Huang *et al. BMC Systems Biology* 2016, **10**(Suppl 2):45
DOI 10.1186/s12918-016-0296-x

BMC Systems Biology

RESEARCH Open Access

Protein-protein interaction prediction based on multiple kernels and partial network with linear programming

Lei Huang[1], Li Liao[1*] and Cathy H. Wu[1,2]

From IEEE International Conference on Bioinformatics and Biomedicine 2015
Washington, DC, USA. 9-12 November 2015

Abstract

Background: Prediction of *de novo* protein-protein interaction is a critical step toward reconstructing PPI networks, which is a central task in systems biology. Recent computational approaches have shifted from making PPI prediction based on individual pairs and single data source to leveraging complementary information from multiple heterogeneous data sources and partial network structure. However, how to quickly learn weights for heterogeneous data sources remains a challenge. In this work, we developed a method to infer *de novo* PPIs by combining multiple data sources represented in kernel format and obtaining optimal weights based on random walk over the existing partial networks.

Results: Our proposed method utilizes Barker algorithm and the training data to construct a transition matrix which constrains how a random walk would traverse the partial network. Multiple heterogeneous features for the proteins in the network are then combined into the form of weighted kernel fusion, which provides a new "adjacency matrix" for the whole network that may consist of disconnected components but is required to comply with the transition matrix on the training subnetwork. This requirement is met by adjusting the weights to minimize the element-wise difference between the transition matrix and the weighted kernels. The minimization problem is solved by linear programming. The weighted kernel fusion is then transformed to regularized Laplacian (RL) kernel to infer missing or new edges in the PPI network, which can potentially connect the previously disconnected components.

Conclusions: The results on synthetic data demonstrated the soundness and robustness of the proposed algorithms under various conditions. And the results on real data show that the accuracies of PPI prediction for yeast data and human data measured as AUC are increased by up to 19 % and 11 % respectively, as compared to a control method without using optimal weights. Moreover, the weights learned by our method Weight Optimization by Linear Programming (WOLP) are very consistent with that learned by sampling, and can provide insights into the relations between PPIs and various feature kernel, thereby improving PPI prediction even for disconnected PPI networks.

Keywords: Protein interaction network, Network inference, Interaction prediction, Random walk, Linear programming

*Correspondence: liliao@udel.edu
[1]Department of Computer and Information Sciences, University of Delaware, 18 Amstel Avenue, 19716, Newark, Delaware, USA
Full list of author information is available at the end of the article

Background

Protein-protein interaction (PPI) plays an essential role in many cellular processes. In order to have a better understanding of intracellular signaling pathways, modeling of protein complex structures and elucidating various biochemical processes, many high-throughput experimental methods, such as yeast two-hybrid system and mass spectrometry method, have been used to uncover protein interactions. However, these methods are known to be prone to having high false-positive rates, besides their high cost. Therefore, great efforts have been made to develop efficient and accurate computational methods for PPI prediction.

Many pair-wise biological similarity based computational approaches have been developed to predict if any given pair of proteins interact with each other, based on various properties such as sequence homology, gene co-expression, phylogenetic profiles, three-dimensional structural information, etc. [1–7]. However, without first principles to tell deterministically if two given proteins interact or not, the pair-wise biological similarity based on various features and attributes can run out its predictive power, as the signals may be weak, noisy, or inconsistent, which can present serious issues even for methods based on integrated heterogeneous pair-wise features, e.g. genomic features, semantic similarities, etc. [8–11].

To circumvent the limitations with using pair-wise biological similarity, pair-wise topological features have been used to measure the similarity for any given node pair to make PPI prediction for the corresponding proteins [12–15], if a PPI network is constructed with nodes representing proteins and edges representing interactions. Moreover, to go beyond these node centric topological features and get the whole network structure involved, variants of random walk [16] based methods [17–19] have been developed, but the computational cost of these methods increases by N times for all-against-all PPI prediction. Thus many kernels on network for link prediction and semi-supervised classification have been systematically studied [20], which can measure the random-walk distance for all node pairs at once. But both the variants of random walk and random walk based kernels do not perform well in detection of interacting proteins when the direct edge connecting them in the network is removed and the remaining path connecting them is long [20]. Besides, instead of computing proximity measures between nodes from the network structure explicitly, many latent features based on rank reduction and spectral analysis have been utilized to do prediction, such as geometric de-noise methods [1, 21], multi-way spectral clustering [22], matrix factorization based methods [23, 24]. Mostly, the prediction task of these methods will be reduced to the convex optimization problem whose objective function should be carefully designed to ensure fast convergence and avoid being stuck in the local optima. Furthermore, biological features and topological features can supplement each other to improve the prediction performance, such as by assigning weights to edges in the network based on pair-wise biological similarity scores. Then, methods based on explicit or latent features, such as supervised random walk [19] or matrix factorization method, can be applied to the weighted network to make prediction, based on multi-modal biological sources. [23, 24]. However, for these methods, only the pair-wise features for the existing edges in the PPI network will be utilized, even though from a PPI prediction perspective what is particularly useful is to incorporate pair-wise features for node pairs that are not currently linked by a direct edge but will if a new edge (PPI) is predicted.

Therefore, it is of great interest if we can infer PPI network directly from multi-modal biological features kernels that involve all node pairs. It not only can help us improve prediction performance but also provide insights into relations between PPIs and various similarity features of protein pairs. Yamanishi et al. [25] developed a method based on kernel canonical correlation analysis to infer PPI networks from multiple types of genomic data. However, in that work all genomic kernels are simply added together, with no weights to regulate these heterogeneous and potentially noisy data sources for their contribution towards PPI prediction. Meanwhile, it seems that the partial network needed for supervised learning based on kernel CCA need to be sufficiently large, e.g., a leave-one-out cross validation is used, to attain good performance. In Huang et al. [26] the weights for different data sources are optimized using a sampling based method, ABC-DEP, which is computationally demanding.

In this paper, we propose a new method to infer de novo PPIs by combining multiple data sources represented in kernel format and obtaining optimal weights based on random walk over the existing partial network. The novelty of the method lies in the use of Barker algorithm to construct the transition matrix for the training subnetwork and find the optimal weights by linear programing to minimize the element-wise difference between the transition matrix and the adjacency matrix, aka, the weighted kernel from multiple heterogeneous data. Then we apply regularized Laplacian kernel (RL) to the weighted kernel to infer missing or new edges in the PPI network. A preliminary version of this work was described in [27]. Relative to that paper, the current work includes extension to handle interaction prediction problem for PPI networks consisting of disconnected components and new results on the human PPI network, which is much more sparse than the yeast PPI network. Our method can circumvent the issue of unbalanced data faced with many machine learning methods in bioinformatics by training on only a small partial network. Our method works particularly well with detecting interactions between nodes that are far apart in the network.

Methods

Problem definition

Formally, a PPI network can be represented as a graph $G = (V, E)$ with V nodes (proteins) and E edges (interactions). G is defined by the adjacency matrix A with $V \times V$ dimension:

$$A(i,j) = \begin{cases} 1, if\, (i,j) \in E \\ 0, if\, (i,j) \notin E \end{cases} \quad (1)$$

where i and j are two nodes in the nodes set V, and (i,j) represents an edge between i and j, $(i,j) \in E$. The graph is called *connected* if there is a path of edges to connect any two nodes in the graph. Given many PPI networks are not connected and has many connected component with various size, we select a large connected component (e.g. largest connected component) as golden standard network to do supervised learning. Specifically, by adopting the same setting in [26], we divide the golden standard network into three parts: connected training network $G_{tn} = (V, E_{tn})$, validation set $G_{vn} = (V_{vn}, E_{vn})$ and testing set $G_{tt} = (V_{tt}, E_{tt})$, such that $E = E_{tn} \cup E_{vn} \cup E_{tt}$, and any edge in G can only belong to one of these three parts.

