Multitask Protein Function Prediction Through Task Dissimilarity

Marco Frasca, Nicolò Cesa Bianchi

Abstract—Automated protein function prediction is a challenging problem with distinctive features, such as the hierarchical organization of protein functions and the scarcity of annotated proteins for most biological functions. We propose a multitask learning algorithm addressing both issues. Unlike standard multitask algorithms, which use task (protein functions) similarity information as a bias to speed up learning, we show that dissimilarity information enforces separation of rare class labels from frequent class labels, and for this reason is better suited for solving unbalanced protein function prediction problems. We support our claim by showing that a multitask extension of the label propagation algorithm empirically works best when the task-relatedness information is represented using a dissimilarity matrix as opposed to a similarity matrix. Moreover, the experimental comparison carried out on three model organism shows that our method has a more stable performance in both “protein-centric” and “function-centric” evaluation settings.

Index Terms—Multitask learning; protein function prediction; label propagation algorithm; Gene Ontology; task dissimilarity.

1 Introduction

The constant increase in the volume and variety of publicly available genomic and proteomic data is a characteristic trait of modern biomedical sciences. A fundamental problem in this area is the assignment of functions to biological macromolecules, especially proteins. Indeed, the accurate annotation of protein function would also have great biomedical and pharmaceutical implications, since several human diseases have genetic causes. While molecular experiments provide the most reliable annotation of proteins, their relatively low throughput and restricted scope have led to an increasing role for automated function prediction (AFP). AFP is characterized by unbalanced functional classes with rare positive instances. Moreover, since only positive membership to functional classes is usually assessed, negative instances are not uniquely defined, and different approaches to choose them have been proposed [1], [2], [3]. Other peculiarities of AFP include: (1) the need of integrating several heterogeneous sources of genomic, proteomic, and transcriptomic data in order to achieve more accurate predictions [4], [5]; (2) the presence of multiple labels and dependencies among class labels; (3) the hierarchical structure of functional classes (a direct acyclic graph for the Gene Ontology GO [6], a forest of trees for the FunCat taxonomy [7]) with different levels of specificity.

Recently, two international challenges for Critical Assessment of Functional Annotation, (CAFA [8] and CAFA2 [9]) were organized to evaluate computational methods that automatically assign protein functions. In particular, CAFA2 emphasized the need for multilabel or structured-output learning algorithms for predicting a set of terms, or a subgraph of the GO ontology for a given protein. In this work we mainly focus on this problem, whose solution however requires paying attention also to the other aspects of AFP.

Several approaches to the prediction of protein functions were proposed in the literature, including sequence-based [10], [11], [12] and network-based methods [13], [14], [15], structured output algorithms based on kernels [3], [16], [17] and hierarchical ensemble methods [18], [19], [20]. In particular, the availability of large-scale networks, in which nodes are genes/proteins and edges their functional pairwise relationships, has promoted the development of several machine learning methods where novel annotations are inferred by exploiting the topology of the resulting biomolecular network. Initially, network-based approaches relied on the so called guilt-by-association (GBA) rule, which makes predictions assuming that interacting proteins are likely to share similar functions [21], [22], [23]. Indirect neighbours were also exploited to modify the notion of pairwise-similarities among nodes by accounting for pairs of nodes connected through intermediate ones [24], [25]. Protein functions can be also propagated through the network with an iterative process until convergence [26], [27], by tuning the amount of propagation allowed in the graph through Markov random walks [28], [29], by evaluating the functional flow through the nodes [30], by exploiting kernelized score functions [31], and by modelling protein memberships through Markov Random Fields [32] and Gaussian Random Fields [33], [34]. Furthermore, methods based on the convergence of classical [35], [36] and multiclass Hopfield networks [37] were recently proposed to specifically tackle the class imbalance.

Although protein functions are clearly dependent (see, e.g., the GO functions, where parent terms include all the proteins of their children), most AFP methods described above predict biological functions independently from each other. Multitask methods, on the other hand, take advantage of existing dependencies by transferring information between related tasks, which typically leads to learning faster than algorithms trained independently on each task.

In this paper we investigate an alternative approach to

- M. Frasca and N. Cesa Bianchi are with the Dipartimento di Informatica, Università degli Studi di Milano, Via Comelico 39 Milano, 20137, Italy. E-mail: {marco.frasca, nicolo.cesa-bianchi}@unimi.it

Manuscript received April 19, 2005; revised August 26, 2015.
multitask learning based on exploiting task dissimilarities rather than similarities. In particular, we consider two multitask extensions of a known label propagation algorithm [26]: the first extension follows a standard multitask approach based on task similarities; the second extension learns instead from task dissimilarities. Both approaches can be naturally applied to the multilabel prediction of proteins. The prediction tasks we consider are the GO protein functions of fly, human, and bacteria model organisms. We compute different measures of similarity/dissimilarity between GO terms, taking into account both GO structure and protein annotations. We show that the approach learning from task dissimilarities greatly helps in unbalanced tasks (by helping instances labeled with the rare class labels to be correctly classified), and does not hurt in the more balanced cases. This is a crucial point in protein function prediction, since terms better describing protein functions—i.e., the most specific ones—are the most unbalanced (proteins annotated with these terms are very rare). On the other hand, learning from similar tasks tends to be more effective on balanced settings. Note that the proposed multitask extensions of the method.

The weighted cutsize can be also expressed as a quadratic form
\[
\Gamma_G^W(f) = \sum_{i,j \in E} w_{ij} (f_i - f_j)^2.
\]

The solution \(f^*_U\) of (2) is smooth on \(G\). Namely, if two vertices \(i, j \in U\) are connected with a large weight \(w_{ij}\), then \(f^*_i\) is close to \(f^*_j\). Indeed, the components \(i \in U\) of \(f^*\) satisfy the harmonic property [26]
\[
f^*_i = \frac{1}{d_i} \sum_{j} w_{ij} f^*_j.
\]

The vector \(f^*_U\) can be also written in closed form as
\[
f^*_U = (D_{UU} - W_{UU})^{-1} W_{US} f^*_S
\]

where
\[
W = \begin{pmatrix} W_{UU} & W_{US} \\ W_{US}^T & W_{SS} \end{pmatrix}
\]
is the weight matrix partitioned in blocks to emphasize the labeled and unlabeled part of the graph (similarly for the matrix \(D\)). As the components of \(f^*_U\) given by (3) are not in \([-1,1]\), the final labeling produced by LP is obtained by thresholding each component \(f^*_i\) for \(i \in U\).

3.2 Multitask label propagation (MTLP)

It is fairly easy to use similarity or dissimilarity information between tasks in order to generalize LP to multitask learning, while preserving the regularity of every task in the sense of (1).

