Bioinformatics, 2017, 1–8
doi: 10.1093/bioinformatics/btx029
Advance Access Publication Date: 14 February 2017
Original Paper

Systems biology

AptRank: an adaptive PageRank model for protein function prediction on bi-relational graphs

Biaobin Jiang[1], Kyle Kloster[2], David F. Gleich[3] and Michael Gribskov[1,3,*]

[1]Department of Biological Sciences, [2]Department of Mathematics and [3]Department of Computer Science, Purdue University, West Lafayette, IN 47907, USA

*To whom correspondence should be addressed.

Associate Editor: Alfonso Valencia

Received on May 23, 2016; revised on December 25, 2016; editorial decision on January 16, 2017; accepted on February 14, 2017

Abstract

Motivation: Diffusion-based network models are widely used for protein function prediction using protein network data and have been shown to outperform neighborhood-based and module-based methods. Recent studies have shown that integrating the hierarchical structure of the Gene Ontology (GO) data dramatically improves prediction accuracy. However, previous methods usually either used the GO hierarchy to refine the prediction results of multiple classifiers, or flattened the hierarchy into a function–function similarity kernel. No study has taken the GO hierarchy into account together with the protein network as a two-layer network model.
Results: We first construct a Bi-relational graph (Birg) model comprised of both protein–protein association and function–function hierarchical networks. We then propose two diffusion-based methods, BirgRank and AptRank, both of which use PageRank to diffuse information on this two-layer graph model. BirgRank is a direct application of traditional PageRank with fixed decay parameters. In contrast, AptRank utilizes an adaptive diffusion mechanism to improve the performance of BirgRank. We evaluate the ability of both methods to predict protein function on yeast, fly and human protein datasets, and compare with four previous methods: GeneMANIA, TMC, ProteinRank and clusDCA. We design four different validation strategies: missing function prediction, *de novo* function prediction, guided function prediction and newly discovered function prediction to comprehensively evaluate predictability of all six methods. We find that both BirgRank and AptRank outperform the previous methods, especially in missing function prediction when using only 10% of the data for training.
Availability and Implementation: The MATLAB code is available at https://github.rcac.purdue.edu/mgribsko/aptrank.
Contact: gribskov@purdue.edu
Supplementary information: Supplementary data are available at *Bioinformatics* online.

1 Introduction

Given a set of functionally uncharacterized genes or proteins from a Genome-Wide Association Study, or differential expression analysis, experimental biologists often have little *a priori* information available to guide the design of hypothesis-based experiments to determine molecular functions. For example, what is the expected phenotype if a particular gene is removed? It would greatly improve hypothesis formation if biologists had prior insight from predicted functions of interesting genes or proteins in databases. Computational annotation of genes or proteins with unknown functions is thus a fundamental research area in computational biology.

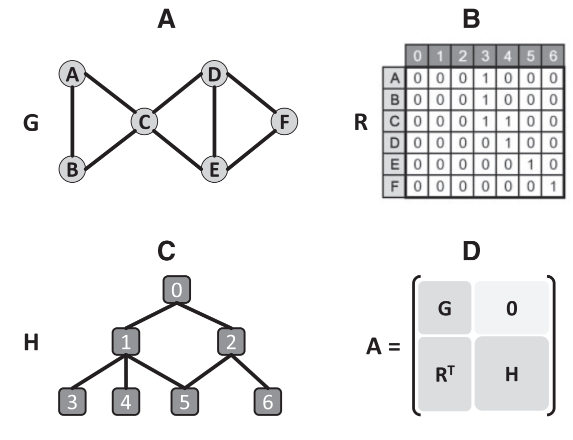
In the past decade, there has been much work to accurately predict functional annotations of genes or proteins using heterogeneous molecular feature data (Peņa-Castillo *et al.*, 2008; Radivojac *et al.*, 2013). The collected molecular features include gene expression, sequence patterns, evolutionary conservation profiles, protein structures and domains, protein–protein interactions (PPIs) and phenotypes or disease associations. In one comprehensive assessment (Peņa-Castillo *et al.*, 2008), one of the methods, GeneMANIA (Mostafavi *et al.*, 2008) slightly outperformed the other eight methods by integrating the multiple molecular features into a functional association network (a.k.a., a kernel). The success story of GeneMANIA suggests two important ideas. First, we can significantly improve prediction methods that rely on a single data type by integrating data of many types. And second, kernel integration is a particularly powerful approach to combining multiple types of data.

Network diffusion is one of the most powerful methods for protein function prediction given an integrated protein association network. This method generally simulates propagating information from functionally known proteins to unknown ones through network connectivity. Nabieva *et al.* (2005) constructed a network flow model with fixed diffusion distances and capacities on network edges. This method was claimed to capture both global network topology as well as local network structure to improve the function predictability over the first two domains of methods mentioned above. Freschi (2007) devised a tool called ProteinRank by utilizing PageRank (Page *et al.*, 1999), the method used by Google to rank webpages, to diffuse functional annotation information throughout a network without setting a fixed diffusion distance or edge capacities. Mostafavi *et al.* (2008) utilized the Label Propagation algorithm (Zhou *et al.*, 2004) to develop GeneMANIA as a classification model with multiple heterogeneous network datasets using weighted kernels and labeled negative samples. The method achieved approximately 70–90% accuracy in three-fold cross validation using a benchmark dataset (Peņa-Castillo *et al.*, 2008). Yu *et al.* (2013) developed the Transductive Multilabel Classifier (TMC), based on a Bi-relational graph (Wang *et al.*, 2011) consisting of a protein interactome and cosine similarities in a protein functional profile as two kernels in each graph layer. Then they used PageRank on this two-layer graph to diffuse functional information to predict protein functions.

Functional annotation data are usually organized in a *tree-like* ontological structure with general terms at the root and specific terms on the leaves (Gene Ontology Consortium, 2004). However, the majority of previous methods disregard this intrinsic hierarchical structure by assuming that the relationships between functions are independent. Recently, several methods have been proposed in order to take into account the interdependent relationships between functional terms in the hierarchical structure. King *et al.* (2003) predicted gene functions using decision trees and Bayesian networks while taking advantage of the annotation dependency between different branches of the GO hierarchy. Notably, when they trained and tested the association of functional terms with genes, they excluded the information from any ancestors and descendants of the terms in question. This ensures a fair cross validation in which prediction does not benefit from the GO annotation rule: if one gene is annotated by a term, then that gene is automatically annotated by all the ancestors of that term. Barutcuoglu *et al.* (2006) and Valentini (2011) proposed a hierarchical Bayesian framework and a True Path Rule, respectively, to perform ensemble learning of the classification results yielded by multiple Support Vector Machines (SVMs). They demonstrated that the accuracy of protein function prediction can be significantly improved by integrating the functional hierarchy (Valentini, 2014). Tao *et al.* (2007) and Pandey *et al.* (2009) utilized Lin's similarity (Lin, 1998) to flatten the functional hierarchy, and then predicted protein functions using a *k*-Nearest Neighbor (*k*-NN) method. Sokolov and Ben-Hur (2010) directly modeled the hierarchical structure of functional ontology using structured SVM (Tsochantaridis *et al.*, 2005), and showed that their method outperformed *k*-NN and other binary classifiers without taking the hierarchy into account. Recently, Yu *et al.* (2015) combined Lin's similarity of protein functional profiles with an ontological hierarchy using downward random walks with restarts, so as to improve the TMC model (Yu *et al.*, 2013), which can predict functions of a protein that are not in its neighborhood, but are present in the hierarchy. Wang *et al.* (2015) proposed clusDCA for protein function prediction by integrating protein networks and a functional hierarchy, using PageRank for network smoothing and low-rank matrix approximation to de-noise the network data.

