Journal of Bioinformatics and Computational Biology
Vol. 6, No. 1 (2008) 203–222
© Imperial College Press

ClusFCM: AN ALGORITHM FOR PREDICTING PROTEIN FUNCTIONS USING HOMOLOGIES AND PROTEIN INTERACTIONS

CAO NGUYEN
Virginia Commonwealth University, VA 23238, USA
cdnguyen@vcu.edu

MICHAEL MANNINO
University of Colorado Denver
Denver, CO 80217, USA
michael.mannino@cudenver.edu

KATHELEEN GARDINER
University of Colorado Denver
Aurora, CO 80045, USA
katheleen.gardiner@uchsc.edu

KRZYSZTOF J. CIOS
Virginia Commonwealth University, VA 23238, USA
University of Colorado Boulder, Boulder, CO 80309, USA
Polish Academy of Sciences, Poland
kcios@vcu.edu

Received 24 April 2007
Revised 26 September 2007
Accepted 24 October 2007

We introduce a new algorithm, called ClusFCM, which combines techniques of clustering and fuzzy cognitive maps (FCM) for prediction of protein functions. ClusFCM takes advantage of protein homologies and protein interaction network topology to improve low recall predictions associated with existing prediction methods. ClusFCM exploits the fact that proteins of known function tend to cluster together and deduce functions not only through their direct interaction with other proteins, but also from other proteins in the network. We use ClusFCM to annotate protein functions for *Saccharomyces cerevisiae* (yeast), *Caenorhabditis elegans* (worm), and *Drosophila melanogaster* (fly) using protein–protein interaction data from the General Repository for Interaction Datasets (GRID) database and functional labels from Gene Ontology (GO) terms. The algorithm's performance is compared with four state-of-the-art methods for function prediction — Majority, χ^2 statistics, Markov random field (MRF), and FunctionalFlow — using measures of Matthews correlation coefficient, harmonic mean, and area under the receiver operating characteristic (ROC) curves. The results indicate that ClusFCM predicts protein functions with high recall while not lowering precision. Supplementary information is available at www.egr.vcu.edu/cs/dmb/ClusFCM/.

Keywords: Protein functions; protein–protein interactions; Markov random field; clustering; fuzzy cognitive maps.

1. Introduction

One of the main goals of proteomics research is to understand and predict protein functions. Even for a simple organism, such as the baker's yeast (*Saccharomyces cerevisiae*), functions are unknown for approximately one third of the proteins. For more complex organisms, functional annotation is lacking for a much larger fraction of the proteome.

A number of computational methods have been developed for protein function prediction. A general method uses sequence similarity (e.g. BLAST) to first identify homologous proteins in protein databases, and then assign functions to the query protein based on known functions of the matches.[1] A more limited approach, known as Rosetta stone sequence, infers protein interactions from genomic sequences using the observation that some pairs of interacting proteins have homologies in another organism fused into a single protein chain. Many of such protein pairs have been confirmed as functionally related[2]; however, the reliability of such methods is not satisfactorily high. Grouping proteins by correlated evolution, correlated messenger RNA expression patterns, plus patterns of domain fusion has been successfully applied to yeast proteins.[3] Troyanskaya *et al.*[4] used Bayesian reasoning to integrate similarly heterogeneous types of high-throughput biological data for protein function prediction. Zhou *et al.*[5] identified correlations between genes that have similar expression patterns and those that have similar functions. Lewis *et al.*[6] used support vector machines to infer gene functional annotations from a combination of protein sequence and structure data.

Information on protein–protein interactions (PPIs) has been used for function prediction. Proteins interact with each other for a common purpose, and thus the function of an unannotated protein may be annotated by information on the functions of its neighbors in an interaction complex. Recent high-throughput experiments have generated large-scale protein physical interaction data for several organisms such as *Caenorhabditis elegans*,[7] *Drosophila melanogaster*,[8] *Helicobacter pylori*,[9] and *Saccharomyces cerevisiae*.[10–13] Using these datasets, a number of protein function prediction methods have been developed, each having its unique strengths and limitations. The Majority method infers functions for a protein using the most frequent annotations among its nearest neighbors in a PPI network.[11] In this method, some functions may have a very high frequency in the network, but will not be annotated for the query if they do not occur in the nearest neighbor set. Chua *et al.*[14] extended the Majority method to predict protein functions by exploiting indirect neighbors and devising a topological weight to estimate functional similarity. Hishigaki *et al.*[15] predicted protein functions based on χ^2 statistics. They extended the Majority method by looking at all proteins within a specified radius within the network, thus taking into account the frequency of all proteins having a particular function; however, the χ^2 statistics method does not take into account any aspect of the underlying topology of the PPI network. Letovsky and Kasif[16] and Deng *et al.*[17] used a Markov random field (MRF) to assign functions based on a probabilistic analysis of graph neighborhoods in a PPI network. This method assumes that the probability distribution for the annotation of any node is conditionally independent of all other nodes, given its neighbors; the method is also sensitive to the neighborhood size and the parameters of the binomial distribution used in function assignments. FunctionalFlow[18] considers each protein of known function as a source of functional flow for that function. This functional flow spreads through the neighborhoods of the sources, and proteins receiving the highest amount of flow of a function are assigned that function. This algorithm, however, does not consider indirect flow of functions to other proteins after labeling of the functions.

To address the shortcomings of the abovementioned methods, we have developed a hybrid approach that takes into account both the homologies between the proteins and the underlying topology of the PPI network. First, we assign biological homology scores to the edges in a PPI network. Next, we use agglomerative clustering on the weighted graphs to cluster the proteins by known function and cellular location. Then, we treat each cluster as a fuzzy cognitive map (FCM), with proteins constituting causal nodes and edges representing cause–effect relationships. For a protein of interest, we simulate effects of proteins having a function to the protein in the FCM to which the protein belongs. We tested the ClusFCM algorithm on PPI networks for yeast, worm and fly, and compared its performance with Majority, χ^2 statistics, MRF, and FunctionalFlow methods.

2. Methods

2.1. *Materials*

Even though there exist programs that take advantage of Gene Ontology (GO) and homology (Blast2GO,[19] GOblet,[20] GOtcha,[21] OntoBlast[22]) to prepare data for experiments, we decided to use our own procedure as described below.

2.1.1. *GO annotations*

GO[23] (http://www.geneontology.org) is composed of three related ontologies: the molecular function of gene products, their associated biological processes, and their physical structure as cellular components. Each ontology is constructed as a directed acyclic graph. We consider functional annotation only at the third level of the GO hierarchies. After converting protein labels through *is-a* ancestors of each label, there are a total of 39 GO terms, consisting of 18 molecular functions, 10 cellular component functions, and 11 biological process functions (for details, see Table S1 in Supplementary Information).

