Active Learning for Protein Function Prediction in Protein-Protein Interaction Networks

Wei Xiong[1], Luyu Xie[1], Jihong Guan[2], and Shuigeng Zhou[1]

[1] School of Computer Science, and Shanghai Key Lab of Intelligent Information Processing, Fudan University, Shanghai, China
{wxiong,10300240052,sgzhou}@fudan.edu.cn
[2] Department of Computer Science & Technology, Tongji University, Shanghai, China
jhguan@tongji.edu.cn

Abstract. The high-throughput technologies have led to vast amounts of protein-protein interaction (PPI) data, and a number of approaches based on PPI networks have been proposed for protein function prediction. However, these approaches do not work well if annotated proteins are scarce in the networks. To address this issue, we propose an active learning based approach that uses graph-based centrality metrics to select proper candidates for labeling. We first cluster a PPI network by using the spectral clustering algorithm and select some proper candidates for labeling within each cluster, and then apply a collective classification algorithm to predict protein function based on these annotated proteins. Experiments over two real datasets demonstrate that the active learning based approach achieves better prediction performance by choosing more informative proteins for labeling. Experimental results also validate that betweenness centrality is more effective than degree centrality and closeness centrality in most cases.

Keywords: Protein function prediction, Active learning, Collective classification, Protein-protein interaction network.

1 Introduction

In recent years, the rapid development of high-throughput experimental biology has led to huge amounts of unannotated protein sequences. Meanwhile, experimentally determining protein function is expensive and time-consuming. So there is a wider and wider gap between the pace of discovery of protein sequences and that of functional annotation of known proteins. Therefore, protein function prediction has been a fundamental challenge of biology in the post-genomic era. Although many efforts have been made to solve this problem, the proportion of annotated proteins is still very low. Among the 13 million protein sequences, there are only 1% sequences having experimentally-validated annotations [1]. Even for the most well-studied model organisms, taking yeast as an example, approximately one-fourth of the proteins have no annotated functions [2].

Due to high cost and long duration of experimentally annotating protein function, there is increasing research on using computational approaches to predict

A. Ngom et al. (Eds.): PRIB 2013, LNBI 7986, pp. 172–183, 2013.

protein function. The recent advent of high-throughput experimental biology has generated vast amounts of protein-protein interaction (PPI) data, which are represented as networks, where a node corresponds to a protein and an edge corresponds to an interaction between a pair of proteins. Thus, a number of prediction approaches based on PPI networks have been proposed. These approaches make use of the observation that proteins with short distance to each other in a PPI network are more likely to have similar functions.

However, current network-based approaches will fail to work when annotated proteins are scarce. To address this issue, in this paper we propose an active learning [3] based approach that uses graph-based centrality metrics to select good candidates for labeling. Our approach consists of two steps: we first cluster a PPI network by using spectral clustering algorithm and select proper candidates for labeling within each cluster, and then apply a collective classification algorithm to predict protein function based on these annotated proteins. To the best of our knowledge, this is the first study where active learning is employed to predict protein functions in PPI networks. The key idea behind active learning is that a machine learning algorithm can achieve higher accuracy with fewer training labels if it is allowed to choose the proper data for labeling from which it learns. Therefore, we let the learning algorithm pick a set of unannotated proteins to be labeled by an oracle (*i.e.*, a lab experiment), which will then be used as the labeled data set. In other words, we let the learning algorithm tell us which proteins to label, rather than select them randomly.

We conduct experiments on the *S.cerevisiae* and *M.musculus* functional annotation datasets, The experimental results show that the active learning based approach achieves better prediction performance by choosing more informative proteins for labeling. Experimental results also validate that betweenness centrality is more effective than degree centrality and closeness centrality in most cases. The rest of this paper is organized as follows: Section 2 presents our approach, Section 3 gives the experimental evaluation results, Section 4 describes related work, and finally Section 5 concludes the paper.