In this study, we propose two methods that directly diffusing information on the functional hierarchy other than a flat functional similarity constructed by Lin's method (Lin, 1998). The first method, which we call BirgRank, constructs a Bi-relational graph model with a protein–protein functional association network as one layer and an unflattened ontological hierarchy as a second layer, and then directly applies PageRank to diffuse annotation information across the two-layer network. The second method, which we call AptRank, employs an adaptive version of PageRank that replaces the standard PageRank parameters with values dynamically chosen to better fit the training data. The main differences between our methods and other diffusion-based methods are (1) we do not require any negative labeled samples since our method is not a traditional classification model; (2) we take full advantage of the functional hierarchy as a two-way directed graph, and do not use Lin's similarity (Lin, 1998), or any kernel trick, to flatten the hierarchy and (3) we avoid using the annotation of a particular term to predict the annotation of its parental terms, i.e. we train and test our methods using the direct annotations only (see Fig. 1(B) and (C)), which guarantees that the functional terms to be tested for each protein are mutually neither ancestors nor descendants in the GO hierarchy.

To avoid the inflated accuracies of network-based methods in protein function prediction noted by Gillis and Pavlidis (Gillis and Pavlidis, 2011, 2012; Gillis *et al.*, 2014; Pavlidis and Gillis, 2013), we conduct a large and strict evaluation of our methods against the other state-of-the-art methods. In addition to three small benchmark



	0	1	2	3	4	5	6
A	0	0	0	1	0	0	0
B	0	0	0	1	0	0	0
C	0	0	0	1	1	0	0
D	0	0	0	0	1	0	0
E	0	0	0	0	0	1	0
F	0	0	0	0	0	0	1

Fig. 1. Given data visualization using simple example. (**A**) Protein–protein binary interaction network, (**B**) protein–function reference matrix (direct annotations only, no parental annotation used in training and testing), (**C**) function–function hierarchy, (**D**) adjacency matrix *A* of a bi-relational graph

datasets, we use an up-to-date protein interaction network dataset and exclude the functional annotations inferred from protein interactions (evidence code: IPI). Rather than two-fold (Freschi, 2007), three-fold (Mostafavi *et al.*, 2008; Wang *et al.*, 2015) or five-fold (Yu *et al.*, 2013) cross validation, we design four different validations: missing function prediction, *de novo* function prediction, guided function prediction (a hybrid of the two strategies), and newly discovered function prediction. For the first three types of validations, we only use 20% or even 10% of the data in training. To overcome the drawback of using Area Under the ROC curve (AUROC) as a criterion in evaluating performance on imbalanced data with a small number of positive samples, we also utilize Mean Average Precision (MAP) which focuses on the ranking of positive samples only, and is widely used in the field of information retrieval.

2 Materials and methods

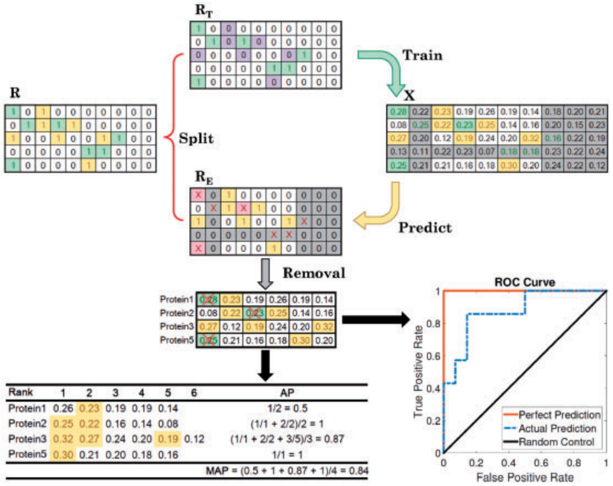
2.1 BirgRank: bi-relational graph PageRank model

This study is motivated by the fact that there are still many proteins whose functions are poorly characterized. The aim of this study is to predict protein functions given a protein–protein association network and a hierarchically structured set of functional terms. The hypothesis is that associated proteins in the protein network are likely to share similar functions. Here, we define a protein–protein association network as pairwise quantitative relationships of proteins. This network either can be sparse and binary, e.g. a protein–protein physical interaction network, or weighted and dense, e.g. a pairwise similarity of protein sequences.

We denote the number of proteins by m and the number of function terms by n. Then the three given datasets (protein–protein association network, protein–function annotations and function–function hierarchy) are denoted by three matrices: $\boldsymbol{G} \in \mathbb{R}^{m\times m}$, a symmetric matrix where $G(i, j)$ denotes to which extent protein i is associated with protein j; $\boldsymbol{R} \in \mathbb{R}^{m\times n}$, a binary matrix where $R(i, j) = 1$ if protein i is annotated by function j, 0 otherwise; and $\boldsymbol{H} \in \mathbb{R}^{n\times n}$, a binary matrix where $H(i, j) = 1$ if functional term i is the child of term j, 0 otherwise. We illustrate these three components in Figure 1(A), (B) and (C), using a small example with 6 proteins and 7 functional terms. For simplicity, Figure 1(A) shows a protein–protein binary interaction network, but it can be replaced by any protein–protein association network. Functional terms are hierarchically structured in a Gene Ontology (Fig. 1(C)) like an upside down 'tree', where the terms on the top (root) are more general and the ones in the bottom (leaves) are more specific. The annotation rule is that if one gene/protein is annotated by one term, then this gene/protein is automatically annotated by all the parental terms of that term in the hierarchy. However, note that in this study we only consider training and predicting the direct annotations of each protein, and do not propagate the corresponding parental annotations using the annotation rule, as shown in Figure 1(B). This ensures that our prediction does not benefit from the annotation rule.