A kernel is a symmetric positive definite matrix K, whose elements are defined as a real-valued function $K(u, v)$ satisfying $K(u, v) = K(u, v)$ for any two proteins u and v in the data set. Intuitively, the kernel built from a given dataset can be regarded as a measure of similarity between protein pairs with respect to the biological properties, from which kernel function takes its value. Treated as an adjacency matrix, a kernel can also be thought of as a complete network in which all the proteins are connected by weighted edges. Kernel fusion is a way to integrate multiple kernels from different data sources by a linear combination. For our task, this combination is made of the connected training network and various feature kernels $K_i,\ i = 1, 2, 3...n$, which formally is defined by Eq. (2)

$$K_{fusion} = W_0 G_{tn} + \sum_{i=1}^{n} W_i K_i,\ where\, K_i(u, v) = \frac{K_i(u, v)}{\sum_{w} K_i(u, w)} \quad (2)$$

Note that the training network is incomplete, i.e., with many edges taken away and reserved as testing examples. Therefore, the task is to infer or recover the interactions in the testing set G_{tt} based on the kernel fusion. Once the kernel fusion is obtained, it will be used to make PPI inference, in the spirit of random walk. However, instead of directly doing random walk, we apply regularized Laplacian (RL) kernel to the kernel fusion, which allows for PPI inference on the whole network level. The regularized Laplacian kernel [28, 29] is also called the normalized random walk with restart kernel in Mantrach et al. [30] because of the underlying relations to the random walk with restart model [17, 31]. Formally, it is defined as Eq. (3), where $L = D - A$ is the Laplacian matrix made of the adjacency matrix A and the degree matrix D, and $0 < \alpha < \rho(L)^{-1}$ and $\rho(L)$ is the spectral radius of L. Here, we use kernel fusion in place of the adjacency matrix, generating a regularized Laplacian matrix RL_K, so that various feature kernels in Eq. (2) are incorporated in influencing the random walk with restart on the weighted networks [19]. With the regularized Laplacian matrix, no random walk is actually needed to measure how "close" two nodes are and then use that closeness to infer if the two corresponding proteins interact. Rather, RL_K is interpreted as a probability matrix P in which $P_{i,j}$ indicates the probability of an interaction for protein i and j.

$$RL = \sum_{k=0}^{\infty} \alpha^k (-L)^k = (I + \alpha * L)^{-1} \quad (3)$$

To ensure good inference, it is important to learn optimal weights for G_{tn} and various K_i to build kernel fusion K_{fusion}. Otherwise, given the multiple heterogeneous kernels from different data sources, the kernel fusion without optimized weights is likely to generate erroneous inference on PPI.

Weight optimization with linear programming (WOLP)

Given a PPI network, the probability of interaction between any two proteins is measured in terms of how likely a random walk in the network starting at one node will reach the other node. Here, instead of solely using the adjacency matrix A to build the transition matrix, we integrate kernel features as edge strength. Then the stochastic transition matrix Q can be built by:

$$Q(i,j) = K_{fusion}(i,j) \quad (4)$$

Assuming the network is reasonably large, for a start node s, the probability distribution p of reaching all nodes via random walk in t steps can be obtained by applying the transition matrix Q t times:

$$p^t = Q^t p^0 \quad (5)$$

where the initial distribution p^0 is

$$p_i^0 = \begin{cases} 1, if\ i = s \\ 0, otherwise \end{cases} \quad (6)$$

The stationary distribution p, when letting t go to infinity, is obtained by solving the following eigenvector equation:

$$p = Q\,p \tag{7}$$

This stationary distribution provides constraints at optimizing the weights. For example, the positive training examples (nodes that are closer to the start node s) should have higher probability than the negative training examples (nodes that are far away from s). In Backstrom et al. [19], this is used as constraint in minimizing the L2 norm of the weights for optimal weights. In the work of Backstrom et al. [19], a gradient descent optimization method is adopted to get optimal weights, and only the pair-wise features for the existing edges in the network are utilized, which means $Q(i,j)$ is nonzero only for edge (i,j) that already exists in the training network. To leverage more information from multiple heterogeneous sources, in our case the $Q(i,j)$, as defined in Eq. (4), are nonzero unless there is no features for edge i,j in all kernels K_a. Having many non-zero elements in Q makes it much more difficult for the traditional gradient descent optimization method to converge and to find the global optima.

In this work, we propose to solve the weights optimization differently. We can consider the random walk with restarts process shown in Eq. (5) as a Markov model, with a stationary distribution p. Knowing the stationary distribution, the transition matrix can be obtained by solving the reverse eign problem using the well-known Metropolis algorithm or Barker algorithm. In this work, we adopt Barker algorithm [32], which gives the transition matrix as follows.

$$Q^b(i,j) = \frac{p_j}{p_i + p_j} \tag{8}$$

Now we can formulate weights optimization by minimizing the element-wise difference between Q^b and Q. Namely,

$$W^* = \underset{W}{argmin}\,||Q - Q^b||^2 \tag{9}$$

As the number of elements in the transition matrix is typically much larger than the number of weights, Eq. (9) provides more equations than the number of variables, making it an overdetermined linear equation system. This overdetermined linear equation system can be solved with linear programming using standard programs in [33, 34].

Now, in the spirit of supervised learning, given the training network G_{tn} and a start node s, we calculate p' by doing random walk that start at s in G_{tn} as an approximation of p, and $Q^b(i,j) = \frac{p'_j}{p'_i+p'_j}$. Note that $Q^b(i,j)$ from Barker algorithm is an asymmetric matrix whereas Q composed from kernel fusion is a symmetric matrix. So, we do not need to use all equations obtained from Eq. (9) to calculate the weights. Instead we can just use equations derived from the upper or lower triangle part of the matrices Q^b and Q. This reduction of number of equations will not pose an issue as the system is overdetermined; rather this will help mitigate the issue of being overdetermined. Specifically, as shown in Fig. 1, for all destination nodes V in G_{tn}, namely reachable from start node s, we divide them into three subsets D, L and M, where D consists of near neighbors of s in G_{tn} with the shortest path between s and nodes D_i satisfying $d(s, D_i) < \epsilon 1$; and L includes faraway nodes of s in G_{tn} with the shortest path between s and nodes L_i satisfying $d(s, L_i) > \epsilon 2$; and the rest of nodes are in subset M. Then the system of equations of Eq. (9) is updated to Eq. (10), where $u < v$ indicates lower triangle mapping, and $u > v$ indicates upper triangle mapping.

$$\begin{aligned} &W_0 G_{tn}(u,v) + \sum_{i=1}^{n} W_i K_i(u,v) = Q^b(u,v), \\ &if\ u,v \in D \cup L\ \wedge\ K_i(u,v)! = 0 \wedge (u < v \vee u > v) \end{aligned} \tag{10}$$

The optimized weights W^* can then be plugged back into Eq. (4) to form an optimal transition matrix for the whole set of nodes, and the random walk from the source node using this optimal transition matrix hence leverages the information from multi data sources and is expected to give more accurate prediction for missing and/or de novo links: nodes that are most frequented by random walk are more likely, if not yet detected, to have a direct link to the source node. The formal procedure for solving this overdetermined linear system and inferring PPIs for a particular node is shown by Algorithm 1.

Algorithm 1 *Basic WOLP*

```
Input: $G_{tn} \leftarrow$ training network
       $s \leftarrow$ start node
       $K \leftarrow$ feature kernels
Output: p
1: $p' \leftarrow RWR(G_{tn}, s)$ // $RWR$ is the Random Walk with Restart algorithm [17]
2: $Q^b(i,j) \leftarrow \frac{p'_j}{p'_i+p'_j}$
3: $W \leftarrow by\ solving\ Eq.$ (10) // $W$ indicates the optimal weights
4: $Q \leftarrow by\ Eq.$ (2) $and\ Eq.$ (4)
5: $p \leftarrow RWR(Q, s)$
```

PPI prediction and network inference

As we discussed in introduction section, the use of random walk from a single start node is not efficient for all-against-all prediction, especially for the large and sparse PPI networks. Therefore, it would be of great interest if the weights learned by WOLP based on a single start node can also work network wide. Actually, it is widely observed that the many biological networks contain several hubs (i.e., nodes with with high degree) [35]. Thus we extend our algorithm to all-against-all PPI inference by hypothesizing that the weights learned based on a start node with

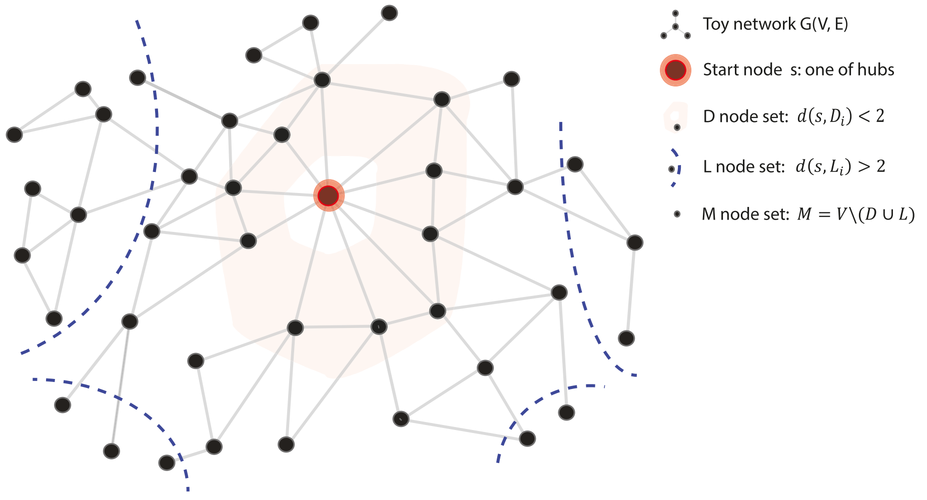
Toy network G(V, E)
Start node s: one of hubs
D node set: $d(s, D_i) < 2$
L node set: $d(s, L_i) > 2$
M node set: $M = V \setminus (D \cup L)$

Fig. 1 Schematic illustration of node sets D, M and L, with respect to source node s

high degree would be utilizable by other nodes. We will verify this hypothesis by doing all-against-all PPI inference for real PPI network.

