BIOINFORMATICS

Vol. 30 ISMB 2014, pages i219–i227
doi:10.1093/bioinformatics/btu263

New directions for diffusion-based network prediction of protein function: incorporating pathways with confidence

Mengfei Cao[1], Christopher M. Pietras[1], Xian Feng[1], Kathryn J. Doroschak[2], Thomas Schaffner[1], Jisoo Park[1], Hao Zhang[1], Lenore J. Cowen[1,*] and Benjamin J. Hescott[1,*]

[1]Department of Computer Science, Tufts University, Medford, MA 02155, USA and [2]Department of Computer Science, University of Minnesota, Minneapolis, MN 55455, USA

ABSTRACT

Motivation: It has long been hypothesized that incorporating models of network noise as well as edge directions and known pathway information into the representation of protein–protein interaction (PPI) networks might improve their utility for functional inference. However, a simple way to do this has not been obvious. We find that diffusion state distance (DSD), our recent diffusion-based metric for measuring dissimilarity in PPI networks, has natural extensions that incorporate confidence, directions and can even express coherent pathways by calculating DSD on an augmented graph.

Results: We define three incremental versions of DSD which we term cDSD, caDSD and capDSD, where the capDSD matrix incorporates confidence, known directed edges, and pathways into the measure of how similar each pair of nodes is according to the structure of the PPI network. We test four popular function prediction methods (majority vote, weighted majority vote, multi-way cut and functional flow) using these different matrices on the Baker's yeast PPI network in cross-validation. The best performing method is weighted majority vote using capDSD. We then test the performance of our augmented DSD methods on an integrated heterogeneous set of protein association edges from the STRING database. The superior performance of capDSD in this context confirms that treating the pathways as probabilistic units is more powerful than simply incorporating pathway edges independently into the network.

Availability: All source code for calculating the confidences, for extracting pathway information from KEGG XML files, and for calculating the cDSD, caDSD and capDSD matrices are available from http://dsd.cs.tufts.edu/capdsd

Contact: lenore.cowen@tufts.edu or benjamin.hescott@tufts.edu

Supplementary information: Supplementary data are available at *Bioinformatics* online.

1 INTRODUCTION

One of the most well-studied problems in computational network biology is the prediction of protein functional labels from distance and neighborhood structure in the protein–protein interaction network (PPI network). In 2013, based on the observation that paths through high-degree 'hub' nodes in the PPI network were less informative than short paths through protein nodes with fewer interaction partners, (Cao *et al*., 2013) introduce the diffusion state distance (DSD) metric that is able to quantify topological similarity in a PPI network in a more fine-grained way. Diffusion-based methods had been previously proposed for clustering similar proteins (Voevodski *et al*., 2009) and for ranking candidate disease genes (Chen *et al*., 2009; Erten *et al*. 2011; Kohler *et al*., 2008; Vanunu *et al*., 2010), but by explicitly taking an L1 norm of the vector of the random walks to all other nodes in the network to measure the distance between nodes, DSD is able to capture a more global view of the network than other prior work we are aware of, with the exception of Vavien (Erten *et al*. 2011) for candidate disease gene ranking, and ISORANK-N (Liao *et al*., 2009), which also is based on a global embedding, but for a very different problem (network alignment).

Cao *et al*. (2013) showed that when a DSD-based distance is substituted for ordinary next-hop shortest-path distance in four classical network-based function prediction methods, functional label prediction performance for the GO (Gene Ontology), as well as all three levels of the MIPS (Munich Information Center For Protein Sequences) ontology, improved across the board in cross-validation experiments on both the *Saccharomyces cerevisiae* and the *S.pombe* PPI networks. However, these results were based only on a simple undirected model of the PPI network, which additionally assumed that all the edges listed in the BioGRID data were uniformly correct.

On the other hand, it is well-established both that there is noise in the PPI interaction network data (Mering *et al*., 2002; Reguly *et al*., 2006; Gandhi *et al*., 2006), and that some interactions are naturally directed in the PPI network (Liu *et al*., 2009; Gitter *et al*., 2011; Du *et al*., 2012). In addition, looking just at pairwise interaction data as edges does not fully capture all the information that is known about the PPI network. In particular, there is increasingly available data on biological pathways, for example, TGF-β binds TGF-β receptor 1, which phosphorylates Smad3, which with importin-β1 enters the nucleus and binds DNA to regulate expression (Moustakas 2002).

In this article, we revisit the DSD metric we designed in earlier work for function prediction in the ordinary undirected PPI network. We find that its diffusion-based framework gives a natural way to incorporate edge confidences and directed edges (when known). However, the main contribution of this article is to show that there is a way to capture the cohesiveness of known pathways by calculating DSD on an augmented network, and that this way of representing pathways results in better performance than just incorporating the pathway edges themselves for most, but not all of the function prediction methods we study. We show this first in cross-validation on the standard network consisting of just experimentally verified physical interaction edges from *S.cerevisiae*, and then on an integrative network with heterogeneous protein association data edges derived from the STRING database (Franceschini *et al*., 2013).

*To whom correspondence should be addressed.

1.1 Overview of DSD

PPI networks are known to be 'small world' networks in the sense that they are small-diameter, and most nodes are close to all other nodes. Thus any method that infers similarity based on proximity will find that a large fraction of the network is proximate to any typical node. In fact, this issue has already been termed the 'ties in proximity' problem in the computational biology literature (Arnau *et al*., 2005).

Furthermore, the fact that two particular nodes are adjacent (i.e., have shortest-path distance 1) in a PPI network can signify something very different than the adjacency of two other nodes. For example, in PPI networks two nodes with many low-degree neighbors in common should be thought of as 'more similar' than nodes with few low-degree neighbors in common; and such nodes should also be thought of as 'more similar' than two nodes whose common neighbors have high-degree. Thus, characterizing node pairs based only on a shortest-path notion of distance fails to capture important knowledge encoded in the structure of the network.

In (Cao *et al*., 2013), DSD is defined on an undirected connected simple graph. In particular, our PPI network is defined with a vertex set V, containing a node for each verified ORF, and an edge set E, containing an unweighted and undirected edge for each physical interaction. We first calculate $He^{\{k\}}(A,B)$ as the expected number of times that a random walk starting at node A and proceeding for k steps, will visit node B; then we further define a n-dimensional vector $He^{\{k\}}(v_i), \forall v_i \in V$, where

$$He^{\{k\}}(v_i) = (He^{\{k\}}(v_i, v_1), He^{\{k\}}(v_i, v_2), ..., He^{\{k\}}(v_i, v_n)).$$

In what follows, the k-step DSD between two vertices u and v, $\forall u, v \in V$ is defined as

$$\mathrm{DSD}^{\{k\}}(u, v) = ||He^{\{k\}}(u) - He^{\{k\}}(v)||_1,$$

where $||He^{\{k\}}(u) - He^{\{k\}}(v)||_1$ denotes the L_1 norm of the He vectors of u and v. As proved in (Cao *et al*., 2013), on the simple connected graph whose random walk one-step transition probability matrix is diagonalizable and ergodic as a Markov chain, the limit of DSD when k approaches infinity exists and can be calculated as

$$\lim_{k\to\infty} \mathrm{DSD}^{\{k\}}(u, v) = ||(\boldsymbol{b_u}^T - \boldsymbol{b_v}^T)(I - P + C)^{-1}||_1,$$

where I is the identity matrix, C is the constant matrix in which each row is a copy of π^T, π^T is the unique steady state distribution, and for any $i \in V$, $\boldsymbol{b_i}^T$ is the i-th basis vector, that is, the row vector of all zeros except for a 1 in the i-th position, and $P = \{p_{ij}\}_{i,j=0}^{n}$ is the n-dimensional one-step transition probability matrix where the (i, j)th entry is given by

$$p_{ij} = \begin{cases} \dfrac{1}{d_i} & \text{if } (v_i, v_j) \in E \\ 0 & \text{otherwise} \end{cases},$$

where d_i is the degree of node v_i. In this work, we use the converged DSD values as the original DSD calculation for comparison.