Next, we construct a bi-relational graph (Wang *et al.*, 2011) that incorporates these three datasets into a single network (Fig. 1(D)). To evaluate prediction performance, we split all the annotations in $\boldsymbol{R}$ into $\boldsymbol{R}_T$, which we use for training during model construction, and $\boldsymbol{R}_E$, which we use for evaluating predictions (see Fig. 2). For each protein i, we predict its functions using Personalized PageRank (Jeh and Widom, 2003), a.k.a., Random Walk with Restart (RWR) in other literature (Tong, 2006) by computing

$$(\boldsymbol{I} - \alpha\overline{\boldsymbol{A}})\mathbf{x} = (1-\alpha)\mathbf{v}, \qquad (1)$$



R

1	0	1	0	0	0	0	0	0	0
0	1	1	1	1	0	0	0	0	0
1	0	0	1	0	0	1	1	0	0
0	0	0	0	0	1	1	0	0	0
1	0	0	0	0	1	0	0	0	0

R_T

1	0	0	0	0	0	0	0	0	0
0	1	0	1	0	0	0	0	0	0
0	0	0	0	0	0	0	1	0	0
0	0	0	0	0	1	1	0	0	0
1	0	0	0	0	0	0	0	0	0

X

0.28	0.22	0.23	0.19	0.26	0.19	0.14	0.18	0.20	0.21
0.08	0.25	0.22	0.23	0.25	0.14	0.16	0.20	0.15	0.23
0.27	0.20	0.12	0.19	0.24	0.20	0.32	0.16	0.22	0.19
0.13	0.11	0.22	0.23	0.07	0.18	0.18	0.23	0.22	0.24
0.25	0.21	0.21	0.16	0.18	0.30	0.20	0.24	0.22	0.12

R_E

X	0	1	0	0	0	0	0	0	0
0	X	1	X	1	0	0	0	0	0
1	0	0	1	0	0	1	X	0	0
0	0	0	0	0	X	X	0	0	0
X	0	0	0	0	1	0	0	0	0

Protein1	~~0.28~~	0.23	0.19	0.26	0.19	0.14
Protein2	0.08	0.22	~~0.23~~	0.25	0.14	0.16
Protein3	0.27	0.12	0.19	0.24	0.20	0.32
Protein5	~~0.25~~	0.21	0.16	0.18	0.30	0.20

Rank	1	2	3	4	5	6	AP
Protein1	0.26	0.23	0.19	0.19	0.14		1/2 = 0.5
Protein2	0.25	0.22	0.16	0.14	0.08		(1/1 + 2/2)/2 = 1
Protein3	0.32	0.27	0.24	0.20	0.19	0.12	(1/1 + 2/2 + 3/5)/3 = 0.87
Protein5	0.30	0.21	0.20	0.18	0.16		1/1 = 1
							MAP = (0.5 + 1 + 0.87 + 1)/4 = 0.84

Fig. 2. Missing Function Prediction Strategy. Split the given annotations R, derived from GOA database, by putting 50% of non-zero entries into the training set R_T and the remaining 50% into the evaluation set R_E. Then run one of the six methods to predict the missing entries of R_T using the produced scoring matrix X. Compare the prediction X against R_E to evaluate the performance of the method using AUROC and MAP, respectively. All-zero rows and columns in R_E and the entries used in training are not used in evaluation

where we set the protein i as the diffusion source, i.e. by computing the diffusion using $\mathbf{v} = \mathbf{e}_i$. And we the column-stochastic version of matrix $\boldsymbol{A}$ as $\overline{\boldsymbol{A}}$ which is computed by dividing each column of the matrix $\boldsymbol{A}$ by the sum of the entries in that column. The parameter $\alpha \in (0, 1)$ controls the decay of the diffusion process. To predict the functions of all proteins, we extend the linear system in Equation (1) to a matrix form:

$$\left(\begin{bmatrix} \boldsymbol{I}_m & 0 \\ 0 & \boldsymbol{I}_n \end{bmatrix} - \alpha\overline{\begin{bmatrix} \boldsymbol{G} & 0 \\ \boldsymbol{R}_T^T & \boldsymbol{H} \end{bmatrix}}\right)\begin{bmatrix} \boldsymbol{X}_G \\ \boldsymbol{X}_H \end{bmatrix} = (1-\alpha)\begin{bmatrix} \boldsymbol{I}_m \\ 0 \end{bmatrix}, \qquad (2)$$

where the bar over the block matrix still indicates the whole matrix is normalized to be column-stochastic. The lower block of the solution, $\boldsymbol{X}_H$, is the output matrix of BirgRank for function prediction, and has the same dimensions as $\boldsymbol{R}^T$.

We note that, although PageRank has an interpretation as a Markov chain, and Markov chains must meet certain conditions to guarantee convergence to a stationary distribution, a unique solution to Equation (1) always exists for any $\alpha \in (0, 1)$ and stochastic matrix $\overline{\boldsymbol{A}}$. Thus, the existence of the unique solution $\mathbf{x}$ is guaranteed regardless of the structure of the matrix $\boldsymbol{A}$. We emphasize this because the form of linear system that we use differs from the traditional PageRank setting, which uses Markov chain analysis in the proof of its convergence; in contrast, our computations do not rely on this Markov chain analysis.

To further control the proportion of diffusion passing between the two layers of the bi-relational graph, we parameterize the model in Equation (2) as

$$\left(\begin{bmatrix} \boldsymbol{I}_m & 0 \\ 0 & \boldsymbol{I}_n \end{bmatrix} - \alpha\overline{\begin{bmatrix} \mu\boldsymbol{G} & 0 \\ (1-\mu)\boldsymbol{R}_T^T & \boldsymbol{H}^* \end{bmatrix}}\right)\begin{bmatrix} \boldsymbol{X}_G \\ \boldsymbol{X}_H \end{bmatrix} = (1-\alpha)\overline{\begin{bmatrix} \theta\boldsymbol{I}_m \\ (1-\theta)\boldsymbol{R}_T^T \end{bmatrix}}, \qquad (3)$$

where $\boldsymbol{H}^* = \lambda\boldsymbol{H} + (1-\lambda)\boldsymbol{H}^T$, and λ controls the diffusion direction on $\boldsymbol{H}$. Specifically, $\lambda = 0$ indicates that the diffusion flows down the hierarchy, and 1 indicates flow up the hierarchy. The parameter $\mu \in (0,1)$ controls the proportion of the diffusion flowing within $\boldsymbol{G}$, and $\theta \in (0,1)$ controls the weighted sources between the proteins and functional annotations in the right-hand side of Equation (3).

