Subject Section

ATEN: And/Or Tree Ensemble for inferring accurate Boolean network topology and dynamics

Ning Shi, Zexuan Zhu, Ke Tang, David Parker and Shan He

1School of Computer Science, University of Birmingham, Birmingham, B15 2TT, UK,
2College of Computer Science and Software Engineering, Shenzhen University, Shenzhen, 518060, China,
3Department of Computer Science and Engineering, Southern University of Science and Technology, Shenzhen, 518055, China.

*To whom correspondence should be addressed.

Abstract

**Motivation:** Inferring gene regulatory networks from gene expression time series data is important for gaining insights into the complex processes of cell life. A popular approach is to infer Boolean networks. However, it is still a pressing open problem to infer accurate Boolean networks from experimental data that are typically short and noisy.

**Results:** To address the problem, we propose a Boolean network inference algorithm which is able to infer accurate Boolean network topology and dynamics from short and noisy time series data. The main idea is that, for each target gene, we use an And/Or tree ensemble algorithm to select prime implicants of which each is a conjunction of a set of input genes. The selected prime implicants are important features for predicting the states of the target gene. Using these important features we then infer the Boolean function of the target gene. Finally, the Boolean functions of all target genes are combined as a Boolean network. Using the data generated from artificial and real-world gene regulatory networks, we show that our algorithm can infer more accurate Boolean network topology and dynamics from short and noisy time series data than other algorithms. Our algorithm enables us to gain better insights into complex regulatory mechanisms of cell life.

**Availability:** Package ATEN is freely available at https://github.com/ningshi/ATEN

**Contact:** s.he@cs.bham.ac.uk

Supplementary information: Supplementary data are available at Bioinformatics online.

1 Introduction

Gene regulatory networks are important to elucidate complex processes of cell life (Karlebach and Shamir, 2008; Chai et al., 2014). A gene regulatory network is composed of interactions among proteins, genes and metabolites in living cells, and models the regulation of gene expression based on the relationships among genes and their products (de Jong, 2002; Hecker et al., 2009; Saadatpour and Albert, 2013; Chai et al., 2014). To infer gene regulatory networks, computational approaches such as Bayesian networks (Friedman et al., 2000; Sanchez-Castillo et al., 2017), ordinary differential equations (Bansal et al., 2007) and Boolean networks (Kaufman, 1969; Liang et al., 1998; Saadatpour and Albert, 2013) have been proposed.

In this paper, we focus on Boolean networks. Although Boolean networks are limited as the state of gene expression level can only be described using two states, they can capture some fundamental characteristics of signalling and gene regulatory mechanisms. They have been widely used to reveal both the gene regulatory network topology (i.e. the interactions between genes) and dynamics (i.e. the regulatory rules) (Lähdesmäki et al., 2003; Hecker et al., 2009; Gates and Rocha, 2016).

Most Boolean network inference algorithms focus on inferring network topology (Liang et al., 1998; Maucher et al., 2011; Pirki et al., 2015). There are only a few algorithms that can infer both the Boolean network topology and its dynamics. For example, the Best-Fit (Lähdesmäki et al., 2003) algorithm can find Boolean functions by solving the consistency problem (Akutsu, 1999). Barman and Kwon (2017) proposed a Boolean inference algorithm named MIBNI based on mutual information. The
algorithm REACT (Vera-Licona et al., 2014) uses Boolean polynomials to infer Boolean networks.

However, despite these advances, it is still challenging to infer accurate Boolean networks from experimental data, which is typically short and noisy (Bar-Joseph, 2004). To address this challenge, we propose an And/Or tree ensemble (ATEN) algorithm for inferring both the Boolean network topology and its dynamics. For each target gene, our ATEN algorithm uses an And/Or tree ensemble to select prime implicants, i.e., the conjunction (AND) of non-redundant input genes. These prime implicants are important features for predicting the states of the target gene. Using these important features, ATEN infers the Boolean function of the target gene. Finally, ATEN combines the Boolean functions of all target genes as a Boolean network.

