Protein Function Prediction Using Multilabel Ensemble Classification

Guoxian Yu, Huzefa Rangwala, Carlotta Domeniconi, Guoji Zhang, and Zhiwen Yu

Abstract—High-throughput experimental techniques produce several kinds of heterogeneous proteomic and genomic data sets. To computationally annotate proteins, it is necessary and promising to integrate these heterogeneous data sources. Some methods transform these data sources into different kernels or feature representations. Next, these kernels are linearly (or nonlinearly) combined into a composite kernel. The composite kernel is utilized to develop a predictive model to infer the function of proteins. A protein can have multiple roles and functions (or labels). Therefore, multilabel learning methods are also adapted for protein function prediction. We develop a *transductive multilabel classifier* (TMC) to predict multiple functions of proteins using several unlabeled proteins. We also propose a method called *transductive multilabel ensemble classifier* (TMEC) for integrating the different data sources using an ensemble approach. The TMEC trains a graph-based multilabel classifier on each single data source, and then combines the predictions of the individual classifiers. We use a directed birelational graph to capture the relationships between pairs of proteins, between pairs of functions, and between proteins and functions. We evaluate the effectiveness of the TMC and TMEC to predict the functions of proteins on three benchmarks. We show that our approaches perform better than recently proposed protein function prediction methods on composite and multiple kernels. The code, data sets used in this paper and supplemental material are available at https://sites.google.com/site/guoxian85/tmec.

Index Terms—Multilabel ensemble classifiers, directed birelational graph, protein function prediction

1 Introduction

Advances in biotechnology have enabled high-throughput experiments that generate high volume of genomic and proteomic data. Examples of these data include protein-protein interaction (PPI) networks, protein sequences, protein structure, gene coexpression data, and genetic interaction networks. Each data source provides a complementary view of the underlying mechanisms within a living cell. Annotating the functions (i.e., biological process functions in the Gene Ontology (GO) [1], transcription, and protein synthesis) of proteins is a fundamental task in the postgenomic era [2], [3]. However, it is time-consuming, expensive, and low productive to manually annotate a protein using complex and large heterogeneous data. Therefore, various computational models have been proposed to automatically infer the functions of proteins by integrating the available data [3], [4].

Kernel-based methods [5], [6] have been widely used to design several bioinformatics-related algorithms. In these approaches, the pairwise similarity between proteins is described by a kernel function $\mathcal{K}$, which captures the underlying biological complexity associated with the proteins. For each data source (i.e., protein sequences, PPI networks), a unique kernel function is defined, and each kernel function captures a different notion of similarity. For example, the string kernel [7] is often used to compute the similarity between protein sequences, and the random walk kernel [2] is utilized on protein-protein interaction (PPI) data. Both the sequence and PPI data sets are transformed into different kernels, each of which captures similarities between protein pairs within different feature spaces (or embeddings). Pavlidis et al. [8], and Noble and Ben-Hur [4] observed that the prediction accuracy can be boosted by taking advantage of complimentary embeddings across different kernels. Many approaches [5], [9], [10] use a linear (or nonlinear) weighted combination (optimal or ad hoc) of multiple kernels obtained from the different sources. These kinds of methods can be categorized as *kernel integration* methods. In addition, supervised ensemble classifiers [8], [11] have also been developed to combine the multiple data sources.

Often, only a few proteins are annotated, and a large volume of proteins remains unlabeled within each single data source. Transductive or semisupervised learning methods are able to make use of unlabeled data to boost the learning results [12]. Therefore, several semisupervised [10], [13], [14] approaches have been proposed for protein function prediction. Further, a protein often holds more than one function and functions are correlated with each other. This fact should be taken into account in protein function prediction. Several approaches [15], [16], [17], [18] formulate the protein function prediction problem within a multilabel learning framework. Multilabel learning methods can make use of the dependences between the different

- *G. Yu is with the College of Computer and Information Science, Southwest University, No. 2 Tiansheng Road, Beibei, Chongqing 410075, P.R. China. E-mail: gxyu@swu.edu.cn.*
- *H. Rangwala and C. Domeniconi are with the Department of Computer Science, George Mason University, 4400 University Drive, Fairfax, VA 22030. E-mail: {rangwala, carlotta}@cs.gmu.edu.*
- *G. Zhang is with the School of Sciences, South China University of Technology, Guangzhou Higher Education Mega Center, Guangzhou, Guangdong 510006, China. E-mail: magjzh@scut.edu.cn.*
- *Z. Yu is with the School of Computer Science and Engineering, South China University of Technology, Guangzhou Higher Education Mega Center, Guangzhou, Guangdong 510006, China. E-mail: zhwyu@scut.edu.cn.*

Manuscript received 3 Apr. 2013; revised 17 Aug. 2013; accepted 29 Aug. 2013; published online 13 Sept. 2013.
For information on obtaining reprints of this article, please send e-mail to: tcbb@computer.org, and reference IEEECS Log Number TCBB-2013-01-0110. Digital Object Identifier no. 10.1109/TCBB.2013.111.

function classes, and they often outperform single-labeled prediction methods.

We propose a protein function prediction method called *transductive multilabel classifier* (TMC). The TMC is based on a directed birelational graph that models the relationship between proteins and functions. To integrate the heterogeneous sources of protein data, we also develop an ensemble-based classifier called the *transductive multilabel ensemble classifier* (TMEC). We performed comprehensive experiments evaluating the performance of TMC and TMEC on three protein function prediction benchmarks (**Yeast**, **Human**, and **Fly**). Each benchmark includes several kinds of heterogeneous data sources (i.e., PPI networks, protein sequences, and gene coexpression data). For more information on these benchmarks, one can refer to prior work by Mostafavi and Morris [9].[1] Our results show that the use of a directed birelational graph achieves higher accuracy than the undirected one. Our proposed TMC outperforms the state-of-the-art protein function prediction approaches, namely, two transductive multilabel classification approaches [15], [18], a transductive classifier [9], and two multilabel multiple kernel learning methods [19], [20]. In addition, the proposed TMEC, which takes advantage of classifier ensembles to make use of multiple heterogeneous data sources, often performs better than the TMC.

This work is an extension of our earlier paper by Yu et al. [21]. In particular, the additional contributions of this paper are as follows:

1. We provide a thorough analysis of the issues due to the use of an undirected birelational graph [22] for network propagation. We advocate the use of a *directed* birelational graph instead, and discuss a toy example to illustrate the risk of using an undirected birelational graph.
2. We test our proposed methods (TMC and TMEC) on three publicly available protein function prediction benchmarks (Yeast, Human, and Fly).
3. We apply different schemes to combine the individual kernels from different data sources into a composite kernel, and compare them against TMEC to further show the advantage of *classifier integration* to *kernel integration* in protein function prediction.

The remainder of this paper is organized as follows: Section 2 reviews related work on protein function prediction using multilabel learning and data integration. In Section 3, we introduce the directed birelational graph approach and the corresponding training procedure, along with a toy example to illustrate the drawbacks associated with the use of an undirected birelational graph. We also describe our ensemble approach to make use of multiple data sources. We describe the experimental protocol, evaluation metrics, and data sources in Section 4. In Section 5, we discuss the experimental results. In Section 6, we provide conclusions along with directions for future work.

2 Related Work

Various computational models have been proposed to predict protein functions. These methods can be categorized according to the terms of methodology, input data, and problem definition. Pandey et al. [3] gave a comprehensive literature review on protein function prediction. Here, we review only the work most related to the scope of this paper.

2.1 Multilabel Learning in Protein Function Prediction

The traditional function prediction methods take protein sequences (or PPI) and functional annotations as input to train one-versus-the-rest binary classification models [2], [8]. These methods ignore the pairwise function correlations. It is observed that a protein performs multiple functions, and structured relationships are prevalent within protein function annotation databases (e.g., Gene Ontology [1][2] is a directed acyclic graph, Functional Catalogue (FunCat) [23][3] is a tree graph). As such, protein function prediction can be formulated as a multilabel classification problem [24].

Multilabel learning is widely used in protein function prediction. Elisseeff and Weston [25] added a ranking loss function into the loss function of SVMs, and proposed a method called RankSVM. RankSVM does not make explicit use of the label correlations [26], which can be important for deciding the label membership [18], [26]. Chen et al. [26] incorporated a label correlation (captured by a hypergraph) term into the objective function of RankSVM. More specifically, in the hypergraph, each label corresponds to a hyperedge, which connects all the proteins that share the same label. Protein function databases like Gene Ontology [1] and FunCat [23] represent protein functions within a hierarchy (or a directed acyclic graph), and several approaches incorporate the parent-children relationship among proteins. Barutcuoglu et al. [27] first train independent binary SVMs for each of the GO function labels, and then integrates the prediction by incorporating GO's hierarchical structure within a Bayes formulation. Valentini [28] used the True Path Rule in FunCat to guide the integration of predictions. Pandey et al. [17] used Lin's measure [29] to define the semantic similarity between the different GO labels and incorporated it within a weighted multilabel k-nearest neighbors (kNN) classifier.

More recently, several semisupervised multilabel learning methods based on PPI networks were proposed for protein function prediction [15], [16], [18]. MCSL [16] is based on a product graph with an objective function similar to the local and global consistency method [30]. MCSL adds another term (function correlation) into the smoothness term (see [16, Eq. (6)]) and results in a Kronecker matrix. For N proteins with C functions, the resulting Kronecker matrix is an $(N \times C) \times (N \times C)$ matrix. This augmented matrix often cannot be loaded in memory and it is computationally expensive. To address this problem, Jiang [15] proposed a protein function prediction approach called PfunBG, which is based on a birelational graph [22] and label propagation [30]. The birelational graph captures three types of relationships: 1) protein-protein similarities, 2) function-function similarities, and 3) protein-function associations, and the association matrix of the birelational graph is an $(N + C) \times (N + C)$

1. http://morrislab.med.utoronto.ca/~sara/SW/.