2.2 Extension to AptRank

In the traditional model of PageRank, which we use in BirgRank, the teleportation parameter $\alpha \in (0,1)$ can be thought of as controlling the rate of decay of the diffusion as it spreads from the nodes in the personalization vector $\mathbf{v}$ to the rest of the graph. After k steps the diffusion has decayed by a factor of α^k, for $k = 1, \ldots, \infty$. There are a variety of other empirical weighting schemes (Baeza-Yates *et al.*, 2006; Chung, 2007; Constantine and Gleich, 2010; Zhu *et al.*, 2014), each with slightly different theoretical properties. For example, the heat kernel diffusion (using coefficients $t^k/k!$ at step k for some fixed input $t > 0$) finds smaller clusters than the standard PageRank diffusion (which uses α^k at step k for a fixed input $\alpha \in (0,1)$) (Kloster and Gleich, 2014). However, one motivation for our current work is to avoid having to choose which coefficients to use by instead computing which coefficients best fit the existing data.

In this section, we seek to replace the standard, fixed diffusion coefficients α^k at each step with an adaptive parameter, denoted by $\gamma^{(k)}$, to optimize the predictive power of the Markov chain. To do this we repeatedly split the training set of protein function annotations, $\boldsymbol{R}_T$, into different subsets to use in fitting and validating the coefficients. We denote the matrix used for fitting by $\boldsymbol{R}_F$, and the matrix used in validation by $\boldsymbol{R}_V$. These matrices have the same dimensions as $\boldsymbol{R}_T$ and consist of entries of $\boldsymbol{R}_T$, i.e. $\boldsymbol{R}_T = \boldsymbol{R}_F + \boldsymbol{R}_V$.

To determine the adaptive coefficients $\gamma^{(k)}$ so that they bias predictions toward the training data, we proceed as follows. The AptRank method begins by computing terms in the following sequence:

$$\boldsymbol{X}^{(k)} = \begin{bmatrix} \boldsymbol{X}_G^{(k)} \\ \boldsymbol{X}_H^{(k)} \end{bmatrix} = \overline{\begin{bmatrix} \boldsymbol{G} & \boldsymbol{R}_F^* \\ \boldsymbol{R}_F^T & \boldsymbol{H}^* \end{bmatrix}}^{k} \boldsymbol{X}^{(0)}, \tag{4}$$

where the bar over the block matrices still denotes column-stochastic normalization,

$$\boldsymbol{X}^{(0)} = \begin{bmatrix} \boldsymbol{X}_G^{(0)} \\ \boldsymbol{X}_H^{(0)} \end{bmatrix} = \begin{bmatrix} \boldsymbol{I}_m \\ 0 \end{bmatrix}, \tag{5}$$

and

$$\boldsymbol{R}_F^* = \begin{cases} 0 & \text{to use a one} - \text{way diffusion} \\ \boldsymbol{R}_F & \text{to use a two} - \text{way diffusion} \end{cases}.$$

We denote AptRank using a one-way diffusion and a two-way diffusion as AptRank-1 and AptRank-2, respectively. These two variations can have significant differences in prediction performance when the underlying networks have different sparsities.

To compute the optimal set of coefficients $\gamma^{(k)}$ that best fits the validation set $\boldsymbol{R}_V$, we solve the following constrained least squares model,

$$\begin{aligned} &\text{minimize}_{\gamma} && \left\| \operatorname{vec}(\boldsymbol{R}_V^T) - \sum_{k=1}^{K} \gamma^{(k)} \operatorname{vec}(\boldsymbol{X}_H^{(k)}) \right\|_2^2 \\ &\text{subject to} && \sum_{k=1}^{K} \gamma^{(k)} = 1, \\ &&& \gamma^{(k)} \geq 0, \end{aligned} \tag{6}$$

where $\operatorname{vec}(\cdot)$ is a matrix-to-vector transformation that stacks the columns of the matrix into a single column vector.

The entire AptRank framework is summarized in Algorithm 1. We perform this fitting-validating process S times, each time (denoted as s) splitting t% of entries in $\boldsymbol{R}_T$ into new matrices $\boldsymbol{R}_F$ and $\boldsymbol{R}_V$ by choosing entries from $\boldsymbol{R}_T$ uniformly at random. Each such iteration generates a new set of coefficients $\gamma_s^{(k)}$, which we store. We call these iterations 'shuffles' because in essence they consist of shuffling the entries of $\boldsymbol{R}_T$ into the two matrices $\boldsymbol{R}_F$ and $\boldsymbol{R}_V$. After the prescribed number of shuffles is completed, we compute the median of the $\gamma_s^{(k)}$ across all shuffles, denoted as $\gamma_*^{(k)}$, and then use those median values to compute the final diffusion values $\boldsymbol{X}_{\text{AptRank}}$. This prediction solution will be compared against the evaluation set $\boldsymbol{R}_E$ (see Section 3).

To investigate the similarities and differences of our methods and the other four previous methods used for evaluation, we summarize the features of each method in Table 1. A more detailed comparison of each method in theory can be found in the Supplementary Text.

Algorithm 1: AptRank

```
Input  : $\boldsymbol{G}$,$\boldsymbol{R}_T$,$\boldsymbol{H}^*$,$K$,$S$,$t$
Output: $\boldsymbol{X}_{\text{AptRank}}$
for $s \leftarrow 1$ to $S$ do
    $[\boldsymbol{R}_F, \boldsymbol{R}_V] \leftarrow$ splitR($\boldsymbol{R}_T$,$t$)
    // Choose $t\%$ of nonzero entries in $\boldsymbol{R}_T$ uniformly at random
    and split to $\boldsymbol{R}_F$, and derive $\boldsymbol{R}_V = \boldsymbol{R}_T - \boldsymbol{R}_F$.
    Initialize $\boldsymbol{X}^{(0)}$ using Equation (5)
    for $k \leftarrow 1$ to $K$ do
        Compute $\boldsymbol{X}^{(k)}$ using Equation (4)
        $\boldsymbol{A}[:,k] \leftarrow$ vec($\boldsymbol{X}_H^{(k)}$)
    end
    $[\boldsymbol{Q}_A,\boldsymbol{R}_A] \leftarrow$ qr($\boldsymbol{A}$) // QR decomposition
    $\mathbf{b} \leftarrow$ vec($\boldsymbol{R}_V$)
    Solve $\underset{\boldsymbol{\gamma}_s^{(k)}}{\text{minimize}} \; \|\boldsymbol{Q}_A^T\mathbf{b} - \boldsymbol{R}_A\boldsymbol{\gamma}_s^{(k)}\|_2^2$
          subject to $\sum_k \gamma_s^{(k)} = 1, \gamma_s^{(k)} \geq 0$
    // Equivalently as Equation (6).
end
$\boldsymbol{\gamma}_*^{(k)} \leftarrow$ median($\boldsymbol{\gamma}_s^{(k)}$)
// Take the median over all $s = 1$ to $S$ for each $k$.
$\begin{bmatrix} \boldsymbol{X}_G^* \\ \boldsymbol{X}_H^* \end{bmatrix} \leftarrow \sum_{k=1}^{K} \gamma_*^{(k)} \overline{\begin{bmatrix} \boldsymbol{G} & \boldsymbol{R}_T^* \\ \boldsymbol{R}_T^T & \boldsymbol{H}^* \end{bmatrix}}^{k} \begin{bmatrix} \boldsymbol{I}_m \\ \mathbf{0} \end{bmatrix}$
Output $\boldsymbol{X}_{\text{AptRank}} \leftarrow \boldsymbol{X}_H^*$ for use in prediction.
```

