Learning Diffusions under Uncertainty

Hao Huang[1], Qian Yan[1], Keqi Han[1], Ting Gan[1], Jiawei Jiang[1], Quanqing Xu[2], Chuanhui Yang[2]

[1]School of Computer Science, Wuhan University, China
[2]OceanBase, Ant Group, China
{haohuang, qy, hankeqi, ganting, jiawei.jiang}@whu.edu.cn, {xuquanqing.xqq, rizhao.ych}@oceanbase.com

Abstract

To infer a diffusion network based on observations from historical diffusion processes, existing approaches assume that observation data contain exact occurrence time of each node infection, or at least the eventual infection statuses of nodes in each diffusion process. They determine potential influence relationships between nodes by identifying frequent sequences, or statistical correlations, among node infections. In some real-world settings, such as the spread of epidemics, tracing exact infection times is often infeasible due to a high cost; even obtaining precise infection statuses of nodes is a challenging task, since observable symptoms such as headache only partially reveal a node's true status. In this work, we investigate how to effectively infer a diffusion network from observation data with uncertainty. Provided with only probabilistic information about node infection statuses, we formulate the problem of diffusion network inference as a constrained nonlinear regression w.r.t. the probabilistic data. An alternating maximization method is designed to solve this regression problem iteratively, and the improvement of solution quality in each iteration can be theoretically guaranteed. Empirical studies are conducted on both synthetic and real-world networks, and the results verify the effectiveness and efficiency of our approach.

Introduction

The spread of viewpoints, rumors, and diseases are often modelled as probabilistic processes over a diffusion network. In the network, a directed edge represents a parent-child influence relationship, which indicates that the parent node can influence the child node with a certain probability. In most cases, such influence relationships are not naturally visible or traceable, and we only observe a set of diffusion results from a limited number of historical diffusion processes (Gan et al. 2021). Diffusion network inference aims to infer the diffusion network structure (i.e., the topology of influence relationships) from the observation data. This problem has received considerable attention in areas such as information propagation (He et al. 2015), viral marketing (Leskovec, Adamic, and Huberman 2007), and epidemic prevention (Wallinga and Teunis 2004) since the inferred diffusion network structure enables an intuitive understanding of the underlying interactions between nodes, and is essential for developing strategies to control future diffusion processes (Huang et al. 2021).

Most existing approaches assume that observation data contain the exact times when node infections occurred. With this temporal information, they determine potential influence relationships between nodes by identifying frequent sequences of node infections, since nodes infected sequentially within a short time interval are considered more likely to possess influence relationships (Gomez-Rodriguez, Leskovec, and Krause 2010; Myers and Leskovec 2010; Gomez-Rodriguez, Balduzzi, and Schölkopf 2011; Gomez-Rodriguez and Schölkopf 2012; Du et al. 2012; Gomez-Rodriguez, Leskovec, and Schölkopf 2013a,b; Daneshmand et al. 2014; Wang et al. 2014). Nonetheless, monitoring every node constantly during each diffusion process to obtain such temporal information of node infections is often expensive in practice. Therefore, some other approaches aim to carry out diffusion network inference without temporal information, using only the eventual infection statuses of nodes observed at the end of each diffusion process (Amin, Heidari, and Kearns 2014; Huang et al. 2019, 2021, 2022, 2023a; Han et al. 2020; Gan et al. 2021). Toward this, they measure the statistical correlations of node infections, and identify influence relationships by checking which node pairs have high infection correlations.

In this paper, we study the problem of diffusion network inference in a less idealized and more realistic setting, i.e., only probabilistic information about node infection statuses is provided. This uncertainty is interpreted differently in various contexts. For example, in epidemic containment, it is difficult to confirm the infection statuses of outpatients based on observable symptoms such as headache and fatigue, since there is a certain probability that these symptoms may be caused by other reasons like lack of sleep; in viral marketing campaigns, it is often the case that the respondents are more prone to probabilistic feedback than a clear-cut answer. To the best of our knowledge, only one existing work (Sefer and Kingsford 2015) has partially addressed the problem of inferring diffusion networks based on probabilistic data. In addition to the probabilistic information about node infection statuses, that work further requires prior knowledge on the transmission probabilities between different node statuses, and the temporal information of node infections.

Aiming at a more general solution to inferring diffusion networks from probabilistic data, we formulate the problem as a constrained nonlinear regression w.r.t. the probabilistic data, and propose an effective and efficient algorithm called PIND (a re-ordered acronym of **I**nferring **N**etworks from **P**robabilistic **D**ata) to solve the regression problem iteratively. In each iteration, an alternating maximization method is adopted to update the estimate on the probability of the existence of each potential influence relationship, and the influence relationship strengthens. The quality of the estimate is theoretically guaranteed to improve across the iterations. After the convergence of iterations, the most probable influence relationships in the objective diffusion network can be easily identified based on the estimate.

The remainder of the paper is organized as follows. We first present our problem statement, and then introduce our proposed PIND algorithm, followed by reporting experimental results and our findings before concluding the paper.

Problem Statement

A diffusion network is represented as a directed graph $G = (V, E)$, where $V = \{v_1, ..., v_n\}$ is the set of n nodes in the network, and E is the set of directed edges (i.e., influence relationship) between the nodes. A directed edge $(v_j, v_i) \in E$ from a parent node $v_j \in V$ to a child node $v_i \in V$ indicates that when v_j is infected and v_i is uninfected, v_j will infect v_i with a certain probability α_{ji}, which is known as infection propagation probability and can be regarded as the strength of the influence relationship from v_j to v_i. We assume that the diffusion processes on G follow the Independent Cascade (IC) model, in which each infected parent node tries to infect each of its uninfected children with corresponding infection propagation probability independently. This assumption is customarily adopted in prior art, and we would like to point out that our proposed algorithm can be extended to other propagation models such as the linear threshold model.

