow \big((1-w)^{j-1} = 0\big) \vee ((1-w-jw) = 0)$$
$$(1-w)^{j-1} = 0 \Longleftrightarrow (w=1) \vee (j \to \infty, w \neq 0, w \neq 1)$$
$$(1-w-jw) = 0 \Longleftrightarrow 1-w(1+j) = 0 \Longleftrightarrow w(1+j) = 1 \Longleftrightarrow w = \frac{1}{j+1}$$

Hence we have:

$$\frac{dg}{dw} = 0 \Longleftrightarrow \begin{cases} w = 1 \\ w \neq 0, w \neq 1, j \to \infty \\ w = \frac{1}{j+1} \end{cases}$$

It is easy to see that $w = 1$ is a minimum, $w \neq 0, w \neq 1, j \to \infty$ is another minimum, while $w = \frac{1}{j+1}$ is a maximum.

APPENDIX C

PROOF OF PROPOSITION 2: INFLUENCE OF POSITIVE DESCENDANT NODES WITH A VARIABLE NUMBER OF CHILDREN

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 or equal than 1 of positive descendant for each of the m lower levels below, the following equality holds for each $m \geq 1$:

$$q_k = w\hat{q}_k + \sum_{j=1}^{m-1} w(1-w)^j a_{k+j} + (1-w)^m a_{k+m} \tag{20}$$

Proof: If a generic node at level k has only one level of positive descendants ($m = 1$), we have, by using eq. 8:

$$q_k = w\hat{q}_k + \frac{1-w}{|\phi_k|} \sum_{j \in \phi_k} \hat{q}^j_{k+1} = w\hat{q}_k + (1-w)\ a_{k+1} \tag{21}$$

If k has two levels of positive descendants ($m = 2$), by using eq. 9, we have:

$$q_k = w\hat{q}_k + \frac{1-w}{|\phi_k|} \sum_{j \in \phi_k} \hat{q}^j_{k+1} \tag{22}$$

$$q^j_{k+1} = w\hat{q}^j_{k+1} + \frac{1-w}{|\phi^j_{k+1}|} \sum_{l \in \phi^j_{k+1}} \hat{q}^l_{k+2} \tag{23}$$

By putting eq. 23 in eq. 22:

$$\begin{aligned} q_k &= w\hat{q}_k + \frac{1-w}{|\phi_k|}\sum_{j\in\phi_k}\left(w\hat{q}^j_{k+1} + \frac{1-w}{|\phi^j_{k+1}|}\sum_{l\in\phi^j_{k+1}}\hat{q}^l_{k+2}\right) \\ &= w\hat{q}_k + \frac{w(1-w)}{|\phi_k|}\sum_{j\in\phi_k}\hat{q}^j_{k+1} + \frac{(1-w)^2}{|\phi_k|}\sum_{j\in\phi_k}\frac{1}{|\phi^j_{k+1}|}\sum_{l\in\phi^j_{k+1}}\hat{q}^l_{k+2} \end{aligned} \tag{24}$$

By using eq. 8 and 9 we finally obtain:

$$q_k = w\hat{q}_k + w(1-w)\,a_{k+1} + (1-w)^2\,a_{k+2} \tag{25}$$

We can repeat this procedure for m levels below a node at level k:

$$\begin{aligned} q_k &= w\hat{q}_k + w(1-w)\,a_{k+1} + w(1-w)^2\,a_{k+2} + \ldots + w(1-w)^{m-1}\,a_{k+m-1} + (1-w)^m\,a_{k+m} \\ &= w\hat{q}_k + \sum_{j=1}^{m-1} w(1-w)^j a_{k+j} + (1-w)^m a_{k+m} \end{aligned} \tag{26}$$

□.

REFERENCES

[1] I. Friedberg, "Automated protein function prediction-the genomic challenge," *Brief. Bioinformatics*, vol. 7, pp. 225–242, 2006.

[2] L. Pena-Castillo *et al.*, "A critical assessment of Mus musculus gene function prediction using integrated genomic evidence," *Genome Biology*, vol. 9, p. S1, 2008.

[3] H. Kriegel, P. Kroger, A. Pryakhin, and M. Schubert, "Using support vector machines for classifying large sets of multi-represented objects," in *Proc. 4th SIAM Int. Conf. on data Mining*, 2004, pp. 102–114.

[4] G. Tsoumakas and I. Katakis, "Multi label classification: An overview," *International Journal of Data Warehousing and Mining*, vol. 3, no. 3, pp. 1–13, 2007.

[5] A. Dimou, G. Tsoumakas, V. Mezaris, I. Kompatsiaris, and I. Vlahavas, "An empirical study of multi-label methods for video annotation," in *Proc. 7th International Workshop on Content-Based Multimedia Indexing, CBMI 09*, Chania, Greece, 2009.

