Association Analysis-based Transformations for Protein Interaction Networks: A Function Prediction Case Study

Gaurav Pandey
Dept of Comp Sc & Engg
Univ of Minnesota, Twin Cities
Minneapolis, MN, USA
gaurav@cs.umn.edu

Michael Steinbach
Dept of Comp Sc & Engg
Univ of Minnesota, Twin Cities
Minneapolis, MN, USA
steinbac@cs.umn.edu

Rohit Gupta
Dept of Comp Sc & Engg
Univ of Minnesota, Twin Cities
Minneapolis, MN, USA
rohit@cs.umn.edu

Tushar Garg
Dept of Comp Sc & Engg
Univ of Minnesota, Twin Cities
Minneapolis, MN, USA
garg@cs.umn.edu

Vipin Kumar
Dept of Comp Sc & Engg
Univ of Minnesota, Twin Cities
Minneapolis, MN, USA
kumar@cs.umn.edu

ABSTRACT

Protein interaction networks are one of the most promising types of biological data for the discovery of functional modules and the prediction of individual protein functions. However, it is known that these networks are both incomplete and inaccurate, i.e., they have spurious edges and lack biologically valid edges. One way to handle this problem is by transforming the original interaction graph into new graphs that remove spurious edges, add biologically valid ones, and assign reliability scores to the edges constituting the final network. We investigate currently existing methods, as well as propose a robust association analysis-based method for this task. This method is based on the concept of h-confidence, which is a measure that can be used to extract groups of objects having high similarity with each other. Experimental evaluation on several protein interaction data sets show that hyperclique-based transformations enhance the performance of standard function prediction algorithms significantly, and thus have merit.

Categories and Subject Descriptors

H.2.8 [**Database Management**]: Database Applications—*Data mining*; J.3 [**Life and Medical Science**]: Biology and genetics; E.1 [**Data Structures**]: Graphs and networks

General Terms

Algorithms, Performance, Reliability, Experimentation

Keywords

Association analysis, h-confidence, protein interaction networks, noise reduction, protein function prediction

KDD'07, August 12–15, 2007, San Jose, California, USA.

1. INTRODUCTION

One of the most promising forms of biological data that are used to study the functions of proteins at a genomic scale are protein interaction networks. These networks are generated by collecting interactions between two or more proteins, which are in turn obtained from various sources such as metabolic or synthetic pathways, protein complexes and ligand-receptor mechanisms [32]. These networks provide a global view of the interactions between various proteins that are essential for the accomplishment of most protein functions. Due to the importance of the knowledge of these interactions, several high-throughput methods have been proposed for discovering them [14]. In fact, several standardized databases, such as DIP [33] and GRID [4] have now been set up to provide systematic access to protein interaction data collected from a wide variety of experiments and sources.

It is easy to see that a protein interaction network can be represented as an undirected graph[1], where proteins are represented by nodes and protein-protein interactions as edges. Due to this systematic representation, several computational approaches have been proposed for the prediction of protein function from protein interaction graphs [18, 23, 25, 20, 30, 15, 16]. These approaches can be broadly categorized into four types, namely neighborhood-based, global optimization-based, clustering-based and association analysis-based [18]. Due to the rich functional information in these networks, several of these approaches have produced very good results, particularly those that use the entire interaction graph simultaneously and use global optimization techniques to make predictions [16, 30]. Indeed, recently, some studies have started using protein interaction networks as benchmarks for evaluating the functional relationships between two proteins [37].

However, despite the advantages of protein interaction networks, they have several significant weaknesses which affect the quality of the results obtained from their analysis. The most prominent of these problems is that of noise in the data, which manifests itself primarily in the form of spurious or false positive edges [22]. Studies have shown that the presence of noise has significant adverse affects on the performance of protein function prediction al-

[1]Unless otherwise specified, a graph means an undirected graph in the rest of this paper. Also, since a graph is the only representation of an interaction network that is used in our study, we use the terms network and graph interchangeably.

gorithms [7]. Another important problem facing the use of these networks is their incompleteness, i.e., the absence of biologically valid interactions even from large sets of interactions [30, 15]. This absence of interactions from the network prevents even the global optimization-based approaches from making effective use of the network beyond what is available, thus leading to a loss of potentially valid predictions. In this paper, we investigate techniques that try to address these two problems with the objective of improving function prediction results.

A possible approach to these problems is to transform the original interaction graph into a new weighted graph such that the weights assigned to the edges in the new graph more accurately indicate the reliability and strength of the corresponding interactions, and their utility for predicting protein function. In our study, a graph transformation converts an undirected graph $G =< V, E, W >$ to a new graph $G' =< V, E', W' >$. Interestingly, the alternate representation of the G as an adjacency matrix $A_{|V|\times|V|}$, where $A(p_1, p_2) = 1$ if p_1 and p_2 interact, allows the application of association analysis techniques to the task of creating a new weighted adjacency matrix $A'_{|V|\times|V|}$ corresponding to G'. Note that the new weighted graph may have some new edges (i.e., edges that had 0 weight in the original graph) and may skip some edges that are present in the original network. This elimination of spurious edges can reduces the noise and the addition of viable edges can reduce the incompleteness of the original interaction network.

Association analysis is a field of data mining that is focused primarily on the analysis of objects that co-occur frequently in a data set, and are thus hypothesized to be *associated* to each other [28]. Several types of such *frequent patterns* and algorithms for deriving them have been proposed, the most common ones being frequent itemsets and the *Apriori* algorithm respectively [1, 3, 2]. Recently, a new type of frequent pattern known as *hyperclique* has been proposed for addressing the problem of skewed distributions of object frequencies in binary data [36]. This pattern is based on the *h-confidence* measure. This measure is just the number of times items appear together divided by the maximum number of times that one of the items occurs by itself. Thus, objects that are part of a hyperclique derived at a high h-confidence are tightly associated with each other, and those that are not a part of any hyperclique are often noisy objects.

Indeed, this idea has produced good results when applied to finding patterns in a number of situations, including those involving noisy data. In one study [35], hyperclique patterns were used to remove noise from document and gene expression data. In another study [34], hyperclique pattern discovery was applied to protein complex data to find functional modules. Specifically, the complex data was represented as a binary data set whose attributes corresponded to the presence or absence of proteins, while the hyperclique patterns found in such data were treated as candidate functional modules, i.e., groups of proteins have related functions. Examination of the hyperclique patterns showed that many of the hypercliques did contain several functionally related proteins.

This success of hypercliques in noise removal from binary data, coupled with the representation of protein interaction graphs as a binary matrix to which association analysis techniques can be applied, motivated us to address the graph transformation problem using an approach based on h-confidence. We perform this task in two ways. In the first, we compute new edge weights for pairs of proteins by computing the h-confidence between them based on the binary adjacency graph, A. This is equivalent to the process of finding size two hyperclique patterns in the adjacency graph. Second, if the weighted adjacency matrix of the original graph A_w is available, we employ a continuous version of h-confidence to compute new edge weights between all pairs of proteins. Thus, depending on whether the input graph was weighted or unweighted, we produce one or two transformed graphs, both of which may have edges and weights different from the original graph. This transformation is expected to reduce noise in the network since the resultant edges that have a high weight are highly likely to connect proteins that have a strong association in the original interaction network.