In the problem of diffusion network inference, the node set V is given, while the network structure, i.e., directed edge set E, is unknown, neither the infection propagation probabilities w.r.t. the structure. To infer the structure of a diffusion network, a set S of diffusion results observed from a number of historical diffusion processes on the network is required. In this paper, we assume that the diffusion results S contain only the probabilistic information of node infection statuses, i.e., $S = \{s_i^\ell \mid i \in \{1, ..., n\}, \ell \in \{1, ..., \beta\}\}$, where β is the number of historical diffusion processes, and $s_i^\ell \in [0, 1]$ refers to the probability that node v_i is infected in the ℓ-th historical diffusion process. Then, our problem statement for diffusion network inference with probabilistic data can be formulated as follows.

***Given*:** a set $S = \{S^1, ..., S^\beta\}$ of diffusion results observed on a diffusion network G at the end of β diffusion processes, where $S^\ell = \{s_1^\ell, ..., s_n^\ell\}$ ($\ell \in \{1, ..., \beta\}$) records the infection probability s_i^ℓ of each node $v_i \in V$ in the ℓ-th diffusion process.

***Infer*:** the edge set E of the diffusion network G.

The PIND Algorithm

In this section, we first elaborate how to estimate the existence of influence relationships in a probabilistic way, and then present how to reduce redundant computation in the influence relationship estimation with a pruning method for candidate parent nodes. We conclude this section with a complexity analysis on our approach.

Estimation of Influence Relationships

Objective Function Let F_i be the set of parent nodes of node $v_i \in V$ in objective diffusion network, according to IC model, the log-likelihood of v_i being infected by any node in set F_i in the ℓ-th diffusion process can be calculated as

$$\log(1 - \prod_{v_j \in F_i} (1 - s_j^\ell \alpha_{ji})), \tag{1}$$

the log-likelihood of v_i not being infected by any node in set F_i in the ℓ-th diffusion process can be calculated as

$$\log(\prod_{v_j \in F_i} (1 - s_j^\ell \alpha_{ji})) = \sum_{v_j \in F_i} \log(1 - s_j^\ell \alpha_{ji}). \tag{2}$$

Since the probability that node v_i is infected in the ℓ-th historical diffusion process is s_i^ℓ and the probability that node v_i is not infected in the ℓ-th historical diffusion process is $1 - s_i^\ell$, then the log-likelihood of observation s_i^ℓ is

$$\begin{aligned} \mathcal{G}(s_i^\ell) =& s_i^\ell \log\big(1 - \prod_{v_j \in F_i} (1 - s_j^\ell \alpha_{ji})\big) + \\ & (1 - s_i^\ell) \sum_{v_j \in F_i} \log(1 - s_j^\ell \alpha_{ji}). \end{aligned} \tag{3}$$

Let variable $x_{ji} \in \{0, 1\}$ indicate whether there is a directed edge from node v_j to node v_i (1 for yes and 0 for no), and $C_i = V \setminus \{v_i\}$ denote the set of all possible candidate parent nodes of v_i if we can infer each variable x_{ji} ($v_i \in V, v_j \in C_i$) accurately, then according to Eq. (3), the following equation should be satisfied.

$$\begin{aligned} \mathcal{G}(s_i^\ell) =& s_i^\ell \log\big(1 - \prod_{v_j \in C_i} (1 - s_j^\ell \alpha_{ji})^{x_{ji}}\big) + \\ & (1 - s_i^\ell) \sum_{v_j \in C_i} x_{ji} \log(1 - s_j^\ell \alpha_{ji}). \end{aligned} \tag{4}$$

Let x be the collection of all x_{ji} ($v_i \in V, v_j \in C_i$), and α be the collection of all α_{ji} ($v_i \in V, v_j \in C_i$), then the overall log-likelihood of data S is

$$\mathcal{G}(S) = \sum_{\ell=1}^{\beta} \sum_{i=1}^{n} \mathcal{G}(s_i^\ell), \tag{5}$$

which is a function w.r.t. x and α. Let

$$\mathcal{L}(x, \alpha) = \mathcal{G}(S). \tag{6}$$

A greater value of $\mathcal{L}(x, \alpha)$ indicates that the corresponding x and α are more close to the truth. Therefore, our goal is

to find optimal x and α that maximize the value of $\mathcal{L}(x, \alpha)$, which can be formulated as follows.

$$\begin{aligned}\max \mathcal{L} = &\sum_{\ell=1}^{\beta}\sum_{i=1}^{n} s_i^\ell \log\big(1 - \prod_{v_j \in C_i}(1 - s_j^\ell \alpha_{ji})^{x_{ji}}\big) + \\ &\sum_{\ell=1}^{\beta}\sum_{i=1}^{n}(1 - s_i^\ell) \sum_{v_j \in C_i} x_{ji} \log(1 - s_j^\ell \alpha_{ji}) \\ &\text{s.t. } x_{ji} \in \{0, 1\}, \alpha_{ji} \in [0, 1], \forall i, j.\end{aligned} \tag{7}$$

The constrained maximization problem above is a nonlinear mixed integer programming, which is difficult to be solved directly. Therefore, we relax the integer constraint on each x_{ji} by allowing $x_{ji} \in [0, 1]$ ($j, i \in \{1, ..., n\}$) such that continuous programming methods can be applied to this problem, where the continuous value of each x_{ji} denotes the probability that there is a directed edge from node v_j to node v_i. This kind of relaxations are commonly used to solve the problem of integer programming. Then, the problem in Eq. (7) is relaxed and reformulated as follows.

$$\begin{aligned}\max \mathcal{L} = &\sum_{\ell=1}^{\beta}\sum_{i=1}^{n} s_i^\ell \log\big(1 - \prod_{v_j \in C_i}(1 - s_j^\ell \alpha_{ji})^{x_{ji}}\big) + \\ &\sum_{\ell=1}^{\beta}\sum_{i=1}^{n}(1 - s_i^\ell) \sum_{v_j \in C_i} x_{ji} \log(1 - s_j^\ell \alpha_{ji}) \\ &\text{s.t. } x_{ji} \in [0, 1], \alpha_{ji} \in [0, 1], \forall i, j.\end{aligned} \tag{8}$$

