BNL-212374-2019-COPA

Bi-Directional Causal Graph Learning through Weight-sharing and Low-rank Neural Network

H. Huang, C. Xu

Submitted to the IEEE International Conference on Data Mining (ICDM) 2019 Conference
to be held at Beijing China
November 08 - 11, 2019

Computational Science Initiative

Brookhaven National Laboratory

U.S. Department of Energy

USDOE Office of Science (SC), Advanced Scientific Computing Research (SC-21)

DISCLAIMER

Bi-Directional Causal Graph Learning through Weight-sharing and Low-rank Neural Network

Hao Huang*, Chenxiao Xu† and Shinjae Yoo‡
* *GE Global Research*, San Ramon, CA
haohuangcssbu@gmail.com
† Stony Brook University, Stony Brook, NY
chenxiao.xu@stonybrook.edu
‡ *Brookhaven National Laboratory*, Upton, NY
sjyoo@bnl.gov

***Abstract*—Discovering the causal graph in multivariate time series data is of great importance for industrial society, yet challenging due to the unknown nonlinearity in the data. Existing works only explore the data in chronological order, and rely on pre-assumed kernels or certain distribution assumption. In this paper, we present a Bi-directional neural network for Causal Graph Learning (Bi-CGL) through weight-sharing and low-rank neural network. It discovers the causal graph by simultaneously exploring input in forward and reverse chronological order. Both directions approach the same causal graph with shared low-rank approximation, which provides robustness and better accuracy against data noise. Experiments on synthetic and real world datasets prove our Bi-CGL's outperformance over existing baselines.**

***Index Terms*—causal graph learning, bi-directional neural network, weight-sharing**

Fig. 1. Retrocausality and causality.

I. Introduction

Imagine you travel to 2050 with a time machine. When the door is open you see the following scene: houses are all torn apart, smashed cars and fallen trees are everywhere. You also notice the cracks on the ground in front of you. You may probably draw a conclusion that there was an earthquake just occurred some time ago. But do you notice that you just utilize a relationship from the effect to the cause, or retrocausality (Figure 1)? In fact, most physical laws are 'time reversible', which means that they would work just as well if time was defined as running backwards [35], [10]. Reasoning from the future to the past can help us discover the underlying causality, which is an important analysis task to gain insight into the underlying mechanisms of the targeted system.

Learning causal graph is of great importance to many modern applications such as social networks, biological gene regulatory networks, advertising and marketing, etc.. It is also very meaningful for developing Explainable Artificial Intelligence (XAI) [12], which explains models' decisions and actions to human users. In this research we will explore the underlying causal graph of multivariate time series data from both forward and backward directions in time.

Causal graph is a qualitative measurement about the direct causal relations among a set of nodes, where each node represents a variable or component of the targeted system. The edges are all directed, and each describes the causality between the two connected nodes. In many cases, the causal graphs are unknown or partially known. Learning such a graph structure is full of challenges: 1) the relationships between variables are unknown and complex (usually nonlinear), 2) there is no label available in learning causality and 3) real world dataset is usually full of noise.

Existing causal graph learning methods are either based on linear models (VAR Granger analysis, Generalized Additive Models) or certain pre-assumed distributions [28], [4]. However, many data are nonlinear and with unknown distribution [23], [7]. Selecting appropriate kernels or distribution models for each time series requires a deep understanding of the domain knowledge and in many cases not even possible. Moreover, these methods' capability of learning causal graph is highly affected by the data noise, system diversity and scales (i.e. number of nodes) [14], [33], [27].

Although deep learning techniques have become increasingly popular in many applications, there are few research about how deep learning can contribute on learning causal graph given multivariate time series data as input. In this research we propose a **Bi**-directional neural network for **C**ausal **G**raph **L**earning, or ***Bi-CGL***. The design of our model is shown in Figure 2. *Bi-CGL* is capable to discover nonlinear causality between any pair of nodes (input variables), without prior knowledge on generalization rules (e.g. pre-assumed kernel or distribution assumption). The causal graph is reconstructed simultaneously from forward and backward directions in time, with a weight-sharing low-rank approximation that provides better accuracy and robustness against data noise.

Our contributions are listed as follows:

1) *Bi-CGL* is the first bi-directional neural network to dis-

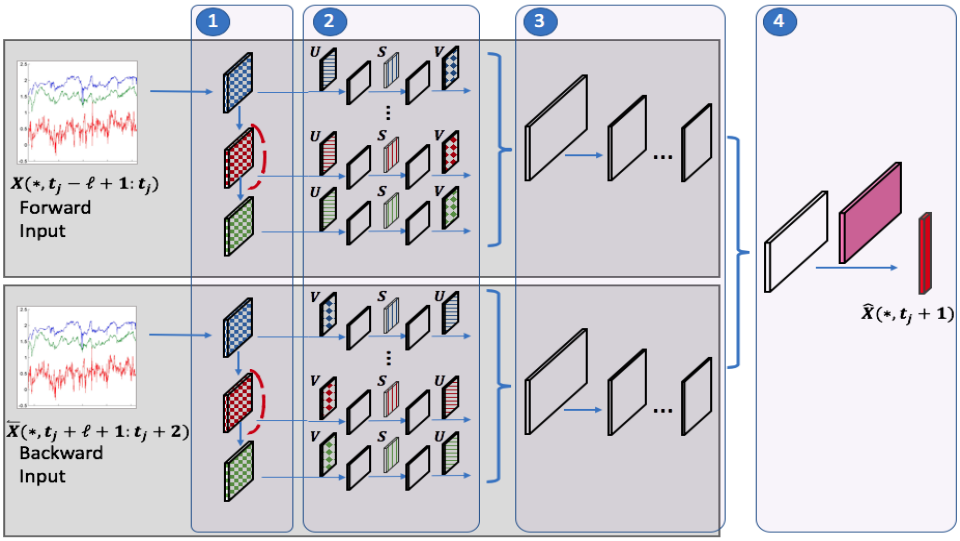


Fig. 2. The framework of our *Bi-CGL*, the layers with the same color and texture are weight-sharing. *Bi-CGL* learns causal graph through regression setting with two directional inputs and the same target. It consists of four modules. Module 1 is a bidirectional temporal convolution module that learns the univariate nonlinearity on different time-lags through forward and backward directions. Module 2 is to discover the underlying causal graph through weight-sharing low-rank approximations. Each path in forward direction shares the same low-rank chain (U, S and V) with the corresponding path in backward direction, but with reverse order. Module 3 is to prospect intervariable nonlinearity, while Module 4 is the regression module.

cover causal graph with better accuracy.

2) *Bi-CGL* is a weight-sharing and low-rank neural network, which is robust to data noise.
3) *Bi-CGL* can explore nonlinear causality without pre-assuming any kernel or distribution of the dataset.

II. Preliminaries

A. Problem Definition

In this section, we mathematically describe our problem setting. We start off by formally defining our notation.

The input of our *Bi-CGL* is the historical data of m time series (normalized) with time length n, i.e., $X = \{X(1,*), X(2,*), ..., X(m,*)\} \in \mathbb{R}^{m\times n}$, where each $X(i,*) \in \mathbb{R}^{1\times n}$ is a single time series associated to the i-th variable. We denote $X(i,t_j)$ as the value of the i-th time series at time t_j, and $X(i, t_j-\ell+1 : t_j) \in \mathbb{R}^{1\times \ell}$ as the values of the i-th time series from time $t_j - \ell + 1$ to t_j (in forward chronological order). The same data in reverse chronological order is noted as $\overleftarrow{X}(i, t_j : t_j - \ell + 1) \in \mathbb{R}^{1\times \ell}$.

