Bioinformatics, 36(26), 2020, i762–i769
doi: 10.1093/bioinformatics/btaa840
ECCB2020

Systems

A neuro-evolution approach to infer a Boolean network from time-series gene expressions

Shohag Barman[1] and Yung-Keun Kwon[2,*]

[1]Department of Computer Science, American International University-Bangladesh (AIUB), Dhaka 1229, Bangladesh and [2]School of IT Convergence, University of Ulsan, Ulsan 44610, Republic of Korea

*To whom correspondence should be addressed.

Abstract

Summary: In systems biology, it is challenging to accurately infer a regulatory network from time-series gene expression data, and a variety of methods have been proposed. Most of them were computationally inefficient in inferring very large networks, though, because of the increasing number of candidate regulatory genes. Although a recent approach called GABNI (genetic algorithm-based Boolean network inference) was presented to resolve this problem using a genetic algorithm, there is room for performance improvement because it employed a limited representation model of regulatory functions.

In this regard, we devised a novel genetic algorithm combined with a neural network for the Boolean network inference, where a neural network is used to represent the regulatory function instead of an incomplete Boolean truth table used in the GABNI. In addition, our new method extended the range of the time-step lag parameter value between the regulatory and the target genes for more flexible representation of the regulatory function. Extensive simulations with the gene expression datasets of the artificial and real networks were conducted to compare our method with five well-known existing methods including GABNI. Our proposed method significantly outperformed them in terms of both structural and dynamics accuracy.

Conclusion: Our method can be a promising tool to infer a large-scale Boolean regulatory network from time-series gene expression data.

Availability and implementation: The source code is freely available at https://github.com/kwon-uou/NNBNI.

Contact: kwonyk@ulsan.ac.kr

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

1 Introduction

A gene regulatory network (GRN) is represented by a directed graph where nodes and edges denote genes/proteins and interactions between them, respectively, and the analysis of GRNs is crucial to understand characteristics of complex biological systems. It is challenging to accurately infer a GRN from time-coarse gene expression data due to the high-dimensional relationship among the regulatory and regulated molecular components. Recent advances of high-throughput RNA-seq technologies can produce the genome-scale gene expression data, leading to development of many inference methods based on the computational models such as Boolean networks (Kauffman, 1969), differential equations (Chen *et al.*, 1999) and Bayesian networks (Imoto *et al.*, 2002). Among them, the Boolean network model, where a gene state is represented by either 0 (ON) or 1 (OFF) value and a regulatory interaction is defined by a Boolean function or logic table, has been popularly used for inference of GRNs because of the lowest computational cost. However, most existing Boolean network inference methods such as the reverse engineering algorithm (REVEAL; Liang *et al.*, 1998), the Best-Fit (Lähdesmäki *et al.*, 2003) and the Bayesian approach (Han *et al.*, 2014) are not so scalable because they need an exhaustive search for all possible combinations of regulatory candidate genes. Accordingly, the feasible number of regulatory inputs was limited to a very small value, even 2 or 3, which means that a relatively complicated regulatory rule cannot be inferred accurately.

To resolve this problem, the mutual information which can measure the significance of relevance between a pair of variables has been frequently considered. For example, the relevance network method (Butte and Kohane, 2000) calculated the mutual information between every gene pair and eliminated interactions with the mutual information values less than a given threshold. The REVEAL (Liang *et al.*, 1998) identified a set of regulatory genes by maximizing a mutual information score for a target gene. The context likelihood of relatedness algorithm (Faith *et al.*, 2007) advanced the REVEAL by employing a background correction of mutual information scores. The algorithm for the reconstruction of accurate cellular networks (ARACNE) algorithm (Margolin *et al.*, 2006) inferred a network by determining a significant gene under a constraint of the data processing inequality and filtering out the weakest connection from every gene triplet. A common

drawback of these methods is that they examined a pairwise mutual information instead of the exact multivariate mutual information due to a high computational cost. This can cause a performance degradation, particularly in large-scale inference problem. To overcome this limitation, we had proposed a mutual information-based Boolean network inference (MIBNI) which computed an approximated multivariate mutual information more accurately than the pairwise mutual information calculation (Barman and Kwon, 2017). It showed better performance than the other existing inference methods in terms of the inference accuracy and the running time. However, the solution found by MIBNI was not still optimal and stuck in local optima, because of not only the approximated measure but also the constructive optimization framework. In this regard, a genetic algorithm-based Boolean network inference (GABNI) was suggested in another previous study (Barman and Kwon, 2018), which can efficiently avoid the local optimum through the genetic search algorithm, and it showed a significant performance improvement over the existing methods. However, GABNI has a disadvantage in the representation of the network dynamics. It used a Boolean truth table to describe the update rule, but the table is often incomplete because the regulatory rule cannot be specified for unobserved input conditions in the expression data. This relaxed constraint can induce the undesirable early stopped solution and an inherent degradation of inference accuracy. In addition, GABNI assumed only one-time-step lag between the regulatory and the target genes, which has limited the representation power of the regulatory relation.

In this article, we propose a novel neural network-based Boolean network inference (NNBNI) method. Similar to GABNI, it exploits the GA as a global search technique. However, our method replaced the incomplete Boolean truth table with a feed-forward multilayered neural network (NN) to represent the regulatory rule. In addition, NNBNI extended the time-step lag parameter between the regulatory and the regulated genes to represent more complex regulatory functions. Our proposed method consists of two stages where MIBNI is first applied because it can efficiently find a small-scale regulatory relation. If it fails to find an optimal solution, a GA starts to search a better regulatory relation. The fitness of a solution in the GA is evaluated by a multilayered NN which computes how accurately the state values of a target gene is fitted by those of the GA-selected candidate regulatory genes. This search process is independently executed for every target gene and all the results are combined to generate the whole inferred Boolean network. We note that there was a trial to combine a GA and a NN for the network inference problem (Marbach *et al.*, 2009). However, it was intended to infer only the network structure by employing simple GA operators such as one-point crossover and the performance over large-scale networks was not validated.