In order to evaluate the efficacy of the resultant networks for protein function prediction, we provided the original and the transformed graphs as input to the *FunctionalFlow* algorithm [16]. *FunctionalFlow* is a graph theory-based algorithm that enables insufficiently connected proteins to obtain functional annotations from distant proteins in the network, and has produced much better results than several other function prediction algorithms. The results obtained from these experiments show that the transformed graphs are significantly more capable of accurately predicting protein function as compared to the original network, as well as other recently proposed transformations methods that we evaluated. In addition, the improvement in performance was larger for networks for which the reliabilities for the edges were estimated indirectly using sources such as gene expression data, as compared to reliabilities computed using experimental means or functional annoations of the interacting protein.

An association analysis based approach to graph transformation is just one of the possible approaches. For comparison, we also consider a couple of other approaches as well. One such algorithm [23] computes the strength of the edge between two nodes in the transformed graph as the probability that they shared a given number of neighbors in the original graph by chance. Other studies have more directly used the number of common neighbors between two proteins, or a minor variation thereof, to estimate the reliability of an interaction between the two proteins in the transformed network [5, 15]. These approaches are detailed in Section 3.

1.1 Contributions of our work

This paper makes a contribution to the task of protein function prediction by proposing novel association analysis-based transformation methods based on h-confidence for protein interaction networks represented as graphs. This includes a technique for evaluating the reliabilities of the edges for unweighted networks, and a method to produce more noise-resistant weights for weighted networks. Through extensive evaluation, we show that the proposed transformations and weighting methods produce more accurate functional annotations for proteins in the network. This is due to the smaller amount of noise and a more complete set of biologically viable interactions in the transformed network.

More generally, this work provides a novel example of an application where frequent patterns (hypercliques here) are extracted in the traditional market-basket setting from a symmetric binary matrix. In addition, we propose a new formulation for the h-confidence measure for pairs of vectors containing continuous values. Although the focus for both these techniques is on producing better graphs for protein function prediction, both approaches could be profitably applied to other data mining problems.

This paper builds upon our preliminary work on a related topic [11], that investigates the utility of association analysis for protein complex and interaction data to enhance SwissProt keyword recovery.

Overview The remainder of the paper is organized as follows. Sections 2 and 3 provide the necessary background information for the rest of the paper by describing the function prediction and graph transformation techniques used. Sections 4 and 5 detail the infrastructure of the study in the form of data sources and the evaluation methodology used respectively. Finally, we present the results of

this evaluation in Section 6 and make concluding remarks in Section 7.

2. PROTEIN FUNCTION PREDICTION USING FUNCTIONALFLOW

As mentioned earlier, due the richness of functional information in protein interaction networks and their systematic representation as a graph, several computation approaches have been proposed for inferring protein function from one or several interaction networks [15, 16]. These approaches can be broadly classified into four categories [18]:

- **Neighborhood-based approaches**: These approaches assign functional labels to an unannotated protein by transferring labels from its neighborhood [25, 15].
- **Clustering-based approaches**: These approaches construct functional modules by discovering densely connected regions in the interaction network, and assign unannotated proteins the most dominant label(s) in their corresponding module(s) [20].
- **Global optimization-based approaches**: These approaches utilize the entire connectivity structure of the network in order to transfer the annotations of distantly connected proteins to the query protein(s) [16, 30].
- **Association analysis-based approaches**: These approaches use association analysis algorithms to detect frequently occurring sets of interactions in interaction networks, and hypothesize that these subgraphs denote function modules [34].

Due to their ability to gather predictions from the whole network and confidently assign them to unannotated proteins, approaches in the last category have generally produced the best results in protein function prediction from interaction networks [18]. In particular, the FunctionalFlow algorithm [16] was shown to outperform several other function prediction approaches in a comprehensive evaluation study by its authors. It also has the advantage of being backed by well-founded graph theoretic concepts. Due to these merits, we chose FunctionalFlow as the base algorithm for evaluating the effectiveness of various transformed graphs for the task of protein function prediction.

FunctionalFlow is based on the concept of network flow in graph theory [31]. However, since network flow is defined for directed graphs, FunctionalFlow uses an iterative algorithm for interaction networks, which are represented as undirected graphs. For each function a, the set of proteins annotated with a are treated as *sources*, the other proteins as *sinks*, and the weights are used as capacities of the corresponding edges in the graph. The algorithm then iteratively "flows" the functional annotations from the sources to the sinks, using a downhill flow strategy. In this strategy, the annotations flow only from a more full node to a less full one directly connected to it, while maintaining the constraint that the flow on the edge does not exceed its capacity. At the end of the pre-specified number of iterations, all the nodes in the network have a certain functional score for a, from which annotations are made using a threshold on this score. Repeating this process for all the functions in the given set of annotations produces the required functional annotations for the set of query proteins.

The above description indicates that the good results of FunctionalFlow can be attributed to the use of annotations from both close and distant neighbors in the network, as well as the effective use of edge weights to control the flow of annotations from one protein to another. In other words, an interaction graph with more accurate edges and weights is expected to yield better function predictions.

3. GRAPH TRANSFORMATION

This section describes the various techniques that we used for graph transformation, i.e., to transform the original interaction graph $G =< V, E, W >$ to a new graph $G' =< V, E', W' >$. As mentioned, edges may be either deleted or added to address the problems of the noisiness and incompleteness of the data, respectively. A number of recent techniques that have been developed for this purpose are described next. They employ a variety of approaches, but the goal is to transform the graph by adding or deleting edges in order to produce a new graph that is more suitable for protein function prediction.

3.1 Adjacency Matrix

The simplest technique is to treat the protein interaction network as an adjacency matrix for the set of proteins, i.e., to make the edge weights of all interacting pairs of proteins equal to 1. If the original matrix does not have weights, than this transformation does not change the graph. If the original graph has weights, then often a threshold is applied to eliminate weak edges. Thus, this approach is primarily used to show the value of weighted interaction graphs for function prediction.

3.2 Common Neighbor

This graph transformation technique is based on the observation that proteins that share a number of neighbors are more likely to have a function in common. Indeed, the evaluation of this approach on real interaction data sets [15] showed that predicting the function of a protein based on the proteins with which it shares a number of common neighbors has better accuracy than predicting function based on proteins that are merely neighbors of the protein. Also, unlike the neighborhood approach, which typically sees accuracy rise and then decline as the number of neighbors increase, an approach based on common neighbors attains a relatively stable level of accuracy as the number of common neighbors increases.

Using the common neighbor strategy for graph transformation is straightforward. Specifically, an edge is placed between two proteins only if those proteins have at least one neighbor, and the weight of that edge is the number of common neighbors. Note that some proteins that originally had an edge in the original graph can become disconnected, i.e., edges may be lost, while two proteins that did not originally have an edge, may be connected in the transformed graph.