2 Method

2.1 Notation and Problem Definition

In this paper, a PPI network is represented as an indirected graph $G = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = (V_1, ..., V_n)$ is a set of n vertices and $\mathcal{E}$ is a set of weighted edges. Each vertex $V_i \in \mathcal{V}$ represents a protein and each edge $E_{i,j} \in \mathcal{E}$ represents an interaction between proteins V_i and V_j. Edge $E_{i,j}$ is labeled with a weight $w_{i,j}$ that indicates the interaction confidence. $\mathcal{F} = (F_1, ..., F_m)$ is the set of m functions assigned to the proteins, and each vertex $V_i \in \mathcal{V}$ is assigned with at least one function. The functions of vertex $V_i \in \mathcal{V}$ are denoted by

$$\Phi(V_i) = [f_{i,1}, f_{i,2}, ..., f_{i,j}, ..., f_{i,m}]^T \quad (1)$$

where

$$\begin{cases} f_{i,j} = 1, & if\ V_i\ has\ the\ function\ F_j; \\ f_{i,j} = 0, & otherwise. \end{cases} \tag{2}$$

$\mathcal{V}$ can further be divided into two sets: $\mathcal{X}$ — the labeled vertices and $\mathcal{Y}$ — the vertices whose functions need to be determined.

In this paper, our goal is to label as few vertices $\{Y_i\} \subset \mathcal{Y}$ as possible with at least one of the functions in $\mathcal{F}$ based on the available information of the corresponding PPI network, so that the labeled vertices $\{Y_i\}$ and $\mathcal{X}$ together constitute the training set, which can be used to train an as good as possible classifier. Here, active learning is used for data selection to be labeled, the collective classification method is employed for classifier training.

2.2 Active Learning Strategies for Protein Function Prediction

As we point out above, experimentally annotating protein function is expensive in terms of cost and effort, and current network-based approaches do not work well if annotated proteins are scarce. Therefore, strategies that minimize the amount of labeled data required in the supervised learning task would be useful. Active learning attempts to overcome the labeling bottleneck by asking queries in the form of unlabeled instances to be labeled by an oracle (*i.e.*, a lab experiment). In this way, the active learner aims to achieve high accuracy using as few labeled instances as possible, thereby minimizing the cost of obtaining labeled data. The key idea behind active learning is that a machine learning algorithm can achieve higher accuracy with fewer training labels if it is allowed to choose the most proper data for labeling from which it learns.

In this study, the PPI network is represented as a graph, so it seems reasonable that we leverage graph structure to identify the nodes (proteins) in the graph that are important (central) for labeling. That is, we expect that such central nodes are proper candidates to label. Furthermore, we also note that nodes of the same class tend to cluster together in the PPI network. This suggests that clustering the graph and then finding central nodes in the clusters my be a good way to find proper candidates. Therefore, we explore the spectral clustering algorithm to cluster the PPI network and then leverage graph-based centrality metrics to select central nodes in the clusters to label.

Under the active learning framework, there is a small set of labeled data and a large pool of unlabeled data available. A fixed number M of labels (usually called the *labeling budget*) is requested. Suppose that the selected nodes are distributed across the clusters of the PPI network, in proportion to the size of the cluster. Let n_i be the number of nodes in cluster C_i and N be total number of nodes in the PPI network. Then, m_i, the number of nodes to be selected from cluster C_i is given by

$$m_i = M * n_i/N \quad and \quad M = \sum_{i=1}^{K} m_i. \tag{3}$$

In each cluster C_i, m_i central nodes are selected to label. In what follows, we describe and discuss the spectral clustering algorithm and graph-based centrality metrics in detail.

Spectral Clustering Algorithm. Spectral clustering [4] is one of the most popular modern clustering algorithms. It is simple to implement, can be solved efficiently by standard linear algebra software, and very often outperforms traditional clustering algorithms such as the k-means algorithm. Detail description of the spectral clustering algorithm is presented as follows.