3 Results

3.1 Experimental setup

We present a comprehensive evaluation of the six methods using three benchmark datasets from yeast, human and fly which were collected by the developers of GeneMANIA-SW in 2010, and can be accessed via http://morrislab.med.utoronto.ca/~sara/SW/. Additionally, we collected one more dataset for human proteins from public databases in March 2015 in order to test all the methods using up-to-date data with a larger size than those collected by GeneMANIA-SW in 2010 (see Table 2). In this human dataset, denoted as human-2015, the network $\boldsymbol{G}$ was downloaded from BioGRID (Stark *et al.*, 2006), and the annotations $\boldsymbol{R}$ and the hierarchy $\boldsymbol{H}$ from the Gene Ontology Consortium (Gene Ontology Consortium, 2015). We primarily investigate and discuss the annotations in the Biological Process (BP)

Table 1. Summary of the six methods

Method Name	Method Type	Functional Hierarchy	Bi-relational Graph	Negative Samples	Random Walk	Stationary PageRank	Reference
GeneMANIA-SW	kernel integration & classification			✓	✓	✓	Mostafavi et al. (2008) and Mostafavi and Morris (2010)
TMC	diffusion		✓		✓	✓	Yu et al. (2013)
ProteinRank	regression				✓	✓	Freschi (2007)
DCA-clusDCA	diffusion & decomposition	✓		✓	✓	✓	Cho et al. (2015) and Wang et al. (2015)
BirgRank	diffusion	✓	✓		✓	✓	This study
AptRank	diffusion	✓	✓		✓		This study

Table 2. Statistics of datasets

Dataset	No. of proteins	No. of direct GO	No. of all GO	No. of kernels
Yeast	3904	1188	1695	44
Human-2010	13281	1952	2919	8
Fly	13562	2195	2919	38
Human-2015	14515	11519	27106	1

category in the main text. The results for the prediction of Molecular Function (MF) and Cellular Component (CC) terms can be found in Supplementary Figure S3. Also, we only use annotations with experimental evidence codes, within which we remove the terms inferred by physical interaction (evidence code: IPI). Within these annotations, we only train and test the direct GO annotations (Table 2, 3rd column) without consideration of the parental annotations (see total number of annotations in Table 2, 4th column). The multiple kernels (Table 2, 5th column) from heterogeneous molecular data were directly downloaded from the GeneMANIA-SW website, and combined into a single network (i.e. $\boldsymbol{G}$) with the weights provided in the datasets.

To evaluate the quality of each method in protein function prediction, we conducted cross validation using three different strategies to split the given functional annotation data $\boldsymbol{R}$ into $\boldsymbol{R}_T$ used for training and $\boldsymbol{R}_E$ used for evaluation (see Section 3.2). The three strategies are:

1. missing function prediction
2. *de novo* function prediction
3. guided function prediction.

All three validation strategies ensure that the matrices $\boldsymbol{R}$, $\boldsymbol{R}_T$ and $\boldsymbol{R}_E$ have the same dimensions, and $\boldsymbol{R} = \boldsymbol{R}_T + \boldsymbol{R}_E$. To measure the prediction quality of each method, we use two evaluation metrics: AUROC (Area Under the Receiver Operating Characteristic curve) which is widely used in protein function prediction, and MAP (Mean Average Precision) which is widely used in information retrieval (Fig. 2). The key advantage of MAP is that MAP does not take true negatives into account, and is thus a more informative metric than AUROC when negative samples outnumber positive samples. This is true in our case since in the human-2015 dataset, for example, we attempt to predict around 45 functions on average for each protein from 11 519 possible annotations (feature space, see Table 2). In addition to the three cross validation, we evaluate each method by using human-2010 dataset to predict human-2015 dataset to investigate whether the methods can predict newly discovered functional annotations from 2010 to 2015.

We determined parameter settings as follows. For the four methods other than our BirgRank and AptRank, we mostly used the default settings specified in the corresponding literature. We only tuned the reduced dimensionality d in clusDCA to be 500, rather than the parameter setting 2500 specified by the authors (Wang *et al.*, 2015), since this parameter is a key factor in time complexity of clusDCA. Empirically, we found that clusDCA is the most time-consuming method as shown in Table 3, and a large d value dramatically increases running time. For the parameters in BirgRank, we set $\lambda = 0.5$ in determining $\boldsymbol{H}^*$, to allow equal diffusion upward and downward the hierarchy. For the other three parameters α, θ and μ in BirgRank (see Equation (3)), we observed that different settings of these three parameters did not yield significant differences in performance, and found that a value of 0.5 empirically achieved good results (Supplementary Fig. S1). For the parameters in AptRank, we set the total iteration number K to be 8, the splitting parameter t to be 50%, and the number of shuffles S to be 5. These setting may vary depending on the validation strategies and the data sizes, which we discuss in Section 3.2.

Table 3. Runtimes of the six methods in minutes (human-2015 dataset)[a]

Methods	Training data proportion					
	10%	20%	40%	50%	70%	80%
GM-SW	252.52	214.47	232.02	231.65	225.54	234.56
TMC	6.71	7.10	7.52	7.58	7.37	7.12
ProteinRank	0.85	0.87	0.87	0.87	0.88	0.88
clusDCA	1054	1019	1072	1061	1025	1050
BirgRank	9.42	9.46	9.46	9.45	9.42	9.49
AptRank-1	51.79	53.48	55.82	55.28	57.85	58.69

[a] The runtimes of 30% and 60% is not shown due to space limit. The AptRank-1 uses 12-core parallel computing for matrix multiplication.

3.2 Comparison of prediction performances

3.2.1 Missing function prediction

We first conducted a numerical experiment to evaluate the ability of the six methods in predicting missing protein functions as follows. We uniformly select a certain percentage of non-zero entries in $\boldsymbol{R}$ at random, move them to a matrix $\boldsymbol{R}_T$ for training, and let $\boldsymbol{R}_E = \boldsymbol{R} - \boldsymbol{R}_T$ be the evaluation set. Figure 2 illustrates how to split matrix $\boldsymbol{R}$ with 14 entries into $\boldsymbol{R}_T$ and $\boldsymbol{R}_E$ when the splitting percentage is specified as 50%. We carried out this random sampling with replacement 5 times for each specified splitting percentage. This is not a circular cross validation since it does not guarantee that each functional annotation is tested once and only once. This strategy aims to test whether the methods can restore incomplete functional annotations for each protein and is unbiased with respect to how many annotations each protein has.