For reliable performance assessment of our method, we further included some recent tree-based approaches which have showed interesting results for comparison. For example, the gene network inference with ensemble of trees (GENIE3) algorithm (Huynh-Thu *et al.*, 2010) which selects a feature by a gene ranking using random forests or extra-trees was recently proposed. It was extended to the dynamical GENIE3 (dynGENIE3; Huynh-Thu and Geurts, 2018) by introducing a new parameter, the decay rate of the genes. In addition, a boosted tree-based gene regulatory network (BTNET; Park *et al.*, 2018) using Adaboost or gradient boosting to compute regulatory interaction scores was successfully suggested. In this study, we compared NNBNI with five other methods, GABNI, ARACNE, GENIE3, dynGENIE3 and BTNET. We tested them in both the large-scale gene expression datasets from the artificial and real networks, and found that NNBNI outperformed all the other methods. Taken together, NNBNI can be considered as a promising tool for the Boolean network inference.

2 Materials and methods

2.1 A Boolean network model

In this study, we employed a Boolean network model to infer a GRN from time-series gene expression data. A Boolean network (Kauffman, 1969) is represented by a directed graph $G(V, A)$ where $V = \{v_1, v_2, .., v_N\}$ is a set of nodes and $A \subseteq V \times V$ is a set of directed interactions. The value of a node is represented by a Boolean value of 1 (ON) or 0 (OFF) and it is updated by a Boolean function. For example, assume that a node $v \in V$ is regulated by k other genes $u_1, u_2, \ldots, u_k$ ($u_i \in V$). It is then formulated as follows: The value of v at time-step $t+1$ denoted by $v(t+1)$ is updated by a Boolean function $f : \{0,1\}^k \rightarrow \{0,1\}$ of the values of $u_1, u_2, \ldots, u_k$ at time-steps $t, t-1, \ldots, t-\tau+1$, where τ denotes the maximum time-step lag between the regulatory gene and the target gene. In other words, the update scheme of v can be written as:

$$v(t+1) = f(u_1(t), \ldots, u_1(t-\tau+1), u_2(t), \ldots, u_2(t-\tau+1),$$

$$\cdots, u_k(t), \ldots, u_k(t-\tau+1)).$$

Accordingly, a total of $2^{2\tau \cdot k}$ Boolean functions are possible for f. We note that the maximum time-step lag value (τ) was fixed to one in most existing studies including our previous method, GABNI (Barman and Kwon, 2018). This limitation was usually set to reduce the search cost which, however, has caused the reduction of representation power of the regulatory function. In this regard, we have loosened the constraint by extending $\tau = 3$ in this study.

2.2 The Boolean network inference problem

The Boolean network inference problem handled in this study is a problem to infer not only a set of regulatory interactions but also a set of update Boolean functions from the time-series gene expression data. The inference performance can be assessed by comparing the Boolean trajectory generated by the inferred network and the observed Boolean time-series gene expression. Let $v'(t)$ the predicted Boolean value of gene v at time t in the inferred Boolean network. We define the *gene-wise dynamics consistency* $C(v, v')$ as the similarity between the Boolean trajectories of the observed gene expression $v(t)$ and the estimated gene expression $v'(t)$, as follows:

$$C(v, v') = \frac{\sum_{t=\tau+1}^{T} I(v(t) = v'(t))}{T - \tau}, \quad (1)$$

where T is the total number of time-steps, and $I(\cdot)$ is an indicator function that returns 1 if the condition is true, otherwise 0. (Note that the comparison can start from $t = \tau + 1$ considering the maximum time-step lag.) Finally, we can define the *dynamics accuracy* of an inferred network as the average of gene-wise dynamics consistency over all genes as follows:

$$Dynamics\ accuracy = \frac{\sum_{i=1}^{N} C(v_i, v_i')}{N}.$$

2.3 Structural performance metrics

In case that the structure of a gold standard or ground-truth network is known, we can further evaluate the inference performance with respect to the network structure. To this end, we used three measures: precision, recall and structural accuracy. Precision is the ratio of correctly inferred connections over the total number of positive predictions as follows:

$$Precision = \frac{TP}{TP + FP},$$

where TP (true positive) and FP (false positive) denote the numbers of correctly and incorrectly predicted connections, respectively. Recall is the ratio of true predicted connections over the total number of actual connections:

$$Recall = \frac{TP}{TP + FN},$$

where FN (false negative) means the number of non-inferred connections in $G(V, A)$. Structural accuracy is the ratio of correct predictions out of all predictions as follows:

$$Structural\ accuracy = \frac{TP + TN}{TP + FP + FN + TN},$$

where TN (true negative) is the number of correct negative predictions.

2.4 MIBNI and GABNI methods

As described before, our proposed method includes the existing method MIBNI (see Barman and Kwon, 2017 for details) because it showed an excellent performance in the case of inferring a regulatory function with a relatively small number of input genes. MIBNI consists of two main subroutine called MIFS and SWAP: The MIFS subroutine selects the most informative k regulatory genes, $U = \{u_1, u_2, \ldots, u_k\} \subseteq V$ for a target gene v based on an approximated multivariate mutual information, and the SWAP subroutine tries to improve the gene-wise dynamics consistency by iteratively swapping the subsets of U and $V \setminus U$ until there is no improvement. MIBNI used a simple update rule representation scheme based on only conjunction or disjunction functions.

GABNI (see Barman and Kwon, 2018 for details) was devised to improve the MIBNI by exploiting a GA which searches an optimal solution represented by a set of regulatory genes. The fitness of a solution is evaluated by the Boolean truth table that is optimally constructed from the observed Boolean gene expression dataset. However, the inferred table can be incomplete because some elements in the table cannot be specified due to the deficiency of the corresponding observation in gene expression data.