2.1.2. *Physical interactions*

(1) *Yeast interaction data*
The yeast General Repository for Interaction Datasets (GRID)[24] database (release 3/2006) contains 20,519 interaction pairs. Of the 4,948 proteins participating in interactions, 4,600 proteins have been labeled with GO terms taken from the yeast genome database at http://www.yeastgenome.org/.

(2) *Worm interaction data*
There are 2,780 unique proteins participating in 4,453 distinct interactions in the worm GRID database (release 3/2006). Of these, 1,680 proteins are labeled with the GO terms taken from the worm genome database (release WS155) at http://www.wormbase.org/.

(3) *Fly interaction data*
The fly GRID database (release 3/2006) includes 28,406 distinct interactions. Of the 7,938 proteins participating in interactions, 5,098 of these are annotated with GO terms taken from the fly genome database at http://www.flybase.org/ (for details, see Table S2 in Supplementary Information).

2.2. *ClusFCM algorithm*

The ClusFCM algorithm works in two stages. In stage 1, it performs clustering; and in stage 2, it performs function flow in a FCM. At both stages, a PPI network is used.

2.2.1. *PPI network definition*

The PPI network is an undirected graph $G = (V, E)$, where V is a set of proteins and E is a set of edges connecting proteins u and v if the corresponding proteins interact physically. We take advantage of homologies between two proteins for the extraction of functions by assigning homology scores to the edges between two interacting proteins. When a protein is more homologous with another protein (in an interacting edge), it has a better chance to be annotated by the functions of its partner. We use the *bl2seq* program[25] to extract the expected values, in the range from 0 to 10, of the strength of the biological relationship between all interacting proteins. The lower the expected value, the higher the possibility the two proteins share the same functions.[1] We observe that, for all three datasets, interacting proteins are more likely to be involved in the same functions if they are more homologous (see Table 1). We use the following notation:

N = total number of proteins in the graph
P_u = protein u $(u = 1, 2, \ldots, N)$
$\mathrm{Nei}(u)$ = set of proteins interacting directly with protein P_u
$w_{u,v}$ = weight assigned to the edges connecting proteins P_u and P_v

Table 1. Average number of common functions of interacting proteins.

	E-value $\leq$ 1E-3	E-value $>$ 1E-3
Yeast	5.28	4.27
Worm	4.02	2.38
Fly	5.75	3.78

where

$$w_{u,v} = \begin{cases} \text{the expected value, if } P_u \text{ and } P_v \text{ directly interact} \\ \infty, \text{ otherwise.} \end{cases}$$

2.2.2. *Stage 1*

Based on the fact that proteins of known function and cellular location tend to cluster together in a PPI network,[11] clustering analysis is obviously a good approach for the extraction of functions from the PPI network. Clustering means assigning objects that share similar properties into the same groups. We use agglomerative clustering[26] on the weighted graph G because it generates many small clusters that facilitate assignment of functions. Initially, we treat each node in the graph as its own cluster, and then merge the two nearest clusters into a new cluster. The distance between clusters C_i and C_h is calculated as $d(C_i, C_h) = \min\{w_{u,v} \text{ with } P_u \in C_i, P_v \in C_h\}$ in a single-linkage method manner. A cluster is excluded from the agglomeration process using the following heuristic rules:

$$\text{the cardinality of the cluster exceeds a threshold } \tau, \text{ and} \tag{1}$$

$$\text{the entropy function of the cluster } H = \frac{-\sum_{s=1}^{c} p_s \log_2 p_s}{\log_2 c} < \gamma, \tag{2}$$

where c is the number of function categories in the cluster and p_s is the relative frequency of category s in the cluster. The merging process stops if no further new clusters are formed. The following pseudocode provides more details:

Algorithm 1: Agglomerative cluster in stage 1
Input: *Matrix* $W = \{w_{u,v}\}$.

```
(1) Initialize distance matrix $D = \{d_{u,v}\}$, with $d_{u,v} = w_{u,v}$ ($u = 1, 2, \ldots, N$, $v = 1, 2, \ldots, N$).
(2) Initialize $N$ clusters corresponding to $N$ proteins.
(3) Find the smallest distance in $D$, and merge the corresponding clusters (say, $C_i$ and $C_h$) to get a new cluster ($C_iC_h$).
```

(4) Recompute the distance between (C_iC_h) and any other cluster C_k in matrix D by

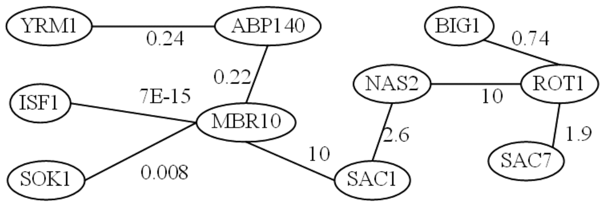
$$d((C_iC_h), C_k) = \begin{cases} \infty, \text{ if the cluster } (C_iC_h) \text{ satisfies rules (1) and (2)} \\ \min\{d(C_i, C_k), d(C_h,\ C_k)\}, \text{ otherwise.} \end{cases}$$

(5) If $D = \{\infty\}$, then stop; otherwise, return to step 3.

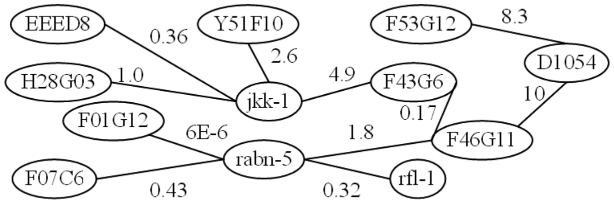
Output: *Cluster sets.*

To determine the values of τ and γ, we initially set them to $\tau = \frac{N}{300}$ and $\gamma = 0.6$, and then gradually increase the cardinality of each cluster as follows: $\frac{N}{290}, \frac{N}{280}$, and so on. When the number of proteins in a cluster is increased, the entropy function of the cluster is also increased. Thus, for each increment of cluster cardinality, we also adjust the value of γ in increments of 0.01. We select the thresholds that generate the best results in the next stage. In our study, the values of τ and γ are experimentally determined to be $\frac{N}{50}$ and 0.85, respectively.