Given a PPI network, let $W \in \mathbb{R}^{n\times n}$ be its weighted adjacency matrix, $W_{ii} = 0$ and $W_{ij} = 0$ if the vertices V_i and V_j are not connected by an edge. The degree of a vertex $V_i \in \mathcal{V}$ is defined as

$$d_i = \sum_{j=1}^{n} W_{ij}. \tag{4}$$

Note that this sum only performs over all vertices adjacent to V_i, as for all other vertices V_j the weight W_{ij} is 0. The degree matrix D is defined as the diagonal matrix with the degrees $d_1, ..., d_n$ on the diagonal. The unnormalized graph Laplacian matrix is defined as

$$L = D - W. \tag{5}$$

Next, we compute the first k eigenvectors $u_1, ..., u_k$ of L, and let $U \in \mathbb{R}^{n\times k}$ be the matrix containing the vectors $u_1, ..., u_k$ as columns. For $i = 1, ..., n$, let $y_i \in \mathbb{R}^k$ be the vector corresponding to the i-th row of U. Finally, we cluster the points y_i in $\mathbb{R}^k$ with the K-means algorithm into clusters $C_1, ..., C_k$.

Graph-Based Centrality Metrics. In this study, we consider three kinds of graph-based centrality metrics for active learning, including degree centrality, closeness centrality and betweenness centrality.

Degree centrality. Graph degree centrality is perhaps the simplest measure of centrality, it is defined as the number of links incident upon a vertex (*i.e.*, the degree of a vertex). So graph degree centrality of a vertex v is defined as follows:

$$C_D(v) = \deg(v). \tag{6}$$

Closeness centrality [5]. In connected graph there is a natural distance metric between all pairs of vertices, defined by the length of their shortest paths. The *farness* of a vertex is defined as the sum of its distances to all other vertices, and its *closeness* is defined as the inverse of the farness. Thus, the more central a vertex is the smaller its total distance to all other vertices. *Graph closeness centrality* measures how close a vertex is to all other vertices in the graph, it is defined as the inverse of the total distance to all other vertices:

$$C_C(v) = \frac{1}{\sum\limits_{t\in V} d(v,t)}. \tag{7}$$

where $d(v,t)$ is the distance from vertex v to vertex t in the graph. In unweighted graph, the distance is defined in terms of the number of edges that connect two vertices. And in weighted graph, we define the distance as the sum of weights of the edges that connect two vertices.

Betweenness centrality [6]. Graph betweenness centrality is perhaps one of the most prominent measures of centrality, it quantifies the number of times a vertex acts as a bridge along the shortest path between two other vertices. That is, vertices that have a high probability to occur on a randomly chosen shortest path between two randomly chosen vertices have a high betweenness. Graph betweenness centrality of a vertex v is evaluated as follows:

$$B(v) = \sum_{s \neq v \neq t} \frac{\sigma_{st}(v)}{\sigma_{st}}. \tag{8}$$

Above, σ_{st} is the total number of shortest paths from vertex s to vertex t and $\sigma_{st}(v)$ is the number of those paths that pass through v. As with closeness, we compute all shortest paths to get the centrality measure for all vertices.

2.3 Collective Classification: The Gibbs Sampling Approach

In this study, Gibbs sampling (GS) [7] is applied to predicting protein function. GS is one of the most commonly used collective classification algorithm that aims at finding the best label estimate for each un-annotated vertex $Y_i \in \mathcal{Y}$ by sampling each vertex label iteratively. Our approach consists of two steps: *bootstrapping* and *iterative classification*, the pseudo-code is illustrated in Algorithm 1. The details of the algorithm are presented in the following subsections.

Bootstrapping. According the observation that proteins with shorter distance to each other in the network are more likely to have similar functions, we use weighted voting to predict an initial functional probability distribution for a query protein (*i.e.* an un-annotated protein).