2. http://www.geneontology.org/.
3. http://mips.helmholtz-muenchen.de/proj/funcatDB/.

matrix. GRF [18] utilizes Jaccard coefficients to measure the correlation between functions, and then incorporates this correlation into a general semisupervised learning framework based on manifold regularization [31]. All the described methods, MCSL, PfunBG, and GRF, utilize pairwise label correlations within a semisupervised learning framework, but are developed for protein function prediction using a single data source (PPI) only.

2.2 Data Integration in Protein Function Prediction

Several methods have been developed to integrate the information from heterogeneous data sources (i.e., PPI, protein sequences) to boost the function prediction accuracy [4]. Lanckriet et al. [5] defined a kernel on each data source, and then utilized semidefinite programming (SDP) to seek the optimal weights to linearly combine these kernels. However, this method is computationally expensive [10]. To overcome this problem, Tsuda et al. [10] made use of the dual problem and gradient descent to efficiently get the optimal weights. Shin et al. [13] sought the optimal weights within an EM [32] framework by iteratively minimizing prediction error and combining weights. Mostafavi et al. [33] propose a heuristic approach derived from ridge regression, to more efficiently determine these weights. Finally, these obtained composite kernels are used in SVMs or graph-based classifiers for binary protein function annotation. These methods determine the set of weights per function class, that not only result in increased time complexity, but also ignore the inherent correlation among function labels.

More recently, some approaches have leveraged kernel integration and function correlation. Mostafavi and Morris [9] introduced a method called "simultaneous weighting" (SW). The SW optimizes a set of weights for a group of correlated functions, and then combines these kernels into a composite kernel. Next, graph-based semisupervised classifiers are trained on this composite kernel for each function. Tang et al. [19] introduced a unified framework, in which selecting a specific composite kernel for each function and for all the functions are two extreme cases. In our experiments, we consider this approach for comparison, and call it MKL-Sum [19]. MKL-Sum first learns linear combination coefficients with respect to all the functions, and then applies SVMs on the composite kernel. Bucak et al. [20] utilized stochastic approximation to speed up multilabel multiple kernel learning, and proposed a method called MKL-SA. Cesa-Bianchi and Alpaydin [34] integrated binary classifiers hierarchical ensembles, cost-sensitive methods, and data fusion for gene function prediction. However, despite various optimization techniques, Tsuda et al. [10], Lewis [6], and Gönen et al. [35] observed that a composite kernel combined with optimized weights has similar performance to a composite kernel combined with equal weights, i.e., without optimization.

In this work, different from traditional kernel integration methods [5], [9], [19], we propose a classifier integration method called the TMEC. The TMEC first trains a transductive multilabel classifier (TMC) on each of the kernels representing a data source, and then integrates the predictions using majority voting. Our experimental results on three public available protein function prediction benchmarks show that the TMEC outperforms classifiers trained on the composite kernels. In addition, we observe that the TMEC outperforms the TMC trained on the composite kernel and the individual kernels derived from different data sources. This observation confirms that the ensemble approach is effective, and the transductive multilabel classifiers trained on individual kernels are complementary to each other.

3 Problem Formulation

We are given R different kinds of features that describe the same set of N proteins with C functions. Each kind of features provide a unique representation for proteins (e.g., vectors, trees, or networks). We assume the first l proteins are already annotated and the remaining u proteins are not annotated ($l + u = N$). The R different representations of these proteins are transformed into R kernels $[K_r]_{r=1}^{R}(K_r \in \mathbb{R}^{N\times N})$, one kernel per source. $K_r(i,j) \geq 0$ describes the kernel induced pairwise similarity between proteins i and j in the rth data source. Our objective is to first train a TMC on a directed birelational graph adapted from the kernel K_r, and then combine these classifiers into an ensemble classifier (TMEC). Finally, we use the TMEC to annotate these u proteins. In this section, we first review the birelational graph approach, analyze the related issues for network propagation, and give a toy example to illustrate this problem.

3.1 Transductive Multilabel Classification on a Directed Birelational Graph

Graph-based transductive or semisupervised learning methods can be extended to multilabel learning by incorporating a label correlation term into its objective function [16], [18], [36]. Wang et al. [22] introduced an undirected birelational graph for image classification and applied a random-walk-based propagation with restart [37]. This graph includes both images and labels as nodes. For consistency, hereinafter, we use proteins instead of images and functions instead of labels. A birelational graph is composed of three kinds of edges: between proteins, between functions, and between proteins and functions. For the latter, if protein i has function c, an edge is set between them.

The interfunction similarity leads to improved prediction accuracy [17], [18] and can be defined in various ways [15], [17], [18]. Here, we define the similarity between functions m and n as follows:

$$S_{FF}(m,n) = \frac{\mathbf{f}_m^{\mathrm{T}}\mathbf{f}_n}{\|\mathbf{f}_m\|\|\mathbf{f}_n\|}, \tag{1}$$

where $\mathbf{f}_m \in \mathbb{R}^N (1 \leq m \leq C)$ is the mth function vector on all proteins: if protein i has function m, then $\mathbf{f}_m(i) = 1$, otherwise $\mathbf{f}_m(i) = 0$.

A random walk on a graph is often described by a propagation matrix. For a random walk on a birelational graph, the propagation matrix W is defined as

$$W = \begin{bmatrix} \beta W_{PP} & (1-\beta)W_{PF} \\ (1-\beta)W_{FP} & \beta W_{FF} \end{bmatrix}, \tag{2}$$

where $W_{PP} \in \mathbb{R}^{N\times N}$ and $W_{FF} \in \mathbb{R}^{C\times C}$ are the propagation matrices of the intra-subgraphs of proteins and functions, respectively. $W_{PF} \in \mathbb{R}^{N\times C}$ and $W_{FP} \in \mathbb{R}^{C\times N}$ are the inter-subgraph propagation matrices between proteins and functions, and β controls the relative importance of the intra- and the inter- subgraphs. It also controls the frequency with which a random walker jumps from a function subgraph to a protein subgraph. W_{PP} and W_{FF} are computed as

$$W_{PP} = D_{PP}^{-1}S_{PP} \quad W_{FF} = D_{FF}^{-1}S_{FF}, \tag{3}$$

where S_{PP} is the pairwise similarity matrix between all proteins, and S_{FF} is the pairwise correlation matrix between all functions. D_{PP} and D_{FF} are the diagonal matrices of the row sums of S_{PP} and S_{FF}, respectively. W_{PF} and W_{FP} are calculated as

$$W_{PF} = D_{PF}^{-\frac{1}{2}}S_{PF}D_{FP}^{-\frac{1}{2}} \quad W_{FP} = D_{FP}^{-\frac{1}{2}}S_{FP}D_{PF}^{-\frac{1}{2}}, \tag{4}$$

where S_{PF} is the relation matrix between proteins and functions, and S_{FP} is the transpose of S_{PF}. D_{PF} is the diagonal matrix of the row sums of S_{PF}, and D_{FP} is the diagonal matrix of the column sums of S_{PF}. We observe that if protein i has function c then $S_{PF}(i,c) = 1$; otherwise $S_{PF}(i,c) = 0$.

The cth function node and the proteins annotated with this function are considered as a group,

$$G_c = v_c^F \cup \{v_i^P \mid S_{PF}(i,c) = 1\}, \tag{5}$$

where v_c^F is the cth function node and v_i^P is the ith protein node of the birelational graph. In the birelational graph, instead of computing the node-to-node relevance between a function node and an unannotated protein node, the relevance between a protein and a group G_c is considered. Let $\tilde{Y} \in \mathbb{R}^{(N+C)\times C}$ be the label distribution on the $N + C$ nodes of the birelational graph with respect to C function labels. Each column corresponds to one function label. For the cth function, the distribution vector $\tilde{Y}_{.c}$ (cth column of $\tilde{Y}$) is

$$\tilde{Y}_{.c} = \begin{bmatrix} \gamma\tilde{Y}_{.c}^P \\ (1-\gamma)\tilde{Y}_{.c}^F \end{bmatrix} \in \mathbb{R}^{N+C}, \tag{6}$$

where $\tilde{Y}_{.c}^P \in \mathbb{R}^N$ is the distribution vector on the protein nodes, and $\tilde{Y}_{.c}^F \in \mathbb{R}^C$ is the distribution vector on the function nodes. $\tilde{Y}_{.c}^P(i) = 1/\sum_{i=1}^{N} S_{PF}(i,c)$ if $S_{PF}(i,c) = 1$, and $\tilde{Y}_{.c}^P(i) = 0$ otherwise; $\tilde{Y}_{.c}^F(j) = 1$ if $j = c$, and $\tilde{Y}_{.c}^F(j) = 0$ otherwise. γ adjusts the distribution of function labels on protein and function nodes.

Based on these preliminaries, an iterative objective function is defined on this birelational graph as follows:

$$F^{(t+1)}(j) = (1-\alpha)\sum_{i=1}^{N+C} W(i,j)F^{(t)}(i) + \alpha\tilde{Y}_j, \tag{7}$$

where $F^{(t)}(i) \in \mathbb{R}^C$ is the predicted likelihood of the ith protein with respect to C function labels in the tth iteration, $W(i,j)$ is the weight of edge between nodes i and j, $\tilde{Y}_j \in \mathbb{R}^C$ is the initial set of functions on the jth node, α is a scalar value to balance the tradeoff between the initial set functions and the predicted functions. From (7), we can see that the functions of a node are predicted by the functions of its connected nodes. This makes TMC, a direct protein function prediction method [2]. Note that, in the birelational graph, the predictions are made on all the nodes (including proteins and functions).

