A New Approach to Link Prediction in Gene Regulatory Networks

Turki Turki[1,2(✉)] and Jason T.L. Wang[2(✉)]

[1] Computer Science Department, King Abdulaziz University, P.O. Box 80221, Jeddah 21589, Saudi Arabia
tturki@kau.edu.sa

[2] Department of Computer Science, New Jersey Institute of Technology, University Heights, Newark, NJ 07102, USA
{ttt2,wangj}@njit.edu

Abstract. Link prediction is an important data mining problem that has many applications in different domains such as social network analysis and computational biology. For example, biologists model gene regulatory networks (GRNs) as directed graphs where nodes are genes and links show regulatory relationships between the genes. By predicting links in GRNs, biologists can gain a better understanding of the cell regulatory circuits and functional elements. Existing supervised methods for GRN inference work by building a feature-based classifier from gene expression data and using the classifier to predict links in the GRNs. In this paper we present a new supervised approach for link prediction in GRNs. Our approach employs both gene expression data and topological features extracted from the GRNs, in combination with three machine learning algorithms including random forests, support vector machines and neural networks. Experimental results on different datasets demonstrate the good performance of the proposed approach and its superiority over the existing methods.

Keywords: Machine learning · Data mining · Feature selection · Bioinformatics · Systems biology

1 Introduction

Link prediction is an important data mining problem that finds many applications in social network analysis. One of the methods for solving the link prediction problem is to extract features from a given partially observed network and incorporate these features into a classifier. The links (i.e., edges) between entities (i.e., nodes or vertices) in the given partially observed network are labeled [2,18]. One then uses the classifier built from the given partially observed network to predict the presence of links for unobserved pairs of entities in the network. Liben-Nowell and Kleinberg [17] showed that topological features can be used to increase the accuracy of link prediction in social network analysis. In this paper

K. Jackowski et al. (Eds.): IDEAL 2015, LNCS 9375, pp. 404–415, 2015.
DOI: 10.1007/978-3-319-24834-9_47

we extend their techniques to solve an important bioinformatics problem in systems biology. Specifically, we present a new supervised method to infer gene regulatory networks (GRNs) through link prediction with topological features.

Several authors have developed supervised methods for GRN inference [3,6,27]. For example, Gillani *et al.* [9] presented CompareSVM, which uses support vector machines (SVMs) to predict the regulatory relationship between a transcription factor (TF) and a target gene where the regulatory relationship is represented by a directed edge (link), and both the TF and target gene are nodes in a gene network. SIRENE [6,19] is another supervised method, which splits the network inference problem into many binary classification problems using one SVM for each TF. The trained SVM classifiers are then used to predict which genes are regulated. The final step is to combine all SVM classifiers to produce a ranked list of TF-gene interactions in decreasing order, and to construct a network based on the ranked list. Cerulo *et al.* [3] developed a SVM-based method for GRN inference, which uses positive and unlabeled data for training the SVM classifier. Ernst *et al.* [7] developed a similar semi-supervised approach for GRN inference. In contrast to the above methods, all of which use gene expression data to predict TF-gene interactions [6,12,19], our approach considers features related to the topological structure of a network, which, to the best of our knowledge, is the first of its kind in GRN inference.

Feature extraction is crucial in building efficient classifiers for link prediction [1,8,17]. We adapt topological features employed in social network analysis for network inference in systems biology. We propose to use these topological features alone or combine them with gene expression data. As our experimental results show later, this new approach outperforms the previously developed supervised methods that use only gene expression data for network inference [3,6,9,19].

The rest of this paper is organized as follows. Section 2 presents our approach, explaining the techniques employed for feature extraction and feature vector construction. These features are used in combination with three machine learning algorithms including random forests, support vector machines and neural networks. Section 3 presents experimental results, showing the relative performance of the machine learning algorithms and demonstrating the superiority of our approach over the existing methods. Section 4 concludes the paper and points out some directions for future research.

2 Proposed Approach

2.1 Feature Extraction

Let $G = (V, E)$ be the directed graph that represents the topological structure of a gene regulatory network (GRN) where E is the set of edges or links and V is the set of nodes or vertices in G. Our goal is to build a classifier that includes topological features alone or combined with gene expression data. There are totally sixteen topological features, which are described in detail below.