Given a query protein V_x, which has N_x neighbors, these corresponding edge weights can be represented as the vector as follows:

$$\mathcal{N}_x^w = [w_{x1}, w_{x2}, ..., w_{xi}, ..., w_{xN_x}]. \tag{9}$$

Then the probability of V_x having the j-th function F_j is computed as follows:

$$P_x^j = \frac{1}{Z_x^w} \sum_{i=1}^{N_x} w_{x,i} f_{i,j} \tag{10}$$

where Z_x^w is the normalizer:

$$Z_x^w = \sum_{j=1}^{m} \sum_{i=1}^{N_x} w_{x,i} f_{i,j}. \tag{11}$$

The larger the value of P_x^j is, the more likely protein V_x has the j-th function F_j. The initial functional probability distribution for query protein V_x is represented as an m-dimensional vector:

$$\boldsymbol{a_x} = [P_x^1, P_x^2, ..., P_x^m]. \tag{12}$$

Note that when predicting the functions of the query protein V_x, we consider only its labeled neighbor proteins. That is why we use $\mathcal{X} \cap \mathcal{N}_x^w$ in Algorithm 1 (Line 3), because unlabeled neighbor proteins can not be exploited in the bootstrapping step. This process is implemented in Alg. 1 from Line 2 to 4.

Iterative Classification. Iterative classification is performed in two steps:

- First, there is a fixed number B of iterations known as "burn-in" period. In this period, we only update $\boldsymbol{a_x}$ using weighted voting in each iteration. Corresponding codes of this period in Algorithm 1 are from Line 6 to 10.
- Second, there is a sampling period. In this period, not only do we update $\boldsymbol{a_x}$ in each iteration but we also maintain the count statistics as to how many times we have sampled the j-th function F_j for protein V_x. Codes corresponding to this period in Algorithm 1 are from Line 12 to 20.

Note that each protein can belong to one or more functions, therefore, we formulate protein functional annotation as a multiclass classification problem. More formally, the most likely function of protein V_x is computed like this:

$$b_x^1 = argmax_{j \in [1,m]} P_x^j \tag{13}$$

where b_x^1 is the value of j that maximizes the value of P_x^j, called the 1st-rank result. The second most likely function is denoted by b_x^2, called the 2nd-rank result. The third most likely function is denoted by b_x^3, called the 3rd-rank result, and so forth. In case that more than one element P_x^j has the same value, their ranks will be assigned randomly. For each protein V_x in the i-th iteration, an m-dimensional vector $\boldsymbol{b_{xi}}$ is created to record the ranking result:

$$\boldsymbol{b_{xi}} = [b_{xi}^1, b_{xi}^2, ..., b_{xi}^m]. \tag{14}$$

When the pre-specified number (threshold) S of iterations have elapsed, we get a matrix M_x with S rows and m columns for query protein V_x:

$$M_x = [\boldsymbol{b_{x1}}, \boldsymbol{b_{x2}}, ..., \boldsymbol{b_{xS}}]^T. \tag{15}$$

In the first column of the matrix M_x, the most frequently sampled function c_x^1 is regard as the first rank predicted function for the query protein V_x. In the second column of the matrix M_x, the most frequently sampled function c_x^2 excluding c_x^1 is regard as the second rank predicted function. In the third column of the matrix M_x, the most frequently sampled function c_x^3 excluding c_x^1 and c_x^2 is regard as the third rank predicted function, and so forth. Finally, we get an m-dimensional vector $\boldsymbol{c_x}$ for query protein V_x:

$$\boldsymbol{c_x} = [c_x^1, c_x^2, ..., c_x^m]. \tag{16}$$

Algorithm 1. Gibbs sampling based collective classification for protein function prediction in PPI networks.

```
// bootstrapping
for each query protein V_x do
    compute the initial a_x using X ∩ N_x^w
end for
// burn-in period
for i=1 to B do
    for each query protein V_x do
        update a_x using current assignments to N_x^w
    end for
end for
// sampling period
for i=1 to S do
    for each query protein V_x do
        update a_x using current assignments to N_x^w
        create b_xi to record the m-rank result
    end for
end for
for each query protein V_x do
    calculate the final result c_x based on matrix M_x
end for
```