Moreover, $X(*,t_j) = \{X(1,t_j), X(2,t_j), ..., X(m,t_j)\} \in \mathbb{R}^{m\times 1}$ are the values of all time series at time t_j, and $X(*, t_j-\ell+1 : t_j) = \{X(1, t_j-\ell+1 : t_j), X(2, t_j-\ell+1 : t_j), ..., X(m, t_j-\ell+1 : t_j)\} \in \mathbb{R}^{m\times \ell}$ are the values of all time series from time $t_j-\ell+1$ to t_j. The same data in reverse chronological order are denoted as $\overleftarrow{X}(*, t_j : t_j - \ell + 1)$

Our goal is to learn the underlying causal graph $A \in \mathbb{R}^{m\times m}$. $A(i,j)$ with absolute value much larger than zero denotes that the i-th time series is among the causes of the j-th time series in time. We use $A(*,i) \in \mathbb{R}^{m\times 1}$ to represent the i-th column of A, which tells the causes of the i-th time series.

B. Multivariate Granger Causality

Causal graph learning with multivariate time series roots from Granger causality analysis, which is performed by fitting a vector autoregressive model (VAR) to the time series input [21]. VAR tries to predict $X(*, t_j+1)$ with $X(*, t_j-\ell+1 : t_j)$:

$$X(*, t_j + 1) = \sum_{r=1}^{\ell} A_r X(*, t_j + 1 - r) + \epsilon(*, t_j + 1), \quad (1)$$

where $\epsilon(*, t_j + 1) \in \mathbb{R}^{m\times 1}$ is a white Gaussian noise at time $t_j + 1$, ℓ is the temporal order (the involving time lags) and A_r is the causal graph for time lag r. Time series $X(i,*)$ is called a Granger cause of time series $X(j,*)$ if at least one of the elements (in absolute value) $A_r(i,j)$, where $r = 1, 2, ..., \ell$, is significantly larger than zero [21].

In [6], bi-directional Granger causality model is built upon two VARs in forward and backward directions respectively:

$$X(*, t_j + 1) = \sum_{r=1}^{\ell} A_r^f X(*, t_j + 1 - r) + \epsilon(*, t_j + 1),$$
$$X(*, t_j + 1) = \sum_{r=1}^{\ell} A_r^b \overleftarrow{X}(*, t_j + 1 + r) + \epsilon(*, t_j + 1), \quad (2)$$

where A_r^f (A_r^b) is the forward causal graph for time lag r. The way they calculate causal graph A_r is as follows:

$$A_r = \frac{1}{2}(A_r^f + (A_r^b)^T), \quad \text{for } r = 1, 2, ..., \ell \quad (3)$$

where $(A_r^b)^T$ denotes the matrix transpose of A_b^r. However, all these methods are purely linear and fail to capture any nonlinear causality [9]. Our objective in this research is to discover more general and complex causality. To this end,

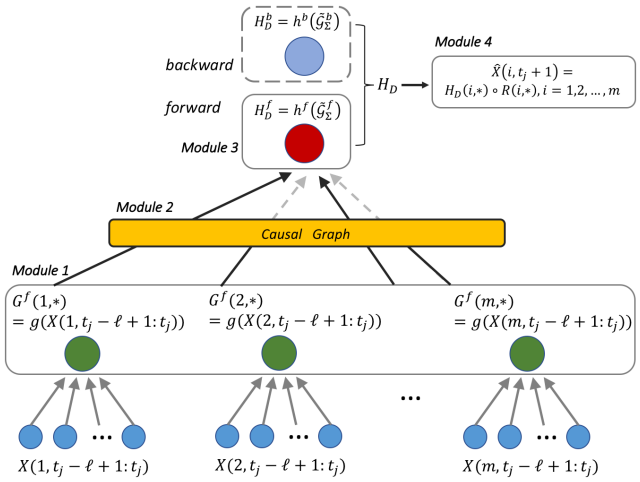


Fig. 3. This figure illustrates the connections of the four modules in Figure 2. The forward direction input (the bottom blue dots) is fed into Module 1, which learns function $g(*)$ that provides the set of univariate nonlinearity among different time lags in each time series. Then causal graph is learned in Module 2 which selects those relevant univariate nonlinearity $\tilde{\mathcal{G}}_{\Sigma}^{f}$. Module 3 learns function $h^f(*)$ that acquires multivariate nonlinearity H_D^f based on $\tilde{\mathcal{G}}_{\Sigma}^{f}$. On the other hand, H_D^b is learned in a similar way (through Module 1 to 3) with input from backward direction. Finally, Module 4 estimates the regression targets with input H_D which is the concatenation of H_D^f (forward) and H_D^b (backward).

we design a deep neural network, which does not involve any kernel assumption like Equation (1) and (2). Furthermore, we only focus on learning one global causal graph A in this research.

III. Our Proposed Model

We approach the underlying causal graph by exploring both forward and backward directions in time. Specifically, we learn the causal graph A through a bi-directional regression setting with two inputs and one output: the forward regression is to regress $X(*, t_j + 1)$ with $X(*, t_j - \ell + 1 : t_j)$, while the backward regression is to regress the same target with $\overleftarrow{X}(*, t_j + \ell + 1 : t_j + 2)$. The underlying causal graph A is approached through a combined learning of forward directional causality and the retrocausality from backward direction. Specifically, the retrocausality is the transpose of causality, after switching the roles of causes and effects.

To learn nonlinear causality in a systematic way, we separate the data nonlinearity into two types: temporal and intervariable nonlinearity. Temporal nonlinearity is explored on univariate level, which describes the nonlinearity between time lags in each time series. Intervariable nonlinearity can be treated as an extension of temporal nonlinearity to multivariate level, which involves different time series.

Figure 3 shows how we separate and learn the two types of nonlinearity, and their connections with the causal graph learning and the final regression. Here we start from the forward directional input first, and the backward input will be analyzed in a similar way. The raw input from each time series on forward direction is denoted as the bottom blue layer. Module 1 learns a nonlinear function $g(*)$ that represents all the temporal nonlinear transformations on each time series from forward direction, i.e. $X(i, t_j - \ell + 1 : t_j)$. To put it another way, $G^f(i, *)$ includes the nonlinear combination between different time lags in $X(i, *)$. Module 2 learns a causal graph, which selects the nonlinear variables relevant to regressing each $X(i, *)$. Module 3 learns a nonlinear function $h^f(*)$ that rolls all the selected time-series $\tilde{\mathcal{G}}_{\Sigma}^{f}$, and returns a more complex nonlinearity set H_D^f in-between time series. Similarly, H_D^b which represents the nonlinearity from backward directional input can be learned from $\overleftarrow{X}(*, t_j+\ell+1 : t_j + 2)$ with Module 1 to 3. Finally the model concatenates H_D^f (forward) and H_D^b (backward) into H_D, which is used in Module 4 to estimate the regression target $X(*, t_j + 1)$.