This approach is closely related to the shared nearest neighbor approach for clustering [13, 8]. In the SNN approach, the nearest neighbors of the objects are determined from their similarity or distance. Then, a new distance measure is defined based on the number of neighbors that appear on both of the nearest neighbor lists of the objects [13]. This approach has been shown to have good performance for clustering in dealing with noisy and high-dimensional data [8].

3.3 P-value

The motivation for this graph transformation method is the same as the Common Neighbor approach, namely that those proteins that share many neighbors are more strongly connected, i.e., more likely to be functionally similar. However, this approach addresses the fact that the significance of two proteins sharing a particular number of neighbors depends on the number of neighbors that each has. To illustrate, if two proteins, each having only two neighbors, share both these neighbors, then this is more significant, in terms of prob-

ability, than two proteins, each having 20 neighbors, that share only two.

The formula for the probability (p-value) of an edge is given by Equation 1, which is taken from Samanta and Liang [24]. Note that N is the number of proteins, n_1 is the number of neighbors of protein p_1, n_2 is the number of neighbors of protein p_2, and m is the number of neighbors shared by the two proteins. In practice, the negative log of this probability is easier to work with and has the property that larger numbers imply stronger edges. In other words, typically we take $w(p_1, p_2) = -\log(prob(N, n_1, n_2, m))$.

$$prob(N, n_1, n_2, m) = \frac{\binom{N}{m}\binom{N-m}{n_1-m}\binom{N-n_1}{n_2-m}}{\binom{N}{n_1}\binom{N}{n_2}} \quad (1)$$

3.4 H-confidence for Binary Data

H-confidence [36], also known as all-confidence [17], is a measure of the association of items (binary attributes). If a set of items has an h-confidence more than a user-specified threshold, h_c, then the itemset is called a hyperclique. Definition 1 defines hypercliques and h-confidence more formally. The quantities of 'support' and 'confidence' are as defined in standard association analysis [1, 28].

DEFINITION 1. ***Hyperclique*** *A set of items (binary attributes), X, forms a hyperclique with a particular level of h-confidence, where h-confidence is defined as*

$$\begin{aligned} \text{hconf}(X) &= \min_{i \in X}\{\text{confidence}(\{i\} \rightarrow \{X - \{i\}\})\} \quad (2) \\ &= \text{support}(X) / \max_{i \in X}\{\text{support}(\{i\})\} \quad (3) \end{aligned}$$

H-confidence is between 0 and 1, with a value of 0 indicating no association and a value of 1 indicating the strongest association between a group of items, i.e., the items always occur together. Thus, h-confidence can be used as a measure of similarity between the attributes in a binary data matrix.

Specifically, the adjacency matrix A of a protein interaction network is considered as a binary data matrix by treating its rows as transactions and columns as items. (Note that both correspond to protein). Then a weighted adjacency matrix A' of the same dimensions as the original one can be generated using $A'(i, j)$ =h-confidence(i, j). Informally, the h-confidence of a pair of proteins, p_1 and p_2, will be high if p_1 tends to be a neighbor of a protein whenever p_2 is and, vice-versa. Using Equation 3 and the terminology introduced for the p-value transformation, h-confidence for a pair of proteins is given by the following equation:

$$\text{hconf}(\{p_1, p_2\}) = \min\left(\frac{m}{n_1}, \frac{m}{n_2}\right) \quad (4)$$

Using the h-confidence of two proteins as the weight of the edge between the proteins, a new graph can be created. However, in addition to an h-confidence threshold, it is also necessary to take into account the absolute number of times two proteins appear together (m) as well as the fraction of times that the occurrence of one protein as a neighbor implies the occurrence of the other protein as a neighbor (min($n_1/m, n_2/m$)). For example, if proteins p_1 and p_2 both have only one edge, which is to protein p_3, then their h-confidence will be one. On the other hand, proteins p_1 and p_2, may have edges with 10 other proteins, 8 of which they share. They will have an h-confidence of 0.8 and thus, seem not to be as strongly connected as the first pair. To deal with such problems it is common to set a support threshold, i.e., to require that m have at least some specified value.

3.5 H-confidence for Continuous Data

As originally defined, h-confidence is only applicable to binary data or, in the context of protein interaction graphs, unweighted graphs [34]. However, the notion of h-confidence can be readily generalized to continuous data.[2] For the situation of weighted interaction networks, this amounts to replacing the counts, m, n_1, and n_2 in Equation 4 by numbers based on the weights. In particular, n_1 and n_2 are the sum of the weights of all edges involving p_1 and p_2, respectively, while m is defined to be the sum of the minimum of the edge weights of p_1 and p_2 on their shared edges. As with h-confidence defined on binary data, h-confidence on continuous data is between 0 and 1, with 1 indicating the strongest connection. More specifically, the $h - confidence$ of two vectors will be non-zero only if they both contain a non-zero value at the same position, while it will be high if both these values are high. Thus, in the domain of interaction networks, two proteins will be linked with an edge carrying a high weight only if they are connected to an overlapping set of proteins with highly reliable interactions.

To illustrate the difference between binary and continuous h-confidence, consider the following example of two proteins, p_1 and p_2. In the weighted adjacency matrix, the two proteins occur together in two rows and have weights 0.2 and 0.4, respectively, in the first row, and weights 0.3 and 0.1, respectively, in the second row. In addition, p_1 also occurs by itself in another row and has a weight of 0.5. Disregarding the weights, and considering only the number of edges, n_1 = 3, n_2 = 2, and m = 2. Using Equation 4, binary h-confidence = $\min(2/3, 2/2) = 2/3$. However, using weights in the manner just described, n_1 = 0.2 + 0.3 + 0.5 = 1.0, n_2 = 0.1 + 0.4 = 0.5, and $m = \min(0.2, 0.4) + \min(0.3, 0.1) = 0.3$. This implies that continuous h-confidence = $\min(0.3/1.0, 0.3/0.5) = 0.3$. Thus, for this example, continuous h-confidence is significantly smaller than binary h-confidence.

3.6 Pruning

Pruning refers to the elimination of edges having a weight below a specified threshold. This approach is sometimes applied to the raw interaction graph to eliminate less reliable, i.e., lower weight edges. However, the transformed graphs produced by the techniques described above typically have substantially more edges than the original graphs since all potential pairwise protein-protein interactions are evaluated. Some of these interactions in the transformed graph may have small non-zero weights due to factors such as a random common neighbor in the original graph. Hence, pruning of the edges is conducted on the basis of the weights assigned in the transformed graph to remove unreliable edges.

4. DATA SOURCES

In this section, we discuss the functional classification scheme and the interaction data sets used in our study.