Node Degree. In considering node degrees, each directed edge $e = (u, v) \in E$ has four topological features, *indeg(u)*, *outdeg(u)*, *indeg(v)*, and *outdeg(v)*, which are defined as the number of edges entering u, leaving u, entering v, and leaving v, respectively.

Normalized Closeness Centrality. Normalized closeness centrality measures the closeness between a node and all other nodes in the graph G. For each node or vertex $v \in V$, the normalized closeness centrality $C(v^{in})$ is defined as

$$C(v^{in}) = \frac{|V| - 1}{\sum\limits_{i \neq v} d(i, v)} \tag{1}$$

where $d(i, v)$, $i \neq v$, is the distance from $i \in V$ to $v \in V$, and $|V|$ is the number of vertices in the graph G [15]. The distance from i to v is the number of edges on the shortest path from i to v. If no such path exists, then the distance is set equal to ∞. Since G is a directed graph, the distance from i to v is not necessarily the same as the distance from v to i. We define the normalized closeness centrality $C(v^{out})$ as

$$C(v^{out}) = \frac{|V| - 1}{\sum\limits_{v \neq i} d(v, i)} \tag{2}$$

where $d(v, i)$, $v \neq i$, is the distance from $v \in V$ to $i \in V$. In considering normalized closeness centrality, each directed edge $e = (u, v) \in E$ has four topological features, $C(u^{in})$, $C(u^{out})$, $C(v^{in})$, and $C(v^{out})$.

Eccentricity. The eccentricity of a vertex $v \in V$ is the maximum distance between v and any other vertex $i \in V$ [24]. For each vertex $v \in V$, the eccentricity $\epsilon(v^{in})$ is defined as

$$\epsilon(v^{in}) = \max_{i \in V} \ d(i, v) \tag{3}$$

The eccentricity $\epsilon(v^{out})$ is defined as

$$\epsilon(v^{out}) = \max_{i \in V} \ d(v, i) \tag{4}$$

In considering eccentricity, each directed edge $e = (u, v) \in E$ has four topological features, $\epsilon(u^{in})$, $\epsilon(u^{out})$, $\epsilon(v^{in})$, and $\epsilon(v^{out})$.

Betweenness Centrality. Betweenness centrality measures the centrality of a vertex in the graph G [28]. For each vertex $v \in V$, the betweenness centrality of v, denoted $Between(v)$, is defined as

$$Between(v) = \sum_{i \neq v \neq j} \frac{\sigma_{i,j}(v)}{\sigma_{i,j}} \tag{5}$$

where $\sigma_{i,j}$ is the total number of shortest paths from vertex i to vertex j and $\sigma_{i,j}(v)$ is the total number of shortest paths from vertex i to vertex j that

pass through v [15,28]. In considering betweenness centrality, each directed edge $e = (u, v) \in E$ has two topological features, $Between(u)$ and $Between(v)$.

Eigenvector Centrality. Eigenvector centrality is another centrality measure where vertices in the graph have different importance. A vertex connected to a very important vertex is different from a vertex that is connected to a less important one. This concept is incorporated into eigenvector centrality [20]. For each vertex $v \in V$, the eigenvector centrality of v, denoted $Eigen(v)$, is defined as [21]

$$Eigen(v) = \frac{1}{\lambda} \sum_{i \in V} a_{v,i} Eigen(i) \tag{6}$$

where λ is a constant and $\mathbf{A} = (a_{v,i})$ is the adjacency matrix, i.e., $a_{v,i} = 1$ if vertex v is linked to vertex i, and $a_{v,i} = 0$ otherwise. The above eigenvector centrality formula can be rewritten in the matrix form as

$$\mathbf{Ax} = \lambda \mathbf{x} \tag{7}$$

where $\mathbf{x}$ is the eigenvector of the adjacency matrix $\mathbf{A}$ with the eigenvalue λ. In considering eigenvector centrality, each directed edge $e = (u, v) \in E$ has two topological features, $Eigen(u)$ and $Eigen(v)$.