In the following subsections we will describe our proposed four modules one by one. Section III-A introduces Module 1 that is to explore the temporal nonlinearity. Section III-B focuses on Module 2 that aims to learn causal graph. Section III-C talks about Module 3 that prospects intervariable nonlinearity between time series, while Module 4 is explained in Section III-B for the final regression.

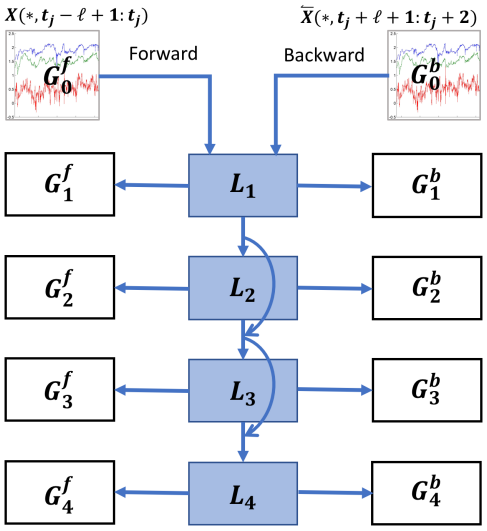


Fig. 4. Module 1 is ResNet with bi-directional input (here use four layers for example). G_q^f (G_q^b) indicates the forward (backward) output on layer q.

A. Module 1: Learning Temporal Nonlinearity

Module 1 applies a bi-directional residual neural network (ResNet) [13]. The aim of using ResNet is to learn the temporal nonlinearity with deep yet easily trainable architecture while avoiding vanishing gradient.

The temporal nonlinearity of both forward and backward inputs is learned through the same ResNet as shown in Figure 4. We introduce the structure by focusing on forward directional learning first. By setting $G_0^f = X(*, t_j - l + 1 : t_j)$, our ResNet is designed as:

$$G_q^f = ReLU(L_q(G_{q-1}^f) + id(G_{q-1}^f)), \quad q = 1, 2, ..., Q, \quad (4)$$

where $G_q^f \in \mathbb{R}^{m\times p}$ denotes the output of the q-th residual block, L_q denotes the q-th layer transformation, id is an identity mapping and Q is the total number of residual blocks. With such design, certain layers can be skipped if they have no contribution to the final target. This is achieved by driving the weights in L_q toward zero during training process [13]. In this way, *Bi-CGL* goes deeper as it will not produce a training error greater than its shallower counterparts. Similarly, for backward direction given $G_0^b = \overleftarrow{X}(*, t_j + \ell + 1 : t_j + 2)$ we have:

$$G_q^b = ReLU(L_q(G_{q-1}^b) + id(G_{q-1}^b)), \quad q = 1, 2, ..., Q. \quad (5)$$

Specifically, each G_q^f (G_q^b) represents certain level of temporal nonlinearity. By collecting the output from all the layers, we have $G_{1:Q}^f(i, *)$ which approximates $G^f(i, *)$ in Figure 3, and so does $G_{1:Q}^b(i, *)$ to $G^b(i, *)$. Therefore, we use $G_{1:Q}^f(i, *)$ and $G_{1:Q}^b(i, *)$ as the input to the next module.

Before we continue to introduce the next part, please note that:

1) The motivation of using a shared ResNet for both forward and backward directions is 1) to construct causal graph more consistently and 2) to alleviate overfitting. Comparatively, bi-LSTM [26] has two separate network for learning different state of forward and backward respectively. However, using different networks in our problem setting is non-trivial, as we have to merge forward and backward network to learn causal graph in a systematic way. Instead, we design a weight-sharing ResNet which gains analogous perspectives of both forward and backward directions. It will also decrease the number of parameters, which is beneficial for avoiding overfitting.
2) Module 1 only learns the temporal nonlinearity on univariate level. On the other hand, intervariable nonlinearity for regressing each variable only relates to the corresponding cause variables, and this is acquired during causal graph learning in Module 2.
3) In the first layer L_1, each column of the weight matrix can be treated as one weight-sharing temporal convolution filter with size ℓ across all the m time series. The reason why we use temporal convolution, instead of the other temporal sequence mining methods (e.g. recurrent neural network, or RNN), is that 1) it is more efficient and 2) it is sufficient to capture the causality in all the experiments we have tried.
4) Since causality may lay on any level of temporal nonlinearity, we feed the output from all ResNet layers to the next model. This is different from traditional ResNet which only uses the last layer output.

B. Module 2: Learning Causal Graph

The objective of the second module is to learn the causal graph. The input here is the learned temporal nonlinearity $G_{1:Q}^f$ (forward) and $G_{1:Q}^b$ (backward) from Module 1. Figure 5 illustrates the design of Module 2. We start by focusing on the forward direction G^f first. For each $G_q^f \in G_{1:Q}^f$ from Module 1, we want to select those variables that relevant in regressing $X(i, t_j + 1), i = 1, 2, ..., m$. This can be modeled through:

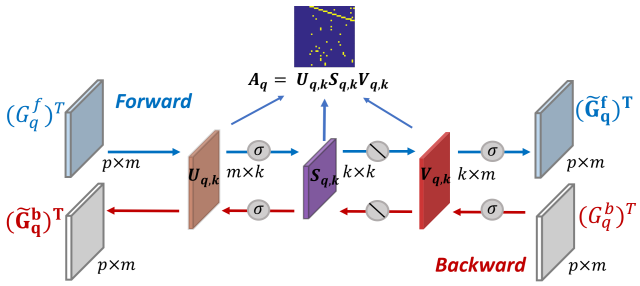


Fig. 5. Module 2 is to learn causal graph using a bi-directional weight-sharing and low-rank network, with different order.

$$(\tilde{\mathcal{G}}_q^f)^T(*, i) = (G_q^f)^T A_q(*, i), \quad (6)$$

where A_q is the causal graph to be learned ($q = 1, 2, ...Q$). We can extend this operation to the whole variable space, i.e., $i = 1, 2, ..., m$.

Instead of learning a full $m \times m$ causal graph, here we learn it in a low-rank format. There are two reasons for this design: 1) In real world the causality is usually sparse and causal graph has low-rank [8], [37], [38]. Approximating causal graph in low-rank can suppress the noise effect; and 2) when the size of variables (m) goes too large, learning low-rank approximation is more efficient. Specifically, we approximate A_q through a k-rank matrix decomposition with $k < m$: Each $G_q^f \in G_{1:Q}^f$ is firstly projected from the input m dimensional space to k-rank embedding space. This is done via nonlinear mapping with a weight matrix $U_{q,k} \in \mathbb{R}^{m\times k}$ and activation function *tanh*. Then a diagonal matrix $S_{q,k} \in \mathbb{R}^{k\times k}$ is learned to scale each low embedding. Finally, the k-rank embeddings is projected back to the original m space with a weight matrix $V_{q,k} \in \mathbb{R}^{k\times m}$ and activation function *Selu* [17]. This process can be described as:

$$(\tilde{\mathcal{G}}_q^f)^T = Selu\Big((tanh((G_q^f)^T U_{q,k})S_{q,k})V_{q,k}\Big). \quad (7)$$

Here, $U_{q,k}$ contains cause mapping information, $V_{q,k}$ contains information of effect mapping and $S_{q,k}$ contains scaling factors. The selections of activation functions here are based on experiment, which give the best performance.