However, the application of (7) for protein function prediction has a major drawback. Suppose i is a protein vertex annotated with a function vertex j. The function j may be overwritten by the functions of the proteins connected to i, thus causing the loss of a reliable information. As such, the functions of initially annotated proteins may be changed during the iterative label propagation. This phenomenon is similar to the one occurring in the local and global consistency method [30], and should be avoided. A toy example in the following section will illustrate this problem.

Equation (7) can be rewritten as follows (for simplicity, the parameter β in (2) is not included):

$$\begin{aligned}\begin{bmatrix} F_P^{(t+1)} \\ F_F^{(t+1)} \end{bmatrix} &= (1-\alpha)\begin{bmatrix} W_{PP} & W_{PF} \\ W_{FP} & W_{FF} \end{bmatrix}\begin{bmatrix} F_P^{(t)} \\ F_F^{(t)} \end{bmatrix} + \alpha\begin{bmatrix} \tilde{Y}_P \\ \tilde{Y}_F \end{bmatrix}, \\ &= (1-\alpha)\begin{bmatrix} W_{PP}F_P^{(t)} + W_{PF}F_F^{(t)} \\ W_{FP}F_P^{(t)} + W_{FF}F_F^{(t)} \end{bmatrix} + \alpha\begin{bmatrix} \tilde{Y}_P \\ \tilde{Y}_F \end{bmatrix},\end{aligned}$$

$$F_P^{(t+1)} = (1-\alpha)\big(W_{PP}F_P^{(t)} + W_{PF}F_F^{(t)}\big) + \alpha\tilde{Y}_P, \tag{8}$$

$$F_F^{(t+1)} = (1-\alpha)\big(W_{FP}F_P^{(t)} + W_{FF}F_F^{(t)}\big) + \alpha\tilde{Y}_F. \tag{9}$$

From (8-9), we can see that W_{PF} propagates the function information from function to protein nodes, and W_{FP} propagates the function information from protein to function nodes. In a birelational graph, we expect that information propagates from function to protein nodes, but not vice versa. Thus, we change the undirected birelational graph into a directed one. An example of the proposed directed birelational graph is shown in Fig. 1. In this graph, information can be propagated in the intra-subgraphs W_{PP} and W_{FF}, and in the inter-subgraph W_{PF}, but not in the inter-subgraph W_{FP}. Therefore, we define the propagation matrix W_d on the directed birelational graph as follows:

$$W_d = \begin{bmatrix} W_{PP} & W_{PF} \\ \mathbf{0} & W_{FF} \end{bmatrix}, \tag{10}$$

where $\mathbf{0} \in \mathbb{R}^{C\times N}$. The TMC takes advantage of network propagation by optimizing local and global consistency functions [30]. By defining directed edges between proteins and functions, the induced propagation matrix of the directed birelational graph becomes nonsymmetric. The directed edges control the information that flows from one side of the bipartite graph (function subgraph) to the other (protein subgraph). As described earlier in (8-9), and through the empirical study in Section 3.2, we show that the TMC trained on a directed birelational graph can avoid the observed problems (i.e., annotation change and function label override) associated with the TMC trained on an undirected birelational graph. Thus, we advocate the use of a directed birelational graph, and define a nonsymmetric propagation matrix on this graph.

Based on (7), we can get the iterative equation on the directed birelational graph

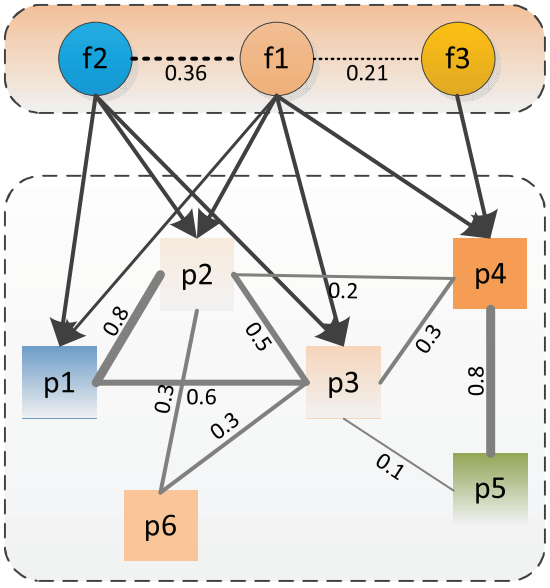


Fig. 1. An example of a directed birelational graph. The circles are function nodes, the rectangles are protein nodes, the directed edges represent function assignment, the undirected edges with weights in solid line represent protein-protein interactions, and the dotted lines with weights represent function correlations. The width of the line reflects the weight of the edge.

$$F^{(t+1)} = (1-\alpha)W_d F^{(t)} + \alpha\tilde{Y}. \tag{11}$$

By setting $F^{(0)} = \tilde{Y}$, we have

$$F^{(t+1)} = ((1-\alpha)W_d)^{(t+1)}\tilde{Y} + \alpha\sum_{k=0}^{t}((1-\alpha)W_d)^k\tilde{Y}. \tag{12}$$

Since $0 < \alpha < 1$ and $\mathbf{0} \leq (1-\alpha)W_d < \mathbf{1}$. Note that when $k = 0$, $((1-\alpha)W_d)^k$ is the identity matrix. The first term in (12) is bound to be $\mathbf{0}$, and the second term (excluding $\tilde{Y}$) is a geometric series with the following limit:

$$\lim_{t\to\infty}\sum_{k=0}^{t}((1-\alpha)W_d)^k = (I-(1-\alpha)W_d)^{-1}, \tag{13}$$

where I is an $(N+C)\times(N+C)$ identity matrix. Thus, the equilibrium solution F of (11) is

$$F = \alpha(I-(1-\alpha)W_d)^{-1}\tilde{Y}. \tag{14}$$

The predicted $F(j)$ is a real-value vector with size C, where each entry reflects the likelihood that protein j has the corresponding function. Thus, we also refer to $F(j)$ as the predicted likelihood score vector of protein j. From (14), we can see that F is determined by W_d and a well-structured birelational graph can produce a competent F.

$F_P^{(t)}$ is initially set according to the original function annotation on the l annotated proteins. For both the directed and undirected birelational graphs, $F_P^{(t)}$ is updated in iterations, but $F_P^{(t)}$ on the directed birelational graph avoids the problems (i.e., annotation change and function label override; see Section 3.2) associated with the undirected birelational graph, and achieves better prediction. In the undirected birelational graph, W_{FP} propagates label information from protein nodes to function nodes, but in the directed birelational graph, no information is propagated from the protein nodes to function nodes. Thus, the term $W_{FP}F_P^{(t)}$ may lead to poor performance due to W_{FP}, while the term $F_P^{(t)}$ alone does not have this problem.

TABLE 1
The Prediction of the TMC on the *undirected* and *directed* Birelational Graph for the Protein Nodes and Function Nodes, and the Original Function Assignment

Nodes	original			undirected			directed		
	f1	f2	f3	f1	f2	f3	f1	f2	f3
p1	1	1	0	1	1	0	1	1	0
p2	1	1	0	1	1	0	1	1	0
p3	1	1	0	1	1	0	1	1	0
p4	1	0	1	1	***1***	0	1	0	1
p5	0	0	0	1	***1***	0	1	0	1
p6	0	0	0	1	1	0	1	1	0
f1	1	0	0	1	0	0	1	0	0
f2	0	1	0	0	1	0	0	1	0
f3	0	0	1	***1***	0	0	0	0	1

3.2 The Problem of Undirected Birelational Graph in Label Propagation

We provide an illustrative example to instantiate the problem of the *undirected* birelational graph in label propagation, and therefore motivate the use of a *directed* birelational graph.

The toy example is illustrated in Fig. 1. In this directed birelational graph, there are three function nodes ($f1$, $f2$, and $f3$) and six protein nodes ($p1,\ldots,p6$). The first four proteins are labeled (as specified by the directed solid lines), and the last two proteins are unlabeled. In the graph, $p5$ is more similar to (or interacted with) $p4$ than $p3$. From the "guilty by association" rule [38] (interacting proteins tend to share similar functions), $p5$ is more likely to have the function set of $p4$. However, the TMC on the undirected birelational graph predicts $p5$ to have the same function set as $p3$, instead of $p4$, as illustrated in the fifth row of Table 1. In contrast, the TMC on the directed birelational graph predicts $p5$ to have the same function set as $p4$.

Another observation is that for the undirected graph, $p4$ is initially annotated with $f1$ and $f3$, but after label propagation, this protein is annotated with $f1$ and $f2$ (see the fourth row of Table 1). In addition, after the label propagation on the undirected birelational graph, the information on the $f3$ node is dominated by the information of $f1$ (see the last row of Table 1), resulting in the function override problem, which was pointed out and analyzed in the previous section. In contrast, by using the directed birelational graph, these problems are avoided. The advantage of the directed birelational graph with respect to the undirected one will be further confirmed in our experiments on real-world benchmarks.

Note, for the experiment in Fig. 1 and Table 1, we compute the function similarities using (1). Since there is no prior information on how to balance the importance of the protein and function nodes, we use the default settings of $\beta = 0.5$ and $\gamma = 0.5$ as in the prior work [15]. For the experiment here, we cannot select the best β and γ by grid search, since there is very scarce training data. In addition, setting higher values of β and γ lead to the observed problems of undirected birelational graphs. We convert the predicted likelihoods F into binary labels using the Top k

scheme [18], [21], [39]. For each protein, the k largest predicted probabilities are chosen as relevant functions and labeled as 1s, and the others are set as irrelevant functions and labeled as 0s. In the toy example, since each training protein has two functions, we set $k = 2$. The other parameter settings for the TMC will be detailed in Section 4.