The advantages of our ATEN algorithm are as follows. First, ATEN is efficient because And/Or trees provide an efficient representation for inferring prime implicants using binary time series data. Second, by using an ensemble algorithm, ATEN can better handle the "large p small n" problem (Genuer et al., 2010), i.e., the number of input genes p is larger than the number of time points n, to select more reliable important features (Huynh-Thu et al., 2010), i.e., prime implicants. Third, based on the selected features, our algorithm can infer a minimal Boolean function without redundant input genes, which results in a more accurate Boolean network.

To summarize, our main contributions are:

1. We propose the first And/Or tree ensemble algorithm. Although And/Or trees have been widely used to search for solutions in combinatorial and artificial intelligence (Marinescu and Dechter, 2009), no And/Or tree ensemble algorithm exists to search for robust consensus solutions. We are the first to bridge the gap between ensemble learning and And/Or trees.
2. We apply our And/Or tree ensemble algorithm to select important prime implicants as features, which is different from most existing algorithms, e.g., GENIE3 (Huynh-Thu et al., 2010) that select individual input genes for each target gene.
3. We validate ATEN on both artificial and real-world gene regulatory networks. Our experimental results show that ATEN compares favourably with existing algorithms in terms of prediction accuracy of network topology and dynamics, even when the gene expression time series data is short and noisy.

2 Methods

2.1 Preliminaries

2.1.1 Boolean network model

A Boolean network $G(X, F)$ is a set of nodes $X = \{x_1, ..., x_n\}$ representing genes and a set of Boolean functions $F = \{f_1, ..., f_n\}$ describing the regulatory rules between genes. Each node (i.e., gene) $x_i$ is a Boolean variable taking 1 or 0 to describe its state, where 1 represents that the gene is expressed and 0 represents that the gene is not expressed. The directed edges, or the interactions between the target gene and input genes are associated with the regulatory rule, i.e., the Boolean function. The Boolean function consists of three basic Boolean operators (i.e., $\lor$, $\land$, and $\neg$) and Boolean variables. The Boolean function $f_i$ determines the dynamical changes of the state of gene $x_i$ over time: $x_i(t + 1) = f_i(x_1(t), ..., x_n(t))$, where $x_i(t + 1)$ represents the state of gene $x_i$ at time point $t + 1$, and $x_i(t)$ represents the state of the $i$th input gene of gene $x_i$ at time point $t$. All genes update their states synchronously. For each gene in a Boolean network, the goal of the network inference problem is to identify the underlying Boolean function associated with input genes from the binary time series data by taking into account every time point.

To represent a Boolean function associated with input genes, we can use a logical formula in disjunctive normal form (DNF). A DNF formula is a disjunction (OR) of one or more implicants. An implicant $p$ is the conjunction of literals (i.e., Boolean variables, or their negations). An implicant $p$ is called a prime implicant if the deletion of any literal from $p$ results in a non-implicant. For example, a Boolean function $(x_1 \land x_2) \lor (x_1 \land \neg x_3 \land x_5)$ for the target gene $x_1$ represents two prime implicants $(x_1 \land x_2)$ and $(x_1 \land \neg x_3 \land x_5)$, where $x_1, x_2, x_3, x_4, x_5$ are input genes. We are interested in obtaining a minimal Boolean function which includes a minimum set of prime implicants by eliminating the redundant prime implicants. For example, a Boolean function $(\neg x_1 \land x_2 \land x_5) \lor (\neg x_1 \land x_2 \land \neg x_3) \lor (x_1 \land \neg x_3 \land x_4)$ can be minimized to $(\neg x_2 \lor x_1) \lor (x_1 \land \neg x_2 \land x_5)$.

2.1.2 And/Or Trees representation of logic functions

A logic function, or more specifically, a DNF formula can be represented as a three-level And/Or tree, which provides an efficient data structure to fit the binary time series data. In such a tree, an internal node represents a Boolean operator $\lor$ or $\land$, and a leaf denotes a Boolean variable or its negation. The output of a Boolean function is determined by the valuation of the And/Or tree with an assignment of "TRUE" or "FALSE" to each of the leaves. In Figure 1, we show an example of a three-level And/Or tree $l$ which is equivalent to the DNF formula $(x_1 \land x_2) \lor (x_1 \land \neg x_3 \land x_5)$.