Then, the causal graph A_q can be approximated by:

$$A_q \approx U_{q,k}S_{q,k}V_{q,k}. \quad (8)$$

It is worth to note that:

1) The reason why we bring in $S_{q,k}$ in the low-rank approximation is that $S_{q,k}$ can tell us the contribution of each rank, which can further contribute to decision support and explainability.
2) Although Equation (8) is in a similar format to singular vector decomposition (SVD), we are not claiming that they are directly relevant. In fact, both sides of the low-rank ($U_{q,k}$ and $V_{q,k}$) are learned in a nonlinear way; on the other hand SVD is linear.

For backward direction where we consider retrocausality, the roles of causes and effectors are switched, so the retrocausal graph is A_q^T. Therefore for each $G_q^b \in G_{1:Q}^b$, we have:

$$(\tilde{\mathcal{G}}_q^b)^T(*,i) = (G_q^b)^T A_q^T(*,i). \tag{9}$$

Similar to Equation (7) but with reverse order of causes and effectors, we apply the following projection process to G_q^b:

$$(\tilde{\mathcal{G}}_q^b)^T = Selu\Big((tanh\big((G_q^b)^T V_{q,k}^T\big) S_{q,k}^T) U_{q,k}^T\Big). \tag{10}$$

Please note that the retrocausal graph in backward direction is the transpose of causal graph in forward direction. Thus for each layer q, the $U_{q,k}$, $S_{q,k}$ and $V_{q,k}$ are with shared weights in forward and backward directions. Besides Figure 5, this design is illustrated in Module 2 in Figure 2, where layers with the same color and texture are weight-sharing between forward and backward directions.

Furthermore, we apply weight penalty $|U_{q,k}^T U_{q,k} - I|$ and $|V_{q,,k}^T V_{q,k} - I|$ to loss function (will be introduced in Section III-D) to force orthonormality of both $U_{q,k}$ and $V_{q,k}$. Such technique has been used in works [5], [34], and is theoretical supported by spectral theorem [29].

For each G_q, $q = 1, 2, ..., Q$ from Module 1, the A_q is learned respectively. Finally, we obtain the global causal graph A by:

$$A \approx \sum_{q=1}^{Q} A_q. \tag{11}$$

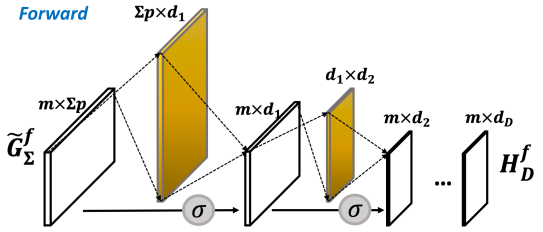


Fig. 6. Module 3 is to learn intervariable nonlinearity using a simple fully-connected chain. Here shows the learning process of forward direction. Backward direction is learned in the same way, but with a different fully-connected chain.

C. Module 3: Learning Intervariable Nonlinearity

Module 3 is to learn the intervariable nonlinearity among time series. We first introduce the forward direction part. The input of forward direction is all $\tilde{\mathcal{G}}_q^f \in \mathbb{R}^{m \times p}, q = 1, 2, ...Q$ from Module 2. We concatenate them column-wisely and denote the whole matrix as $\tilde{\mathcal{G}}_\Sigma^f \in \mathbb{R}^{m \times \sum_p}$. Module 3 consists of a series of fully connected layers, of which the j-th layer is defined as:

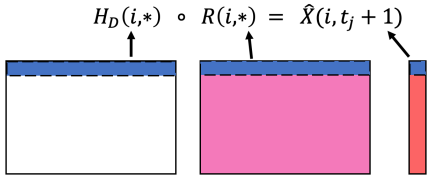
$$H_j^f = tanh(H_{j-1}^f W_j^f + b_j^f), \quad j = 1, 2, ..., D. \tag{12}$$

Given $H_0^f = \tilde{\mathcal{G}}_\Sigma^f$, $H_{j-1}^f \in \mathbb{R}^{m \times d_{j-1}}$ is output of layer $j-1$, $W_j^f \in \mathbb{R}^{d_{j-1} \times d_j}$ is a weight matrix and b_j^f is a bias vector. The output of the last layer is noted as $H_D^f \in \mathbb{R}^{m \times d_D}$ given total number of layers D in this module. Specifically, $\tilde{\mathcal{G}}_\Sigma^f(i,*)$ from Module 2 aggregates the nonlinearity of cause variables of $X(i,*)$ from forward direction. Such information is further convoluted nonlinearly in Module 3, so that intervariable nonlinearity is learned.

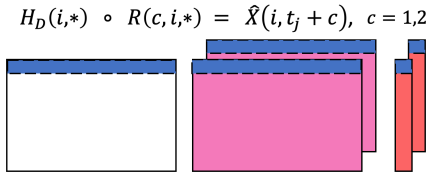
Separately, for backward direction, we apply another series of fully connected layers as:

$$H_j^b = tanh(H_{j-1}^b W_j^b + b_j^b), \quad j = 1, 2, ..., D \tag{13}$$

with $H_0^b = \tilde{\mathcal{G}}_\Sigma^b$. The last layer output is noted as $H_D^b \in \mathbb{R}^{m \times d_D}$. Here please note that we do not let the forward and backward directions share the same fully connected chain. This is to learn different intervariable nonlinearity on two directions respectively.



(a) Module 4 with 1 time-stamp regression.



(b) Module 4 with 2 time-stamp regression.

Fig. 7. Module 4 is to approximate the regression target.

D. Module 4: Regression

Finally, Module 4 is to regress the target $\hat{X}(*, t_i + 1) \in \mathbb{R}^{m \times 1}$ using the output H_D^f and H_D^b from Module 3. We first concatenate H_D^f and H_D^b column-wisely, and denote as $H_D \in \mathbb{R}^{m \times 2d_D}$. Then we learn a weight matrix $R \in \mathbb{R}^{m \times 2d_D}$, and the regression is performed by row-wise dot product (as shown in Figure 7(a)):

$$\hat{X}(i, t_i + 1) = H_D(i,*) \circ R(i,*), \quad i = 1, 2, ..., m \tag{14}$$

The residual between the estimated $\hat{X}(*, t_i+1)$ and the actual $X(*, t_i + 1)$ is calculated by mean square error (MSE) with regularization terms. We define the loss function as:

$$\begin{aligned} loss = & \frac{1}{mn} \sum \Big(X(*, t_i + 1) - \hat{X}(*, t_i + 1)\Big)^2 \\ & + \lambda_1 \sum_q \big(\|U_{q,k}\|_2 + \|V_{q,k}\|_2\big) \\ & + \lambda_2 \sum_q \Big(\big\|U_{q,k}^T U_{q,k} - I\big\|_2 + \big\|V_{q,k}^T V_{q,k} - I\big\|_2\Big), \end{aligned}$$

where n is the total number of samples and m is the number of variables. The first regularization term is to avoid overfitting on the causal graph, while the second one is to impose orthonormality to $U_{q,k}$ and $V_{q,k}$ (as mentioned in Section

III-B). The calculated loss is then backpropagated through the model to update all the parameters.

Specifically, the i-th row of H_D from Module 3 contains all the possibly-contributing nonlinear forms of the causes of $X(i, t_j + 1)$ from both forward and backward inputs. Module 4 is to learn the optimal combination to approach $X(i, t_j + 1)$.