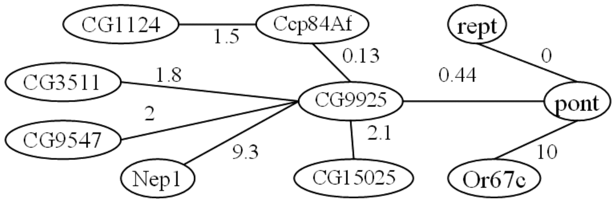
Figure 1 shows clusters generated from this stage for the yeast, worm, and fly datasets. The figure indicates that the method takes into account not only the



(a) A cluster generated by ClusFCM in yeast network.



(b) A cluster generated by ClusFCM in worm network.



(c) A cluster generated by ClusFCM in fly network.

Fig. 1. Clusters generated by the ClusFCM algorithm using (a) yeast, (b) worm, and (c) fly data.

homologies (distances), but also the functions shared among the proteins. For example, although the yeast proteins NAS2 and ROT1 are not homologous, they share GO terms for the same functions [physiological process (GO: 0007582), cell (GO: 0005623), and cellular process (GO: 0009987)]; thus, they are put together in the same cluster. Similarly, the worm proteins jkk-1 and F43G6 share binding molecular function (GO: 0005488); while the fly proteins pont and Or67c share binding (GO: 0005488), physiological process (GO: 0007582), and response to stimulus (GO: 0050896) functions.

2.2.3. *Stage 2*

Each cluster is now considered as a FCM.[27,28] A PPI network is a dynamic system where proteins, through interacting edges, flow their functions to other proteins. We use the FCM to simulate how likely a protein is annotated with a function from its neighbors and, in turn, how the newly annotated protein propagates the function of interest through the network. In the terminology of the FCMs, the nodes (proteins) indicate causal concepts and the edges (interactions) indicate the cause–effect relationships (Fig. 2). For a function of interest, we denote $S = (s_1, s_2, \ldots, s_N)$ as an instantaneous state vector, where $s_u = 1$ (on) if the protein u is annotated or predicted with the function and $s_u = 0$ (off) otherwise. An unannotated protein u is switched on if rule (3) is satisfied:

$$f(x) = \frac{1}{1 + e^{-x}} > \theta, \tag{3}$$

where $x = \frac{\sum_{v \in \mathrm{Nei}(u)} (10 - w_{v,u}) s_v}{\sum_{v \in \mathrm{Nei}(u)} (10 - w_{v,u})}$ and θ is an activation threshold (explained later).

Starting with an initial state vector, we iteratively use rule (3) on all unannotated proteins until the equilibrium state (hidden pattern) is reached. Before the FCM conducts its task, we make an initial labeling (using uniform distribution) of the functions that have the highest scores for each unannotated protein u in the network: $|\mathrm{Nei}(u)| \times \pi_a$, where $|\mathrm{Nei}(u)|$ is the number of neighborhoods of the protein u and π_a is the fraction of a function a in the entire network. The algorithm for prediction of a function of interest is outlined below.

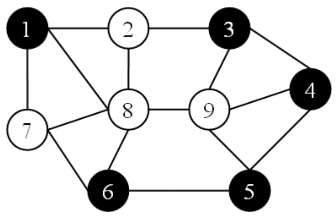


Fig. 2. In this FCM, proteins shown in black are annotated. The initial state vector is (101111000).

Algorithm 2: Finding the hidden pattern in stage 2
Input: A cluster C and an initial state vector.

(1) Set the initial state vector $S_0 = (s_{0,1}, s_{0,2}, \ldots, s_{0,|C|})$ where $|C|$ = cardinality of cluster C, and

$$S_{0,u} = \begin{cases} 1, & \text{if protein } u \text{ is annotated or predicted with the function of interest} \\ 0, & \text{otherwise.} \end{cases}$$

(2) Apply rule (3) for each protein in cluster C.
(3) If $S_i = S_{i-1}$, then S_i is the hidden pattern, otherwise, return to step 2.

Output: The hidden pattern of the cluster.

2.3. *Description of methods used in comparisons*

We perform a head-to-head comparison of performance of the ClusFCM algorithm with four methods: Majority,[11] χ^2 statistics,[15] MRF,[16] and FunctionalFlow.[18]

2.3.1. *Majority*

For each protein in a set, we first assume that it is unannotated and then we count the number of times each function occurs in its nearest neighbors. The functions with the highest frequencies are assigned to the query protein.

2.3.2. χ^2 *statistics*

First, for each function a in the 39 GO terms, we derive the fraction π_a = (number of proteins having function a/N). Then, we calculate n_a as the number of proteins among the query neighbors that have the function a, and e_a as the expected number (e_a = number of proteins in its nearest neighbors $\times$ π_a). The query protein is annotated with the function with the highest χ^2 value among the functions of all proteins in its nearest neighborhood, where $\chi^2 = (n_a - e_a)^2/e_a$.

2.3.3. *FunctionalFlow*

We implemented this method based on Nabieva *et al.*[18] using the following notation:

- $R_t^a(u)$ is the amount of functional flow in the reservoir for function a that node u has at time t. Initially ($t = 0$), $R_0^a(u) = \infty$ if node u is annotated with a; otherwise, 0. At time t,

$$R_t^a(u) = R_{t-1}^a(u) + \sum_{v:(u,v)\in E} (g_t^a(v,u) - g_t^a(u,v)).$$

- $g_t^a(u,v)$ is the flow of function a at time t from protein u to protein v. Initially $(t = 0)$, $g_0^a(u,v) = 0$; otherwise, 0. At time t, $g_t^a(u,v) = \min(1, \frac{R_t^a(u)}{\sum_{(u,y)\in E}})$ if $R_{t-1}^a(u) \geq R_{t-1}^a(v)$.
- $fa(u)$ is the functional score for node u and function a over d iterations, calculated as the total amount of flow that has entered node u:

$$f_a(u) = \sum_{t=1}^{d} \sum_{v:(u,v)\in E} g_t^a(v,u). \tag{4}$$

This algorithm is iteratively run d times, where d is set to half the diameter of the interaction network, i.e. 6 for the yeast, 7 for the worm, and 6 for the fly physical interaction networks. Functions which have the highest scores (Eq. 4) for a protein are assigned to that protein.

2.3.4. *MRF*

This algorithm exploits the difference between two probabilities:

- p_1^a is the probability that two nodes in an interacting pair have the same function a.
- p_0^a is the probability that two nodes in an interacting pair do not share the same function.