3.3 Transductive Multilabel Ensemble Classification

The TMC avoids the risk of overwriting the information given by function nodes. However, because of noisy edges (i.e., false positive interactions) and isolated proteins present in a single birelational graph, it is still limited in providing a confident likelihood score vector $F(j)$ from a single data source. To avoid this limitation, we can leverage the various graphs (or kernels) associated to the same set of proteins (e.g., PPI network, gene interaction network, and coparticipation network in a protein complex) [4], [13]. These graphs are, to some extent, independent to one another, and also carry complementary information.

Here, we predict protein functions using multiple kernels derived from multiple sources by performing *classifier integration*. More specifically, we first transform each kernel into a directed birelational graph. We then train a TMC on each of these graphs. Finally, we combine these TMCs into the TMEC using an ensemble approach. The TMEC is described in Algorithm 1. In (15), we combine the F_r values using a weighted majority vote. This is due to the fact that different kernels have different qualities (see Figs. 2 and 3), and have different levels of confidence on the predicted functions of a protein. For example, if kernel K_1 is more confident on annotating protein i with function m, and K_2 is more confident on annotating protein i with function n, then K_1 will have more influence on determining the mth function of protein i, and K_2 will have more influence on determining the nth function of protein i. On the other hand, if unlabeled protein i in K_1 is isolated (TMC on K_1 cannot predict protein i), but it is connected with some proteins in K_2, then the functions of protein i can be predicted by the TMC on K_2.

Algorithm 1. TMEC: Transductive Multilabel Ensemble Classification.

Input:

$\{K_r\}_{r=1}^{R}$ from R data sources
$Y = [\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_l]$
α, β, γ

Output:

Predicted likelihood score vectors $\{F(j)\}_{j=l+1}^{N}$

1: Specify $\widetilde{Y}$ using (6)
2: **for** $r = 1$ to R **do**
3: Set $W_{PP} = K_r$ and construct a directed birelational graph using (3-4) and (10)
4: Get the rth annotator F_r using (14)
5: **end for**
6: Integrate the R annotators $\{F_r\}_{r=1}^{R}$ into an ensemble classifier as:

$$F(j) = \frac{1}{R}\sum_{r=1}^{R} F_r(j) \tag{15}$$

Due to the different structures across different kernels, the base classifiers F_r in (15) are diverse. In addition, because of the complementary information between different kernels, the predicted likelihoods $F_r(j)$ are also complementary to each other. In contrast, the annotator trained on the composite graph cannot make use of these predicted likelihoods. In ensemble learning, diversity between base classifiers is paramount to gain a consensus classifier with a good generalization ability [40]. For these reasons, the TMEC can annotate proteins with higher confidence than approaches trained on the composite kernel and single kernels. Our experimental results in Section 5 confirm this advantage. An additional merit of the TMEC is that it does not require to have all the data sources available beforehand, and each TMC can be trained individually or incrementally. Thus, new data sources can be appended into the TMEC without repeating the entire training process.

TABLE 2
Protein Function Prediction Benchmarks Statistics ($\mathrm{Avg} \pm \mathrm{std}$ Means Average Number of Functions for Each Protein and Its Standard Deviation)

Dataset	#Kernels	#Proteins	#Functions	Avg±Std
Yeast	44	1809	57	4.35 ± 3.28
Human	8	3704	254	3.73 ± 3.67
Fly	38	3509	426	6.53 ± 7.47

4 Experimental Setup

4.1 Data Set Description

We evaluate the performance of our algorithms on three previously defined protein function prediction benchmarks, namely **Yeast**, **Human** and **Fly**. All these benchmarks were downloaded from the study by Mostafavi and Morris [9][4] and annotated according to the biological process function categories in the GO database [1]. The Yeast benchmark includes 3,904 proteins annotated with 1,188 functions. There are 44 different kernel functions in Yeast data set, most of which are protein-protein interactions obtained from different experiments. The Human benchmark includes 13,281 proteins annotated with 1,952 GO functions. There are eight different kernel functions in the Human data set, which are derived from various domains, i.e., protein-protein interactions, tissue expression, protein-DNA/RNA-interaction. The Fly benchmark includes 13,562 proteins annotated with 2,195 GO functions. There are 38 different kernel functions in the Fly data set, including gene expression, protein-protein interactions, and Pfam.

To avoid too general or too small functions, as done in previous studies [9], [18], we filtered the proteins in Yeast to include only those GO functions that had at least 100 proteins and at most 300 proteins. After this preprocessing, there are 1,809 proteins annotated with 57 functions in Yeast. We filtered the proteins in Human and Fly to include only those GO functions that had at least 30 proteins and at most 100 proteins. Thus, there are 3,704 proteins annotated with 254 functions in Human, and 3,509 proteins annotated with 426 functions in Fly. The statistics of these filtered benchmarks are listed in the Table 2.

4. http://morrislab.med.utoronto.ca/~sara/SW/.

4.2 Evaluation Metrics

We evaluate the protein function prediction problem as a multilabel classification problem. There are various evaluation metrics in multilabel learning [24], here we adopt three evaluation metrics, namely, *Ranking Loss*, *Coverage*, and adapted *Area Under the Curve* (AUC). Given C different protein functions, our approach results in a predicted likelihood vector that assigns a protein i to a function c with probability $F(i,c)$. There is no standard rule to transform the predicted likelihood vector into a binary indication label vector [24], [41]. Thus, we do not apply the evaluation metrics that depend on transforming the predicted likelihood vector into a binary label vector. For brevity, the definition of *Ranking Loss* and *Coverage* is introduced in the online supplemental material.

The adapted *Area Under the Curve* (AUC) for multilabel learning was utilized in [42]. AUC first ranks all the labels for each test instance in the descending order of their scores; it then varies the number of predicted labels from 1 to the total number of labels, and computes the receiver operating characteristic curve by calculating the true positive rate and the false positive rate for each number of predicted labels. It finally, computes the area under the curve of all labels to evaluate the multilabel learning methods.

For maintaining consistency with *AUC*, we use *1-RankingLoss*. Thus, the higher the value of *1-RankingLoss* and *AUC*, the better the performance. In contrast, the lower the value of *Coverage*, the better the performance. The reported *1-RankingLoss* and *AUC* are actually *(1-RankingLoss)*100* and *AUC*100*, respectively.

5 Experiment Analysis

We evaluate the TMC and TMEC by comparing them against PfunBG [15], GRF [18], SW [9], MKL-Sum [19], and MKL-SA [20]. PfunBG and GRF are recently proposed semisupervised multilabel classifiers based on PPI networks for protein function prediction. SW is a recently introduced efficient protein function prediction method based on the composite kernel. MKL-Sum and MKL-SA are two multilabel multiple kernel learning approaches, whose performance is verified in gene function prediction. We adopted the codes of SW,[5] MKL-SA,[6] and MKL-Sum[7] for our experiments. We implemented PfunBG and GRF as the authors presented in the original papers, and set the parameter values as the authors reported. TMC and PfunBG have a similar objective function. The main difference between them is that TMC is trained on the *directed* birelational graph and PfunBG is trained on the *undirected* birelational graph. Here, we use the similar settings of undirected birelational graph in [22] and [15], and set α in (7) equal to 0.01, and β and γ in the birelational graph equal to 0.5. We set equal weights for protein nodes and function nodes, since there is no prior knowledge on the relative importance between protein and function nodes. Both MKL-Sum and MKL-SA depend on the setting of soft margin parameter η of SVM; in the experiments, η is chosen among $\{10^{-4}, 10^{-3}, \ldots, 10^{3}, 10^{4}\}$ using fivefold cross validation.

TMC, PfunBG, and GRF work on a single graph (or kernel). As the authors reported in [6] and [10], the composite kernel combined with different weights has similar performance to the composite kernel combined with equal weights. Therefore, in the following experiments, TMC, PfunBG, and GRF are all trained on the same composite kernel combined with individual kernels using equal weights. We also study TMC on the composite kernels combined with individual kernels in other ways, which will be detailed in Section 5.3. In addition, we compare the TMEC with SW, MKL-Sum and MKL-SA, which learn different weights for different kernels. In Section 5.2, we investigate the performance of TMC on each single kernel. In the following experiments, we varied the ratio of labeled proteins from 10 percent to 60 percent, and used the remaining proteins as unlabeled for testing. Unless otherwise specified, all the results are the average of 20 independent runs in each fixed labeled ratio.

5.1 Directed Birelational Graph versus Undirected Birelational Graph

To investigate the difference between directed and undirected birelational graphs, we compare our TMC against PfunBG, GRF, SW, MKL-Sum, and MKL-SA on the composite kernel of Yeast, Human, and Fly. The composite kernel for TMC, PfunBG, and GRF is a linear combination with equal weights, and the composite kernel for SW, MKL-Sum, and MKL-SA is a linear combination with optimized weights. Table 3 reports the *1-RankingLoss*, Table 4 lists the *Coverage*, and Table 5 reports the *AUC* on the three benchmarks. Standard deviations are also reported. In these tables, results reported in boldface are significantly better, with significance level $p < 0.05$. Particularly, for each fixed labeled ratio, we repeat independent experiments for each method 20 times and record the results. Then, we apply a pairwise t-test to check the significance of the difference among these comparative methods.

From these tables (see Tables 3, 4, and 5), we can observe that TMC trained on the directed birelational graph almost always performs better than PfunBG trained on the undirected birelational graph. These results corroborate the advantage of the directed birelational graph with respect to the undirected one. GRF extends the function assignment on the annotated proteins and then makes use of a semisupervised classifier on the graph (only consisting of proteins, one protein corresponding to one node) to infer protein functions. From these (see Tables 3, 4, and 5), we can also see that TMC often outperforms GRF. This observation indicates that the directed birelational graph (consisting of protein and function nodes) is also more effective than the sole protein graph.