4.1 Functional Classification Scheme: FunCat

Since our evaluation of the various graph transformations is based on the improvement provided by each of them over the raw network in the task of protein function prediction, it is important to define a set of reliable functional labels to be assigned to each protein. We chose the set of functional labels at a depth of two in the FunCat

[2]We had previously extended the notion of h-confidence to continuous data [26], but in a manner slightly different from that given here. The previous approach addressed the general case which involved continuous attributes that could have widely different scales. The current formulation is better suited when all the attributes have similar scales.

classification scheme of the MIPS database [21]. We made this selection since this scheme has been widely used in function prediction literature [18], and the selected labels represent a good trade-off between the generality and specificity of the labels in this hierarchy. Also, all our experiments are performed on yeast proteins, and there are about 4500 proteins in yeast that can be annotated using the labels that we selected. Since we use a cross-validation-based evaluation methodology, we only consider this set of proteins in our study.

4.2 Protein Interaction Data Sets

In order to be able to conduct a general evaluation study of the graph transformation methods, we selected the high-throughput protein-protein interaction networks of budding yeast (*S. cerevisiae*) listed in Table 2, since each of these data sets follows a different weighing scheme for the constitent interactions. Table 2 specifies the sizes of these networks in terms of the number of proteins and interactions constituting them, considering only the proteins annotated using our selection of functional labels. We removed any instances of redundant interactions, such as the interaction $B - A$ when $A - B$ is already present in the data set, and self interactions, such as $A - A$, where A and B are proteins. A short description of each of the data sets and the weighing scheme adopted by them follows.

4.2.1 DIPCore

Deane *et al* [6] proposed two methods for assessing the reliability of high-throughput protein interactions, namely the paralogous verification method (PVM) and the expression profile reliability index (EPRI). Using this method, they prepared the DIPCore data set, which is a set of highly reliable interactions selected from the Database of Interacting Proteins [33]. This set consists of 5731 interactions between 2526 yeast proteins. However, in its publicly available format[3], the interactions in DIPCore do not have any associated weights. Hence, in our study, we assumed these weights to be 1 for all interactions.

4.2.2 The *combined* data set

In order to illustrate the case of interaction data sets whose reliabilities are estimated indirectly using other types of genomic data, we constructed a combination of three high-throughput yeast interaction sets, namely those of Gavin *et al* [9], Uetz *et al* [29] and Ito *et al* [12], and refer to it as the *combined* data set. This data set consists of 7753 interactions between 3781 proteins, and the weights for these interactions are derived as follows. The reliabilities of each of the individual data sets was estimated using the EPR Index [6] tool provided by DIP [4], which computes the reliability of a data set by comparing the distribution of gene expression distances between the pairs of interacting proteins in the given data set, with that obtained from the DIPCore data set. The reliabilities computed for the above three data sets are tabulated in Table 1. Finally, the individual edge weights are calculated using the commonly used formula of $w(e) = 1 - \prod_i (1 - r_i)$ [16, 19], where the product runs over all the data sets i where edge e is found, and r_i is the corresponding reliability of data set i. Overall, this methodology provides us a set of indirectly derived weights for the edges constituting the *combined* data set.

4.2.3 Krogan *et al*'s data set

Recently, Krogan *et al* [10] have reported a large high-throughput and reliable data set of 7123 interactions among 2708 yeast proteins. In addition, they have also assigned likelihood values for each interaction in their data set using various machine learning algorithms that tried to estimate the experimental reproducibility of these interactions. Thus, we treated these likelihood scores as the edge weights and used the entire set as an example of a data set whose edges are weighted directly using experimentally observed interaction data.

In summary, our selection of interaction data sets does indeed reflect a variety of weighing schemes used. Also, it can be seen that none of these data sets cover the entire yeast genome, and thus are highly likely to be incomplete.

Data set	# Interactions	Reliability
Gavin *et al* [9]	3210	0.744
Uetz *et al* [29]	822	0.492
Ito *et al* [12]	3959	0.201

Table 1: Reliabilities of data sets computed using EPRI

5. EVALUATION METHODOLOGY

The previous sections detailed the different data sets used in our study and the graph transformations that were used to process them and produce different transformed variants of the original interaction graph. These graphs were then input into the FunctionalFlow algorithm to produce predictions of functions for the constituent proteins. However, since it is hard to evaluate the predictions made for unannotated proteins, we restricted our evaluation to the proteins annotated with at least one functional label at depth two in the FunCat hierarchy. Table 2 details the number of proteins and interactions after imposing this restriction on each of the data sets used in this study.

Data set	# Annotated Proteins	# Corresponding Interactions
DIPCore	2315	5413
Combined	3026	6490
Krogan *et al*	2291	6180

Table 2: Details of interaction sets used

Using this set of annotations, we used the FunctionalFlow algorithm in a five-fold cross validation procedure, which produces a likelihood score for each protein being annotated with each label (henceforth referred to as a protein-label pair). Now, in order to convert these scores into annotations, we follow a global scoring strategy. In this strategy, we sort the entire set of protein-label scores in descending order, and then selected the k^{th} score as the threshold for annotation, i.e., all protein-label pairs with scores greater than this threshold are predicted as annotations. Constantly increasing the value of k thus provided us a set of functional annotation at different stringencies, and we used these annotation to calculate the following metrics for evaluating the performance of the algorithm.

5.1 Precision-Recall

In order to evaluate the overall performance, we used the precision-recall framework of evaluation [28]. However, since the traditional precision and recall metrics are defined only for binary classification problem, it needs to be modified for function prediction, given that a protein may ideally have multiple labels. Thus, we adopt the following definition of precision and recall used by other function prediction studies [7, 15].

[3] `dip.doe-mbi.ucla.edu/dip/Download.cgi?SM=6`
[4] `dip.doe-mbi.ucla.edu/dip/Services.cgi?SM=1`

$$Precision = \frac{\sum_{i=1}^{K} k_i}{\sum_{i=1}^{K} m_i}$$

$$Recall = \frac{\sum_{i=1}^{K} k_i}{\sum_{i=1}^{K} n_i}$$

Here, K is the total number of proteins with known functional labels, and for each protein i, m_i is the number of labels predicted by the algorithm, n_i is the actual number of labels possessed by the protein, and k_i is the number of labels common to the actual and predicted set of labels. Sccording to these definitions, $Precision$ denotes the proportion of correctly predicted annotations out of all the functional predictions made, while $Recall$ measures of the proportion of correctly predicted functions out of all the known annotations [28]. Thus, these measures are a suitable generalization of the original precision-recall framework to the multi-label scenario.

5.2 Accuracy of Top-k Predictions

A biological researcher in the area of functional genomics may choose a number of predictions of protein function for experimental investigation. Since the number of experiments that can be performed is quite limited, it is important to choose the most promising candidates for investigation, i.e., to focus on those functional predictions most likely to be correct. Thus, for this situation, a list of the top *k* predictions is often more relevant than a precision-recall or ROC curve.

The details of the top-*k* evaluation methodology are as follows. First, using the global scoring method, the *k* protein-label pairs with the highest functional score are identified, and are produced as the functional predictions of the algorithm. Next, the prediction accuracy, or the precision, of this set of predictions is evaluated with respect to the known protein-label annotations. Then, a curve of prediction accuracy versus number of protein-label pairs predicted is produced by considering various values for *k*. We used values of *k* up to 500 or 1000 in our experiments.