Using the Markov assumption, the probability distribution for the annotating function a of node u, L_u^a, is conditionally independent of its neighbors. To derive the probability that protein u has function a in condition with its neighbors Nei(u) and the number of neighbors which are annotated with function a, n_u^a, Bayes' rule is used:

$$P(L_u^a|\text{Nei}(u), n_u^a) = \frac{P(n_u^a|L_u^a, \text{Nei}(u)) \cdot P(L_u^a)}{P(n_u^a|\text{Nei}(u))}, \tag{5}$$

where $P(L_u^a) = f_a$ or the frequency of function a in the network.

The probability of having n proteins labeled with function a in the neighbors Nei(u) in the context of protein u having function a is expected to follow a binomial distribution:

$$P(n_u^a|L_u^a, \text{Nei}(u)) = B(\text{Nei}(u),\ n_u^a, p_1^a), \quad \text{with } B(N,k,p) = \binom{N}{k} (p)^k (1-p)^{N-k}.$$

The probability of having n proteins labeled with function a in the neighbors Nei(u) is

$$\begin{aligned} P(n_u^a|\text{Nei}(u)) &= f_a \cdot P(n_u^a|L_u^a, \text{Nei}(u)) + \overline{f_a} \cdot P(n_u^a|\overline{L_u^a}, \text{Nei}(u)) \\ &= f_a \cdot B(\text{Nei}(u), n_u^a, p_1^a) + \overline{f}_a \cdot B(\text{Nei}(u), n_u^a, p_0^a). \end{aligned}$$

Thus, Eq. (5) becomes

$$P(L_u^a|\mathrm{Nei}(u), n_u^a) = \frac{f_a \cdot B(\mathrm{Nei}(u), n_u^a, p_1^a)}{f_a \cdot B(\mathrm{Nei}(u), n_u^a, p_1^a) + \overline{f}_a \cdot B(\mathrm{Nei}(u), n_u^a, p_0^a)}. \tag{6}$$

Equation (6) is iteratively applied for each unannotated protein u in the network separately for each function a. If the probability in Eq. (6) exceeds a given threshold, the protein u is reclassified with the function a. The process stops when no further labeling occurs.

2.4. *Assessment of predictions*

We use the leave-one-out cross-validation procedure to evaluate predictions resulting from each method. For each protein in the datasets, we assume that it is unannotated. Then, we use each of the above methods to predict the protein functions. Let A be the annotated function set, P be the predicted function set, and F be the whole 39 GO function collection set. We calculate the number of true positives (TP), false positives (FP), true negatives (TN), and false negatives (FN) as follows:

- TP (the number of predicted functions that are actually annotated) = $|A \cap P|$
- FP (the number of predicted functions that are not annotated) = $|P \backslash A|$
- FN (the number of annotated functions that are not predicted) = $|A \backslash P|$
- TN (the number of not-predicted functions that are actually not annotated) = $|F \backslash \{A \cup P\}|$.

This calculation, used in information retrieval[29] (where A, the annotated function set, is the relevant document; and P, the predicted function set, is the retrieved document), is illustrated in Fig. 3.

The following measures are used for assessing the performance of ClusFCM:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} = \frac{|A \cap P|}{P}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} = \frac{|A \cap P|}{A}$$

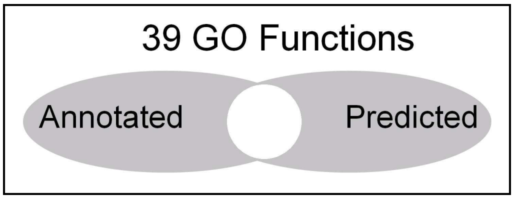


Fig. 3. TPs are represented by the white circle (intersection), FPs are shown as the gray area on the right, FNs are shown as the gray area on the left, and TNs are the rest of the entire area.

$$\text{Matthews correlation coefficient (MCC)} = \frac{\text{TP} \times \text{TN} - \text{FP} \times \text{FN}}{\sqrt{(\text{TP} + \text{FN}))(\text{TP} + \text{FP})(\text{TN} + \text{FP})(\text{TN} + \text{FN})}}$$

$$\text{Harmonic mean (HM)} = \frac{2}{1/\text{Precision} + 1/\text{Recall}}$$

Precision (a.k.a. positive predictive value) is the probability of correctly predicting a function ($|A \cap P|/P$), while recall (a.k.a. sensitivity) is the probability that a function prediction is correct ($|A \cap P|/A$). HM combines precision and recall into a single number ranging from 0 to 1. If HM is 0, then no annotated proteins have been predicted; if HM is 1, then all predicted functions are correct.[29] The MCC value is between -1 and $+1$, and measures how well the predicted class labels agree with the actual class labels. An MCC value of -1 (when TP $= 0$ and TN $= 0$) means complete disagreement, an MCC value of $+1$ (when FP $= 0$ and FN $= 0$) means complete agreement, and an MCC value of 0 (when TP $=$ FP and TN $=$ FN) means that the prediction is random.[30] We did not use accuracy because it is sensitive to the distribution of functions among proteins and our data is highly skewed (negative class $\gg$ positive class). Thus, if we sacrificed true positives to predict all examples as negative, we could have obtained high accuracy (but not correct) of a classifier.[31–33]

3. Results and Discussion

ClusFCM and the other four methods are implemented in Java and tested on the three datasets. Table 2 shows the performance of our algorithm. The relationship between precision and recall is shown in Fig. 4, using different thresholds in stage 2. To determine the values of the threshold θ in rule (3), we use the algorithm with different values of θ in the range from 0.5 to 1, in increments of 0.01. The value of θ for which the algorithm yields the highest MCC value is used. In this case, the chosen thresholds are 0.64 for yeast, 0.74 for worm, and 0.64 for fly data.

The results of the four methods using leave-one-out cross-validation are shown in Table 3(a). For the Majority, FunctionalFlow, and χ^2 statistics methods, we select the top ten functions having the highest scores and assign these functions to each unannotated protein. For the MRF method, we use threshold values from 0 to 1, in increments of 0.1, as used in Eq. (6), to predict the functions. For each method, we

Table 2. Performance measures of ClusFCM on three datasets using leave-one-out validation.