2.2 Feature Vector Construction

Suppose we are given n genes where each gene has p expression values. The gene expression profile of these genes is denoted by $G \subseteq R^{n \times p}$, which contains n rows, each row corresponding to a gene, and p columns, each column corresponding to an expression value [19]. To train a classifier, we need to know the regulatory relationships among some genes. Suppose these regulatory relationships are stored in a matrix $H \subseteq R^{m \times 3}$. H contains m rows, where each row shows a known regulatory relationship between two genes, and three columns. The first column shows a transcription factor (TF). The second column shows a target gene. The third column shows the label, which is $+1$ if the TF is known to regulate the target gene or -1 if the TF is known not to regulate the target gene. The matrix H represents a partially observed or known gene regulatory network for the n genes. If the label of a row in H is $+1$, then the TF in that row regulates the target gene in that row, and hence that row represents a link or edge of the network. If the label of a row in H is -1, then there is no link between the corresponding TF and target gene in that row.

Given a pair of genes g_1 and g_2 where the regulatory relationship between g_1 and g_2 is unknown, our goal is to use the trained classifier to predict the label of the gene pair. The predicted label is either $+1$ (i.e., a link is predicted to be present between g_1 and g_2) or -1 (i.e., a link is predicted to be missing between g_1 and g_2). Using biological terms, the present link means g_1 (transcription factor) regulates g_2 (target gene) whereas the missing link means g_1 does not regulate g_2.

To perform training and predictions, we construct a feature matrix $D \subseteq R^{k \times 2p}$ with k feature vectors based on the gene expression profile G. For a pair of genes g_1 and g_2, we create their feature vector d, which is stored in the feature matrix D, denoted by D_d and defined as

$$D_d = [g_1^1, g_1^2, \ldots, g_1^p, g_2^1, g_2^2, \ldots, g_2^p] \tag{8}$$

where $g_1^1, g_1^2, \ldots, g_1^p$ are the gene expression values of g_1, and $g_2^1, g_2^2, \ldots, g_2^p$ are the gene expression values of g_2. The above feature vector definition has been used by the existing supervised network inference methods [3,6,9,19]. In the rest of this paper we will refer to the above technique for constructing feature vectors as Ge, indicating that it is based on gene expression data only.

In addition, we propose to construct another feature matrix $D^{'} \subseteq R^{k \times 16}$. Each feature vector d in the feature matrix $D^{'}$, denoted by $D_d^{'}$, is defined as

$$D_d^{'} = [t_1, t_2] \tag{9}$$

$$\begin{aligned} t_1 = indeg(g_1), C(g_1^{in}), \epsilon(g_1^{in}), outdeg(g_1), \\ C(g_1^{out}), \epsilon(g_1^{out}), Between(g_1), Eigen(g_1) \end{aligned} \tag{10}$$

$$\begin{aligned} t_2 = indeg(g_2), C(g_2^{in}), \epsilon(g_2^{in}), outdeg(g_2), \\ C(g_2^{out}), \epsilon(g_2^{out}), Between(g_2), Eigen(g_2) \end{aligned} \tag{11}$$

We will refer to this feature vector construction technique as To, indicating that it is based on the sixteen topological features proposed in the paper.

Finally, we construct the third feature matrix $D^{''} \subseteq R^{k \times (2p+16)}$. Each feature vector d in the feature matrix $D^{''}$, denoted by $D_d^{''}$, contains both gene expression data and topological features, and is defined as

$$D_d^{''} = [g_1^1, g_1^2, ..., g_1^p, g_2^1, g_2^2, ..., g_2^p, t_1, t_2] \tag{12}$$

We will refer to this feature vector construction technique as All, indicating that it is based on all the features described in the paper.

3 Experiments and Results

We conduct a series of experiments to evaluate the performance of our approach and compare it with the existing methods for gene regulatory network (GRN) inference [13]. Below, we describe the datasets used in our study, our experimental methodology, and the experimental results.

3.1 Datasets

We used GeneNetWeaver [23] to generate the datasets related to yeast and E. coli. We first built five different networks, taken from yeast, where the networks

Table 1. Yeast networks used in the experiments

Network	Directed	#Nodes	#Edges	#Pos examples	#Neg examples
Yeast 50	Yes	50	63	63	63
Yeast 100	Yes	100	281	281	281
Yeast 150	Yes	150	333	333	333
Yeast 200	Yes	200	517	517	517
Yeast 250	Yes	250	613	613	613