We design a supervised WOLP version that can learn weights more accurately for the large and sparse PPI network. Similarly, if the whole PPI network is connected, then the golden standard network is itself; otherwise, the golden standard network that used to do supervised learning should be a large component of the disconnected PPI network. To do so, we divide the golden standard network into three parts: connected training network $G_{tn} = (V, E_{tn})$, validation set $G_{vn} = (V_{vn}, E_{vn})$ and testing set $G_{tt} = (V_{tt}, E_{tt})$, such that $E = E_{tn} \cup E_{vn} \cup E_{tt}$, and any edge in G can only belong to one of these three parts. Then we use WOLP to learn weights based on G_{tn} and G_{vn}, and finally use G_{tt} to verify the prediction capability of these weights. The main structure of our method is shown by Algorithm 2, and the supervised version of WOLP is shown by Algorithm 3. The *while* loop in Algorithm 3 is used to find optimal setting of *D*, *L* and mapping strategy(upper or lower) that can generate best weights W_{opt} with respect to inferring and G_{tn} and G_{vn}.

Moreover, many existing network-level link prediction or matrix completion methods [1, 19, 21, 23, 24] can only work well on connected PPI networks, but detection of interacting pairs for disconnected PPI networks has been a challenge for these methods. However, our WOLP method can solve the problem effectively. Because various feature kernels can connect all the disconnected components of the originally disconnected PPI network; and we believe once the optimal weights have been learned based on the training network generated from a large connected component (e.g. largest connected component), they can also be used to build the kernel fusion when the prediction task scale up to the originally disconnected PPI network.

Algorithm 2 *PPI Inference(WOLP) for the Connected PPI Network*

```
Input: $RL$ ← Regularized Laplacian prediction kernel
       $G$ ← PPI network
       $K$ ← feature kernels with same size of $G$
Output: Inferred network
1: $\{G_{tn}, G_{vn}, G_{tt}\} \leftarrow G$
2: $W^{opt} \leftarrow$ Supervised WOLP$(G_{tn}, G_{vn}, G_{tt}, RL, K)$
3: $OPT\text{-}K \leftarrow W_0^{opt} G_{tn} + \sum_{i=1}^{n} W_i^{opt} K_i$ // $OPT\text{-}K$ is the optimal kernel fusion based on weights learned by WOLP
4: $RL_{OPT\text{-}K} \leftarrow RL(OPT\text{-}K)$ // Apply RL model to the kernel fusion
5: Rank $RL_{OPT\text{-}K}$ and infer $G_{tt}$
```

To do so, we update the Algorithm 2 to Algorithm 4 that shows the detailed process of interaction prediction for disconnected PPI networks. Given an originally disconnected network *G*, firstly, we learn the optimal weights by Algorithm 3 based on a large connected component G_{cc} of *G*. After that, we randomly divide the edge set *E* of the disconnected *G* into training edge set G_{tn} and testing edge set G_{tt}, and use the optimal weights we learned before directly to linearly combine G_{tn} and other corresponding feature kernels to build the kernel fusion, and finally evaluate the performance through predicting G_{tt}. Here we call G_{tn} training edge set, because G_{tn} no longer needs to be connected to learn any weights.

Results and discussion

We examine the soundness and robustness of the proposed algorithms with use of both synthetic and real data. Our goal here is to demonstrate that the weights

Algorithm 3 *Supervised WOLP*

```
Input: $G_{tn}, G_{vn}, G_{tt}, RL, K$
Output: $W^{opt}$
 1: $s \leftarrow$ a start node with large degree in $G_{tn}$
 2: $r \leftarrow$ diameter of $G_{tn}$ respect to the start node $s$
 3: $D \leftarrow$ direct neighbors of start node $s$
 4: while $r > 1$ do
 5:     $L \leftarrow V_i$ if $d(s, V_i) >= r$ // $V$ is the nodes set of $G_{tn}$, $d$ is the shortest path
 6:     $p' \leftarrow RWR(G_{tn}, s)$ // random walk with restarts from start node $s$ in $G_{tn}$
 7:     $Q^b(i,j) \leftarrow \frac{p'_j}{p'_i+p'_j}$
 8:     $W1 \leftarrow$ by solving Eq. (10) with upper triangle mapping
 9:     $W2 \leftarrow$ by solving Eq. (10) with lower triangle mapping
10:     $OPT\text{-}K1 \leftarrow W1_0 G_{tn} + \sum_{i=1}^{n} W1_i K_i$
11:     $OPT\text{-}K2 \leftarrow W2_0 G_{tn} + \sum_{i=1}^{n} W2_i K_i$
12:     $R1 \leftarrow Inference(RL, OPT\text{-}K1, G_{vn})$
13:     $R2 \leftarrow Inference(RL, OPT\text{-}K2, G_{vn})$
        // In the Inference function, RL has been applied to kernel fusion OPT-Ki to infer validation edges $G_{vn}$. Ri represent results of inferring $G_{vn}$
14:     if $R1 > R^{opt}$ then
15:         $R^{opt} \leftarrow R1$
16:         $W^{opt} \leftarrow W1$ // $R^{opt}$ indicates the optimal result of inferring $G_{vn}$, $W^{opt}$ indicates the optimal weights
17:     end if
18:     if $R2 > R^{opt}$ then
19:         $R^{opt} \leftarrow R2$
20:         $W^{opt} \leftarrow W2$
21:     end if
22:     $r \leftarrow r - 1$
23: end while
```

Algorithm 4 *PPI Inference(WOLP) for the Disconnected PPI Network*

```
Input: $RL \leftarrow$ Regularized Laplacian prediction kernel
       $G \leftarrow$ disconnected PPI network
       $K \leftarrow$ feature kernels with same size of $G$
Output: Inferred network
 1: $G_{cc} \leftarrow G$ // get a large connected component from $G$
 2: $\{G_{tn'}, G_{vn'}, G_{tt'}\} \leftarrow G_{cc}$
 3: $W^{opt} \leftarrow Supervised\ WOLP(G_{tn'}, G_{vn'}, G_{tt'}, RL, K')$ // $K'$ is a sub matrix of $K$ with same size of $G_{tn'}$
 4: $\{G_{tn}, G_{tt}\} \leftarrow G$
 5: $OPT\text{-}K \leftarrow W_0^{opt} G_{tn} + \sum_{i=1}^{n} W_i^{opt} K_i$ // OPT-K is the optimal kernel fusion based on weights learned by WOLP
 6: $RL_{OPT\text{-}K} \leftarrow RL(OPT\text{-}K)$ // Apply RL model to the kernel fusion
 7: Rank $RL_{OPT\text{-}K}$ and infer $G_{tt}$
```

obtained by our method can help build a better kernel fusion leading to more accurate PPI prediction.

Experiments on single start node and synthetic data

A synthetic scale-free network G_{syn} with 5,093 nodes is generated by Copying model [36]: G_{syn} starts with three nodes connected in a triad. Remaining nodes have been added one by one with exactly two edges for each. For instance, when a node u is added, two edges(u, v_i), $i = 1, 2$ between u and existing nodes v_i will be added accordingly. Node v_i is randomly selected with probability 0.8, and otherwise v_i is selected with probability proportional to its current degree. The parameters we chose is to guarantee G_{syn} has similar size and density to DIP yeast PPI network [37] that we will use to do PPI inference later. Then we build eight synthetic feature kernels for G_{syn}. The feature kernels can be classified into three categories: 3 noisy kernels, 4 positive kernels and a mixture kernel, which are defined by Eq. (11)

$$\begin{cases} K_{noise} = R_{5093} + (R_{5093} + \eta).*rand_{diff}(J_{5093}, G_{syn}, \rho_i) \\ \\ K_{postive} = R_{5093} + (R_{5093} + \eta).*rand_{sub}(G_{syn}, \rho_i) \\ \\ K_{mixture} = R_{5093} + (R_{5093} + \eta).*rand_{sub}(G_{syn}, \rho_i) \\ \qquad +(R_{5093} + \eta).*rand_{diff}(J_{5093}, G_{syn}, \rho_i) \end{cases} \tag{11}$$

Where R_{5093} indicates a 5093 by 5093 random matrix with elements between $[0, 1]$, which can also be seen asbackground noise matrix; J_{5093} indicates a 5093 by 5093 all-one matrix, $rand_{diff}(J_{5093}, G_{syn}, \rho_i)$ is used to randomly generate a difference matrix (if (i, j) = 1 in G_{syn} and (i, j) should be 0 in the difference matrix) between J_{5093} and G_{syn} with density ρ_i; $rand_{sub}(G_{syn}, \rho_i)$ is used to generate a subnetwork from G_{syn} with density ρ_i; ρ_i are different for each kernel; η is a positive parameter between $[0, 1]$ and R_{5093} will be rebuilt every time for each kernel.