	Yeast	Worm	Fly
Precision	0.63	0.33	0.34
Recall	0.70	0.70	0.73
MCC	0.61	0.43	0.44
HM	0.66	0.45	0.47

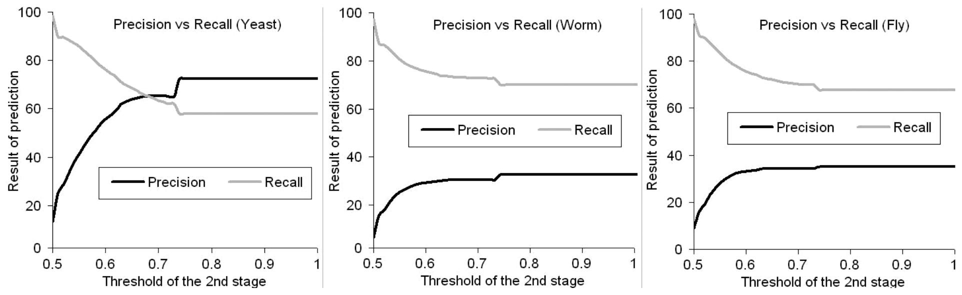


Fig. 4. Results of ClusFCM prediction for different thresholds in the three datasets.

choose the threshold which yields the highest MCC value. Interestingly, we found that with the selected thresholds, the HM measure also achieves the highest value for each method.

As shown, ClusFCM outperforms the other methods in terms of recall, MCC, and HM measures. The results of χ^2 statistics, when compared with Majority and FunctionalFlow, are worse on the yeast data, but better on the worm and fly data; this is because the fraction of proteins having more than one neighbor is smaller in the fly and worm datasets, prominently showing that the underlying topology of the PPI network plays a crucial role. FunctionalFlow performs similarly to Majority on the three datasets; these methods, however, do not take into account indirect effects in which a protein, after being annotated, is used to annotate other proteins. MRF performs well on the yeast dataset, where almost all proteins are annotated; but in the worm and fly datasets, ClusFCM performs significantly better in terms of recall, MCC, and HM. Because the results for ClusFCM and MRF are similar, we provide additional analysis using a bootstrap percentile test.

Several studies have addressed the correlation between molecular functions and homologies of proteins (Altschul *et al.*,[1] Blast2GO,[19] GOblet,[20] GOtcha,[21] OntoBlast,[22] GOFigure,[34] GeneQuiz[35]). In addition, some studies have shown that molecular functions are predictive of cellular components.[36] On the other hand, it is also known that protein–protein interactions are more reflective of biological processes. There are 28 molecular function and cellular component GO terms versus 11 biological processes in our study. Statistically, in our datasets, the average number of neighboring proteins sharing the same biological process GO term with a protein (6.6, 1.5, and 4.2 in yeast, worm, and fly datasets, respectively) is higher than that of proteins sharing the same molecular function or cellular component GO term (5.3, 1.0, and 2.5 in yeast, worm, and fly datasets, respectively). Hence, in such a context, we ask whether the ClusFCM performs well in predicting only biological processes. To determine to what extent proteins are correctly annotated as biological processes (as well as molecular functions and cellular components), we use ClusFCM and the other four methods to predict those functions in yeast, worm, and fly interaction datasets; we could not modify other methods used in the

Table 3(a). Performance measures of ClusFCM and the other methods in yeast, worm, and fly interaction networks using the leave-one-out cross-validation test.

	ClusFCM	Majority	χ^2 statistics	FunctionalFlow	MRF
			Yeast network		
Precision	0.63	0.63	0.27	0.64	0.72
Recall	0.70	0.45	0.53	0.44	0.57
MCC	0.61	0.47	0.24	0.47	0.59
HM	0.66	0.52	0.35	0.52	0.64
			Worm network		
Precision	0.33	0.21	0.21	0.28	0.29
Recall	0.70	0.37	0.71	0.34	0.63
MCC	0.43	0.22	0.32	0.26	0.38
HM	0.45	0.27	0.32	0.31	0.40
			Fly network		
Precision	0.34	0.28	0.20	0.34	0.34
Recall	0.73	0.32	0.63	0.28	0.55
MCC	0.44	0.24	0.25	0.25	0.37
HM	0.47	0.30	0.30	0.31	0.42

Table 3(b). Prediction of biological process GO terms[a] by ClusFCM and the other methods in yeast, worm, and fly interaction networks using a leave-one-out cross-validation test.

	ClusFCM	Majority	χ^2 statistics	FunctionalFlow	MRF
			Yeast network		
Precision	0.66 (0.61)	0.66 (0.60)	0.28 (0.25)	0.66 (0.62)	0.72 (0.71)
Recall	0.75 (0.66)	0.54 (0.39)	0.56 (0.51)	0.53 (0.38)	0.68 (0.50)
MCC	0.64 (0.59)	0.52 (0.43)	0.20 (0.25)	0.51 (0.44)	0.64 (0.56)
HM	0.71 (0.63)	0.59 (0.47)	0.38 (0.34)	0.59 (0.47)	0.70 (0.59)
			Worm network		
Precision	0.39 (0.28)	0.34 (0.15)	0.25 (0.19)	0.34 (0.23)	0.37 (0.24)
Recall	0.82 (0.61)	0.44 (0.32)	0.72 (0.71)	0.44 (0.26)	0.75 (0.55)
MCC	0.51 (0.38)	0.32 (0.16)	0.33 (0.30)	0.32 (0.21)	0.47 (0.31)
HM	0.53 (0.39)	0.38 (0.21)	0.37 (0.30)	0.38 (0.25)	0.50 (0.33)
			Fly network		
Precision	0.47 (0.27)	0.44 (0.17)	0.23 (0.18)	0.43 (0.20)	0.46 (0.25)
Recall	0.77 (0.69)	0.46 (0.21)	0.64 (0.62)	0.47 (0.12)	0.70 (0.41)
MCC	0.53 (0.38)	0.37 (0.13)	0.22 (0.25)	0.36 (0.12)	0.48 (0.28)
HM	0.58 (0.39)	0.45 (0.19)	0.33 (0.27)	0.45 (0.15)	0.55 (0.31)

[a]Predicted performance measures of molecular functions and cellular components are shown in parentheses.

comparison to work with our weighted graphs because these methods are not able to use this information. The results, shown in Table 3(b), indicate that even in the case of prediction of biological processes, which are less influenced by homologies in the PPI networks than other GO categories, the performance of ClusFCM is significantly better.

We note that functional annotations for the proteins are incomplete at present. In the yeast database, there are 348 (7%) unannotated proteins out of 4,948 proteins that participate in interactions; this fraction is approximately 40% for worm proteins and 35% for fly proteins. Therefore, a protein may have a function that has not yet been experimentally verified. We wish to decrease the number of annotated functions that are not predicted, and increase the number of predicted functions that are actually annotated. The fact that recall values are always higher than precision values in all datasets increases confidence in our methods. Table 4 shows the number of TPs and FNs predicted by each method. ClusFCM identifies more TPs and fewer FNs than any other method over the three datasets, except for the worm network where χ^2 statistics has higher TPs and lower FNs than ClusFCM.