1.2 New directions

In the first DSD paper, we modified four classical function prediction methods (including Neighborhood Majority Vote (Schwikowski *et al*., 2000), χ^2 Neighborhood (Hishigaki *et al*., 2001), Multi-way Cut (Vazquez *et al*., 2003) and Functional Flow (Nabieva *et al*., 2005)) to use this dissimilarity metric rather than next-hop shortest-path distance as a dissimilarity metric, and showed that performance improved across the board. Now we extend the calculation of DSD to incorporate confidence, then confidence and directed and undirected pathway edges, then confidence, pathway edges and full biological pathways. We present three new dissimilarity measures, which we call cDSD, caDSD or capDSD, respectively, where capDSD stands for *confidence, augmented pathway diffusion state distance*. These measures can be substituted for original DSD in the four classical function prediction methods we studied (or in any functional prediction method that incorporates a pairwise dissimilarity measure between nodes).

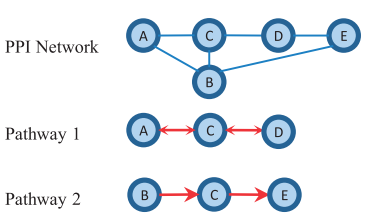
First, to define cDSD, similar to the approach suggested by Gitter *et al*. (2011), we assign a confidence to each PPI interaction edge in BioGRID (Stark *et al*., 2006), based on the number of publications in which that PPI appears, and whether the reported experiments are high-throughput or low-throughput. Given the formal definition of DSD, there is a natural way to incorporate these confidences simply as edge weights, and the k-step DSD calculation is generalized to a weighted matrix in the natural way (see Section 2.1.3 for full details). We show that incorporating confidence values in this way improves performance over the basic DSD method (which in turn improved the performance compared to the corresponding method based on shortest-path distances (Cao *et al*., 2013)) in cross-validation on each of the classical network-based function prediction methods we consider.

On top of the confidence values, we then seek to augment the network by adding edges from the KEGG PATHWAY database in two ways. We find that 2471 of these edges are not already in BioGRID, and an additional 177 are in BioGRID, but we would have assigned them lower confidence without the additional information that they also appeared in KEGG, so it is not surprising that adding in these edges improves our results as compared to DSD and cDSD. In the first and simplest way, which we call caDSD, we augment the graph by adding undirected and directed edges from the KEGG database; where edges of the types: activation, inhibition, phosphorylation, dephosphorylation and ubiquination are considered naturally directed as in (Liu *et al*., 2009) and all other KEGG edges are considered undirected (however, an undirected edge being included in the KEGG database raises its edge weight because KEGG is manually curated). However, we also create capDSD which creates an augmented graph that represents the signaling pathways coherently using new sets of nodes and edges. In this new augmented graph, pathways can be thought of as being represented by 'controlled-access highways', in the sense that once the diffusion random walk enters a pathway, it stays on that pathway with some fixed probability r and only leaves that pathway to walk in the regular PPI network (still augmented with directed edges, where known, and confidence) with probability $1-r$, where the fixed r is a parameter of the method. Just like DSD, capDSD is not a

function prediction method in itself, it is a dissimilarity matrix: for each pair of nodes, capDSD gives a value that measures their similarity in this (now augmented, confidence weighted) network. For the best performing function prediction methods we test, we find that adding in the KEGG pathway edges using the highway approach is superior to just adding in the KEGG edges naively. Furthermore, the performance increase is even stronger when using an integrative network derived from the STRING database (see Section 2.1.2).

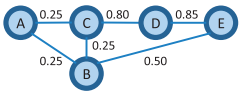
Figure 1 shows an example of the modifications to the network involved in computing cDSD, caDSD and finally capDSD. Of the four different classical methods we test with all of DSD, cDSD, caDSD and capDSD, we find that our best function prediction method, over all three levels of the MIPS hierarchy is the one that predicts v's label based on the t closest neighbors in terms of their values in the capDSD matrix, and has them vote on the functional label of v, with a vote weight inversely proportional to their capDSD value, assigning v the function with the highest weighted vote. Significantly, the improvement is greater at the lower (more specific) levels of the MIPS hierarchy.

(a) The undirected graph G_0 from the original PPI network and two KEGG pathways (on the left) as well as the table for the number of publications that verify each PPI in the original PPI network (on the right);

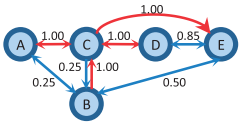


(A, C)	1 publication with high-throughput techniques
(A, B)	1 publication with high-throughput techniques
(B, C)	1 publication with high-throughput techniques
(C, D)	1 publication with low-throughput techniques
(D, E)	1 publication with low-throughput technique and 1 publication with high-throughput technique
(B, E)	2 publications with high-throughput techniques

(b) The weighted graph G_{conf} by assigning PPI confidence scores as weights on edges;



(c) The directed graph G_{aug} by simply adding KEGG PPIs (note that one edge from node C to node E is added with weight 1.00 and weights for edge (A, C), (C, D) and <B, C> are changed);



(d) The augmented graph G_{path} based on the original PPI network and KEGG pathways (note that the weight will be set dependent on parameters r and m).

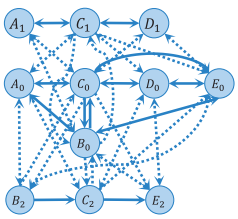


Fig. 1. An example of constructing auxiliary graphs for calculating different DSDs (with our BioGRID confidence scores). **(a)** The original PPI network and two KEGG pathways; **(b)** the weight graph with PPI confidence score as edge weights; **(c)** the directed graph with KEGG PPIs added; and **(d)** the augmented graph by incorporating KEGG pathways as weighted paths

2 MATERIALS AND METHODS

2.1 Datasets

2.1.1 Physical protein interaction network from BioGRID The *S.cerevisiae* protein–protein physical interaction network is constructed as follows: the list of 5064 verified ORFs downloaded from the SGD website (Saccharomyces Genome Database, version date October 25, 2013) defines the nodes, and the 133 705 protein–protein physical interactions from BioGRID (Stark *et al.*, 2006) between nodes that are verified by at least one wet-lab experiment define the edges. After removing edge redundancy, self-loops, and edges incident to unverified ORF nodes, we extract the largest connected component and obtain a simple undirected graph with $n = 5001$ nodes and $m = 76\,025$ unique undirected edges; we denote by $G_0(V_0, E_0, W_0)$ this simple undirected graph with unit-weight for all edges, where $V_0 = \{v_1, v_2, \ldots, v_n\}$ and W_0, the weight matrix, is the n-dimensional square matrix with value 1 for entry (i, j) if and only if (v_i, v_j) is in E_0, and 0 otherwise.