Note that these two metrics are related by the following equation:

$$\frac{Recall}{Precision} = \frac{k}{Total\ \#\ true\ protein - label\ annotations}$$

Thus, they provide two related perspectives on the performance of a function prediction algorithm.

6. EXPERIMENTAL RESULTS

We evaluated several graph transformation methods using a wide variety of protein interaction data sets by testing the performance of the FunctionalFlow algorithm on the resultant interaction graph. Our data sets were selected to reflect the various types of interaction weighting schemes currently in use to estimate the reliabilities of protein-protein interactions. The following sections detail the results of our evaluation on each of these data sets. Note that all the results reported were obtained by a five-fold cross validation-based evaluation of the predictions produced by FunctionalFlow. Also, it was mentioned in Section 3.6 that the three transformations, namely p-value-based, common neighbor-based and h-confidence-based, may contain some spurious links in the tranformed graph. Hence, we tried several pruning thresholds for each of these methods, the details of which are provided in Table 3. Note that the best value is the most commonly best performing value for the parameter, and is used to report the evaluation results, unless some other value is specified.

Method	Parameter	Values tried	Best value
P-value	Max(p)	1,10^{-3},10^{-5}	10^{-3}
Common nbr	Min(cmn nbrs)	1,2,3	2
Cont hconf	Min(hconf)	0,0.1,0.2	0.1
Bin hconf	Min(support)	1,2,3	2
	Min(hconf)	0,0.1,0.2	0

Table 3: Paramater values tried for different transformations

Finally, in order to make the discussion clearer, we use the following notation in the rest of this section. The transformation of the binary adjacency matrix of an interaction graph to its transformed h-confidence-based adjacency matrix is referred to as the *bin hconf* (binary h-confidence) method, while the transformation of the weighted adjacency matrix of a graph to its transformed h-confidence-based adjacency matrix is referred to as the *cont hconf* (continuous h-confidence) method. The other notations are self-explanatory. Also, note that the plots in this section are best viewed in color.

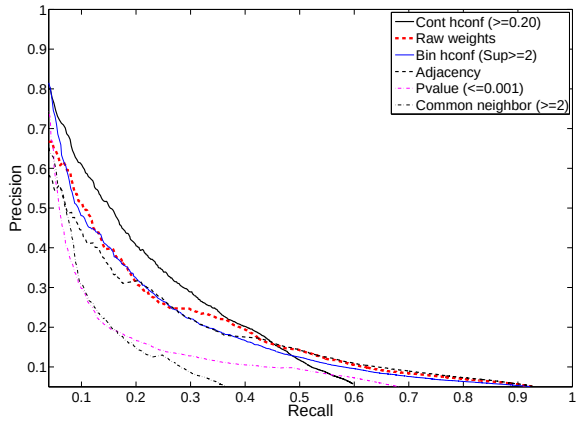
6.1 The combined data set

In this experiment, we investigated the applicability of various graph transformation methods to enhance the weighted networks produced by indirect weighting methods such as EPRI [6]. The representative of this category was the *combined* data set described earlier in Section 4. We transformed this weighted network using the continuous h-confidence-based method, and its unweighted version using three transformation methods, namely the binary h-confidence-based, the p-value-based and the common neighbor-based. Figure 1 shows the performance of the FunctionalFlow algorithm on the the original *combined* graph, its unweighted adjacency matrix, and the four transformations computed above. It can be seen from Figure 1(a) that continuous h-confidence has better overall performance in terms of precision and recall. This improvement is shown much more clearly by Figure 1(b), which shows that the continuous h-confidence-based transformed graph produces the most accurate predictions when only the top 1000 predictions are considered, and outperforms the raw weighted network by nearly 10% throughout. For instance, if only the top 500 predictions are considered, the accuracy of the predictions made using the raw weighted network is only about 70%, while that of the predictions made using the continuous h-confidence-based transformations is over 80%. Also, for this data set, p-value-based transformation outperforms the raw network, though by a smaller margin than the h-confidence-based transformations.

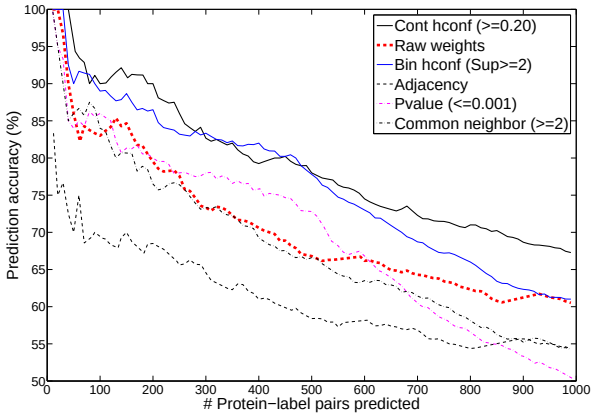
6.2 Krogan et al's data set

Krogan *et al*'s data set was discussed earlier in Section 4, and was used in their study [10] to discover overlapping protein complexes, which were subsequently categorized into cores, modules and isoforms. In addition to the good results obtained, a major component of this study was the use of machine learning algorithms to compute the likelihood of an observed interaction to be valid. In this experiment, we tested if the use of h-confidence could enhance these weights for effective function prediction. Thus, we used the continuous version of the h-confidence measure, defined in Section 3.5, to transform the original weighted interaction network and calculate the reliability of an edge connecting two proteins i and j on the basis of the strengths of their interactions with each other or with other proteins. Similar computations are carried out for the other transformation methods such as p-value and common neighbor.

The results of this experiment are shown in Figure 2. Figure 2(a) shows that h-confidence-based transformation was indeed able to

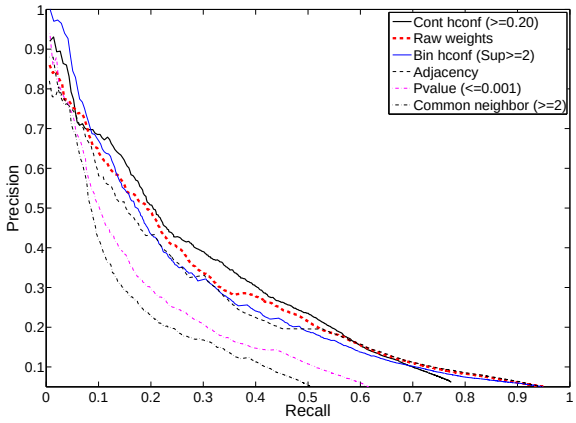


(a) Overall precision-recall performance

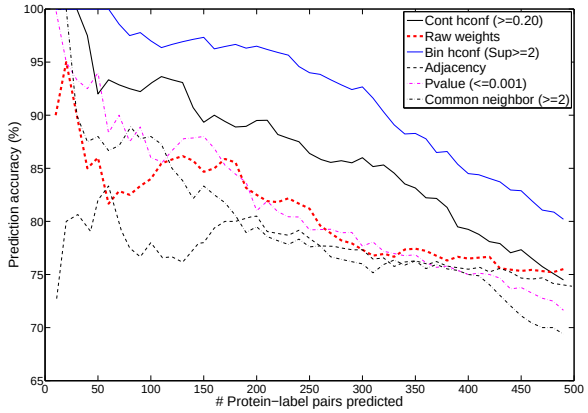


(b) Accuracy of top 1000 predictions

Figure 1: Performance of graph transformation methods on the *combined* interaction set



