BIOINFORMATICS ORIGINAL PAPER Vol. 23 no. 23 2007, pages 3217–3224
doi:10.1093/bioinformatics/btm511

Systems biology

Graph sharpening plus graph integration: a synergy that improves protein functional classification

Hyunjung Shin[1,*,†], Andreas Martin Lisewski[2,†] and Olivier Lichtarge[2]

[1]Department of Industrial & Information Systems Engineering, Ajou University, San 5, Wonchun-dong, Yeoungtong-gu, 443−749, Suwon, Korea and [2]Department of Molecular & Human Genetics, Baylor College of Medicine, One Baylor Plaza, Houston, Texas 77030, USA

Received on May 4, 2006; revised on August 24, 2007; accepted on October 8, 2007

Advance Access publication October 31, 2007

Associate Editor: Jonathan Wren

ABSTRACT

Motivation: Predicting protein function is a central problem in bioinformatics, and many approaches use partially or fully automated methods based on various combination of sequence, structure and other information on proteins or genes. Such information establishes relationships between proteins that can be modelled most naturally as edges in graphs. A priori, however, it is often unclear which edges from which graph may contribute most to accurate predictions. For that reason, one established strategy is to integrate all available sources, or graphs as in *graph integration*, in the hope that the positive signals will add to each other. However, in the problem of functional prediction, noise, i.e. the presence of inaccurate or false edges, can still be large enough that integration alone has little effect on prediction accuracy. In order to reduce noise levels and to improve integration efficiency, we present here a recent method in graph-based learning, *graph sharpening*, which provides a theoretically firm yet intuitive and practical approach for disconnecting undesirable edges from protein similarity graphs. This approach has several attractive features: it is quick, scalable in the number of proteins, robust with respect to errors and tolerant of very diverse types of protein similarity measures.

Results: We tested the classification accuracy in a test set of 599 proteins with remote sequence homology spread over 20 Gene Ontology (GO) functional classes. When compared to integration alone, graph sharpening plus integration of four vastly different molecular similarity measures improved the overall classification by nearly 30% [0.17 average increase in the area under the ROC curve (AUC)]. Moreover, and partially through the increased sparsity of the graphs induced by sharpening, this gain in accuracy came at negligible computational cost: sharpening and integration took on average 4.66 (±4.44) CPU seconds.

Availability: Software and Supplementary data will be available on http://mammoth.bcm.tmc.edu/

Contact: shin@ajou.ac.kr or lisewski@bcm.edu

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

*To whom correspondence should be addressed.

†The authors wish it to be known that, in their option, the first two authors should be regarded as joint First Authors.

1 INTRODUCTION

The prediction of biological function is a central challenge in modern biology since few genomic or proteomic data have detailed annotations (Friedberg, 2006). A new class of computer aided solutions to this problem is currently emerging that aims to integrate diverse sources of relevant information, rather than relying only on single methods that have specific limitations (Friedberg *et al*., 2006; Laskowski *et al*., 2005; Pal and Eisenberg, 2005). Function information may come in many forms, ranging from genomic and molecular to cellular and tissue contexts. The unprecedented availability of data from primary to tertiary protein structure in particular motivated many methods that use computational comparisons of specific molecular attributes to define functional relationships (Watson *et al*., 2005). From this, a clearer signal for function prediction may emerge by considering multiple relevant relationships.

A natural model of relationships between proteins is a network (or graph), where nodes depict genes or proteins and edges represent their possible interactions or correlations (Kanehisa *et al*., 2004; Uetz *et al*., 2000; von Mering *et al*., 2002) (see Fig. 1). Among these there are networks of weighted edges that evaluate molecular similarity between protein pairs extracted from sequence or structure comparisons and classify or predict protein function (Adai *et al*., 2004; Friedberg and Godzik, 2005; Hou *et al*., 2005; Yona *et al*., 1999). Protein function prediction can benefit by combining a diverse set of similarity measures from protein sequences, physical interactions, gene regulatory networks and metabolic pathways, etc. And there have been such approaches to combining multiple networks, for instance, majority vote (Hishigaki *et al*., 2001; Schwikowski *et al*., 2000), Bayesian networks (Deng *et al*., 2003), discriminative learning methods (Lanckriet *et al*., 2004a; Vert and Kanehisa, 2003), probabilistic integration by log-likelihood scores (Lee *et al*., 2004) and semidefinite programming (SDP) based SVM method (Lanckriet *et al*., 2004b), etc. Recently, it was shown that graphs based on semi-supervised learning (Chapelle *et al*., 2006; Zhou *et al*., 2004) yield better classification performance for sequence similarity networks than conventional local sequence comparison (Noble *et al*., 2005). This suggested that graph-based semi-supervised learning can capture global network information

that is functionally relevant. Later, and in a more general approach, multiple protein networks were then combined into a single computationally efficient semi-supervised learning problem, and this further not merely improved functional classification over individual networks but also enormously reduced computation time when compared with other state-of-the-art methods for multiple networks (Tsuda *et al*., 2005).

Here, we further extend these ideas to improve the annotation of protein function based on graphs. Starting from a diverse set of protein similarity measures, some that are standard and based on sequence and others that are more novel and based on structure, we apply two most recent graph-based algorithms in machine-learning field. Compared to previous work on *graph integration*, the novelty of this approach lies in taking into explicit account the directionality of edges in each graph through a method that we call *graph sharpening*, since it effectively eliminates those edges that tend to corrupt functional information represented in the underlying network. With a test set of 15 out of 20 functional GO terms among 599 non-redundant protein structures from the PDBselect25 list (Hobohm and Sander, 1994), we show that sharpening and integration raise the overall classification performance by nearly 30%. In absolute values, an average increase of the AUC—the area under the Receiver Operating Characteristic (ROC) curve by 0.17 up to a level of 0.75.

This article is organized as follows. First, we introduce graph-based semi-supervised learning in Section 2. In Section 3, the basic idea and some elements of the mathematical framework behind graph sharpening are explained (Shin *et al*., 2006). In Section 4, we present a specific graph-integration approach (Tsuda *et al*., 2005). Then, we demonstrate how these methods can be successfully applied to protein function prediction (Section 5). We conclude with some future work remarks.