TMC often outperforms the other comparing methods, some of which (i.e., SW, MKL-Sum, and MKL-SA) are trained on the optimized composite kernel. SW takes advantage of regression to seek the optimal combining weights, and graph-based classification to predict protein functions; it achieves higher *AUC* on Human benchmark, but it loses to TMC on Yeast and Fly benchmarks with respect to all the evaluation metrics. Both MKL-Sum and

5. http://morrislab.med.utoronto.ca/~sara/SW/.
6. http://www.cse.msu.edu/~bucakser/ML-MKL-SA.rar.
7. http://www.public.asu.edu/~ltang9/code/mkl-multiple-label/.

TABLE 3
1-RankingLoss ($\mathrm{Avg} \pm \mathrm{std}$) on **Yeast** (**44 Kernels**), **Human** (**8 Kernels**), and **Fly** (**38 Kernels**) on the Composite Kernel

DataSet	Method	Labeled Ratio					
		10%	20%	30%	40%	50%	60%
Yeast	SW	50.63±2.17	55.58±2.37	60.36±3.38	64.74±1.75	67.41±2.42	69.90±1.95
	MKL-Sum	53.23±3.01	54.97±3.54	54.46±4.09	58.53±2.63	61.31±9.76	67.83±2.16
	MKL-SA	62.34±0.77	65.40±0.95	67.87±0.88	69.14±0.64	70.61±0.50	71.42±0.83
	GRF	48.94±1.48	54.94±1.62	60.68±1.98	66.12±1.18	69.08±1.36	71.51±1.52
	PfunBG	65.17±1.56	67.65±1.42	68.46±1.28	69.84±1.00	68.97±1.36	67.79±1.83
	TMC	**68.69±1.24**	**72.58±1.07**	**74.78±0.78**	**76.98±0.66**	**77.83±0.75**	**78.60±0.82**
Human	SW	53.11±1.10	63.24±1.68	69.29±0.94	73.99±0.73	**76.75±0.48**	**78.49±0.58**
	MKL-Sum	52.11±5.58	58.77±7.69	53.46±5.73	47.2±10.7	46.19±7.71	56.77±10.89
	MKL-SA	52.30±1.93	55.55±0.88	55.57±0.39	56.79±0.15	57.13±0.47	56.40±0.44
	GRF	55.56±0.90	63.21±1.27	66.95±0.90	69.96±0.64	71.40±0.62	72.31±0.42
	PfunBG	64.60±0.86	68.25±0.82	69.87±0.74	71.48±0.53	71.93±0.70	72.15±0.50
	TMC	**67.31±0.76**	**71.55±0.70**	**73.43±0.56**	**75.03±0.37**	75.79±0.44	76.25±0.49
Fly	SW	43.81±0.78	52.92±0.86	61.47±0.97	67.07±1.11	70.37±0.87	72.99±0.66
	MKL-Sum	40.49±10.14	58.57±1.95	63.66±1.19	65.08±1.49	65.97±0.58	66.54±1.23
	MKL-SA	63.41±1.13	65.37±0.26	66.78±0.70	68.16±0.55	69.13±0.31	69.80±1.14
	GRF	44.59±0.77	54.21±0.74	61.54±0.72	66.90±1.11	70.17±0.91	73.08±0.73
	PfunBG	61.76±1.30	69.22±0.64	72.20±0.45	74.10±0.79	74.98±0.69	75.67±0.57
	TMC	**64.92±1.41**	**73.10±0.48**	**76.39±0.43**	**78.43±0.49**	**79.74±0.56**	**80.69±0.37**

The numbers in boldface denote the best performance (significance is examined by pairwise t-test with significance level $p < 0.05$).

MKL-SA are based on the optimized composite kernel and SVM to predict protein functions, but they are always outperformed by TMC. These results indicate that our equal combination of single kernels (derived from various data sources) is reasonable. The possible explanations are that the composite kernel used in TMC captures complementary information spread in different kernels [6] and multilabel classification is more suitable for protein function prediction than binary classification methods.

5.2 Multiple Kernels versus Single Kernel

In this section, we conduct experiments on these benchmarks to investigate the advantage of classifier ensembles. We compare the TMEC, GRF-MK, and PfunBG-MK against SW and TMC. Among the five comparing approaches, the first three are classifier integration methods, and the last two are kernel integration methods. In the experiments, the base classifiers of PfunBG-MK, and GRF-MK on the individual kernels are combined in the same way as TMEC in (15). For brevity, we just report *1-RankingLoss* and *Coverage* on these three data sets with 30 and 60 percent annotated proteins in Tables 6 and 7. The results with respect to other labeled ratios are similar, and are excluded for brevity.

We can observe that the TMEC on the multiple kernels often outperforms the TMC on the composite kernel, and the TMEC always performs better than the SW. GRF-MK and PfunBG-MK sometimes also outperform TMC, whereas TMC outperforms GRF and PfunBG on the composite

TABLE 4
Coverage ($\mathrm{Avg} \pm \mathrm{std}$) on **Yeast** (**44 Kernels**), **Human** (**8 Kernels**), and **Fly** (**38 Kernels**) on the Composite Kernel

DataSet	Method	Labeled Ratio					
		10%	20%	30%	40%	50%	60%
Yeast	SW	37.43±1.14	35.38±1.06	33.06±1.59	30.43±0.86	29.18±1.13	27.40±1.06
	MKL-Sum	34.46±1.26	34.60±1.41	34.54±2.61	32.64±1.30	30.61±5.08	27.08±1.46
	MKL-SA	29.80±0.43	28.42±0.62	26.64±0.60	26.09±0.34	25.18±0.53	25.21±0.78
	GRF	36.69±0.95	33.45±0.77	30.49±0.85	27.17±0.67	25.37±0.78	23.82±0.79
	PfunBG	27.45±1.10	26.02±0.83	25.67±0.67	24.91±0.68	25.59±0.77	26.25±0.84
	TMC	**25.05±0.89**	**22.61±0.67**	**21.20±0.52**	**19.80±0.47**	**19.37±0.44**	**18.81±0.57**
Human	SW	150.52±2.75	127.37±3.97	111.05±2.37	98.65±1.73	89.83±1.65	83.76±2.12
	MKL-Sum	151.7±13.2	135.5±17.9	149.1±15.8	165.5±28.0	168.1±19.7	141.0±26.8
	MKL-SA	147.51±6.24	140.41±2.20	142.09±1.29	139.49±0.92	139.97±2.22	143.25±1.80
	GRF	137.61±2.32	116.65±3.36	105.76±2.43	97.36±1.99	92.50±1.81	89.35±1.53
	PfunBG	111.74±2.40	101.76±2.29	97.48±2.03	93.87±1.64	92.74±2.02	92.50±1.72
	TMC	**105.48±2.22**	**93.32±2.07**	**87.50±1.58**	**83.20±1.35**	**80.48±1.47**	**78.73±1.64**
Fly	SW	307.65±2.82	278.59±2.64	246.06±3.21	222.72±4.92	206.64±3.81	193.92±3.14
	MKL-Sum	276.89±18.72	251.72±9.76	231.04±5.57	227.34±7.51	223.84±3.25	221.19±5.67
	MKL-SA	228.59±6.13	220.05±1.41	214.54±4.29	209.37±2.42	203.61±1.88	201.02±5.94
	GRF	293.58±2.95	254.63±3.26	223.75±2.60	200.58±5.02	184.23±3.26	171.40±3.42
	PfunBG	223.49±5.54	189.68±3.21	176.24±1.93	168.82±3.64	164.84±2.66	163.99±3.18
	TMC	**211.78±6.49**	**173.48±2.86**	**156.77±2.34**	**147.41±2.87**	**140.09±2.83**	**135.80±2.75**

The numbers in boldface denote the best performance (significance is examined by pairwise t-test with significance level $p < 0.05$). The lower the Coverage, the better the performance.

TABLE 5
AUC ($\mathrm{Avg} \pm \mathrm{std}$) on **Yeast** (**44 Kernels**), **Human** (**8 Kernels**), and **Fly** (**38 Kernels**) on the Composite Kernel

DataSet	Method	Labeled Ratio					
		10%	20%	30%	40%	50%	60%
Yeast	SW	54.72±0.65	59.22±1.28	63.08±1.53	65.38±1.58	67.63±1.14	71.18±0.86
	MKL-Sum	54.46±2.11	57.02±2.68	55.80±3.56	58.46±2.49	60.38±8.27	66.38±2.37
	MKL-SA	62.71±0.91	64.74±0.60	66.54±0.76	67.80±0.81	69.22±0.55	70.11±1.29
	GRF	56.29±1.48	62.67±1.19	68.52±1.48	70.88±1.45	73.41±0.70	75.32±1.08
	PfunBG	68.26±1.05	71.08±0.93	71.51±1.39	71.90±0.60	72.11±0.53	72.11±0.77
	TMC	**69.02±0.81**	**72.94±0.93**	**74.23±1.00**	**75.59±0.41**	**76.53±0.66**	**77.68±0.69**
Human	SW	60.73±1.06	67.91±0.70	72.51±0.37	75.15±0.74	**77.35±0.51**	**78.70±0.79**
	MKL-Sum	51.07±6.83	57.58±8.24	51.71±4.80	47.33±9.57	46.74±7.59	55.6±10.4
	MKL-SA	53.03±1.03	54.98±0.52	55.33±0.33	55.68±0.48	55.54±0.64	55.27±0.55
	GRF	63.50±1.19	68.38±0.70	70.95±0.28	72.69±0.56	73.79±0.66	74.76±0.54
	PfunBG	68.48±0.70	70.76±0.56	72.50±0.50	73.15±0.54	73.66±0.56	73.95±0.68
	TMC	**69.52±0.43**	**72.48±0.37**	**73.97±0.38**	74.89±0.38	75.34±0.40	76.36±0.44
Fly	SW	53.39±0.69	58.85±0.85	63.56±0.87	66.77±0.79	68.17±0.75	69.99±0.31
	MKL-Sum	51.75±2.40	55.96±1.24	60.06±0.73	60.94±1.23	61.49±0.70	62.40±0.98
	MKL-SA	59.81±0.66	61.10±0.48	62.02±0.45	63.00±0.26	63.84±0.54	64.68±0.88
	GRF	54.39±0.59	61.44±0.80	65.75±0.66	69.28±0.87	72.10±0.52	73.62±0.66
	PfunBG	65.67±0.67	70.18±0.49	71.94±0.34	72.95±0.40	73.77±0.52	73.66±0.47
	TMC	**66.82±0.52**	**71.61±0.35**	**73.65±0.30**	**74.87±0.38**	**76.08±0.63**	**76.42±0.71**

The numbers in boldface denote the best performance (significance is examined by pairwise t-test with significance level $p < 0.05$).

