Reconstructing Diffusion Networks from Incomplete Data

Hao Huang, **Keqi Han**, **Beicheng Xu** and **Ting Gan**
School of Computer Science, Wuhan University, China
{haohuang, hankeqi, beichengxu, ganting}@whu.edu.cn

Abstract

To reconstruct the topology of a diffusion network, existing approaches customarily demand not only eventual infection statuses of nodes, but also the exact times when infections occur. In real-world settings, such as the spread of epidemics, tracing the exact infection times is often infeasible; even obtaining the eventual infection statuses of all nodes is a challenging task. In this work, we study topology reconstruction of a diffusion network with incomplete observations of the node infection statuses. To this end, we iteratively infer the network topology based on observed infection statuses and estimated values for unobserved infection statuses by investigating the correlation of node infections, and learn the most probable probabilities of the infection propagations among nodes w.r.t. current inferred topology, as well as the corresponding probability distribution of each unobserved infection status, which in turn helps update the estimate of unobserved data. Extensive experimental results on both synthetic and real-world networks verify the effectiveness and efficiency of our approach.

1 Introduction

The topology of a diffusion network provides an explicit view for the influence relationships between network nodes, and is crucial for revealing the intrinsic mechanisms of diffusion dynamics. In practice, diffusion network topologies are often not naturally accessible, and need to be reconstructed based on diffusion results observed from history [Gan *et al.*, 2021].

To reconstruct the topology of a diffusion network, most existing approaches rely on the assumption that nodes infected sequentially within a short time interval are more likely to possess influence relationships. Therefore, these approaches demand not only the eventual infection statuses of nodes, but also the exact times (called cascades) when infections occur, and aim to find an optimal diffusion network topology that maximizes the likelihood of the cascades [Gomez-Rodriguez *et al.*, 2010; Myers and Leskovec, 2010; Gomez-Rodriguez *et al.*, 2011; Gomez-Rodriguez and Schölkopf, 2012; Du *et al.*, 2012; Gomez-Rodriguez *et al.*, 2013; Daneshmand *et al.*, 2014; Wang *et al.*, 2014; Pouget-Abadie and Horel, 2015; Narasimhan *et al.*, 2015; Rong *et al.*, 2016; Kalimeris *et al.*, 2018]. Nevertheless, in many real-world settings, such as the spread of epidemics and viral marketing campaigns, monitoring the entire diffusion process to obtain precise cascades is expensive and time consuming. A few existing approaches try to reconstruct diffusion networks with only the eventual infection statuses of nodes [Amin *et al.*, 2014; Huang *et al.*, 2019; Han *et al.*, 2020; Huang *et al.*, 2021], which are more easily accessible than cascades in most cases.

Whether the existing approaches use cascades or the eventual infection statuses of nodes, they usually assume that the observed data are complete. Unfortunately, this assumption often fails to hold in practice [Gan *et al.*, 2021]. For example, it is difficult to collect infection situations of all individuals during the spread of epidemics because of population mobility and limited medical resources. It is also hard to guarantee that every respondent provides feedback in viral marketing campaigns. To deal with incomplete observed data, a simple and straightforward strategy is replacing each missing data with its corresponding expected value in the observed part [Sefer and Kingsford, 2015], under the assumption that the missing and observed data have the same probability distribution. However, this strategy has no accuracy guarantee, and may result in a severe bias for the estimate of missing data. To minimize bias, a few studies utilize constrained optimization to handle incomplete observed data, but most of them aim to learn either the strengths of influence relationships [Lokhov, 2016; Wilinski and Lokhov, 2020] or influence functions [He *et al.*, 2016] when the topology of diffusion network is given. Only one work aims to reconstruct diffusion network topologies with incomplete observed data [Gan *et al.*, 2021]. Nonetheless, this work tries to maximize a relaxed objective function rather than the original objective function, and thus can not guarantee to obtain the optimal solution or a local optimal solution of the original problem.

Aiming at an accurate solution for diffusion network reconstruction with incomplete observations of node infection statuses, we propose LIDO (**L**earning from **I**ncomplete **D**iffusion **O**bservations) algorithm. It consists of three iterative steps, i.e., (1) inferring the topology of target diffusion network based on the observed node infection statuses and estimated values for unobserved node infection statuses, (2) jointly learning the optimal infection propagation probabilities between adjacent nodes in current inferred topology and

the corresponding probability distribution of each unobserved node infection status, that are most likely to generate the observed data, and (3) updating the estimate of unobserved infection statuses by sampling values from the latest learned probability distributions. LIDO repeats these three steps until the inferred topology converges to a stable solution.

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

2 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). 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 p_{ji}, which is known as infection propagation probability. We assume that the diffusion processes on G follow the Independent Cascade (IC) model [Kempe *et al.*, 2003], in which each infected parent node tries to infect each of its uninfected children with corresponding infection propagation probability independently. This assumption is commonly used in existing approaches, and we would like to point out that our proposed algorithm can be also extended to other propagation models such as the linear threshold model.