2 GRAPH-BASED LEARNING

In the graph-based semi-supervised learning algorithm (Zhou *et al*., 2004), a data point $\boldsymbol{x}_i$ $(i = 1,\ldots,n)$ is represented as a node i in a graph, and the relationship between data points is represented by an edge where the connection strength from each node j to each other node i is encoded in element w_{ij} of a weight matrix W. A weight w_{ij} can take a binary value (0 or 1) in the simplest case. Often, a Gaussian function of Euclidean distance between points, with length scale σ, is used to specify connection strength:

$$w_{ij} = \begin{cases} \exp\left(-\frac{(\boldsymbol{x}_i-\boldsymbol{x}_j)^\top(\boldsymbol{x}_i-\boldsymbol{x}_j)}{\sigma^2}\right) & \text{if } i \sim j, \\ 0 & \text{otherwise.} \end{cases}$$

The $i \sim j$ stands for node i and j having an edge between them that can be established either by k nearest neighbors or by Euclidean distance within a certain radius r, $||\boldsymbol{x}_i - \boldsymbol{x}_j||^2 < r$.[1] The labelled nodes have labels $\boldsymbol{y}_l \in \{-1,1\}$, while the unlabelled nodes have zeros $\boldsymbol{y}_u = \boldsymbol{0}$. Our algorithm

[1]We represent scalars as lower case, vectors as boldface lower case and martrices are uppercase. $\boldsymbol{0}$ (or $\boldsymbol{1}$) is a vector or matrix of variable-dependent size containing of all zeros (or ones).

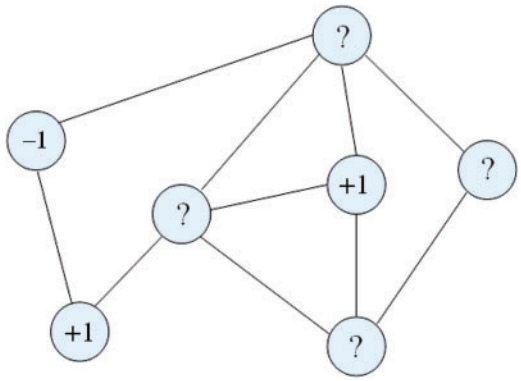

Fig. 1. A graph model of relationships between proteins. Nodes depict genes or proteins and edges represent their possible interactions or correlations, e.g. molecular similarities extracted from sequence or structure comparisons. An annotated protein is labelled either by '+1' or '−1', indicating either it belongs to a particular functional class or not. Graph-based function prediction seeks to classify the unannotated (unlabelled) proteins marked as '?'.

will output an n-dimensional real-valued vector $\boldsymbol{f} = [\boldsymbol{f}_l^\top \boldsymbol{f}_u^\top]^\top = (f_1,\ldots,f_l,f_{l+1},\ldots,f_{n=l+u})^\top$, which can be thresholded to make label predictions on $f_{l=1},\ldots,f_n$ after learning. It is assumed that (a) f_i should be close to the given label y_i in labelled nodes, and (b) overall, f_i should not be too different from the f_j of adjacent nodes $(i \sim j)$. One can obtain $\boldsymbol{f}$ by minimizing the following quadratic functional (Belkin *et al*., 2004, Chapelle *et al*., 2003; Zhou *et al*., 2004):

$$\min_{\boldsymbol{f}} (\boldsymbol{f}-\boldsymbol{y})^\top(\boldsymbol{f}-\boldsymbol{y}) + \mu \boldsymbol{f}^T L \boldsymbol{f} \tag{1}$$

where $\boldsymbol{y} = (y_1,\ldots,y_l,0,\ldots,0)^\top$, and the matrix L, called the *graph Laplacian matrix* (Chung, 1997), is defined as $L = D - W$ where $D = \mathrm{diag}(d_i)$, $d_i = \sum_j w_{ij}$. The first term corresponds to the *loss function* in terms of condition (a), and the second term represents the *smoothness* of the predicted outputs in terms of condition (b). The parameter μ trades off loss versus smoothness. The solution of this problem is obtained as

$$\boldsymbol{f} = (I + \mu L)^{-1}\boldsymbol{y} \tag{2}$$

where I is the identity matrix.

3 GRAPH SHARPENING

Now we present a method that is directly applicable to the weight matrix W, and which is based on the following intuition: in an undirected graph as in Figure 1, all connections are reciprocated and so the matrix of edge weights W is symmetric as shown in Figure 2a. However, when W describes relationships between labelled and unlabelled points, it is not necessarily desirable to regard all such relationships as symmetric. That is, we may differentiate the contribution of all edges to information flow by not weighing them equally. First, some edges may convey more useful information in one direction (e.g. from labelled to unlabelled) than in the reverse direction. Propagating