2.1.2 Protein association network from STRING STRING (Franceschini *et al.*, 2013) is a database that integrates known and predicted protein associations from various sources, such as BioGRID (Stark *et al.*, 2006), BIND (Bader *et al.*, 2003), DIP (Xenarios *et al.*, 2002), MINT (Licata *et al.*, 2012), KEGG PATHWAY (Kanehisa and Goto 2000) and gene co-expression data (Franceschini *et al.*, 2013). STRING assigns normalized confidence scores to many different types of protein associations: some from experiments (physical and genetic protein interactions), or derived from co-expression, and others either inferred by literature annotation or transferred from homology. Because including edges inferred by literature annotation could invalidate the separation of training and testing in our cross-validation experiments, we could not use all the association categories in STRING. We extract all protein associations from the 'experiments' and 'co-expression' categories for yeast (with confidence score > 0 for at least one of the two categories), where 'experiments' covers all physical and genetic protein interactions and 'co-expression' refers to protein associations that are inferred from similar transcriptional patterns in terms of gene co-expression levels. We also want to include KEGG PATHWAY PPIs that have already been incorporated in STRING; however, such information is mixed with and cannot be separated from other data sources in the 'database' category, including GO, which we do not want to include so as to avoid possible overlapping between test data and training data in our function prediction evaluation framework. Therefore, we directly extract association links for pathway neighbors and subunits of the same enzyme/complex from the KEGG PATHWAY database, the same fashion as what STRING utilizes. We extract 454 600 protein–protein associations (being sure to exclude homology-based transferred interologs) from STRING version 9.05, release date: March 3, 2013 (Note that there is also a more recent December 27, 2013 version 9.1 of STRING now available, but it has no simple way to exclude interologs, so we used the previous version.) We also include edges directly from KEGG (all but 249 of these also appear in the portion of the STRING database we use for our network; the discrepancy of 249 additional edges comes from the fact that we use the December 2013 version of KEGG while STRING version 9.05 uses the August 2012 version of KEGG). We further filter the network by removing associations that are incident with at least one unverified ORF from SGD. Afterward we compile the undirected graph where a node corresponds to an ORF and an undirected edge is added if there exists an association link between the two ORFs (we did not add directed edges for the STRING experiment, since they were shown to

matter so little on the BioGRID experiment, see Table 3). The resulting graph G_{str} is undirected, connected, has diameter 5, and contains 5058 nodes and 404 358 edges.

2.1.3 PPI confidence assignment Because there is no confidence score provided by BioGRID, we create confidence weights for BioGRID PPI edges in G_0 using a scoring scheme similar to previous work by (Gitter *et al*., 2011), according to the following premises:

- Low-throughput experiments, due to their lower false positive rate, are considered to provide more reliable PPIs than high-throughput experiments.
- If a PPI is verified experimentally by more experiments from curated publications, we hold higher confidence in the existence of the PPI.

There are more than 7000 publications associated with the physical interaction PPI data we collect from BioGRID, making a manual assignment of whether the experiment supporting the PPI is high- or low-throughput highly impractical. Instead, we automatically and efficiently determine a close proxy for this information by simply counting the *number* of different PPIs that a particular publication vouches for in BioGRID. If there are at least 100 PPIs associated with a particular publication, we classify that publication's endorsements as *high-throughput* and otherwise *low-throughput*. In total, 7112 publications are classified as low-throughput and 97 publications are classified as high-throughput. Note that these 97 high-throughput publications actually generate more than two-thirds of the physical interactions. (We tried other cutoff values for distinguishing low-throughput/high-throughput and the results were similar; in fact, very few publications lie close to the 100 threshold; most low-throughput publications have substantially less, and most high-throughput publications have substantially more.) If an interaction edge is endorsed by only experiments of one type (either high- or low-throughput) we assign confidence weights according to Table 1. If an interaction edge is endorsed by both high confidence and low confidence experiments, we use the confidence score from the low-throughput column in Table 1 plus 5% times the number of high-throughput endorsements; however, if this value exceeds 95%, we still assign a maximum confidence score of 95%.

For all pairs of nodes in G_0, we can assign the confidence score as their weight. We denote by $W_{conf} = \{w_{ij}\}_{i,j=1}^{n}$ the weight matrix, where w_{ij} is the confidence score for the node pair (v_i, v_j) (also denoted as $w_{vi,vj}$ when confusion does not exist). Note that $w_{ij} = 0, \forall (v_i, v_j) \notin E_0$ and $w_{ij} > 0, \forall (v_i, v_j) \in E_0$. We denote by $G_{conf}(V_{conf}, E_{conf}, W_{conf})$ this simple undirected graph where $V_{conf} = V_0, E_{conf} = E_0$ and W_{conf} is defined above.

For the edge weights in G_{str} we simply take the confidence scores p_1, p_2 and p_3 from STRING for each selected category: 'experiments', 'co-expression' and 'database' (Note that we assign 0.9 for 'database' confidence score if the association link is in the KEGG PATHWAY PPIs, and 0 otherwise; the choice of 0.9 for KEGG PPIs is similar to STRING's.); then we calculate the combined confidence score as $p = 1 - (1 - p_1) * (1 - p_2) * (1 - p_3)$ in the Bayesian scheme, which is exactly how STRING (Franceschini *et al*., 2013) suggests individual confidence scores be combined.

2.1.4 Functional pathway maps We use all 105 *S.cerevisiae* signaling pathways from the KEGG PATHWAY database (Kanehisa and Goto, 2000) (version date: December 12, 2013) where there are 75 pathways from the metabolism category, 21 from the genetic information processing category, 3 from the environmental information processing category and 6 from the cellular processes category. Just as suggested in (Liu *et al*., 2009), in the BioGRID experiments, we run both caDSD with all edges undirected, and we also run the version of caDSD where we additionally consider the following five protein relations that appear in the KEGG database as directional: activation, inhibition, phosphorylation, dephosphorylation and ubiquination. Any PPIs extracted with only one of these five types are considered directed, while all the other PPIs annotated with types such as 'compound' are considered undirected. In total, there are 206 directed PPIs and 6951 undirected PPIs separately, involving 1120 proteins in the KEGG PATHWAY database; since we only consider edges of which both endpoints appear in the connected PPI network G_0, we extract 157 directed PPIs, the set of which is denoted by D, and 3374 undirected PPIs, the set of which is denoted by U, involving 1083 unique ORFs total. Because the results for the caDSD adding so few directed edges were very similar to the fully undirected version of caDSD, we do not add directions to the edges in the STRING experiment.

Table 1. Confidence score assignment for PPIs when either only low-throughput or only high-throughput experiments are present

No. of experiments	Low-throughput	High-throughput
0	0	0
1	0.80	0.25
2	0.90	0.50
3	0.95	0.75
≥4	0.95	0.85

2.1.5 Functional annotation We consider both the MIPS functional catalogue (FunCat) (Ruepp *et al*., 2004) and GO annotations (Ashburner *et al*., 2000). We use the latest version of FunCat (version 2.1) and the first, second and third level functional categories, retaining only those labels annotating at least three proteins in our dataset. We present results for MIPS annotations at the first level (4443 proteins with 10 569 annotations in 17 functional categories in BioGRID), second level (4428 proteins with 12 378 annotations in 74 out of 80 functional categories annotating at least 3 proteins in BioGRID) and third level (4061 proteins with 9441 annotations in 154 out of 181 functional categories annotating at least 3 proteins in BioGRID). We also present results for the popular GO (Ashburner *et al*., 2000), where the variable depth hierarchy of the annotation labels makes the evaluation of predicted labels more complicated, in the Supplementary Material.