[6] K. Punera and J. Ghosh, "Enhanced hierarchical classification via isotonic smoothing," in *WWW 2008*. Beijing, China: ACM, 2008, pp. 151–160.

[7] 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.

[8] The Gene Ontology Consortium, "Gene ontology: tool for the unification of biology," *Nature Genet.*, vol. 25, pp. 25–29, 2000.

[9] A. Ruepp, A. Zollner, D. Maier, K. Albermann, J. Hani, M. Mokrejs, I. Tetko, U. Guldener, G. Mannhaupt, M. Munsterkotter, and H. Mewes, "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.

[10] G. Lanckriet, T. De Bie, N. Cristianini, M. Jordan, and W. Noble, "A statistical framework for genomic data fusion," *Bioinformatics*, vol. 20, pp. 2626–2635, 2004.

[11] O. Troyanskaya *et al.*, "A Bayesian framework for combining heterogeneous data sources for gene function prediction (in *saccharomices cerevisiae*)," *Proc. Natl Acad. Sci. USA*, vol. 100, pp. 8348–8353, 2003.

[12] K. Tsuda, H. Shin, and B. Scholkopf, "Fast protein classification with multiple networks," *Bioinformatics*, vol. 21, no. Suppl 2, pp. ii59–ii65, 2005.

[13] U. Karaoz *et al.*, "Whole-genome annotation by using evidence integration in functional-linkage networks," *Proc. Natl Acad. Sci. USA*, vol. 101, pp. 2888–2893, 2004.

[14] H. Chua, W. 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.

[15] J. Xiong *et al.*, "Genome wide prediction of gene function via a generic knowledge discovery approach based on evidence integration," *BMC Bioinformatics*, vol. 7, no. 268, 2006.

[16] W. Tian, L. Zhang, M. Tasan, F. Gibbons, O. King, J. Park, Z. Wunderlich, J. Cherry, and F. Roth, "Combining guilt-by-association and guilt-by-profiling to predict saccharomices cerevisiae gene function," *Genome Biology*, vol. 9, p. S7, 2008.

[17] W. Kim, C. Krumpelman, and E. Marcotte, "Inferring mouse gene functions from genomic-scale data using a combined functional network/classification strategy," *Genome Biology*, vol. 9, p. S5, 2008.

[18] A. Sokolov and A. Ben-Hur, "A structured-outputs method for prediction of protein function," in *MLSB08, the Second International Workshop on Machine Learning in Systems Biology*, 2008.

[19] K. Astikainen, L. Holm, E. Pitkanen, S. Szedmak, and J. Rousu, "Towards structured output prediction of enzyme function," *BMC Proceedings*, vol. 2, no. Suppl 4:S2, 2008.

[20] B. Done, P. Khatri, A. Done, and S. Draghici, "Predicting novel human gene ontology annotations using semantic analysis," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 2008, (in press) available on line http://doi.ieeecomputersociety.org/10.1109/TCBB.2008.29.

[21] R. Eisner, B. Poulin, D. Szafron, and P. Lu, "Improving protein prediction using the hierarchical structure of the Gene Oontology," in *IEEE Symposium on Computational Intelligence in Bioinformatics and Computational Biology*, 2005.

[22] H. Blockeel, L. Schietgat, and A. Clare, "Hierarchcial 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. Tuusula, Finland: Helsinki University Printing House, 2006.

[23] B. Shahbaba and M. Neal, "Gene function classification using Bayesian models with hierarchy-based priors," *BMC Bioinformatics*, vol. 7, no. 448, 2006.

[24] C. Vens, J. Struyf, L. Schietgat, S. Dzeroski, and H. Blockeel, "Decision trees for hierarchical multi-label classification," *Machine Learning*, vol. 73, pp. 185–214, 2008.

[25] G. Obozinski, G. Lanckriet, C. Grant, J. M., and W. Noble, "Consistent probabilistic output for protein function prediction," *Genome Biology*, vol. 9, no. S6, 2008.

[26] Gene Ontology Consortium, "True path rule," 2009, http://www.geneontology.org/GO.usage.shtml#truePathRule.

[27] X. Jiang, N. Nariai, M. Steffen, S. Kasif, and E. Kolaczyk, "Integration of relational and hierarchical network information for protein function prediction," *BMC Bioinformatics*, vol. 9, no. 350, 2008.

[28] Y. Guan, C. Myers, D. Hess, Z. Barutcuoglu, A. Caudy, and O. Troyanskaya, "Predicting gene function in a hierarchical context with an ensemble of classifiers," *Genome Biology*, vol. 9, no. S2, 2008.