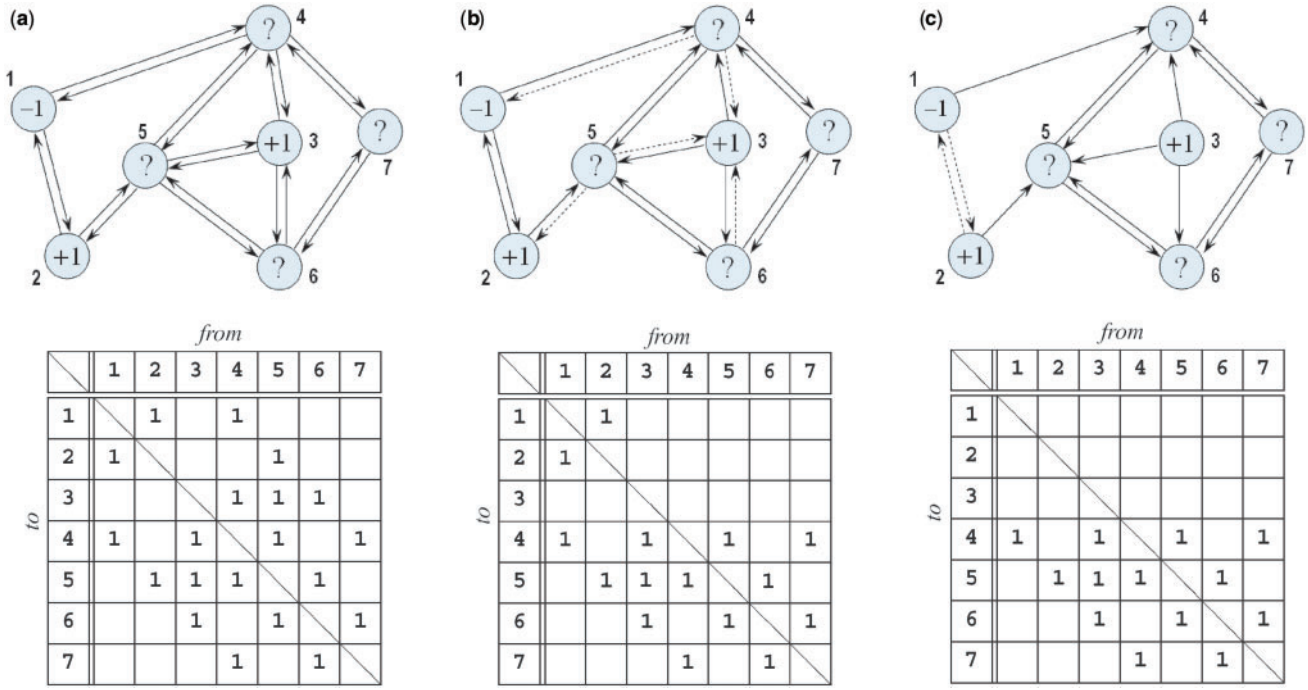

Fig. 2. Graph sharpening: (**a**) an original undirected (bi-directed) graph and its symmetric weight matrix W. Note that in the interpretation of W, the row index denotes the destination and the column index the source—so, e.g. w_{ij} should be read as 'the weight of the edge *from j to i* ($j \rightarrow i$)'. (**b**) Edges from unlabelled to labelled points (denoted as dotted arrows) are disconnected. (**c**) Edges between oppositely labelled points are further removed. Sharpening leaves the edges between unlabelled points intact.

information in the reverse direction, from unlabelled to labelled, may be undesirable since it allows points about which information is uncertain to corrupt the source of information in the system. Since we are already using the language of 'points' and 'edges', we will say that this causes the point and its outgoing edges to be 'blunted', reducing their effectiveness. There are many problem settings (e.g. protein function prediction and other applications in the field of bioinformatics) in which (a) there is a high degree of certainty about the input space representation of each labelled point and its label, and (b) the number of labelled points is low. In this situation, it is indicated to avoid blunting, thus to preserve the effectiveness of information sources. Second, edges directly connecting oppositely labelled points may propagate unhelpful information in either direction. The smoothness condition in Equation (1) plays the role of forcing both predicted scores to be similar to each other and again both important points, the very sources of information are blunted. Further, those edges unnecessarily incur a conflict of the opposite flows in the area of a high certainty in the system. Third, propagation of information between unlabelled points is different—-while some edges of the graph may be more helpful than others in solving the overall problem, a priori we do not know which these might be. Allowing the unlabelled points to harmonize with their neighbours (thus implementing smoothness condition) common to most such learning approaches) is a desirable process.

Figure 2 illustrates the concept of graph sharpening. Figure 2a shows an original graph of seven nodes (points) and its (so far) symmetric weight matrix W. For simplicity, we set the value of '1' as a weight on every edge. Remember that, in the interpretation of W, the row index denotes the destination and the column index the source, and w_{ij} reads 'the weight of the edge *from j to i* ($j \rightarrow i$)'. In Figure 2b, the edges from unlabelled to labelled points w_{ij} (denoted as dotted arrows) are disconnected where i belongs to the set of labelled points $l := \{1, 2, 3\}$ and j to the set of unlabelled points $u := \{4, 5, 6, 7\}$. Figure 2c presents the case of connection between oppositely labelled points, w_{12} and w_{21}. Therefore, both edges are further removed from Figure 2b. Finally, we obtain a *sharpened* graph in Figure 3. Note that the weight matrix becomes asymmetric, and the edges between unlabelled points remain intact. A detailed mathematical foundation of *sharpening* is given in (Shin *et al.*, 2006). Let us give a schematic flow of the proof. We pose the general question: what if W is not considered fixed? Is it possible to change some or all of the w_{ij} such that our algorithm performs better? We begin by re-formulating the objective function Equation (1) in terms of W, which is based on the formulation of Belkin *et al.* (2004). Blockwise consideration of the weight matrix,

$$W = \begin{bmatrix} W_{ll} & W_{lu} \\ W_{ul} & W_{uu} \end{bmatrix}$$

allows us to state a condition which solutions W must satisfy if the objective function is to be optimized—there are many such solutions. Exploring every class of solutions is currently regarded as intractable and is left as an open problem. However, we can specify one simple *ad hoc* solution, concordant with the intuition already stated. By setting W_{ll} to

a non-negative diagonal matrix (including the null matrix) and W_{lu} to $\mathbf{0}$ such as

$$W_s = \begin{bmatrix} \text{diagonal matrix} & 0 \\ W_{ul} & W_{uu} \end{bmatrix} \quad (3)$$

(see the final W in Fig. 2c), we obtain the output prediction for unlabelled data points

$$\boldsymbol{f}_u = \mu(I + \mu(D_{uu} - W_{uu}))^{-1} W_{ul}\, \boldsymbol{y}_l. \quad (4)$$

In spreading activation network terms, Equation (4) is equivalent to information being propagated from labelled to unlabelled data *once* ($W_{ul}\boldsymbol{y}_l$) to set the initial condition for subsequent spreading activation among $u \longleftrightarrow u$, analogous to Equation (2) but now *excluding* the labelled points. This also has intuitive appeal. First, for labelled points, it assures $\boldsymbol{f}_l = \boldsymbol{y}_l$—*there is no loss of information on labelled data points*. By disconnecting unnecessary and unhelpful edges, we allow the labelled points and their outgoing edges to stay 'sharp' in their influence on the rest of the graph. Second, for unlabelled points, it preserves an important principle of SSL, namely *exploitation of the manifold structure inferred from unlabelled data points*, by keeping the edges, $u \longleftrightarrow u$ and $l \rightarrow u$, of W.