In practice, more than one timestamps can be set as the regression target, so that more information can be involved in each training step to learn a more accurate causal graph. For example in Figure 7(b), the model uses two timestamps $X_{*,t_j+1:t_j+2}$ as regression target and learns two different layers of R respectively. The rationality of such design will be verified in the experiment section.

IV. Discussion

In this section, we justify our proposed *Bi-CGL* method by building its connection with a few relevant researches.

Granger Causality. Classical works were proposed in [4], [15] for detecting Granger causality, which was originally defined in [11]. Among m variables, a VAR model is first established on each pair of two variables and then certain statistics tests are performed on residuals of one variable with and without considering the other. Such procedure can be performed $m(m-1)$ times to learn the complete causal graph. These methods have three limitations. The most notable one is its high false alarm rate, because it only models two variables at one time. Any more than three joint relationships cannot be effectively detected in the final causal graph. Second, since statistical tests are involved, such methods require the input data to follow certain distribution assumptions, which is not guaranteed in real applications. Third, the VAR model is still inherently linear. In contrast, our proposed approach does not rely on any distribution assumptions or pairwise causal graph construction, and can detect nonlinear causality.

MLP and LSTM. Work in [32], [31] proposed multi-layer perception (MLP) and LSTM deep neural network to learn causal graph through time series prediction. These two models can be considered as an extension of VAR, where more complicated temporal relationship can be learned in hidden layers. However, both models detect causality by considering the weights of neurons from the first layer only. This can only serve as the sufficient but not the necessary condition of causality and thus give a lower recall rate. Because of this, such methods have worse performance than our *Bi-CGL* method (please refer to the comparison in experiment section).

Bi-directional Models. Work in [26], [30], [3] proposed bi-direction LSTM models for sequence classification/regression and natural language generation, with improved performance than traditional LSTM. In [6], a bi-direction VAR model is proposed for Granger causality analysis. However, currently there is no bi-directional deep learning framework for causality analysis, especially for detecting complicated nonlinear temporal relationship.

Graph Learning Methods. Work in [1] learn causality by adding lasso or ridge regularization to VAR. The complete causal graph is generated in one shot by examining the absolute value of the weights of each two variables across all temporal order, as described in Section II-B. However, it can only detect linear temporal relationship, while our approach is capable of capturing nonlinear relationships. Recent work in [16] uses graph neural network to learn latent interaction in multi-variate physical dynamics system. It is a type of message passing neural network where node-edge message passing operations are defined through MLP or LSTM as encoder and decoder in variational autoencoder setting. However, 1) the model is developed for a dynamic physical interaction graph instead of causal graph, 2) without low-rank approximation, the message passing operations and variational autoencoder require to learn a large number of parameters. Therefore, this model suffers from overfitting and are not scalable to large number of variables. On the other hand, our *Bi-CGL* has good scalability both in time and computational memory. hi

Transfer Entropy. This type of research was firstly proposed in [22] and then extended in [19] to be more robust and computationally efficient, such method can learn pairwise causality without distribution assumptions and linear setting. One variable X is considered a cause of another variable Y, if the past values of X significantly decrease the Shannon entropy, which measures the level of uncertainty of current Y values. However, this type of methods only considers the pairwise causality and thus shares similar limitations of Granger causality.

Nonlinear Kernel-based Methods. Work in [18], [27], [22] proposed a series of Granger causality analysis using kernel-based VAR model. Several types of kernels are used to capture the nonlinear temporal relationship. However, such methods come with a heavy computational cost, as claimed, in [27], that their computational complexity is $\mathcal{O}(m^3 + n^3)$, with m the number of variables and n number of total timestamps. Besides, all these methods are based on pre-assumed kernels which is not guaranteed in the applications. On the other hand, our model is more scalable and not built upon pre-assumed kernels.

Supervised Learning Methods. Work in [20] formulates the causal inference learning as a cause-effect classification problem. Concretely, it classifies relationship into three types: 1 (X causes Y), -1 (Y causes X), and 0 (no causal relationship). However, such methods request labeled relationship for training, which is difficult to collect in many cases. Comparatively, our *Bi-CGL* is purely unsupervised that learns causal graph through time series regression without any relationship label.

V. Experiment

A. Experiment Setup

The experiment is performed on both synthetic and real-world datasets *.

Construction of Synthetic Dataset. Two synthetic datasets are constructed for model comparison. We first construct

*The code and synthetic datasets are available in https://drive.google.com/open?id=1dvRul9AKooRz_T6KYZnxLbzWNyrtlQhe

causal graphs, which are used to 1) generate multivariate time series data and 2) later serve as groundtruth to evaluate the performance. Based on each causal graph, the time series data is generated with nonlinear temporal relationships. The causal graph is constructed based on a three-layer hierarchical structure setting (illustrated in Figure 8). Such hierarchical structure commonly appears in biology science [36]. Nodes in the top layer regulate nodes in the second layer which then are the causes of the bottom layer nodes. Random directed edges within the second layer are also added to increase the problem complexity. We call the nodes in the top layers as *masters*, and the remaining nodes as *effectors*.

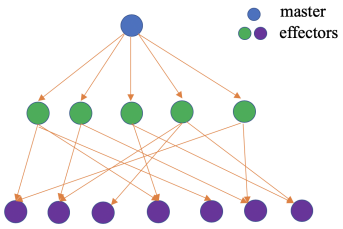


Fig. 8. Illustration of our three layers hierarchical causal structure.

We assign 5-15-45 nodes in the three layers of *Synthetic Dataset A*, while *Synthetic Dataset B* has a hierarchical structure of 10-30-90 nodes. Based on the causal graph, the time series expression values are generated as follows:

Synthetic Dataset A. For *masters*, we add the temporal nonlinearity as:

$$X(i,t_j) = \cos(\sqrt{3}d(i) \times X(i,t_j-1) \times d(i)^2 \times X(i,t_j-2)) + \epsilon(i,t_j),$$

where $d(i)$ simulates the decay effect for node i and is selected from uniform distribution $\mathcal{U}(0.95,1)$, and $\epsilon(i,t_j)$ is the random noise sampled from normal distribution $\mathcal{N}(0,1)$. The expression values of *effectors* are simulated as:

$$X(k,t_j) = \sum_{l=1}^{m} \log((r(l \Rightarrow k) \times X(l,t_j - p(l \Rightarrow k)))^2) + \epsilon(k,t_j),$$

with $r(l \Rightarrow k)$ following $\mathcal{U}(-1,1)$ if node l is among the causes of node k and set to 0 otherwise. *Synthetic Dataset A* is designed to have the model order 3, so term $p(l \Rightarrow k)$ represents the time lag of the intervariable nonlinearity from l to k and is randomly selected from $\{1,2,3\}$. The first three timestamp values of all nodes are randomly generated from $\mathcal{N}(0,1)$. We generate $n = 600$ timestamps in this dataset.

