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

Hyunjung Shin¹,*,†, Andreas Martin Lisewski²,† and Olivier Lichtarge²

¹Department of Industrial & Information Systems Engineering, Ajou University, San 5, Wonchun-dong, Yeoungtong-gu, 443—749, Suwon, Korea and ²Department 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

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.

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 PDSelect25 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 \( x_i \) (\( i = 1, \ldots, n \)) 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 \( \sigma \), is used to specify connection strength:

\[
\begin{align*}
W_{ij} = \begin{cases} 
\exp\left(-\frac{(x_i-x_j)\cdot(x_i-x_j)}{\sigma^2}\right) & \text{if } i \sim j, \\
0 & \text{otherwise.}
\end{cases}
\end{align*}
\]

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 \( r \) that is encoded in element \( w_{ij} \) of weight matrix \( W \) with Euclidean distance \( ||x_i-x_j||^2 < r \).

The labelled nodes have labels \( y_i \in \{-1, 1\} \), while the unlabelled nodes have zeros \( y_i = 0 \). Our algorithm

\[ w_{ij} = \begin{cases} \exp\left(-\frac{(x_i-x_j)\cdot(x_i-x_j)}{\sigma^2}\right) & \text{if } i \sim j, \\
0 & \text{otherwise.} \end{cases} \]

will output an \( n \)-dimensional real-valued vector \( f = [f_1, f_2, \ldots, f_n]^T = (f_1, f_2, \ldots, f_{n-m+u})^T \), which can be thresholded to make label predictions on \( f_{n-m+u+1}, \ldots, f_n \) 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 \( f \) by minimizing the following quadratic functional (Belkin et al., 2004, Chapelle et al., 2003; Zhou et al., 2004):

\[
\min f^T (f - y) + \frac{1}{2} \| f \|^2
\]

where \( y = (y_1, \ldots, y_n, 0, \ldots, 0)^T \), and the matrix \( L \), called the graph Laplacian matrix (Chung, 1997), is defined as \( L = D - W \) where \( D = \text{diag}(d_i, \ldots, d_n) \), which is based on 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 \( \mu \) trades off loss versus smoothness. The solution of this problem is obtained as

\[
f = (I + \mu L)^{-1} y
\]

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. 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 ‘?’.

1We represent scalars as lower case, vectors as boldface lower case and matrices are uppercase. 0 (or 1) is a vector or matrix of variable-dependent size containing of all zeros (or ones).
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 $\mathbf{W}$. For simplicity, we set the value of ‘1’ as a weight on every edge. Remember that, in the interpretation of $\mathbf{W}$, 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.

The diagram shows:

- **Figure 2a**: An original undirected (bi-directed) graph and its symmetric weight matrix $\mathbf{W}$. Note that in the interpretation of $\mathbf{W}$, 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.

**Graph sharpening plus graph integration**
a non-negative diagonal matrix (including the null matrix) and $W_{lu}$ to 0 such as

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

(3)

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

$$f_u = \mu(I + \mu(D_{uu} - W_{uu}))^{-1} W_{ul} y_l.$$  

(4)

In spreading activation network terms, Equation (4) is equivalent to information being propagated from labelled to unlabelled data once $(W_{ul} y_l)$ to set the initial condition for subsequent spreading activation among $u \rightarrow u$, analogous to Equation (2) but now excluding the labelled points. This also has intuitive appeal. First, for labelled points, it assures $f_l = y_l$—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, $u \rightarrow u$ 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 $\alpha$ 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,

$$\min_{\alpha} y^T(I + \sum_{k=1}^{K} \alpha_k L_k)^{-1} y$$

(5)

and its output prediction

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

(6)

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

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. 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.](image)

![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 $\alpha$ for individual graph Laplacians $L_k$.](image)
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.

### 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)

### Table 1. Twenty functional GO terms from the category ‘Molecular function’

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

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

**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 \(\times 15\) GO terms \(\times 4\) graphs) pairs in total, sharpening gave a higher AUC; (b) integrated (sharpened) versus individuals (sharpened): for 212 out of 300 (= 5 \(\times 15\) \(\times 4\)) experiments, integration gave a higher AUC and (c) integration with sharpening (2007) versus without sharpening (2005): for 70 out of 75 (= 5 \(\times 15\)) 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 (Gribkov 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 negative 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 μ ∈ {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 α 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 × 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 × 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 previous 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 × 10^-15). Integration alone can be insignificant and

---

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

<table>
<thead>
<tr>
<th>GO terms</th>
<th>BLAST</th>
<th>CM</th>
<th>FAST</th>
<th>ETA</th>
<th>Integrated</th>
</tr>
</thead>
<tbody>
<tr>
<td>Original</td>
<td>0.71</td>
<td>0.64</td>
<td>0.65</td>
<td>0.51</td>
<td>0.66</td>
</tr>
<tr>
<td>Sharpened</td>
<td>0.69</td>
<td>0.63</td>
<td>0.64</td>
<td>0.50</td>
<td>0.68</td>
</tr>
<tr>
<td>AUC increase by sharpening</td>
<td>0.02</td>
<td>0.01</td>
<td>0.06</td>
<td>0.01</td>
<td>0.08</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>GO terms</th>
<th>CM</th>
<th>FAST</th>
<th>ETA</th>
<th>Integrated</th>
</tr>
</thead>
<tbody>
<tr>
<td>Original</td>
<td>0.64</td>
<td>0.65</td>
<td>0.51</td>
<td>0.66</td>
</tr>
<tr>
<td>Sharpened</td>
<td>0.63</td>
<td>0.64</td>
<td>0.50</td>
<td>0.68</td>
</tr>
<tr>
<td>AUC increase by integration</td>
<td>0.01</td>
<td>0.06</td>
<td>0.01</td>
<td>0.08</td>
</tr>
</tbody>
</table>

---

H.Shin et al.
even worse (see, value \(0.02\pm0.08\) 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 \(0.17\pm0.08\), see Table 2. Hence, given the performance of single unsharpened (original) graphs, one can raise the AUC as much as \(0.17\pm0.08\) 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 (±10.86) CPU seconds for original, while it was avg. 4.66 (±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 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.

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 synergistic 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