BIOINFORMATICS ORIGINAL PAPER Vol. 26 no. 14 2010, pages 1759–1765
doi:10.1093/bioinformatics/btq262

Data and text mining Advance Access publication May 27, 2010

Fast integration of heterogeneous data sources for predicting gene function with limited annotation

Sara Mostafavi[1,2,*] and Quaid Morris[1,2,*]
[1]Department of Computer Science and [2]Center for Cellular and Biomolecular Research, University of Toronto, Canada
Associate Editor: Jonathan Wren

ABSTRACT

Motivation: Many algorithms that integrate multiple functional association networks for predicting gene function construct a composite network as a weighted sum of the individual networks and then use the composite network to predict gene function. The weight assigned to an individual network represents the usefulness of that network in predicting a given gene function. However, because many categories of gene function have a small number of annotations, the process of assigning these network weights is prone to overfitting.

Results: Here, we address this problem by proposing a novel approach to combining multiple functional association networks. In particular, we present a method where network weights are simultaneously optimized on sets of related function categories. The method is simpler and faster than existing approaches. Further, we show that it produces composite networks with improved function prediction accuracy using five example species (yeast, mouse, fly, *Esherichia coli* and human).

Availability: Networks and code are available from: http://morrislab.med.utoronto.ca/~sara/SW

Contact: smostafavi@cs.toronto.edu; quaid.morris@utoronto.ca

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

Received on March 5, 2010; revised on April 25, 2010; accepted on May 16, 2010

1 INTRODUCTION

The past decade has seen a dramatic increase in the quantity and variety of publicly available genomic and proteomic data, and a parallel increase in the number of computational methods to integrate these heterogeneous data in generating predictions about protein and gene function [see Noble and Ben-Hur (2007) for a review]. Many of these methods, often called gene (or protein) function prediction algorithms, use the same basic framework: first, they generate so-called functional association networks that capture information about shared gene (or protein) function implicit in each dataset, then they integrate these networks to generate a single *composite* network which they input, along with a set of labels that describe gene function, to a kernel- or network-based classification algorithm (e.g. Lanckriet *et al.*, 2004; Marcotte *et al.*, 1999; Mostafavi *et al.*, 2008; Myers and Troyanskaya, 2007; Tsuda *et al.*, 2005). Once trained, these classification algorithms assign discriminant values to each gene that can then be thresholded to generate hypotheses about the function of unlabeled genes.

*To whom correspondence should be addressed.

The functional association network is a natural and widely used representation for capturing information about shared gene function from high-throughput data sources. In this representation, nodes correspond to genes or proteins and the edges are weighted according to the evidence implied by a given data source for shared function of the connected nodes. These edge weights are calculated using a similarity metric matched to a given data type; for example, the Pearson's correlation coefficient (PCC) is often used to measure pairwise similarities between gene expression profiles. Once calculated, it is relatively easy to translate these networks into kernels for kernel-based learning methods [e.g. by using a diffusion kernel (Kondor and Lafferty, 2002; Qi *et al.*, 2008)].

An important step in predicting gene function is the construction of a composite network from multiple functional association networks. A common approach is to construct a *function-specific* composite network as a weighted sum of the individual networks such that the weight of each network is determined based on the network's predictiveness of a set of positively labeled genes that are deemed to have the same specific function (Lanckriet *et al.*, 2004; Mostafavi *et al.*, 2008; Tsuda *et al.*, 2005). The positive gene labels are derived from online databases such as Gene Ontology (GO; Ashburner *et al.*, 2000), KEGG (Kanehisa and Goto, 2000) and Enzyme Commission (EC; Bairoch, 2000). These databases provide a controlled vocabulary describing categories of gene function and curated lists of genes annotated to these functions.

There are two challenges in constructing function-specific composite networks. First, because many functional categories have only a few annotations, it is difficult to assign network weights without overfitting. Second, for an algorithm to be widely applicable it must be fast and scalable to combine dozens of networks with over 10 000 nodes (genes) each.

Here, we investigate a number of network weighting schemes to avoid overfitting. In particular, we propose a new approach that we refer to as *Simultaneous Weights* (SWs). SW is based on our previous algorithm, GeneMANIA (Mostafavi *et al.*, 2008), which constructs function-specific composite network by solving a constrained linear regression problem. However, instead of assigning function-specific network weights, we simultaneously optimize the weights on a group of related function categories by solving a single-constrained linear regression problem. We evaluate the impact of several regularization schemes such as LASSO (Tibshirani, 1996), elastic net (Zou and Hastie, 2005), ridge regularization on our previous weighting scheme (Mostafavi *et al.*, 2008) and SW. Compared with other state-of-the-art methods in gene function prediction, SW results in a drastic improvement in performance while reducing the computation time requirement of gene function prediction on five example species (yeast, fly, mouse, human and *Escherichia coli*).