3 Our proposed method

In this work, we propose a novel NN-combined GA for Boolean network inference (NNBNI), and Figure 1 illustrates the overall framework. A real-valued time-series gene expression dataset is given as input. Then, it is converted into a binarized dataset using a K-means discretization method (MacQueen *et al.*, 1967), which classifies all expression values of each gene into two clusters marked by 1 (ON) and 0 (OFF) to denote higher and lower expression level, respectively. Given a target gene, our method searches an optimal regulatory function through two stages. In the first stage, MIBNI is applied to infer an optimal update rule because it was proven to be efficient if a target gene is regulated by a relatively small number of regulatory genes (Barman and Kwon, 2017). If the found solution is optimal [i.e. the gene-wise dynamics consistency is 1.0; see Equation (1)], it is marked as the resultant regulatory relation for the target gene. Otherwise, a GA starts on the second stage to search a list of optimal regulatory genes. It can be formulated as a combinatorial optimization problem to select an optimal subset. The GA first generates a population consisting of a set of solutions randomly initialized. Then it selects a pair of parent solutions from the population, and a new offspring solution is created by the crossover and the mutation operations in sequence. The fitness of the solution is evaluated by the NN learning model, and the GA updates the population by replacing a parent solution with the offspring solution. The loop is repeated and the GA stops after a fixed number of generations. This search process is independently executed for every target gene and all obtained solutions constitutes a regulatory network along with the regulatory functions. Table 1 shows the specified values of the parameters used in GA and NN in this study. In the following subsections, we describe each operation of our GA in detail.

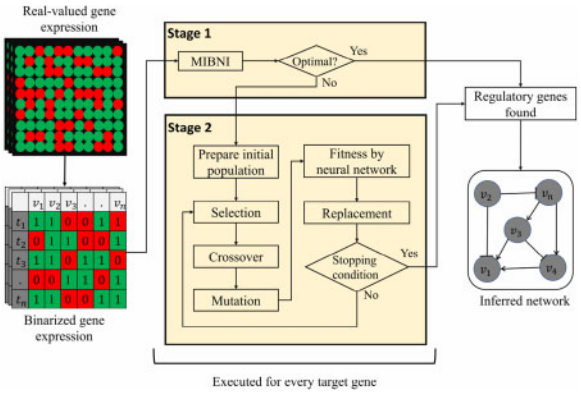

Fig. 1. Overall framework of NNBNI. A real-valued time-series gene expression data are converted into binary values. Given a target gene, MIBNI is first applied to infer a simple regulatory Boolean function. If an optimal solution is found, it is included in the final inferred network. Otherwise, a hybrid GA combined with NN is applied to find a better solution. In the GA, the fitness of a solution is evaluated by the NN learning. This process is independently executed for every target gene. All inference results are integrated into a final inferred Boolean network where the update function of each gene is represented by a NN

3.1 Solution representation

A solution in our GA is used to represent a set of selected regulatory genes. Let $v \in V = \{v_1, \ldots, v_N\}$ a target gene. Then, a solution is represented by a binary vector of length N such as $s = s_1 s_2 \cdots s_N \in \{0,1\}^N$ where s_i indicates the inclusion or exclusion of v_i in the set of regulatory genes. For an efficient search, we set two constraints. The first one is the maximum number of regulatory genes which prevents a solution from having too many regulatory inputs (it was set to $0.6 \times N$ in this study; see Table 1). The other one is the minimum mutual information with a target gene. A gene whose mutual information value with the target gene is lower than a given threshold is not eligible for the selection, because it is not informative enough to explain the relation of the target gene (it was set to 0.05 in this study; see Table 1).

3.2 Fitness evaluation based on a NN

To evaluate a solution s, we employed a feed-forward NN with a single hidden layer. Figure 2 shows an illustrative example where s selects $\{v_3, v_5, v_8\}$ among $V = \{v_1, v_2, \ldots, v_{10}\}$. Assume that the maximum number of the time-step lag was set to three and $v \in V$ is a target gene. Then, nine inputs neurons, $v_i(t), v_i(t-1), v_i(t-2)(i = 3, 5, 8)$, and one output neuron of $v(t+1)$ are generated as shown in the figure. The NN learns the gene expression data using the back-propagation algorithm which is a gradient descent method to minimize the mean square error between the estimated and the actual outputs. Once the learning is finished, we define the fitness function of s using the gene-wise dynamics consistency [see Equation (1)] as follows:

$$Fitness(s) = \frac{1}{(1 - C(v, v')) \cdot \gamma + k}$$

where v' is the predicted expression by the learned NN, γ is a weight factor and k is the number of regulatory genes chosen by the solution, in favor of a simpler regulation function.

Table 1. Parameters of GA and NN in this study

	Parameters	Setting
GA	Population size	$N + 10$
	Maximum number of regulatory genes	$0.6 \times N$
	Minimum mutual information	0.05
	γ in fitness function	2^8
	H in adjusted fitness	3
	Number of GA iterations	1000
	Mutation probability	0.01
NN	Learning algorithm	Back-propagation
	Activation function	Sigmoid
	Momentum	0.90
	Number of hidden neurons	Number of input neurons × 2
	Learning rate	0.15