4 GRAPH INTEGRATION

A protein graph model represents relationships between proteins. Nodes depict proteins, and edges represent their possible interactions or similarities; for instance, one can extract an edge set (or similarity measurements) from amino acid sequences, or from structures, or from interactions, etc. Since there are many ways to represent a set of edges for a given set of proteins there exist many graphs. And each graph is regarded as partly independent from and partly complementary to others. Frequently, it can be difficult to decide which graph will perform best for function prediction for unlabelled proteins. Individual graphs (from single data sources) are often not sufficient for reliable prediction. One way to enhance reliability may be to integrate the given multiple graphs. Integrating multiple graphs stands for finding an optimum value of the linear combination coefficient for the individual graphs. In the semi-supervised learning framework, this translates to finding the combination coefficients $\boldsymbol{\alpha}$ for the individual Laplacians, as shown in Figure 4. Recently, Tsuda *et al.* (2005) proposed an integration method for multiple graphs. One can see how the illustration in Figure 4 is related to their formulation,

$$\min_{\alpha} \boldsymbol{y}^{\top}(I + \sum_{k=1}^{K} \alpha_k L_k)^{-1}\boldsymbol{y} \quad \sum_{k} \alpha_k \le \mu \quad (5)$$

and its output prediction

$$\boldsymbol{f} = (I + \sum_{k=1}^{K} \alpha_k L_k)^{-1}\boldsymbol{y} \quad (6)$$

where K is the number of graphs and an L_k is the corresponding *graph Laplacian* to graph G_k.

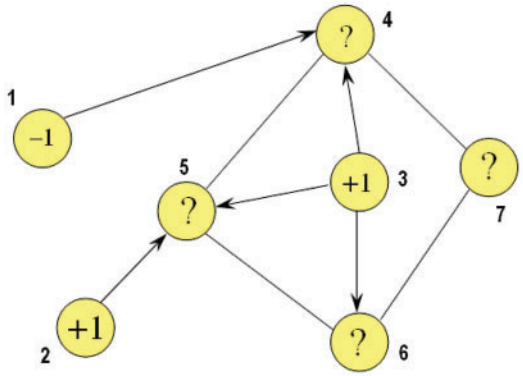

Fig. 3. The sharpened graph: in contrast to the original in Figure 1, the *sharpened* graph is no longer fully reciprocated and so the matrix of edge weights W becomes asymmetric.

5 FUNCTION PREDICTION EXPERIMENTS

5.1 Data

In automated protein functional class classification, accurate detection of functional relationships beyond close sequence homology is desirable. Thus, we have restricted our selection to protein chains from the PDBselect25 list (version October 2004), where any pair shares no more than 25% sequence identity. Functional information was assigned in terms of the Gene Ontology (GO) in the category 'Molecular function' and mapped through the gene ontology annotation database (GOA PDB 24.0). We chose the 20 highly populated GO categories as in Hou *et al.* (2005). Of all PDBselect25 chains, 599 proteins share at least one of these functional GO terms (Table 1). Therefore, we took them as our experimental protein set.

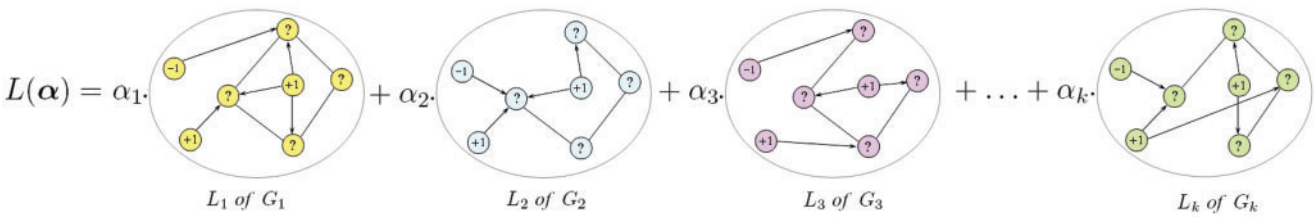

Fig. 4. Graph integration: every single graph G_k can solely be used for label prediction. However, since different graphs contain partly independent and partly complementary pieces of information, integrating them into one may increase the reliability of predictions. In semi-supervised learning framework, graph integration stands for finding an optimum value of the linear combination coefficient $\boldsymbol{\alpha}$ for individual graph Laplacians L_k.

To generate weighted edges between nodes (or proteins), we used four different computational measures of molecular similarity. Each measure assigned to every protein pair (i,j) a positive weight $(0 \leq w_{ij} \leq 1)$ meaning the degree of molecular similarity between chain i and j. The four different similarity measures are Basic Local Alignment and Search Tool (BLAST, Altschul *et al.*, 1990), the standard approach to alignment of primary sequence that resulted in a degree of sequence identity; length-corrected Contact Metric (CM, Lisewski and Lichtarge, 2006), a metric-based similarity score by considering vector representations of contacts from the entire tertiary structure; Fast Alignment and Search Tool (FAST, Zhu and Weng, 2005), an accurate and computationally efficient 3D geometrical alignment algorithm calculating positive similarity scores; Evolutionary Trace Annotation (ETA, Kristensen *et al.*, 2006), a method that measures similarity based on local similarity of protein substructure, specifically 3D templates that are small structural motifs of evolutionarily important residues identified by the evolutionary trace method (Lichtarge *et al.*, 1996). Our choice represented all currently and commonly used approaches to protein similarity measurement (Watson *et al.*, 2005): sequence alignment (BLAST), global 3D alignment (FAST), local 3D alignment of small motifs (ETA), and alignment-independent vector-based approaches (CM). An edge-weight from BLAST/CM/FAST is a continuous value while that of ETA is a binary value from a support vector classifier, i.e. the edge-weight between two proteins is set to 1 ($w_{ij}=1$) if they were predicted functionally related; no edge ($w_{ij}=0$) otherwise.