2 RELATED WORK

There are large number of algorithms that extend simple guilt-by-association when predicting gene function from a single network including (Karaoz *et al.*, 2003; Nabieva *et al.*, 2005; Vazquez *et al.*, 2003). The approaches closest to those presented in this article are methods for integrating multiple functional association networks into one composite network with the goal of predicting gene function from the composite network. In the seminal work of Marcotte *et al.* (1999), a composite network is constructed with an edge between two genes if the two genes are linked together in the majority of the underlying functional association networks. Similarly, in Pavlidis *et al.* (2002) a composite network is constructed as an unweigted sum of several functional association networks, each derived from a different data source. More recently, in Lanckriet *et al.* (2004); Tsuda *et al.* (2005) and Mostafavi *et al.* (2008), function-specific composite networks are constructed as a weighted sum where the weight of each network is determined based on the function being predicted. In Lanckriet *et al.* (2004) and Tsuda *et al.* (2005), the network weights are assigned to optimize the performance of support vector machine (SVM) and Gaussian random fields (GRFs), respectively, which use the composite network to predict gene function. In Mostafavi *et al.* (2008), we use linear regression to optimize an objective function inspired by the kernel target alignment (Cristianini *et al.*, 2002) of the composite network and the class labels. Another method for combining multiple association network was presented in Myers and Troyanskaya (2007) where a combined network was constructed using a naive Bayes classifier.

The new approach that we present here, SW, extends GeneMANIA algorithm (Mostafavi *et al.*, 2008) that was previously shown to have the state-of-art performance on yeast and mouse benchmark datasets (Mostafavi *et al.*, 2008; Pena-Castillo *et al.*, 2008). However, achieving good performance with the GeneMANIA algorithm in categories with a small number of annotations required a time-consuming regularization procedure. Here, we investigate how to improve the performance for function categories with few annotations without increasing computation time.

3 ALGORITHM

Following the framework of Mostafavi *et al.* (2008), our approach for predicting gene function from multiple networks consists of two steps: (i) it constructs a composite network from multiple functional association networks and (ii) it predicts gene function from a single composite network. Below, we first review the constrained linear regression problem solved by the GeneMANIA algorithm for assigning network weights; next we describe SW, our new approach for assigning network weights using related categories of gene function. Finally, we briefly review how gene function is predicted from a single composite network.

3.1 Combining networks with linear regression

We assume that we are given as input m networks, which we index by d, $W_d \in \mathbb{R}^{n\times n}$, $W_d = W_d{}^{\mathsf{T}}$, where the (i,j)-th element of W_d, $w_{ij}^d \geq 0$ for all i and j. We interpret w_{ij}^d as the strength of the evidence of co-functionality between genes i and j as derived from dataset d. Using annotation databases such as GO, for each GO term that describes a given category of gene function c, positive genes are defined as genes that are annotated to c and we consider all other genes as negatives: that is, we define a label vector $\vec{y} \in \{+1,-1\}^n$, where positive and negative genes are labeled as $+1$, -1, respectively. Our goal is to construct a composite network as a weighted sum of the m networks $W^* = \sum_d^m \mu_d W_d$, where μ_d is the weight assigned to network d, such that W^* can be used to predict other positive genes.

To assign the network weights, GeneMANIA solves a constrained linear regression problem by minimizing the least squares error between the composite network and the *target* network T which represents the pairwise functional relationships implied by the label vector:

$$\vec{\mu}^* = \text{argmin}_{\vec{\mu}} \ \text{trace}((T-W^*)^{\mathsf{T}}(T-W^*)), \quad (1)$$

$$\text{s.t.} \quad W^* = \sum_{d=1}^{m} \mu_d W_d, \ \mu_d \geq 0 \ \ d = \{1,...,m\}$$

where the target network T has elements T_{ij} taking one of the two values: $(\frac{n^-}{n})^2$ if genes i and j are both positive and $-\frac{n^+ n^-}{n^2}$ if genes i and j have opposite signs, where n^+ and n^- are the number of positives and negatives in $\vec{y}$. Since the negative–negative pairs of genes typically do not form a coherent class, it is more appropriate to solve a one-class problem: in GeneMANIA this is addressed by removing the entries in T and each network W_d that correspond to negative pairs of genes (i.e. T_{ij} and w_{ij}^d with i and j both negative).

The non-negative constraint in Equation (1) ensures that the Laplacian matrix L which is derived from W^*, $L = D - W^*$ (where D is a diagonal matrix of the row sums of W^*, i.e. $D_{ii} = \sum_{j=1}^n w_{ij}^*$) is positive semi-definite. As we will show later, we need this condition to use W^* for making predictions about gene function.

By using the fact that $\text{trace}(WT) = \text{vec}(W)^{\mathsf{T}}\text{vec}(T)$, where $\text{vec}(W)$ is an operator that stacks the columns of matrix W atop of each other, we can write (1) as a non-negative linear regression problem:

$$\vec{\mu}^* = \text{argmin}_{\vec{\mu}} \ (\vec{t} - \Omega\vec{\mu})^{\mathsf{T}}(\vec{t} - \Omega\vec{\mu}), \ \ \mu_d \geq 0, d = \{1,...,m\} \quad (2)$$

where $\vec{t} = \text{vec}(T)$, $\Omega = [\text{vec}(W_1),\ldots,\text{vec}(W_m)]$ and $\vec{\mu} = [\mu_1,...,\mu_m]^{\mathsf{T}}$. In practice, we include a column of ones in Ω and calculate a bias μ_0 that we discard when constructing W^*. Unlike the other values of μ_d, μ_0 is not constrained to be positive. Solving Equation (2) requires at most m iterations (though in practice the number of iterations is much smaller), each iteration involves solving a system of linear equations with m variables and a matrix-vector product. As m (the number of networks) tends to be smaller than 100, we can compute the network weights very fast (e.g. in seconds on a standard computer).