We start by considering multitask LP based on similarity information. Suppose a \(m \times m\) symmetric matrix \(C\) is given, where each entry \(C_{kr} \in [0,1]\) quantifies the relatedness between tasks \(k\) and \(r\). Let \(\mathbf{A} = \gamma \mathbf{I}_m + \mathbf{C}\) be the matrix where \(\gamma > 0\), \(\mathbf{I}_m\) is the \(m \times m\) identity matrix, and \(\mathbf{C}\) is the Laplacian of \(\mathbf{C}\). The matrix \(\mathbf{A}\) is symmetric and positive definite, since \(\mathbf{A}\) is diagonally dominant with positive diagonals, and thus invertible. Denote by \(\mathbf{Y}\) the \(n \times m\) label matrix whose \(k\)-th column is the vector \(y^{(k)}\), and by \(\mathbf{F}\) the \(n \times m\) matrix whose \(k\)-th column is the vector \(f^{(k)}\).

When learning multiple related tasks, a widely used approach is requiring that similar tasks be assigned sim-
ilar labelings. To this end, we introduce the linear map
\[ \psi_{\mathcal{A}^{-1}} : \mathbb{R}^{n \times m} \rightarrow \mathbb{R}^{n \times m}, \]
defined as follows:
\[ \psi_{\mathcal{A}^{-1}}(Y) = YA^{-1}. \] (4)

It can be shown that the map \( \psi_{\mathcal{A}^{-1}} \) acts on a multitask labeling matrix \( Y \) by getting closer (in Euclidean distance) the labelings (columns of \( Y \)) corresponding to tasks that are similar according to \( \mathcal{C} \).

By means of \( \psi_{\mathcal{A}^{-1}} \), the exploitation of task similarities can be encoded into the learning problem (2) as follows:
\[
\begin{aligned}
\text{min} \quad & \text{trace}(F^T LF) \\
\text{s.t.} \quad & F_{ik} = \tilde{Y}_{ik} \quad i \in S, \; k = 1, \ldots, m
\end{aligned}
\] (5)
where \( \tilde{Y} = \psi_{\mathcal{A}^{-1}}(Y) = YA^{-1} \). The solution to (5) is
\[ \tilde{F}_U = (DUU - WUU)^{-1}WUSY_{\mathcal{S}} \]
where \( \tilde{F}_U \) is the submatrix of \( F \) including only the rows indexed by \( U \), and \( Y_{\mathcal{S}} \) is the submatrix of \( Y \) including only the rows indexed by \( S \). By observing that \( Y_{\mathcal{S}} = Y_{\mathcal{S}}A^{-1} \), we have
\[ \tilde{F}_U = (DUU - WUU)^{-1}WUSY_{\mathcal{S}}A^{-1} = F^*_u A^{-1} \]
where \( F^*_u \) is the solution of (5) with constraints \( F_{ik} = Y_{ik} \) for \( i \in S \) and \( k = 1, \ldots, m \). The equality \( \tilde{F}_U = F^*_u A^{-1} \) shows that it is equivalent to apply the task feature map (4) before or after performing label propagation. This ensures that the multitask mapping does not increase the label propagation complexity.

As we show next, this solution does not perform well on unbalanced classification problems, where some class (typically the positive class) is largely underrepresented. We propose here an alternative approach, which exploits the prior information about task relatedness in an “inverse” manner. Specifically, we propose a multitask label propagation algorithm which learns multiple tasks by requiring that discriminant tasks be assigned dissimilar labelings. As we see in the experiments, this approach turns out to work particularly well on unbalanced classification problems.

The first component of this method is a dissimilarity matrix \( \mathcal{C} \), where \( \mathcal{C}_{kr} \in [0, 1] \) is a measure of dissimilarity between tasks \( k \) and \( r \) (we discuss in Section 3.2.2 possible choices for the matrices \( \mathcal{C} \) and \( \mathcal{\mathcal{C}} \)).

Given the matrix \( \mathcal{C} \), we consider the multitask map \( \psi_{\mathcal{A}} : \mathbb{R}^{n \times m} \rightarrow \mathbb{R}^{n \times m} \), defined as
\[ \psi_{\mathcal{A}}(Y) = YA \] (6)
where \( \mathcal{A} = \tau \mathcal{L}_N + \mathcal{C}, \tau > 0 \), and \( \mathcal{L} \) is the Laplacian matrix of \( \mathcal{C} \). Unlike the inverse transformation (4), the map \( \psi_{\mathcal{A}} \) moves the columns of matrix \( Y \) farther away from each other in the corresponding \( n \)-dimensional space (in the sense of the Euclidean distance). We formally show that in Section 3.2.1. Using \( \psi_{\mathcal{A}} \) instead of \( \psi_{\mathcal{A}^{-1}} \) in (5), we obtain the following optimization problem:
\[
\begin{aligned}
\text{min} \quad & \text{trace}(F^T LF) \\
\text{s.t.} \quad & F_{ik} = \tilde{Y}_{ik} \quad i \in S, \; k = 1, \ldots, m
\end{aligned}
\] (7)
with \( \tilde{Y} = \psi_{\mathcal{A}}(Y) \). Similarly to (5), the solution of (7) is
\[ \tilde{F}_U = (DUU - WUU)^{-1}WUSY_{\mathcal{S}}A = F^*_u \mathcal{A} \]
where \( F^*_u \) is the solution of (7) with constraints \( F_{ik} = Y_{ik} \) for \( i \in S \) and \( k = 1, \ldots, m \). Just like in the previous case, the equality \( \tilde{F}_U = F^*_u \mathcal{A} \) shows that it is equivalent to apply the task feature map (6) before or after performing label propagation.

We call MTLP-inv the similarity-based method (5) and MTLP the dissimilarity-based method (7). In the next section we show some interesting properties of the map \( \psi_{\mathcal{A}} \) which make MTLP suitable for unbalanced classification problems.

### 3.2.1 Analysis of the multitask map \( \psi_{\mathcal{A}} \)

Given \( M \in \mathbb{R}^{n \times m} \), let \( M_i \) and \( M_k \) be, respectively, the \( i \)-th row and the \( k \)-th column of the matrix \( M \). Let also \( P_i = \{1 \leq k \leq m : Y_{ik} = 1\} \) be the set of tasks for which the instance \( i \) is positive, and \( N_i \) the set of tasks for which the instance \( i \) is negative. We introduce the following notation: for each \( k = 1, \ldots, m \)
\[ a_{k,i}^+ = \sum_{r \in P_i} A_{rk}, \quad a_{k,i}^- = \sum_{r \in N_i} A_{rk}, \quad a_k = \sum_{r=1}^m A_{rk}. \]
and
\[ d_{k,i}^+ = \sum_{r \in P_i} d_{r,i}, \quad d_{k,i}^- = \sum_{r \in N_i} d_{r,i}, \quad d_k = \sum_{r=1}^m d_{r,i}. \]

The next result shows that the action of the linear map \( \psi_{\mathcal{A}} \) on the label matrix \( Y \) changes the value of each label without altering the sign. The label of an instance \( i \) in task \( k \) is made roughly proportional to the weighted sum of tasks in \( \mathcal{C} \) that are dissimilar to \( k \) and have a different label for instance \( i \) —see also Corollary 1.