Table 2. E. coli networks used in the experiments

Network	Directed	#Nodes	#Edges	#Pos examples	#Neg examples
E. coli 50	Yes	50	68	68	68
E. coli 100	Yes	100	177	177	177
E. coli 150	Yes	150	270	270	270
E. coli 200	Yes	200	415	415	415
E. coli 250	Yes	250	552	552	552

contained 50, 100, 150, 200, 250 genes (or nodes) respectively. For each network, we generated three files of gene expression data. These files were labeled as knockouts, knockdowns and multifactorial, respectively. A knockout is a technique to deactivate the expression of a gene, which is simulated by setting the transcription rate of this gene to zero [9,23]. A knockdown is a technique to reduce the expression of a gene, which is simulated by reducing the transcription rate of this gene by half [9,23]. Multifactorial perturbations are simulated by randomly increasing or decreasing the activation of the genes in a network simultaneously [23].

Table 1 presents details of the yeast networks, showing the number of nodes (edges, respectively) in each network. The edges or links in a network form positive examples. In addition, we randomly picked the same number of negative examples where each negative example corresponds to a missing link in the network. The networks and gene expression profiles for E. coli were generated similarly. Table 2 presents details of the networks generated from E. coli.

3.2 Experimental Methodology

We considered nine classification algorithms, denoted by RF + All, NN + All, SVM + All, RF + Ge, NN + Ge, SVM + Ge, RF + To, NN + To, SVM + To, respectively. Table 3 lists these algorithms and their abbreviations. RF + All (RF + Ge, RF + To, respectively) represents the random forest algorithm combined with all features including both gene expression data and topological features (RF combined with only gene expression data, RF combined with only topological features, respectively). NN + All (NN + Ge, NN + To, respectively)

Table 3. Nine classification algorithms and their abbreviations

Abbreviation	Classification algorithm and features
RF + All	Random forests with all features
NN + All	Neural networks with all features
SVM + All	Support vector machines with all features
RF + Ge	Random forests with gene expression features
NN + Ge	Neural networks with gene expression features
SVM + Ge	Support vector machines with gene expression features
RF + To	Random forests with topological features
NN + To	Neural networks with topological features
SVM + To	Support vector machines with topological features

represents the neural network algorithm combined with all features (NN combined with only gene expression data, NN combined with only topological features, respectively). SVM + All (SVM + Ge, SVM + To, respectively) represents the support vector machine algorithm combined with all features (SVM combined with only gene expression data, SVM combined with only topological features, respectively). SVM + Ge is adopted by the existing supervised network inference methods [3,6,9,19].

Software used in this work included: the random forest package in R [16], the neuralnet package in R [10], and the SVM with linear kernel in the LIBSVM package [4]. We used R to write some utility tools for performing the experiments, and employed the package, igraph, to extract topological features from a network [5].

The performance of each classification algorithm was evaluated through 10-fold cross validation. The size of each fold was approximately the same, and each fold contained the same number of positive and negative examples. On each fold, the *balanced error rate* (BER) [11] of a classification algorithm was calculated where the BER is defined as

$$BER = \frac{1}{2} \times \left(\frac{FN}{TP + FN} + \frac{FP}{FP + TN} \right) \tag{13}$$

FN is the number of false negatives (i.e., present links that were mistakenly predicted as missing links). TP is the number of true positives (i.e., present links that were correctly predicted as present links). FP is the number of false positives (i.e., missing links that were mistakenly predicted as present links). TN is the number of true negatives (i.e., missing links that were correctly predicted as missing links). For each algorithm, the mean BER, denoted MBER, over 10 folds was computed and recorded. The lower MBER an algorithm has, the better performance that algorithm achieves. Statistically significant performance differences between classification algorithms were calculated using Wilcoxon signed rank tests [13,14]. As in [22,25], we consider p-values below 0.05 to be statistically significant.

3.3 Experimental Results

Table 4 shows the MBERs of the nine classifications on the fifteen yeast datasets used in the experiments. For each dataset, the algorithm having the best performance (i.e., with the lowest MBER) is in boldface. Table 5 shows, for each yeast dataset, the p-values of Wilcoxon signed rank tests between the best algorithm, represented by '-', and the other algorithms. A p-value in boldface ($p \leq 0.05$) indicates that the corresponding result is significant. It can be seen from Table 4 that random forests performed better than support vector machines and neural networks. In particular, random forests combined with all features (i.e., RF + All) performed the best on 10 out of 15 yeast datasets. For the other five yeast datasets, RF + All was not statistically different from the best algorithms according to Wilcoxon signed rank tests ($p > 0.05$); cf. Table 5.