2.2 cDSD, caDSD and capDSD

2.2.1 cDSD: incorporating PPI confidence We build the undirected weighted simple graph $G_{conf}(V_{conf}, E_{conf}, W_{conf})$ where $V_{conf} = V_0$ and $E_{conf} = E_0$ are simply defined by assigning the confidence score to all pairs of nodes in V_0. The confidence scores are assigned as described in Section 2.1.3. Let $P' = \{p'_{ij}\}_{i,j=0}^{n}$ be the n-dimensional one-step transition matrix where the (i, j)th entry is given by

$$p'_{ij} = \begin{cases} \frac{w_{ij}}{\sum_{l=1}^{n} w_{il}} & \text{if } (v_i, v_j) \in E_{conf} \\ 0 & \text{otherwise} \end{cases}.$$

Note that P' represents the probability to reach each neighbor in the random walk. Then the definition of k-step transition probability matrix $P'\{k\} = P'k$ follows for all positive k. It is easy to show that the expected number of times that a random walk starting at node v_i and proceeding for k steps will visit node v_j, denoted as $He'^{\{k\}}(v_i, v_j)$, can be calculated as $\sum_{l=0}^{k} p'^{\{l\}}_{ij}$, where $p'^{\{l\}}_{ij}$ is the (i,j)th entry of l-step transition probability matrix. The n-dimensional vector $He'^{\{k\}}(v_i), \forall v_i \in V_{conf}$ can be constructed accordingly. Therefore, when we fix the number of random walk steps k, the definition of DSD with PPI confidence follows:

$$cDSD^{\{k\}}(u, v) = ||He'^{\{k\}}(u) - He'^{\{k\}}(v)||_1.$$

2.2.2 caDSD: adding KEGG PPIs We consider PPIs from KEGG PATHWAY database highly reliable since they are manually drawn by domain experts; for the BioGRID experiments, we will re-assign maximum confidence score 1 to these PPIs no matter whether or not the PPI is present in the BioGRID database (For the STRING experiments, note that every KEGG edge is already assigned a confidence value of at least 0.9 by cDSD (and maybe larger if there is additional independent evidence) so we just retain cDSD confidence values on these edges).

Thus, based on the undirected graph $G_{\text{conf}}(V_{\text{conf}}, E_{\text{conf}}, W_{\text{conf}})$, the undirected edge set U and the directed edge set D from KEGG pathways, we build a directed graph $G_{\text{aug}}(V_{\text{aug}}, E_{\text{aug}}, W_{\text{aug}})$, where $V_{\text{aug}} = V_0$, E_{aug} and $W_{\text{aug}} = \{w_{ij}^{\{\text{aug}\}}\}_{i,j=0}^{n}$ are constructed as follows (we use $\langle\rangle$ to denote directed edges compared to () for undirected edges):

(1) Initialize E_{aug} by adding $\langle v_i, v_j\rangle$ and $\langle v_j, v_i\rangle$ with weight $w_{ij}^{\{\text{aug}\}} = w_{ji}^{\{\text{aug}\}} = w_{ij}^{\{\text{conf}\}}, \forall (v_i, v_j) \in E_{\text{conf}}$;

(2) For each edge $(v_i, v_j) \in U$, if (v_i, v_j) already exists in E_{conf}, set $w_{ij}^{\{\text{aug}\}} = w_{ji}^{\{\text{aug}\}} = 1$, otherwise add $\langle v_i, v_j\rangle$ and $\langle v_j, v_i\rangle$ into E_{aug} with weight 1; and

(3) For each edge $\langle v_i, v_j\rangle \in D$, if (v_i, v_j) already exists in E_{conf}, set $w_{ij}^{\{\text{aug}\}} = 1$, otherwise add $\langle v_i, v_j\rangle$ into E_{aug} with weight 1.

Again, we define the one-step transition probability matrix $P_{\text{aug}} = \{p_{ij}^{\{\text{aug}\}}\}_{i,j=0}^{n}$ as follows:

$$p_{ij}^{\{\text{aug}\}} = \begin{cases} w_{ij}^{\{\text{aug}\}} / \sum_{l=1}^{n} w_{il}^{\{\text{aug}\}} & \text{if } \langle v_i, v_j\rangle \in E_{\text{aug}}; \\ 0 & \text{otherwise.} \end{cases}$$

Similarly we define the k-step transition probability matrix $P_{\text{aug}}^{\{k\}} = P_{\text{aug}}^{k}$ and calculate the expected number of times that a random walk starting at node v_i and proceeding for k steps will visit node v_j, $He_{\text{aug}}^{\{k\}}(v_i, v_j) = \sum_{l=0}^{k} p_{ij}^{\{\text{aug},l\}}$, where $p_{ij}^{\{\text{aug},l\}}$ is the (i, j)th entry of the l-step transition probability matrix $P_{\text{aug}}^{\{l\}}$. Thus the n-dimensional vector $He_{\text{aug}}^{\{k\}}(v_i), \forall v_i \in V_{\text{aug}}$ follows similarly and when we fix the number of random walk steps k, the definition of DSD with KEGG PPIs is

$$\text{caDSD}^{\{k\}}(u, v) = ||He_{\text{aug}}^{\{k\}}(u) - He_{\text{aug}}^{\{k\}}(v)||_1.$$

2.2.3 capDSD: the augmented graph with explicit pathways The previous caDSD makes use of the fact that the PPIs from the KEGG PATHWAY database are high-quality, and sometimes known to be directional; however it incorporates the KEGG pathway information as individual interaction edges and retains no notion of each pathway as a cohesive whole. In particular, some graph paths may not be meaningful at all when mapped to a chain of ORFs, while other graph paths correspond to signaling pathways. We hypothesize that if we can make the random walks used to calculate DSD values hew more tightly to the known pathways, the resulting diffusion process might better capture the notion of functional similarity. However, doing so directly would destroy the 'memoryless' structure of the underlying random walk, and make the probabilities too difficult to calculate. Our solution is to instead build a new network, where nodes in pathways are replicated, into ordinary and 'highway' versions, where the 'highway' version is chosen with some probability, and if the 'highway' is taken, edge probabilities for the highway nodes are set so that it is highly likely to continue along the pathway. More specifically, we build a network $G_{\text{path}}(V_{\text{path}}, E_{\text{path}}, W_{\text{path}})$ where W_{path} will be a mapping: $W_{\text{path}} : V_{\text{path}} \times V_{\text{path}} \to \mathbb{R}, \forall a, b \in V_{\text{path}}$ (instead of an n-dimensional square matrix because the size of V_{path} will be different from n) as follows:

(1) Denote by $\{P_1, P_2, \ldots, P_g\}$ where g is the number of pathways, the set of pathways; denote by $PE_1, PE_2, \ldots, PE_g$ the sets of directed edges from the g pathways where each undirected edge is considered as two directed edges; denote by $PV_1, PV_2, \ldots, PV_g$ the sets of proteins involved in each of the g pathways where each set is a subset of V_{aug}, namely the ORF list;

(2) We initialize V_{path} with $\{v_1^0, v_2^0, \ldots v_n^0\}$ by relabeling each ORF node $v_i \in V_{\text{aug}}$ with a superscript 0, which stands for the original PPI network; we initialize W_{path} as the empty map;

(3) We initialize E_{path} by adding $\langle v_i^0, v_j^0\rangle$ with weight $W_{\text{path}}(v_i^0, v_j^0) = w_{v_i, v_j}^{\{\text{aug}\}}$ for all $\langle v_i, v_j\rangle \in E_{\text{aug}}$;

(4) For each pathway $P_\alpha \in \{P_1, P_2, \ldots, P_g\}$:

(a) For each protein $v_i \in PV_\alpha$, add a pathway node v_i^α into V_{path};

(b) For each pathway node $v_i^\alpha \in V_{\text{path}}$: for each edge $\langle v_i, v_j\rangle \in E_{\text{aug}}$, we add an edge $\langle v_i^\alpha, v_j^0\rangle$ into E_{path} with weight $w_{ij}^{\{\text{aug}\}}$; and for each edge $\langle v_j, v_i\rangle \in E_{\text{aug}}$, we add an edge $\langle v_j^0, v_i^\alpha\rangle$ into E_{path} with weight $w_{ji}^{\{\text{aug}\}}$; these newly added edges are called cross edges; and

(c) For each edge $\langle v_i, v_j\rangle \in PE_\alpha$ which we call a pathway edge, add an edge $\langle v_i^\alpha, v_j^\alpha\rangle$ into E_{path}, and the weight assignment will not be set but the transition probability will be assigned specially in Step 7 when all the pathways are processed.