Solving Method To solve the problem in Eq. (8), the straightforward method is solving equations $\frac{\partial \mathcal{L}}{\partial x} = 0$ and $\frac{\partial \mathcal{L}}{\partial \alpha} = 0$, where $\frac{\partial \mathcal{L}}{\partial x}$ and $\frac{\partial \mathcal{L}}{\partial \alpha}$ are the derivatives of $\mathcal{L}(x, \alpha)$ w.r.t. x and α, respectively, which can be calculated as

$$\frac{\partial \mathcal{L}}{\partial \alpha_{ji}} = \sum_{\ell=1}^{\beta} \frac{x_{ji} s_j^\ell}{1 - s_j^\ell \alpha_{ji}} L_i^\ell, \tag{9}$$

$$\frac{\partial \mathcal{L}}{\partial x_{ji}} = -\sum_{\ell=1}^{\beta} \log(1 - s_j^\ell \alpha_{ji}) L_i^\ell. \tag{10}$$

where

$$L_i^\ell = s_i^\ell \frac{\prod_{v_k \in C_i}(1 - s_k^\ell \alpha_{ki})^{x_{ki}}}{1 - \prod_{v_k \in C_i}(1 - s_k^\ell \alpha_{ki})^{x_{ki}}} - (1 - s_i^\ell). \tag{11}$$

Nevertheless, directly solving equations $\frac{\partial \mathcal{L}}{\partial x} = 0$ is still difficult, and its solution may not satisfy the constraints in Eq. (8). To address this issue, we adopt an alternating maximization method, which works as follows.

Step 1. Initializing α and x under constraints that $x_{ji} \in [0, 1]$, $\alpha_{ji} \in [0, 1]$, $\forall i, j$.

Step 2. Selecting an appropriate update direction y (based on $\frac{\partial \mathcal{L}}{\partial x}$) and an appropriate step size θ, and updating x as

$$x \leftarrow x + \theta y. \tag{12}$$

Step 3. Selecting an appropriate update direction z (based on $\frac{\partial \mathcal{L}}{\partial \alpha}$) and an appropriate step size λ, and updating α as

$$\alpha \leftarrow \alpha + \lambda z. \tag{13}$$

Step 4. Repeating Steps 2 and 3 until convergence.

In the above alternating maximization method, how to selecting appropriate update directions and step sizes for x and α is of central importance. Let $x^{(T)}$ be the value of x, and $\alpha^{(T)}$ be the value of α, in the T-th iteration of Steps 2 and 3, $f(x) = \mathcal{L}(x, \alpha^{(T)})$ be a function w.r.t. x, and $g(x) = \mathcal{L}(x^{(T)}, \alpha)$ be a function w.r.t. α, then we have

$$\frac{\partial f}{\partial x} = \frac{\partial \mathcal{L}}{\partial x} \mid_{\alpha = \alpha^{(T)}}, \tag{14}$$

$$\frac{\partial g}{\partial \alpha} = \frac{\partial \mathcal{L}}{\partial \alpha} \mid_{x = x^{(T)}}. \tag{15}$$

As the direction of gradient is the steepest ascent direction, we can utilize the gradient to update x and α. Furthermore, to make the constraints in Eq. (8) satisfied, we modify the gradient and select the update directions y and z of variables x and α as follows.

$$y_{ji} = \begin{cases} 0, \ \text{if } x_{ji}^{(T)} = 0, \frac{\partial \mathcal{L}}{\partial x_{ji}} \mid_{(x,\alpha)=(x^{(T)}, \alpha^{(T)})} < 0; \\ 0, \ \text{if } x_{ji}^{(T)} = 1, \frac{\partial \mathcal{L}}{\partial x_{ji}} \mid_{(x,\alpha)=(x^{(T)}, \alpha^{(T)})} > 0; \\ \frac{\partial \mathcal{L}}{\partial x_{ji}} \mid_{(x,\alpha)=(x^{(T)}, \alpha^{(T)})}, \ otherwise. \end{cases} \tag{16}$$

$$z_{ji} = \begin{cases} 0, \ \text{if } \alpha_{ji}^{(T)} = 0, \frac{\partial \mathcal{L}}{\partial \alpha_{ji}} \mid_{(x,\alpha)=(x^{(T)}, \alpha^{(T)})} < 0; \\ 0, \ \text{if } \alpha_{ji}^{(T)} = 1, \frac{\partial \mathcal{L}}{\partial \alpha_{ji}} \mid_{(x,\alpha)=(x^{(T)}, \alpha^{(T)})} > 0; \\ \frac{\partial \mathcal{L}}{\partial \alpha_{ji}} \mid_{(x,\alpha)=(x^{(T)}, \alpha^{(T)})}, \ otherwise. \end{cases} \tag{17}$$

Next, we discuss how to select step sizes θ and λ. To this end, first, we define θ_{ji} and λ_{ji} as follows.

$$\theta_{ji} = \begin{cases} \frac{x_{ji}}{-y_{ji}}, & \text{if } y_{ji} < 0; \\ \frac{1 - x_{ji}}{y_{ji}}, & \text{if } y_{ji} > 0; \\ +\infty, & \text{if } y_{ji} = 0. \end{cases} \tag{18}$$

$$\lambda_{ji} = \begin{cases} \frac{\alpha_{ji}}{-z_{ji}}, & \text{if } z_{ji} < 0; \\ \frac{1 - \alpha_{ji}}{z_{ji}}, & \text{if } z_{ji} > 0; \\ +\infty, & \text{if } z_{ji} = 0. \end{cases} \tag{19}$$

Let

$$\theta' = \min\{\theta_{ji} \mid i \in \{1, ..., n\}, v_j \in C_i\}, \tag{20}$$

relationship $\theta \leqslant \theta'$ can guarantee that if $x^{(T)}$ is feasible (i.e., $x_{ji}^{(T)} \in [0, 1]$, $\forall i, j$), then $x^{(T)} + \theta y$ is also feasible. Similarly, let

$$\lambda' = \min\{\lambda_{ji} \mid i \in \{1, ..., n\}, v_j \in C_i\}, \tag{21}$$

relationship $\lambda \leqslant \lambda'$ can guarantee that if $\alpha^{(T)}$ is feasible, then $\alpha^{(T)} + \lambda z$ is also feasible. In other words, θ' and