**Fact 1.** Given \( Y \in \{-1, 1\}^{n \times m} \), the task interaction matrix \( \mathcal{C} \in \mathbb{R}^{m \times m} \), and the map \( \psi_{\mathcal{A}} : \mathbb{R}^{n \times m} \rightarrow \mathbb{R}^{n \times m} \) such that \( \tilde{Y} = \psi_{\mathcal{A}}(Y) = YA \), where \( \mathcal{A} = \tau \mathcal{L}_N + \mathcal{C} \), then for all \( i = 1, \ldots, n \) it holds
\[ \tilde{Y}_{ik} = \left\{ \begin{array}{ll}
\tau + 2d_{k,i}^- & \text{if } Y_{ik} = +1 \\
-\tau - 2d_{k,i}^+ & \text{if } Y_{ik} = -1
\end{array} \right. \]

**Proof:** By definition, \( \tilde{Y}_{ik} = \sum_{r=1}^m Y_{ir}A_{rk} = a_{k,i}^+ - a_{k,i}^- \). We distinguish the following two cases.

**Case 1.** \( k \in P_i \). In this case we have \( a_{k,i}^+ = \mathcal{A}_{kk} - d_{k,i}^- = \tau + d_k - d_{k,i}^+ = \tau + d_k - d_{k,i}^- = \tau - d_{k,i}^+ \), since by definition \( d_k = d_k^+ + d_k^- \) for any \( i = 1, 2, \ldots, n \). Moreover, since \( k \in P_i \), we have \( d_{k,i}^- = -d_{k,i}^+ \) (by the definition of Laplacian), and accordingly
\[ \tilde{Y}_{ik} = \tau + d_{k,i}^- - (d_{k,i}^+) = \tau + 2d_{k,i}^- \]

**Case 2.** \( k \in N_i \). In this case, it holds \( d_{k,i}^+ = -d_{k,i}^- \), whereas \( a_{k,i} = \mathcal{A}_{kk} - d_{k,i}^- = \tau + d_k - d_{k,i}^+ = \tau + d_k - d_{k,i}^- \), which follows
\[ \tilde{Y}_{ik} = -d_{k,i}^- - \tau - d_{k,i}^+ = -\tau - 2d_{k,i}^- \]

The property is proven by observing that \( k \in P_i \) implies \( Y_{ik} = +1 \) and \( k \in N_i \) implies \( Y_{ik} = -1 \). \( \square \)

Using Fact 1 we can show that the map \( \psi_{\mathcal{A}} \) tends to increase the distance between the labelings \( Y_s \) and \( Y_{sr} \) for any pair of distinct tasks \( r, s \in \{1, 2, \ldots, m\} \). Indeed, we can prove the following.
Fact 2. Given $Y \in \{-1,1\}^{n \times m}$, the task interaction matrix $\mathbf{C} \in \mathbb{R}^{m \times m}$, and the map $\psi \mathbf{A} : \mathbb{R}^{n \times m} \rightarrow \mathbb{R}^{n \times m}$ such that $\mathbf{Y} = \psi \mathbf{A}(Y) = \mathbf{Y} \mathbf{A}$, where $\mathbf{A} = \mathbf{C} \mathbf{L}_m + \mathbf{C}$. Then for every $r, s \in \{1, 2, \ldots, m\}$ it holds

$$
\|Y_r - Y_s\| \leq \|\hat{Y}_r - \hat{Y}_s\|^2
$$

for every $\gamma \geq 1$, where $\| \cdot \|$ is the Euclidean norm.

Proof: We prove this property by showing that $(Y_r - Y_s)^2 \leq (\hat{Y}_r - \hat{Y}_s)^2$ for all $r \in \{1, 2, \ldots, n\}$. We distinguish the following four cases:

Case 1: $Y_r = Y_s = 1$. In this case $(Y_r - Y_s)^2 = 0$, and by Fact 1, $(Y_r - \hat{Y}_s)^2 = (\gamma + 2\gamma r, \gamma - 2\gamma s)^2 = 4(\gamma r, \gamma s)^2 \geq 0$.

Case 2: $Y_r = Y_s = -1$. Even in this case $(Y_r - Y_s)^2 = 0$, whereas $(\hat{Y}_r - \hat{Y}_s)^2 = (\gamma - 2\gamma r, -\gamma + 2\gamma s)^2 = 4(\gamma r, -\gamma s)^2 \geq 0$.

Case 3: $Y_r = 1$ and $Y_s = -1$. In this case, $(Y_r - Y_s)^2 = 4$, and $(\hat{Y}_r - \hat{Y}_s)^2 = (\gamma + 2\gamma r, -\gamma + 2\gamma s)^2 = 4(\gamma r, -\gamma s)^2 \geq 0$.

Case 4: $Y_r = -1$ and $Y_s = 1$. Again $(Y_r - Y_s)^2 = 4$, and $(\hat{Y}_r - \hat{Y}_s)^2 = (\gamma - 2\gamma r, -\gamma - 2\gamma s)^2 = 4(\gamma r, -\gamma s)^2 \geq 0$.

The map $\psi \mathbf{A}$ not only increases the distance between the instance-indexed label vector for two distinct tasks (as we just showed), but it also increases the distance between the task-indexed label vector for two distinct instances. Indeed, since $\mathbf{C}$ is positive semidefinite, it is easy to show that when $\gamma \geq 1$ the transformation $\psi \mathbf{A}$ increases the distance between the labelings $Y_i$ and $Y_j$, for any pair of distinct instances $i, j \in \{1, 2, \ldots, n\}$.

We now focus our discussion on another important feature of the algorithm, which makes our multitask label propagation appropriate for tasks with very unbalanced labelings. Specifically, when most entries of each column in the label matrix $Y$ are $-1$. In this case, the rows of $Y$ also contain mostly negative entries. Accordingly, by Fact 1, we can compensate the preponderance of negatives by applying the map $\psi \mathbf{A}$. We show this with an example.

Consider the task interaction matrix $\mathbf{C}$ such that $\mathcal{C}_{rs} = 1$ for all $r \neq s$. That is, all tasks are strongly dissimilar to each other. Then

$$
\mathbf{A} = \begin{bmatrix}
\gamma + m - 1 & ... & -1 \\
... & \mathbf{I} & ... & -1 \\
-1 & ... & \gamma + m - 1 \\
\end{bmatrix}
$$

By Fact 1, it is straightforward to prove the following.