The general process of experimenting with synthetic data is: we generate synthetic network G_{syn}, synthetic feature kernels K firstly, and then divide nodes V of G_{syn} into D, L and M, where D and L can be seen as training nodes, M can be seen as testing nodes. By using G_{syn}, start node s and K, we can get the stationary distribution p based on the optimized kernel fusion $K_{OPT} = W_0 G_{syn}(u, v) + \sum_{i=1}^{n} W_i K_i(u, v)$. Finally, we try to prove that K_{OPT} is better than the control kernel fusion $K_{EW} = G_{syn} + \sum_{i=1}^{n} K_i$ built by equal weights, if the $p(M)$ is more similar to $p'(M)$ based on G_{syn}, as compared to $p''(M)$ based on the control kernel fusion K_{EW}, where $p(M)$ indicates the rank of stationary probabilities respect to the testing node M. We evaluate the rank similarity between pairs $(p(M), p'(M))$ and $(p''(M), p'(M))$ by discounted cumulative gain (DCG) [38].

We carry out 10 experiments, each time we select one of the oldest 3 nodes as start node, and rebuild synthetic kernel K. In Table 1, the results show that DCG@20 between $p(M)$ and $p'(M)$ is consistently higher than that between $p''(M)$ and $p'(M)$ in all 10 experiments, indicating that the optimal weights W obtained by WOLP can help us build optimized kernel fusion that with better prediction capability, as compared to the control kernel fusion.

Experiments on network inference with real data

We use the yeast PPI network downloaded from DIP database (Release 20150101) [37] and the high-confidence human PPI network downloaded from PrePPI database [39] to test our algorithm.

Data and kernels of yeast PPI networks

For the yeast PPI network, some interactions without Uniprotkb ID have been filtered out in order to do name mapping and make use of genomic similarity kernels [40]. As a result, the originally disconnected PPI network contains 5093 proteins and 22,423 interactions. The largest connected component consists of 5030 proteins and 22,394 interactions, and is used to serve as the golden standard network.

Six feature kernels are included in PPI inference for the yeast data.

G_{tn}: G_{tn} is the connected training network that provides connectivity information. It can also be thought of as a base network to do the inference.

$K_{Jaccard}$ [41]: This kernel measure the similarity of protein pairs i, j in term of $\frac{neigbors(i) \cap neighbors(j)}{neighbors(i) \cup neighbors(j)}$.

K_{SN}: It measures the total number of neighbors of protein i and j, $K_{SN} = neighbors(i) + neighbors(j)$.

K_B [40]: It is a sequence-based kernel matrix that is generated using the BLAST [42].

K_E [40]: This is a gene co-expression kernel matrix constructed entirely from microarray gene expression measurements.

K_{Pfam} [40]: Similarity measure derived from Pfam HMMs [43]. All these kernels are normalized to the scale of $[0, 1]$ in order to avoid bias.

Data and kernels of human PPI networks

The originally disconnected human PPI network has 3993 proteins and 6669 interactions, which is much sparser than the yeast PPI network. The largest connected component that serve as the golden standard network contains 3285 proteins and 6310 interactions.

Eight feature kernels are included in PPI inference for the human data.

G_{tn}: G_{tn} is the connected training network that provides connectivity information. It can also be thought of as a base network to do the inference.

$K_{Jaccard}$ [41]: This kernel measure the similarity of protein pairs i, j in term of $\frac{neigbors(i) \cap neighbors(j)}{neighbors(i) \cup neighbors(j)}$.

K_{SN}: It measures the total number of neighbors of protein i and j, $K_{SN} = neighbors(i) + neighbors(j)$.

K_B: It is a sequence-based kernel matrix that is generated using the BLAST [42].

K_D: It is a domain-based similarity kernel matrix measured by the method of neighborhood correlation [44].

K_{BP}: It is a biological process based semantic similarity kernel measured by Resnik with BMA [45].

K_{CC}: It is a cellular component based semantic similarity kernel measured by Resnik with BMA [45].

K_{MF}: It is a molecular function based semantic similarity kernel measured by Resnik with BMA [45].

PPI inference based on the largest connected component

For cross validation, like in [26], the golden standard PPI network (largest connected component) is randomly divided into three parts that are connected training network G_{tn}, validation edge set G_{vn} and testing edge set G_{tt}, where G_{vn} is used to find optimal weights for feature kernels and G_{tt} is used to evaluate the inference capability of our method. The Table 2 shows detailed division for yeast and human PPI networks.

With the weights learned by WOLP and using i_{th} hub as the start node, we build the kernel fusion *WOLP-K-i* by Eq. (2). PPI network inference is made by RL kernel

Table 1 DCG@20 of rank comparison

Repetition	DCG@20$(p(M), p'(M))$	DCG@20$(p''(M), p'(M))$
1	0.7101	0.6304
2	0.9305	0.4423
3	0.4035	0.2657
4	0.8524	0.5690
5	0.7256	0.4417
6	0.3683	0.3009
7	0.7707	0.2753
8	1.0034	0.3663
9	0.7119	0.4603
10	0.6605	0.6123

Table 2 Division of golden standard PPI networks

Species	G_{tn}	G_{vn}	G_{tt}
Yeast	$V, E = \{5,030, 5,394\}$	$V, E = \{-, 1,000\}$	$V, E = \{-, 16,000\}$
Human	$V, E = \{3,285, 3,310\}$	$V, E = \{-, 300\}$	$V, E = \{-, 2,700\}$

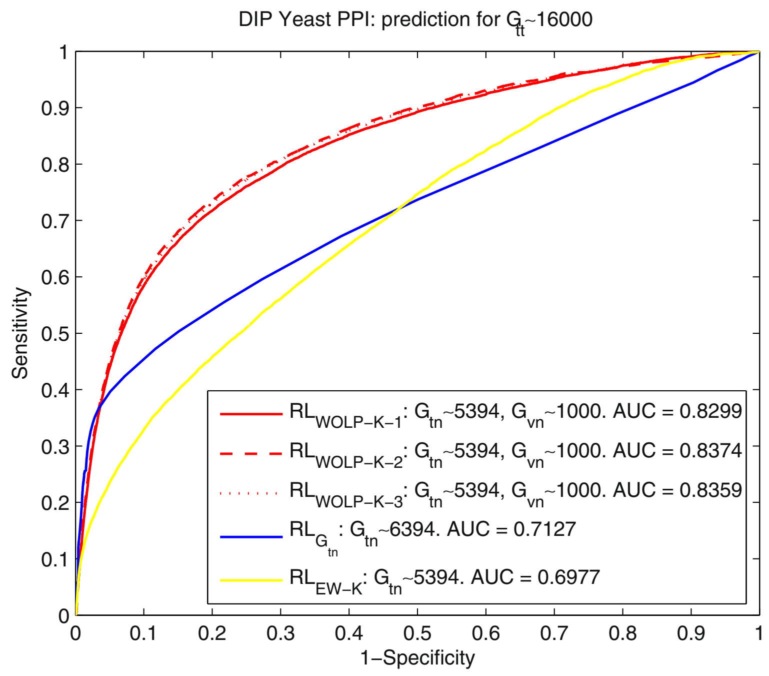

Fig. 2 Yeast: ROC curves of predicting $G_{tt} \sim 16000$ by $RL_{G_{tn}\sim 5394}$, $RL_{WOLP\text{-}K\text{-}i}$ and $RL_{EW\text{-}K}$

Eq. (3), and named as $RL_{WOLP\text{-}K\text{-}i}$, $i = 1, 2, 3$. The performance of inference is evaluated by how well the testing set G_{tt} is recovered. Specifically, all node pairs are ranked in decreasing order by their edge weights in the RL matrix, and edges in the testing set G_{tt} are labeled as positive and node pairs with no edges in G are labeled as negative. An ROC curve is plotted for true positive v.s. false positives, by running down the ranked list of node pairs. To make comparison, besides the PPI inferences $RL_{WOLP\text{-}K\text{-}i}$, $i = 1, 2, 3$ learned by our WOLP, we also include other two