Table 1. Twenty functional GO terms from the category 'Molecular function'

GO term	Molecular function	Number of proteins
0003677	DNA binding	137
0005515	Protein binding	49
0016491	Oxidoreductase activity	39
0005524	ATP binding	95
0003723	RNA binding	50
0006118	Electron transport	87
0003676	Nucleic acid binding	63
0003824	Catalytic activity	57
0005198	Structural molecule activity	22
0005509	Calcium ion binding	32
0000287	Magnesium ion binding	3*
0008270	Zinc ion binding	56
0005489	Electron transporter activity	37
0004872	Receptor activity	7*
0016798	Hydrolase activity	0*
0004519	Endonuclease activity	9*
0004871	Signal transducer activity	18
0004672	Protein kinase activity	29
0004518	Nuclease activity	5*
0004867	Serine-type endopeptidase inhibitor activity	15

GO terms with less than 10 annotated proteins were excluded from our classification experiment (asterisk mark in the last column).

5.2 Results

Since a protein can belong to several GO categories, we posed a binary-class classification problem for each GO term in Table 1. The GO categories having only a few labelled proteins (less than 10) were excluded; therefore, our experiment became 15 binary-class classification problems, determining membership or non-membership of unannotated (unlabelled) proteins for the respective GO terms. For each GO term, we calculated the 5-fold cross-validation (5 CV) AUC (area under the curve) of ROC (receiver operating characteristic) as a performance measurement. The ROC curve plots true positive rate (sensitivity) as a function of false positive rate (1-specificity)

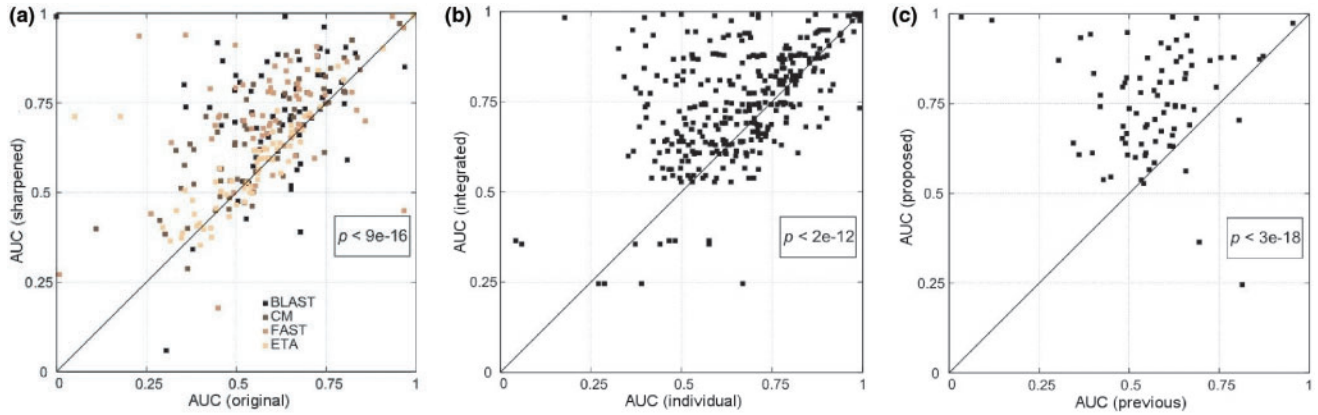

Fig. 5. Positive effect of sharpening in pairwise AUC comparison for two competing methods in three different cases. More dots in upper diagonal half indicate that the method in vertical axis outperforms the other assigned in the horizontal axis; *P*-values (from a sign test) represent the statistical deviation from a random distribution of dots in the upper and lower half. (**a**) Sharpened versus. original: for 221 out of 300 (= 5 repetitions × 15 GO terms × 4 graphs) pairs in total, *sharpening* gave a higher AUC; (**b**) integrated (sharpened) versus individuals (sharpened): for 212 out of 300 (= 5 × 15 × 4) experiments, *integration* gave a higher AUC and (**c**) integration with sharpening (2007) versus without sharpening (2005): for 70 out of 75 (= 5 × 15) experiments, *integration with sharpening* had a higher ROC score. More detailed results can be found in Appendix A from our Supplementary Material.

for all possible thresholds (as opposed to choosing a single threshold (Gribskov and Robinson, 1996), see Figure 7. Hence, it has become a standard qualifier for classification algorithms. An AUC of 0.5 corresponds to random guessing, while an AUC of 1.0 implies the algorithm successfully outputs a higher predicted value for any positive example than that for any negatives example (perfect classification).

We compared the proposed method *integration on sharpened graphs*, with the four individual graphs obtained from BLAST, FAST, CM and ETA, respectively, and also with the previous method of Tsuda *et al*. (2005)—*integration on original graphs*. In every comparison, we examined the performance change in terms of *original versus sharpened* and *individual versus integrated*. This setting enabled us to see the effect of the proposed method from two separate viewpoints, *sharpening* and *integration*. The value of smoothing parameter μ—from Equation (2) of *original*, Equation (4) of *sharpening* and Equation (6) of *integration*—was obtained by 5CV searching over $\mu \in \{0.1, 1, 5, 10, 50, 100\}$. For each of the available settings per GO category, we compared the results at their best parameters. A table with the best parameter is published at the Internet address (http://mammoth.bcm.tmc.edu/biocomp 2007/). Remarkably, *sharpening* does not include any additional parameters, nor *integration*—the linear combination coefficient $\boldsymbol{\alpha}$ in Equation (6) is automatically set as a solution of the optimization.