(a) Overall precision-recall performance



(b) Accuracy of top 500 predictions

Figure 2: Performance of graph transformation methods on Krogan *et al*'s interaction set [10]

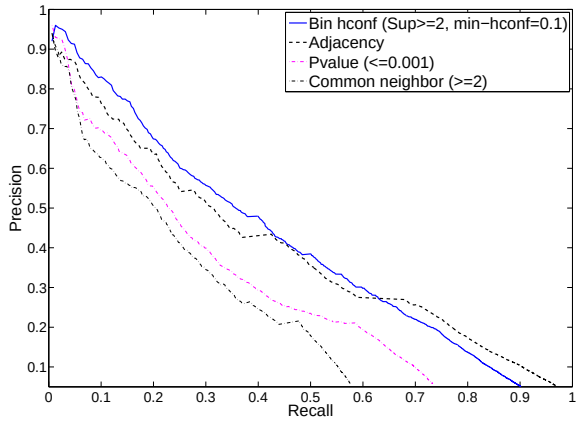
obtain better results under the precision-recall framework, as compared to the raw weighted graph and other transformations. This difference in performance between the various methods is accentuated by the relative performance of the different transformations of the original networks when only the top 500 predictions are considered. As shown by Figures 2(b), continuous h-confidence- and binary h-confidence-based transformations outperform the raw weighted graph by a large margin. For instance, for the top 150 predictions, a margin of 5% and 10% is observed respectively. These results show the merits of using association analysis-based transformation methods for this data set.

6.3 DIPCore

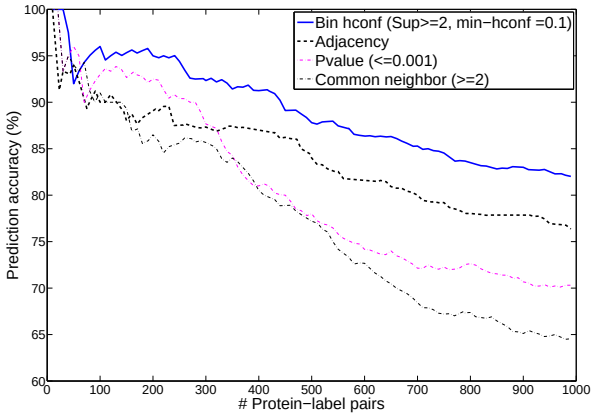
The final category of protein interaction data sets that we considered were those that did not contain the reliabilities of the edges explicitly, and since they form a single data set, it is difficult to weigh their edges using an approach such as the one used to estimate the edge weights of the *combined* data set. Thus, we focused this experiment on investigating the utility of graph transformation methods for unweighted interaction networks. We selected the DIPCore [6] data set for this experiment, which is a set of about 6000 highly reliable unweighted interactions selected from within DIP. The interaction graph was represented as a binary adjacency matrix A, and was transformed using the binary h-confidence, p-value and common neighbor methods.

We executed the FunctionalFlow algorithm with the four graphs (one original and three transformed) as input, and obtained the overall precision-recall curves for each of them, which are shown in Figure 3(a). These curves show that the transformed graph obtained using binary h-confidence perform better than the original graph for a large part of the precision-recall range, while those produced by the p-value and common neighbor methods performed worse throughout. It is important to point out that even though the improvement in performance may seem small, it is indeed significant when seen in the light of the fact that DIPCore is a very reliable data set [6], and is expected to be much richer in functional content than several other interaction data sets [15, 27].

As shown in Figure 3(b), even among these top predictions, where most methods are expected to produce good results due to the functional richness of the network, the binary h-confidence-based graph



(a) Overall precision-recall performance



(b) Accuracy of top 1000 predictions

Figure 3: Performance of graph transformation methods on the DIPCore data set [6]

performs significantly better than the raw network. More specifically, there is a significant improvement of about 5% throughout the top 1000 predictions. Also, graphs produced by p-value- and combined neighbor-based transformations perform worse than the original adjacency matrix.

Finally, it should also be noted that we applied our evaluation methodology for weighted interaction networks to the data set used by Nabieva *et al* to compare FunctionalFlow against other function prediction algorithms [16]. Even here, the continuous h-confidence-based transformation is able to outperform the original raw network. However, the difference between the performance of the two graphs is very small, since the weights of Nabieva *et al*'s data set are assigned using the functional labels of the interacting proteins. Thus, although it would be hard to outperform this network's and its weights' performance at the task of protein function prediction, the improvement achieved by the h-confidence-based transformation demonstrates its ability to enhance the functional information even in very precise networks.

In summary, through the evaluation of several protein interaction data sets that use a wide variety of reliability estimation schemes for their interaction, we showed that the h-confidence-based graph transformation method produces the most precise network, which can be used to predict protein function accurately. We believe that the success of this method is due to the changes made by it to the original network, namely removal of noisy edges and addition of biologically viable ones, in combination with effective reliability estimation of the edges in the resultant data set. We tested the validity of this hypothesis in the following final component of our study.

6.4 Effect of Noise on H-confidence-based Graph Transformation

We designed the following test in order to test the effect of noisy interactions in the input interaction graph on the accuracy of the graph produced by transforming it using the binary h-confidence-based transformation method. Given an interaction graph G containing n edges, and a noise level of $\alpha\%$, we generated $\frac{n \times \alpha}{100}$ edges that are not already present in G, and added these "spurious" edges to G to create a noisy interaction graph G'. However, it is difficult to generate weights randomly to these new edges, since the edge weights in each data set follow some unknown distribution. Thus, we applied this test only to the DIPCore data set, since its edges do not carry weights. The spurious edges generated are added as they are to the original data set without assigning them any weights. Next, the function prediction algorithm FunctionalFlow is executed using the raw and this noisy interaction graphs, as well as their transformed versions, which in this experiment are generated only using the binary h-confidence method. (Thus, the transformed graph in the following text refers only to the graph transformed using this method.) The results are compiled both in the precision-recall, as well the accuracy of the top-k predictions frameworks. The precision-recall results show the expected result that the precision-recall curves of the noisy versions of the raw graph and the transformed graph are inferior to their original counterparts (data not shown). However, the encouraging part of these results is that the gap between the performance of the noisy and the transformed noisy graphs is larger than that between the raw and the transformed raw graphs.