Synthetic Dataset B. For *masters*, we set the temporal nonlinearity as:

$$X(i,t_j) = \arctan(\sqrt{3}d(i) \times X(i,t_j-1) + d(i)^2 \times X(i,t_j-5)^2) + \epsilon(i,t_j).$$

For *effectors*, we first generate a latent variable $Z_l(k,t_j)$, which represents the contributing factor of node l to node k at timestamp t_j. This can be formulated as:

$$Z_l(k,t_j) = r(l \Rightarrow k) \times X(l, t_j - p(l \Rightarrow k)),$$

Then, we let:

$$X(k,t_j) = \sum_{l=1}^{m}(Z_l^2(k,t_j) + \tanh(Z_l(k,t_j) \times X(k,t_j - p(l \Rightarrow k)))) + \sum_{s=1}^{m}\sum_{w=1}^{m}(\sin(Z_s(k,t_j) \times Z_w(k,t_j))) + \epsilon(k,t_j).$$

where $r(*)$ and $p(*)$ are defined in the same manner as *Synthetic Dataset A*. This dataset is designed to have model order 5 with timestamps $n = 1200$. Here, the value of node k at timestamp t_j is not only affected by the interaction of its own previous value and its cause node values, but also by the interaction of values of each pair of nodes among its common causes.

Evaluation Metrics and Baseline Methods

To evaluate the learned causal graph with groundtruth, we convert all the learned graph to non-negative by taking absolute value first, and use two metrics: Area Under Precision and Recall curve (**AUPR**) and Area Under Receiver Operating Characteristics curve (**AUROC**). As claimed in [25], AUPR is a more proper metric for a sparse groundtruth. We compare our *Bi-CGL* method with the following six popular baselines:

1) *VAR* [1]. It applies the vector autoregressive model (VAR) with ridge regularization for causal graph learning.
2) *PCKGC* [24]. For each variable, it first selects a subset of other variables with highest mutual information scores, and then applies conditional kernel Granger causality analysis in [22], to reduce the computational cost.
3) *Copula* [2]. It transforms the marginal distribution of each variable to Gaussian distribution and then applies *VAR*. Therefore it is capable to capture certain levels of nonlinearity.
4) *FBLG-CP* [6]. It applies forward and backward *Copula* and learns two causal graphs from two direction respectively. The final causal graph is simply calculated as the average of the two graphs.
5) *cLSTM* [32]. It applies LSTM for time series prediction with group lasso regularization on the first layer neurons.
6) *pTE* [19]. It calculates the phase transfer entropy by transforming temporal signals to discrete phases with certain histogram-based probability functions.

B. Comparison on Synthetic Datasets

Our model is trained using the following hyperparameter settings. For *Synthetic Dataset A*, we set the input time series window as 5, and the prediction window size (pre-win) as 3. The number of temporal layers is set to be $Q = 5$ in Module 1, each with dimension $p = 40$ and low-rank embedding $k = 50$ in Module 2. A single fully connected layer with dimension $d_1 = 80$ is used in Module 3. For *Synthetic Dataset B*, we set the input window to 10 and the prediction window as 5. We

set $Q = 6$ with dimension $p = 40$ and $k = 75$. In Module 3, we set $d_1 = 50$. Please refer to our public code for more details. For other baselines, we tuned their hyperparameters to achieve as highest accuracy as we can.

We show the performance of all models on two synthetic datasets in Figure 9. It it obvious that our *Bi-CGL* achieves the best performance. The AUPR by our *Bi-CGL* is significantly higher than all the baselines (almost 46% improvement than the second best). The AUROC of our *Bi-CGL* is also the best. Among other baselines, *pTE, Copula* and *FBLG-CP* give relatively better results, since they do not require any statistical assumptions and can capture certain level of nonlinearity. *FBLG-CP* has slightly worse AUPR compared with *Copula*. Since it obtains the final causal graph by simply taking average of the causal graphs from two directions, the comparison here suggests that this may not be a proper way to incorporate bi-direction causality. Comparatively, our *Bi-CGL* is designed to train with a weight-sharing network on both directions, which is more robust and systematic than *FBLG-CP*. On the other hand, *cLSTM* does not perform well either, since it only utilizes its first layer neurons to represent causality. Finally, *PCKGC* and *VAR* give the worst performance. It is because that *PCKGC* only detects the conditional Granger causality conditioned on a subset of all variables, while *VAR* suffers from its linear regression setting.

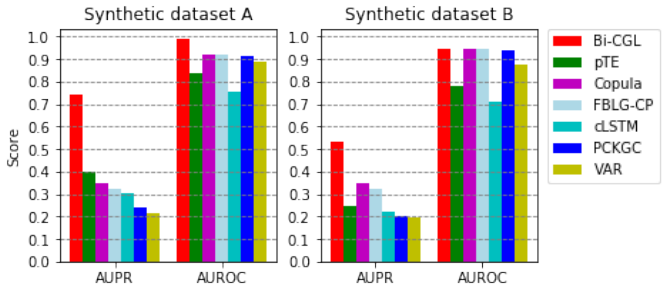


Fig. 9. Comparison of AUPR and AUROC of all methods.

Figure 10 shows the convergence of our *Bi-CGL* method with training loss, validation loss, and AUPR across training epochs. Apparently, *AUPR* rises simultaneously when the training loss and validation loss converge. This illustrates that an accurate causal graph is generated through the well-training of the regression model.

C. Comparison on a Real-world Dataset

We also compare on a real world engine operating data, which contains six months data (number of samples is over $180k$) from 50 different engines. Here we use the following setting for our model: input time series window $= 10$, pre-win$= 2$, $Q = 4$, $p = 30$, $k = 10$ and $d_1 = 50$.

We show the learned causal graphs in Figure 11, which only includes the 14 key variables of the dataset for visualization purpose. Besides our *Bi-CGL*, we include the second best method *Copula* in our synthetic experiment. Our *Bi-CGL* model successfully captures the following physical rules:

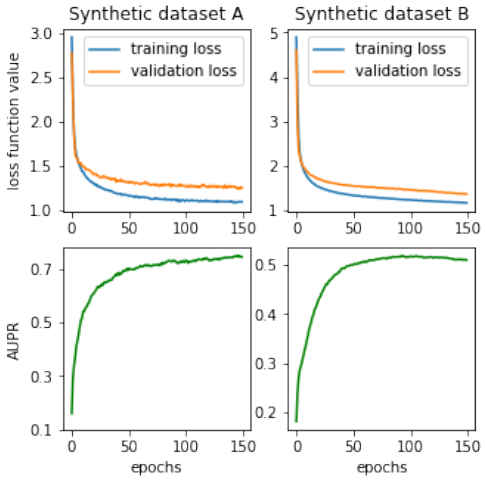


Fig. 10. Training/Validation loss and AUPR across epochs.

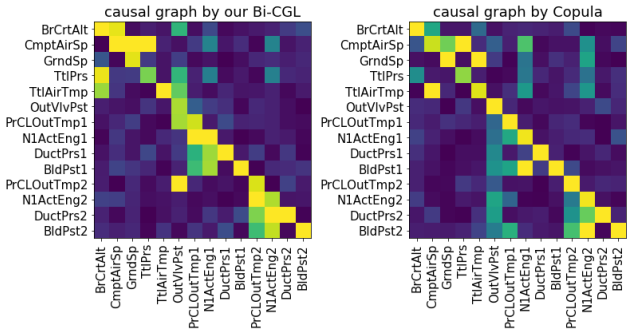


Fig. 11. Learned causal graph of real-world data.