λ' should be the upper bounds of step sizes θ and λ, respectively. Furthermore, to guarantee the value of objective function $\mathcal{L}$ becoming greater after the updating of x and α, we could select step sizes θ and λ based on the following two lemmas. Note that all the proofs related to this section are given in the supplementary material (Huang et al. 2023b).

lemma 1. *If $y \neq 0$ (i.e., $\exists y_{ji} \neq 0$), then there exists a nonnegative integer m such that $f(x^{(T)} + \frac{\theta'}{2^m} y) > f(x^{(T)})$.*

lemma 2. *If $z \neq 0$ (i.e., $\exists z_{ji} \neq 0$), then there exist a nonnegative integer k such that $g(\alpha^{(T)} + \frac{\lambda'}{2^k} z) > g(\alpha^{(T)})$.*

Based on Lemmas 1 & 2, we can set the step sizes θ and λ as $\theta = \frac{\theta'}{2^m}$ and $\lambda = \frac{\lambda'}{2^k}$, and then gradually increase the values of nonnegative integers m and k until relationships $f(x^{(T)} + \frac{\theta'}{2^m} y) > f(x^{(T)})$ and $g(\alpha^{(T)} + \frac{\lambda'}{2^k} z) > g(\alpha^{(T)})$ are satisfied. In this way, the following two theorems can guarantee that the value of objective function $\mathcal{L}$ will become greater in the next iteration of Steps 2 & 3.

Theorem 1. *Let $x^{(T)}$ and $\alpha^{(T)}$ be the current values of x and α, respectively, and m be a great enough nonnegative integer such that $f(x^{(T)} + \frac{\theta'}{2^m} y) < f(x^{(T)})$, if we update $x^{(T)}$ to $x^{(T+1)}$ by Eq. (12), then we have*

$$\mathcal{L}(x^{(T+1)}, \alpha^{(T)}) \geqslant \mathcal{L}(x^{(T)}, \alpha^{(T)}), \tag{22}$$

where the equal sign holds if and only if $x^{(T)} = x^{(T+1)}$.

Theorem 2. *Let $x^{(T)}$ and $\alpha^{(T)}$ be the current values of x and α, respectively, and k be a great enough nonnegative integer such that $g(\alpha^{(T)} + \frac{\lambda'}{2^k} z) > g(\alpha^{(T)})$, if we update $\alpha^{(T)}$ to $\alpha^{(T+1)}$ by Eq. (13), then we have*

$$\mathcal{L}(x^{(T)}, \alpha^{(T+1)}) \geqslant \mathcal{L}(x^{(T)}, \alpha^{(T)}), \tag{23}$$

where the equal sign holds if and only if $\alpha^{(T)} = \alpha^{(T+1)}$.

With the selected update directions and step sizes, the problem in Eq. (8) can be solved by applying the alternating maximization method. Let (x^*, α^*) be the solution found by this method, then x^* indicates our estimated probabilities of the existence of each potential parent-child influence relationship. As the original problem in Eq. (7) aims at an integral solution for x, we carry out the following two steps: (1) By repeatedly sampling the value of each $x_{ji} \in \{0, 1\}$ in x from probability $x^*_{ji} \in [0, 1]$, we obtain a set of samples $\{\hat{x}_1, ..., \hat{x}_r\}$ for x, where r is the times of sampling; (2) we select an optimal sample $\hat{x}^*$ from $\{\hat{x}_1, ..., \hat{x}_r\}$ by solving

$$\hat{x}^* = \arg\max_{\hat{x}_i, i \in \{1,...,r\}} \mathcal{L}(\hat{x}_i, \alpha^*). \tag{24}$$

Then, $\hat{x}^*$ is our estimated integral solution for x.

Pruning of Candidate Parent Nodes

Given the fact that the infections of nodes are only caused by their parent nodes with a certain probability, the infections of the parent nodes and corresponding child nodes should have relatively great positive correlations. In contrast, if the infection statuses of two nodes have no or an extremely low positive correlation, there is a very low probability that these two nodes have an influence relationship between them. To quantify the correlations of node infections, mutual information (abbreviated as MI) is a commonly used criterion (Huang et al. 2019). In our problem, it can be calculated as

$$\begin{aligned} &MI(X_i, X_j) \\ &= \sum_{a=0}^{1} \sum_{b=0}^{1} p(X_i = a, X_j = b) \ln \frac{p(X_i = a, X_j = b)}{p(X_i = a) p(X_j = b)}, \end{aligned} \tag{25}$$

where X_i is the infection status variable of node v_i,

$$\begin{aligned} p(X_i = a) &= \frac{1}{\beta} \sum_{\ell=1}^{\beta} p(X_i^\ell = a), \\ p(X_j = b) &= \frac{1}{\beta} \sum_{\ell=1}^{\beta} p(X_j^\ell = b), \\ p(X_i = a, X_j = b) &= \frac{1}{\beta} \sum_{\ell=1}^{\beta} p(X_i^\ell = a) p(X_j^\ell = b), \end{aligned} \tag{26}$$

and $p(X_i^\ell = 0) = 1 - s_i^\ell$, $p(X_i^\ell = 1) = s_i^\ell$.

A greater MI value indicates a stronger correlation between the infection statuses of nodes v_i and v_j. For a node that has no influence relationship with v_i, its infection status often has no (or very low) correlations to the infection status of v_i, resulting in a very small MI value (close to 0).