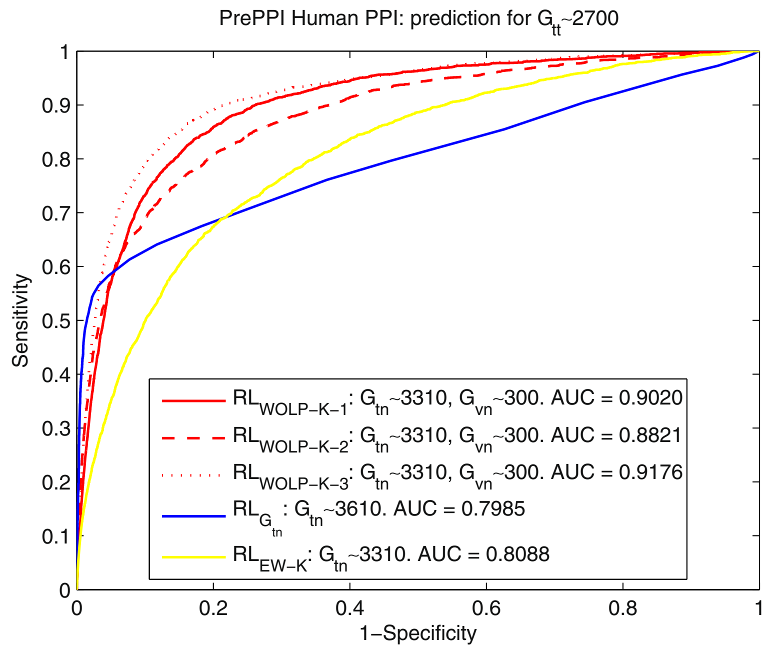

Fig. 3 Human: ROC curves of predicting $G_{tt} \sim 2700$ by $RL_{G_{tn}\sim 3610}$, $RL_{WOLP\text{-}K\text{-}i}$ and $RL_{EW\text{-}K}$

PPI network inferences: $RL_{G_{tn}}$ and $RL_{EW\text{-}K}$, where $RL_{G_{tn}}$ indicates RL based PPI inference is solely from the training network G_{tn}, and $RL_{EW\text{-}K}$ represents RL based PPI inference is from kernel fusion built by equal weights, e.g. $w_i = 1,\ i = 0, 1...n$. Additionally, $G_{set} \sim n$ indicates there is n number of edges in the set G_{set}, e.g. $G_{tn} \sim 5394$ means the connected training network G_{tn} contains 5394 edges.

The comparisons in terms of ROC curve and AUC for yeast and human data are shown by Fig. 2 and 3, the PPI reference $RL_{WOLP\text{-}K\text{-}i},\ i = 1, 2, 3$ based on our WOLP method significantly outperforms the two basic control methods, with about 17 % increase over $RL_{G_{tn}}$ and about 19.6 % over $RL_{EW\text{-}K}$ in term of AUC for the yeast data, and about 12.7 % increase over $RL_{G_{tn}}$ and about 11.3 % over $RL_{EW\text{-}K}$ in term of AUC for the human data. It is noted that the AUC of PPI inference $RL_{EW\text{-}K}$ based on the equally weighted built kernel fusion is no better or even worse than that of $RL_{G_{tn}}$ based on a really small training network, especially for the yeast data. It means there should be a lot of noises if we just naively combine different feature kernels to do PPI prediction.

Besides inferring PPI network by using weights learned based on the top three hubs in G_{tn}, we also test the predicting capability of PPI inferences by using top ten hubs as start nodes to learn the weights. We make 10 repetitions for the whole process: generating G_{tn}, choosing $i_{th},\ i = 1, 2, ...10$ hub as start node to learn the weights, then using these weights to build kernel fusion and finally to do the PPI inference. For the results based on top ten hubs in each repetition, the average AUC of inferring G_{tt} for yeast data and human data are shown in Tables 3 and 4 respectively. And the comparison shows the predicting capability of our method is consistently better than that of $RL_{G_{tn}}$ and $RL_{EW\text{-}K}$ for both yeast and human data.

Effects of the training data

Usually, given a golden standard data, we need to retrain the prediction model for different division of training set and testing set. However, if optimal weights have been found for building kernel fusion, our PPI network inference method enable us to train the model once, and do prediction or inference for different testing sets. To demonstrate that, we keep the two PPI inferences $RL_{WOLP\text{-}K\text{-}1}$ and $RL_{EW\text{-}K}$ obtained before (in last section) unchanged, and evaluate the prediction ability for different testing sets. We also examine how performance is affected by sizes of various sets. Specifically, while the size of training network G_{tn} for $RL_{G_{tn}}$ increases, sizes of G_{tn} for $RL_{WOLP\text{-}K\text{-}1}$ and $RL_{EW\text{-}K}$ are kept unchanged. Therefore, we design several experiments by dividing the golden standard network into G^i_{tn} and G^i_{tt}, $i = 1, ..., n$, and building PPI inference $RL_{G^i_{tn}}$ to predict G^i_{tt} for every time. To make comparison, we also use $RL_{WOLP\text{-}K\text{-}1}$ and $RL_{EW\text{-}K}$ to predict G^i_{tt}. As shown by the Table 5, for yeast data, $RL_{WOLP\text{-}K\text{-}1}$ trained on only 5,394 golden standard edges still performs better than the control methods, even for the $RL_{G_{tn}}$ that employ significantly more golden standard edges. Similarly, for the result of human data as shown by the Table 6, $RL_{WOLP\text{-}K\text{-}1}$ trained on only 3,310 golden standard edges still performs better than the control method

Table 4 Comparison of AUCs for human PPI prediction

Rep	Avg AUC($RL_{WOLP\text{-}K\text{-}1\sim10}$)	AUC($RL_{G_{tn}}$)	AUC($RL_{EW\text{-}K}$)
1	0.8871± 0.0122	0.8228	0.7823
2	0.8986± 0.0144	0.8106	0.8127
3	0.8988± 0.0088	0.8216	0.8088
4	0.8955± 0.0114	0.8161	0.8142
5	0.8994± 0.0089	0.8190	0.8088
6	0.8875± 0.0182	0.7927	0.8067
7	0.8904± 0.0237	0.8302	0.8096
8	0.8978± 0.0121	0.8205	0.8153
9	0.9011± 0.0101	0.7995	0.8130
10	0.8818± 0.0281	0.8078	0.8104

Table 3 Comparison of AUCs for yeast PPI prediction

Rep	Avg AUC($RL_{WOLP\text{-}K\text{-}1\sim10}$)	AUC($RL_{G_{tn}}$)	AUC($RL_{EW\text{-}K}$)
1	0.8367± 0.0134	0.7127	0.6976
2	0.7937± 0.0584	0.7768	0.7014
3	0.7802± 0.0545	0.7732	0.7009
4	0.7811± 0.0507	0.7406	0.7029
5	0.8349± 0.0301	0.7477	0.6991
6	0.8160± 0.0492	0.7180	0.7091
7	0.7670± 0.0636	0.7513	0.6992
8	0.8018± 0.0539	0.7739	0.7042
9	0.7989± 0.0552	0.7302	0.7017
10	0.8172± 0.0388	0.7387	0.6953

Table 5 Effects of training data size on prediction performance (AUC) for yeast

	$G_{tt} \sim 15000$	$G_{tt} \sim 14000$	$G_{tt} \sim 13000$
$RL_{WOLP\text{-}K\text{-}1:G_{tn}\sim5394}$	0.8658	-	-
$RL_{G_{tn}\sim7394}$	0.7931	-	-
$RL_{EW\text{-}K:G_{tn}\sim5394}$	0.7519	-	-
$RL_{WOLP\text{-}K\text{-}1:G_{tn}\sim5394}$	-	0.8659	-
$RL_{G_{tn}\sim8394}$	-	0.8538	-
$RL_{EW\text{-}K:G_{tn}\sim5394}$	-	0.7537	-
$RL_{WOLP\text{-}K\text{-}1:G_{tn}\sim5394}$	-	-	0.8659
$RL_{G_{tn}\sim9394}$	-	-	0.8619
$RL_{EW\text{-}K:G_{tn}\sim5394}$	-	-	0.7520

Table 6 Effects of training data size on prediction performance (AUC) for human

	$G_{tt} \sim 2600$	$G_{tt} \sim 2100$	$G_{tt} \sim 1600$
$RL_{WOLP\text{-}K\text{-}1:G_{tn}\sim 3310}$	0.9277	-	-
$RL_{G_{tn}\sim 3710}$	0.8359	-	-
$RL_{EW\text{-}K:G_{tn}\sim 3310}$	0.8590	-	-
$RL_{WOLP\text{-}K\text{-}1:G_{tn}\sim 3310}$	-	0.9305	-
$RL_{G_{tn}\sim 4210}$	-	0.8779	-
$RL_{EW\text{-}K:G_{tn}\sim 3310}$	-	0.8620	-
$RL_{WOLP\text{-}K\text{-}1:G_{tn}\sim 3310}$	-	-	0.9338
$RL_{G_{tn}\sim 4710}$	-	-	0.9227
$RL_{EW\text{-}K:G_{tn}\sim 3310}$	-	-	0.8639

$RL_{G_{tn}}$ that employ over 1,000 more golden standard edges.

Detection of interacting pairs far apart in the network

It is known that the basic idea of using random walk or random walk based kernels [17–20] for PPI prediction is that good interacting candidates usually are not far-away from the start node, e.g. only 2, 3 edges away in the network. Consequently, the testing nodes have been chosen to be within a certain distance range, which largely contributes to the good performance reported by many network-level link prediction methods. In reality, however, a method that is capable and good at detecting interacting pairs far apart in the network can be even more useful, such as in uncovering cross talk between pathways that are not nearby in the PPI network.