3.2 Combining networks with SWs

Although the above approach is fast, it often performs poorly in predicting categories that have a small number of annotations (as discussed in Section 5). In Mostafavi *et al.* (2008), it was shown that an ℓ_2 norm regularization (also known as ridge regression) to a *mean weight* prior, improves performance in such categories (Section 5.1.1). However, assessing this prior requires solving several regression problems. Here, we define a simple modification that improves the performance without increasing the computational time and show that it performs better than previous approaches. In particular, in SW, instead of assigning network weights for each category separately, we fit the network weights to a set of related function categories. To do so, we assign the network weights by solving the following problem:

$$\vec{\mu}^* = \underset{\vec{\mu}}{\arg\min} \sum_{c=1}^{h} (\vec{t}_c - \Omega\vec{\mu})^\mathsf{T}(\vec{t}_c - \Omega\vec{\mu}),\ \mu_d \geq 0$$

where t_c for $c=1,...,h$ are constructed from h positively labeled genes sets (categories) that are related to each other. Once we obtain $\vec{\mu}^*$, we construct W^* and use it to predict all h categories.

If we include all entries of Ω (i.e. not excluding negative–negative pairs of genes for each category h as described above), we can then write the above problem as:

$$\vec{\mu}^* = \arg\min_{\vec{\mu}} -2\vec{\mu}^\mathsf{T}\Omega^\mathsf{T}\tilde{\vec{t}} + h\vec{\mu}^\mathsf{T}\Omega^\mathsf{T}\Omega\vec{\mu}$$

where $\tilde{\vec{t}} = \sum_{c=1}^{h} \vec{t}_c$ and so we only need to solve the regression problem once to get the SWs. As such, for each category, T_{ij}^c takes on one of the three possible values: $(\frac{n_c^+}{n})^2, (\frac{n_c^-}{n})^2, -\frac{n_c^- n_c^+}{n^2}$ when i,j are both negative, both positive and have the opposite signs, respectively, and $n_c^+ (n_c^-)$ is the number of positives (negatives) in category c. As we will show, including the negative–negative pairs of genes in t_c and Ω does not degrade the performance of the constructed composite network.

In our experiments, constructing $\tilde{\vec{t}}$ takes <5 s with $h=1000$ on a standard computer (2.4 GHz Intel Core 2 Duo, 4 GB RAM). Further, for a given set of networks, we can always precompute $\Omega^\mathsf{T}\Omega$ and thus only need to calculate $\Omega^\mathsf{T}\tilde{\vec{t}}$ for a group of categories of interest. As we will show in Section 5, combining weights by SWs results in an improvement in the performance of the composite networks in predicting the relevant h gene categories while it reduces the computation time (as now we are only required to solve for the network weights once when predicting h categories).

3.3 Predicting protein function from a single network

We evaluate a composite network, W^*, by its ability to predict a given gene function. As done in Mostafavi *et al.* (2008), we use the GRFs algorithm (Zhou *et al.*, 2004; Zhu *et al.*, 2003) to predict gene function from a single composite network. In particular, given a label vector $\vec{y}$ where y_i represents the prior evidence for gene i having the function of interest, the GRF algorithm assigns a discriminant score $f_i \in [-1,1]$ to each node (gene) i in the network which we can then threshold to classify the genes. In particular, $y_i = \{-1, k, +1\}$ where known negative and positive genes are assigned -1 and $+1$, respectively, and the unlabeled genes (i.e. the possibility set) are assigned a value $-1 \leq k \leq +1$, for example, k can be adjusted based on a gene's annotations in GO (Mostafavi and Morris, 2009).

We can write the GRFs algorithm in the following general form:

$$\begin{aligned}\vec{f}^* &= \underset{\vec{f}}{\arg\min} \sum_{i=1}^{n} \sigma_i (y_i - f_i)^2 + \sum_{i,j=1}^{n} w_{ij}^* (f_i - f_j)^2 \qquad (3)\\ &= \underset{\vec{f}}{\arg\min}\ (\vec{f} - \vec{y})^\mathsf{T} \Sigma (\vec{f} - \vec{y}) + \vec{f} L \vec{f}\\ &= (\Sigma + L)^{-1} \Sigma \vec{y}\end{aligned}$$

where $[\sigma_1,...,\sigma_n]^\mathsf{T}$ are model parameters, Σ is a diagonal matrix with $\Sigma_{ii} = \sigma_i$, $L = D - W$ is the graph Laplacian and D is a diagonal matrix with $D_{ii} = \sum_j w_{ij}^*$. The above objective ensures that the discriminant scores remain close to their initial labels [first term in (3)] and that the discriminant scores of genes likely to share a function (measured by high w_{ij}^*) are similar to each other [second term in (3)]. As done in Mostafavi *et al.* (2008), we set $k = \frac{n^+ - n^-}{n^+ + n^-}$, the mean of the labels of the labeled nodes; this modification results in considerable performance improvement in unbalanced classification problems such as gene function prediction.