We can represent a Boolean function in DNF as a three-level And/Or tree as follows:

- (First Level) The root is labeled by the Boolean operator $\lor$;
- (Second Level) The internal node is labeled by the Boolean operator $\land$;
- (Third Level) A leaf is labeled by a Boolean variable, or by a negation (i.e., a Boolean variable associated with $\neg$).

2.2 Algorithm

Algorithm 1. ATEN for Boolean network inference

Data: A binary time series dataset with $n$ genes.
Result: A Boolean network.

for each target gene $x_i$, $1 \leq i \leq n$ do
 chatter a set of prime implicants using Algorithm 3;
 chatter a set of important prime implicants using Algorithm 4;
 chatter the final associated Boolean function using Algorithm 2;
 Combine all Boolean functions of all target genes as a Boolean network;

2.2.1 Algorithm overview

To infer an accurate Boolean network from short and noisy gene expression time series data, our And/Or tree ensemble (ATEN) algorithm consists of three key steps as summarized in Algorithm 1. For each target gene, the first step is to infer a set of prime implicants based on an And/Or tree ensemble. Each tree in the ensemble represents a putative Boolean function of the target gene. The second step is to select a set of prime implicants as important features using a recursive feature construction and elimination
2.2.2 And/Or tree inference
We summarize the And/Or tree inference algorithm in Algorithm 2. Given binary time series data including $N$ time points and $n$ genes, ATEN infers an And/Or tree that best predicts the state of the target gene $x_i, 1 \leq i \leq n$. The inference is achieved by searching the optimal input features (i.e., genes or prime implicants) and Boolean operators of the And/Or tree using a simulated annealing (SA) algorithm (van Laarhoven and Aarts, 1987).

Specifically, the And/Or tree of a target gene is initialized with an input feature (e.g., a random gene, or a random prime implicant, or known prime implicants). Then at each iteration $k$ of the SA algorithm, a new tree $I_k$ is proposed by randomly executing a permissible move based on the old tree $I_{k-1}$, e.g., adding a leaf, deleting a leaf, etc (for the fully permissible move set, see the Supplementary Data). The acceptance probability for the new tree $I_k$ is determined by the misclassification rates of $I_{k-1}$ and $I_k$ and the temperature $T$ specified in the SA cooling scheme, i.e., $Pr_{I_k} = \min(1, \exp((E_{I_{k-1}} - E_{I_k})/T))$, where $E_{I_{k-1}}$ and $E_{I_k}$ are the misclassification rates of $I_{k-1}$ and $I_k$ respectively. The misclassification rate $E_{I_k}$ of $I_k$ is calculated by

$$E_{I_k} = \frac{1}{N-1} \sum_{i=2}^{N} I(y_i(t-1) \neq x_i(t)),$$

where $I(\cdot)$ is an indicator function $I(x) = \begin{cases} 1, & \text{if } y_i(t-1) \neq x_i(t) \\ 0, & \text{if } y_i(t-1) = x_i(t) \end{cases}$, and $y_i(t-1)$ is the output state predicted by $I_k$ based on the states of the input genes at time point $t-1$ and $x_i(t)$ is the true state of the target gene at time point $t, 2 \leq t \leq N$. The acceptance probability indicates: 1) if $I_k$ is better than $I_{k-1}$ (i.e., $E_{I_k} < E_{I_{k-1}}$), then $I_k$ is always accepted; 2) if $I_k$ is not better than $I_{k-1}$ (i.e., $E_{I_k} \geq E_{I_{k-1}}$), then $I_k$ is accepted with a very small probability which gradually converges to zero as the temperature decreases. The maximum number of iterations is predefined to stop the searching scheme. The tree size which is defined as the number of input genes in the tree can be set according to prior knowledge, i.e., the maximum in-degree of the network, or determined experimentally or empirically. In addition, if the interactions between some pairs of nodes are known, ATEN can include those interactions as fixed prime implicants when inferring the tree.