3 Experimental Evaluation

3.1 Interaction and Annotation Data

We evaluate the performance of our approach with two functional annotation datasets. These two datasets are both based on Functional Catalogue (FunCat) annotation scheme [8] taken from Munich Information Center for Protein Sequences (MIPS)[1]. FunCat is organized as a hierarchically structured annotation system and consists of 28 main functional categories. FunCat annotations for S.cerevisiae are downloaded from Comprehensive Yeast Genome Database (CYGD) [9]. CYGD is a frequently used public resource for yeast related information. There are a total of 6168 proteins in the dataset, of which 4774 are annotated. These proteins belong to 17 functional categories. The second functional annotation dataset is Mouse functional Genome Database (MfunGD) [10]. MfunGD provides a resource for annotated mouse proteins and comprises 17643 annotated proteins. These annotated proteins belong to 24 functional categories.

In this study, protein interaction data is download from the STRING database [11], which is an integrated protein interaction database containing known and predicted protein interactions. These interactions were mainly derived from four data sources: genomic context, high-throughput experiments, conserved coexpression and previous knowledge. The most recent version of STRING covers about 5.2 million proteins from 1133 organisms.

We construct two protein interaction networks (one for S.cerevisiae and another for M.musculus) where a node corresponds to a protein and a weighted edge corresponds to an interaction between two proteins. Each node is assigned with at least one functional category and each edge is labeled with a weight

[1] `http://www.helmholtz-muenchen.de/en/ibis`

based on the interaction confidence. Proteins without interaction and annotation data are deleted. As a result, in the S.cerevisiae interaction network, there are 4687 proteins and 388846 interactions, and in the M.musculus interaction network there are 14277 proteins and 832128 interactions.

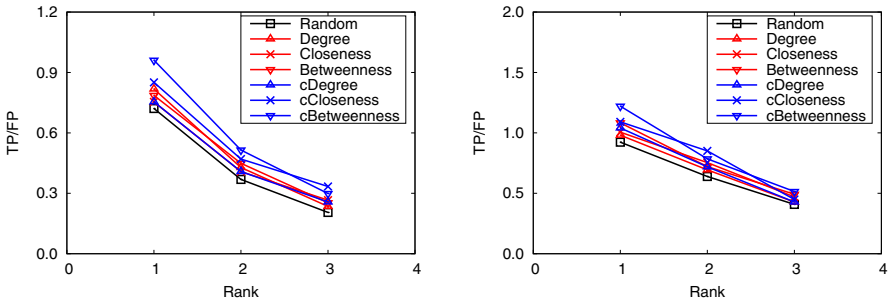
3.2 Experimental Methodology

In the experiments, we compare the performance of three kinds of data selection strategies. The first is *random data selection strategy* (baseline), which randomly selects nodes in the PPI network to label. The second is *graph structure based data selection strategy*, which leverages graph-based centrality metrics to select central nodes in the PPI network to label. The last is our proposed strategy, which first uses the spectral clustering algorithm to cluster the PPI network and then leverages graph-based centrality metrics to select central nodes in each cluster to label. Note that there are three kinds of graph-based centrality metrics (*degree centrality*, *closeness centrality* and *betweenness centrality*). Thus, in fact, we compare the performance of seven kinds of data selection strategies.

We set the proportion of annotated proteins to 5%, and for each data selection strategy, we run 20 experiments and report the average performance. In spectral clustering, we set the number of clusters K to 30 and 50 for S.cerevisiae and M.musculus respectively, this value is chosen by trial and error. As for collective classification, we set the burn-in period to 10 iterations (*i.e.* B=10) and collect 50 samples (*i.e.* S=50) in the sampling period. Since protein functional annotation is a multiclass classification problem, all competing methods calculate an m-rank predicted function vector $\boldsymbol{c_x}$ for each query protein V_x. In this setup, we define the i-th rank overall *true positive* (TP) as the number of proteins whose i-th rank predicted function c_x^i is one of the true functions of the protein V_x and the i-th rank overall *false positive* (FP) as the number of proteins whose i-th rank predicted function c_x^i is not one of the true functions of the protein V_x. Accordingly, as in [12] we use the ratio of TP/FP as the measure of performance, which depicts the relative magnitude between TP and FP.