We start with 10% split for training and increase by increments of 10% up to 80% (Fig. 3). Generally, the resulting AUROCs and MAPs of the six methods show that both BirgRank and AptRank outperform the other four previous methods in all 8 groups of experiments with different amounts of training data. In the 10% group of human-2010 and fly datasets, clusDCA slightly outperforms our methods in AUROC, but its MAP is lower than those of our methods (Fig. 3(C) and (E)). When more data are given for training, our methods outperform the other four methods in terms of MAP with approximately 2- to 3-fold improvement.

To investigate the effect of the GO functional hierarchy in prediction, we compare the performance of non-hierarchy-integrated methods (GeneMANIA-SW, TMC and ProteinRank) with hierarchy-integrated methods (clusDCA, BirgRank and AptRank). We find that the integration of the functional hierarchy clearly improves the prediction accuracy (Fig. 3). Furthermore, our methods, for the most part, perform better than clusDCA, which suggests that using a bi-relational graph framework (Fig. 1) to integrate the hierarchy is better than seeking for projection between the protein network and the functional hierarchy. The significance of the GO hierarchical structure was demonstrated by replacing it into a random graph, a hierarchy with shuffled labels, and an identity matrix, respectively (see Supplementary Fig. S2).

Comparing the performances of BirgRank and AptRank, we find that the performance of the algorithms differs as the network sparsity varies (Fig. 3 (B), (D), (F) vs. (H)). The three benchmark datasets are smaller and denser than Human-2015 dataset due to the integration of multiple kernels (Table 2). We can see that AptRank with a two-way diffusion performs better on the dense network, while BirgRank is better on the sparse network. This could be because a dense network restricts network diffusion within a local region of the source node, and two-way diffusion forms a feedback loop that enhances the contributions of the annotations within local regions. However, the two-way diffusion spreads out of this local region in a sparse network and provides irrelevant feedback to the source node.

In addition, we find that GeneMANIA-SW and ProteinRank achieve similar performance in both AUROC and MAP. The key difference between these two models is that GeneMANIA-SW requires negative samples in its classification framework. This demonstrates that negative samples have a very limited contribution to the performance of GeneMANIA-SW on these datasets. This could be in part because it can be difficult to confirm that a protein does not have a function.

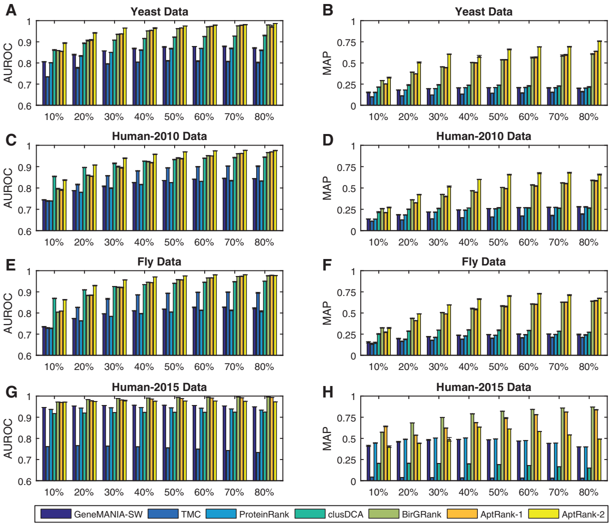


Fig. 3. Missing function prediction. The *x*-axis represents the percentages of data used in training. The error mark on top of each bar indicates the standard deviation of AUROCs or MAPs over 5 repetitions of each experiment

Lastly, we find that BirgRank outperforms TMC. Theoretically, the models of TMC and BirgRank are quite similar, differing mainly in how the two methods direct the diffusion between the two network layers, $\boldsymbol{G}$ and $\boldsymbol{H}$. BirgRank diffuses information from $\boldsymbol{G}$ to $\boldsymbol{H}$, while TMC does the reverse. Our results support the idea that diffusion from proteins to functional terms is the more useful direction in the context of protein function prediction.

3.2.2 De novo function prediction

To investigate whether the six methods can accurately predict the functions of one protein without any annotation for training, we design a *de novo* circular cross validation as follows. Uniformly partition a certain percentage, denoted as c, of proteins into b groups at random. Letting $[v]$ denote the nearest-integer operation we can calculate

$$b = \begin{cases} [1/c] & \text{if } 0 < c \le 0.5 \\ [1/(1-c)] & \text{if } 0.5 < c \le 1 \end{cases}.$$

In practice, we set c as 20%, 50% and 80% as shown in the *x*-axis of Figure 4. When $c = 80\%$, it is equivalent to a conventional five-fold cross validation with 80% of proteins as the training set and the complementary 20% as the evaluation set. On the contrary, $c = 20\%$ means we only use 20% of proteins for training and evaluate the prediction performance by the complementary 80%. Lastly, $c = 50\%$ is equivalent to a two-fold cross validation. Normally, three-fold cross validation ($c = 66.7\%$) is used in the four reference methods. Here, our cross validation design is aimed to explore the potential predictive power of all of the methods with a more stringent criterion.

As shown in Figure 4, our methods generally perform no worse than the four reference methods. Interestingly, GeneMANIA has nearly the same performance as ProteinRank in both AUROC and MAP metrics, which occurs in our missing function prediction experiment as well (Fig. 3). Furthermore, they both perform better than the other two reference methods, TMC and clusDCA. Our methods perform slightly better than GeneMANIA and ProteinRank in AUROC, but do slightly worse in MAP. This leads us to conclude that (1) a classification model that includes negative samples (GeneMANIA) is little different from a diffusion model (ProteinRank) in *de novo* function prediction; and (2) integrating the GO hierarchy (BirgRank and AptRank) cannot significantly improve the accuracy in function

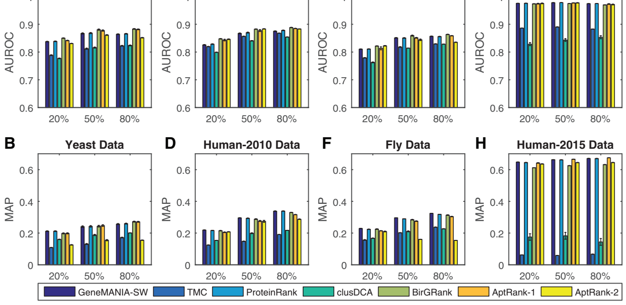


Fig. 4. *De novo* function prediction. The *x*-axis represents the percentages of data used in training. The error mark on top of each bar indicates the standard deviation of AUROCs or MAPs over 3 repetitions of each experiment

prediction for newly found proteins without known functional information.