Algorithm 2: And/Or tree inference for a target gene

**Data:** Binary time series data.

**Result:** An And/Or tree $I_k$ of the target gene.

$k = 0$;
Initialize an And/Or tree $I_0$ using an input feature;
repeat
$k \leftarrow k + 1$;
$I_k \leftarrow$ Update $I_{k-1}$ to propose a new tree;
if $I_k$ is accepted according to the probability $Pr_{I_k}$ then
$\{ I_k \leftarrow I_{k-1} \}$;
until $k$ reaches the maximum iteration;

The finally accepted And/Or tree may contain redundant regulatory interactions (i.e., prime implicants) between the target gene and input genes that are unnecessary for predicting the state of the target gene. Thus we introduce a Boolean minimization method aimed at obtaining a minimal And/Or tree (i.e., a minimal DNF) based on the Quine-McCluskey algorithm (Quine, 1955; McCluskey, 1956), where the Quine-McCluskey algorithm tries to find essential prime implicants and other necessary prime implicants to simplify Boolean functions.

Algorithm 3: Compute the importance of prime implicants of a target gene

**Data:** A binary time series dataset $D$.

**Result:** The prime implicants with their importances

**Initialization:** The number of bootstrap samples $B$;
Generate bootstrap samples $S_b$ by drawing $B$ samples;
foreach $S_b \in B, 1 \leq b \leq B$ do
$I_b \leftarrow$ Infer an And/Or tree using Algorithm 2;
$O_b \leftarrow$ Generate an oob sample;
$PI_i \leftarrow$ Extract all distinct prime implicants $p_i, i = 1, 2, \ldots$;
foreach $p_i \in PI$ do
foreach $I_b$ do
$E_{I_b} \leftarrow$ Compute the misclassification of $I_b$ using $O_b$;
$E_{I_b}^{\mathcal{B}_b}/E_{O_b} \leftarrow$ Remove/Append $p_i$ from/oob $I_b$;
$E_{I_b}^{\mathcal{B}_b}/E_{O_b} \leftarrow$ Compute the misclassification of $I_b^{\mathcal{B}_b}$ using $O_b$;
Determine the importance of $p_i$ by $VI(p_i) = \sum_{b=1}^B \left( \sum_{p_i \in I_b} (E_{I_b}^{\mathcal{B}_b} - E_{I_b}^{\mathcal{B}_b}/E_{O_b}) + \sum_{p_i \notin I_b} (E_{I_b}^{\mathcal{B}_b} - E_{I_b}^{\mathcal{B}_b}/E_{O_b}) \right)$.

2.2.3 Prime implicant importance measure
We compute the importances of prime implicants as shown in Algorithm 3 based on logisticFS (Ruczinski et al., 2003; Schwender and Ickstadt, 2008). The main idea is to construct an And/Or tree ensemble to quantify how much a prime implicant improves the predictive accuracy of the state of the target gene. As shown in Figure 2, given a binary time series dataset $D$, for a target gene, we draw $B$ pairs of a bootstrap sample $S_b$ and an out-of-bag (oob) sample $O_b, 1 \leq b \leq B$ for inferring ensembles of trees and computing the importances of prime implicants, respectively.

Formally, given $J$ time series covering $N$ time points each, the $j$th ($1 \leq j \leq J$) time series can be presented as a matrix of states of all genes, denoted as $X_j = (x^j(t_1), \ldots, x^j(t_n), \ldots, x^j(N))$, where $x^j(t) = (x^j(t_1), x^j(t_2), \ldots, x^j(t_n))^T$ is a vector containing the states of all $n$ genes at time point $t$. We then construct the binary time series dataset matrix $D$ by concatenating the $J$ time series, i.e.,

$D = \{ X^j_1, \ldots, X^j_j, \ldots, X^j_J \}$.  


implicant. In the end, the importance of each prime implicant is evaluated to recursively reconstruct the selected important prime implicants, and eliminate the non-important ones. We summarize RFRE in Algorithm 4.

2.2.4 Recursive feature reconstruction and elimination

We propose a recursive feature reconstruction and elimination (RFRE) procedure for selecting a set of important prime implicants as features, which will be used for inferring Boolean functions. The main idea is to recursively reconstruct the selected important prime implicants, and eliminate the non-important ones. We summarize RFRE in Algorithm 4.

Initially, we apply Algorithm 3 to a binary time series dataset matrix \( D \) containing \( J \) time series covering \( N \) time points each, which results in a set of prime implicants \( PL_0 \) with their importances. We then eliminate the least important prime implicants, i.e., prime implicants with negative importance and the top 20% of prime implicants with the lowest importances.