3.3 Experimental Results

In the experiments, there are two PPI networks (corresponding to S.cerevisiae and M.musculus). For *S.cerevisiae*, the average number of functions that each protein has is 2.13, so we consider only the top 3 ($3=\lfloor 2.13 \rfloor + 1$) predictions. Fig. 1(a) shows the performance comparison of seven kinds of data selection strategies for the top-3 predictions. And for *M.musculus*, because the average number of functions that a protein possesses is 2.58, we consider also only the first 3 ($3=\lfloor 2.58 \rfloor + 1$) predictions. The results are shown in Fig. 1(b). In Fig. 1, for simplification, *Random* indicates the random data selection strategy; *Degree/Closeness/Betweeness* means the graph structure based strategy with the metric of *degree centrality/closeness centrality/betweenness centrality*; And *cDegree/cCloseness/cBetweeness* is our strategy with clustering plus the metric of *degree centrality/closeness centrality/betweenness centrality*.



(a) *S.cerevisiae* ($K = 30$) (b) *M.musculus* ($K = 50$)

Fig. 1. Performance comparison of seven kinds of data selection strategies

It can be seen from Fig. 1 that all the six graph structure based data selection strategies obtain more accurate predictions than the random data selection strategy, due to using graph-based centrality metrics to select central nodes in the PPI network to label. The results clearly show that the active learning based approach achieve a better prediction performance than the baseline approach. This means that given a similar number of labeled proteins, our active learning approach can achieve outstanding performance by choosing the most informative proteins to be labeled. We also notice that our proposed data selection strategies outperform other three graph structure based data selection strategies. As we explore the spectral clustering algorithm to cluster the PPI network before selecting protein candidates for labeling, this result shows that clustering is an important pre-processing step in active learning algorithm. The reason is that selecting candidates across clusters will make the distribution of selected candidates over different classes more balanced.

The experimental results also validate that using betweenness centrality as the graph-based centrality metric generally can achieve the best performance in most cases, which means betweenness centrality is more effective than degree centrality and closeness centrality. In addition, it is worth noting that higher rank functions are predicted better than lower ones, implying that the protein functions are well ranked by our approach.

4 Related Work

In a recent review [2], the existing network-based methods for protein function prediction are categorized into two main groups: direct methods and module-assisted methods. Direct methods propagate functional information through a PPI network and use the propagated information for functional annotation, examples include neighborhood counting methods and graph theoretic methods.

The majority method [13] and the indirect neighbors method [14] are two typical direct network-based approaches. Majority method [13] is the simplest direct method, it utilizes the biological hypothesis that interacting proteins probably

have similar functions, it ranks each candidate function based on the function's occurrences in the immediate neighbors. Indirect neighbors method [14] assumes that proteins interacting with the same proteins may also have some similar functions, It exploits both indirect and immediate neighbors to rank each candidate function. Functional flow method [15] is a graph theoretic method, it simulates a discrete-time flow of functions from all proteins. At every time step, the function weight transferred along an edge is proportional to the edge's weight and the direction of transfer is determined by the functional gradient.

Module-assisted methods first identify functional modules in the network and then assign functions to all the proteins in each module, representatives are hierarchical clustering-based method and graph clustering method. A key problem of this kind of methods is how to define the similarity between two proteins. Arnau *et al.* [16] used the shortest path between proteins as a distance measure and apply hierarchical clustering to detecting functional modules. Up to now, numerous graph-clustering algorithms have been applied to detecting functional modules, such as clique percolation [17] and edge-betweenness [18] clustering.