TABLE 6
1-RankingLoss ($\mathrm{Avg} \pm \mathrm{std}$) on the Composite Kernel and Multiple Kernels of **Yeast** (**44 Kernels**), **Human** (**8 Kernels**), and **Fly** (**38 Kernels**)

	Yeast		Human		Fly	
	30%	60%	30%	60%	30%	60%
SW	60.36±3.38	69.90±1.95	69.29±0.94	78.49±0.58	61.47±0.97	72.99±0.66
TMC	74.78±0.78	78.60±0.82	73.43±0.56	76.25±0.49	76.39±0.43	80.69±0.37
GRF-MK	68.74±1.66	74.12±1.16	76.02±0.71	81.11±0.51	68.13±0.65	75.45±0.58
PfunBG-MK	73.48±0.90	75.70±1.05	77.57±0.52	81.60±0.50	74.67±0.40	78.31±0.42
TMEC	**76.75±0.57**	**80.18±0.61**	**79.30±0.43**	**83.40±0.46**	**77.24±0.41**	**81.16±0.37**

TABLE 7
Coverage ($\mathrm{Avg} \pm \mathrm{std}$) on the Composite Kernel and Multiple Kernels of **Yeast** (**44 Kernels**), **Human** (**8 Kernels**), and **Fly** (**38 Kernels**)

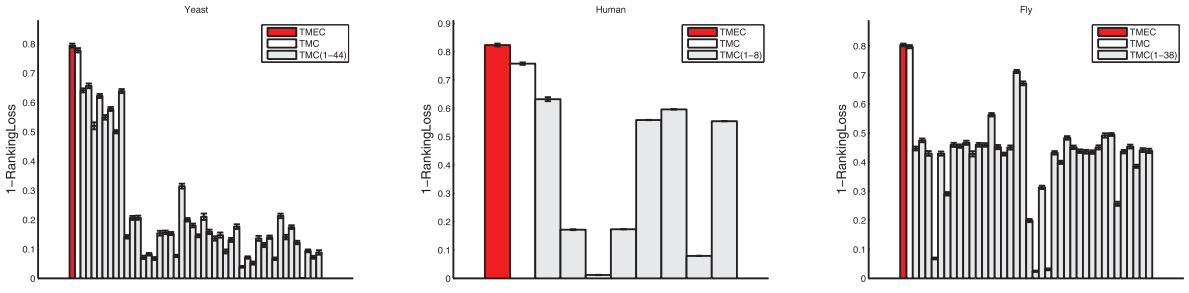
	Yeast		Human		Fly	
	30%	60%	30%	60%	30%	60%
SW	33.06±1.59	27.40±1.06	111.05±2.37	83.76±2.12	246.06±3.21	193.92±3.14
TMC	21.20±0.52	18.81±0.57	87.50±1.58	78.73±1.64	156.77±2.34	135.80±2.75
GRF-MK	25.80±0.83	22.18±0.66	81.39±2.00	66.08±1.85	194.44±2.37	160.68±3.03
PfunBG-MK	22.51±0.53	21.22±0.63	76.73±1.46	65.63±1.80	164.40±1.96	149.65±2.95
TMEC	**19.94±0.36**	**17.72±0.50**	**71.86±1.25**	**59.67±1.72**	**151.63±2.11**	**132.02±2.94**

kernel. This fact demonstrates the advantage of *classifier integration* with respect to the *kernel integration* method in protein function prediction.

We conduct additional experiments (with 80 percent proteins annotated and 20 percent used for testing) to investigate the difference between the TMC on the composite kernel, TMC on a single kernel from one data source, and TMEC on multiple kernels. We show the result with respect to *1-RankingLoss* and *Coverage* in Figs. 2 and 3. In all these figures, the first two bars represent TMEC (red bar) and TMC (white bar) trained on the composite kernel; the remaining bars (gray bars) describe the results of TMC trained on a single kernel. The highest bar (*1-RankingLoss*) in Fig. 2 and the lowest bar (*Coverage*) in Fig. 3 indicate that the corresponding results are significantly better than the other bars.

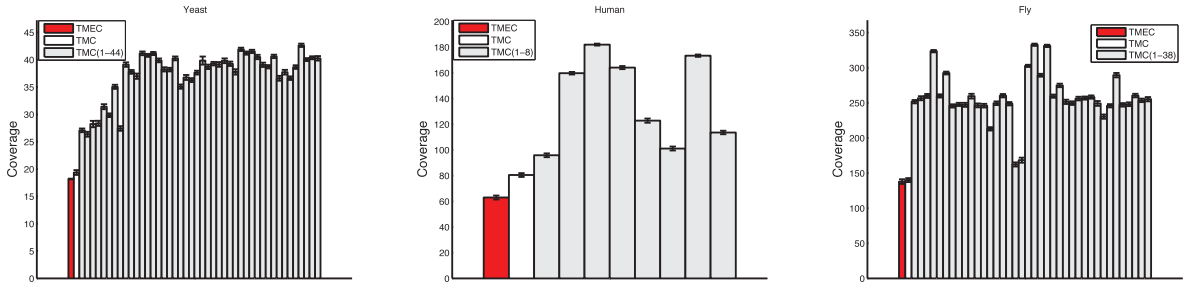
We observe that the TMC trained on the composite kernel always outperforms the TMC trained on a single kernel. There are three possible reasons. First, there are some isolated proteins in each data source, whose functions cannot be predicted using a single kernel derived from a single data source. Second, an isolated protein in one kernel may be connected with other proteins in other data sources. Therefore, there are few (or no) isolated proteins in the composite kernel, which includes the complementary information across different kernels. Third, the similarity between two proteins from the composite kernel is often more reliable than that from a single kernel (derived from a single data source).

TMEC performs better than TMC trained on the single kernel, and it outperforms TMC trained on the composite kernel. TMEC not only takes advantage of the



(a) Yeast (b) Human (c) Fly

Fig. 2. Difference between multiple kernels (red bar), composite kernel (white bar), and single kernels (gray bars) of **Yeast**, **Human**, and **Fly**, with respect to *1-RankingLoss*.



(a) Yeast (b) Human (c) Fly

Fig. 3. Difference between multiple kernels (red bar), composite kernel (white bar), and single kernels (gray bars) of **Yeast**, **Human**, and **Fly**, with respect to *Coverage*.

TABLE 8
1-RankingLoss ($\mathrm{Avg} \pm \mathrm{std}$) on **Yeast** (**44 Kernels**), **Human** (**8 Kernels**), and **Fly** (**38 Kernels**), with Respect to Different Weighting Schemes

	Yeast		Human		Fly	
	20%	50%	20%	50%	20%	50%
TMC-B	69.41±1.01	74.70±0.79	75.08±0.68	81.75±0.56	70.24±0.73	77.85±0.41
TMC-BW	71.98±1.00	77.12±0.80	75.35±0.68	82.10±0.56	72.17±0.67	79.70±0.45
TMC-E	72.58±1.07	78.60±0.75	71.55±0.70	76.25±0.44	73.10±0.48	80.69±0.56
TMEC	**74.71±0.80**	**80.18±0.67**	**76.86±0.53**	**83.40±0.50**	**74.29±0.39**	**81.16±0.55**

complementary information between different kernels, but also makes use of the structural difference among different kernels and the complementary predicted likelihood score vectors. These results also corroborate the advantage of *classifier integration* versus *kernel integration*.

5.3 Influence of Different Weighting Schemes

In this section, we investigate the performance of TMC on different composite kernels, obtained by combining individual kernels using different weighting schemes. Here, we use two approaches to combine the single kernels, plus the equal weight one already used in TMC. The first one is a *binary* way. We set the weight of the edge between protein i and j as

$$K_B(i,j) = \begin{cases} 1, & \text{if } \sum_{r=1}^{R} K_r(i,j) > 0, \\ 0, & \text{otherwise.} \end{cases}$$

Thus, if there is an edge between proteins i and j in any of the individual kernels, we specify the edge weight between them as 1. The second one is a *weighted* version; we set the weight of the edge between proteins i and j as

$$K_{BW}(i,j) = \frac{\sum_{r=1}^{R} \delta(K_r(i,j))}{R},$$

where $\delta(K_r(i,j)) = 1$ if $K_r(i,j) > 0$, and $\delta(K_r(i,j)) = 0$ otherwise. Thus, $K_{BW}(i,j)$ is proportional to the number of edges between proteins i and j in the individual kernels. The composite kernel used in TMC is a summation of single kernels with *equal* weights, it is defined as

$$K_E(i,j) = \frac{\sum_{r=1}^{R} K_r(i,j)}{R}.$$

We then train TMC on K_B, K_{BW}, and K_E, and name the resulting classifiers as TMC-B, TMC-BW, and TMC-E, respectively. For brevity, we just report the results (along with the results of TMEC on multiple kernels) with respect to *1-RankingLoss* in Table 8 with 20 and 50 percent proteins annotated. The results with respect to *Coverage* are reported in the online supplemental material. The results with respect to other labeled ratios have similar observations.