Corrected Altitude (BrCrtAlt) is calculated by a function of total pressure (TtlPrs) and total air temperature (TtlAirTmp); BrCrtAlt is relevant to air-density changes that affects Computed Airspeed (CmptAirSp); N1ActEng is fan speed which is related to cooling outlet temperature (PrCLOutTmp), and it is affected by Pressure (DuctPrs) and Bleed Position (BldPst); Valve Position (OutVlvPst) is partially open until selected altitude is reached (BrCrtAlt). The change of OutVlvPst can be also triggered by the change of PrCLOutTmp.

Comparatively, the other baselines such as *Copula* fail to discover these physical rules.

D. Robustness against Noise Levels

Figure 12 shows the robustness of all models against different level of noise. Three different noise levels (10%, 20%, and 30%) are injected into data. For example, noise 30% is injected in the following way:

$$X_{30\%}(i, t_j) = X(i, t_j) \times \mathcal{U}(1 - 30\%, 1 + 30\%) \qquad (15)$$

where $i = 1 \cdots m$ and $t_j = 1 \cdots n$, and $\mathcal{U}$ indicates uniform distribution. Please note that even for the dataset with 0% injected noise, there is basic noise from ϵ during construction (please refer to synthetic dataset construction). For each noise level, we run 10 times with different random seeds and collect

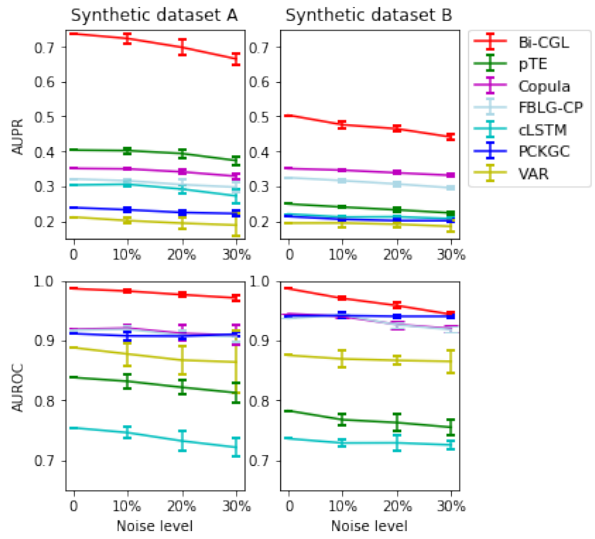
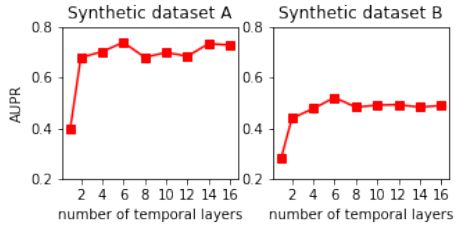
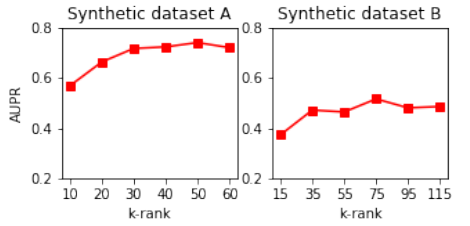


Fig. 12. Comparison in mean and standard deviation with various noise levels.



(a) AUPR with various temporal layers in **Module 1**



(b) AUPR with various k-rank in **Module 2**

Fig. 13. Effects of two hyper-parameters in *Bi-CGL*.

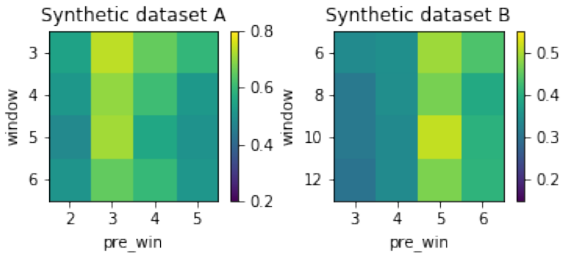


Fig. 14. AUPR of various size of input window and predicting-window (pre-win) on the two synthetic datasets.

the mean and standard deviation. Generally, all methods' performances drop as noise increases. However, *Bi-CGL* still maintains AUPR no less than 0.6 and 0.4 for the two datasets respectively even with 30% noise, with very small standard deviation. This confirms *Bi-CGL*'s robustness against noise by weight-sharing and low-rank embedding, and our discussion in Section III-B.

E. Effect of Number of Low-rank and Temporal Layers

Here, we discuss the effect of two important hyperparameters in *Bi-CGL*. Figure 13(a) shows the AUPR by our *Bi-CGL* model with different numbers of temporal layers (Q in Module 1). With increasing Q, AUPR climbs in the beginning and then keeps stable with slight fluctuations. This tells that *Bi-CGL* remains stable with a reasonably large number of temporal layers in Module 1.

Figure 13(b) displays the performance with various k of k-rank embeddings in Module 2. It shows that our *Bi-CGL* can learn an accurate causal graph with k being only half of number of variables (m). This can reduce computational cost and suppress noise influence from data and modeling process.

F. Robustness against Input Window and Pre-win Size

In Figure 14, we test AUPR of our *Bi-CGL* with various numbers of input time windows (denoted as *window*) and timestamps that we want to regress (noted as *pre-win*) in Module 4. We find that AUPR remains reasonably stable by increasing the input time windows, which means that the temporal convolution in Module 1 correctly weights different time-lags. On the other hand, for a given input window size, the AUPR decreases with too small or too large pre-win. One possible reason is that more than one predicting windows can include more information that may contribute to learning better causal graph. But too many pre-wins will increase the difficulty in regression, which confuses the backpropagation and leads to poor quality of the learned causal graph.

G. Scalability Comparison

Figure 15 displays the scalability plot of all methods with increasing number of variables (m). All experiments are performed with Intel Xeon E5-2670 processors with 128G memory. It shows that the computing time of *VAR, Copula, FBLG-CP, PCKGC* and *pTE* have complexity of $\mathcal{O}(m^3)$. Our *Bi-SCL* has the second-lowest order while *cLSTM* has the most lowest order. However, *Bi-CGL* has much better accuracy than *cLSTM* in learning causal graph. This confirms that our *Bi-CGL* is both scalable and effective.

PCKGC consumes too much time when the number of variable reaches 10^3 or greater, whereupon we stopped measuring its computing time.

VI. Conclusion

By exploring forward and backward time series, we propose an unsupervised bi-directional neural network, for causal graph learning in multivariate and nonlinear time series data. Specifically, 1) it simultaneously explores the forward and backward time series to obtain more accurate causal graph, and 2) it is built upon a weight-sharing low-rank approximation, which makes the model more robust against noise. This method can

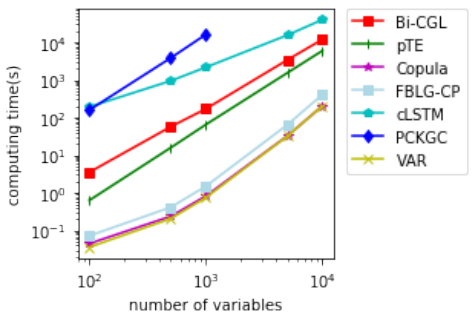


Fig. 15. Scalability test of all models.

discover nonlinear causality without any pre-assumed kernel or distribution model. Our empirical results on both synthetic and real world datasets demonstrate that our proposed method outperforms other baselines by a significant margin.

ACKNOWLEDGMENT

This work was supported in part by United State, Department of Energy, under Contract Number DE-SC0012704.