Additionally, Chua et al [19] presented a simple framework for integrating large amount of diverse information for protein function prediction by using simple weighting strategies and a local prediction method. Hu et al [20] hybridized the PPI information and the biochemical/physicochemical features of protein sequences to predict protein function. The prediction is carried out as follows: if the query protein has PPI information, the network-based method is applied; Otherwise, the hybrid-property based method is employed.

Active learning [3] is a form of supervised machine learning in which a learning algorithm is able to interactively query the user (or some other information source) to obtain the desired outputs at some unlabeled data points. The key issue is to design the query strategy such that as few data points as possible are queried to achieve as large learning performance improvement as possible. The simplest and most commonly used query strategy is *uncertainty sampling* [21]. In this framework, an active learner queries the instance that the classifier is most uncertain. This strategy is often straightforward for probabilistic learning models. The *query-by-committee* (QBC) [22] strategy maintains a committee, each committee member is allowed to vote on the labelings of query candidates, the most informative query is considered to be the instance about which they most disagree. The fundamental premise behind the QBC strategy is minimizing the version space. The *expected model change* [23] strategy uses a decision-theoretic approach, it selects the instance that would impart the greatest change to the current model. The *expected error reduction* [24] strategy aims to measure not how much the model is likely to change, but how much its generalization error is likely to be reduced. It selects the instance that offer maximal expected error reduction to the classifier. The *density-weighted* [25] strategy suggests that the informative instances should not only be those which are uncertain, but also those which are representative of the underlying distribution.

Active learning has been applied to some bioinformatic problems, such as cancer classification [26], DNA microarray data analysis [27] and protein-protein

interaction prediction [28] etc. However, to the best of our knowledge, there is no work on active learning for protein function prediction in the literature.

5 Conclusion

In this study, we proposed an active learning based approach to conducting protein function prediction based on PPI networks. It first clusters a PPI network by using the spectral clustering algorithm and select some appropriate candidates for labeling within each cluster by using graph-based centrality metrics, and then applies a collective classification algorithm to predict protein function based on these annotated proteins. We conducted experiments on two real, publicly available PPI datasets. The experimental results show that the proposed active learning based approach, by choosing more informative proteins for labeling, achieves obviously better prediction performance than the baseline approach. Furthermore, betweenness centrality is more effective than degree centrality and closeness centrality in most cases.

Acknowledgments. This study was supported by China 863 Program (grant No. 2012AA020403), and NSFC (grants No. 61173118 and No. 61272380). Jihong Guan was also supported by the "Shuguang Scholar" Program of Shanghai Education Foundation.

References

1. Barrell, D., Dimmer, E., Huntley, R., Binns, D., O'Donovan, C., Apweiler, R.: The goa database in 2009 an integrated gene ontology annotation resource. Nucleic Acids Research 37, D396–D403 (2009)
2. Sharan, R., Ulitsky, I., Shamir, R.: Network-based prediction of protein function. Molecular Systems Biology 3, 1–13 (2007)
3. Settles, B.: Active learning literature survey. University of Wisconsin, Madison (2010)
4. Von Luxburg, U.: A tutorial on spectral clustering. Statistics and Computing 17(4), 395–416 (2007)
5. Sabidussi, G.: The centrality index of a graph. Psychometrika 31, 581–603 (1966)
6. Freeman, L.C.: A set of measures of centrality based upon betweenness. Sociometry 40, 35–41 (1977)
7. Sen, P., Namata, G., Bilgic, M., Getoor, L., Gallagher, B., Eliassi-Rad, T.: Collective classification in netwok data. AI Magazine 29, 93–106 (2008)
8. Ruepp, A., Zollner, A., Maier, D., Albermann, K., et al.: The funcat, a functional annotation scheme for systematic classification of proteins from whole genomes. Nucleic Acids Research 32, 5539–5545 (2004)
9. Güldener, U., Münsterkötter, M., Kastenmüller, G., Strack, N., et al.: Cygd: the comprehensive yeast genome database. Nucleic Acids Research 33, D364–D368 (2005)
10. Ruepp, A., Doudieu, O., Van den Oever, J., Brauner, B., et al.: The mouse functional genome database (mfungd): functional annotation of proteins in the light of their cellular context. Nucleic Acids Research 34, D568–D571 (2006)