Corollary 1. Fix $Y \in \{-1,1\}^{n \times m}$ and the map $\psi \mathbf{A} : \mathbb{R}^{n \times m} \rightarrow \mathbb{R}^{n \times m}$ such that $\mathbf{Y} = \psi \mathbf{A}(Y) = \mathbf{Y} \mathbf{A}$, where $\mathbf{A}$ is defined as in (8). Then, for all $i = 1, \ldots, n$ it holds that

$$
\mathbf{Y}_{ik} = \begin{cases}
\gamma + 2|N_i| & \text{if } Y_{ik} = +1 \\
-\gamma - 2|P_i| & \text{if } Y_{ik} = -1.
\end{cases}
$$

Corollary 1 shows that, when $|P_i| \ll |N_i| = m - |P_i|$ (that is, the multitask labeling for vertex $i$ is unbalanced towards positives), the map $\psi \mathbf{A}$ assigns to positives $(Y_{ik} = +1)$ an absolute value higher than that assigned to negatives $(Y_{ik} = -1)$. An analogous behavior characterizes our method when a generic matrix $\mathbf{C}$ is considered, as stated in Fact 1. This simple property allows the rare positive labels to propagate in the graph. This is unlike the standard LP algorithm, where positive vertices are easily overwhelmed by the negative vertices during the label propagation process. The toy example in Figure 1 shows that the application of the map $\psi \mathbf{A}$, where $\mathbf{A}$ is defined as in (8), allows to improve the final classification of vertices. These observations are empirically confirmed in Section 4.

3.2.2 Task similarities

While MTLP and MTLP-inv are designed to work with any task matrix, similarity and dissimilarity measures are typically tailored to specific domains. Different tasks may share different types of similarities, or may be organized in a hierarchy with a specific structure —such as a tree or a directed acyclic graph— where the positive instances of the children tasks are subsets of the positive instances of their parent tasks. In the case of a hierarchy, different approaches for computing the task matrix are possible: considering only the structure of the hierarchy [40], [41], or combining the hierarchical information with the information content of the tasks [42].

In this work we consider two dissimilarity measures ($\text{diss}_0$ and $\text{diss}_3$) and three similarity measures ($\text{sim}_1, \text{sim}_2, \text{sim}_3$). The similarity measures $\text{sim}_1$ and $\text{sim}_2$ were introduced by Jiang [43] and Lin [44], respectively. Both measures are derived from the dissimilarity measure $\text{diss}_0$, whose definition requires a hierarchy over the tasks. The dissimilarity $\text{diss}_3$ is computed directly from the similarity $\text{sim}_3$, which does not require any hierarchical information.

When tasks are organized in a hierarchy, we denote by $\text{ancestors}(k) \subseteq \{1, \ldots, m\}$ the set of ancestor tasks of task $k$ in the hierarchy. Moreover, we use $\nu(k)$ to denote the frequency of positive instances for task $k$. Since a positive instance for
a task is also positive for any \( r \in \text{anc}(k) \), it holds that \( \nu(k) \leq \nu(r) \). Finally, we denote by \( \text{MA}(k, r) \) the common ancestor of two tasks \( k \) and \( r \) whose frequency \( \nu(\text{MA}(k, r)) \) is the lowest among all ancestors of \( k \) and \( r \).

Let \( \log(\nu(k)) \) be the information content of task \( k \). We start by recalling the hierarchical dissimilarity measure introduced in [44],

\[
\text{disso}(k, r) = -\log(\nu(k)) - \log(\nu(r)) + 2\log(\nu(\text{MA}(k, r))).
\]

This is the sum of the information content of \( k \) and \( r \) minus the information content of their closest common ancestor \( \text{MA}(k, r) \). Note that \( \text{disso}_0 \) is always positive, as \( \text{MA}(k, r) \geq \max\{\nu(k), \nu(r)\} \). The two hierarchical similarity measures associated with \( \text{disso}_0 \) are defined as follows:

Jiang similarity measure:

\[
\text{sim}_1(k, r) = \frac{1}{1 + \text{disso}_0(k, r)}.
\]

Lin similarity measure:

\[
\text{sim}_2(k, r) = \frac{2\log(\nu(\text{MA}(k, r)))}{\log(\nu(k)) + \log(\nu(r))}.
\]

Our third similarity measure does not rely on a hierarchy of tasks. Let \( P(k) \) the set of instances that are positive for the task \( k \).

Information content measure:

\[
\text{sim}_3(k, r) = \begin{cases} 
\frac{|P(k) \cap P(r)|}{|P(k) \cup P(r)|} & \text{if } P(k) \cup P(r) \neq \emptyset \\
0 & \text{otherwise.}
\end{cases}
\]

This is the ratio between the number of examples that are positive for both tasks and the number of examples that are positive for at least one task. The higher the number of shared positive examples, the higher the similarity (up to 1). When two tasks do not share any positive example, their similarity is zero. In a hierarchy of tasks, tasks with many positive examples are usually closer to the root (less specific). In this case the denominator of \( \text{sim}_3 \) tends to reduce the similarity between the two tasks as opposed to the case in which tasks have a small number of positive annotations. Indeed, sharing annotations between two specific tasks (closer to leaves) is more informative than sharing annotations between two more general tasks (closer to the root).

In the experiments, we compare learning with similarities \( \text{sim}_1(k, r) \) and \( \text{sim}_2(k, r) \) against learning with the dissimilarity \( \text{disso}_0(k, r) \). We also compare learning with \( \text{sim}_3(k, r) \) against \( \text{disso}_3(k, r) = 1 - \text{sim}_3(k, r) \). For each one of the similarity/dissimilarity measures defined above, we set \( c_{kr} = \text{sim}(k, r) \) and \( c_{kr} = \text{disso}(k, r) \) (where necessary, values are normalized so that all matrix entries lie in the range \([0, 1]\)).

### 4 Results and Discussion

In this section we evaluate our multitask algorithms on the prediction of the bio-molecular functions of proteins belonging to some considered model organisms. We start by describing the experimental setting. Then we compare the performance of our algorithms against that of state-of-the-art methods.

#### 4.1 Experimental setting

#### 4.1.1 Data

We considered three different experiments to predict the protein functions of three model organisms: *Drosophila melanogaster* (fly), *Homo sapiens* (human) and *Escherichia coli* (bacteria). Gene networks for model organisms have been downloaded from the GeneMANIA website (www.genemania.org), and selected in order to cover different types of data, including co-expression, genetic interactions, shared domains, and physical interactions. The selected networks are described in Tables 1, 2 and 3.