3.2.3 Guided function prediction

To examine the extent to which our methods benefit from limited known annotations of tested proteins, we devise a validation strategy called guided function prediction which is a hybrid of the missing function prediction (Section 3.2.1) and the *de novo* prediction (Section 3.2.2) strategies. In this validation, the strategy of partitioning training and evaluation sets is identical to that used in *de novo* prediction except that it gives *one* functional annotation as guidance for each evaluated protein that has more than one annotation. The proteins in the evaluation set with only one or no annotation are not taken into account.

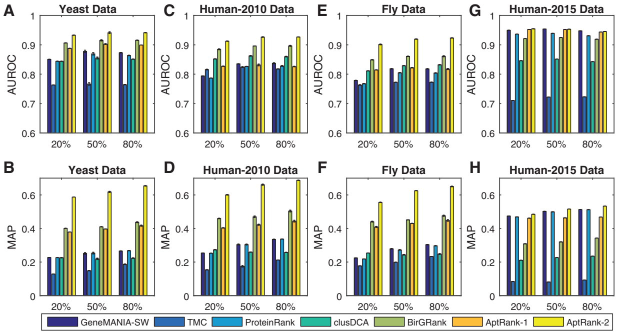


Fig. 5. Guided function prediction. The *x*-axis represents the percentages of data used in training. The error mark on top of each bar indicates the standard deviation of AUROCs or MAPs over 3 repetitions of each experiment

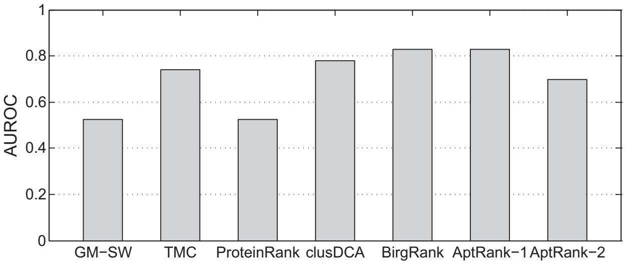


Fig. 6. Performance of predicting human-2015 using human-2010

Table 4. Medians of γ in prediction of yeast and human-2015 datasets

Dataset	Training (%)	Markov chain iteration							
		1st	2nd	3rd	4th	5th	6th	7th	8th
Yeast	10%	0	0	0	0	0	0	0.08	0.92
	20%	0	0.11	0	0	0	0	0.23	0.66
	30%	0	0.34	0	0.08	0	0	0.58	0
	40%	0	0	0	0	0	0	1	0
	50%	0	0	0	0	0.84	0	0.16	0
	60%	0	0	0	0	1	0	0	0
	70%	0	0	0.09	0	0.91	0	0	0
	80%	0	0	0.64	0	0.36	0	0	0
Human 2015	10%	0	0.20	0	0	0	0	0.31	0.49
	20%	0	0.65	0	0	0	0	0.11	0.24
	30%	0	1	0	0	0	0	0	0
	40%	0	1	0	0	0	0	0	0
	50%	0	1	0	0	0	0	0	0
	60%	0	1	0	0	0	0	0	0
	70%	0	1	0	0	0	0	0	0
	80%	0	1	0	0	0	0	0	0

We can see in Figure 5 that in the evaluations using the three benchmark datasets with dense network data, our methods, especially AptRank-2, can take full advantages of the single given annotation to improve prediction performance by approximately 2-fold in AUROC and 3-fold in MAP, compared to the other four methods. In the sparse network data (Human-2015), we find that the given annotations worsen the performances of all the methods (Fig. 4(G, H) vs. Fig. 5(G, H)). We conclude that sparse network datasets may cause underfitting of our model training, and reducing the model complexity can alleviate this problem, e.g. setting a small α in BirgRank or a small K in AptRank. On the contrary, we also find that in some experiments, the more data we provide for training, the worse the testing accuracy is (e.g. AptRank-2 in Fig. 4(F)). In these cases, Verleyen *et al.* (2015) proposed using sampling of the training data to overcome this overfitting.

3.2.4 Newly discovered function prediction

In addition to separate evaluations of prediction using human-2010 and human-2015 datasets, we performed prediction using the human-2010 data as the training set and the human-2015 data as the testing set. Denote the three input matrices of human-2010 as $\boldsymbol{G}^{(0)}$, $\boldsymbol{R}^{(0)}$ and $\boldsymbol{H}^{(0)}$, and those of human-2015 as $\boldsymbol{G}^{(5)}$, $\boldsymbol{R}^{(5)}$ and $\boldsymbol{H}^{(5)}$. In this task, we input $\boldsymbol{G}^{(5)}$, $\boldsymbol{R}^{(0)}$ and $\boldsymbol{H}^{(5)}$ into each method, and then evaluated their predictions using $\boldsymbol{R}^{(5)}$ as the testing set. In Figure 6, we show that BirgRank and AptRank-1 achieve higher AUROCs than the previous methods in this task, which demonstrates the ability of our methods to successfully predict new protein functions in human discovered from 2010 to 2015.

3.3 Analysis of adaptive coefficients

The adaptive coefficients of AptRank (γ) are the unique feature that differs from traditional PageRank. To investigate their behaviors in prediction, we list the medians of γ over the different shuffles in the prediction of yeast and human-2015 datasets in Table 4. We can see that there are three main features of γ's behaviors,

1. $\gamma^{(1)}$ is always zero, since the information diffusing within $\boldsymbol{G}$, from proteins at the first step, has not yet reached the hierarchy;
2. as shown in the yeast dataset, the distribution of γ is not uniform, but concentrates on specific terms of Markov chains, which demonstrates that AptRank can adaptively select the most predictive terms rather than weighting all terms with power-decays like traditional PageRank; and
3. in comparison of γ in yeast and human-2015 datasets, we find that AptRank mostly selects the 2nd term in the human-2015 dataset, but a few more terms in the yeast dataset, which is due to the different network densities of the two datasets. The yeast dataset is smaller but denser, since it integrates 44 different kernels into $\boldsymbol{G}$; the human-2015 dataset is larger but sparser, and all the entries in the raw human-2015 dataset are binary. This implies that for a sparse dataset, our AptRank might be equivalent to neighbor-voting methods.

3.4 Comparison of runtimes

The average computational time of the six methods compared in this study are shown in Figure 4. In this comparison, the computational time is recorded for the prediction using the largest dataset, human-2015. We can clearly see AptRank requires the third longest computational time, likely because it involves many dense matrix operations.

