NeurIPS 2023 CSL Competition Report

Hao Song Yifan Chen Jizhe Wang Mingda Li Xinyi Wang
Weinan Zhang*
Harbin Institute of Technology
{hsong, yfchen, jzwang, mdli, xywang, wnzhang}@ir.hit.edu.cn

1 Problem Definition

This competition aims to find a methodlogy to learn the causal structure from multi-type discrete-time event sequences with prior knowledge and topology constraint for alarm detection in AIOps (Artificial Intelligence for IT Operations). Concretely, the competition provides three types of datasets, historical alarm data, prior knowledge and device topology knowledge. Historical alarm data is a time-series data records the ocurrence time of each alarm id and the corresponding abnormal device id during a period of time. Prior knowledge data contains a prior belief causal graph and some RCA (Root Cause Analysis) snapshots. Device topology data is the connection graph of all devices. The goal of the competition task is to identify the potential causal structure between all alarm types (ids) based on given information. Formally, we denote all alarm ids as $\mathcal{V} = \{V_i\}$, we treat the occurance times of the alarm i during a fixed time window T as a Random Variable X_i (for temporal-based methods, X_i representes a stochastic process $X_i = \{X_t^i\}$). By slicing the historcial alarm data with time sliding window T, we can get the observational data (trajactories) $x_i = \{x_j^i\}_j$, for each X_i, we use $\mathcal{O} = \{x_i\}_i$ to denote all observational data. Our goal is to find a causal graph $\mathcal{G}_\mathcal{V}$ which can best explain the observational data $\mathcal{O}$ and RCA snapshot data $\mathcal{R}$ subject to the constraint of device topology graph $\mathcal{G}_\mathcal{D}$ and prior graph $\mathcal{G}_\mathcal{P}$.

2 Proposed Solution

As the same in phase 1, our key method is to add an initial matrix before running PTHP, which means we only check where the value is 1 in initial matrix. Using initial matrices can improve the accuracy of the results and speed up the PTHP algorithm. In our experiments, the initial matrices are obtained from DAGMA, PC and causal effect. For higher accuracy, we use the prior matrix as a filter covering our result matrices in the end. Followings are the brief descriptions of the methods including PTHP.

2.1 DAGMA

DAGMA(Directed Acyclic Graphs via M-matrices for Acyclicity) has demonstrated the capability of learning causal structures of DAGs(Directed Acyclic Graphs) from data Bello et al. [2022].

Inspired by the methodology presented in this paper, we employed the DAGMA approach for generating initial matrices, which yielded impressive results. The authors introduced a novel acyclicity feature based on the log-det(log-determinant) function, effectively leveraging the nilpotency characteristic of DAGs. To address the inherent

*Instructor

Preprint. Under review.

asymmetry in DAGs, the domain of the log-det feature was associated with the set of M-matrices. This association marks a significant departure from the traditional definition of the log-det function over the cone of positive definite matrices. The algorithmic process in the original paper is shown in Algorithm 1.

Algorithm 1 DAGMA

Require: Data matrix X, initial central path coefficient $\mu^{(0)}$ (e.g., 1), decay factor $\alpha \in (0, 1)$ (e.g., 0.1), ℓ_1 parameter $\beta_1 > 0$ (e.g., 0.01), log-det parameter $s > 0$ (e.g., 1), number of iterations T.

1: Initialize $\theta^{(0)}$ so that $W(\theta^{(0)}) \in W^s$.
2: **for** $t = 0, 1, 2, \ldots T-1$ **do**
3: Starting at $\theta^{(t)}$, solve $\theta^{(t+1)} = \arg\min_\theta \mu^{(t)}(Q(f_\theta; X) + \beta_1\|\theta\|_1) + h^s_{ldet}(W(\theta))$
4: Set $\mu^{(t+1)} = \alpha\mu^{(t)}$
5: **end for**

Ensure: $W(\theta^{(T)})$

2.2 PC

The PC algorithm (Peter-Clark Algorithm) Kalisch and Bühlman [2007] is a causal relationship identification method based on conditional independence tests. It begins with a fully connected graph and iteratively removes the connections between variables that are independent under the condition of other variables. This iterative process results in a sparse graph that represents direct causal relationships.

In this competition, we utilized the PC algorithm to process high-dimensional data from telecommunications equipment alarms, generating an initial matrix to reveal potential causal relationships between alarms. This initial matrix provided a crucial foundation for more in-depth analysis and the development of strategies for alarm prevention. It effectively facilitated a deeper understanding of the alarm data and a thorough exploration of its causal relationships.

2.3 Causal Effect

In addition to the DAGMA and PC algorithms, we also employed the Causal Effect method to generate the initial matrix. This method involves ordering all the alarm data and then calculating the causal effect of causal relationships between alarms within a sliding window. As the window slides, causal effects across the entire dataset are obtained. The causal effect between alarm i and alarm j is quantified by the following formula:

$$Y = (x_j|do(x_i = 1)) - Y = (x_j|do(x_i = 0)) > E(Y(x_j))$$

Unlike the PC algorithm, the Causal Effect method not only identifies potential causal relationships between variables but also accurately calculates the magnitude of these causal effects. This is crucial for understanding the complex interactions between alarm and predicting future alarm patterns.