In the problem of diffusion network reconstruction, the node set V is given, while the topology (i.e., the directed edge set E) of target diffusion network is unknown, so are the infection propagation probabilities w.r.t. the topology. To reconstruct the 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 contain only the final infection statuses of nodes in each diffusion process, i.e., $S = \{s_i^\ell \mid i \in \{1, ..., n\}, \ell \in \{1, ..., \beta\}\}$, where $s_i^\ell \in \{0, 1\}$ is the final infection status of node v_i in the ℓ-th diffusion process, and β is the number of historical diffusion processes. Furthermore, we focus on a complex yet more realistic situation, i.e., partial data in S are unobserved. Let S^{obs} and S^{mis} denote the observed part and unobserved part in S, respectively, then our problem statement can be formulated as follows.

Given: partial observations S^{obs} of node infection statuses observed on a diffusion network G at the end of β historical diffusion processes.

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

3 The LIDO Algorithm

To reconstruct the topology of G with partial observations S^{obs}, LIDO performs the following three steps iteratively, namely (1) inferring the topology with S^{obs} and estimated values $\hat{S}^{mis}$ for S^{mis}, (2) learning the optimal infection propagation probabilities and corresponding probability distributions of S^{mis} w.r.t. the current inferred topology, and (3) updating $\hat{S}^{mis}$ by sampling from the learned probability distribution. The details of these steps as well as a complexity analysis for LIDO are provided in what follows.

Topology inference. In general, the goal of diffusion network reconstruction with node infection statuses S is to find an optimal directed edge set E^* that satisfies the following requirement.

$$E^* = \arg\max_E P(E \mid S). \tag{1}$$

Nonetheless, in our problem setting, the node infection statuses is incomplete, containing only partial observations S^{obs}. To estimate $P(E \mid S)$ with less bias, we can repeatedly sample S^{mis} to obtain a set of samples of unobserved data $\{\hat{S}_1^{mis}, ..., \hat{S}_m^{mis}\}$, where m is the number of sampling rounds. Then, we can obtain a set of samples of complete data $\{\hat{S}_1, ..., \hat{S}_m\}$, in which the r-th sample $\hat{S}_r$ consists of S^{obs} and $\hat{S}_r^{mis}$, i.e., $\hat{S}_r = (S^{obs}, \hat{S}_r^{mis})$. If the sampling is sufficient and follows the underlying probability distributions of unobserved data, then

$$P(E \mid S) \simeq P(E \mid \hat{S}_1, ..., \hat{S}_m). \tag{2}$$

Let $\{\hat{S}_1^{(t)}, ..., \hat{S}_m^{(t)}\}$ denote the sample set of complete data sampled by LIDO after the t-th iteration, then in the $(t+1)$-th iteration, the task of topology inference is to find a directed edge set $E^{(t+1)}$ that satisfies the following requirement.

$$E^{(t+1)} = \arg\max_E P(E \mid \hat{S}_1^{(t)}, ..., \hat{S}_m^{(t)}). \tag{3}$$

Note that the problem in Eq. (3) is equivalent to reconstructing diffusion network topologies from only the complete data of final node infection statuses. Existing approaches to this problem, such as TENDS [Han *et al.*, 2020], can be utilized to infer $E^{(t+1)}$. Therefore, how to complete the data through sampling is essential for topology inference.

Learning probability distribution for sampling. After inferring $E^{(t+1)}$ with $\{\hat{S}_1^{(t)}, ..., \hat{S}_m^{(t)}\}$, the subsequent task is to find the optimal infection propagation probabilities $\{p_{ji} \mid (v_j, v_i) \in E^{(t+1)}\}$ and the corresponding probability distributions of unobserved data S^{mis} that are most likely to generate the observed data S^{obs}. Then, by sampling S^{mis} from the learned probability distributions, we can obtain an updated sample set of complete data $\{\hat{S}_1^{(t+1)}, ..., \hat{S}_m^{(t+1)}\}$ for the next iteration of topology inference.

According to IC model, when the infection status s_i^ℓ of node v_i in the ℓ-th diffusion process is observed and $s_i^\ell = 1$, the likelihood that the infection of v_i is caused by any of its parent nodes is $1 - \prod_{v_j \in F_i}(1 - p_{ji} s_j^\ell)$, where F_i refers to the set of parent nodes of v_i; when s_i^ℓ is observed and $s_i^\ell = 0$, the likelihood that none of the parent nodes of v_i has successfully infected v_i can be estimated as $\prod_{v_j \in F_i}(1 - p_{ji} s_j^\ell)$.