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

[30] H. Lin, C. Lin, and R. Weng, "A note on Platt's probabilistic outputs for support vector machines," *Machine Learning*, vol. 68, pp. 267–276, 2007.

[31] A. Valencia, "Automatic annotation of protein function," *Curr. Opin. Struct. Biol.*, vol. 15, pp. 267–274, 2005.

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

[33] P. Pavlidis, J. Weston, J. Cai, and W. Noble, "Learning gene functional classification from multiple data," *J. Comput. Biol.*, vol. 9, pp. 401–411, 2002.

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

[35] N. Cesa-Bianchi and G. Valentini, "Hierarchical cost-sensitive algorithms for genome-wide gene function prediction," in *Machine Learning in Systems Biology, Proceedings of the Third international workshop*, Ljubljana, Slovenia, 2009.

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

[37] J. Rousu, C. Saunders, S. Szdemak, and J. Shawe-Taylor, "Learning hierarchical multi-category text classification models," in *Proc. of the 22nd Int. Conf. on Machine Learning*. Omnipress, 2005, pp. 745–752.

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

[39] 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.

[40] M. Deng, T. Chen, and F. Sun, "An integrated probabilistic model for functional prediction of proteins," in *Proc 7th Int Conf Comp Mol Biol*, 2003, pp. 95–103.

[41] R. Finn, J. Tate, J. Mistry, P. Coggill, J. Sammut, H. Hotz, G. Ceric, K. Forslund, S. Eddy, E. Sonnhammer, and A. Bateman, "The Pfam protein families database," *Nucleic Acids Research*, vol. 36, pp. D281–D288, 2008.

[42] P. Spellman *et al.*, "Comprehensive identification of cell cycle-regulated genes of the yeast Saccharomices cerevisiae by microarray hybridization," *Mol. Biol. Cell*, vol. 9, pp. 3273–3297, 1998.

[43] P. Gasch *et al.*, "Genomic expression programs in the response of yeast cells to environmental changes," *Mol.Biol.Cell*, vol. 11, pp. 4241–4257, 2000.

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

[45] C. von Mering, R. Krause, B. Snel, M. Cornell, S. Oliver, S. Fields, and P. Bork, "Comparative assessment of large-scale data sets of protein-protein interactions," *Nature*, vol. 417, pp. 399–403, 2002.

[46] S. Eddy, "Profile hidden markov models," *Bioinformatics*, vol. 14, no. 9, pp. 755–763, 1998.

[47] P. Uetz, L. Giot, G. Cagney, T. Mansfield, R. Judson, J. Knight, D. Lockshon, V. Narayan, M. Srinivasan, P. Pochart *et al.*, "A comprehensive analysis of protein-protein interactions in Saccharomyces cerevisiae," *Nature*, vol. 403, pp. 623–627, 2000.

[48] Y. Ho, A. Gruhler, A. Heilbut, G. Bader, L. Moore, S. Adams, A. Millar, P. Taylor, K. Bennett, K. Boutilier *et al.*, "Systematic identification of protein complexes in Saccharomyces cerevisiae by mass spectrometry," *Nature*, vol. 415, pp. 180–183, 2002.

[49] A. Davierwala, J. Haynes, Z. Li, R. Brost, M. Robinson, L. Yu, S. Mnaimneh, H. Ding, H. Zhu, Y. Chen *et al.*, "The synthetic genetic interaction spectrum of essential genes," *Nature Genet*, vol. 37, pp. 1147–1152, 2005.

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

[51] A. Ben-Hur and W. Noble, "Choosing negative examples for the prediction of protein-protein interactions," *BMC Bioinformatics*, vol. 7, no. Suppl 1/S2, 2006.

[52] K. Verspoor, J. Cohn, S. Mnizewski, and C. Joslyn, "A categorization approach to automated ontological function annotation," *Protein Science*, vol. 15, pp. 1544–1549, 2006.

[53] J. Demsar, "Statistical comparisons of classifiers over multiple data sets," *Journal of Machine Learning Research*, vol. 7, pp. 1–30, 2006.

[54] T. Dietterich, "Approximate statistical test for comparing supervised classification learning algorithms," *Neural Computation*, vol. 10, no. 7, pp. 1895–1924, 1998.

[55] T. Hastie, R. Tibshirani, and R. Friedman, *The Elements of Statistical Learning*. New York: Springer, 2001.

[56] M. Re and G. Valentini, "Simple ensemble methods are competitive with state-of-the-art data integration methods for gene function prediction," in *Machine Learning in Systems Biology, Proceedings of the Third international workshop*, Ljubljana, Slovenia, 2009, pp. 95–104.