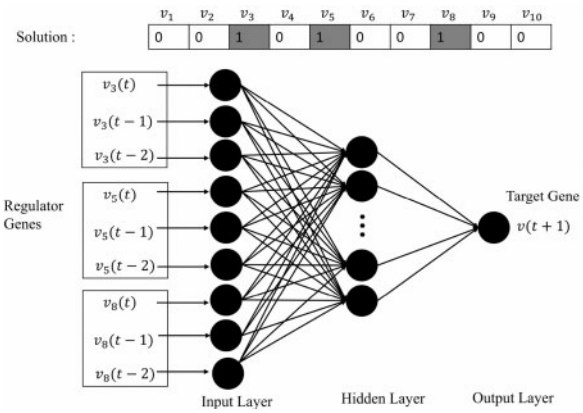

Fig. 2. An example of the NN-based evaluation of a solution in GA. The example solution selects $\{v_3, v_5, v_8\}$ among $V = \{v_1, \ldots, v_{10}\}$ as the set of candidate regulatory genes. Let $v \in V$ the target gene and assume that the maximum time-step lag is set to 3. Accordingly, nine variables of $v_3(t), \ldots, v_8(t-2)$ are used for the input neurons whereas $v(t+1)$ corresponds to the output neuron. Once the NN learns the gene expression data, the gene-wise dynamics consistency can be obtained to represent the fitness of the solution

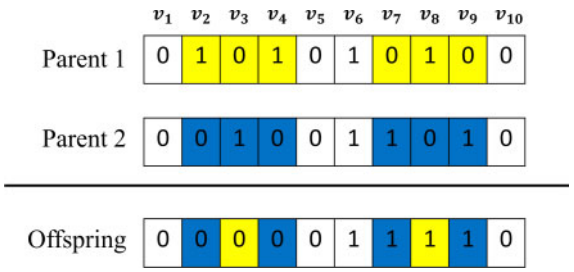

Fig. 3. The crossover operator. Parents 1 and 2 include $\{v_2, v_4, v_6, v_8\}$ and $\{v_3, v_6, v_7, v_9\}$ for regulatory gene sets, respectively. If a bit value in a gene is equivalent to both parents, it is copied to the corresponding gene in the offspring (white color). Otherwise, a bit value selected between two parents uniformly at random (yellow or blue color) is copied

3.3 Selection, crossover and mutation

For simplicity, we employed the same selection, crossover, and mutation operators used in GABNI (see details in Barman and Kwon, 2018). Our GA selects two parent solutions from a population at each generation. We defined a variant roulette wheel selection, which is one of the most representative selection operators in the GA field. Specifically, the selection probability of a solution s is proportional to the adjusted fitness as follows:

$$Adjusted - Fitness(s) = A \times Fitness(s) + B$$

where $A = \frac{H}{F_{MAX} - F_{MIN}}$ and $B = 1 - A \cdot F_{MIN}$. Here, F_{MAX} and F_{MIN} denote the maximum and the minimum fitness values in the population, respectively, and H is a parameter which adjusts the selection probability of the best solution to H times of that of the worst solution in the population.

After two parent solutions are chosen, a crossover operator is applied to produce a new solution called offspring (Fig. 3). If the binary value of an element is common in both parents, it is copied to the offspring (the white element in Fig. 3). Otherwise, it is selected between two parents uniformly at random (the yellow or the blue element in Fig. 3). Finally, the mutation operator flips the value of each element in the offspring with a small probability (Fig. 4), which helps to prevent the GA fallen in the local optimum.

3.4 Replacement and stopping criterion

The fitness of the offspring created by the crossover and mutation operators is evaluated by the NN as explained before. If the offspring is superior to one of the parent solutions, the former replaces the latter in the population for the next generation. The GA stops after a fixed number of generations.

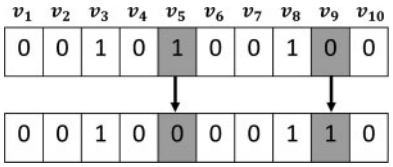

Fig. 4. The mutation operator. The example offspring solution selects $\{v_3, v_5, v_8\}$ as regulatory genes. Each gene is mutated with a small probability. In this example, the values of v_5 and v_9 were flipped

4 Results

For reliable performance evaluation, we compared NNBNI with five other methods: GABNI (our previous method), ARACNE (a mutual information-based constructive method), and three recent tree-based approaches such as GENIE3, dynGENIE3 and BTNET. We tested them with the gene expression datasets of both the artificial and real regulatory networks as shown in the following subsections. Since it is necessary to determine a threshold about the confidence level of an inferred interaction in the compared methods, we specified the best threshold value by trial-and-error over the tested dataset. In addition, we added a function which searches a Boolean update function for a given set of regulatory inputs into the original ARACNE method so that the dynamics accuracy can be examined. Finally, we note that all compared methods have been modified to consider the same maximum time step with that used in our method (see Supplementary Table S1 for the parameter values of ARACNE, GENIE3, dynGENIE3 and BTNET specified in this study).

4.1 Performance on the gene expression data of the artificial networks

To generate the artificial gene expression datasets, we first created randomly structured artificial networks by employing the Barabasi–Albert (BA) model (Barabasi and Albert, 1999; see Supplementary Fig. S1). Specifically, a total of 300 random networks with different network sizes ($|V| = 10, 20, \ldots, 100$ and $|A| = 2.5 \times |V|$) were created. The update function of each gene was chosen uniformly at random between a logical conjunction (AND) and disjunction (OR) functions. Considering the definition of the maximum time-step lag (τ), the Boolean state values of all genes for the first τ time-steps (i.e. $t = 1, 2, \ldots, \tau$) were randomly initialized (see Section 2). On the other hand, the state values from time-step $\tau + 1$ to T [the maximum time-step; see Equation (1)] are synchronously determined by the update functions. In this study, we set τ and T to three and $|V| + 20$, respectively, and generated 30 different gene expression datasets for each random network size. Therefore, a total of 300 random gene expression datasets were generated for test.