11. Damian, S., Andrea, F., Michael, K., Milan, S., et al.: The string database in 2011: functional interaction networks of proteins, globally integrated and scored. Nucleic Acids Research 39, D561–D568 (2011)
12. Bogdanov, P., Singh, A.K.: Molecular Function Prediction Using Neighborhood Features. IEEE/ACM Transactions on Computational Biology and Bioinformatics 7, 208–217 (2010)
13. Schwikowski, B., Uetz, P., Fields, S.: A Network of Protein-Protein Interactions in Yeast. Nature Biotechnology 18, 1257–1261 (2000)
14. Chua, H.N., Sung, W.K., Wong, L.: Exploiting indirect neighbours and topological weight to predict protein function from protein-protein interactions. Bioinformatics 22, 1623–1630 (2006)
15. Nabieva, E., Jim, K., Agarwal, A., Chazelle, B., Singh, M.: Whole-proteome prediction of protein function via graph-theoretic analysis of interaction maps. Bioinformatics 21(suppl. 1), i302-i310 (2005)
16. Arnau, V., Mars, S., Marin, I.: Iterative cluster analysis of protein interaction data. Bioinformatics 21, 364–378 (2005)
17. Adamcsek, B., Palla, G., Farkas, I.J., Derenyi, I., Vicsek, T.: Cfinder: locating cliques and overlapping modulesin biological networks. Bioinformatics 22, 1021–1023 (2006)
18. Dunn, R., Dudbridge, F., Sanderson, C.: The use of edge-betweenness clustering to investigate biological function in protein interaction networks. BMC Bioinformatics 6, 39 (2005)
19. Chua, H.N., Sung, W.K., Wong, L.: An efficient strategy for extensive integration of diverse biological data for protein function prediction. Bioinformatics 23(24), 3364–3373 (2007)
20. Hu, L., Huang, T., Shi, X., Lu, W., Cai, Y., et al.: Predicting functions of proteins in mouse based on weighted protein-protein interaction network and protein hybrid properties. PLoS ONE 6(1), e14556 (2011)
21. Settles, B., Craven, M.: An analysis of active learning strategies for sequence labeling tasks. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing, pp. 1069–1078. ACL Press (2008)
22. Körner, C., Wrobel, S.: Multi-class ensemble-based active learning. In: Fürnkranz, J., Scheffer, T., Spiliopoulou, M. (eds.) ECML 2006. LNCS (LNAI), vol. 4212, pp. 687–694. Springer, Heidelberg (2006)
23. Settles, B., Craven, M., Ray, S.: Multiple-instance active learning. In: Advances in Neural Information Processing Systems, vol. 20, pp. 1289–1296. MIT Press (2008b)
24. Guo, Y., Greiner, R.: Optimistic active learning using mutual information. In: Proceedings of International Joint Conference on Artificial Intelligence, pp. 823–829. AAAI Press (2007)
25. Xu, Z., Akella, R., Zhang, Y.: Incorporating diversity and density in active learning for relevance feedback. In: Amati, G., Carpineto, C., Romano, G. (eds.) ECiR 2007. LNCS, vol. 4425, pp. 246–257. Springer, Heidelberg (2007)
26. Liu, Y.: Active learning with support vector machine applied to gene expression data for cancer classification. Journal of Chemical Information and Computer Sciences 44(6), 1936–1941 (2004)
27. Vogiatzis, D., Tsapatsoulis, N.: Active learning for microarray data. International Journal of Approximate Reasoning 47(1), 85–96 (2008)
28. Mohamed, T.P., Carbonell, J.G., Ganapathiraju, M.K.: Active learning for human protein-protein interaction prediction. BMC Bioinformatics 11(suppl. 1), S57 (2010)