REFERENCES

[1] Andrew Arnold, Yan Liu, and Naoki Abe. Temporal causal modeling with graphical granger methods. In *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '07, pages 66–75, New York, NY, USA, 2007. ACM.

[2] Mohammad Taha Bahadori and Yan Liu. An examination of practical granger causality inference. In *Proceedings of the 2013 SIAM International Conference on data Mining*, pages 467–475. SIAM, 2013.

[3] Yi Bin, Yang Yang, Fumin Shen, Xing Xu, and Heng Tao Shen. Bidirectional long-short term memory for video description. In *Proceedings of the 24th ACM international conference on Multimedia*. ACM, 2016.

[4] Steven L Bressler and Anil K Seth. Wiener–granger causality: a well established methodology. *Neuroimage*, 58(2):323–329, 2011.

[5] Andrew Brock, Theodore Lim, James M Ritchie, and Nick Weston. Neural photo editing with introspective adversarial networks. *arXiv preprint arXiv:1609.07093*, 2016.

[6] Dehua Cheng, Mohammad Taha Bahadori, and Yan Liu. Fblg: A simple and effective approach for temporal dependence discovery from time series data. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '14, pages 382–391, New York, NY, USA, 2014. ACM.

[7] Belkacem Chikhaoui, Mauricio Chiazzaro, and Shengrui Wang. A new granger causal model for influence evolution in dynamic social networks: The case of dblp. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.

[8] Alessandro Chiuso and Gianluigi Pillonetto. A bayesian approach to sparse dynamic network identification. *Automatica*, 48(8), 2012.

[9] Cees Diks and Valentyn Panchenko. A new statistic and practical guidelines for nonparametric granger causality testing. *Journal of Economic Dynamics and Control*, 30(9-10):1647–1669, 2006.

[10] Thomas Gold. The arrow of time. *American Journal of Physics*, 1962.

[11] Clive Granger. Investigating causal relations by econometric models and cross-spectral methods. *Econometrica*, 37(3):424–38, 1969.

[12] David Gunning. Explainable artificial intelligence (xai). *Defense Advanced Research Projects Agency (DARPA), nd Web*, 2017.

[13] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016.

[14] Patrik O Hoyer, Dominik Janzing, Joris M Mooij, Jonas Peters, and Bernhard Schölkopf. Nonlinear causal discovery with additive noise models. In *Advances in neural information processing systems*, 2009.

[15] Dominik Janzing, David Balduzzi, Moritz Grosse-Wentrup, and Bernhard Scholkopf. Quantifying causal influences. 2013.

[16] Thomas N. Kipf, Ethan Fetaya, Kuan-Chieh Wang, Max Welling, and Richard S. Zemel. Neural relational inference for interacting systems. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018*, 2018.

[17] Günter Klambauer, Thomas Unterthiner, Andreas Mayr, and Sepp Hochreiter. Self-normalizing neural networks. In *Advances in neural information processing systems*, pages 971–980, 2017.

[18] Néhémy Lim, Florence DAlché-Buc, Cédric Auliac, and George Michailidis. Operator-valued kernel-based vector autoregressive models for network inference. *Machine learning*, 99(3):489–513, 2015.

[19] Muriel Lobier, Felix Siebenhhner, Satu Palva, and J. Matias Palva. Phase transfer entropy: A novel phase-based measure for directed connectivity in networks coupled by oscillatory interactions. *NeuroImage*, 85:853 – 872, 2014. New Horizons for Neural Oscillations.

[20] David Lopez-Paz, Krikamol Muandet, Bernhard Schlkopf, and Iliya Tolstikhin. Towards a learning theory of cause-effect inference. In Francis Bach and David Blei, editors, *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, Lille, France, 07–09 Jul 2015. PMLR.

[21] Helmut Lütkepohl. *New introduction to multiple time series analysis*. Springer Science & Business Media, 2005.

[22] D. Marinazzo, M. Pellicoro, and S. Stramaglia. Kernel-granger causality and the analysis of dynamical networks. *Phys. Rev. E*, 77, May 2008.

[23] Daniele Marinazzo, Wei Liao, Huafu Chen, and Sebastiano Stramaglia. Nonlinear connectivity by granger causality. *Neuroimage*, 58(2), 2011.

[24] Daniele Marinazzo, Mario Pellicoro, and Sebastiano Stramaglia. Causal information approach to partial conditioning in multivariate data sets. *Computational and mathematical methods in medicine*, 2012, 2012.

[25] Takaya Saito and Marc Rehmsmeier. The precision-recall plot is more informative than the roc plot when evaluating binary classifiers on imbalanced datasets. *PloS one*, 10(3):e0118432, 2015.

[26] Mike Schuster and Kuldip K Paliwal. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, 45(11), 1997.

[27] Vikas Sindhwani, Minh Ha Quang, and Aurélie C Lozano. Scalable matrix-valued kernel learning for high-dimensional nonlinear multivariate regression and granger causality. *arXiv preprint arXiv:1210.4792*, 2012.

[28] Linda Sommerlade, Marco Thiel, Bettina Platt, Andrea Plano, Gernot Riedel, Celso Grebogi, Jens Timmer, and Björn Schelter. Inference of granger causal time-dependent influences in noisy multivariate time series. *Journal of neuroscience methods*, 203(1):173–185, 2012.

[29] Petre Stoica and Randolph L Moses. *Introduction to spectral analysis*, volume 1. Prentice hall Upper Saddle River, NJ, 1997.

[30] Martin Sundermeyer, Tamer Alkhouli, Joern Wuebker, and Hermann Ney. Translation modeling with bidirectional recurrent neural networks. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 14–25, 2014.

[31] Alex Tank, Ian Cover, Nicholas J Foti, Ali Shojaie, and Emily B Fox. An interpretable and sparse neural network model for nonlinear granger causality discovery. *arXiv preprint arXiv:1711.08160*, 2017.

[32] Alex Tank, Ian Covert, Nicholas Foti, Ali Shojaie, and Emily Fox. Neural granger causality for nonlinear time series. *arXiv preprint arXiv:1802.05842*, 2018.

[33] Martin Vinck, Lisanne Huurdeman, Conrado A Bosman, Pascal Fries, Francesco P Battaglia, Cyriel MA Pennartz, and Paul H Tiesinga. How to detect the granger-causal flow direction in the presence of additive noise? *Neuroimage*, 108:301–318, 2015.

[34] Eugene Vorontsov, Chiheb Trabelsi, Samuel Kadoury, and Chris Pal. On orthogonality and learning recurrent networks with long term dependencies. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 3570–3578. JMLR. org, 2017.

[35] Satosi Watanabe. Symmetry of physical laws. part iii. prediction and retrodiction. *Reviews of Modern Physics*, 27(2):179, 1955.

[36] Shun Yao, Shinjae Yoo, and Dantong Yu. Prior knowledge driven granger causality analysis on gene regulatory network discovery. *BMC Bioinformatics*, 16(1):273, Aug 2015.

[37] Mattia Zorzi and Alessandro Chiuso. A bayesian approach to sparse plus low rank network identification. In *2015 54th IEEE Conference on Decision and Control (CDC)*. IEEE, 2015.

[38] Mattia Zorzi and Alessandro Chiuso. Sparse plus low rank network identification: A nonparametric approach. *Automatica*, 76, 2017.