4.1.1 Structural accuracy comparison

Figure 5 shows the results with respect to the structural accuracy of the inferred networks. For more precise analysis, we classified all target genes into 10 classes according to the number of incoming links (D) ranged from 1 to 10. As shown in the figure, precision, recall and structural accuracy of all methods decrease as D increases, because it denotes the degree of difficulty in the network inference problem. We also observed that both NNBNI and GABNI always found the optimal solution in the case of $D = 1$. This is because MIBNI is applied on the first stage in both methods. On the other hand, NNBNI shows significantly higher precision, recall and structure accuracy than the other methods for all cases of $D > 1$. In other words, our new method stably outperformed the other methods, irrespective of the degree of inference difficulty. We also note that the overwhelming performance of NNBNI is consistently observed

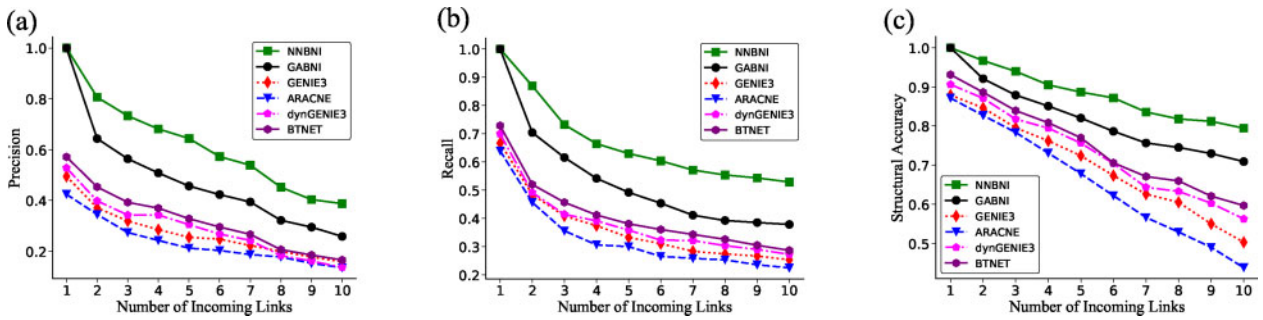

Fig. 5. Comparison of precision, recall and structural accuracy between NNBNI and other methods in BA random networks. Results of (**a**) precision, (**b**) recall and (**c**) structural accuracy, respectively. A total of 300 random networks with 10 different network sizes $|V| = 10, 20, \ldots, 100$ were created and 30 Boolean gene expression datasets were generated for each network size. A total number of 16 500 nodes in those networks were examined and classified into 10 groups according to the number of incoming links. Y-axis values show the average precision, recall and structural accuracy values in each group. Detailed results per the network size are shown in Supplementary Figures S2–4

Table 2. The structural accuracies in inferring the Boolean GRN from a noisy gene expression.

Noise level	Inference methods	TP	FP	FN	Structural accuracy
0% Noise	NNBNI	16	3	9	**0.8787**
	GABNI	13	8	12	0.8039
	ARACNE	10	20	15	0.6666
	GENIE3	10	22	15	0.6476
	dynGENIE3	14	16	11	0.7326
	BTNET	14	10	11	0.7920
5% Noise	NNBNI	14	14	11	**0.7524**
	GABNI	10	14	15	0.7238
	ARACNE	7	27	18	0.5833
	GENIE3	8	27	17	0.5887
	dynGENIE3	8	19	17	0.6635
	BTNET	8	16	17	0.6915
10% Noise	NNBNI	12	17	13	**0.7087**
	GABNI	9	21	16	0.6509
	ARACNE	6	30	19	0.5504
	GENIE3	6	29	19	0.5596
	dynGENIE3	7	22	18	0.6296
	BTNET	6	22	19	0.6238

Note: The structural accuracy of NNBNI was significantly higher than those of other methods (All p-values < 0.01).

regardless of the network size (see Supplementary Figs S2–4). Furthermore, we have analyzed the inference performance in the noisy gene expression dataset. The noise was added by flipping the Boolean gene value at 5% or 10% probability in each dataset. We observed that NNBNI robustly outperformed the other methods in all terms of precision, recall and structural accuracy (Table 2). In addition, we conducted the sensitivity analysis with respect to four main parameters in Table 1 and observed that the structural accuracy can be sensitive to them (see Supplementary Table S2). This implies that it is needed to carefully specify the parameter values in NNBNI.

4.1.2 Dynamical accuracy comparison

Figure 6 shows the results with respect to the dynamical accuracy of the inferred networks. As shown in Figure 6a, we compared the dynamics accuracy of every method in BA random networks against the network size ($|V| = 10, 20, 30, \ldots, 100$). Intriguingly, both NNBNI and GABNI outperformed the other methods over all tested datasets. In particular, the performance gap with other methods increased as the network size grows. Moreover, the dynamics accuracies of both NNBNI and GABNI were 1.0 (i.e. perfect prediction) in all networks, which means that the GAs in NNBNI and GABNI always found the optimal solution in terms of the gene-wise dynamics consistency [see Equation (1)]. This result can cause a confusion considering that NNBNI showed better performance than GABNI with respect to the structural accuracy in Figure 5. In an effort to discover the reason for it, we further compared the convergence speed of the GAs in NNBNI and GABNI (Figure 6b). Specifically, we examined the change of average dynamics accuracy of the best solution in each GA against the number of generations. As shown in the figure, the GA in NNBNI converges more slowly than that of GABNI. This is because GABNI uses the incomplete truth table to represent the regulatory function. In other words, GABNI can be early stopped although every field in a truth table is not specified due to the deficiency in the gene expression dataset. On the other hand, the regulatory function found by NNBNI is represented by a NN and thus it represents a complete Boolean function which can eventually specify all fields of a truth table. This observation implies that the change of the regulatory function representation from the incomplete truth table to the NN in the GAs led to the performance improvement with respect to the structural accuracy without loss of the dynamical accuracy. In addition, NNBNI can have another advantage over GABNI even in case that both correctly infer the regulatory relation, because the former is more likely to be able to derive a Boolean function from the complete truth table induced by the NN (see e.g. Supplementary Fig. S5). Finally, we examined the convergence speed of NNBNI according to the degree of problem difficulty (Fig. 6c). Specifically, we classified all the genes into 10 groups according to the number of incoming links and examined the average dynamics accuracy of the best solutions in NNBNI for every 100 generations. It is obvious that the GA converges to the optimal solution more slowly as D increases. In the most difficult cases, NNBNI found the best solution at about 700 generations.