To investigate how our proposed method performs at detecting faraway interactions, we still use $RL_{G_{tn}\sim 6394}$, $RL_{WOLP\text{-}K\text{-}1}$ and $RL_{EW\text{-}K}$ for yeast data, and $RL_{G_{tn}\sim 3610}$, $RL_{WOLP\text{-}K\text{-}1}$ and $RL_{EW\text{-}K}$ for human data to infer PPIs, but we select node pairs (i,j) that satisfy $dist(i,j) > 3$ *given* G_{tn} from G_{tt} as new testing set and name it $G_{tt}^{(dist(i,j)>3)}$. Figures 4 and 5 show the results of yeast and human data respectively, which demonstrate that $RL_{WOLP\text{-}K\text{-}1}$ has not only a significant margin over the control methods in detecting long-distance PPIs but also maintains a high ROC scores of 0.8053 (for yeast data) and 0.8833 (for human data) comparable to that of all PPIs. In contrast, in both Figs. 4 and 5, $RL_{G_{tn}}$ performs poorly and worse than $RL_{EW\text{-}K}$, which means the traditional RL kernel based on adjacent training network alone cannot detect faraway interactions well.

Detection of interacting pairs for disconnected PPI networks

For the originally disconnected yeast PPI network, we randomly divide the edge set E into training edge set G_{tn} with 6295 edges and testing edge set G_{tt} with 16,128 edges. Similarity, based on a random division, the number of edges of training edge set G_{tn} and testing edge set G_{tt} are 3305 and 3364 for the originally disconnected human PPI network. The detailed information of the originally disconnected yeast and human PPI networks can be found in the subsection of data description. The Figs. 6 and 7 show the predicting results of yeast and human data respectively, which indicate $RL_{WOLP\text{-}K\text{-}i}$, $i = 1,2,3$ perform steady well on inferring interactions for both yeast and human data and are obviously better than $RL_{EW\text{-}K}$. $RL_{G_{tn}}$ is

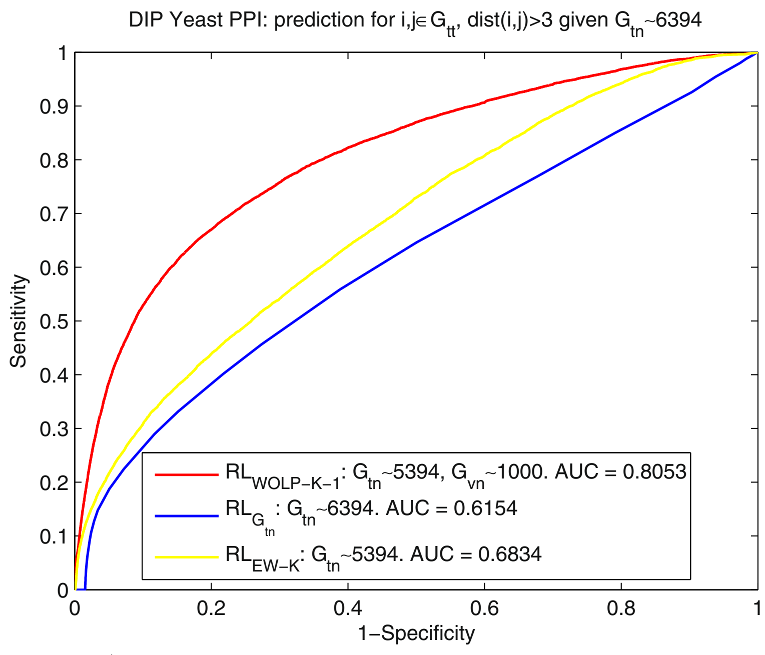

Fig. 4 Yeast: ROC curves of predicting $G_{tt}^{(dist(i,j)>3)}$ by $RL_{G_{tn}\sim 6394}$, $RL_{WOLP\text{-}K\text{-}1}$ and $RL_{EW\text{-}K}$

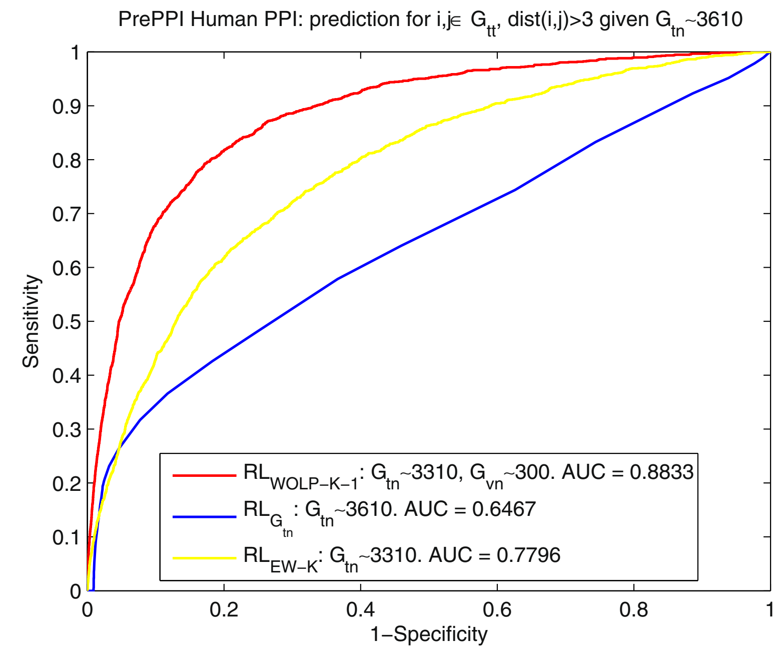

Fig. 5 Human: ROC curves of predicting $G_{tt}^{(dist(i,j)>3)}$ by $RL_{G_{tn}\sim 3610}$, $RL_{WOLP\text{-}K\text{-}1}$ and $RL_{EW\text{-}K}$

not included in this comparison, because it is not feasible for prediction tasks of disconnected PPI networks.

Analysis of weights

As our method incorporates multiple heterogeneous data, it can be insightful to inspect the final optimal weights. Therefore, we compare the average of weights learned by WOLP to the average of weights learned from revised ABC-DEP sampling method [26, 46], which is more computationally demanding. For the yeast data, the Fig. 8 shows that these two methods produce consistent results: these weights indicate that K_{SN} and K_{Pfam} are the

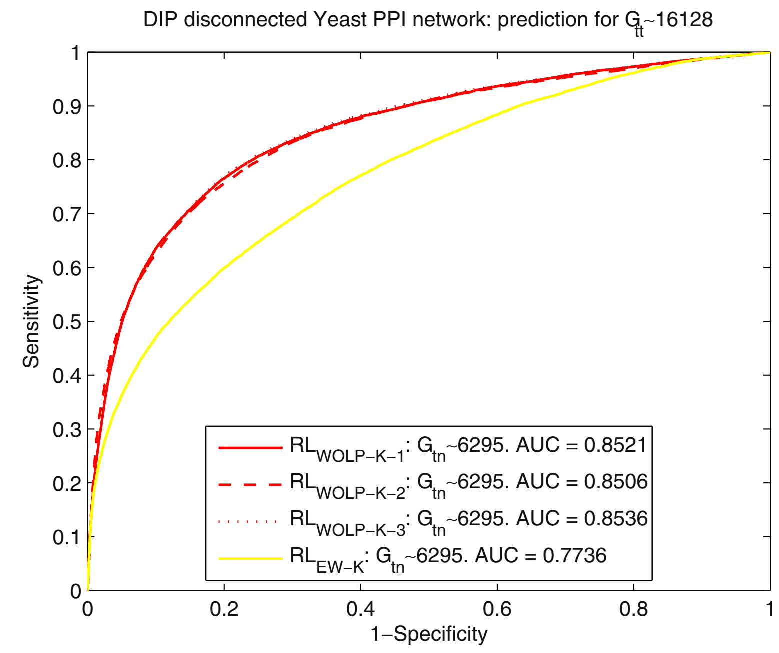

Fig. 6 Yeast: ROC curves of predicting $G_{tt} \sim 16128$ by $RL_{WOLP\text{-}K\text{-}1}$ and $RL_{EW\text{-}K}$ for disconnected PPI network