5.2.1 Graph sharpening on individual graphs Figure 5a shows the distribution of 300 (= 15 GO terms × 5 CVs × 4 graphs) AUC pairs from the original (unsharpened) and sharpened graphs. Most dots are scattered above the diagonal, thus indicating better performance of the sharpened graphs against originals. A *sign test* statistically rejected the null hypothesis, 'the distribution would be evenly scattered below and above the diagonal', with significant confidence ($p < 9 \times 10^{-16}$). Table 2 confirms this result: on average, sharpening increases the original AUC by 0.03 (ETA) up to 0.12 (FAST).

5.2.2 Integrated graph versus individual graphs The next test was whether individual prediction in sharpened graphs could be further improved through their *integration* as stated in Section 4. Figure 5b shows that this was the case: 222 of the 300 ROC AUC pairs are above the diagonal ($p < 9 \times 10^{-16}$). In Figure 6, the average over 5CV AUCs are assigned to their function classes. The figure assures that *integration* on sharpened graphs achieves the highest AUCs when compared with the *individuals* (sharpened)—except for GO 0003677 (DNA binding) and GO 0008270 (Zinc ion binding). Integration effect on sharpened individual graphs show more significance and less volatility than the effect of integration on the original (unsharpened) networks (Table 2): 0.09 (±0.04) versus 0.02 (±0.08) in average AUC increase by integration.

5.2.3 Proposed method versus previous method In the third test, we verified how integration on sharpened graphs upgrades the performance of the previous method (Tsuda *et al*., 2005), *integration alone*, so to speak. The results show that sharpening significantly contributes to AUC improvement, see Figure 5c; 70 out of 75 ROC pairs are in the upper diagonal ($p < 3 \times 10^{-18}$). Integration alone can be insignificant and

Table 2. AUC comparison for all the combination of *it original versus sharpened* and *individual versus integrated*. Other statistics such as sensitivity and specificity can be found in Appendix B in our Supplementary Material

Graphs		GO terms													AUC increase by sharpening
		0003677	0005515	0016491	0005524	0003723	0006118	0003676	0003824	0005198	0005509	0008270	0005489	0004871	0004672	0004867	
BLAST	Original	0.71	0.70	0.41	0.56	0.60	0.68	0.70	0.60	0.65	0.59	0.74	0.59	0.68	0.60	0.83	**0.09**
	Sharpened	0.71	0.76	0.72	0.56	0.59	0.74	0.72	0.80	0.71	0.75	0.76	0.84	0.74	0.55	0.92	±0.10
CM	Original	0.64	0.46	0.75	0.49	0.41	0.61	0.56	0.76	0.67	0.50	0.59	0.36	0.39	0.51	0.75	**0.10**
	Sharpened	0.64	0.73	0.76	0.52	0.56	0.63	0.68	0.75	0.74	0.56	0.65	0.59	0.56	0.66	0.93	±0.08
FAST	Original	0.65	0.53	0.64	0.55	0.43	0.67	0.55	0.70	0.68	0.71	0.60	0.62	0.37	0.61	0.76	**0.12**
	Sharpened	0.65	0.78	0.73	0.63	0.62	0.72	0.69	0.81	0.64	0.82	0.67	0.82	0.84	0.73	0.88	±0.13
ETA	Original	0.51	0.43	0.53	0.57	0.46	0.57	0.45	0.45	0.53	0.56	0.50	0.59	0.75	0.64	0.53	**0.03**
	Sharpened	0.50	0.49	0.55	0.58	0.46	0.60	0.48	0.45	0.45	0.57	0.52	0.59	0.74	0.67	0.56	±0.07
Integrated	Original	0.68	0.49	0.75	0.51	0.41	0.64	0.56	0.76	0.73	0.63	0.64	0.49	0.39	0.58	0.79	**0.16**
	Sharpened	0.66	0.78	0.80	0.59	0.61	0.73	0.71	0.83	0.77	0.83	0.71	0.81	0.84	0.73	0.96	±0.12
		AUC increase by integration															
Original	avg.	**0.05**	**−0.04**	**0.17**	**−0.03**	**−0.06**	**0.01**	**0.00**	**0.13**	**0.10**	**0.04**	**0.03**	**−0.05**	**−0.16**	**−0.01**	**0.07**	**0.02**
	std.	±0.09	±0.12	±0.14	±0.04	±0.09	±0.05	±0.10	±0.13	±0.07	±0.09	±0.10	±0.12	±0.19	±0.06	±0.13	±0.08
Sharpened	avg.	**0.03**	**0.09**	**0.11**	**0.01**	**0.05**	**0.06**	**0.06**	**0.13**	**0.11**	**0.15**	**0.05**	**0.10**	**0.12**	**0.08**	**0.13**	**0.09**
	std.	±0.09	±0.14	±0.09	±0.05	±0.07	±0.07	±0.11	±0.17	±0.08	±0.13	±0.10	±0.14	±0.12	±0.08	±0.18	±0.04
		AUC increase by sharpening and integration															
	avg.	**0.03**	**0.25**	**0.22**	**0.05**	**0.14**	**0.10**	**0.14**	**0.20**	**0.14**	**0.24**	**0.10**	**0.27**	**0.29**	**0.13**	**0.24**	**0.17**
	std.	±0.08	±0.10	±0.12	±0.03	±0.07	±0.05	±0.09	±0.11	±0.06	±0.08	±0.09	±0.10	±0.17	±0.05	±0.11	±0.08

avg. = average and Std. = standard deviation.