4.1.3 Running time

To compare the running time of NNBNI and other methods, we examined the average running time over a total of 300 gene expression datasets on a PC with Intel Core i7 3.4 GHz CPU and 8 GB of RAM (Fig. 7). As shown in the figure, the dynGENIE3 was fastest on average whereas NNBNI and GABNI were slowest due to the genetic search. In addition, NNBNI was slower than GABNI because the GA of the former converges more slowly than that of the latter as we observed in Figure 6b. However, we note that the scalability of NNBNI was not worse than the other methods as the network size increases. As a result, it is most desirable to employ our method when the highest inference accuracy is required, in spite of the increased running time.

4.2 Performance on the gene expression dataset of the real networks

4.2.1 Case study 1: small-scale real networks

We applied our novel approach to infer four real small-scaled networks (see Supplementary Fig. S6 for the gold-standards). The

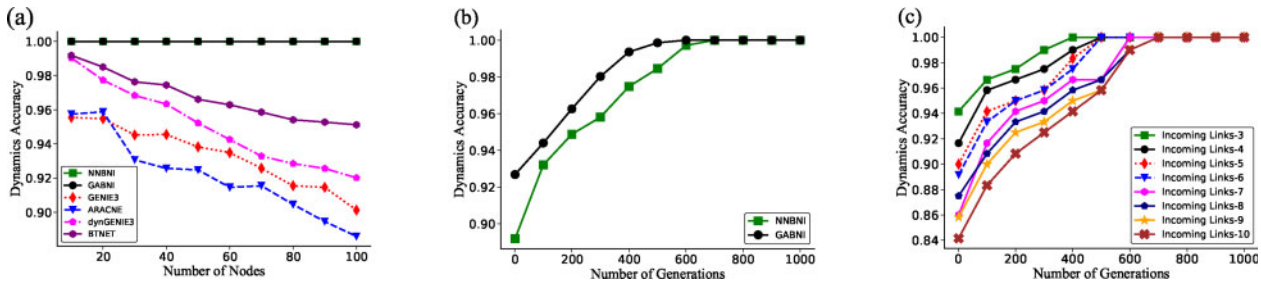

Fig. 6. Dynamics accuracy analysis. (**a**) Comparison of dynamic accuracies between NNBNI and other methods in BA random networks. A total of 300 random networks with 10 different network sizes were created and 30 Boolean gene expression datasets were generated for each network size. Each point denotes the average dynamics accuracy over 30 datasets. Note that the dynamic accuracies of both NNBNI and GABNI were 1.0 in all datasets. (**b**) Comparison of convergence between GABNI and NNBNI. BA random networks with $|V| = 100$ were examined. (**c**) Convergence of dynamics accuracy against the number of generations in NNBNI. BA random networks with $|V| = 100$ were analyzed and the nodes were classified into 10 groups according to the number of incoming links. The average dynamics accuracy of the best solutions was shown for every 100 generation

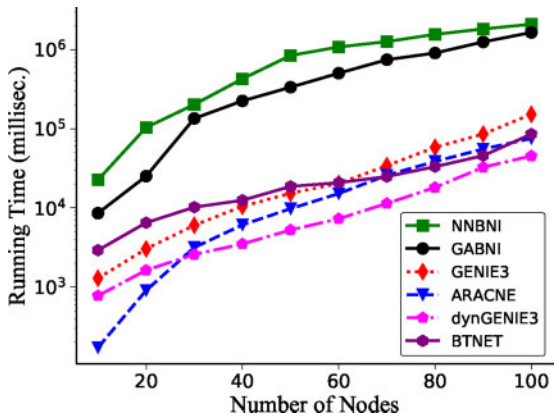

Fig. 7. Comparison of running time of NNBNI and other methods. The Y-axis values represent the average log-scaled running time. Among all methods, dynGENIE3 was fastest, whereas NNBNI and GABNI were slowest

first network is the yeast cell-cycle network consisting of 9 genes and 17 interactions (Simon *et al.*, 2001). It has two types of expression datasets (Spellman *et al.*, 1998), CDC-15 and CDC-28, measured over 24 and 17 time points, respectively. The next two networks, In Silico-1 and -2, are from the benchmark datasets in DREAM4 challenge (Marbach *et al.*, 2010). Both of them consist of 10 genes and 16 interactions and the gene expression datasets were measured over 21 time points. The last one is Saccharomyces *cerevisiae* G1 cell-cycle network consisting of 11 genes and 22 interactions (Rubiolo *et al.*, 2018) where the expression dataset was measured over 18 time points. These five real-valued gene expression datasets were converted into the Boolean values using a *K*-means discretization method (MacQueen *et al.*, 1967; see Supplementary Tables S3–12). Table 3 shows the structural accuracies of all methods. As shown in the table, NNBNI showed the best structural accuracies in all datasets. We depicted five inference results of NNBNI in Figure 8 (see Supplementary Figs S7–11 for the results of other methods). We also computed the dynamics accuracy (Table 4) and observed that NNBNI and GABNI significantly outperformed all other methods. Moreover, NNBNI achieved the perfect dynamics accuracy in four datasets except for *S.cerevisiae*.