Setting $\sigma_i > 0$, ensures that $\Sigma + L$ is invertible because $\Sigma + L$ is diagonally dominant; in our experiments, we set $\Sigma = I$ (the identity matrix). To solve for $\vec{f}$, we only need to solve a linear system of equations $M\vec{y} = \vec{f}$, where $M = (I + L)$, which we can do with various existing fast iterative solvers (Nocedal and Wright, 2006). We use conjugate gradient (CG). CG is guaranteed to terminate in n steps, the most time-consuming operation at each step being a matrix-vector product with a computational complexity proportional to the number of non-zero elements in L; in our setting L is very sparse, with $\mathcal{O}(n)$ non-zero elements, and CG terminates in fewer than 20 iterations.

4 METHODS

In this section, we describe our benchmark datasets, how we construct functional association networks, our evaluation criterion and how we group function categories in SW (see the Supplementary Material for more detailed information).

4.1 Yeast, fly, mouse, human and *E.Coli* datasets

We evaluate our methodology on benchmark networks in five species: yeast, fly, mouse, human and *E.coli*. For yeast, we constructed 44 networks that include interactions derived from gene expression, protein and genetic interaction [downloaded from BIOGRD (Stark *et al.*, 2006)] and protein localization. For mouse, we use the MouseFunc benchmark (Pena-Castillo *et al.*, 2008), which consists of 10 networks and covers 21 603 mouse genes. For fly, we have constructed 38 networks from various gene expression data [downloaded from GEO (Edgar *et al.*, 2002)], protein interaction (downloaded from BioGRID) and domain composition [downloaded from BioMART (Kasprzyk *et al.*, 2004)] that cover 13 562 fly genes. For *E.coli*, we use seven networks from Hu *et al.* (2009) that include co-inheritance and protein interactions for 4175 *E.coli* genes. Similarly, our human benchmark consists of eight networks constructed from various gene expression, protein interaction, domain composition and phenotype data and covers 13 281 human genes obtained from HPRD (Prasad *et al.*, 2006).

4.2 Functional association networks

We construct networks from each profile-based high-throughput data source using the PCC. For network-based data (e.g. protein interaction), we use both a direct interaction network and a correlation-based network using the PCC on the frequency-corrected data [as done in Mostafavi *et al.* (2008)]. For efficiency, we sparsify our correlation-based networks by setting by keeping the top K interactions for each gene and setting the rest to zero. See the Supplementary Material for more details. We then normalized all our networks by: $\tilde{W}_d = D_d^{-1/2} W_d D_d^{-1/2}$ where D_d is the diagonal row sum matrix of W_d. Similarly, we also normalize the combined network W^*.

4.3 Evaluation

To evaluate gene function prediction, we use the GO biological process (BP) function categories (Ashburner *et al.*, 2000) for *Saccharomyces cerevisiae*

(June 2006), *Mus musculus* [downloaded from MouseFunc data (Pena-Castillo *et al.*, 2008)], *Drosophila melanogaster* (July 2009), *Homo sapiens* (July 2009) and *E.coli* (April 2010). Following common practice, we have removed Inferred from Electronic Annotation (IEA) annotations. These annotations, which constitute the majority of GO annotations, are not reviewed by a curator and, as such, are believed to be less accurate. Furthermore, doing so helps us to avoid circularity because IEAs are themselves computationally predicted using some of the data that we make available to our algorithms.

We evaluate each method's composite networks by using them as input to the GRF algorithm (the second step in GeneMANIA). We report the performance in terms of both average area under the receiver operating characteristic (ROC) curve (AUC of ROC) and average precision at 10% recall over all BP GO categories with 3–300 annotations using 3-fold cross-validation (CV). We focus on BP categories because they make up the majority of functions in the GO hierarchy. Our results for cellular component (CC) and molecular function (MF) categories are similar and are described in the Supplementary Material.

4.4 Grouping GO categories for simultaneous weights

We have examined several methods for grouping GO categories when assigning SWs including grouping by (i) GO hierarchy (i.e. BP, CC and MF) (ii) GO hierarchy and number of annotations (iii) clustering of GO categories based on annotations and (iv) ancestor and descendant terms with ancestors having a maximum category size (300 annotations). We only report results for the grouping of GO categories by GO hierarchy and number of annotations (e.g. all categories with less than 300 annotations) as we found it to have the best performance (Supplementary Material).

5 RESULTS

Here, we first evaluate SW and compare its performance to several other approaches: various regularized linear regression methods, the TSS algorithm and a simpler correlation-based method (described below), using the yeast benchmark networks. We then show analogous results using mouse, fly, human and *E.coli* benchmark data.