Algorithm 4: RFRE for prime implicants selection

**Data:** A binary time series dataset \( D \).

**Result:** A small set of prime implicants.

1. **Initialization:**
   - \( PL_0 \leftarrow \text{Apply Algorithm 3 to } D \) and eliminate the non-important prime implicants
   - \( r = 1 \);

2. **Repeat**
   - \( D_{r-1} \leftarrow \text{Update a dataset matrix based on } PL_{r-1} \)
   - \( PL_r \leftarrow \text{Apply Algorithm 3 to } D_{r-1} \) and eliminate the non-important prime implicants
   - \( PL_{add} \leftarrow PL_r \setminus PL_{r-1} \)
   - \( PL_{del} \leftarrow PL_r \setminus PL_{add} \)
   - \( r \leftarrow r + 1 \)
   - until \( V I(PL_r) > V I(PL_{r-1}) \), \( \forall p_u \in PL_{add}, \forall p_w \in PL_{del} \);

Then at each recursion, we first update the time series dataset matrix \( D_{r-1} \) based on the important prime implicants:

\[
D_{r-1} = \{ P_{j-1}^1, \ldots, P_{j-1}^j, \ldots, P_{j-1}^T \},
\]

where \( P_{j-1}^t \) denotes the \( j \)th time series dataset matrix of all prime implicants, i.e., \( P_{j-1}^t = \{ P_{j-1}^t(1), \ldots, P_{j-1}^t(1), \ldots, P_{j-1}^t(N) \} \). The whole network of a single parasegment primordium which consists of 32 nodes (not shown in Figure 3) in the adjacent cells. In the experiment, the whole network of a single parasegment primordium which consists of 154+80 nodes was used for network inference.

2.3 Datasets

2.3.1 Artificial network

We first validated the performance of ATEN by inferring three artificial Boolean networks generated by BoodNet (Mussel et al., 2010), where three networks include 50 nodes, 100 nodes, and 150 nodes, respectively. The maximum in-degree of each network is 5. For each network we generated 12 datasets \( D_i, 1 \leq i \leq 12 \) where each dataset includes 10 time series. Each time series in \( D_i = \{ 1, 2, 3 \}, D_i(1, 4, 5, 6), D_i(1, 7, 8, 9) \) and \( D_i(10, 11, 12) \) has 5, 10, 15, 20 time points, respectively. Additionally, we added 1% and 5% noise to \( D_i(i = 2, 5, 8, 11) \) and \( D_i(i = 3, 6, 9, 12) \). The noise was introduced by randomly flipping the state of each node with the probability of 1% or 5%.

2.3.2 Real-world network

We also applied ATEN to infer a real-world gene regulatory network using experimentally-observed gene expression data. The network is derived from a well-studied Drosophila segment polarity gene regulatory network (Albert and Othmer, 2003) and modified by REACT (Vera-Licona et al., 2014). It is considered as a single parasegment primordium of four identical cells (Albert and Othmer, 2003; Laidenbacher and Stigler, 2004). As shown in Figure 3 and Supplementary Data Table 12, we present part of the network topology and its dynamic model (i.e., Boolean functions), respectively. The network of one cell consists of 15 nodes and 6 extracellular nodes, where the 6 extracellular nodes are connected with other nodes (not shown in Figure 3) in the adjacent cells. In the experiment, the whole network of a single parasegment primordium which consists of 154+80 nodes was used for network inference.

For the dataset used for inference, we introduced three datasets that were used in REACT. The datasets were initialized using experimentally observed gene expression data (Albert and Othmer, 2003) and knockout perturbations. These three datasets included 0%, 1%, 5% data noise, respectively. Each dataset has 24-time series containing 202 time points in total. Moreover, we used the same initialization to generate 9 additional datasets \( D_j, 1 \leq j \leq 9 \), where each dataset has 10 time series. Each time series in \( D_j = \{ 1, 2, 3 \}, D_j(4, 5, 6) \) and \( D_j(7, 8, 9) \) has 5, 10, 15 time points, respectively. We also added 1% and 5% noise into \( D_j(i = 2, 5, 8) \) and \( D_j(i = 3, 6, 9) \), respectively.