(5) For each cross edge in the form of $\langle v_i^0, v_j^\alpha\rangle \in E_{\text{path}}$, $\forall i \in \{1, 2, \ldots, n\}$, $v_j \in PV_\alpha$, $\alpha \in \{1, 2, \ldots, g\}$, boost the weight by multiplying $W_{\text{path}}(v_i^0, v_j^\alpha)$ by the factor of m and update the weight with the boosted value, where m is a multiplication factor parameter;

(6) For all the directed node pairs $\langle v_i^\alpha, v_j^\beta\rangle \notin E_{\text{path}}, \forall \alpha, \beta \in \{0, 1, \ldots, g\}$, $v_i^\alpha, v_j^\beta \in V_{\text{path}}$, assign 0 as the weight since we do not have any evidence for the existence of the PPI pair $\langle v_i, v_j\rangle$;

(7) Let $N = |V_{\text{path}}|$, where $N = n + \sum_{\alpha=1}^{g} |PV_\alpha|$. Now we calculate the N-dimensional one-step transition probability square matrix P_{path} where we denote by p_{i_α, j_β} as the one-step transition probability from v_i^α to $v_j^\beta, \forall v_i^\alpha, v_j^\beta \in V_{\text{path}}$:

(a) For each pathway node $v_i^\alpha \in V_{\text{path}}$, where $\alpha > 0$, the pathway edge $\langle v_i^\alpha, v_j^\alpha\rangle \in E_{\text{path}}$, will have transition probability set as $p_{i_\alpha, j_\alpha} = r/d_i^\alpha$, where $r \in (0, 1)$ is a parameter and d_i^α is the number of pathway edges starting from v_i^α; the cross edge $\langle v_i^\alpha, v_j^0\rangle \in E_{\text{path}}$ will have transition probability set as $p_{i_\alpha, j_0} = (1 - r) \cdot W_{\text{path}}(v_i^\alpha, v_j^0) / \sum_{\langle v_i^\alpha, v_l^0\rangle \in E_{\text{path}}} W_{\text{path}}(v_i^\alpha, v_l^0)$ if $d_i^\alpha > 0$, and $p_{i_\alpha, j_0} = W_{\text{path}}(v_i^\alpha, v_j^0) / \sum_{\langle v_i^\alpha, v_l^0\rangle \in E_{\text{path}}} W_{\text{path}}(v_i^\alpha, v_l^0)$ otherwise (no edges across two pathway nodes from two different pathways exist); and

(b) For each node $v_i^0 \in V_{\text{path}}$, the transition probability will be set as $p_{i_0, j_\alpha} = W_{\text{path}}(v_i^0, v_j^\alpha) / \sum_{\langle v_i^0, v_l^\beta\rangle \in E_{\text{path}}} W_{\text{path}}(v_i^0, v_l^\beta)$, if $\langle v_i^0, v_j^\alpha\rangle \in E_{\text{path}}$, and 0 otherwise.

Step 5 is used so that the probability of entering pathways can be adjusted higher by setting the multiplication factor $m > 1$; in the Results section, we report the results where $m = 25$. Step 7(a) is used so that the total probability of staying on the same pathway after one transition from a non-terminal pathway node (the node that has outgoing pathway edges) will be r, which in our case we set as $r = 0.7$. We tried different values for r and m empirically; and results are fairly robust to different choices of r and m (results of weighted majority voting capDSD over different choices of r and m appear in the Supplementary Material). Given the one-step transition probability matrix P_{path} as well as the l-step transition probability matrix $P_{\text{path}}^{\{l\}} = P_{\text{path}}^{l}, \forall l \geq 0$, we can calculate the expected number of times that a random walk starting at node v_i^α and proceeding for k steps will visit node v_j^β, $\text{EXP}^{\{k\}}(v_i^\alpha, v_j^\beta) = \sum_{l=0}^{k} p_{i_\alpha, j_\beta}$. Then we define the *He* value for each

pair of ORF nodes $v_i, v_j \in V_0$:

$$He^{\{k\}}_{\text{path}}(v_i, v_j) = \sum_{\alpha: v_j^\alpha \in V_{\text{path}}} \text{EXP}^{\{k\}}(v_i^0, v_j^\alpha),$$

as well as the n-dimensional vector:

$$He^{\{k\}}_{\text{path}}(v_i) = (He^{\{k\}}_{\text{path}}(v_i, v_1), He^{\{k\}}_{\text{path}}(v_i, v_2), ..., He^{\{k\}}_{\text{path}}(v_i, v_n)).$$

The definition of DSD with external paths follows:

$$\text{capDSD}^{\{k\}}(v_i, v_j) = ||He^{\{k\}}_{\text{path}}(v_i) - He^{\{k\}}_{\text{path}}(v_j)||_1, \forall v_i, v_j \in V_0.$$

2.3 Evaluation

As shown in (5), the original DSD improves all the tested classical protein function prediction algorithms in 2-fold cross-validation for functional label prediction for all three levels of the MIPS hierarchy by simply replacing the shortest-path distance with the DSD matrix, where the best performing method overall was the DSD version of weighted majority vote. In this work, we similarly evaluate four methods (majority vote, weighted majority vote, multi-way cut and functional flow) using cDSD, caDSD and capDSD as the distance metric. While the results in (Cao *et al.*, 2013) were based on the converged DSD as $k \to \infty$, we have not yet been able to prove convergence for our new cDSd, caDSD and capDSD variants. Thus, in our experiments, we set the length of random walk step $k = 7$ for all the three variants of DSDs (we also tested other values of k and empirically observed that when $k \geq 5$, the performance is almost unchanged even though we have not been able to prove the convergence of the variants of DSDs.)

We stress that in each of our experiments, the function prediction method is unchanged, and does not explicitly incorporate confidence or pathway information in any way, except in that it uses the values from the cDSD, caDSD or capDSD matrix instead of from the DSD (or ordinary shortest-path distance) matrix.

2.3.1 Cross-validation task We consider 2-fold cross-validation tasks. In each of the 2-fold cross-validation tasks, we first randomly split the annotated proteins into two sets. For each set, we use its annotations as the training set to predict the annotations on proteins in the other set. We then average the performance over the 2-folds of the cross-validation. We conduct 10 runs of 2-fold cross-validation. For MIPS function prediction we report the means and standard deviations of the two performance measures over these 10 runs: accuracy and F1 score (Cao *et al.*, 2013). The accuracy is calculated as the percentage of proteins that are assigned a correct function annotation (Schwikowski *et al.*, 2000). The F1 score for each protein function is calculated as (Darnell *et al.*, 2007)

$$F1 = \frac{2 * \text{precision} * \text{recall}}{\text{precision} + \text{recall}},$$

where precision and recall are calculated by looking at the top α (in our case, we present results for $\alpha = 3$) predicted annotations. We average F1 scores over the individual functions and obtain the overall F1 score for each algorithm. Our GO (Ashburner *et al.*, 2000) results take into account partial matches based on the deep hierarchy of the GO labels according to the methods of (Deng *et al.*, 2003, 2004) and appear in the Supplementary Material.