5.1 Performance on yeast networks

5.1.1 Comparison of performance of SW with various function-specific linear regression methods We first extensively compare the performance of SW in predicting gene function in yeast to that of GeneMANIA. In particular, as discussed in Section 4, one way to improve the performance of function-specific constrained linear regression in GeneMANIA is to use regularization; in fact, Mostafavi *et al.* (2008) showed that ridge regression (i.e. ℓ_2 norm regularization) to a *mean weight* prior, where the mean weights refer to the average weight assigned to each network in a large number of function predictions, considerably improves the performance with the drawback of increasing the computation time to estimate the mean weights. Here, we investigate the effect of several forms of regularization on the performance of GeneMANIA algorithm where we find the network weights by solving the following problem:

$$\vec{\mu}^* = \underset{\vec{\mu}}{\operatorname{argmin}} \ (\vec{t}-\Omega\vec{\mu})^{\mathsf{T}}(\vec{t}-\Omega\vec{\mu})+J(\vec{\mu}) \ , \ \vec{\mu}\geq 0$$

where $J \geq 0$ is the regularization function. In particular, we investigated the performance of four different regularizations: (i) ridge with uniform prior, (ii) ridge with mean prior, (iii) LASSO and (iv) elastic net. In LASSO (Tibshirani, 1996), $J(\vec{\mu})=\alpha_1\sum_{d=1}^{m}|\mu_d|$, whereas in standard ridge regression $J(\vec{\mu})=\alpha_2\sum_{d=1}^{m}\mu_d^2$ where α_1 and α_2 are regularization constants and determine the strength of the regularization. The elastic net regularization (Zou and Hastie, 2005) combines l_2- and l_1-norm penalties: $\alpha_1\sum_{d=1}^{m}|\mu_d|+\alpha_2\sum_{d=2}^{m}\mu_d^2$. In Zou and Hastie (2005), it was shown that the elastic net results in a sparse solution and often performs better than the LASSO.

For ridge with a prior, we define $J(\vec{\mu})=\sum_{d=1}^{m}(\mu_d-v_d)^2 s_d$, where $\vec{v}$ is a prior weight vector and s_d determines the strength of the regularization on μ_d. In Mostafavi *et al.* (2008), the mean weight prior was obtained as the average weight assigned to each category (using unregularized regression) in predicting all categories in the same GO hierarchy (we will refer to this method as ridge with mean prior). In addition, if we set $v_d = 1$ the network weights are shrunk to a uniform value, we call this second method ridge with uniform prior. In our experiments, we set $s_d = 1/\sum_{ij} w_{ij}^d$; thus, the strength of the regularizer is higher on sparser networks.

Figure 1a summarizes the performance of each method in five evaluation categories: predicting gene functions which have [3–10], [11–30], [31–100], [101–300] positive annotations and [3–300] (i.e. overall) positive annotations. In ridge with mean prior, we set the prior on each network's weight to the average weight that network received in predicting all 1188 GO BP categories with 3–300 annotations. We used the LARS (Efron *et al.*, 2004) algorithm to solve for the LASSO and elastic net solutions; we set the number of positive coefficients using F-statistics (Hastie *et al.*, 2001). For elastic net, we set α_2=1e-6 using CV.[1] For SW, we used all 1188 BP GO categories to fit the networks weights. In Uniform, the network weights are all set to $1/m$ where m is the number of networks.

Figure 1a shows that SW significantly outperforms ridge regression with mean prior overall in terms of ROC ($P = 4.368 \times 10^{-23}$, Wilcoxon signed rank test) and slightly improves on the performance in terms of precision ($P = 0.0437$, Wilcoxon signed rank test) with the advantage that it only requires solving one linear regression problem to predict all 1188 GO categories (instead of 1188 for the function-specific network weighting methods). In addition, this figure shows that unregularized linear regression performs as well as or better than LASSO, ridge or elastic net regularization whereas ridge with a prior results in a better performance in all evaluation categories. However, as expected, we see that the performance of unregularized regression improves with increasing number of positives and thus it is more appropriate to use function-specific weighting in such instances.

One explanation for the observed trend in Figure 1a is that regularization methods that shrink the network weights toward zero are too selective and often identify only a few relevant networks. For example, on average 45% (20/44), 54% (24/44) and 95% (42/44), 97% (43/44) of the networks are assigned a non-zero weight using LASSO, unregularized linear regression and ridge with mean prior, and SW, respectively (see Supplementary Fig. S1). Note that the best performing networks on their own are significantly worse than the combined data (Fig. 1b). SW results in a better measure of network relevancy and with the current available genomics and proteomics datasets, one integrated composite functional association network can sufficiently and accurately predict a broad range of functional relationships.

[1]We picked α_2 from the set [1e-8, 1e-6, 1e-4, 1e-2, 1e-1, 1] by examining the mean ROC using 3-fold CV.