TABLE 9
Performance (*1-RankingLoss*) with Respect to Different Ensemble Techniques

Methods	Yeast	Human	Fly
TMEC	**79.21±0.76**	**82.17±0.45**	**80.58±0.50**
TMEC-DT	61.94±0.40	65.07±0.41	68.12±0.32
TMEC-Reg	61.17±1.00	70.30±1.08	61.50±1.19

From these tables, we observe that none of these three kernel integration methods perform significantly better than others, and TMC-E often works better than the other two kernel integration techniques. Another observation is that TMEC always achieves better performance than the three different kernel integration methods (TMC-B, TMC-BW, and TMC-E). These results further demonstrate the advantage of classifier ensemble with respect to kernel integration.

5.4 Influence of Different Ensemble Techniques

In this section, we conduct experiments to explore the performance of TMEC with respect to several other ensemble techniques, namely decision templates [43] and linear regression based ensemble [44]. The detail of these two ensemble techniques is introduced in the online supplemental material.

We utilize decision templates, linear-regression-based ensemble, and (15) to integrate TMCs trained on individual kernels, and name the resulting ensemble classifiers as TMEC-DT, TMEC-Reg, and TMEC, respectively. For brevity, we just report the results with respect to *1-RankingLoss* in Table 9. We report the results with respect to *Coverage* in the online supplemental material. In the experiment, 50 percent of the data are randomly selected for training and the remaining 50 percent are used for testing. The other settings of the experiments are kept the same as described in Section 4.

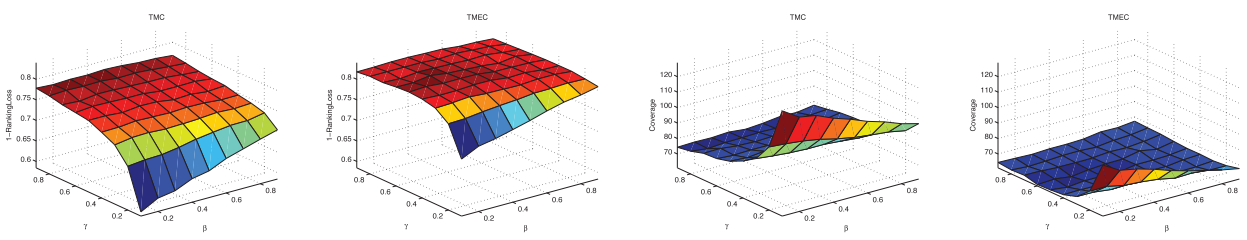
From these tables, we can observe that TMEC, which uses simple weighted majority vote as in (15) to combine base classifiers, outperforms TMEC-DT and TMEC-Reg, which use optimized weights to integrate base classifiers. This is because there were many isolated proteins in each single data source of our experimental data sets. The functions of these isolated proteins cannot be predicted by using a single data source alone. In addition, the predicted likelihoods from each single data source are not reliable. Learning from these unreliable predictions results in incorrect learned weights for each of the base classifiers, which deteriorates the ensemble classification performance.

5.5 Parameter Sensitivity Analysis

There are three parameters: α, β, and γ in TMC and TMEC. α is used to balance the tradeoff between label propagation and initial label assignment; α is often set to a small value (i.e., $\alpha = 0.01$) [15], [22]. In this section, to explore the sensitivity of our methods with respect to β and γ, we vary β and γ between 0.1 and 0.9 with step size 0.1. The settings of $\beta = 0, 1$ and $\gamma = 0, 1$ are not reasonable for the directed birelational graph. If we set $\beta = 0$, there is no function propagation among proteins. If we set $\beta = 1$, there is no function propagation from function to protein nodes. If we set $\gamma = 0$, there is no function annotations on the protein nodes in the directed birelational graph. If we set $\gamma = 1$, there is no propagation probability between protein and function nodes. Given these reasons, we vary β and γ in $[0.1, 0.9]$ to investigate the parameter sensitivity of TMC and TMEC. The recorded *1-RankingLoss* and *Coverage* with respect to different settings of β and γ are shown in Fig. 4 for the Human benchmark. The recorded *1-RankingLoss* and *Coverage* for the Yeast benchmark are included in the online supplemental material. Here, we utilize the surf figure to visualize the parameter sensitivity of TMC and TMEC with respect to β and γ. The density plot shows that similar colors in the same figure are parameters with similar predictive performance. We also fix the scale of TMC and TMEC in the same range for each evaluation metric on each data set.

From these figures, we can observe that TMEC is less sensitive than TMC to parameter selection. TMEC has wider ranges of effective parameter values than TMC, and it can often achieve better performance than TMC in the same parameters setting. These advantages can be attributed to the fact that TMEC not only makes use of the structure differences among the single kernels from various data sources, but also takes advantage of the complimentary information among the base classifiers on each single kernel. These results again support the advantage of *classifier integration* over *kernel integration*.

Both TMC and TMEC are more sensitive to γ than to β. γ adjusts the distribution of function labels on protein and function nodes. β adjusts the importance of the inter-subgraph (between proteins and functions) and intra-subgraph (between pairwise proteins, or pairwise functions), it determines the speed of label information propagated from function nodes to protein nodes. TMEC achieved relatively stable performance when $\gamma \in [0.4, 0.9]$ and TMC reaches relative stable performance when $\gamma \in [0.5, 0.9]$. Bigger γ means more emphasis on the initial function assignment. This fact reinforces that it is important to keep the original



(a) TMC (1-RankingLoss) (b) TMEC (1-RankingLoss) (c) TMC (Coverage) (d) TMEC(Coverage)

Fig. 4. *1-RankingLoss* and *Coverage* on different β and γ (**Human**).

label assignment in label propagation, it also indicates the rationality of the proposed directed birelational graph.

6 Conclusions

In this paper, we analyze the drawback of using undirected birelational graphs in label propagation. To avoid this limitation, we propose to use a directed birelational graph, and define a TMC on it. We further improve the performance by combining various TMCs trained on multiple data sources (TMEC). Different from traditional methods that make use of multiple data sources by kernel integration, TMEC takes advantage of multiple data sources by classifier integration. TMEC does not require to collect all the data sources beforehand. Our experimental results show that classifier integration is a valuable methodology to leverage multiple biological data sources.

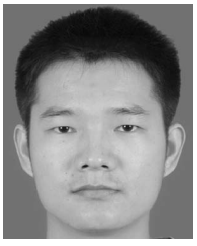
Acknowledgments

The authors would like to thank the anonymous reviewers and editors for their valuable comments. This paper was partially supported by grants from the US National Science Foundation (NSF) (no. IIS-0905117), the NSF Career Award (no. IIS-1252318), the Natural Science Foundation of China (project nos. 61070090, 61003174, 61101234), the Natural Science Foundation of Guangdong Province (S2012010009961), the Specialized Research Fund for the Doctoral Program of Higher Education (20110172120027), Fundamental Research Funds for the Central Universities of China (XDJK2010B002), the Doctoral Fund of Southwest University (project nos. SWU110063 and SWU113034), and the China Scholarship Council.