2.3.2 Neighborhood majority voting algorithm: weighted and unweighted These are the simplest of all function prediction methods. Directly applying the concept of 'guilt by association', (Schwikowski *et al.*, 2000) consider for each protein u its neighboring proteins. Each neighbor votes for their own annotations, and the majority is used as the predicted functional label. To incorporate DSD, the neighborhood of u is defined simply as the t nearest neighbors of u under the DSD metric. Furthermore, two schemes are considered: an unweighted scheme where all new neighbors vote equally, and a DSD weighted scheme where all new neighbors get a vote proportional to the reciprocal of their DSD distance. As in (Cao *et al.*, 2013), we set $t = 10$.

Multi-way cut algorithm Similar to (Nabieva *et al.*, 2005), we implement the minimal multi-way k-cut algorithm of (Vazquez *et al.*, 2003) whose motivation is to minimize the number of times that annotations associated with neighboring proteins differ, by approximately solving the integer linear programming problem:

$$\text{maximize} \sum_{(u,v)\in E, a\in \text{FUNC}} X_{u,v,a}$$

subject to the constraints $\sum_{a\in\text{FUNC}} X_{u,a} = 1$, $X_{u,v,a} \leq X_{u,a}$, $X_{u,v,a} \in \{0, 1\}$, $X_{v,a} \in \{0, 1\}$ where the edge variables $X_{u,v,a}$ are defined for each function a in the function set FUNC, whenever there exists an edge between proteins u and v in the edge set E. $X_{u,v,a}$ is set to 1, if protein u and v both are assigned function a, and 0 otherwise. The node variable $X_{u,a}$ are set to 1 when u is labeled with function a and 0 otherwise. The first constraint insures that each protein is only given one annotation. The second constraint makes sure only annotations that appear among the vertices can be assigned to the edges. While this problem is NP-hard, the ILP is tractable in practice; in our case we use the IBM CPLEX solver (version 12.4, http://www.ilog.com/products/cplex/). For the DSD version of this algorithm, we simply add additional edges between vertices whose DSD is below a threshold . We set a global threshold D based on the average DSD of all pairs, specifically we set $D = \mu - c * \sigma$, where μ is the average, and σ is the standard deviation of the global set of DSD values among all pairs of nodes in the graph. As in (Cao *et al.*, 2013), we set $c = 1.5$.

Functional flow algorithm Nabieva *et al.* (2005) use a network flow algorithm on the graph of protein interactions to label proteins. The idea is to consider each protein having a known function annotation as a 'reservoir' of that function, and to simulate flow of functional association through the network to make predictions. We adapt the approach to use DSD by creating an edge between each node pair, with a weight inversely proportional to DSD. For computational efficiency we do not create edges when the reciprocal of DSD is below a small value. This global threshold for DSD values is set the same as in the multi-way cut algorithm. As in the original functional flow, we calculate flow through this new network at each time step. We denote the size of the reservoir of function a at node u and time step i, to be $R^a_i(u)$. For a given function (annotation) a we initialize the reservoir size at node u to be infinite if protein u has been annotated with function a; otherwise we set it to be 0. More formally: $R^a_0(u) = \infty$ if u is annotated with a and 0 otherwise. We then update the reservoir over a sequence of time steps (we use six time steps, as in the original version (Nabieva *et al.*, 2005)):

$$R^a_t(u) = R^a_{t-1}(u) + \sum_{v:(u,v)\in E} (g^a_t(v, u) - g^a_t(u, v)),$$

where $g^a_t(v, u)$ is the amount of flow a that moves from u to v at time t. We incorporate DSD into the edge weight as follows:

$$g^a_t(u, v) = \begin{cases} 0, & \text{if } R^a_{t-1}(u) < R^a_{t-1}(v) \\ \min(\frac{1}{\text{DSD}(u, v)}, \text{flow}_{u,v}) & \text{otherwise.} \end{cases},$$

where $\text{flow}_{u,v} = \frac{\frac{1}{\text{DSD}(u,v)}}{\sum_{(u,y)\in E} \frac{1}{\text{DSD}(u,y)}}$. The final functional score for node u and function a is computed as the total amount of incoming flow.

Table 2. Summary of protein MIPS function prediction performance for the physical PPI network using DSD, cDSD, caDSD and capDSD compared to the original methods in 10 runs of 2-fold cross-validation (as a percentage)

	MIPS 1		MIPS 2		MIPS 3	
	Accuracy	F1 score	Accuracy	F1 score	Accuracy	F1
Majority Vote (MV)	50.08 ± 0.72	41.45 ± 0.40	40.69 ± 0.49	30.85 ± 0.33	38.03 ± 0.37	29.50 ± 0.14
MV with original DSD	62.96 ± 0.45	47.40 ± 0.28	49.41 ± 0.65	35.71 ± 0.33	43.87 ± 0.47	32.33 ± 0.18
MV with cDSD	66.16 ± 0.56	49.10 ± 0.24	53.08 ± 0.54	38.12 ± 0.16	47.73 ± 0.56	35.13 ± 0.33
MV with caDSD (directed edges)	67.61 ± 0.56	50.37 ± 0.22	59.11 ± 0.67	41.58 ± 0.19	52.14 ± 0.55	38.09 ± 0.16
MV with caDSD (no directed edges)	67.61 ± 0.42	50.36 ± 0.24	59.11 ± 0.57	41.57 ± 0.25	52.13 ± 0.56	38.07 ± 0.21
MV with capDSD	67.60 ± 0.37	50.28 ± 0.27	59.46 ± 0.57	41.58 ± 0.22	52.97 ± 0.59	38.19±0.23
Weighted MV (WMV) with original DSD	63.40 ± 0.51	48.29 ± 0.25	50.69 ± 0.82	36.74 ± 0.36	45.20 ± 0.58	33.72 ± 0.27
WMV with cDSD	67.07 ± 0.45	50.12 ± 0.35	54.82 ± 0.56	39.53 ± 0.18	49.56 ± 0.49	36.71 ± 0.32
WMV with caDSD (directed edges)	68.69 ± 0.40	51.48 ± 0.29	60.96 ± 0.51	43.13 ± 0.23	54.51 ± 0.51	39.91 ± 0.28
WMV with caDSD (no directed edges)	68.68 ± 0.41	51.48 ± 0.25	60.96 ± 0.53	43.13 ± 0.22	54.51 ± 0.46	39.90 ± 0.32
WMV with capDSD	**68.90 ± 0.49**	**51.61 ± 0.21**	**61.82 ± 0.59**	**43.54 ± 0.26**	**56.16 ± 0.59**	**40.42 ± 0.35**
Multi-way Cut (GMC)	55.31 ± 0.41	42.18 ± 0.29	42.02 ± 0.43	28.21 ± 0.36	36.69 ± 0.50	24.98 ± 0.21
GMC with original DSD	58.36 ± 0.32	42.51 ± 0.19	44.63 ± 0.32	29.51 ± 0.27	38.20 ± 0.40	25.49 ± 0.22
GMC with cDSD	61.11 ± 0.37	42.85 ± 0.23	47.11 ± 0.35	30.52 ± 0.25	40.83 ± 0.61	26.66 ± 0.22
GMC with caDSD (directed edges)	62.71 ± 0.30	43.46 ± 0.24	52.59 ± 0.25	32.47 ± 0.30	44.29 ± 0.63	28.46 ± 0.19
GMC with caDSD (no directed edges)	62.76 ± 0.31	43.45 ± 0.25	52.61 ± 0.25	32.50 ± 0.30	44.31 ± 0.63	28.46 ± 0.19
GMC with capDSD	62.44 ± 0.31	43.43 ± 0.17	52.30 ± 0.46	32.48 ± 0.31	44.18 ± 0.59	28.34 ± 0.32
Functional Flow (FF)	50.48 ± 0.48	37.17 ± 0.25	32.57 ± 0.48	22.64 ± 0.32	25.29 ± 0.39	18.27 ± 0.14
FF with original DSD	53.58 ± 0.36	40.75 ± 0.11	38.20 ± 0.65	26.71 ± 0.29	30.70 ± 0.45	22.29 ± 0.28
FF with cDSD	57.78 ± 0.49	42.82 ± 0.27	42.17 ± 0.58	29.29 ± 0.38	35.68 ± 0.48	25.72 ± 0.17
FF with caDSD (directed edges)	60.09 ± 0.55	44.81 ± 0.24	49.73 ± 0.41	33.89 ± 0.32	40.82 ± 0.60	28.94 ± 0.27
FF with caDSD (no directed edges)	60.18 ± 0.47	44.80 ± 0.20	49.67 ± 0.51	33.89 ± 0.28	40.82 ± 0.51	28.97 ± 0.23
FF with capDSD	58.98 ± 0.53	43.80 ± 0.27	49.32 ± 0.61	33.32 ± 0.29	41.04 ± 0.33	28.83 ± 0.33

Note: Weighted majority vote with capDSD (in bold) gives the best results over all three levels of the MIPS hierarchy.