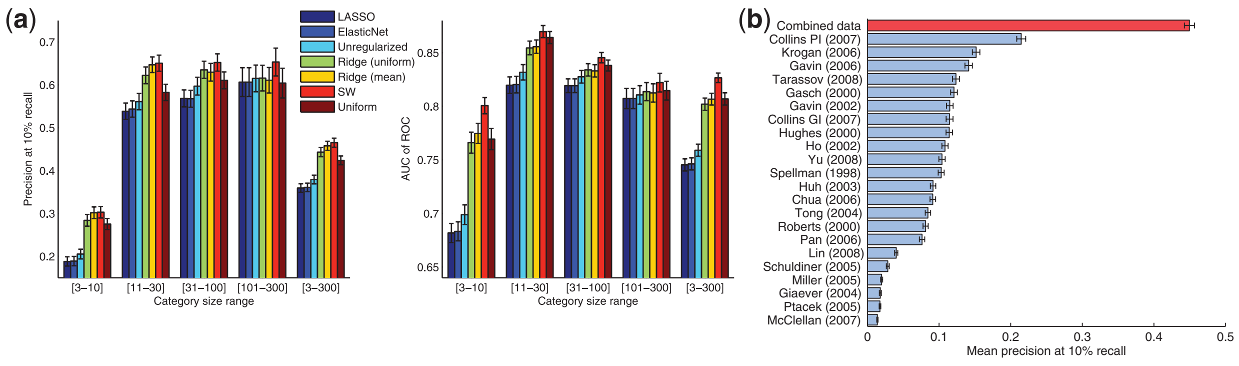

Fig. 1. (**a**) Comparison of performance of LASSO, elastic net (ElasticNet), unregularized linear regression (Unregularized), ridge with uniform prior [Ridge (Uniform)], ridge with mean prior [Ridge (mean)], SW and a network combination with uniform weights (Uniform) in predicting BP categories with [3–10] ($n=635$), [11–30] ($n=305$), [31–100] ($n=191$), [101–300] ($n=57$) and [3–300] ($n=1188$) annotations. Error bars show one standard error. (**b**) Mean precision of combined and individual data sources (separated by publications) in predicting BP categories. When applicable (in the case of protein and genetic interaction) we combined all networks derived from the same publication (e.g. direct and correlation network). The combined network was constructed using SW.

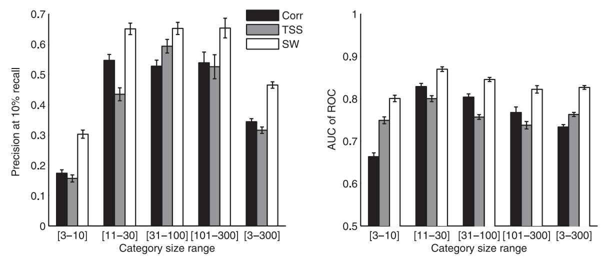

Fig. 2. Performance of SW, TSS and correlation in predicting gene function in yeast according to BP categories.

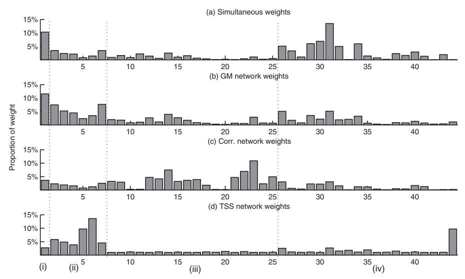

Fig. 3. Each colored bar represents the average weight assigned to each network while predicting 1188 gene functions. Networks are divided into four types (i) co-localization (network 1), (ii) gene expressions (networks 2–7), (iii) protein interaction (networks 8–25) and genetic interactions (networks 26–44).

5.1.2 Comparison of SW with TSS and correlation-based network weights We also compared the performance of SW with two other methods: TSS algorithm (Tsuda *et al.*, 2005) and a simpler correlation-based network weighting method (Fig. 2). In the correlation network weighting, each network is assigned a weight that is inspired by the Kernel Target Alignment score—we set $\mu_d = \frac{\vec{y}^\top W_d \vec{y}}{\langle W_d, W_d \rangle} = \frac{\sum_{ij} w_{ij}^d y_i y_j}{\sum_{ij}(w_{ij}^2)}$. Unlike the linear regression methods, correlation-based weighting does not account for the redundancy between the networks. The TSS algorithm (Tsuda *et al.*, 2005) assigns the network weights by optimizing the performance of the GRFs algorithm with the resulting composite network. In our experiments, we set the regularization parameters of the TSS algorithm by CV to $c_0 = 0.5$ and $c = 1$. As done in code provided in Tsuda *et al.* (2005), we also set a lower bound of 0.01 on μ_d. We note that the absence of the lower bound results in a decrease in the performance of the TSS algorithms. As shown in Figure 2, SW significantly outperforms correlation-based network weights and TSS in all evaluation categories.

To further understand the differences between these various approaches, we compare the network weights that were assigned to individual networks. As shown in Figure 3, we observed that the TSS algorithm tends to be very selective, often assigning large weights to a few networks and a very low weight (the weight lower bound) to the rest. The correlation weights are similar to the linear regression weights; however, the redundancy between the networks is not accounted for. For example, the protein interaction networks (shown as cyan and green) drawn from separate publications tend to include similar information and the average of weights assigned to these networks by correlation weights is higher than that of linear regression. As expected, the mean weight assigned by linear regression to individual networks is similar to SW for that network. In general, consistent with previous studies (Marcotte *et al.*, 1999), we observed that all methods assign a high proportion of the network weights to the networks derived from gene expression datasets and the protein localization dataset.