Table 6 shows the MBERs of the nine classification algorithms on the fifteen E. coli datasets used in the experiments. Table 7 shows, for each E. coli dataset, the p-values of Wilcoxon signed rank tests between the best algorithm, represented by '-', and the other algorithms. It can be seen from Table 6 that random forests combined with topological features (i.e., RF + To) performed the best on 6 out of 15 E. coli datasets. For the other nine E. coli datasets, RF + To was not statistically different from the best algorithms according to Wilcoxon signed rank tests ($p > 0.05$); cf. Table 7.

Table 4. MBERs of nine classification algorithms on fifteen yeast datasets

Dataset	RF + All	NN + All	SVM + All	RF + Ge	NN + Ge	SVM + Ge	RF + To	NN + To	SVM + To
Yeast 50 knockouts	16.6	**15.0**	20.0	18.3	15.8	20.0	17.7	16.3	19.4
Yeast 50 knockdowns	**16.6**	17.5	20.0	19.1	22.7	16.9	17.7	18.8	19.4
Yeast 50 multifactorial	**16.1**	17.5	20.8	15.5	24.7	17.2	17.7	16.6	19.4
Yeast 100 knockouts	12.8	13.7	17.3	14.4	20.0	16.7	**11.6**	22.2	14.5
Yeast 100 knockdowns	14.2	14.9	15.2	14.1	29.4	14.1	**11.8**	17.5	14.5
Yeast 100 multifactorial	12.0	17.8	18.0	12.3	21.3	18.4	**11.4**	20.0	14.5
Yeast 150 knockouts	**5.10**	10.7	14.1	5.40	10.4	13.6	6.10	15.4	11.0
Yeast 150 knockdowns	**5.00**	10.5	10.4	5.80	14.7	10.9	5.80	16.0	11.0
Yeast 150 multifactorial	**4.10**	12.4	13.9	**4.10**	15.1	16.1	5.80	10.8	11.0
Yeast 200 knockouts	**1.90**	4.00	5.30	**1.90**	4.90	5.60	2.50	13.7	3.70
Yeast 200 knockdowns	**1.90**	5.10	5.80	**1.90**	6.30	10.5	2.70	9.80	3.70
Yeast 200 multifactorial	**1.90**	6.80	10.1	**1.90**	4.70	14.3	2.50	5.50	3.70
Yeast 250 knockouts	**3.80**	8.40	7.60	4.10	5.50	7.20	7.40	10.6	10.4
Yeast 250 knockdowns	**4.00**	9.70	7.40	**4.00**	6.20	7.30	8.10	7.70	10.4
Yeast 250 multifactorial	4.00	8.50	8.50	**3.90**	6.00	9.00	7.50	11.3	10.4

Table 5. P-values of Wilcoxon signed rank tests between the best algorithm, represented by '-', and the other algorithms for each yeast dataset

Dataset	RF + All	NN + All	SVM + All	RF + Ge	NN + Ge	SVM + Ge	RF + To	NN + To	SVM + To
Yeast 50 knockouts	0.75	-	0.28	0.44	0.67	0.07	0.46	0.78	0.27
Yeast 50 knockdowns	-	0.79	0.46	0.17	0.13	0.52	0.58	0.86	0.33
Yeast 50 multifactorial	-	0.70	0.28	0.89	0.07	0.68	0.58	0.68	0.33
Yeast 100 knockouts	0.34	0.44	**0.05**	0.14	**0.01**	**0.02**	-	**0.04**	0.15
Yeast 100 knockdowns	0.22	0.24	0.16	0.22	**0.00**	**0.03**	-	0.08	0.12
Yeast 100 multifactorial	0.46	0.13	**0.02**	0.27	**0.01**	**0.04**	-	**0.00**	0.12
Yeast 150 knockouts	-	**0.01**	**0.02**	1.00	**0.01**	**0.01**	0.10	**0.01**	**0.02**
Yeast 150 knockdowns	-	**0.02**	**0.02**	0.17	**0.03**	**0.01**	0.17	**0.00**	**0.02**
Yeast 150 multifactorial	-	**0.01**	**0.01**	-	**0.01**	**0.01**	0.17	**0.01**	**0.02**
Yeast 200 knockouts	-	**0.01**	**0.01**	-	**0.01**	**0.02**	0.50	**0.01**	0.17
Yeast 200 knockdowns	-	**0.04**	**0.01**	-	**0.01**	**0.02**	0.10	**0.01**	0.17
Yeast 200 multifactorial	-	**0.04**	**0.01**	-	**0.06**	**0.00**	0.50	**0.01**	0.17
Yeast 250 knockouts	-	**0.03**	0.20	0.17	0.06	0.10	0.17	0.09	**0.02**
Yeast 250 knockdowns	-	**0.01**	**0.10**	-	0.06	**0.05**	0.07	0.20	**0.05**
Yeast 250 multifactorial	1.00	**0.05**	0.11	-	**0.01**	**0.03**	0.18	**0.05**	**0.03**