3 RESULTS

3.1 Performance of function prediction methods and their DSD variants on MIPS

Cao *et al*. (2013) show how to modify several classical function prediction methods, including the four we study here (majority vote, weighted majority vote, multi-way cut and functional flow) to utilize the DSD pairwise dissimilarity metric in place of ordinary shortest-path distance. In this work, we use the same DSD-based methods as in Cao *et al*. (2013), but instead substitute the cDSD, caDSD and capDSD matrices to incorporate confidence measures and pathways. Full MIPS results on BioGRID data appear in Table 2, where we have two versions of caDSD: one that adds directions to the 157 edges which are of the five types identified by Gitter *et al*. (2011) as naturally directed, and one where all edges are left undirected. Table 3 then gives the results on the integrative STRING database. Note that for the STRING database, we already include all the KEGG edges, so cDSD is equivalent to (undirected) caDSD, so this merges the two lines in the table. GO results appear in the Supplementary Material.

We observe that, on both BioGRID and STRING, over 10 runs of 2-fold cross-validation, the best method overall is weighted majority vote with capDSD. For example, weighted majority vote with capDSD achieves an average 68.90% accuracy and 51.61% F1 score on the first level of the MIPS hierarchy on BioGRID, and an average 71.30% accuracy and 52.91% F1 score on the first level of the MIPS hierarchy using STRING. Several other observations are interesting. On the BioGRID data, substituting original DSD for the ordinary shortest-paths metric improved all the function prediction methods we tested across the board. On STRING, this was not the case: when additional edges such as co-expression were added in, ordinary DSD (without confidence weights) no longer improved the classical function prediction methods we tested with the exception of functional flow, where there was a large improvement. But functional flow did much worse overall on the STRING database compared to BioGRID. This implies that when adding in additional edges from sources that might be more weakly correlated to functional transfer of annotation, it is crucial to include confidence values. Once we go from unweighted DSD to DSD with confidence, we again see improvements over classical methods. Going from unweighted DSD to cDSD improves everything, but it is even more crucial for STRING than for BioGRID to include a confidence measure.

Now let us consider all the different ways to incorporate high-confidence KEGG edges. In the BioGRID experiments, as remarked above, it is not surprising that caDSD and capDSD, which use these edges perform better than cDSD, since not all these edges appear already in BioGRID. In the STRING experiment, these edges are already present in cDSD, so cDSD = caDSD gives the naive way to put in these edges, whereas capDSD puts them in as augmented pathways. In the BioGRID experiments, we also experimentally tried assigning directions to some of the

Table 3. Summary of protein MIPS function prediction performance for the STRING integrative network G_{str} using DSD, cDSD/caDSD and capDSD compared to the original methods in 10 runs of 2-fold cross-validation (as a percentage)

	MIPS 1		MIPS 2		MIPS 3	
	Accuracy	F1 score	Accuracy	F1 score	Accuracy	F1
Majority Vote (MV)	65.71 ± 0.36	49.50 ± 0.25	53.95 ± 0.47	37.96 ± 0.19	46.17 ± 0.50	33.75 ± 0.33
MV with original DSD	64.93 ± 0.56	48.55 ± 0.42	50.99 ± 0.35	36.10 ± 0.24	44.47 ± 0.35	31.85 ± 0.22
MV with cDSD/caDSD	69.38 ± 0.71	51.54 ± 0.36	58.01 ± 0.50	40.41 ± 0.32	51.48 ± 0.46	36.86 ± 0.32
MV with capDSD	70.25 ± 0.47	52.22 ± 0.39	61.22 ± 0.57	42.52 ± 0.29	55.54 ± 0.44	39.36 ± 0.21
Weighted MV (WMV) with original DSD	65.25 ± 0.45	49.15 ± 0.44	52.19 ± 0.42	37.10 ± 0.29	45.64 ± 0.41	33.00 ± 0.16
WMV with cDSD/caDSD	69.67 ± 0.56	52.20 ± 0.37	59.41 ± 0.42	41.62 ± 0.26	53.21 ± 0.37	38.29 ± 0.28
WMV with capDSD	**71.30 ± 0.44**	**52.97 ± 0.38**	**62.88 ± 0.54**	**43.98 ± 0.39**	**57.84 ± 0.50**	**41.07 ± 0.21**
Multi-way Cut (GMC)	63.48 ± 0.56	43.03 ± 0.20	52.66 ± 0.54	31.67 ± 0.18	43.37 ± 0.60	26.20 ± 0.19
GMC with original DSD	63.29 ± 0.68	42.80 ± 0.23	52.34 ± 0.56	31.60 ± 0.21	43.59 ± 0.33	26.39 ± 0.18
GMC with cDSD/caDSD	65.18 ± 0.38	43.39 ± 0.16	53.59 ± 0.47	31.89 ± 0.18	44.46 ± 0.36	26.50 ± 0.17
GMC with capDSD	65.21 ± 0.46	43.31 ± 0.15	51.09 ± 0.37	30.74 ± 0.20	40.73 ± 0.40	25.49 ± 0.21
Functional Flow (FF)	39.91 ± 0.77	31.61 ± 0.25	22.26 ± 0.53	17.25 ± 0.21	18.48 ± 0.49	14.26 ± 0.09
FF with original DSD	47.44 ± 0.42	36.46 ± 0.18	29.46 ± 0.30	21.06 ± 0.25	23.08 ± 0.21	16.68 ± 0.16
FF with cDSD/caDSD	51.70 ± 0.43	38.57 ± 0.21	34.67 ± 0.27	24.03 ± 0.19	28.32 ± 0.35	19.39 ± 0.20
FF with capDSD	53.00 ± 0.37	39.73 ± 0.19	37.93 ± 0.50	26.56 ± 0.18	31.18 ± 0.36	21.59 ± 0.20

Note: Weighted majority vote with capDSD (in bold) gives the best results over all three levels of the MIPS hierarchy.

KEGG edges as well, as in the method of Gitter *et al*. (2011) (see Methods section). However, we find that directing 157 edges is much too small a number to affect results; as can be seen in Table 2, results are nearly identical to the undirected caDSD. We therefore used only undirected caDSD which is the same as cDSD for the STRING experiments.

So it remains to answer the main question of the article, whether using the augmented pathways as controlled-access highways is a better way to incorporate pathway information than just using individual edges. The best performing method, weighted majority vote, improved things only very slightly (by <1–1.5 pp) for BioGRID, on different levels of the MIPS hierarchy, with more improvement at the lower levels of the hierarchy. However, on STRING, with the presence of more edges that were more weakly correlated to function, the improvement is much greater. In the STRING experiments (Table 3), going to pathways (capDSD) improved weighted majority vote by over 1.5 pp on the first level of the MIPS hierarchy, by over 3 pp on the second level of the MIPS hierarchy and by over 4 pp on the third level of the MIPS hierarchy. Similar improvements are seen for capDSD with unweighted majority vote and functional flow on STRING, though these are not the best performing methods overall, while performance of multi-way cut degrades with augmented pathways. We next discuss why that might be the case.