2.4 Computational experiments

We compared ATEN with the other three existing network inference algorithms, i.e., MIBNI (Barrick and Kwon, 2017), Best-Fit (Ladhensnaki et al., 2003) and REACT (Vera-Licona et al., 2014). Because ATEN uses a
heuristic algorithm, we executed ATEN 30 times with $B = 100$ on every dataset. Due to the unavailable implementation of REACT, we adopted the results from the literature (i.e., Vera-Licona et al. (2014)) for comparison. We present the details of computing resources in the Supplementary Data. For inferring the artificial network, we compared the performance of ATEN with that of Best-Fit and MIBNI. For inferring the real-world gene regulatory network, we compared ATEN with MIBNI, Best-Fit and REACT.

One important control parameter of ATEN is the upper limit on tree size, i.e., the maximum number of input genes of a target gene. This parameter essentially defines the search space of the problem, which affects not only the prediction accuracy but also the computational time of ATEN. However, finding a suitable value of this parameter is challenging not only for ATEN but also for other algorithms (Barman and Kwon, 2017). To determine the value of this parameter, we executed experiments on an artificial network with 150 nodes (See Supplementary Data). Based on the experimental results, we set the upper limit on tree size to 8. For fair comparison, MIBNI and Best-Fit were limited to infer up to 8 input genes for each target gene. Additionally, we also presented the results with known maximum number of input genes of the artificial and real-world networks in the Supplementary Data.

### Performance matrices

To quantify the quality of the inferred network topology, we used the performance metrics used in Maucher et al. (2011), namely, Recall = |TP|/|TP + FN|, False Positive Rate (FPR) = |FP|/|FP + TN|, and F-score = 2TP/(2TP + FP + FN), where TP denotes the number of inferred interactions that are present in the ground truth network; FP denotes the number of inferred interactions that are absent in the ground truth network; TN denotes the number of absent interactions in the inferred network that are also absent in the ground truth network; FN denotes the number of absent edges in the inferred network that are present in the ground truth network.

To evaluate the quality of the dynamic model of the inferred networks, we used 6 steady states that have been investigated in existing literature as detailed in (Vera-Licona et al., 2014) and Supplementary Data Table 13. We checked whether the Boolean functions inferred by each algorithm can retrieve these steady states.

### 3 Results

#### 3.1 Performance on artificial network

We summarize experimental results from ATEN, Best-Fit and MIBNI on the artificial networks with 100 nodes inferred from noisy data (with 1% and 5% noise) where we set the upper limit on tree size to 8 in Table 1 (The results from the noise-free data are detailed in Supplementary Data Table 3.). We summarize the results of inferring the networks with 50 and 150 nodes without known maximum in-degree in Supplementary Data Tables 1-2 and 4-5. In addition, the results of inferring all artificial networks using known maximum in-degree are detailed in Supplementary Data Tables 6-11.

From Table 1, we can see that ATEN always obtained the best Recall, FPR and F-score when the datasets are noisy (i.e., 1% and 5% noise). We can also see that since Best-Fit uses exhaustive search, it cannot successfully infer the networks from the datasets including 200 time points with 1% and 5% noise within the walltime. When the data is noise-free, from Supplementary Data Table 3, Best-Fit outperformed than other algorithms. For the networks consisting of 50 and 150 nodes, from Supplementary Data Tables 1-2 and 4-5, we can observe that ATEN inferred the best results from noisy data and Best-Fit obtained the best results from noise-free data.

#### 3.2 Performance on real-world network

We used the real-world gene regulatory network to evaluate ATEN’s performance of inferring network topology and its dynamic model (i.e., Boolean functions), in comparison with other three algorithms (i.e., MIBNI, Best-Fit and REACT).

#### 3.2.1 Inference of network topology

Overall, ATEN outperformed the other three algorithms for inferring the network topology from short and noisy data. We present their performance among the datasets including 1% and 5% noise with setting the maximum in-degree of target genes to 8 in Figure 4. The results from the noise-free data are detailed in Supplementary Data Table 14.) We also discuss the experimental results where we set the upper limit on tree size to 4, the exact maximum in-degree of the network.