Table 6. MBERs of nine classification algorithms on fifteen E. coli datasets

Dataset	RF + All	NN + All	SVM + All	RF + Ge	NN + Ge	SVM + Ge	RF + To	NN + To	SVM + To
E. coli 50 knockouts	14.5	5.70	9.80	18.8	22.0	21.5	**5.00**	11.6	18.4
E. coli 50 knockdowns	14.5	9.50	10.7	19.1	19.0	16.7	**5.00**	10.8	18.4
E. coli 50 multifactorial	15.3	10.3	8.20	15.7	22.9	18.0	**5.00**	10.3	18.4
E. coli 100 knockouts	10.3	9.50	14.2	12.0	14.4	17.7	**5.60**	6.00	13.6
E. coli 100 knockdowns	10.6	11.6	14.2	11.4	14.1	18.0	**5.40**	9.80	13.6
E. coli 100 multifactorial	9.80	10.4	11.4	9.80	12.4	13.6	**7.10**	9.90	13.6
E. coli 150 knockouts	**2.40**	4.40	2.90	**2.40**	8.80	5.90	2.70	5.90	3.30
E. coli 150 knockdowns	**2.20**	4.40	2.70	**2.20**	8.10	3.70	2.70	3.80	3.30
E. coli 150 multifactorial	**2.20**	3.80	2.40	**2.20**	8.70	2.40	2.70	3.30	3.30
E. coli 200 knockouts	5.50	5.50	5.10	5.50	**4.60**	**4.60**	6.40	11.0	6.00
E. coli 200 knockdowns	5.30	5.00	5.80	6.10	**4.40**	5.50	6.40	8.50	6.00
E. coli 200 multifactorial	5.00	4.20	3.90	4.90	**2.80**	3.40	6.40	8.00	6.00
E. coli 250 knockouts	**6.60**	10.3	7.00	7.70	8.80	8.00	6.30	12.7	10.0
E. coli 250 knockdowns	5.30	9.30	5.90	**5.10**	6.70	7.50	6.20	11.9	10.0
E. coli 250 multifactorial	5.40	11.1	8.00	**5.20**	11.0	9.70	6.30	10.6	10.0

Table 7. P-values of Wilcoxon signed rank tests between the best algorithm, represented by '-', and the other algorithms for each E. coli dataset

Dataset	RF + All	NN + All	SVM + All	RF + Ge	NN + Ge	SVM + Ge	RF + To	NN + To	SVM + To
E. coli 50 knockouts	0.06	1.00	0.46	**0.03**	**0.00**	**0.01**	-	0.10	**0.04**
E. coli 50 knockdowns	0.06	0.07	0.46	**0.01**	**0.01**	**0.01**	-	0.46	**0.04**
E. coli 50 multifactorial	0.06	0.49	0.46	0.08	**0.01**	**0.04**	-	**0.04**	**0.04**
E. coli 100 knockouts	0.08	**0.04**	**0.01**	**0.01**	**0.00**	**0.01**	-	0.68	**0.04**
E. coli 100 knockdowns	**0.02**	0.16	**0.01**	**0.01**	**0.01**	**0.01**	-	0.27	**0.04**
E. coli 100 multifactorial	0.67	0.22	0.17	0.67	0.11	0.06	-	0.79	0.17
E. coli 150 knockouts	-	0.10	0.25	-	**0.02**	0.06	0.65	**0.04**	0.65
E. coli 150 knockdowns	-	**0.04**	0.17	-	0.06	0.06	1.00	0.10	1.00
E. coli 150 multifactorial	-	0.06	1.00	-	0.10	1.00	1.00	0.10	1.00
E. coli 200 knockouts	0.91	0.50	0.46	0.91	0.68	-	0.41	**0.03**	0.46
E. coli 200 knockdowns	0.89	0.50	0.27	0.75	-	0.46	0.46	0.11	0.68
E. coli 200 multifactorial	0.67	0.09	0.14	0.67	-	0.28	0.13	0.13	0.20
E. coli 250 knockouts	-	**0.02**	0.23	0.65	**0.00**	0.12	0.71	0.12	**0.05**
E. coli 250 knockdowns	0.17	**0.02**	**0.02**	-	**0.02**	**0.02**	0.10	**0.02**	**0.01**
E. coli 250 multifactorial	1.00	**0.02**	**0.01**	-	**0.01**	**0.01**	0.72	**0.01**	**0.02**