Inspired by this line of reasoning, we screen out insignificant candidate parent nodes for each node by the following two steps. First, we calculate the MI value for each two nodes in the network. Then, we perform a modified K-means algorithm with $K = 2$ and one of the two means fixed at 0 through all iterations of K-means, to efficiently partition all MI values into two groups, where one group has a small mean close to 0. Let η be the largest value in the group with a mean close to 0. Then, for each $MI(X_i, X_j) \leqslant \eta$, we regard the corresponding node v_j as an insignificant candidate parent node for node v_i and exclude v_j from the candidate parent node set C_i of v_i.

This heuristic pruning method should be performed before the estimation of influence relationship, to screen out insignificant candidate parent nodes from the set C_i of the candidate parent nodes for each node v_i, and enables the PIND algorithm to focus on influence relationships that are more likely to exist in the real diffusion network.

Complexity Analysis

PIND algorithm consists of the following two parts. (1) In the phase of pruning candidate parent nodes, calculating MI values requires $O(\beta n^2)$ time, and performing K-means on these MI values takes $O(\tau n^2)$ time, where β is the number of historical diffusion processes, n is the number of network nodes, and τ is the number of iterations of K-means. (2) In the phase of estimating influence relationship, the optimization problem in Eq. (8) is iteratively solved by an alternating maximization method. In each iteration of this method, calculating the partial derivatives can be finished within $O(\beta c^2 n)$ time, and updating variables x and α takes about $O(cn)$ time, where c is the upper bound of the number of candidate parent nodes of each node in the network, i.e.,

Graphs	Number n of Nodes	Average Degree
G1–G5	1000, 1500, 2000, 2500, 3000	4
G6–G10	2000	2, 3, 4, 5, 6

Table 1: Properties of LFR benchmark graphs.

$c = \max\{|C_i| \mid i = 1, ..., n\} \ll n$. Therefore, solving the problem in Eq. (8) takes $O(t\beta c^2 n)$ time, where t is the number of iterations of alternating maximization method.

In summary, the overall time complexity of PIND algorithm is $O(\beta n^2 + \tau n^2 + t\beta c^2 n)$.

Experimental Evaluation

In this section, we first introduce the experimental setup, and then evaluate our PIND algorithm on synthetic and real-world networks. We investigate the effects of diffusion network size, the average degree of diffusion network, the uncertainty of observed infection data, and the amount of diffusion processes, on the accuracy and running time of PIND. All algorithms in the experiments are implemented in Python, running on a MacBook Pro with Intel Core i5-1038NG7 CPU at 2.00GHz and 16GB RAM. The source code of PIND and the data used in the experiments are available at https://github.com/DiffusionNetworkInference/PIND.

Experimental Setup

Network. We adopt LFR benchmark graphs (Lancichinetti, Fortunato, and Radicchi 2008) as the synthetic diffusion networks. By setting different generation parameters, such as the number n of nodes and the average degree of each node, we generate a series of LFR benchmark graphs with properties summarized in Table 1. Similar synthetic network generation methods are commonly used in existing studies (Huang et al. 2019, 2021; Gan et al. 2021; Han et al. 2020). In addition, we also adopt two commonly used real-world microblogging networks (Wang et al. 2014), namely, (1) DUNF, which contains 750 users and 2974 following relationships, and (2) DPU, which contains 1038 users and 11385 following relationships.

Infection Data. The diffusion results $S = \{S_1, ..., S_\beta\}$ can be generated by simulating β times of diffusion processes on each network with randomly selected initially infected nodes in each simulation (the ratio of initially infected nodes is 15%). In each diffusion process, each infected node tries to infect its uninfected child nodes with a certain probability, which subjects to a Gaussian distribution with a mean of 0.3 and a standard deviation of 0.05, to make about 95% of infection propagation probabilities within a range from 0.2 to 0.4. Besides diffusion results S, the cascades (i.e., the exact times when node infections occurred) are also recorded for cascade-based tested algorithms in the experiments. Similar generation methods for infection data are commonly used in existing studies (Gomez-Rodriguez, Leskovec, and Krause 2010; Amin, Heidari, and Kearns 2014; Wang et al. 2014; Huang et al. 2019, 2021, 2023a; Han et al. 2020; Gan et al. 2021; Yan et al. 2017). To add uncertainty into the infection data, for each exact node infection status $s \in \{0, 1\}$, we replace it with $|s - u|$, where u is a random uncertainty factor and its value subjects to a Gaussian distribution with a mean μ and a standard deviation of 0.1 (if $\mu = 0$, the standard deviation is 0). All generated infection data are stored by OceanBase (Yang et al. 2023).

Performance Criterion. We evaluate the performance of PIND algorithm in terms of the accuracy of structure inference. For an inferred diffusion network, the accuracy of structure inference can be measured by the F-score of the inferred directed edges: $F\text{-}score = \frac{2 \cdot precision \cdot recall}{precision + recall}$, where $precision = \frac{N_{TP}}{N_{TP}+N_{FP}}$, $recall = \frac{N_{TP}}{N_{TP}+N_{FN}}$, N_{TP} is the number of true positives, i.e., the true edges which are correctly inferred by the algorithm; N_{FP} is the number of false positives, i.e., the wrong inferred edges which are not in the real network; and N_{FN} is the number of false negatives, i.e., the true edges which are not correctly inferred by the algorithm. On each network, we execute PIND 10 times, and report the average F-score as the accuracy of PIND on this network (the corresponding standard deviations are always less than 0.001 in all experiments).