To visualize the performance of ClusFCM, we use receiver operating characteristic (ROC) curves, which plot the TP rate [TP/(TP + FN)] against the FP rate [FP/(FP + TN)] for different thresholds.[37] For comparison, we also show the ROC curves for the Majority, χ^2 statistics, FunctionalFlow, and MRF methods by different top-scoring functions (from 1 to 10) and different thresholds (from 0 to 1, in increments of 0.1), as shown in Fig. 5. The closer the curve follows the top left-hand area of the ROC space, the more accurate the classifier. A random classifier would have its ROC curve lying along the diagonal line connecting points (0, 0) and (1, 1). We see that the performance of ClusFCM is better than those of Majority, χ^2 statistics, and FunctionalFlow. The values of areas under the ROC curve (AUC) for all classifiers are shown in Table 5.

Next, we analyze the predictive power of ClusFCM and the other four methods by using leave-one-out cross-validation on proteins having at least two neighbors, at

Table 4. The number of TPs, FPs, TNs, and FNs of ClusFCM and the other methods on the yeast, worm, and fly physical interaction networks using a leave-one-out test on the 39 protein functions.

	ClusFCM	Majority	χ^2 statistics	FunctionalFlow	MRF
			Yeast network (4948 × 39 data points)		
TP	17,411	11,213	13,178	11,035	14,302
FP	10,195	6,658	36,302	6,316	5,671
TN	157,792	161,329	131,685	161,671	162,316
FN	7,574	13,772	11,807	13,950	10,683
			Worm network (2780 × 39 data points)		
TP	4,539	2,409	4,603	2,190	4,104
FP	9,311	9,108	17,567	5,656	9,992
TN	92,629	92,832	84,373	96,284	91,948
FN	1,941	4,071	1,877	4,290	2,376
			Fly network (7938 × 39 data points)		
TP	18,072	8,099	1,5776	7,032	1,3608
FP	34,555	20,516	64,736	13,780	26,040
TN	250,104	264,143	219,923	270,879	258,619
FN	6,851	16,824	9,147	17,891	11,315

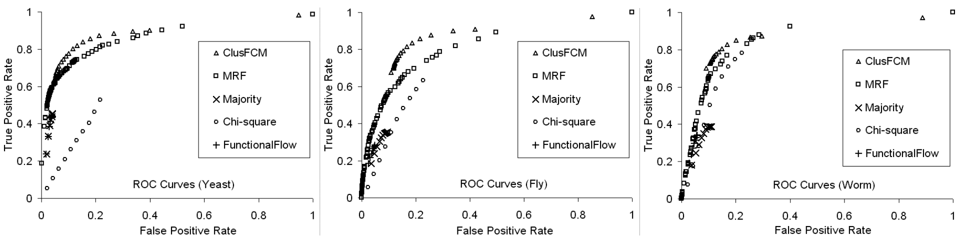


Fig. 5. Comparison of ClusFCM results with four other methods on three datasets.

Table 5. The AUC values of ClusFCM and the other methods in yeast, worm, and fly interaction networks.

	ClusFCM	Majority	χ^2 statistics	FunctionalFlow	MRF
Yeast	0.89	0.62	0.63	0.62	0.87
Worm	0.86	0.61	0.72	0.60	0.86
Fly	0.86	0.61	0.67	0.59	0.82

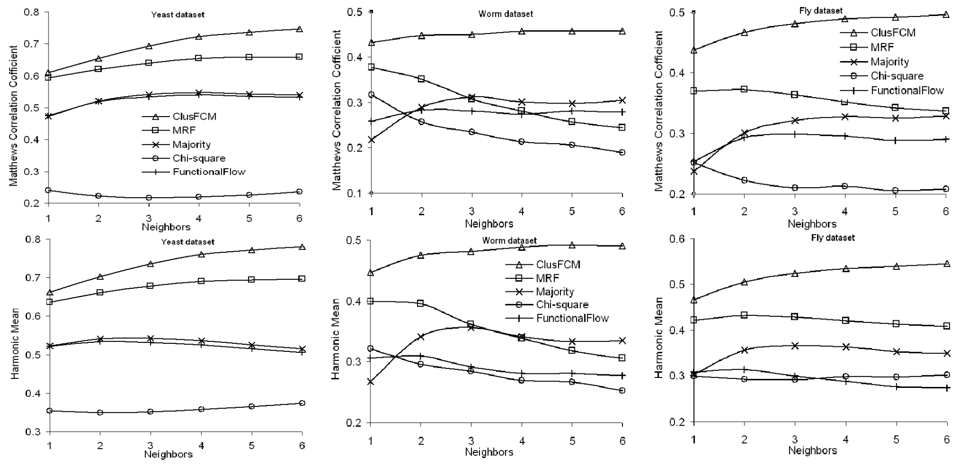


Fig. 6. Performance of ClusFCM and the other methods on the yeast, worm, and fly interaction networks using the leave-one-out procedure for proteins having at least one neighbor, two neighbors, etc.

least three neighbors, and so on. The results show that the performance of ClusFCM improves as the number of neighbors increases (Fig. 6). Moreover, all performance measures for ClusFCM increase monotonically over all datasets. This important characteristic is absent in the other four methods.

To gauge the uncertainty of the algorithms' performance, we next use the bootstrap percentile test[38–40] to calculate confidence intervals. Bootstrapping is a nonparametric technique involving a large number of repetitive computations:

Table 6. 95% confidence intervals for the difference in MCC, HM, and AUC values between ClusFCM and the four methods over 1,000 bootstrap datasets.