These results show that using random forests with the proposed topological features alone or combined with gene expression data performed well. In particular, the RF + All algorithm achieved the best performance on 14 out of all 30 datasets. This is far better than the SVM + Ge algorithm used by the existing supervised network inference methods [3,6,9,19], which achieved the best performance on one dataset only (i.e., the E. coli 200 knockouts dataset in Table 6).

It is worth pointing out that, for a fixed dataset size (e.g., 200), the SVM+To algorithm always yielded the same mean balanced error rate (MBER) regardless of which technique (knockout, knockdown or multifactorial) was used to generate the gene expression profiles. This happens because these different gene expression profiles correspond to the same network, and SVM+To uses only the topological features extracted from the network without considering the gene expression data. On the other hand, due to the randomness introduced in random forests and neural networks, RF+To and NN+To yielded different MBERs even for the same network.

4 Conclusion

We present a new approach to network inference through link prediction with topological features. Our experimental results showed that using the topological features alone or combined with gene expression data performs better than

the existing network inference methods that use only gene expression data. Our work assumes that there are exactly the same number of positive examples (i.e., links that are present) and negative examples (i.e., links that are missing) in the datasets. In many biological networks, however, negative datasets (majority class) are usually much larger than positive datasets (minority class). In future work we plan to extend our previously developed imbalanced classification algorithms and boosting algorithms [26,29,30] to tackle the imbalanced link prediction problem for gene network construction. We have adopted default parameter settings for the three machine learning algorithms studied in the paper. We also plan to explore other parameter settings (e.g., different kernels with different parameter values in SVMs) in the future.

References

1. Al Hasan, M., Chaoji, V., Salem, S., Zaki, M.: Link prediction using supervised learning. In: SDM06: Workshop on Link Analysis, Counter-terrorism and Security (2006)
2. Al Hasan, M., Zaki, M.J.: A survey of link prediction in social networks. In: Aggarwal, C.C. (ed.) Social Network Data Analytics, pp. 243–275. Springer, New York (2011)
3. Cerulo, L., Elkan, C., Ceccarelli, M.: Learning gene regulatory networks from only positive and unlabeled data. BMC Bioinformatics **11**(1), 228 (2010)
4. Chang, C.C., Lin, C.J.: Libsvm: a library for support vector machines. ACM Trans. Intell. Syst. Technol. (TIST) **2**(3), 27 (2011)
5. Csardi, G., Nepusz, T.: The igraph software package for complex network research. Int. J. Complex Syst. **1695**(5), 1–9 (2006)
6. De Smet, R., Marchal, K.: Advantages and limitations of current network inference methods. Nat. Rev. Microbiol. **8**(10), 717–729 (2010)
7. Ernst, J., Beg, Q.K., Kay, K.A., Balázsi, G., Oltvai, Z.N., Bar-Joseph, Z.: A semi-supervised method for predicting transcription factor-gene interactions in escherichia coli. PLoS Comput. Biol. **4**(3), e1000044 (2008)
8. Fire, M., Tenenboim, L., Lesser, O., Puzis, R., Rokach, L., Elovici, Y.: Link prediction in social networks using computationally efficient topological features. In: 2011 IEEE Third International Conference on Privacy, Security, Risk and Trust (PASSAT) and 2011 IEEE Third Inernational Conference on Social Computing (SocialCom), pp. 73–80. IEEE (2011)
9. Gillani, Z., Akash, M.S., Rahaman, M., Chen, M.: CompareSVM: supervised, support vector machine (SVM) inference of gene regularity networks. BMC Bioinformatics **15**(1), 395 (2014)
10. Günther, F., Fritsch, S.: neuralnet: training of neural networks. R Journal **2**(1), 30–38 (2010)
11. Guyon, I., Alamdari, A.R.S.A., Dror, G., Buhmann, J.M.: Performance prediction challenge. In: International Joint Conference on Neural Networks, IJCNN 2006, pp. 1649–1656. IEEE (2006)
12. Hecker, M., Lambeck, S., Toepfer, S., Van Someren, E., Guthke, R.: Gene regulatory network inference: data integration in dynamic models: a review. Biosystems **96**(1), 86–103 (2009)
13. Japkowicz, N., Shah, M.: Evaluating Learning Algorithms. Cambridge University Press, Cambridge (2011)