The SVD computations required in clusDCA are likely responsible for clusDCA having the longest running time. Without a parallel implementation of SVD, clusDCA might be impractical unless we sacrifice prediction accuracy by using a small *d* value. GeneMANIA-SW is the second most computationally expansive method, since it computes the prediction scores function by function. This is extremely expensive when the number of functions is large, even though we only used direct GO terms in GeneMANIA-SW. BirgRank and TMC both use bi-relational graphs, and take only several minutes to solve the PageRank linear system. ProteinRank has the most simple model, and it takes the shortest time, since it needs only to solve a PageRank linear system with approximately half the dimension of the systems involved in BirgRank and TMC.

4 Conclusion

In this paper we present two network-diffusion-based methods for protein function prediction. Our first method, BirgRank, uses PageRank on a bi-relational graph model that incorporates protein–protein and function–function networks. Our second method, AptRank, introduces an adaptive mechanism to the PageRank framework that computes an optimal set of weights for the first several steps of diffusion so as to maximize recovery of a subset of known function annotations. We show that both methods outperform the four existing state-of-the-art methods in almost all cases, and in particular, outperform those methods that do not incorporate information about the functional hierarchy. Our results also suggest that diffusion-based methods are still among the most competitive in network-based protein function predictions, compared to classification-based and decomposition-based methods.

Funding

This research has been supported in part by NSF CAREER CCF-1149756, NSF CAREER CCF-093937 and DARPA SIMPLEX.

Conflict of Interest: none declared.

References

Baeza-Yates,R. *et al.* (2006). Generalizing PageRank: Damping functions for link-based ranking algorithms. In: *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 308–315. ACM.

Barutcuoglu,Z. *et al.* (2006) Hierarchical multi-label prediction of gene function. *Bioinformatics*, **22**, 830–836.

Cho,H. *et al.* (2015). Diffusion component analysis: Unraveling functional topology in biological networks. In: *Research in Computational Molecular Biology*, pp. 62–64. Springer.

Chung,F. (2007) The heat kernel as the PageRank of a graph. *Proc. Natl. Acad. Sci. U. S. A.*, **104**, 19735–19740.

Constantine,P.G. and Gleich,D.F. (2010) Random alpha PageRank. *Internet Math.*, **6**, 189–236.

Freschi,V. (2007). Protein function prediction from interaction networks using a random walk ranking algorithm. In: *Bioinformatics and Bioengineering, 2007. BIBE 2007. Proceedings of the 7th IEEE International Conference on*, pp. 42–48. IEEE.

Gene Ontology Consortium (2004) The Gene Ontology (GO) database and informatics resource. *Nucleic Acids Res.*, **32**, D258–D261.

Gene Ontology Consortium (2015) Gene Ontology Consortium: going forward. *Nucleic Acids Res.*, **43**, D1049–D1056.

Gillis,J. and Pavlidis,P. (2011) The impact of multifunctional genes on "guilt by association" analysis. *PloS One*, **6**, e17258.

Gillis,J. and Pavlidis,P. (2012) "Guilt by association" is the exception rather than the rule in gene networks. *PLoS Comput. Biol.*, **8**, e1002444.

Gillis,J. *et al.* (2014) Bias tradeoffs in the creation and analysis of protein–protein interaction networks. *J. Proteomics*, **100**, 44–54.

Jeh,G. and Widom,J. (2003). Scaling personalized web search. In: *Proceedings of the 12th International Conference on the World Wide Web*, pp. 271–279. ACM, Budapest, Hungary.

King,O.D. *et al.* (2003) Predicting gene function from patterns of annotation. *Genome Res.*, **13**, 896–904.

Kloster,K. and Gleich,D.F. (2014). Heat kernel based community detection. In: *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1386–1395. ACM.

Lin,D. (1998). An information-theoretic definition of similarity. *ICML*, **98**, 296–304.

Mostafavi,S. and Morris,Q. (2010) Fast integration of heterogeneous data sources for predicting gene function with limited annotation. *Bioinformatics*, **26**, 1759–1765.

Mostafavi,S. *et al.* (2008) GeneMANIA: a real-time multiple association network integration algorithm for predicting gene function. *Genome Biol.*, **9**, S4.,

Nabieva,E. *et al.* (2005) Whole-proteome prediction of protein function via graph-theoretic analysis of interaction maps. *Bioinformatics*, **21**, i302–i310.

Page,L. *et al.* (1999) The PageRank citation ranking: bringing order to the web. *Tech. Report 1999-66*, Stanford University, Stanford, CA. pp. 1–17.

Pandey,G. *et al.* (2009) Incorporating functional inter-relationships into protein function prediction algorithms. *BMC Bioinformatics*, **10**, 142.

Pavlidis,P. and Gillis,J. (2013). Progress and challenges in the computational prediction of gene function using networks: 2012–2013 update. *F1000Research*, **2**, 1–13.

Peņa-Castillo,L. *et al.* (2008) A critical assessment of mus musculus gene function prediction using integrated genomic evidence. *Genome Biol.*, **9**, S2.

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

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

Stark,C. *et al.* (2006) BioGRID: a general repository for interaction datasets. *Nucleic Acids Res.*, **34**, D535–D539.,

Tao,Y. *et al.* (2007) Information theory applied to the sparse gene ontology annotation network to predict novel gene function. *Bioinformatics*, **23**, i529–i538.

Tong,H. (2006). Fast random walk with restart and its applications. In: *Proceedings of the 6th IEEE International Conference on Data Mining*.

Tsochantaridis,I. *et al.* (2005) Large margin methods for structured and interdependent output variables. *J. Mach. Learn. Res.*, **6**, 1453–1484.

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

Valentini,G. (2014) Hierarchical ensemble methods for protein function prediction. *Int. Sch. Res. Notices*, **2014**, 1–34.

Verleyen,W. *et al.* (2015) Positive and negative forms of replicability in gene network analysis. *Bioinformatics*, btv734.

Wang,H. *et al.* (2011). Image annotation using bi-relational graph of images and semantic labels. In: *2011 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 793–800. IEEE.

Wang,S. *et al.* (2015) Exploiting ontology graph for predicting sparsely annotated gene function. *Bioinformatics*, **31**, i357–i364.

Yu,G. *et al.* (2013) Protein function prediction using multi-label ensemble classification. *IEEE/ACM Trans. Comput. Biol. Bioinf. (TCBB)*, **10**, 1–1.

Yu,G. *et al.* (2015) Predicting protein function via downward random walks on a gene ontology. *BMC Bioinformatics*, **16**, 271.

Zhou,D. *et al.* (2004) Learning with local and global consistency. *Adv. Neural Inf. Process. Syst.*, **16**, 321–328.

Zhu,X. *et al.* (2014). An adaptive teleportation random walk model for learning social tag relevance. In: *Proceedings of the 37th International ACM SIGIR Conference on Research & Development in Information Retrieval*, pp. 223–232. ACM.