For every organism, networks were integrated through unweighted sum on the union of genes in the individual networks. No preprocessing was applied to the individual networks, whereas each network, denoted by the corresponding connection matrix \( W \), was normalized as follows:

\[
\hat{W} = D^{-1/2}WD^{-1/2}
\]

| Type                        | Source                      | Nodes |
|-----------------------------|-----------------------------|-------|
| Co-expression               | Baradaran-Heravi et al. [45] | 8857  |
| Co-expression               | Busser et al. [46]          | 8857  |
| Co-expression               | Colombi et al. [47]         | 8857  |
| Co-expression               | Lundberg et al. [48]        | 8857  |
| Genetic interactions        | BioGRID [49]                | 929   |
| Genetic interactions        | Wu et al. [50]              | 1414  |
| Physical interactions       | Guruharsha et al. A [51]    | 1866  |
| Physical interactions       | Gurohharsha et al. B [51]   | 3833  |
| Physical interactions       | BioGRID [49]                | 558   |
| Shared protein domains      | Pfam [62]                   | 5627  |

| Type                        | Source                      | Nodes |
|-----------------------------|-----------------------------|-------|
| Co-expression               | Baradaran-Heravi et al. [45] | 7611  |
| Co-expression               | Balogbindi et al. [54]      | 17522 |
| Co-expression               | Bigrer et al. [55]          | 17522 |
| Co-expression               | Botling et al. [56]         | 17522 |
| Co-expression               | Clarke et al. [57]          | 17458 |
| Co-expression               | Vallat et al. [58]          | 17521 |
| Common biological pathways  | PATHWAYCOMMONS [59]         | 2133  |
| Common biological pathways  | Wu et al. [60]              | 5319  |
| Physical interactions       | BioGRID [49]                | 15800 |
| Physical interactions       | iRef-GRID [61]              | 9403  |
| Physical interactions       | iRef-HPRD [61]              | 9403  |
| Physical interactions       | iRef-OPHID [61]             | 9403  |
| Physical interactions       | IREF SMALL-SCALE-STUDIES [61]| 9036  |
| Shared protein domains      | InterPro [52]               | 15980 |
| Shared protein domains      | Pfam [62]                   | 15251 |

**TABLE 1**

**Fly networks.**

| Type                        | Source       | Nodes |
|-----------------------------|--------------|-------|
| Co-expression               | Baradaran-Heravi et al. [45] | 7611  |
| Co-expression               | Balogbindi et al. [54]      | 17522 |
| Co-expression               | Bigrer et al. [55]          | 17522 |
| Co-expression               | Botling et al. [56]         | 17522 |
| Co-expression               | Clarke et al. [57]          | 17458 |
| Co-expression               | Vallat et al. [58]          | 17521 |
| Common biological pathways  | PATHWAYCOMMONS [59]         | 2133  |
| Common biological pathways  | Wu et al. [60]              | 5319  |
| Physical interactions       | BioGRID [49]                | 15800 |
| Physical interactions       | iRef-GRID [61]              | 9403  |
| Physical interactions       | iRef-HPRD [61]              | 9403  |
| Physical interactions       | iRef-OPHID [61]             | 9403  |
| Physical interactions       | IREF SMALL-SCALE-STUDIES [61]| 9036  |
| Shared protein domains      | InterPro [52]               | 15980 |
| Shared protein domains      | Pfam [62]                   | 15251 |

**TABLE 2**

**Human networks.**
where $D$ is the diagonal matrix with diagonal entries $d_{ii} = \sum_j W_{ij}$.

Protein functions were downloaded from the Gene Ontology. This ontology is structured as a directed acyclic graph with different levels of specificity and contains three branches: Biological Process (BP), Molecular Functions (MF), and Cellular Components (CC). We considered the experimental annotations in the releases 07.03.16, 16.03.16, and 17.10.16 respectively for fly, human and bacteria organisms. We performed a dedicated experiment for every branch.

For predicting the most specific terms in the ontology (i.e., those best describing protein functions), and in order to consider terms with a minimum amount of prior information, we selected all the GO terms with $5 - 100$ positive annotated genes, obtaining 2657 (1742 BP, 539 MF, 376 CC), 5312 (3799 BP, 957 MF, 556 CC), and 1324 (653 BP, 610 MF, 61 CC) terms for fly, human, and bacteria, respectively. We considered two groups of GO terms according to their specificity: GO terms with 5-20 and 21-100 annotated proteins, for a total of 2 categories for every GO branch. In the end, we obtained a total of 10329 fly, 15262 human, and 4132 bacteria genes which have at least one GO positive annotation in the considered GO release. The obtained tasks are therefore severely unbalanced toward negatives.

### 4.1.2 Evaluation metrics

In order to evaluate the generalization performance of the compared methods, we applied a 3-fold cross-validation experimental setting and adopted the Area Under the Precision-Recall Curve (AUPRC) as “per term” ranking measure. AUPRC is indeed more informative on unbalanced settings than the classical area under the ROC curve [68]. Furthermore, following the recent CAF A2 international challenge, we also considered a “protein-centric evaluation” to assess performance accuracy in predicting all ontological terms associated with a given protein sequence [9]. In this scenario, the multiple-label F-score is used as performance measure. More precisely, if we indicate as $TP_j(t)$, $TN_j(t)$ and $FP_j(t)$ respectively the number of true positives, true negatives, and false positives for the protein $j$ at threshold $t$, we can define the “per-protein” multiple-label precision $Prec(t)$ and recall $Rec(t)$ at a given threshold $t$ as:

$$Prec(t) = \frac{\sum_{j=1}^{n} TP_j(t)}{\sum_{j=1}^{n} TP_j(t) + FP_j(t)}$$

$$Rec(t) = \frac{\sum_{j=1}^{n} TP_j(t)}{\sum_{j=1}^{n} TP_j(t) + FN_j(t)}$$

where $n$ is the number of proteins. In other words, $Prec(t)$ (resp., $Rec(t)$) is the average multilabel precision (resp., recall) across proteins. The multilabel F-measure depends on $t$ and according to CAF2 experimental setting, the maximum achievable F-score ($F_{\text{max}}$) is adopted as the main multilabel “per-protein” metric:

$$F_{\text{max}} = \max_t \frac{2Prec(t)Rec(t)}{Prec(t) + Rec(t)}$$  \hspace{1cm} (9)

### 4.2 Results

#### 4.2.1 Evaluating GO semantic similarities

This section investigates the impact of the task similarity/dissimilarity measures described in Section 3.2.2 on the performance of the proposed multitask label propagation algorithms. Table 4 shows the obtained results. In this experiment we set $\gamma = \tau = 1$ (the choice of parameter $\tau$ is discussed in Section 4.2.5). When MTLP-inv uses the similarity measures $\sim_{\text{sim}}$, $\sim_{\text{diss}}$ and MTLP uses $\sim_{\text{diss}}$, MTLP outperforms MTLP-inv in both AUPRC and $F_{\text{max}}$. Nevertheless, the GO term similarity $\sim_{\text{sim}}$ is much more informative for MTLP-inv, which achieves in this case results competitive with MTLP (whose performance instead is nearly indistinguishable when using $\sim_{\text{diss}}$). The differences in favor of MTLP seem to increase with the data imbalance: on human data set, the most unbalanced, we observe the highest gap in favor of MTLP; whereas on the Bacteria data set, the least unbalanced, the gap is reduced and—in some cases like for the MF terms—MTLP-inv significantly outperforms MTLP in terms of average AUPRC. In terms of $F_{\text{max}}$, however, MTLP is always the top method.