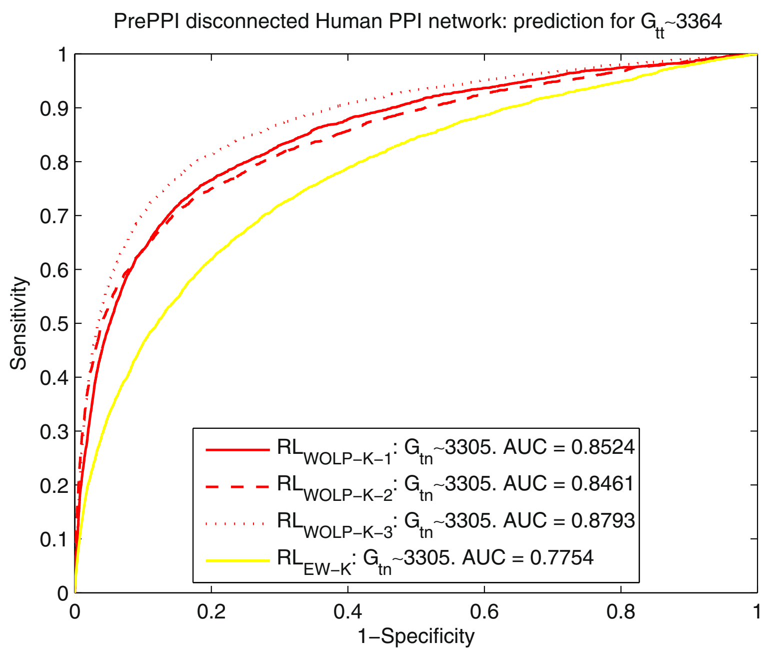

Fig. 7 Human: ROC curves of predicting $G_{tt} \sim 3364$ by $RL_{WOLP\text{-}K\text{-}1}$ and $RL_{EW\text{-}K}$ for disconnected PPI network

predominant contributors to PPI prediction. This observation is consistent with the intuition that proteins interact via interfaces made of conserved domains [47], and PPI interactions can be classified based on their domain families and domains from the same family tend to interact [48–50]. For the human data, due to the extreme sparsity of the human PPI network, limited golden standard interactions can be included in the validation set to help optimize weights, which makes the weight optimization problem more challenging, especially for the sampling method. Although the result of human data that shown in Fig. 9 is not good as that of the yeast data, these two methods also produce quite consistent distribution, and K_{SN} is the most predominant contributor. Although the true strength of our method lies in integrating multiple heterogeneous data for PPI network inference, the optimal weights can serve as a guidance to select most relevant features when time and resources are limited.

Conclusion

In this work we developed a novel and fast optimization method using linear programming to integrate multiple heterogeneous data for PPI inference problem. The proposed method, verified with synthetic data and tested

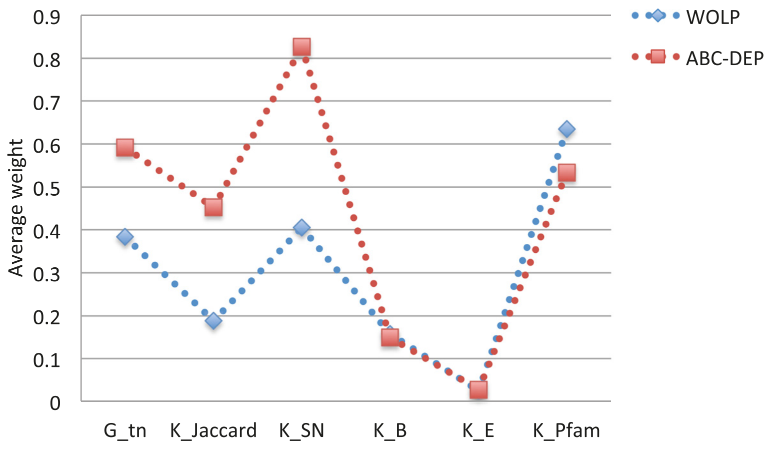

Fig. 8 Yeast: comparison of average weights learned by WOLP and ABC-DEP sampling method

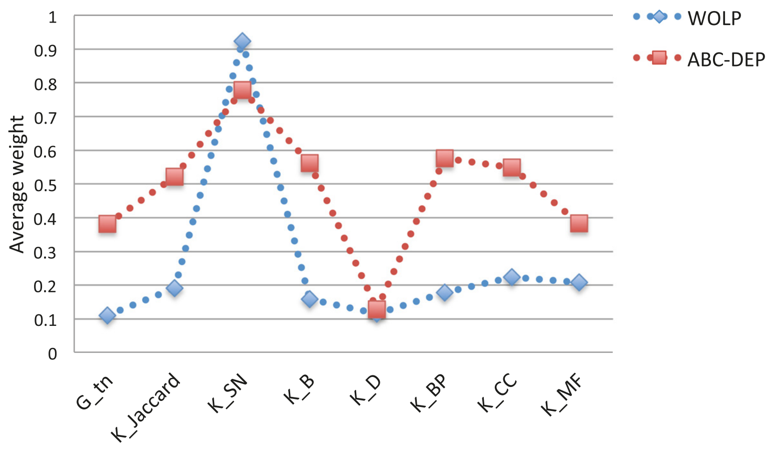

Fig. 9 Human: comparison of average weights learned by WOLP and ABC-DEP sampling method

with DIP yeast PPI network and PrePPI high-confidence human PPI network , enables quick and accurate inference of PPI networks from topological and genomic feature kernels in an optimized integrative way. Compared to the baseline (G_{tn} and *EW-K*), our WOLP method achieved performance improvement in PPI prediction with over 19 % higher AUC on yeast data and 11 % higher AUC on human data, and this margin is maintained even when the control methods use a significantly larger training set. We also demonstrated that by integrating topological and genomic features into regularized Laplacian kernel, the method avoids the short-range problem encountered by random-walk based methods – namely the inference becomes less reliable for nodes that are far from the start node of the random walk, and shows obvious improvements on predicting faraway interactions; The weights learned by our WOLP are highly consistent with the weights learned by sampling based method, which can provide insights into the relations between PPIs and various similarity features of protein pairs, thereby helping us make good use of these features. Moreover, we further demonstrated those relations are also maintained when the golden standard network (largest connected component) scale up to the original PPI network that consists of disconnected components. That is to say, the weights learned based on the connected training subnetwork of the largest connected component can also help to detect interactions for the originally disconnected PPI networks effectively and accurately. As more features with respect to proteins are collected from various -omics studies, they can be used to characterize protein pairs in terms of feature kernels from different perspectives. Thus we believe that our method can provide us a quick and accurate way to fuse various feature kernels from heterogeneous data, thereby improving PPI prediction.

Declarations
Publication of this article is funded by Delaware INBRE program, with grant from the National Institute of General Medical Sciences-NIGMS (8 P20 GM103446-12) from the National Institutes of Health. This article has been published as part of BMC Systems Biology Vol 10 Suppl 2 2016: Selected articles from the IEEE International Conference on Bioinformatics and Biomedicine 2015: systems biology. The full contents of the supplement are available online at http://bmcsystbiol.biomedcentral.com/articles/supplements/volume-10-supplement-2.

Authors' contributions
LH designed the algorithm and experiments, and performed all calculations and analyses. LL and CHW aided in interpretation of the data and preparation of the manuscript. LH wrote the manuscript, LL and CHW revised it. LL and CHW conceived of this study. All authors have read and approved this manuscript.

Competing interests
The authors declare that they have no competing interests.

Author details
[1]Department of Computer and Information Sciences, University of Delaware, 18 Amstel Avenue, 19716, Newark, Delaware, USA. [2]Center for Bioinformatics and Computational Biology, University of Delaware, 15 Innovation Way, 19711, Newark, Delaware, USA.

Published: 1 August 2016

References
1. Kuchaiev O, Rašajski M, Higham DJ, Pržulj N. Geometric de-noising of protein-protein interaction networks. PLoS Comput Biol. 2009;5(8):1000454.
2. Murakami Y, Mizuguchi K. Homology-based prediction of interactions between proteins using averaged one-dependence estimators. BMC Bioinformatics. 2014;15(1):213.
3. Salwinski L, Eisenberg D. Computational methods of analysis of protein–protein interactions. Curr Opin Struct Biol. 2003;13(3):377–82.
4. Craig R, Liao L. Phylogenetic tree information aids supervised learning for predicting protein-protein interaction based on distance matrices. BMC Bioinformatics. 2007;8(1):6.
5. Gonzalez A, Liao L. Predicting domain-domain interaction based on domain profiles with feature selection and support vector machines. BMC Bioinformatics. 2010;11(1):537.
6. Zhang QC, Petrey D, Deng L, Qiang L, Shi Y, Thu CA, Bisikirska B, Lefebvre C, Accili D, Hunter T, Maniatis T, Califano A, Honig B. Structure-based prediction of protein-protein interactions on a genome-wide scale. Nature. 2012;490(7421):556–60.