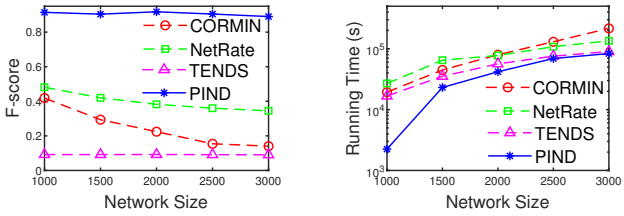
Benchmark Algorithms. We compare PIND with (1) CORMIN (Sefer and Kingsford 2015), which is, to the best of our knowledge, the only existing approach to diffusion network inference with probabilistic data, (2) NetRate (Gomez-Rodriguez, Balduzzi, and Schölkopf 2011), which is a classical cascade-based approach to diffusion network inference, and (3) TENDS (Han et al. 2020), which is a high-performance node infection status-based approach to diffusion network inference. In our PIND algorithm, the number r of sampling rounds for x is set to 100, and the stop condition for the iterative updates of x and α is that the variations of each $x_{ji} \in x$ and each $\alpha_{ji} \in \alpha$ are less than 0.01. Since CORMIN requires to know the node statuses at different timestamps, we provide it with the temporal information of node infections. Since NetRate takes only cascades as input, the uncertainties of node infection statuses do not work for NetRate. Therefore, we execute NetRate with exact cascades in the experiments. Since TENDS cannot directly deal with probabilistic data, we repeatedly sample 0/1 value 50 times from each probability s_i^ℓ ($i \in \{1, ..., n\}$, $\ell \in \{1, ..., \beta\}$) to obtain 50 groups of samples of exact node infection statuses, and then report the best F-score of TENDS on these 50 groups of samples as the accuracy of this approach.

Effect of Diffusion Network Size

To study the effect of diffusion network size on algorithm performance, we adopt five synthetic networks, i.e., G1–G5, whose sizes vary from 1000 to 3000. We simulate 300 times of diffusion processes on each network (i.e., $\beta = 300$), and set the mean μ of uncertainty factor to 0.3.

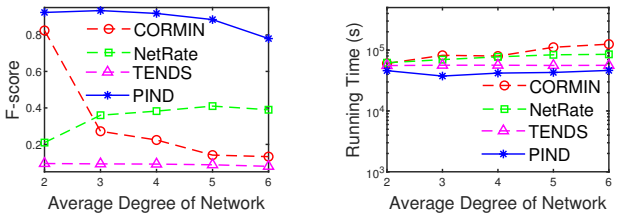
Fig. 1 illustrates the F-score and execution time of each tested algorithm, from which we can observe that PIND outperforms CORMIN, NetRate and TENDS in terms of accuracy, and we can also have the following observations.

(1) The accuracy of PIND and TENDS is reasonably insensitive to diffusion network size. TENDS's insensi-



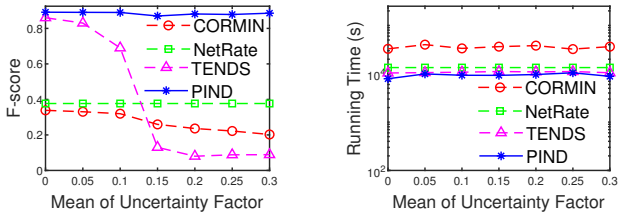
(a) Accuracy (b) Running time

Figure 1: Effect of diffusion network size.

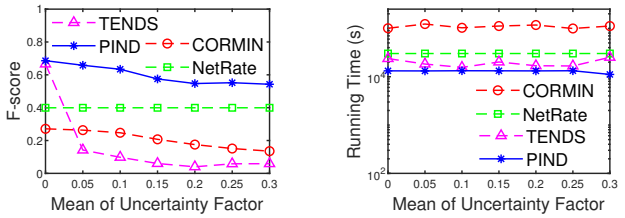


(a) Accuracy (b) Running time

Figure 2: Effect of network's average degree.



(a) Accuracy on DUNF (b) Running time on DUNF



(c) Accuracy on DPU (d) Running time on DPU

Figure 3: Effect of infection data uncertainty.

tivity to diffusion network size has also been verified in existing study (Han et al. 2020). Nonetheless, TENDS's accuracy in this experiment is poor. This is because the samples of node infection statuses are far from the real situation, leading to large inference errors for TENDS. This observation indicates that straightforward sampling from the probabilistic data is not a very suitable approach to dealing with probabilistic data. In contrast, in our PIND algorithm, the theoretical guarantee on the improvement of its solution quality helps it achieve a reasonably robust accuracy on networks with different sizes.

(2) Larger network sizes degrade the accuracy of CORMIN and NetRate. This is because CORMIN and NetRate infer influence relationships by checking whether the infections of two nodes are often within a time interval in historical diffusion processes. Nonetheless, a few nodes may be infected at nearly the same time, even though they have no direct influence relationship. In a larger network, more nodes are likely to be involved in each diffusion process, causing more aforementioned phenomena, which will result in more false positives in the inference results of CORMIN and NetRate.

(3) The running time of each algorithm increases with the diffusion network size. PIND has better running time performance than CORMIN and NetRate. Although TENDS shows comparable efficiency to PIND on larger diffusion networks, it has a poor accuracy as it has no effective strategy to resist to the uncertainty in infection data.

In addition, based on extensive testing on larger networks, we have found that the running time of CORMIN and NetRate rapidly increase with the growth of network size, and soon exceed acceptable levels. Given this fact, we have selected networks with sizes varying from 750 to 3000. Most existing related work adopt similar network sizes.

Effect of Network's Average Degree

To study the effect of the average degree of diffusion network on algorithm performance, we test the algorithms on five synthetic networks with the same size, i.e., G6–G10, whose average degree varies from 2 to 6. We simulate 300 times of diffusion processes on each network (i.e., $\beta = 300$), and set the mean μ of uncertainty factor to 0.3.

Fig. 2 illustrates the F-score and running time of each tested algorithm, from which we can observe that PIND has the best accuracy compared with other tested algorithms, and we can also have the following observations.

(1) With the increase of the average degree of diffusion network, the accuracy of CORMIN and PIND decrease. The accuracy of NetRate increases when the average degree increases from 2 to 5 and then tends to decrease when the average degree reaches 6. The reason behind is that a greater average degree often brings more complicated influence relationships between nodes, and thus adding complexity to the task of diffusion network inference.

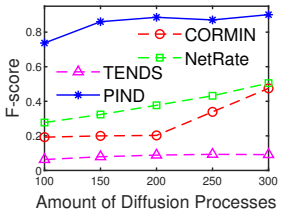
(2) The running time of each tested algorithm increases with the growth of average degree, and PIND is faster than CORMIN, NetRate and TENDS.