		Majority		χ^2 statistics		FunctionalFlow		MRF	
		low	up	low	up	low	up	low	up
Yeast	MCC	0.136041	0.136465	0.370374	0.370917	0.135682	0.136096	0.016199	0.016530
	HM	0.139726	0.139119	0.308030	0.308444	0.140698	0.141098	0.025778	0.026078
	AUC	0.259770	0.259997	0.256976	0.257223	0.262113	0.262330	0.016500	0.016841
Worm	MCC	0.215037	0.215894	0.116154	0.116763	0.173820	0.174690	0.055099	0.055582
	HM	0.178593	0.179347	0.124988	0.125462	0.140512	0.141335	0.047477	0.047882
	AUC	0.244011	0.244480	0.141463	0.141948	0.260912	0.261352	−0.002379	−0.001815
Fly	MCC	0.199280	0.199709	0.185497	0.185852	0.182837	0.183273	0.066934	0.067247
	HM	0.163064	0.163491	0.166621	0.166905	0.158066	0.158527	0.044335	0.044620
	AUC	0.251137	0.251342	0.191224	0.191451	0.267226	0.267428	0.039924	0.040251

from each protein dataset of size N, we sample $n = 50/\alpha$ datasets of size N (with replacement), where α is the confidence level of the test. With a confidence level of 95%, we have 1,000 datasets for each of three organisms: yeast, worm, and fly. For each dataset i, we compute the difference in the performance measures $d_{\mathrm{MCC},i}$, $d_{\mathrm{HM},i}$, and $d_{\mathrm{AUC},i}$ between our method and the other four methods. These 1,000 values represent the nonparametric distribution of the random variables $d_{\mathrm{MCC}}, d_{\mathrm{HM}}$, and d_{AUC}. We consider a confidence interval $[\mathrm{low}_{ms}, \mathrm{up}_{ms}]$ such that $p(\mathrm{low}_{ms} < d_{ms} < \mathrm{up}_{ms}) = 1 - \alpha$ centered around the mean of $p(d_{ms})$, where ms is the MCC, HM, or AUC measure. Formally, with a confidence level of 95%, the lower bound $\mathrm{low}_{ms} = \Phi\left(z_0 + \frac{z_0+z^{0.025}}{1-a(z_0+z^{0.025})}\right)$ and the upper bound $\mathrm{up}_{ms} = \Phi\left(z_0 + \frac{z_0+z^{0.975}}{1-a(z_0+z^{0.0975})}\right)$, where Φ is the standard normal cumulative distribution function, z_α is the αth quantile of standard normal distribution, z_0 is the bias correction, and a is the acceleration. The null hypothesis, that both algorithms perform equally well, can be rejected if 0 lies outside the interval. Table 6 shows confidence intervals for ClusFCM and the other methods over 1,000 bootstrap datasets for yeast, worm, and fly. The intervals show that ClusFCM performs significantly better in terms of the MCC, HM, and AUC (except for the similarity of the AUC measure in the worm dataset for MRF).

4. Conclusions

We have developed and extensively tested the ClusFCM algorithm for protein function predictions based on protein interaction networks. The algorithm uses several characteristics of interaction networks: direct and indirect interactions, underlying topology networks, clusters, and edges weighted by homology measures between the interacting proteins. It takes into consideration both the protein–protein interaction network and homologies between interacting proteins. We have shown ClusFCM's robustness by testing it on the curated interaction data from the GRID database using the leave-one-out cross-validation method. In addition, we used the bootstrap percentile test for establishing statistical significance of the results. The results show that ClusFCM outperforms the Majority, χ^2 statistics, FunctionalFlow, and MRF methods. In the future, we plan to modify and extend the association-rule algorithm to discover relationships between functions of a protein and its neighborhood in order to assign possibly new functions to the protein.

Acknowledgments

The authors acknowledge support from the Vietnamese Ministry of Education and Training (C. Nguyen), the National Institutes of Health (HD047671 and HD49460) (K. Gardiner), and the Butcher Foundation (K. J. Cios and C. Nguyen).

References

1. Altschul S, Madden T, Schaffer A, Zhang J, Zhang Z, Miller W, Lipman D, Gapped BLAST and PSI-BLAST: A new generation of protein database search programs, *Nucleic Acids Res* **25**:3389–3402, 1997.
2. Marcotte E, Pellegrini M, Ng H, Rice D, Yeates T, Eisenberg D, Detecting protein function and protein–protein interactions from genome sequences, *Science* **285**:751–753, 1999.
3. Marcotte E, Pellegrini M, Thompson M, Yeates T, Eisenberg D, A combined algorithm for genome-wide prediction of protein function, *Nature* **402**:83–86, 1999.
4. Troyanskaya O, Dolinski K, Owen A, Altman R, Botstein D, A Bayesian framework for combining heterogeneous data sources for gene function prediction (in *Saccharomyces cerevisiae*), *Proc Natl Acad Sci USA* **100**:8348–8353, 2003.
5. Zhou X, Kao M, Wong WH, Transitive functional annotation by shortest-path analysis of gene expression data, *Proc Natl Acad Sci USA* **99**:12783–12788, 2002.
6. Lewis D, Jebara T, Noble W, Support vector machine learning from heterogeneous data: An empirical analysis using protein sequence and structure, *Bioinformatics* **22**:2753–2760, 2006.
7. Li S *et al.*, A map of the interactome network of the metazoan *C. elegans*, *Science* **303**:540–543, 2004.
8. Giot L *et al.*, A protein interaction map of *Drosophila melanogaster*, *Science* **302**:1727–1736, 2003.
9. Rain J *et al.*, The protein–protein interaction map of *Helicobacter pylori*, *Nature* **409**:211–215, 2001.
10. Fromont-Racine M, Rain JC, Legrain P, Toward a functional analysis of the yeast genome through exhaustive two-hybrid screens, *Nat Genet* **16**:277–282, 1997.
11. Schwikowski B, Uetz P, Fields S, A network of protein–protein interactions in yeast, *Nat Biotechnol* **18**:1257–1261, 2000.
12. Uetz P *et al.*, A comprehensive analysis of protein–protein interactions in *Saccharomyces cerevisiae*, *Nature* **403**:623–627, 2000.
13. Ho Y *et al.*, Systematic identification of protein complexes in *Saccharomyces cerevisiae* by mass spectrometry, *Nature* **415**:180–183, 2002.
14. Chua H, Sung W, Wong L, Exploiting indirect neighbours and topological weight to predict protein function from protein–protein interactions, *Bioinformatics* **22**:1623–1630, 2006.