7. Singh R, Park D, Xu J, Hosur R, Berger B. Struct2net: a web service to predict protein–protein interactions using a structure-based approach. Nucleic Acids Res. 2010;38(suppl 2):508–15.
8. Deng Y, Gao L, Wang B. ppipre: predicting protein-protein interactions by combining heterogeneous features. BMC Syst Biol. 2013;7(Suppl 2):8.
9. Sun J, Sun Y, Ding G, Liu Q, Wang C, He Y, Shi T, Li Y, Zhao Z. Inpreppi: an integrated evaluation method based on genomic context for predicting protein-protein interactions in prokaryotic genomes. BMC Bioinformatics. 2007;8(1):414.
10. Cho YR, Mina M, Lu Y, Kwon N, Guzzi P. M-finder: Uncovering functionally associated proteins from interactome data integrated with go annotations. Proteome Sci. 2013;11(Suppl 1):3.
11. Jung SH, Jang WH, Han DS. A computational model for predicting protein interactions based on multidomain collaboration. IEEE/ACM Trans Comput Biol Bioinformatics. 2012;9(4):1081–90.
12. Chen HH, Gou L, Zhang XL, Giles CL. Discovering missing links in networks using vertex similarity measures. In: Proceedings of the 27th Annual ACM Symposium on Applied Computing. SAC '12. New York, NY, USA: ACM; 2012. p. 138–43.
13. Lü L, Zhou T. Link prediction in complex networks: A survey. Physica A. 2011;390(6):11501170.
14. Lei C, Ruan J. A novel link prediction algorithm for reconstructing protein-protein interaction networks by topological similarity. Bioinformatics. 2012. doi:10.1093/bioinformatics/bts688. http://bioinformatics.oxfordjournals.org/content/early/2012/12/11/bioinformatics.bts688.full.pdf+html.
15. Pržulj N. Protein-protein interactions: Making sense of networks via graph-theoretic modeling. BioEssays. 2011;33(2):115–23.
16. Page L, Brin S, Motwani R, Winograd T. The pagerank citation ranking: Bringing order to the web. Technical Report 1999-66, Stanford InfoLab (November 1999). Previous number = SIDL-WP-1999-0120. http://ilpubs.stanford.edu:8090/422/.
17. Tong H, Faloutsos C, Pan JY. Random walk with restart: fast solutions and applications. Knowl Inform Syst. 2008;14(3):327–46.
18. Li RH, Yu JX, Liu J. Link prediction: The power of maximal entropy random walk. In: Proceedings of the 20th ACM International Conference on Information and Knowledge Management. CIKM '11. New York, NY, USA: ACM; 2011. p. 1147–1156. doi:10.1145/2063576.2063741. http://doi.acm.org/10.1145/2063576.2063741.
19. Backstrom L, Leskovec J. Supervised random walks: Predicting and recommending links in social networks. In: Proceedings of the Fourth ACM International Conference on Web Search and Data Mining. WSDM '11. New York, NY, USA: ACM; 2011. p. 635–44.
20. Fouss F, Francoisse K, Yen L, Pirotte A, Saerens M. An experimental investigation of kernels on graphs for collaborative recommendation and semisupervised classification. Neural Netw. 2012;31(0):53–72.
21. Cannistraci CV, Alanis-Lobato G, Ravasi T. Minimum curvilinearity to enhance topological prediction of protein interactions by network embedding. Bioinformatics. 2013;29(13):199–209.
22. Symeonidis P, Iakovidou N, Mantas N, Manolopoulos Y. From biological to social networks: Link prediction based on multi-way spectral clustering. Data Knowl Eng. 2013;87(0):226–42.
23. Wang H, Huang H, Ding C, Nie F. Predicting protein–protein interactions from multimodal biological data sources via nonnegative matrix tri-factorization. J Comput Biol. 2013;20(4):344–58. doi:10.1089/cmb.2012.0273.
24. Menon AK, Elkan C. Link prediction via matrix factorization. In: Proceedings of the 2011 European Conference on Machine Learning and Knowledge Discovery in Databases - Volume Part II. ECML PKDD'11. Berlin, Heidelberg: Springer; 2011. p. 437–52.
25. Yamanishi Y, Vert JP, Kanehisa M. Protein network inference from multiple genomic data: a supervised approach. Bioinformatics. 2004;20(suppl 1):363–70.
26. Huang L, Liao L, Wu CH. Inference of protein-protein interaction networks from multiple heterogeneous data. EURASIP J Bioinformatics Syst Biol. 2016;2016(1):1–9. doi:10.1186/s13637-016-0040-2.
27. Huang L, Liao L, Wu CH. Protein-protein interaction network inference from multiple kernels with optimization based on random walk by linear programming. In: Proceedings of 2015 IEEE International Conference on Bioinformatics and Biomedicine (BIBM). Washington DC, USA: IEEE computer society; 2015. p. 201–7.
28. Ito T, Shimbo M, Kudo T, Matsumoto Y. Application of kernels to link analysis. In: Proceedings of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery in Data Mining. KDD '05. New York, NY, USA: ACM; 2005. p. 586–92.
29. Smola AJ, Kondor R. Kernels and Regularization on Graphs In: Schölkopf B, Warmuth MK, editors. Learning Theory and Kernel Machines: 16th Annual Conference on Learning Theory and 7th Kernel Workshop, COLT/Kernel 2003, Washington, DC, USA, August 24–27, 2003. Proceedings. Berlin, Heidelberg: Springer; 2003. p. 144–58.
30. Mantrach A, van Zeebroeck N, Francq P, Shimbo M, Bersini H, Saerens M. Semi-supervised classification and betweenness computation on large, sparse, directed graphs. Pattern Recognit. 2011;44(6):1212–24.
31. Pan JY, Yang HJ, Faloutsos C, Duygulu P. Automatic multimedia cross-modal correlation discovery. In: Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. KDD '04. New York, NY, USA: ACM; 2004. p. 653–8.
32. Baker J. An algorithm for the location of transition states. J Comput Chem. 1986;7(4):385–95. doi:10.1002/jcc.540070402.
33. Paige CC, Saunders MA. Lsqr: An algorithm for sparse linear equations and sparse least squares. ACM Trans Math Softw. 1982;8(1):43–71.
34. Fong DC-L, Saunders M. Lsmr: An iterative algorithm for sparse least-squares problems. SIAM J Sci Comput. 2011;33(5):2950–71.
35. Barabási AL. Scale-free networks: A decade and beyond. Science. 2009;325(5939):412–3. doi:10.1126/science.1173299. http://science.sciencemag.org/content/325/5939/412.full.pdf.
36. Kumar R, Raghavan P, Rajagopalan S, Sivakumar D, Tomkins A, Upfal E. Stochastic models for the web graph. In: Proceedings of the 41st Annual Symposium on Foundations of Computer Science. FOCS '00. Washington, DC, USA: IEEE Computer Society; 2000. p. 57.
37. Salwinski L, Miller CS, Smith AJ, Pettit FK, Bowie JU, Eisenberg D. The database of interacting proteins: 2004 update. Nucleic Acids Res. 2004;32(90001):449–51.
38. Christopher D, Manning HS. Prabhakar Raghavan: Introduction to Information Retrieval. New York, USA: Cambridge University Press; 2008.
39. Zhang QC, Petrey D, Garzón JI, Deng L, Honig B. Preppi: a structure-informed database of protein–protein interactions. Nucleic Acids Res. 2013;41(D1):828–33. doi:10.1093/nar/gks1231. http://nar.oxfordjournals.org/content/41/D1/D828.full.pdf+html.
40. Lanckriet GRG, De Bie T, Cristianini N, Jordan MI, Noble WS. A statistical framework for genomic data fusion. Bioinformatics. 2004;20(16):2626–635.
41. Jaccard P. Étude comparative de la distribution florale dans une portion des Alpes et des Jura. Bulletin del la Société Vaudoise des Sciences Naturelles. 1901;37:547–79.
42. Altschul SF, Gish W, Miller W, Myers EW, Lipman DJ. Basic local alignment search tool. J Mol Biol. 1990;215(3):403–10.
43. Sonnhammer ELL, Eddy SR, Durbin R. Pfam: A comprehensive database of protein domain families based on seed alignments. Proteins: Struct Funct Bioinformatics. 1997;28(3):405–20.
44. Song N, Joseph JM, Davis GB, Durand D. Sequence similarity network reveals common ancestry of multidomain proteins. PLoS Comput Biol. 2008;4(5):1–19. doi:10.1371/journal.pcbi.1000063.
45. Resnik P. Using information content to evaluate semantic similarity in a taxonomy. In: Proceedings of the 14th International Joint Conference on Artificial Intelligence - Volume 1. IJCAI'95. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc; 1995. p. 448–53. http://dl.acm.org/citation.cfm?id=1625855.1625914.
46. Huang L, Liao L, Wu CH. Evolutionary model selection and parameter estimation for protein-protein interaction network based on differential evolution algorithm. Comput Biol Bioinformatics, IEEE/ACM Trans. 2015;12(3):622–31.
47. Deng M, Mehta S, Sun F, Chen T. Inferring domain–domain interactions from protein–protein interactions. Genome Res. 2002;12(10):1540–8.
48. Itzhaki Z, Akiva E, Altuvia Y, Margalit H. Evolutionary conservation of domain-domain interactions. Genome Biol. 2006;7(12):125.
49. Park J, Lappe M, Teichmann SA. Mapping protein family interactions: intramolecular and intermolecular protein family interaction repertoires in the {PDB} and yeast1. J Mol Biol. 2001;307(3):929–38.
50. Betel D, Isserlin R, Hogue CWV. Analysis of domain correlations in yeast protein complexes. Bioinformatics. 2004;20(suppl 1):55–62.