In Figure 4, the average and standard deviation values of Recall and FPR obtained from ATEN were used for comparison. From Figure 4, among all these four algorithms ATEN always obtained the best FPR and F-score values. Compared with the performance of MIBNI, it can be seen

---

Table 1. ATEN inferred more accurate artificial network topology from short and noisy data. Experimental results of ATEN, MIBNI and Best-Fit for the networks with 100 nodes inferred from the datasets including 1% and 5% noise. The maximum in-degree of each target gene inferred by ATEN, MIBNI and Best-fit was limited to 8. The results of ATEN are presented using average and standard deviation values. The best result among three algorithms is highlighted in boldface.

<table>
<thead>
<tr>
<th>Number of Time Points</th>
<th>Algorithm</th>
<th>Noise 1%</th>
<th>Noise 5%</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Recall</td>
<td>FPR</td>
<td>F-score</td>
</tr>
<tr>
<td>50</td>
<td>ATEN</td>
<td>0.899±0.010</td>
<td>0.007±0.001</td>
</tr>
<tr>
<td></td>
<td>MIBNI</td>
<td>0.758</td>
<td>0.071</td>
</tr>
<tr>
<td></td>
<td>Best-Fit</td>
<td>0.797</td>
<td>0.018</td>
</tr>
<tr>
<td>100</td>
<td>ATEN</td>
<td>0.950±0.010</td>
<td>0.007±0.001</td>
</tr>
<tr>
<td></td>
<td>MIBNI</td>
<td>0.773</td>
<td>0.071</td>
</tr>
<tr>
<td></td>
<td>Best-Fit</td>
<td>0.906</td>
<td>0.027</td>
</tr>
<tr>
<td>150</td>
<td>ATEN</td>
<td>0.953±0.010</td>
<td>0.008±0.001</td>
</tr>
<tr>
<td></td>
<td>MIBNI</td>
<td>0.758</td>
<td>0.071</td>
</tr>
<tr>
<td></td>
<td>Best-Fit</td>
<td>0.930</td>
<td>0.029</td>
</tr>
<tr>
<td>200</td>
<td>ATEN</td>
<td>0.901±0.009</td>
<td>0.007±0.001</td>
</tr>
<tr>
<td></td>
<td>MIBNI</td>
<td>0.766</td>
<td>0.071</td>
</tr>
<tr>
<td></td>
<td>Best-Fit</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>
that ATEN was significantly better. With respect to the network inferred from the dataset including 50 time points and 1% noise, ATEN achieved similar experimental results with Best-Fit. However, when the datasets including 5% noise, we can find that ATEN achieved significantly better recall, FPR and F-score values than Best-Fit. Compared with the best recall and FPR values of Best-Fit, ATEN achieved similar but more stable results. Moreover, from Figure 4, it can be concluded that ATEN outperformed the other three algorithms when the data was shorter (i.e., 50 time points) with higher noise level (i.e., 5% noise). When we inferred the network from noise-free data (see Supplementary Data Table 14), Best-Fit achieved better results than other algorithms.

3.2.2 Inference of dynamic model
The overall performance of ATEN for predicting the dynamic model was better than MIBNI, Best-Fit and similar to REACT. Table 2 presents the number of steady states that can be retrieved by ATEN and other algorithms when the maximum in-degree of inferred network is unknown.

From Table 2, We can observe that ATEN always retrieved at least 50% steady states even when the input data was short and noisy (i.e., 50 time points and 5% noise). The MIBNI did not find any steady states, since MIBNI can infer only conjunctive or disjunctive Boolean functions. Compared with Best-Fit and MIBNI, we can see that ATEN was significantly better than both the compared algorithms among all datasets. Compared with REACT, ATEN correctly retrieved the same number of steady states. The experimental results also show that ATEN was more robust to different levels of noise and different numbers of time points for inferring the network dynamic model.

4 Discussion
Inferring topological structure and network dynamics of gene regulatory networks from gene expression time series data is a major challenge in systems biology. To tackle this challenge, we propose ATEN, which is the first algorithm that combines ensemble learning and And/Or trees.