Overall, these results suggests that MTLP should be preferred when the proportion of positives is drastically smaller than that of negatives. When data are more balanced, MTLP-inv better exploits the similarities among tasks and, at least in term of AUPRC, is a valid option. In terms of multilabel accuracy, MTLP is always better than MTLP-inv. Finally, it is worth noting that both methods outperform LP in terms of AUPRC (see Section 4.2.3 for LP results), whereas in terms of $F_{\text{max}}$ only MTLP achieves better results than LP. In order to investigate the reasons why, unlike MTLP-inv, MTLP performance slightly varies with the task dissimilarity measure, we run MTLP on the fly organism and CC tasks by randomly generating the matrix $\mathbf{C}$. We generated matrices with different sparsity (from 5% to 95%, with steps of 10%) and with different ranges of weight values. Specifically, we uniformly selected weights in the interval $[0, \tau]$, with $\tau$ ranging from 0.1 to 1, by steps of 0.1. In Figure 2, we show the heatmap of the average AUPRC obtained in each experiment. As expected, the results are considerably worse than those obtained when considering real dissimilarity matrices (see Table 4). There is a small

| Type                    | Source                     | Nodes |
|-------------------------|----------------------------|-------|
| Co-expression           | Graham et al. [63]         | 3959  |
| Co-expression           | Robbins-Manke et al. [64]  | 3912  |
| Genetic interactions    | Babu et al. [65]           | 715   |
| Genetic interactions    | Butland et al. [66]        | 3497  |
| Physical interactions   | Hu et al [67]              | 1537  |
| Physical interactions   | IREF-Dip [61]             | 633   |
| Physical interactions   | Y2H - PPI                 | 1063  |
| Shared protein domains  | InterPro [52]             | 3005  |
| Shared protein domains  | Pfam [62]                 | 2726  |

**TABLE 3**

Bacteria networks.
### METHODS

|        | BP | MF | CC |
|--------|----|----|----|
|        | All | 5-20 | 21-100 | F_{max} | All | 5-20 | 21-100 | F_{max} | All | 5-20 | 21-100 | F_{max} |
| MTLPP diss_{0} | 0.140 | 0.133 | 0.153 | 0.247 | 0.333 | 0.322 | 0.355 | 0.411 | 0.262 | 0.265 | 0.253 | 0.354 |
| MTLPP diss_{3} | 0.140 | 0.133 | 0.153 | 0.246 | 0.333 | 0.322 | 0.355 | 0.410 | 0.262 | 0.265 | 0.253 | 0.357 |
| MTLPP inv sim_{1} | 0.020 | 0.013 | 0.031 | 0.183 | 0.198 | 0.179 | 0.238 | 0.374 | 0.150 | 0.138 | 0.181 | 0.306 |
| MTLPP inv sim_{2} | 0.020 | 0.014 | 0.031 | 0.170 | 0.192 | 0.172 | 0.235 | 0.351 | 0.101 | 0.082 | 0.147 | 0.259 |
| MTLPP inv sim_{3} | 0.135 | 0.129 | 0.146 | 0.244 | 0.328 | 0.318 | 0.352 | 0.381 | 0.261 | 0.265 | 0.251 | 0.333 |

**FLY**

|        | HUMAN | BACTERIA |
|--------|--------|----------|
|        |        |          |
|        |        |          |

#### TABLE 4

Comparison according to average AUPRC and multilabel F-measure ($F_{max}$) between MTLPP and MTLPP-inv using the semantic similarity measures described in Section 3.2.2. Column All is the average across all GO terms, column 5-20 is the average across GO terms with at most 20 positive genes, and column 21-100 is the average across terms with more than 20 positives. Best results are in boldface. Results are underlined when the difference between MTLPP and MTLPP-inv is statistically significant (Wilcoxon signed rank test, $p$-value < 0.05).

AUPRC variation from the different random data, with higher AUPRC when the dissimilarity matrix is denser and with larger entries (the former seems to affect the results more than the latter). This is consistent with Fact 1, since the lower the weight and/or the sparser the matrix, the closer MTLPP is to LP. Finally, on randomly generated dissimilarity matrices MTLPP performs even worse than LP, as we can see from Figure 4.
4.2.2 Grouping GO terms for multitask mapping

Following the approach proposed in [4], in addition to the strategy grouping GO terms by branch (i) adopted in the previous section, we have examined an alternative way for grouping the terms to be considered in the multitask maps (4) and (6) when running MTLP-inv and MTLP algorithms, respectively. Specifically, we grouped GO terms not just by GO branch (BP, MF, and CC), but also by taking into account the number of annotated proteins (ii), obtaining 6 groups: BP with 5-20 (1119 terms) and 21-100 (623 terms) annotations, MF 5-20 (362 terms), 21-100 (177 terms), and CC 5-20 (267 terms), 21-100 (109 terms). The corresponding results on fly data are reported in Figure 3. AUPRC results show negligible differences between strategies (i) and (ii), for both MTLP and MTLP-inv. More clear is the difference in terms of \( F_{\text{max}} \), with opposite behaviour between MTLP and MTLP-inv: MTLP has worse performance in all GO branches; MTLP-inv instead tends to perform better (see for instance MF results). Indeed, black lines (grouping strategy (ii)) in correspondence of \( F_{\text{max}} \) are always between grey lines (grouping strategy (i)). However, the best results are still achieved by MTLP when grouping terms by GO branch, and accordingly we consider this strategy in the rest of the paper.

4.2.3 Prediction of GO functions for fly, human, and bacteria organisms

MTLP \( (\tau = 1) \) was compared with state-of-the-art graph-based methodologies applied to the prediction of protein functions. We considered: LP, the label propagation algorithm described in Section 3.1; COSNetM [15], an extension of a node classifier designed for unbalanced settings [36]; RW, the classical \( t \)-step random walk algorithm [69]; GBA, a method based on the guilt-by-association assumption [23]; MS-kNN, one of the best methods in the recent CAFA2 challenge applying the \( kn \)NN algorithm to each network independently, and then combining the obtained predictions [70].

In order to deal with label imbalance in LP, we applied a label normalization step before running label propagation. This step normalizes the labels of each GO term so that positive and negative labels sum to 1. In our experiments, this variant of LP performs much better than the vanilla LP algorithm. For the RW algorithm we set the limit on the number of iterations to 100, since higher values did not improve the performance while increasing the computational burden. Finally, we set to 5 the parameter \( k \) for the \( kn \)NN algorithm, as a result of a tuning process on training data.