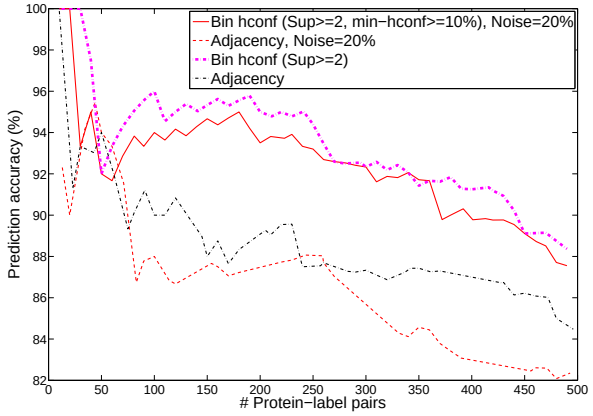


Figure 4: Effect of adding 20% noise

A more interesting result is presented by Figures 4 and 5, which illustrate the results of the accuracy of the top 500 predictions made using each of the noisy and original versions of the raw DIPCore network and their transformed counterparts which were further pruned

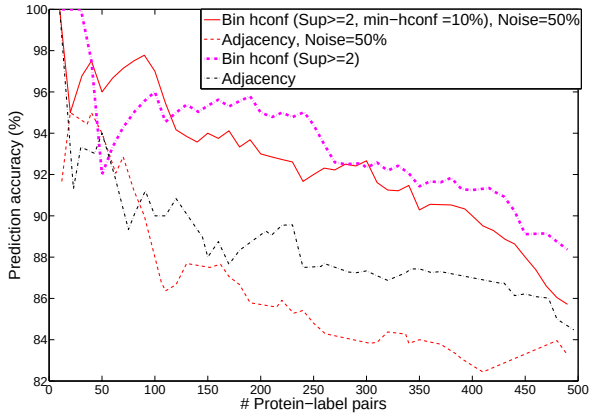


Figure 5: Effect of adding 50% noise

using a minimum h-confidence threshold of 0.1. These plots show that while the performance of FunctionalFlow deteriorates significantly when spurious edges are added to the original network, the corresponding loss in performance using the transformed graph is much less in comparison. For instance, it can be easily seen from Figure 4 that the performance of the transformed noisy graph is very close in performance to the transformed raw graph. On the other hand, the performance of the noisy network is significantly worse than that of the raw network. More specifically, in Figure 5, if only the first 300 predictions are considered, then the accuracy of the raw network is approximately 87%, which deteriorates to 84% using the noisy raw network. On the other hand, the performance of both the binary h-confidence-based graphs is almost 93%. Although these precise values fluctuate throughout the first 500 predictions, the general trend is that noise effects the raw interaction network much more adversely than the binary h-confidence-based network. This behavior of the binary h-confidence method can be explained on the basis of the noise resilience characteristic of hyperclique patterns, as well as the incompleteness of the interaction data sets.

In an earlier study [35], we showed that hypercliques are very effective in identifying large number of noisy objects from binary data sets, i.e., objects that are expected to have been generated by processes other than that used for the regular objects. It was shown that this removal of noisy objects significantly improved the results of important data mining operations such as clustering and association analysis. A similar phenomenon is expected to reduce the effect of the spurious edges added to the interaction graph. More specifically, protein interaction networks, particularly small ones such as DIPCore, are expected to be significantly incomplete [15, 30], and several interactions other than those already in the data set are expected to be biologically valid. Thus, among the set of noisy edges added to the original network, many are expected to be noisy, while a substantial number are also expected to be biologically valid. Now, due to their noise removal ability, the binary h-confidence-based transformation is able to identify the biologically more viable edges, and use them for prediction. On the other hand, the performance of the raw noisy network suffers, since the positive contribution of these valid edges is negated significantly by the noisy edges.

In summary, this test shows that our hypothesis that the function prediction performance of the h-confidence-based graph transformation is significantly better than that of the raw interaction network due to the three important changes made to the original network, namely the removal of spurious edges, the addition of biologically viable ones, and effective weighting of the resultant set of edges, is indeed highly likely to be true.

7. CONCLUDING REMARKS

The previous sections detailed several experiments that we conducted to evaluate the use of the various methods for transforming a given protein interaction network. Several detailed conclusions can be derived from these results, which we discuss below:

1. Given a variety of interaction networks, such as the *combined*, Krogan *et al*'s and the DIPCore interaction sets, the h-confidence-based transformations generally produce more accurate and more reliably weighted interaction graphs, which produce better results for protein function prediction than the original network.

2. Throughout our extensive evaluation, we observe that the transformed graphs produced by the p-value- and the common neighbor-based methods perform worse than the raw interaction networks when used for protein function prediction task. This phenomenon may occur due to several reasons. Firstly, these methods operate on the binary adjacency matrix of the input graph, which leads to a loss of information in the case of weighted interaction networks. Also, the filtering operation on these matrices usually leads to the loss of all the interactions of some proteins, thus disconnecting them from the rest of the network. For instance, for the common neighbor-based transformed graph of Krogan et al's data set derived using a filtering threshold of 2, there are as many as 1136 such disconnected proteins, out of 2291 proteins in the data set. Clearly, FunctionalFlow can not produce any predictions for these proteins. Since our definition of precision and recall does not take this case of disconnected proteins into account, the results of these two methods will naturally be affected adversely. However, a more detailed analysis of these methods is needed to evaluate their performance.

3. Looking deeper into the results of the experiments on several interaction networks, it can be seen that the less reliable the weights assigned to the edges in the raw network, the greater improvement in performance obtained by using a continuous h-confidence-based graph transformation. For instance, Figure 1(b) shows that for the *combined* data set, whose weights are derived from gene expression data, our proposed transformation is able to improve predictions by nearly 10% throughout, which is very encouraging. On the other hand, Figure 3(b) shows that for the DIPCore data set, which is known to be very reliable [15, 6], the performance benefits obtained by h-confidence, though significant, are of a relatively smaller magnitude, i.e., approximately 5% throughout. Thus, for applications using protein interaction networks whose edge weights are not available or are inappropriate, continuous h-confidence is expected to produce better performance than the raw weights and other transformations such as p-value- and common neighbor-based ones.

4. Finally, through a test on the DIPCore data set, in which a certain number of spurious edges were added to the original interaction network, it was shown that the performance of the h-confidence-based transformation suffers much less as compared to the noisy interaction network. This illustrates that the significantly better performance of the h-confidence-based graph transformation method is indeed due to the removal of spurious edges, the addition of biologically viable ones, and effective weighting of the resultant set of edges.

This detailed analysis of results indicates that association analysis-based graph transformation methods are useful, both from a data mining, as well as a functional genomics perspective.

8. ACKNOWLEDGMENTS

We thank Elena Nabieva and Mona Singh for providing us access to their interaction data and for several fruitful discussions. This work was supported by NSF grants #CRI-0551551, #IIS-0308264, and #ITR-0325949. Access to computing facilities was provided by the Minnesota Supercomputing Institute.

9. REFERENCES

[1] R. Agrawal, T. Imielinski, and A. N. Swami. Mining association rules between sets of items in large databases. In P. Buneman and S. Jajodia, editors, *Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data*, pages 207–216, Washington, D.C., 26–28 1993.