Effect of Infection Data Uncertainty

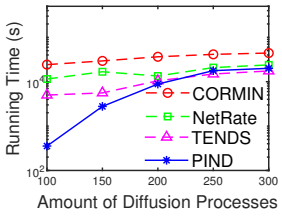
To study the effect of the uncertainty of infection data on algorithm performance, we test the algorithms on two real-world networks, i.e., DUNF and DPU, varying the mean μ of uncertainty factor from 0 to 0.3 (with $\beta = 200$).

Fig. 3 illustrates the F-score and running time of each tested algorithm on DUNF and DPU. From the figure we can have the following observations.

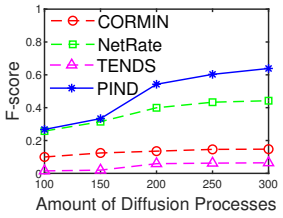
(1) Compared with other tested algorithms, PIND often shows a significant advantage on accuracy, while TENDS achieves a very close accuracy to PIND when $\mu \leqslant 0.05$ on DUNF and when $\mu = 0$ on DPU. The reason behind is that with a small enough mean of uncertainty factor, the sampled node infection statuses for TENDS will be equal or close



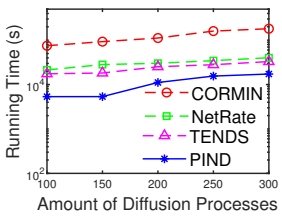
(a) Accuracy on DUNF



(b) Running time on DUNF



(c) Accuracy on DPU



(d) Running time on DPU

Figure 4: Effect of the amount of diffusion processes.

enough to the real situation. With correct infection data, TENDS is able to achieve a high accuracy performance.

(2) A higher uncertainty tends to degrade the accuracy of PIND, CORMIN and TENDS. This is because a higher uncertainty makes measuring the correlations between node infections more difficult. When the uncertainty exceed a threshold, for example, $\mu \geqslant 0.5$, the distinction between infected and uninfected statues will totally lose, then it becomes almost impossible to recover the real influence relationships. Note that the accuracy of NetRate does not change with uncertainty, since NetRate uses exact cascades for diffusion network inference in this experiment.

(3) PIND outperforms CORMIN in running time performance, and is often relatively faster than NetRate and TENDS. In addition, the uncertainty has mild effect on the running time of tested algorithms. This is because the time complexity of each tested algorithm is mainly dominated by the network size and the amount of diffusion processes.

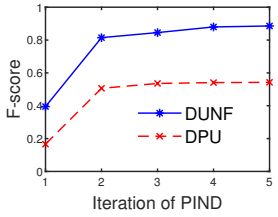
Effect of Diffusion Process Amount

To study the effect of diffusion process amount on algorithm performance, we test the algorithms on DUNF and DPU with different number β of diffusion processes, varying from 100 to 300. For the diffusion results obtained with each β, we set the mean μ of uncertainty factor to 0.3.

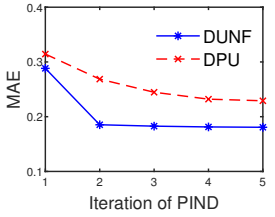
Fig. 4 illustrates the F-score and running time of each tested algorithm on DUNF and DPU. From the figure we can have the following observations.

(1) A greater amount of diffusion processes often helps the algorithms achieve higher accuracy. This is because more diffusion processes tend to involve more nodes, and activate more influence relationships (i.e., infections spread through these influence relationships), which enable the algorithms to learn a more complete network structure.

(2) Compared with other algorithms, PIND achieves significantly better accuracy in all but two settings, i.e., $\beta = 100$ and $\beta = 150$ on DPU. This is because when $\beta \leqslant 150$,



(a) Accuracy



(b) MAE of learned $\boldsymbol{\alpha}_{ji}$

Figure 5: Effect of the iterations of PIND.

the amount of diffusion processes are insufficient to activate enough influence relationships in DPU, and thus degrade the accuracy of PIND in these settings.

(3) The running time of each algorithm increase with the amount of diffusion processes. PIND has significantly better running time performance than CORMIN, and tends to be relatively more efficient than NetRate. Although TENDS shows comparable (or even slightly better) efficiency to PIND on larger diffusion networks, it suffers from a poor accuracy as it can not effectively deal with probabilistic data.

Effect of Iterations of PIND

To study the effect of iterations on the performance of PIND, we execute PIND on DUNF and DPU (with $\beta = 200$, $\mu = 0.3$) and report the accuracy of its latest inference result at the end of each iteration.

Fig. 5 (a) illustrates the F-score of PIND at the end of each iteration, from which we can observe that the accuracy of PIND can be improved with more iterations, and shows a fast convergence property.

The direct reason behind this is that PIND can accurately estimate the existence of influence relationships. Another important reason is that PIND can accurately infer infection propagation probabilities. To demonstrate this effectiveness, Fig. 5 (b) illustrates the MAE (Mean Absolute Error) of infection propagation probabilities learned by PIND at the end of each iteration. A lower MAE indicates a higher accuracy. From Fig. 5 (b), we can observe that the MAE of learned α_{ji} can also be improved with more iterations. For comparison, the corresponding MAE values of NetRate on DUNF and DPU are 0.2822 and 0.2791, respectively, which are significantly higher than that of PIND.

Conclusion

In this paper, we have investigated the problem of how to infer a diffusion network using only the probabilistic data of the node infection statuses observed in historical diffusion processes. Towards this, we have formulated the problem as a constrained nonlinear regression w.r.t. the probabilistic data, and proposed an effective and efficient algorithm, PIND, to solve the regression problem in an iterative way. The improvement of solution quality in each iteration can be theoretically guaranteed. Extensive experimental results on both synthetic and real-world networks have verified the effectiveness and efficiency of PIND.

Acknowledgments

This work was supported by the National Key Research and Development Program of China (2022YFB3105000), the National Natural Science Foundation of China (61976163), the Key R&D Program of Hubei Province (2023BAB077), and the Fundamental Research Fund for Central Universities (2042023kf0219). This work was supported by Ant Group. Qian Yan is the corresponding author.