4.2.2 Case study 2: large-scale real networks

In this subsection, we tested whether the inference method is applicable to a large-scale real network. To this end, we generated a large-scale network structure by using the GeneNetWeaver (GNW) tool (Schaffter *et al.*, 2011) which extracts modules from known biological interaction networks. The gold standard or reference networks are extracted from a transcriptional regulatory network of

Table 3. Structural accuracies of inference of small-scale real networks

Methods	CDC-15	CDC-28	In Silico-1	In Silico-2	*S.cerevisiae*
NNBNI	0.7820	0.7654	0.7653	0.8144	0.7661
GABNI	0.7160	0.7073	0.7373	0.7777	0.7120
ARACNE	0.6585	0.5476	0.6633	0.6000	0.6796
GENIE3	0.6125	0.5421	0.5533	0.6237	0.6720
dynGENIE3	0.6097	0.5609	0.6435	0.6336	0.6666
BTNET	0.5802	0.5662	0.6800	0.7600	0.6929

S.cerevisiae. Then real-valued time-series expression datasets were generated based on stochastic ordinary differential equations. Five networks with a different size of $(|V|, |A|) = (100, 251)$, $(200, 431)$, $(300, 712)$, $(400, 1221)$ and $(500, 2073)$, which are denoted by GNW100, GNW200, GNW300, GNW400 and GNW500, respectively, were tested. Table 5 shows the structural accuracies. As shown in the table, NNBNI significantly outperformed all other methods and GABNI showed the second best performance. In addition, we examined the dynamical accuracy (Table 6) and observed that NNBNI was best overall datasets. Taken together, our method outperformed all other methods in terms of both the structural and the dynamical inference accuracies in the large-scale gene expression dataset of the real regulatory networks.

5 Conclusions

In this work, we developed NNBNI which is a hybrid GA combined with a supervised NN model to infer a Boolean network from time-series gene expression data. Although many inference methods have been developed, most of them were not efficient to infer large networks due to the scalability problem. In this study, we proposed a novel method which is a NN-combined GA. A NN was used not only to evaluate the quality of a solution in the GA but also to represent the regulatory function. It is notable that the NN replaced a Boolean truth table which was used for regulatory function representation in our previous model. In addition, we more generalized the regulatory function representation by increasing the time-step lag parameter between the regulatory and the target genes. We conducted extensive simulations with the gene expression datasets of both the artificial and real networks and compared the performance of our method with those of five well-known existing methods. Our method significantly outperformed them in terms of both structural and dynamics accuracy. These results indicate that the proposed approach is a promising tool for accurate regulatory networks from time-series gene expression data. One limitation of our method is that the NN is a black-box model, so the regulatory function is not explained in the form of rules. In addition, it is necessary to implement our method in parallel to reduce the running time. Finally, we

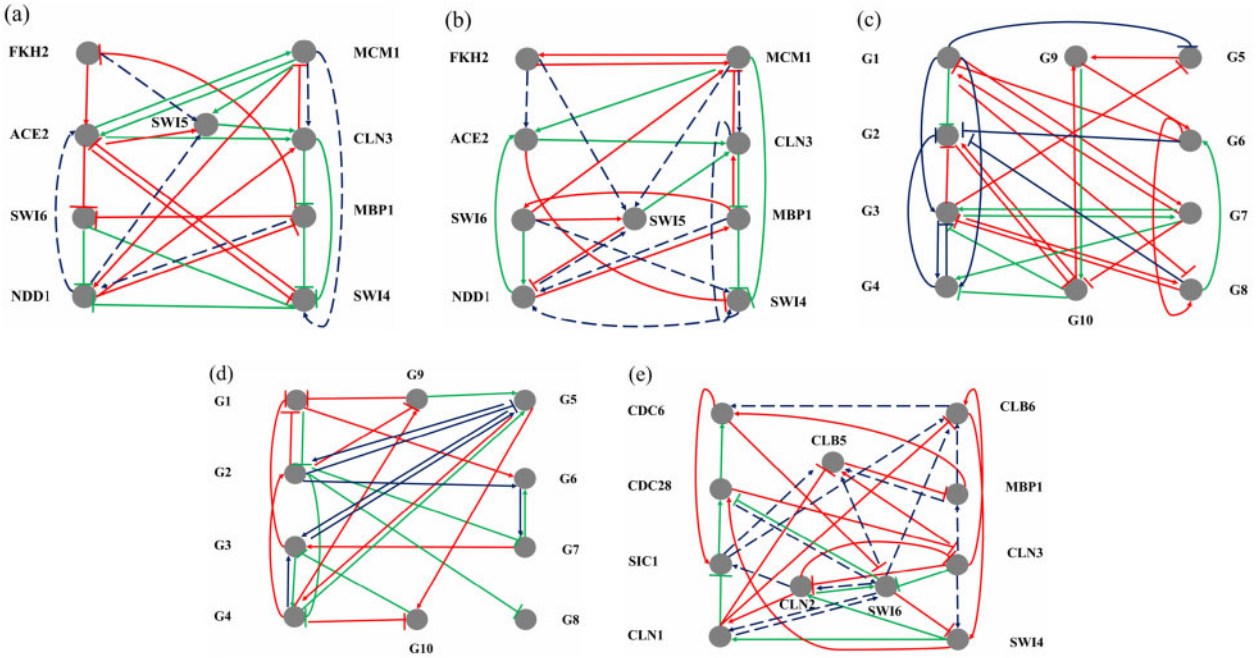

Fig. 8. Inference result of NNBNI with five biological network datasets. The green, red and blue interactions denote TP, FP and FN predictions, respectively. (**a**) Result of CDC15 GRN. There found 11 true positives, 11 FPs and 6 FNs. (**b**) Result of CDC28 GRN. There found 8 true positives, 10 FPs and 9 FNs. (**c**) Result of Silico-1 GRN. There found 8 true positives, 15 FPs and 8 FNs. (**d**) Result of Silico-2 GRN. There found 9 TPs, 11 FPs and 7 FNs. (**e**) Result of *S.cerevisiae* (G1 step) GRN. There found 8 TPs, 15 FPs and 14 FNs. The structural accuracies in (a–e) were 0.7820, 0.7654, 0.7653, 0.8144 and 0.7661, respectively