[2] R. Agrawal, H. Mannila, R. Srikant, H. Toivonen, and A. I. Verkamo. Fast discovery of association rules. In *Advances in knowledge discovery and data mining*, pages 307–328. 1996.

[3] R. Agrawal and R. Srikant. Fast algorithms for mining association rules. In *Proc. 20th Int. Conf. Very Large Data Bases, VLDB*, pages 487–499, 1994.

[4] B.-J. Breitkreutz, C. Stark, and M. Tyers. The GRID: the General Repository for Interaction Datasets. *Genome Biology*, 4(3):R23, 2003.

[5] C. Brun, C. Herrmann, and A. Guenoche. Clustering proteins from interaction networks for the prediction of cellular functions. *BMC Bioinformatics*, 5:95, 2004.

[6] C. M. Deane, L. Salwinski, I. Xenarios, and D. Eisenberg. Protein interactions: two methods for assessment of the reliability of high throughput observations. *Mol Cell Proteomics*, 1(5):349–356, 2002.

[7] M. Deng, F. Sun, and T. Chen. Assessment of the reliability of protein–protein interactions and protein function prediction. In *Pac Symp Biocomputing*, pages 140–151, 2003.

[8] L. Ertoz, M. Steinbach, and V. Kumar. Finding clusters of different sizes, shapes, and densities in noisy, high dimensional data. In *Proc. SIAM International Conference on Data Mining*, 2003.

[9] A.-C. Gavin et al. Functional organization of the yeast proteome by systematic analysis of protein complexes. *Nature*, 415(6868):141–147, 2002.

[10] N.J. Krogan et al. Global landscape of protein complexes in the yeast Saccharomyces cerevisiae. *Nature*, 440:637–643, 2006.

[11] R. Gupta, T. Garg, G. Pandey, M. Steinbach, and V. Kumar. Comparative study of various genomic data sets for protein function prediction and enhancements using association analysis. In *SIAM Workshop on Data Mining for Biomedical Informatics*, 2007.

[12] T. Ito, T. Chiba, R. Ozawa, M. Yoshida, M. Hattori, and Y. Sakaki. A comprehensive two-hybrid analysis to explore the yeast protein interactome. *PNAS*, 98(8):4569–4574, 2001.

[13] R. Jarvis and E. Patrick. Clustering using a similarity measure based on shared nearest neighbors. *IEEE Trans. Comput.*, 22:1025–1034, 1973.

[14] P. Legrain, J. Wojcik, and J.-M. Gauthier. Protein–protein interaction maps: a lead towards cellular functions. *Trends in Genetics*, 17(6):346–352, 2001.

[15] C. Lin, D. Jiang, and A. Zhang. Prediction of protein function using common-neighbors in protein-protein interaction networks. In *Proc. IEEE Symposium on BionInformatics and BioEngineering (BIBE)*, pages 251–260, 2006.

[16] E. Nabieva, K. Jim, A. Agarwal, B. Chazelle, and M. Singh. Whole-proteome prediction of protein function via graph-theoretic analysis of interaction maps. *Bioinformatics*, 21(Suppl. 1):i1–i9, 2005.

[17] E. R. Omiecinski. Alternative interest measures for mining associations in databases. *IEEE TKDE*, 15(1):57–69, January/February 2003.

[18] G. Pandey, V. Kumar, and M. Steinbach. Computational approaches for protein function prediction: A survey. Technical Report 06-028, Department of Computer Science and University of Minnesota, October 2006.

[19] P. Pei and A. Zhang. A topological measurement for weighted protein interaction network. In *Proc IEEE Computational Systems Bioinformatics Conference (CSB)*, pages 268–278, 2005.

[20] J. B. Pereira-Leal, A. J. Enright, and C. A. Ouzounis. Detection of functional modules from protein interaction networks. *Proteins*, 54(1):49–57, 2003.

[21] A. Ruepp et al. The FunCat, a functional annotation scheme for systematic classification of proteins from whole genomes. *Nucleic Acids Research*, 32(18):5539–5545, 2004.

[22] L. Salwinski and D. Eisenberg. Computational methods of analysis of protein-protein interactions. *Curr Opin Struct Biology*, 13(3):377–382, 2003.

[23] M. P. Samanta and S. Liang. Predicting protein functions from redundancies in large-scale protein interaction networks. *Proc Natl Acad Sci U.S.A.*, 100(22):12579–12583, 2003.

[24] M. P. Samanta and S. Liang. Predicting protein functions from redundancies in large-scale protein interaction networks. *PNAS*, 100(22):12579–12583, 2003.

[25] B. Schwikowski, P. Uetz, and S. Fields. A network of protein-protein interactions in yeast. *Nature Biotechnology*, 18(12):1257–1261, 2000.

[26] M. Steinbach, P.-N. Tan, H. Xiong, and V. Kumar. Generalizing the notion of support. In *KDD '04: Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 689–694, New York, NY, USA, 2004. ACM Press.

[27] S. Sun, Y. Zhao, Y. Jiao, Y. Yin, L. Cai, Y. Zhang, H. Lu, R. Chena, and D. Bu. Faster and more accurate global protein function assignment from protein interaction networks using the MFGO algorithm. *FEBS Letters*, 580(7):1891–1896, 2006.

[28] P.-N. Tan, M. Steinbach, and V. Kumar. *Introduction to Data Mining*. Addison Wesley, 2006.

[29] P. Uetz et al. A comprehensive analysis of protein–protein interactions in *Saccharomyces cerevisiae*. *Nature*, 403(6770):623–627, 2000.

[30] A. Vazquez, A. Flammini, A. Maritan, and A. Vespignani. Global protein function prediction from protein–protein interaction networks. *Nat Biotechnology*, 21(6):697–700, 2003.

[31] D. B. West. *Introduction to Graph Theory*. Prentice Hall, 2001.

[32] I. Xenarios and D. Eisenberg. Protein interaction databases. *Curr Opin Biotechnology*, 12(4):334–339, 2001.

[33] I. Xenarios, L. Salwinski, X. J. Duan, P. Higney, S.-M. Kim, and D. Eisenberg. DIP, the Database of Interacting Proteins: a research tool for studying cellular networks of protein interactions. *Nucleic Acids Research*, 30(1):303–305, 2002.

[34] H. Xiong, X. He, C. Ding, Y. Zhang, V. Kumar, and S. R. Holbrook. Identification of functional modules in protein complexes via hyperclique pattern discovery. In *Proc. Pacific Symposium on Biocomputing (PSB)*, pages 221–232, 2005.

[35] H. Xiong, G. Pandey, M. Steinbach, and V. Kumar. Enhancing data analysis with noise removal. *IEEE Transactions on Knowledge and Data Engineering*, 18(3):304–319, 2006.

[36] H. Xiong, P.-N. Tan, and V. Kumar. Hyperclique pattern discovery. *Data Min. Knowl. Discov.*, 13(2):219–242, 2006.

[37] G. Yona, W. Dirks, S. Rahman, and D. M. Lin. Effective similarity measures for expression profiles. *Bioinformatics*, 22(13):1616–1622, 2006.