References

Amin, K.; Heidari, H.; and Kearns, M. 2014. Learning from Contagion(Without Timestamps). In *Proceedings of the 31st International Conference on Machine Learning (ICML 2014)*, 1845–1853.

Daneshmand, H.; Gomez-Rodriguez, M.; Song, L.; and Schölkopf, B. 2014. Estimating Diffusion Network Structures: Recovery Conditions, Sample Complexity & Soft-thresholding Algorithm. In *Proceedings of the 31st International Conference on Machine Learning (ICML 2014)*, 793–801.

Du, N.; Song, L.; Smola, A.; and Yuan, M. 2012. Learning Networks of Heterogeneous Influence. In *Advances in Neural Information Processing Systems 25 (NIPS 2012)*, 2780–2788.

Gan, T.; Han, K.; Huang, H.; Ying, S.; Gao, Y.; and Li, Z. 2021. Diffusion Network Inference from Partial Observations. In *Proceedings of the 35th AAAI Conference on Artificial Intelligence (AAAI 2021)*, 7493–7500.

Gomez-Rodriguez, M.; Balduzzi, D.; and Schölkopf, B. 2011. Uncovering the Temporal Dynamics of Diffusion Networks. In *Proceedings of the 28th International Conference on Machine Learning (ICML 2011)*, 561–568.

Gomez-Rodriguez, M.; Leskovec, J.; and Krause, A. 2010. Inferring Networks of Diffusion and Influence. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2010)*, 1019–1028.

Gomez-Rodriguez, M.; Leskovec, J.; and Schölkopf, B. 2013a. Modeling Information Propagation with Survival Theory. In *Proceedings of the 30th International Conference on Machine Learning (ICML 2013)*, 666–674.

Gomez-Rodriguez, M.; Leskovec, J.; and Schölkopf, B. 2013b. Structure and Dynamics of Information Pathways in Online Media. In *Proceedings of the 6th ACM International Conference on Web Search and Data Mining (WSDM 2013)*, 23–32.

Gomez-Rodriguez, M.; and Schölkopf, B. 2012. Submodular Inference of Diffusion Networks from Multiple Trees. In *Proceedings of the 29th International Conference on Machine Learning (ICML 2012)*, 489–496.

Han, K.; Tian, Y.; Zhang, Y.; Han, L.; Huang, H.; and Gao, Y. 2020. Statistical Estimation of Diffusion Network Topologies. In *Proceedings of the 36th IEEE International Conference on Data Engineering (ICDE 2020)*, 625–636.

He, X.; Rekatsinas, T.; Foulds, J.; Getoor, L.; and Liu, Y. 2015. HawkesTopic: A Joint Model for Network Inference and Topic Modeling from Text-Based Cascades. In *Proceedings of the 32nd International Conference on Machine Learning (ICML 2015)*, 871–880.

Huang, H.; Han, K.; Xu, B.; and Gan, T. 2022. Reconstructing diffusion networks from incomplete data. In *Proceedings of the 31st International Joint Conference on Artificial Intelligence (IJCAI 2022)*, 3085–3091.

Huang, H.; Han, K.; Xu, B.; and Gan, T. 2023a. Multi-aspect difusion network inference. In *Proceedings of the 2023 ACM Web Conference (WWW 2023)*, 82–90.

Huang, H.; Yan, Q.; Chen, L.; Gao, Y.; and Jensen, C. S. 2021. Statistical Inference of Diffusion Networks. *IEEE Transactions on Knowledge and Data Engineering*, 33(2): 742–753.

Huang, H.; Yan, Q.; Gan, T.; Niu, D.; Lu, W.; and Gao, Y. 2019. Learning Diffusions without Timestamps. In *Proceedings of the 33rd AAAI Conference on Artificial Intelligence (AAAI 2019)*, 582–589.

Huang, H.; Yan, Q.; Han, K.; Gan, T.; Jiang, J.; Xu, Q.; and Yang, C. 2023b. Learning Diffusions under Uncertainty. *arXiv preprint*, arXiv:2312.07942.

Lancichinetti, A.; Fortunato, S.; and Radicchi, F. 2008. Benchmark Graphs for Testing Community Detection Algorithms. *Physical Review E*, 78(4).

Leskovec, J.; Adamic, L. A.; and Huberman, B. A. 2007. The Dynamics of Viral Marketing. *ACM Transactions on the Web*, 1(1): 5.

Myers, S.; and Leskovec, J. 2010. On the Convexity of Latent Social Network Inference. In *Advances in Neural Information Processing Systems 23 (NIPS 2010)*, 1741–1749.

Sefer, E.; and Kingsford, C. 2015. Convex Risk Minimization to Infer Networks from Probabilistic Diffusion Data at Multiple Scales. In *Proceedings of the 31st IEEE International Conference on Data Engineering (ICDE 2015)*, 663–674.

Wallinga, J.; and Teunis, P. 2004. Different Epidemic Curves for Severe Acute Respiratory Syndrome Reveal Similar Impacts of Control Measures. *American Journal of Epidemiology*, 160(6): 509–516.

Wang, S.; Hu, X.; Yu, P.; and Li, Z. 2014. MMRate: Inferring Multi-aspect Diffusion Networks with Multi-pattern Cascades. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2014)*, 1246–1255.

Yan, Q.; Huang, H.; Gao, Y.; Lu, W.; and He, Q. 2017. Group-Level Influence Maximization with Budget Constraint. In *Proceedings of the 22nd International Conference on Database Systems for Advanced Applications (DASFAA 2017)*, 625–641.

Yang, Z.; Xu, Q.; Gao, S.; Yang, C.; Wang, G.; Zhao, Y.; Kong, F.; Liu, H.; Wang, W.; and Xiao, J. 2023. OceanBase Paetica: A Hybrid Shared-nothing/Shared-everything Database for Supporting Single Machine and Distributed Cluster. *PVLDB*, 16(12): 3728–3740.