Table 4. Dynamics accuracies of inference of small-scale real networks

Methods	CDC-15	CDC-28	In Silico-1	In Silico-2	*S.cerevisiae*
NNBNI	1.0000	1.0000	1.0000	1.0000	0.9900
GABNI	0.9900	1.0000	1.0000	0.9900	0.9900
ARACNE	0.9600	0.9600	0.9200	0.9700	0.9700
GENIE3	0.9800	0.9900	0.9700	0.9400	0.9600
dynGENIE3	0.9700	0.9900	0.9900	0.9600	0.9800
BTNET	0.9900	0.9900	0.9800	0.9500	0.9800

Table 5. Structural accuracies of inference of GNW datasets

Methods	GNW100	GNW200	GNW300	GNW400	GNW500
NNBNI	0.8851	0.9016	0.9199	0.9260	0.9428
GABNI	0.8388	0.8436	0.8648	0.8746	0.9025
ARACNE	0.7596	0.7715	0.7843	0.8025	0.8189
GENIE3	0.7872	0.8067	0.8151	0.8314	0.8587
dynGENIE3	0.8163	0.8320	0.8395	0.8455	0.8617
BTNET	0.8085	0.8268	0.8473	0.8561	0.8711

Table 6. Dynamics accuracies of inference of GNW datasets

Methods	GNW100	GNW200	GNW300	GNW400	GNW500
NNBNI	0.9700	0.9800	0.9600	0.9600	0.9500
GABNI	0.9700	0.9600	0.9500	0.9400	0.9400
ARACNE	0.9300	0.9200	0.9100	0.8900	0.8700
GENIE3	0.9500	0.9400	0.9300	0.9100	0.8900
dynGENIE3	0.9600	0.9500	0.9300	0.9200	0.9000
BTNET	0.9500	0.9600	0.9400	0.9200	0.9100

note that NNBNI does not rank the regulatory relations and automatically determines the number of regulatory genes according to the found best solution. This implies that it is not available to compute the area under the precision recall curve and the area under the receiver operating characteristic scores for performance comparison.

Funding

This work was supported by the 2020 Research Fund of University of Ulsan.

Conflict of Interest: none declared.

References

Barabasi,A.-L. and Albert,R. (1999) Emergence of scaling in random networks. *Science*, **286**, 509–512.

Barman,S. and Kwon,Y.-K. (2017) A novel mutual information-based Boolean network inference method from time-series gene expression data. *PLoS One*, **12**, e0171097.

Barman,S. and Kwon,Y.-K. (2018) A Boolean network inference from time-series gene expression data using a genetic algorithm. *Bioinformatics*, **34**, i927–i933.

Butte,A.J. and Kohane,I.S. (2000) Mutual information relevance networks: functional genomic clustering using pairwise entropy measurements. *Pac. Symp. Biocomput.*, **5**, 418–429.

Chen,T. *et al.* (1999). Modeling gene expression with differential equations. *Pac. Symp. Biocomput.*, **4**, 29–40.

Faith,J.J. *et al.* (2007) Large-scale mapping and validation of *Escherichia coli* transcriptional regulation from a compendium of expression profiles. *PLoS Biol.*, **5**, e8.

Han,S. *et al.* (2014) A full Bayesian approach for Boolean genetic network inference. *PLoS One*, **9**, e115806.

Huynh-Thu,V.A. and Geurts,P. (2018) dynGENIE3: dynamical GENIE3 for the inference of gene networks from time series expression data. *Sci. Rep.*, **8**,

Huynh-Thu,V.A. *et al.* (2010) Inferring regulatory networks from expression data using tree-based methods. *PLos One*, **5**, e12776.

Imoto,S. *et al.* (2002). Estimation of genetic networks and functional structures between genes by using Bayesian networks and nonparametric regression. *Pac. Symp. Biocomput.*, **7**, 175–186.

Kauffman,S.A. (1969) Metabolic stability and epigenesis in randomly constructed genetic nets. *J. Theor. Biol.*, **22**, 437–467.

Lähdesmäki,H. *et al.* (2003) On learning gene regulatory networks under the Boolean network model. *Mach. Learn.*, **52**, 147–167.

Liang,S. *et al.* (1998). Reveal, a general reverse engineering algorithm for inference of genetic network architectures. *Pac. Symp. Biocomput.*, **3**, 18–29.

MacQueen,J. *et al.* (1967). Some methods for classification and analysis of multivariate observations. In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, Vol. **1**, pp. 281–297. *University of California Press, Berkeley, CA*.

Marbach,D. *et al.* (2009) Generating realistic in silico gene networks for performance assessment of reverse engineering methods. *J. Comput. Biol.*, **16**, 229–239.

Marbach,D. *et al.* (2010) Revealing strengths and weaknesses of methods for gene network inference. *Proc. Natl. Acad. Sci. USA*, **107**, 6286–6291.

Margolin,A.A. *et al.* (2006) ARACNE: an algorithm for the reconstruction of gene regulatory networks in a mammalian cellular context. *BMC Bioinformatics*, **7**, S7.

Park,S. *et al.* (2018) BTNET: boosted tree based gene regulatory network inference algorithm using time-course measurement data. *BMC Syst. Biol.*, **12**, 20.

Rubiolo,M. *et al.* (2018) Extreme learning machines for reverse engineering of gene regulatory networks from expression time series. *Bioinformatics*, **34**, 1253–1260.

Schaffter,T. *et al.* (2011) GeneNetWeaver: in silico benchmark generation and performance profiling of network inference methods. *Bioinformatics*, **27**, 2263–2270.

Simon,I. *et al.* (2001) Serial regulation of transcriptional regulators in the yeast cell cycle. *Cell*, **106**, 697–708.

Spellman,P.T. *et al.* (1998) Comprehensive identification of cell cycle-regulated genes of the yeast *Saccharomyces cerevisiae* by microarray hybridization. *Mol. Biol. Cell*, **9**, 3273–3297.