5.2 Performance on fly, mouse, human and *E.coli* benchmarks

We also investigated the performance of unregularized linear regression, SW and uniform network weights on fly, mouse, human and *E.coli* networks in all GO categories that have between 3 and 100 annotations (2195 for fly, 1626 for mouse, 1952 for human, and 809 for *E. coli*). Figure 4 summarizes the performance in terms of AUC of ROC curve and precision at 10% recall in the four species. As shown, SW is significantly better than unregularized linear regression in the overall category in fly, mouse, human and *E.coli* in terms of AUC of ROC. As well, SW is significantly better than uniform

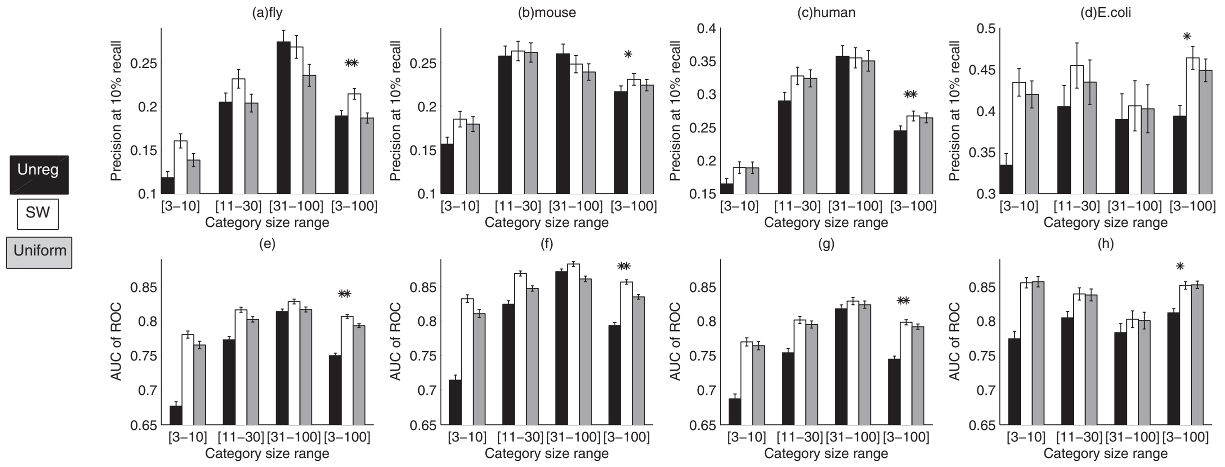

Fig. 4. Comparison of performance of unregularized linear regression (Unreg), SW and a fixed uniform combination of networks in predicting gene function in fly (**a** and **e**), mouse (**b** and **f**), human (**c** and **g**) and *E.coli* (**d** and **h**). The bars show average performance in BP categories with [3–10] ($n=1101$ for fly, 952 for mouse, for 1188 for human, 528 for *E.coli*) [11–30]($n=668$ for fly, 435 for mouse, 510 for human and 177 for *E.coli*), [31–100] ($n=426$ for fly, 239 for mouse, 254 for human and 104 for *E.coli*) and [3–100] (overall). Error bars show the standard error. Asterisk indicate significant difference in overall performance ([3–100] category size range) using paired Wilcoxon signed rank test with a Bonferroni correction: double asterisk indicate SW performs significantly better than both of the other methods, asterisk indicates that the differences were significant only between SW and unregularized.

and unregularized linear regression in terms of precision in fly and human. In mouse, SW significantly outperforms unregularized linear regression in terms of precision. We note that the human networks are sparser than the other organisms, which makes it hard to assign accurate network weights (mean number of interactions is 391 240 in human networks compared with 1 011 400 in mouse), which may explain the smaller (but significant) improvements of SW compared with uniform weights. As well, we note that the performance of uniform weights tends to degrade as the number of networks increases—this is because of the abundance of gene expression datasets and thus the number of co-expression networks. For example, out of the 38 networks for fly, 32 are co-expression networks. By not accounting for redundancy, the performance of uniform weights is significantly worse than that of SW.

6 CONCLUSION

We have introduced a new network weighting scheme for combining multiple networks that are derived from genomic and proteomic data in order to construct a composite network that is predictive of gene function. We have shown that by fitting network weights that are simultaneously optimized on a group of functions from the same branch of GO, we greatly improve prediction performance. We have shown that we can obtain these SWs by solving a constrained linear regression problem. In our experiments, the SW method results in a significant improvement in predicting gene function in yeast, mouse and fly. In human and *E.coli*, SWs performs only slightly better than a uniform network combination; this is because these networks tend to be sparser than the other networks making it hard to assign accurate network weights.

In our experiments, we have observed that adding a small amount of ridge regularization to SW results in a slight performance improvement; the regularization parameter can be set using CV; alternatively, we have observed good performance by setting it to $\sim 0.001\frac{n}{2}(n-1)$ (i.e. 0.1% of the total number of observations). Our results show that fitting the SWs to GO categories in the same hierarchy with a broad range of specificities (those with [3–300] annotations) outperform more specific groupings of the GO categories. Note that, because we adjust the target vector $\vec{t}_c$ to balance the number of positives and negatives in each category c, the larger GO categories contribute more to the overall target vector $\vec{\tilde{t}}$; on the other hand, there are many more categories with [3–10] annotations.

In summary, we have demonstrated the feasibility and the utility of constructing a single composite network with SWs for predicting various GO categories. Unlike a fixed network combination with uniform weights, SWs account for noisy and redundant networks. This observation can in turn speed up gene function prediction from multiple networks.