14. Kanji, G.K.: 100 Statistical Tests. Sage, London (2006)
15. Kolaczyk, E.D.: Statistical Analysis of Network Data: Methods and Models. Springer, New York (2009)
16. Liaw, A., Wiener, M.: Classification and regression by randomforest. R News **2**(3), 18–22 (2002). http://CRAN.R-project.org/doc/Rnews/
17. Liben-Nowell, D., Kleinberg, J.: The link-prediction problem for social networks. J. Am. Soc. Inform. Sci. Technol. **58**(7), 1019–1031 (2007)
18. Menon, A.K., Elkan, C.: Link prediction via matrix factorization. In: Gunopulos, D., Hofmann, T., Malerba, D., Vazirgiannis, M. (eds.) ECML PKDD 2011, Part II. LNCS, vol. 6912, pp. 437–452. Springer, Heidelberg (2011)
19. Mordelet, F., Vert, J.P.: Sirene: supervised inference of regulatory networks. Bioinformatics **24**(16), i76–i82 (2008)
20. Newman, M.: Networks: An Introduction. Oxford University Press, Oxford (2010)
21. Newman, M.E.: The mathematics of networks. In: The New Palgrave Encyclopedia of Economics, vol. 2, pp. 1–12 (2008)
22. Nuin, P.A.S., Wang, Z., Tillier, E.R.M.: The accuracy of several multiple sequence alignment programs for proteins. BMC Bioinformatics **7**, 471 (2006). http://dx.doi.org/10.1186/1471-2105-7-471
23. Schaffter, T., Marbach, D., Floreano, D.: Genenetweaver: in silico benchmark generation and performance profiling of network inference methods. Bioinformatics **27**(16), 2263–2270 (2011)
24. Takes, F.W., Kosters, W.A.: Computing the eccentricity distribution of large graphs. Algorithms **6**(1), 100–118 (2013)
25. Thompson, J.D., Plewniak, F., Poch, O.: A comprehensive comparison of multiple sequence alignment programs. Nucleic Acids Res. **27**(13), 2682–2690 (1999). http://dx.doi.org/10.1093/nar/27.13.2682
26. Turki, T., Wei, Z.: IPRed: Instance reduction algorithm based on the percentile of the partitions. In: Proceedings of the 26th Modern AI and Cognitive Science Conference MAICS, pp. 181–185 (2015)
27. Wang, J.T.L.: Inferring gene regulatory networks: challenges and opportunities. J. Data Min. Genomics Proteomics **06**(01), e118 (2015). http://dx.doi.org/10.4172/2153-0602.1000e118
28. Ye, J., Cheng, H., Zhu, Z., Chen, M.: Predicting positive and negative links in signed social networks by transfer learning. In: Proceedings of the 22nd International Conference on World Wide Web, pp. 1477–1488. International World Wide Web Conferences Steering Committee (2013)
29. Zhong, L., Wang, J.T.L., Wen, D., Aris, V., Soteropoulos, P., Shapiro, B.A.: Effective classification of microRNA precursors using feature mining and adaboost algorithms. OMICS **17**(9), 486–493 (2013)
30. Zhong, L., Wang, J.T.L., Wen, D., Shapiro, B.A.: Pre-mirna classification via combinatorial feature mining and boosting. In: 2012 IEEE International Conference on Bioinformatics and Biomedicine, BIBM 2012, Philadelphia, PA, USA, 4–7 October 2012, pp. 1–4 (2012). http://doi.ieeecomputersociety.org/10.1109/BIBM.2012.6392700