In Figures 4 and 5 we show the obtained results in terms of AUPRC and \( F_{\text{max}} \), on BP and CC terms respectively (on MF terms the methods showed a similar behaviour). Interestingly, MTLP always achieves the highest AUPRC averaged over all tasks (All), with statistically significant improvements over the second top method \( (p\text{-value} < 0.001) \), except for bacteria data and for BP terms on fly data. When comparing with LP method, the improvement is always significant, except for CC (bacteria data). COSNetM is the second method on human and fly data sets, while on bacteria LP (CC) and RW (BP) rank as second method. Furthermore, and more importantly, MTLP improvements are more noticeable on the most unbalanced terms, which are those best characterizing the biological functions of genes. GBA, MS-kNN and RW methods seem suffer the strongly unbalanced setting, and perform worse than LP, with the exception of RW on bacteria data set. The good performance of COSNetM in this unbalanced setting is likely due to its cost-sensitive strategy, which requires learning two model parameters. This extra learning step increases its computation time. Indeed, COSNetM takes on average around 4 seconds on a Linux machine with Intel Xeon(R) CPU 3.60GHz and 32 Gb RAM to perform an entire cross validation cycle for one task on fly data, whereas both LP and MTLP take on average slightly less than one second. This confirms our observation that applying the map \( \psi \) after label propagation does not increase the algorithm complexity, and just slightly increases the execution time for computing \( \psi \).

Even in terms of \( F_{\text{max}} \) MTLP obtains the best results, with LP second-best method (except on BP — fly data). This shows that our method can achieve good predictive capabilities both when predicting single GO terms and when predicting a GO multilabel for single proteins. On the other side, the compared methods tend to have competitive performance in only one scenario. For instance, RW poorly performs in terms of \( F_{\text{max}} \), whereas, unlike AUPRC, MS-kNN achieves good \( F_{\text{max}} \) results: on BP (fly data) it is the best method after MTLP. Even COSNetM, which is the second method in terms of AUPRC, achieves the third or the fourth best \( F_{\text{max}} \) rank.

4.2.4 Evaluating different powers of the Laplacian matrix

A further experiment was carried out to analyze how MTLP performance changes when using the map \( \psi_{\nu, p}(Y) = Y\mathcal{A}^p \) for \( p \geq \frac{1}{\nu} \), instead of \( \psi(Y) = Y\mathcal{A} \). We empirically tested on the fly organism different values of \( p \), fixing the parameter \( \tau = 1 \) and using the diss measure. The results are shown in Figure 6. We considered \( p = \frac{1}{2}, 1, 3, 4, 5 \). Except for BP terms, where the map \( \psi_{1/2} \) performs slightly better than \( \psi_{1/2} \), all choices of \( p \neq 1 \) lead to worse results. In particular, the performance strongly decays for \( p > 2 \).

In order to deal with label imbalance in LP, we applied a label normalization step before running label propagation. This step normalizes the labels of each GO term so that positive and negative labels sum to 1. In our experiments, this variant of LP performs much better than the vanilla LP algorithm. For the RW algorithm we set the limit on the number of iterations to 100, since higher values did not improve the performance while increasing the computational burden. Finally, we set to 5 the parameter \( k \) for the \( kn \)NN algorithm, as a result of a tuning process on training data.

In Figures 4 and 5 we show the obtained results in terms of AUPRC and \( F_{\text{max}} \), on BP and CC terms respectively (on MF terms the methods showed a similar behaviour). Interestingly, MTLP always achieves the highest AUPRC averaged over all tasks (All), with statistically significant improvements over the second top method \( (p\text{-value} < 0.001) \), except for bacteria data and for BP terms on fly data. When comparing with LP method, the improvement is always significant, except for CC (bacteria data). COSNetM is the second method on human and fly data sets, while on bacteria LP (CC) and RW (BP) rank as second method. Furthermore, and more importantly, MTLP improvements are more noticeable on the most unbalanced terms, which are those best characterizing the biological functions of genes. GBA, MS-kNN and RW methods seem suffer the strongly unbalanced setting, and perform worse than LP, with the exception of RW on bacteria data set. The good performance of COSNetM in this unbalanced setting is likely due to its cost-sensitive strategy, which requires learning two model parameters. This extra learning step increases its computation time. Indeed, COSNetM takes on average around 4 seconds on a Linux machine with Intel Xeon(R) CPU 3.60GHz and 32 Gb RAM to perform an entire cross validation cycle for one task on fly data, whereas both LP and MTLP take on average slightly less than one second. This confirms our observation that applying the map \( \psi \) after label propagation does not increase the algorithm complexity, and just slightly increases the execution time for computing \( \psi \).

Even in terms of \( F_{\text{max}} \) MTLP obtains the best results, with LP second-best method (except on BP — fly data). This shows that our method can achieve good predictive capabilities both when predicting single GO terms and when predicting a GO multilabel for single proteins. On the other side, the compared methods tend to have competitive performance in only one scenario. For instance, RW poorly performs in terms of \( F_{\text{max}} \), whereas, unlike AUPRC, MS-kNN achieves good \( F_{\text{max}} \) results: on BP (fly data) it is the best method after MTLP. Even COSNetM, which is the second method in terms of AUPRC, achieves the third or the fourth best \( F_{\text{max}} \) rank.

4.2.4 Evaluating different powers of the Laplacian matrix

A further experiment was carried out to analyze how MTLP performance changes when using the map \( \psi_{\nu, p}(Y) = Y\mathcal{A}^p \) for \( p \geq \frac{1}{\nu} \), instead of \( \psi(Y) = Y\mathcal{A} \). We empirically tested on the fly organism different values of \( p \), fixing the parameter \( \tau = 1 \) and using the diss measure. The results are shown in Figure 6. We considered \( p = \frac{1}{2}, 1, 3, 4, 5 \). Except for BP terms, where the map \( \psi_{1/2} \) performs slightly better than \( \psi_{1/2} \), all choices of \( p \neq 1 \) lead to worse results. In particular, the performance strongly decays for \( p > 2 \).

![Fig. 6. Average AUPRC values achieved by MTLP of fly data with different values of the parameter \( p \).](image-url)
Fig. 4. Average AUPRC performance across all GO terms (All), across GO terms with at most 20 positive instances (5), and across terms with more than 20 positives (21).

Fig. 5. Average multi-label F-measure performance across all GO terms.
since \( \mathcal{A} \) is diagonally dominant and absolute labels assigned to positives and negatives vertices by the map \( \psi \) tend to be almost the same (see Fact 1). Hence, this allows to "regulate" to some extent the method between multitask and singletask label propagation. We experimentally tuned \( \gamma \) expected, since 0.25 to 1.5 with step size 0.25. It turns out there is a negligible difference, with results reported in Table 4 and corresponding to \( \gamma = 1 \). This is expected, since \( m \) is much larger than 1 in the considered experiments. For this reason, we also performed another experiment in which we selected a smaller subset of terms in experiments. For this reason, we also performed another experiment in which we selected a smaller subset of terms in experiments. We experimentally tuned \( \gamma \) to some extent the method between multitask and singletask label propagation. We experimentally tuned \( \gamma \) to some extent the method between multitask and singletask label propagation. We experimentally tuned \( \gamma \) to some extent the method between multitask and singletask label propagation.