ACKNOWLEDGMENTS

We would like to thank Khalid Zuberi, Quentin Shao and Javier Diaz for their help with collecting the networks.

Funding: Ontario Graduate Scholarship (partially funded to S.M.), Genome Canada Technology Development Project administered through Ontario Genomics Institute and a Natural Science and Engineering Operating grant (to Q.M.).

Conflict of Interest: none declared.

REFERENCES

Ashburner,M. *et al.* (2000) Gene ontology: tool for unification of biology. *Nat. Genet.*, **25**, 25–29.
Bairoch,A. (2000) The enzyme database in 2000. *Nucleic Acids Res.*, **28**, 304–305.
Cristianini,N. *et al.* (2002) On kernel target alignment. In *Proceedings of the Fourteen Conference on Advances in Neural Information Processing Systems*, Vancouver, BC, Canada, pp. 367–373.
Edgar,R. *et al.* (2002) Gene expression omnibus: NCBI gene expression and hybridization array data repository. *Nucleic Acids Res.*, **30**, 207–210.

Efron,B. *et al*. (2004) Least angle regression. *Ann. Stat.*, **32**, 407–499.

Hastie,T. *et al*. (2001) *The Elements of Statistical Learning*. Springer, NY, USA.

Hu,P. *et al*. (2009) Global functional atlas of escherichia coli encompassing previously uncharacterized proteins. *PLoS Biol.*, **7**, e96.

Kanehisa,K. and Goto,S. (2000) KEGG: Kyoto encyclopedia of genes and genome. *Nucleic Acids Res.*, **28**, 27–30.

Karaoz,U. *et al*. (2003) Whole-genome annotation by using evidence integration in functional-linkage networks. *Proc. Natl Acad. Sci. USA*, **101**, 2888–2893.

Kasprzyk,A. *et al*. (2004) Ensmart: a generic system for fast and flexible access to biological data. *Genome Res.*, **14**, 160–169.

Kondor,R. and Lafferty,J. (2002) Diffusion kernels on graphs and other discrete structures. *Int. Conf. Mach. Learn. (ICML)*, **11**, 463–475.

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

Marcotte,E. *et al*. (1999) A combined algorithm for genome-wide prediction of protein function. *Nature*, **42**, 83–86.

Mostafavi,S. and Morris,Q. (2009) Using the gene ontology hierarchy when predicting gene function. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*. Montreal, QC, Canada.

Mostafavi,S. *et al*. (2008) Genemania: a real-time multiple association network integration algorithm for predicting gene function. *Genome Biol.*, **9** (Suppl. 1), S4.

Myers,C. and Troyanskaya,O. (2007) Context-sensitive data integration and prediction of biological networks. *Bioinformatics*, **23**, 2322–2330.

Nabieva,E. *et al*. (2005) Whole-proteome prediction of protein function via graph-theoretic analysis of interaction maps. *Bioinformatics*, **2** (Suppl. 1).

Noble,W. and Ben-Hur,A. (2007) Integrating Information for Protein Function Prediction. In Lengauer,T. (ed) *Bioinformatics-From Genomes to Therapies*. Wiley-VCH Verlag GmbH & Co KGaA, Weinheim, Germany.

Nocedal,J. and Wright,S. (2006) *Numerical Optimization*. Springer, NY, USA.

Pavlidis,P. *et al*. (2002) Learning gene functional classification from multiple data types. *J. Comput. Biol.*, **9**, 401–411.

Pena-Castillo,L. *et al*. (2008) A critical assessment of *Mus musculus* gene function prediction using integrated genomic evidence. *Genome Biol.*, **9** (Suppl. 1), S2.

Prasad,T.K. *et al*. (2006) Human protein reference database – 2006 update. *Nucleic Acids Res.*, **34**, D411–D414.

Qi,Y. *et al*. (2008) Finding friends and enemies in an enemies-only network: a graph diffusion kernel for predicting novel genetic interactions and co-complex membership from yeast genetic intearctions. *Genome Res.*, **18**, 1991–2004.

Stark,C. *et al*. (2006) BioGRID: a general repository for interaction datasets. *Nucleic Acids Res.*, **1**, D539.

Tibshirani,R. (1996) Regression shrinkage and selection via the lasso. *J. R. Stat. Soc. B.*, **58**, 267–288.

Tsuda,K. *et al*. (2005) Fast protein classification with multiple networks. *Bioinformatics*, **21** (Suppl. 2), ii59–ii65.

Vazquez,A. *et al*. (2003) Global protein function prediction from protein-protein interaction networks. *Nat. Biotechnol.*, **21**, 697–700.

Zhou,D. *et al*. (2004) Learning with local and global consistency. *Adv. Neural Inf. Process. Syst.*, **16**, 321–328.

Zhu,X. *et al*. (2003) Semi-supervised learning using Gaussian fields and harmonic functions. In *Proceedings of the Twentieth International Conference on Machine Learning*, Washington DC, USA.

Zou,H. and Hastie,T. (2005) Regularization and variable selection via the elastic net. *J. R. Stat. Soc. B.*, **67**, 301–320.