15. Hishigaki H, Nakai K, Ono T, Tanigami A, Takagi T, Assessment of prediction accuracy of protein function from protein–protein interaction data, *Yeast* **18**:523–531, 2001.
16. Letovsky S, Kasif S, Predicting protein function from protein/protein interaction data: A probabilistic approach, *Bioinformatics* **19**:197–204, 2003.
17. Deng M, Zhang K, Mehta S, Chen T, Sun F, Prediction of protein function using protein–protein interaction data, *J Comput Biol* **10**:947–960, 2003.
18. 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**:302–310, 2005.
19. Conesa A, Gotz S, Garcia-Gomez J, Terol J, Talon M, Robles M, Blast2GO: A universal tool for annotation, visualization and analysis in functional genomics research, *Bioinformatics* **21**:3674–3676, 2005.
20. Groth D, Lehrach H, Hennig S, GOblet: A platform for Gene Ontology annotation of anonymous sequence data, *Nucleic Acids Res* **32**:W313–W317, 2004.
21. Martin D, Berriman M, Barton G, GOtcha: A new method for prediction of protein function assessed by the annotation of seven genomes, *BMC Bioinformatics* **5**:178–195, 2004.
22. Zehetner G, OntoBlast function: From sequence similarities directly to potential functional annotations by ontology terms, *Nucleic Acids Res* **31**:3799–3803, 2003.
23. Ashburner M *et al.*, Gene Ontology: Tool for the unification of biology. The Gene Ontology Consortium, *Nat Genet* **25**:25–29, 2000.
24. Breitkreutz B, Stark C, Tyers M, The GRID: The General Repository for Interaction Datasets, *Genome Biol* **4**(3):R23, 2003.
25. Tatusova A, Madden T, Blast 2 sequences — A new tool for comparing protein and nucleotide sequences, *FEMS Microbiol Lett* **174**:247–250, 1999.
26. Jain K, Murty N, Flynn J, Data clustering: A review, *ACM Comput Surv* **31**:264–323, 1999.
27. Kosko B, Fuzzy cognitive maps, *Int J Man Mach Stud* **24**:65–75, 1986.
28. Fletcher D, Nguyen D, Cios K, Autonomous synthesis of fuzzy cognitive maps from observational data, *IEEE Aerospace Conf Proc*, Big Sky, MT, pp. 1–9, 2005.
29. van Rijsbergen C, Information retrieval: Theory and practice, *Proceedings of the Joint IBM/University of Newcastle upon Tyne Seminar on Data Base Systems*, pp. 1–14, 1979.
30. Baldi P, Brunak S, Chauvin Y, Andersen C, Assessing the accuracy of prediction algorithms for classification: An overview, *Bioinformatics* **16**:412–424, 2000.
31. Cios K, Kurgan L, CLIP4: Hybrid inductive machine learning algorithm that generates inequality rules, *Inf Sci* **163**(1–3):37–83, 2004.
32. Kurgan L, Cios K, Scott D, Highly scalable and robust rule learner: Performance evaluation and comparison, *IEEE Trans Syst Man Cybern B Cybern* **36**(1):32–53, 2006.
33. Cios K, Pedrycz W, Swiniarski R, Kurgan L, *Data Mining: A Knowledge Discovery Approach*, Springer, New York, 2007.
34. Khan S, Situ G, Decker K, Schmidt CJ, GoFigure: Automated Gene Ontology annotation, *Bioinformatics* **19**:2484–2485, 2003.
35. Andrade M, Brown N, Leroy C, Hoersch S, de Daruvar A, Reich C, Franchini A, Tamames J, Valencia A, Ouzounis C, Sander C, Automated genome sequence analysis and annotation, *Bioinformatics* **15**:391–412, 1999.
36. Lu Z, Hunter L, GO molecular function terms are predictive of subcellular localization, *Pac Symp Biocomput*, pp. 151–161, 2005.

37. Bradley A, The use of the area under the ROC curve in the evaluation of machine learning algorithms, *Pattern Recognit* **30**(7):1145–1159, 1997.
38. DiCiccio T, Efron B, Bootstrap confidence intervals (with discussion), *Stat Sci* **11**:189–228, 1996.
39. Fawcett T, Using rule sets to maximize ROC performance, *Conference on Data Mining (ICDM-2001)*, pp. 131–138, 2001.
40. Davison A, Hinkley S, *Bootstrap Methods and Their Application*, Cambridge University Press, Cambridge, UK, 1997.

Cao Nguyen received his Ph.D. degree in Computer Science (with an option in Computational Biology) from the University of Colorado at Denver and Health Sciences Center, Aurora, CO, USA; and is currently a Post-Doctoral Research Associate in the Department of Computer Science at Virginia Commonwealth University, Richmond, VA, USA. He conducts research with Dr. Cios in the area of mathematical modeling (hidden Markov models, clustering, and fuzzy cognitive maps) for prediction of protein interfaces and protein functions.

Michael Mannino is an Associate Professor in the Business School of the University of Colorado at Denver and Health Sciences Center, Aurora, CO, USA. Previously, he was on the faculty of the University of Florida, University of Texas at Austin, and University of Washington. He has been active in research on database management, knowledge representation, and organizational impacts of technology. He has published articles in major journals of the IEEE (*IEEE Transactions on Knowledge and Data Engineering* and *IEEE Transactions on Software Engineering*), ACM (*Communications of the ACM* and *ACM Computing Surveys*), and INFORMS (*INFORMS Journal on Computing and Information Systems Research*). His research includes several popular survey and tutorial articles as well as many papers describing original research. He is the author of the textbook *Database Design, Application Development, and Administration*, published by McGraw-Hill/Irwin.

Katheleen Gardiner received her Ph.D. degree from the University of Colorado, and is currently a Professor in the Department of Pediatrics at the University of Colorado Denver, Anschutz Medical Campus, Aurora, CO, USA. Dr. Gardiner is an internationally recognized researcher in the field of Down syndrome. Her specific research interests are in the identification of genes encoded by human chromosome 21 that contribute to learning and memory deficits in Down syndrome, and the use of data from mouse and other model organisms to predict associated pathway perturbations. Dr. Gardiner has authored over 100 journal articles, meeting reports, and book chapters. Her research is currently funded by the National Institutes of Health, the Fondation Jérôme Lejeune, and the Anna and John J. Sie Foundation.

Krzysztof J. Cios received his M.S. and Ph.D. degrees from the AGH University of Science and Technology, Krakow, Poland; his MBA degree from the University of Toledo, Toledo, OH, USA; and his D.Sc. degree from the Polish Academy of Sciences Warsaw, Poland. He is currently a Professor and Chair of the Computer Science Department at the Virginia Commonwealth University, Richmond, VA, USA. Dr. Cios' areas of research are biomedical informatics and data mining. NASA, NSF, American Heart Association, Ohio Aerospace Institute, NATO, US Air Force, and NIH have funded his research. He has published 3 books and about 150 journal and conference articles. He serves on the editorial boards of *Neurocomputing*, *Journal of Integrative Neuroscience*, *IEEE Engineering in Medicine and Biology Magazine*, and *International Journal of Computational Intelligence*. Dr. Cios has been the recipient of the Norbert Wiener Outstanding Paper Award, the *Neurocomputing* Best Paper Award, the University of Toledo Outstanding Faculty Research Award, and the Fulbright Senior Scholar Award. He is a Foreign Member of the Polish Academy of Arts and Sciences.