### Table 5

| \( \gamma \) | A11 | 5–20 | 20–100 |
|---|---|---|---|
| 0.25 | 0.158 | 0.150 | 0.182 |
| 0.5 | 0.157 | 0.151 | 0.177 |
| 0.75 | 0.145 | 0.134 | 0.178 |
| 1 | 0.144 | 0.133 | 0.178 |
| 1.25 | 0.140 | 0.130 | 0.175 |
| 1.5 | 0.139 | 0.129 | 0.174 |

The authors would like to thank the reviewers of BIOKDD16 for useful comments on an earlier draft of this paper.

### References

[1] 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.

[2] M. Frasca and D. Malchiodi, "Selection of negative examples for node label prediction through fuzzy clustering techniques," in Advances in Neural Networks: Computational Intelligence for ICT, S. Bassis, A. Esposito, C. F. Morabito, and E. Pasero, Eds. Cham: Springer International Publishing, 2016, pp. 67–76. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-33747-0_2.6

[3] S. Mostafavi and Q. Morris, "Using the gene ontology hierarchy when predicting gene function," in Proceedings of the Twenty-Fifth Annual Conference on Uncertainty in Artificial Intelligence (UAI-09). Corvallis, Oregon: AUAI Press, 2009, pp. 419–427.

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

[5] M. Frasca, A. Bertoni et al., "UNIPred: unbalance-aware Network Integration and Prediction of protein functions," Journal of Computational Biology, vol. 22, no. 12, pp. 1057–1074, 2015.

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

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

[8] P. Radivojac et al., "A large-scale evaluation of computational protein function prediction," Nature Methods, vol. 10, no. 3, pp. 221–227, 2013.

[9] Y. Jiang, T. R. Oron et al., "An expanded evaluation of protein function prediction methods shows an improvement in accuracy," Genome Biology, vol. 17, no. 1, p. 184, 2016. [Online]. Available: http://dx.doi.org/10.1186/s13059-016-1037-6.

[10] D. Martin, M. Berriman, and G. Barton, "Gotcha: a new method for prediction of protein function assessed by the annotation of seven genomes," BMC Bioinformatics, vol. 5, p. 178, 2004.

[11] T. Hawkins, M. Chitale, S. Lubahn et al., "Ptp: Automated prediction of gene ontology functional annotations with confidence scores using protein sequence data." Proteins, vol. 74, no. 3, pp. 566–82, 2009.

[12] A. Juncker, L. Jensen, A. Perleoni, A. Bernsel, M. Tress, P. Bork, G. von Heijne, A. Valencia, A. Ouzounis, R. Casadio, and S. Brunak, "Sequence-based feature prediction and annotation of proteins," Genome Biology, vol. 10:206, 2009.

[13] A. Vazquez, A. Flammini, A. Maritan, and A. Vespignani, "Global protein function prediction from protein-protein interaction networks," Nature Biotechnology, vol. 21, pp. 697–700, 2003.

[14] R. Sharan, I. Ulitsky, and R. Shamir, "Network-based prediction of protein function," Mol. Syst. Biol., vol. 6, no. 8, pp. 0007.

[15] M. Frasca, "Automated gene function prediction through gene multifunctionality in biological networks," Neurocomputing, vol. 162, pp. 48–56, 2015.

[16] 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.

[17] 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, no. Suppl 3:S10, 2013.

[18] G. Obozinski, G. Lanchriet, C. Grant, J. M., and W. Noble, "Consistent probabilistic output for protein function prediction," Genome Biology, vol. 9, no. 56, 2008.

[19] 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. 52, 2008.

[20] G. Valentini, "Hierarchical Ensemble Methods for Protein Function Prediction," ISRN Bioinformatics, vol. 2014, no. Article ID 901419, pp. 1–34, 2014.

[21] E. Marcotte, M. Pellegrini, M. Thompson, T. Yeates, and D. Eisenberg, "A combined algorithm for genome-wide prediction of protein function," Nature, vol. 402, pp. 83–86, 1999.

[22] S. Oliver, "Guilty-by-association goes global," Nature, vol. 403, pp. 601–603, 2000.
transient behaviour of the fur and soxs regulators,” *Microbiology*, vol. 158, no. 1, pp. 284–292, 2012.

[64] J. L. Robbins-Manke, Z. Z. Zdraveski, M. Marinus, and J. M. Essigmann, “Analysis of global gene expression and double-strand-break formation in dna adenine methyltransferase- and mismatch repair-deficient escherichia coli,” *Journal of bacteriology*, vol. 187, no. 20, pp. 7027–7037, October 2005. [Online]. Available: http://europepmc.org/articles/PMC1251628

[65] M. Babu et al., “Genetic interaction maps in escherichia coli reveal functional crosstalk among cell envelope biogenesis pathways,” *PLoS Genet*, vol. 7, no. 11, pp. 1–15, 11 2011.

[66] G. Butland et al., “esga: E. coli synthetic genetic array analysis,” *Nat Meth*, vol. 5, no. 3, pp. 789–795, jan 2008.

[67] P. Hu, S. C. Janga et al., “Global Functional Atlas of Escherichia coli Encompassing Previously Uncharacterized Proteins,” *PLoS Biol*, vol. 7, no. 4, pp. e1 000 096+, Apr. 2009. [Online]. Available: http://dx.doi.org/10.1371/journal.pbio.1000096

[68] T. Saito and M. Rehmsmeier, “The precision-recall plot is more informative than the roc plot when evaluating binary classifiers on imbalanced datasets,” *PLoS ONE*, vol. 10, no. 3, pp. 1–21, 03 2015.

[69] L. Lovász, “Random walks on graphs: A survey,” in *Combinatorics, Paul Erdős is Eighty*, D. Miklós, V. T. Sós, and T. Szőnyi, Eds. Budapest: János Bolyai Mathematical Society, 1996, vol. 2, pp. 353–398.

[70] L. Lan, N. Djuric, Y. Guo, and V. S., “MS-kNN: protein function prediction by integrating multiple data sources,” *BMC Bioinformatics*, vol. 14, no. Suppl 3:S8, 2013.

Marco Frasca received the Ph.D. degree in Computer Science from University of Milano, Italy in 2012. He is currently a post-doc research fellow in Computer Science at the University of Milano. His research interests include the study of neural networks models for unbalanced classification problem and the development of machine learning techniques for emerging problems life sciences, such as protein function prediction, gene-disease prioritization and drug repositioning.

Nicolò Cesa Bianchi is professor of Computer Science at the University of Milano, Italy. He held visiting positions with UC Santa Cruz, Graz Technical University, Ecole Normale Superieure (Paris), Google, and Microsoft Research. He received a Google Research Award and a Xerox University Affairs Committee Award. His research interests include theory and applications of machine learning, sequential optimization, and algorithmic game theory. On these topics, he published two monographs: *Prediction, Learning, and Games* and *Regret Analysis of Stochastic and Non-stochastic Multi-armed Bandit Problems*. 