References

[1] G.O. Consortium et al., "Gene Ontology: Tool for the Unification of Biology," *Nature Genetics,* vol. 25, no. 1, pp. 25-29, 2000.
[2] R. Sharan, I. Ulitsky, and R. Shamir, "Network-Based Prediction of Protein Function," *Molecular Systems Biology,* vol. 3, no. 1, article 88, 2007.
[3] G. Pandey, V. Kumar, and M. Steinbach, "Computational Approaches for Protein Function Prediction," Technical Report TR 06-028, Dept. of Computer Science and Eng., Univ. of Minnesota, 2006.
[4] W. Noble and A. Ben-Hur, "Integrating Information for Protein Function Prediction," *Bioinformatics—From Genomes to Therapies,* vol. 3, T. Lengauer, ed., Wiley-VCH, pp. 1297-1314, 2007.
[5] G. Lanckriet, T. De Bie, N. Cristianini, M. Jordan, and W. Noble, "A Statistical Framework for Genomic Data Fusion," *Bioinformatics,* vol. 20, no. 16, pp. 2626-2635, 2004.
[6] D. Lewis, "Combining Kernels for Classification," PhD dissertation, Columbia Univ., 2006.
[7] C. Leslie, E. Eskin, A. Cohen, J. Weston, and W. Noble, "Mismatch String Kernels for Discriminative Protein Classification," *Bioinformatics,* vol. 20, no. 4, pp. 467-476, 2004.
[8] P. Pavlidis, J. Weston, J. Cai, and W. Noble, "Learning Gene Functional Classifications from Multiple Data Types," *J. Computational Biology,* vol. 9, no. 2, pp. 401-411, 2002.
[9] S. Mostafavi and Q. Morris, "Fast Integration of Heterogeneous Data Sources for Predicting Gene Function with Limited Annotation," *Bioinformatics,* vol. 26, no. 14, pp. 1759-1765, 2010.
[10] K. Tsuda, H. Shin, and B. Schölkopf, "Fast Protein Classification with Multiple Networks," *Bioinformatics,* vol. 21, no. suppl. 2, pp. 59-65, 2005.
[11] M. Re and G. Valentini, "Ensemble Based Data Fusion for Gene Function Prediction," *Proc. Eighth Int'l Workshop Multiple Classifier Systems,* pp. 448-457, 2009.
[12] O. Chapelle et al., *Semi-Supervised Learning,* vol. 2, MIT Press, 2006.
[13] H. Shin, K. Tsuda, and B. Schölkopf, "Protein Functional Class Prediction with a Combined Graph," *Expert Systems with Applications,* vol. 36, no. 2, pp. 3284-3292, 2009.
[14] J. Weston, C. Leslie, E. Ie, D. Zhou, A. Elisseeff, and W. Noble, "Semi-Supervised Protein Classification Using Cluster Kernels," *Bioinformatics,* vol. 21, no. 15, pp. 3241-3247, 2005.
[15] J. Jiang, "Learning Protein Functions from Bi-Relational Graph of Proteins and Function Annotations," *Proc. 11th Int'l Conf. Algorithms in Bioinformatics,* pp. 128-138, 2011.
[16] J. Jiang and L. McQuay, "Predicting Protein Function by Multi-Label Correlated Semi-Supervised Learning," *IEEE/ACM Trans. Computational Biology and Bioinformatics,* vol. 9, no. 4, pp. 1059-1069, July/Aug. 2012.
[17] G. Pandey, C. Myers, and V. Kumar, "Incorporating Functional Inter-Relationships into Protein Function Prediction Algorithms," *BMC Bioinformatics,* vol. 10, no. 1, article 142, 2009.
[18] X. Zhang and D. Dai, "A Framework for Incorporating Functional Interrelationships into Protein Function Prediction Algorithms," *IEEE/ACM Trans. Computational Biology and Bioinformatics,* vol. 9, no. 3, pp. 740-753, May/June 2012.
[19] L. Tang, J. Chen, and J. Ye, "On Multiple Kernel Learning with Multiple Labels," *Proc. 21st Int'l Joint Conf. Artifical Intelligence (IJCAI '09),* pp. 1255-1260, 2009.
[20] S. Bucak, R. Jin, and A. Jain, "Multi-Label Multiple Kernel Learning by Stochastic Approximation: Application to Visual Object Recognition," *Proc. Advances in Neural Information Processing Systems (NIPS '10),* pp. 1145-1154, 2010.
[21] G. Yu, C. Domeniconi, H. Rangwala, G. Zhang, and Z. Yu, "Transductive Multi-Label Ensemble Classification for Protein Function Prediction," *Proc. 18th ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining (SIGKDD '12),* pp. 1077-1085, 2012.
[22] H. Wang, H. Huang, and C. Ding, "Image Annotation Using Bi-Relational Graph of Images and Semantic Labels," *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR '11),* pp. 793-800, 2011.
[23] A. Ruepp et al., "The FunCat, a Functional Annotation Scheme for Systematic Classification of Proteins from Whole Genomes," *Nucleic Acids Research,* vol. 32, no. 18, pp. 5539-5545, 2004.
[24] G. Tsoumakas, I. Katakis, and I. Vlahavas, "Mining Multi-Label Data," *Data Mining and Knowledge Discovery Handbook,* pp. 667-685, Springer, 2010.
[25] A. Elisseeff and J. Weston, "A Kernel Method for Multi-Labeled Classification," *Proc. Advances in Neural Information Processing Systems (NIPS '01),* pp. 681-687, 2001.
[26] G. Chen, J. Zhang, F. Wang, C. Zhang, and Y. Gao, "Efficient Multi-Label Classification with Hypergraph Regularization," *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR '09),* pp. 1658-1665, 2009.
[27] Z. Barutcuoglu, R. Schapire, and O. Troyanskaya, "Hierarchical Multi-Label Prediction of Gene Function," *Bioinformatics,* vol. 22, no. 7, pp. 830-836, 2006.
[28] G. Valentini, "True Path Rule Hierarchical Ensembles for Genome-Wide Gene Function Prediction," *IEEE/ACM Trans. Computational Biology and Bioinformatics,* vol. 8, no. 3, pp. 832-847, May/June 2011.
[29] D. Lin, "An Information-Theoretic Definition of Similarity," *Proc. 15th Int'l Conf. Machine Learning,* pp. 296-304, 1998.
[30] D. Zhou, O. Bousquet, T. Lal, J. Weston, and B. Schölkopf, "Learning with Local and Global Consistency," *Proc. Advances in Neural Information Processing Systems (NIPS '04),* pp. 321-328, 2004.
[31] M. Belkin, P. Niyogi, and V. Sindhwani, "Manifold Regularization: A Geometric Framework for Learning from Labeled and Unlabeled Examples," *J. Machine Learning Research,* vol. 7, pp. 2399-2434, 2006.
[32] A. Dempster, N. Laird, and D. Rubin, "Maximum Likelihood from Incomplete Data via the Em Algorithm," *J. Royal Statistical Soc. Series B (Methodological),* vol. 39, pp. 1-38, 1977.
[33] S. Mostafavi, D. Ray, D. Warde-Farley, C. Grouios, and Q. Morris, "GeneMANIA: A Real-Time Multiple Association Network Integration Algorithm for Predicting Gene Function," *Genome Biology,* vol. 9, no. suppl. 1, article S4, 2008.
[34] N. Cesa-Bianchi, M. Re, and G. Valentini, "Synergy of Multi-Label Hierarchical Ensembles, Data Fusion, and Cost-Sensitive Methods for Gene Functional Inference," *Machine Learning,* vol. 88, nos. 1/2, pp. 1-33, 2012.
[35] M. Gönen and E. Alpaydin, "Multiple Kernel Learning Algorithms," *J. Machine Learning Research,* vol. 12, pp. 2211-2268, 2011.

[36] Z. Zha, T. Mei, J. Wang, Z. Wang, and X. Hua, "Graph-Based Semi-Supervised Learning with Multiple Labels," *J. Visual Comm. and Image Representation,* vol. 20, no. 2, pp. 97-103, 2009.
[37] H. Tong, C. Faloutsos, and J. Pan, "Random Walk with Restart: Fast Solutions and Applications," *Knowledge and Information Systems,* vol. 14, no. 3, pp. 327-346, 2008.
[38] B. Schwikowski et al., "A Network of Protein-Protein Interactions in Yeast," *Nature Biotechnology,* vol. 18, no. 12, pp. 1257-1261, 2000.
[39] P. Bogdanov and A. Singh, "Molecular Function Prediction Using Neighborhood Features," *IEEE/ACM Trans. Computational Biology and Bioinformatics,* vol. 7, no. 2, pp. 208-217, Apr.-June 2010.
[40] L. Kuncheva and C. Whitaker, "Measures of Diversity in Classifier Ensembles and Their Relationship with the Ensemble Accuracy," *Machine Learning,* vol. 51, no. 2, pp. 181-207, 2003.
[41] R. Fan and C. Lin, "A Study on Threshold Selection for Multi-Label Classification," techical report, Dept. of Computer Science, Nat'l Taiwan Univ., 2007.
[42] S. Bucak, R. Jin, and A. Jain, "Multi-Label Learning with Incomplete Class Assignments," *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR '11),* pp. 2801-2808, 2011.
[43] L.I. Kuncheva, J.C. Bezdek, and R. Duin, "Decision Templates for Multiple Classifier Fusion: An Experimental Comparison," *Pattern Recognition,* vol. 34, no. 2, pp. 299-314, 2001.
[44] G. Yu, G. Zhang, Z. Zhang, Z. Yu, and L. Deng, "Semi-Supervised Classification Based on Subspace Sparse Representation," *Knowledge and Information Systems,* http://link.springer.com/article/10.1007%2Fs10115-013-0702-2, 2013.



Guoxian Yu received the BSc degree in software engineering from Xi'an University of Technology, China, in 2007 and the PhD degree in computer science from South China University of Technology, Guangzhou, in 2013. He is an associate professor at the College of Computer and Information Science, Southwest University, Chongqing, China. He visited the Data Mining Lab at George Mason University, Virginia, from 2011 to 2013. His current research interests include machine learning, data mining, and bioinformatics. He is a recipient of the Best Poster Award of SDM12 Doctral Forum and the Best Student Paper Award of the 10th IEEE International Conference on Machine Learning and Cybernetics.

Huzefa Rangwala received the PhD degree in computer science from the University of Minnesota in 2008. He is an assistant professor in the Department of Computer Science, George Mason University, Virginia. His core research interests include bioinformatics, machine learning, and high-performance computing. He was the recipient of the 2013 US National Science Foundation Early Faculty Career Award, the 2013 Volgenau Outstanding Teaching Faculty Award, the 2012 Computer Science Department Outstanding Teaching Faculty Award, and the 2011 Computer Science Department Outstanding Junior Researcher Award.

Carlotta Domeniconi received the BSc degree in computer science from the University of Milan, Italy, in 1992 and the PhD degree in computer science from the University of California, Riverside, in 2002. She is an associate professor in the Department of Computer Science, George Mason University, Virginia. Her research interests include pattern recognition, machine learning, data mining, and feature relevance estimation. She has published extensively in premier data mining and machine learning conferences and journals. She is a recipient of an US National Science Foundation CAREER Award.

Guoji Zhang received the BSc degree in computer application and the PhD degree in circuits and systems from the South China University of Technology, Guangzhou, in 1977 and 1999, respectively. He is a professor at the School of Sciences, South China University of Technology, Guangzhou. His research interests include computational intelligence, computational electromagnetics, and cryptology. He has published more than 50 research papers.

Zhiwen Yu received the BSc and MPhil degrees from the Sun Yat-Sen University, China, in 2001 and 2004, respectively, and the PhD degree in computer science from the City University of Hong Kong, New Kowloon, in 2008. He is a professor at the School of Computer Science and Engineering, South China University of Technology, Guangzhou. His research interests include bioinformatics, machine learning, pattern recognition, intelligent computing, and data mining. He has published more than 70 technical articles in referred journals and conference proceedings in the areas of bioinformatics, artificial intelligence, pattern recognition, and multimedia. He is a member of the IEEE.