2.4 PTHP

Our main method is PTHP郝志峰 [2022], which is based on THPCai et al. [2022] and begins iterations from an initialed graph provided by other methods. THP methodCai et al. [2022] models events' generation as temporally and topologically accumulative effects from other events. Specifically, the causal effect of one event with specific time and node, is an aggregation of an inherent effect intrinsic to its event type, and total influences of other events which are their occurrence times accumulated on the past

Algorithm 2 PC

Require: Set of nodes N.

Construct a fully connected undirected graph G using all nodes in N.
for each pair of nodes $A, B \in N$ **do**
    Consider the set of all adjacent nodes of A and B, denoted as $Adj(A, B)$.
    if there exists a subset $S \subseteq Adj(A, B) \setminus \{A, B\}$ such that A and B are independent given S **then**
        Remove the edge between A and B from G.
    end if
end for
for each triplet of nodes $A, B, C \in N$ in the pattern $A - -B - -C$ **do**
    Consider the full set of nodes.
    if there does not exist a subset $S \subseteq N$ containing B but not A and C such that A and C are independent given S **then**
        Rewrite $A - -B - -C$ as $A \to B \leftarrow C$ in G.
    end if
end for
repeat
    for each directed edge $A \to B$ in G **do**
        for each node C adjacent to B but not to A **do**
            if B has no outgoing edges **then**
                Rewrite the edge $B - -C$ as $B \to C$ in G.
            end if
        end for
    end for
    for each pair of adjacent nodes $A, B \in N$ **do**
        if there is a directed path from A to B **then**
            Rewrite the edge $A - -B$ as $A \to B$ in G.
        end if
    end for
until no more undirected edges can be changed in G

Ensure: Modified graph G.

time and neighbors with a trainable weights. It can be formulated as:

$$\lambda_v(n,t) = \mu_v + \sum_{v' \in V} (\psi_{v',v} * \mathrm{d}C_{v'})_{G_N \times T}(n,t),$$

where $\lambda_v(n,t)$ is intensity function which denotes the potential occurrence times of event v on node n in time t. Here, μ_v is base intensity of event of type v, $\psi_{v',v}$ denotes the causal influence of event v' on event v, $\mathrm{d}C_{v'}$ is the occurrence time of event v' in time interval $[t - \mathrm{d}, t]$, and $G_N \times T$ represents it's accumulated on $[0, t]$ and all nodes.

The main difference between PTHP and THP lies that PTHP uses some faster and reliable methods, like DAGMA, PC, Causal Effect, for the pre-selection of causal edges, which means that some $\psi_{v',v}$ can be obtained in advance and speed up train procedure.

3 Deployment

3.1 Some Files

In our folder, `CD_methods.py` is a collection of the methods for causal discovery. `Utils.py` is the collection of some tool functions. `to_get_initial_matrices.py` is the python file to generate the initial matrices. For convenience, we have generated the required initial matrices in the folder `_init_matrices`. And the `main.py` is the main file to run by command `python ./main.py`. Folders `pthp` and `trustworthyAI`

表 1: Hyperparameter setting

datasets	Best iteration	max_hop	delta	epsilon
4	2	1	0.01	1
5	56	2	0,01	1
6	6	1	0.01	1

are the needed frameworks. The results will be saved in the folder `./FINAL_RESULTS` as defined in the file `datapath.json`. And `environment.yml` is the environment file.

3.2 Process

First, clone the repos by command: `git clone https://github.com/huawei-noah/trustworthyAI.git`.

Before running the main program, the environment can be set by command:`conda env create -f environment.yml`, and its name is `hsong_CD`.

Then run the command `python ./main.py`. And we suggest running in parallel for each dataset.

4 Conclusion

In summary, our method can be devided into three stages, they are initial matrix, PTHP then prior matrix filter.

Our results are mainly affected by the number of iterations. We explored the best iterations and other parameters on our method as table 1.

Moreover, we have tried to find the usages of topology and rca snapshots. We leverage the topology file to be a filter to sift out impossible causal edges. The idea is that if two alarms are not in the same connectivity graph, then there can be no causal relationship between them. But in terms of results, it did not appear to be effective. For rca prior, we tried to add some initial edges from the root cause to other alarms. For example, 18 is the root cause, 26 is another root cause and 18 is in the snapshot of 26, while 26 is not in the snapshot of 18, which indicates that (26, 18) may exist. So we could put this edge into initial matrix. But as far as the performance is concerned, this method show no visible effect.

We also found that for dataset 5 as well as the synthesized datasets, our method based on data dependency works well but bad on the real datasets. Through exploring the datasets we found that there are few strong correlation in the real datasets so we cannot detect more causal relationships. That is why the methods based on data dependency performed bad. This problem could be a future work.

References

Kevin Bello, Bryon Aragam, and Pradeep Ravikumar. Dagma: Learning dags via m-matrices and a log-determinant acyclicity characterization. Advances in Neural Information Processing Systems, 35:8226–8239, 2022.

Markus Kalisch and Peter Bühlman. Estimating high-dimensional directed acyclic graphs with the pc-algorithm. Journal of Machine Learning Research, 8(3), 2007.

蔡瑞初，刘跃群，黄正婷，黄晓楷，陈薇 & 郝志峰. 融合先验约束的拓扑霍克斯过程格兰杰因果发现算法. 计算机应用研究, 39(1668-1672), 2022. ISSN 1001-3695. doi: 10.19734/j.issn.1001-3695.2021.12.0642.

Ruichu Cai, Siyu Wu, Jie Qiao, Zhifeng Hao, Keli Zhang, and Xi Zhang. Thp: Topological hawkes processes for learning causal structure on event sequences, 2022.