Our experimental results (Tables 1 and 2, Supplementary Data Tables 2, 5, 7, 9, 11 16 and 17) show that, when the data was noisy, ATEN performed better than the other three algorithms in terms of the prediction accuracy. When the data was noise-free, Best-Fit achieved the best results. We therefore recommend using ATEN for inferring Boolean networks from noisy data, while Best-Fit for noise-free data.

We used Simulated Annealing algorithm to infer And/Or trees because of its simplicity and good search performance. Any other heuristic algorithms such as Genetic Algorithms could be suitable for this task. It would be interesting to investigate whether other heuristic algorithms could improve the performance. Another reason that we used heuristic algorithm is because it is faster than exhaustive search method (e.g. Best-Fit). However it cannot guarantee that the inferred network is optimal. To find the optimal solution, it is possible to introduce mathematical programming (Knijnenburg et al., 2016) into ATEN.

ATEN has been evaluated on Boolean networks with synchronous updating scheme due to its computational efficiency. However, asynchronous updates is more biological plausible although the inference is more difficult. It will be our future work to utilize Boolean state space scoring function (Lim et al., 2016) for inferring asynchronous Boolean networks.

Another interesting direction is to extend ATEN to infer Probabilistic Boolean networks (PBNs) (Shmulevich et al., 2002a,b) which improves Boolean networks for handling noise and uncertainty. The PBN allows each target gene to be associated with multiple Boolean functions with corresponding selection probabilities. We expect to extend our ATEN algorithm to infer probabilistic Boolean networks from the gene expression time series data in the future.

Acknowledgements
Funding
This work has been supported by the

Conflict of Interest: none declared

References

Table 2. ATEN inferred more accurate real-world dynamic model from short and noisy data. The maximum in-degree of each target gene inferred by ATEN. MBNI and Best-fit was limited to 8. The numbers indicate the number of steady states that was successfully retrieved. The best result among three algorithms is highlighted in boldface. Note that the experimental results of REACT was collected from Vera-Licona et al. (2014) and the symbol ‘*’ indicates that the corresponding results cannot be collected.

<table>
<thead>
<tr>
<th>Number of Time Points</th>
<th>Algorithms</th>
<th>Noise 0%</th>
<th>1% 5%</th>
<th>Noise 0%</th>
<th>1% 5%</th>
<th>Noise 0%</th>
<th>1% 5%</th>
<th>Noise 0%</th>
<th>1% 5%</th>
</tr>
</thead>
<tbody>
<tr>
<td>50</td>
<td>ATEN</td>
<td>3 4</td>
<td>5 5</td>
<td>4 4</td>
<td>5 5</td>
<td>4 4</td>
<td>5 5</td>
<td>4 4</td>
<td>5 5</td>
</tr>
<tr>
<td></td>
<td>MIBNI</td>
<td>0 0 2</td>
<td>0 0 0</td>
<td>0 0 0</td>
<td>0 0 0</td>
<td>0 0 0</td>
<td>0 0 0</td>
<td>0 0 0</td>
<td>0 0 0</td>
</tr>
<tr>
<td></td>
<td>Best-Fit</td>
<td>3 4</td>
<td>2 1 1</td>
<td>4 5 2</td>
<td>2 2</td>
<td>5 2 2</td>
<td>5 2 2</td>
<td>5 2 2</td>
<td>5 2 2</td>
</tr>
<tr>
<td></td>
<td>REACT</td>
<td>* * *</td>
<td>* * *</td>
<td>* * *</td>
<td>* * *</td>
<td>* * *</td>
<td>* * *</td>
<td>* * *</td>
<td>* * *</td>
</tr>
</tbody>
</table>

“output” — 2019/7/6 — page 6 — #6
Fig. 4. ATEN inferred more accurate real-world network topology from short and noisy data. Experimental results of ATEN, MIBNI, Best-Fit and REACT for the inferred network from the datasets including 1% noise and 5% noise. The maximum in-degree of each target gene inferred by ATEN, MIBNI and Best-fit was limited to 8. The results of ATEN are presented using average and standard deviation values. The experimental results of REACT, i.e., the maximum and minimum Recall and FPR values were collected from the literature (i.e., Vera-Licona et al. (2014)), and the F-score values cannot be collected.