4 DISCUSSION

Incorporating confidence and pathways into our diffusion-based distance metric DSD, we studied whether it was best to incorporate pathway information as edges or as controlled-access highways in an augmented graph. We showed that the augmented graph improved the best function prediction method we tested, weighted majority vote, especially in our experiments on the STRING database, where there were additional edges whose correlation with function was weaker. The performance of other methods was not as clearly served by the augmented pathways; capDSD improved functional flow in the noisier STRING setting, but not on BioGRID. The performance of multi-way cut degraded across the board. We hypothesize that the methods that will improve using capDSD versus just caDSD are those that use only some sort of information about the local neighborhood of a node to predict its function; here, making pathways 'closer' with highways is helpful, whereas the amount of distortion in augmenting the graph causes too much noise for more global methods such as multi-way cut. Functional flow, has both local and global aspects, so its mixed performance would be consistent with this theory.

Finally, the best modern function prediction methods are all integrative methods, and may do something more sophisticated than adding in data from other high-throughput data sources as edges with different confidences (Sharan *et al*., 2005, 2007; Borgwardt *et al*., 2005; Cozzetto *et al*., 2013; Dutkowski *et al*., 2013). Thus the next step would be to integrate our results into a hybrid method along these lines.

We note that all code for calculating the confidences, for extracting pathway information from KEGG XML files, and for calculating the cDSD, caDSD and capDSD matrices is available from http://dsd.cs.tufts.edu/capdsd.

ACKNOWLEDGEMENTS

Thanks to the CRA-W DREU program which supported K.J.D. to spend the summer doing research with L.J.C. at Tufts. Thanks to Mark Crovella, Donna Slonim and the entire Tufts BCB group for helpful feedback.

Funding: J.P. was partially supported by NIH grant R01 HD076140 (to D. K. S.).

Conflict of interest: none declared.

REFERENCES

Arnau,V. *et al.* (2005) Iterative cluster analysis of protein interaction data. *Bioinformatics*, **21**, 364–378.

Ashburner,M. *et al.* (2000) Gene Ontology: tool for the unification of biology. *Nat. Genet.*, **25**, 25–29.

Bader,G.D. *et al.* (2003) BIND: the biomolecular interaction network database. *Nucleic Acids Res.*, **31**, 248–250.

Borgwardt,K.M. *et al.* (2005) Protein function prediction via graph kernels. *Bioinformatics*, **21** (**Suppl. 1**), i47–i56.

Cao,M. *et al.* (2013) Going the distance for protein function prediction: a new distance metric for protein interaction networks. *PLoS One*, **8**, e76339.

Chen,J. *et al.* (2009) Disease candidate gene identification and prioritization using protein interaction networks. *BMC Bioinformatics*, **10**, doi:10.1186/1471–2105–10–73.

Cozzetto,D. *et al.* (2013) Protein function prediction by massive integration of evolutionary analyses and multiple data sources. *BMC Bioinformatics*, **14** (**Suppl. 3**), S1.

Darnell,S.J. *et al.* (2007) An automated decision-tree approach to predicting protein interaction hot spots. *Prot. Struct. Funct. Bioinform.*, **68**, 813–823.

Deng,M. *et al.* (2003) Assessment of the reliability of protein-protein interactions and protein function prediction. *Pacific Symposium on Biocomputing*, 140–151.

Deng,M. *et al.* (2004) Mapping Gene Ontology to proteins based on protein–protein interaction data. *Bioinformatics*, **20**, 895–902.

Du,D. *et al.* (2012) Systematic differences in signal emitting and receiving revealed by pagerank analysis of a human protein interactome. *PLoS One*, **7**, e44872.

Dutkowski,J. *et al.* (2013) A Gene Ontology inferred from molecular networks. *Nat. Biotechnol*, **31**, 38–45.

Erten,S. *et al.* (2011) VAVIEN: an algorithm for prioritizing candidate disease genes based on topological similarity of protein interaction networks. *J. Comput. Biol.*, **18**, 1561–1574.

Franceschini,A. *et al.* (2013) String v9. 1: protein-protein interaction networks, with increased coverage and integration. *Nucleic Acids Res.*, **41**, D808–D815.

Gandhi,T. *et al.* (2006) Analysis of the human protein interactome and comparison with yeast, worm and fly interaction datasets. *Nat. Genet.*, **38**, 285–293.

Gitter,A. *et al.* (2011) Discovering pathways by orienting edges in protein interaction networks. *Nucleic Acids Res.*, **39**, e22–e22.

Hishigaki,H. *et al.* (2001) Assessment of prediction accuracy of protein function from protein-protein interaction data. *Yeast*, **18**, 523–531.

Kanehisa,M. and Goto,S. (2000) KEGG: Kyoto encyclopedia of genes and genomes. *Nucleic Acids Res.*, **28**, 27–30.

Kohler,S. *et al.* (2008) Walking the interactome for prioritization of candidate disease genes. *Am. J. Hum. Genet.*, **82**, 949–958.

Liao,C.-S. *et al.*IsoRankN: spectral methods for global alignment of multiple protein networks. *Bioinformatics*, **25**, i253–i258.

Licata,L. *et al.* (2012) Mint, the molecular interaction database: 2012 update. *Nucleic Acids Res.*, **40**, D857–D861.

Liu,W. *et al.* (2009) Proteome-wide prediction of signal flow direction in protein interaction networks based on interacting domains. *Mol. Cell. Proteom.*, **8**, 2063–2070.

Mering,V.C. *et al.* (2002) Comparative assessment of large-scale data sets of protein-protein interactions. *Nature*, **417**, 399–403.

Moustakas,A. (2002) Smad signalling network. *J. Cell Sci.*, **115**, 3355–3356.

Nabieva,E. *et al.* (2005) Whole-proteome prediction of protein function via graph-theoretic analysis of interaction maps. *Bioinformatics*, **21**, 302–310.

Reguly,T. *et al.* (2006) Comprehensive curation and analysis of global interaction networks in *Saccharomyces cerevisiae*. *J. Biol.*, **5**, 11.

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

Schwikowski,B. *et al.* (2000) A network of protein-protein interactions in yeast. *Nat. Biotechnol.*, **18**, 1257–1261.

Sharan,R. *et al.* (2005) Conserved patterns of protein interaction in multiple species. *Proc. Natl Acad. Sci. USA*, **102**, 1974–1979.

Sharan,R. *et al.* (2007) Network-based prediction of protein function. *Mol. Syst. Biol.*, **3**, 88.

Stark,C. *et al.* (2006) BioGRID: a general repository for interaction datasets. *Nucleic Acids Res.*, **34** (**Suppl. 1**), D535–D539.

Vanunu,O. *et al.* (2010) Associating genes and protein complexes with disease via network propogation. *PLoS Comput. Biol.*, **6**, e1000641.

Vazquez,A. *et al.* (2003) Global protein function prediction from protein-protein interaction networks. *Nat. Biotechnol.*, **21**, 696–700.

Voevodski,K. *et al.* (2009) Spectral affinity in protein networks. *BMC Syst. Biol.*, **3**, 112.

Xenarios,I. *et al.* (2002) DIP, the database of interacting proteins: a research tool for studying cellular networks of protein interactions. *Nucleic Acids Res.*, **30**, 303–305.