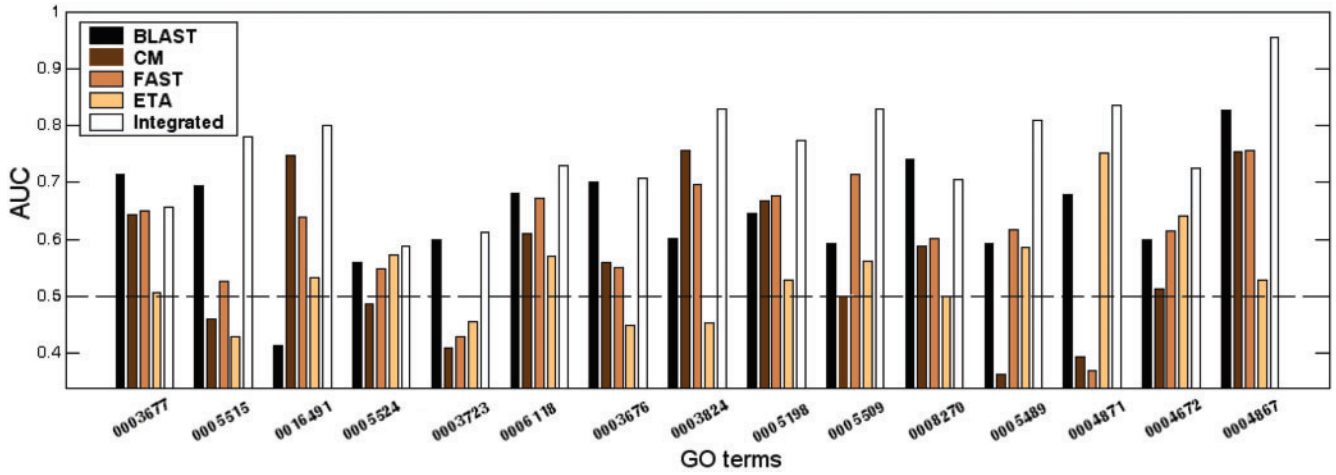

Fig. 6. AUC comparison (avg. of five cross-validation) between individual and the integrated graphs: bars within a group correspond to BLAST, CM, FAST, ETA and the integrated graph in due order. For 13 out of 15 categories, *integration with sharpening* significantly surpasses individual performance.

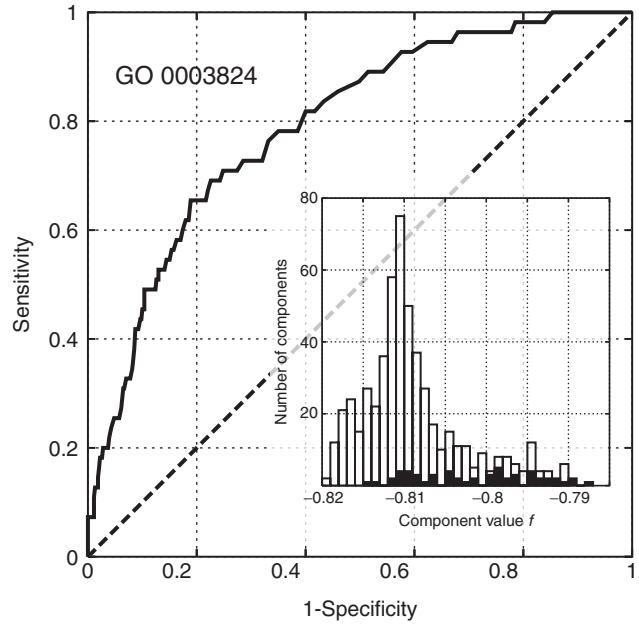

Fig. 7. ROC curve on GO 0003824 ('Catalytic activity')—the curve shows that graph sharpening and integration can reliably discriminate enzymes (catalytic proteins) from non-enzymes. Insert: the distribution of predicted values (scores), ***f***, for the unlabelled nodes (unannotated proteins)—empty bars stand for the distribution of non-enzymes, whereas solid bars stand for that of enzymes. An enrichment of enzymes toward larger values is evident.

even worse (see, value 0.02 ± 0.08 in Table 2), particularly if given graphs are noisy, which can often be the case in protein similarity networks. But with sharpening, integration becomes more effective, which is reflected by an AUC increase of 0.17 ± 0.08, see Table 2. Hence, given the performance of single unsharpened (original) graphs, one can raise the AUC as much as 0.17 ± 0.08 by applying integration with sharpening.

5.2.4 Computation time Sharpening does not require computation. The computation time of an original graph, the time for solving the sparse linear system in Equation (2), was nearly trivial (less than 0.001 CPU second with MATLAB in a standard 1500 MHz PC with 1 GB of memory). It became faster for a *sharpened* graph since the Laplacian matrix gets sparser. *Integration* in Equation (6) took avg. 24.27 ($\pm$ 10.86) CPU seconds for original, while it was avg. 4.66 ($\pm$4.44) for sharpened weights, and graph sparsity through sharpening benefitted the computation time for integration.

5.2.5 Enrichment Figure 7 shows a typical ROC curve of the proposed method for GO 0003824 'Catalytic activity'. The curve shows that sharpening and integration can reliably discriminate enzymes (catalytic proteins) from non-enzymes among proteins with $<25\%$ sequence similarity. The inner figure shows the distribution of predicted values (scores), ***f***, for the unannotated proteins. An enrichment of enzymes toward larger values is evident.

6 CONCLUSION AND DISCUSSION

We applied to the problem of protein function prediction two recent developments in machine-learning methods based on graph, sharpening (Shin *et al.*, 2006) and integration (Tsuda *et al.*, 2005). Graph sharpening is a formal (details were omitted for brevity) yet intuitive approach for disconnecting undesirable edges in a graph. The result yields sparser graphs that contain less noise and, in turn, reduce computational expense and improve prediction without need for additional parameters. Using an already established optimization framework, graph integration can then be applied with greater efficiency and effectiveness in order to pool information from multiple sources. Both strategies, graph sharpening and graph integration, lend themselves well to the protein function prediction problem. Graph sharpening offers a general framework to address the noise that is pervasive among functionally relevant measures of protein similarity, while graph integration allows

overall predictions to take into account a wide variety of complementary types of information on protein similarity graphs, each one representing a different aspect or type of functional information. While either strategy can be applied alone, with sharpening having a greater effect than integration, the best result was reached when sharpening and integration are used together and thus yield a synergestic effect on function prediction performance at lesser computational cost. This improvement is noteworthy for its size (0.17, or nearly 30% average increase in the area under the ROC curve) and in view of the diversity of similarity scores that were integrated: BLAST is used over sequences, CM and FAST are applied over whole structures and ETA is applied to a local structural motif (and has only been optimized for enzymes, rather than for the GO annotations used here).

This work motivates possible future studies. First, the method is general and its full application for function prediction will still require a continued refinement of individual methods, as well as broadening the number of similarity measures whose graphs are sharpened and then integrated together. Towards this goal, we plan an internet accessible software tool for sharpening and integration to allow large scale function prediction using these methods. Second, from a theoretical perspective, statistical formalization or framework on how sharpening and integration work together has not been conclusively established. This should be studied further.

ACKNOWLEDGEMENTS

H.S. was supported by the Korea Research Foundation Grant funded by the Korean Government (MOEHRD) and by the grant for Post Brain Korea 21. O.L. would like to gratefully acknowledge support from NSF DBI-0547695, NIH GM066099, and the March of Dimes MOD FY06-371.

Conflict of Interest: none declared.

REFERENCES

Adai,A.T. *et al*. (2004) Connecting the protein structure universe by using sparse recurring fragments. *J. Mol. Biol*., **340**, 179–190.

Altschul,S.F. *et al*. (1990) Basic local alignment search tool. *J. Mol. Biol*., **215**, 403–410.

Belkin,M. *et al*. (2004) Regularization and Semi-supervised Learning on Large Graphs. *In Proceedings of the 17th Annual Conference on Learning Theory (COLT) 3120. Lecture Notes in Computer Science*. Springer, pp. 624–638.

Chapelle,O. *et al*. (2003) Cluster kernels for semi-supervised learning. In Becker,S. *et al*. (eds.), *Advances in Neural Information Processing Systems (NIPS) 15*. MIT Press, Cambridge, MA, pp. 585–592.

Chapelle,O. *et al*. (2006) *Semi-Supervised Learning*. MIT Press, Cambridge, MA.

Chung,F.R.K. (1997) Spectral Graph Theory. *Number 92 in Regional Conference Series in Mathematics*. American Mathematical Society, Providence, RI.

Deng,M. *et al*. (2003) An integrated probabilistic model for functional prediction of proteins. *J. Comp. Biol*., **11**, 463–475.

Friedberg,I. (2006) Automated protein function prediction–the genomic challenge. *Brief. Bioinformatics*, **7**, 225–242.

Friedberg,I. and Godzik,A. (2005) Connecting the protein structure universe by using sparse recurring fragments. *Structure*, **13**, 1213–1224.

Friedberg,I. *et al*. (2006) Jafa: a protein function annotation meta-server. *Nucleic Acids Res*., **34** (Web server issue), W379–W381.

Gribskov,M. and Robinson,N.L. (1996) Use of receiver operating characteristic (roc) analysis to evaluate sequence matching. *Comput. Chem*., **20**, 25–33.

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

Hobohm,U. and Sander,C. (1994) Enlarged representative set of protein structures. *Protein Sci*., **3**, 522–524.

Hou,J.T. *et al*. (2005) Global mapping of the protein structure space and application in structure-based inference of protein function. *Proc. Natl Acad. Sci. USA*, **102**, 3651–3656.

Kanehisa,M. *et al*. (2004) The KEGG resources for deciphering genome. *Nucleic Acids Res*., **32**, D277–D280.

Kristensen,D.M. *et al*. (2006) Recurrent use of evolutionary importance for functional annotation of proteins based on local structural similarity. *Protein Sci*., **15**, 1530–1536.

Lanckriet,G.R.G. *et al*. (2004a) Kernel-based data fusion and its application to protein function prediction in yeast. In *Proceedings of the Pacific Symposium on Biocomputing (PSB)*, **9**, pp. 300–311.

Lanckriet,G.R.G. *et al*. (2004b) A statistical framework for genomic data fusion. *Bioinformatics*, **20**, 2626–2635.

Laskowski,R.A. *et al*. (2005) Profunc: a server for predicting protein function from 3d structure. *Nucleic Acids Res*., **33**, W89–W93.

Lee,I. *et al*. (2004) A probabilistic functional network of yeast genes. *Science*, **306**, 1555–1558.

Lichtarge,O. *et al*. (1996) An evolutionary trace method defines binding surfaces common to protein families. *J. Mol. Biol*., **257**, 342–358.

Lisewski,A. M. and Lichtarge,O. (2006) Rapid detection of similarity in protein structure and function through contact metric distances. *Nucleic Acids Res*., **34**, e152.

Noble,W.S. *et al*. (2005) Idetifying remote protein homologs by network propagation. *FEBS J*., **272**, 5099–5100.

Pal,D. and Eisenberg,D. (2005) Inference of protein function from protein structure. *Structure*, **13**, 121–130.

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

Shin,H. *et al*. (2006) Graph-based semi-supervised learning with sharper edges. In *Proceedings of the 17th European Conference on Machine Learning (ECML 2006). Lecture Notes in Computer Science 4212*. Springer, Berlin-Heidelberg, pp. 402–413.

Tsuda,K. *et al*. (2005) Fast protein classification with multiple networks. *Bioinformatics*, **21**, 59–65.

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

Vert,J.P. and Kanehisa,M. (2003) Graph-driven features extraction from microarray data using diffusion kernels and kernel CCA. In Becker,S. *et al*. (eds.), In *Advances in Neural Information Processing Systems 15*. MIT Press, Cambridge, MA, pp. 1425–1432.

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

Watson,J.D. *et al*. (2005) Predicting protein function from sequence and structural data. *Curr. Opin. Struct. Biol*., **15**, 275–284.

Yona,G. *et al*. (1999) Protomap: automatic classification of protein sequences, a hierarchy of protein families, and local maps of the protein space. *Proteins Struct., Funct., Genet*., **37**, 360–678.

Zhou,D. *et al*. (2004) Learning with local and global consistency. In *Advances in Neural Information Processing Systems (NIPS) 16*. MIT Press, Cambridge, MA, pp. 321–328.

Zhu,J. and Weng,Z. (2005) Fast: a novel protein structure alignment algorithm. *Proteins*, **14**, 417–423.