When s_i^ℓ is unobserved, i.e., $s_i^\ell \in S^{mis}$, we regard s_i^ℓ with the probability that $X_i^\ell = 1$, i.e., $s_i^\ell = P(X_i^\ell = 1 \mid S^{obs})$, where $X_i^\ell \in \{0, 1\}$ is the infection status variable of v_i in the ℓ-th diffusion process. In other words, we substitute the unknown s_i^ℓ with its expectation $P(X_i^\ell = 1 \mid S^{obs})$.

Then, the likelihood of the observed data S^{obs} can be

written as $\prod_{\ell=1}^{\beta}\prod_{s_i^\ell\in S^{obs}} P(X_i^\ell=s_i^\ell)$ with constraints:

$$\begin{aligned} P(X_i^\ell=s_i^\ell) &= 1-\prod_{v_j\in F_i^{(t+1)}}(1-p_{ji}s_j^\ell),\ s_i^\ell=1,\\ P(X_i^\ell=s_i^\ell) &= \prod_{v_j\in F_i^{(t+1)}}(1-p_{ji}s_j^\ell), \qquad s_i^\ell=0,\\ s_i^\ell &= 1-\prod_{v_j\in F_i^{(t+1)}}(1-p_{ji}s_j^\ell), \qquad s_i^\ell\in S^{mis}, \end{aligned} \tag{4}$$

where $F_i^{(t+1)}$ refers to the set of parent nodes of node v_i in current inferred topology $E^{(t+1)}$.

Then, we can learn the optimal $\{p_{ji}\mid (v_j,v_i)\in E^{(t+1)}\}$ and corresponding $\{s_i^\ell\in S^{mis}\}$ by solving the following optimization problem.

$$\begin{aligned} \max\ \mathcal{G} = &\sum_{\ell=1}^{\beta}\sum_{s_i^\ell=1}\ln\Big(1-\prod_{v_j\in F_i^{(t+1)}}\big(1-p_{ji}s_j^\ell\big)\Big)+\\ &\sum_{\ell=1}^{\beta}\sum_{s_i^\ell=0}\ln\Big(\prod_{v_j\in F_i^{(t+1)}}\big(1-p_{ji}s_j^\ell\big)\Big)\\ \text{s.t. }\ & c_i^\ell=0,\ s_i^\ell\in S^{mis}, \end{aligned} \tag{5}$$

where

$$c_i^\ell = 1-s_i^\ell-\textstyle\prod_{v_j\in F_i^{(t+1)}}(1-p_{ji}s_j^\ell). \tag{6}$$

Lagrange multiplier method can be used to solve the above problem, and the corresponding Lagrangian is as follows.

$$\mathcal{L}=\mathcal{G}+\textstyle\sum_{\ell=1}^{\beta}\sum_{s_i^\ell\in S^{mis}}\lambda_i^\ell c_i^\ell, \tag{7}$$

where λ_i^ℓ is the Lagrange multiplier corresponding to constraint $c_i^\ell=0$ ($s_i^\ell\in S^{mis}$).

Given this Lagrangian $\mathcal{L}$, its derivative w.r.t. p_{ji} is

$$\begin{aligned} \frac{\partial\mathcal{L}}{\partial p_{ji}} = &\sum_{\ell,\ s_i^\ell=1}\frac{\prod_{v_k\in F_i^{(t+1)}\setminus v_j}(1-p_{ki}s_k^\ell)s_j^\ell}{1-\prod_{v_k\in F_i^{(t+1)}}(1-p_{ki}s_k^\ell)}-\\ &\sum_{\ell,\ s_i^\ell=0}\frac{s_j^\ell}{1-p_{ji}s_i^\ell}+\\ &\sum_{\ell,\ s_i^\ell\in S^{mis}}\lambda_i^\ell\prod_{v_k\in F_i^{(t+1)}\setminus v_j}(1-p_{ki}s_k^\ell)s_j^\ell, \end{aligned} \tag{8}$$

and the derivative of $\mathcal{L}$ w.r.t. $s_j^\ell\in S^{mis}$ is

$$\begin{aligned} \frac{\partial\mathcal{L}}{\partial s_j^\ell} = &\sum_{s_i^\ell=1,\ v_j\in F_i^{(t+1)}}\frac{\prod_{v_k\in F_i^{(t+1)}\setminus v_j}(1-p_{ki}s_k^\ell)p_{ji}}{1-\prod_{v_k\in F_i^{(t+1)}}(1-p_{ki}s_k^\ell)}-\\ &\sum_{s_i^\ell=0,\ v_j\in F_i^{(t+1)}}\frac{p_{ji}}{1-p_{ji}s_i^\ell}\ -\ \lambda_j^\ell+\\ &\sum_{s_i^\ell\in S^{mis},\ v_j\in F_i^{(t+1)}}\lambda_i^\ell\prod_{v_k\in F_i^{(t+1)}\setminus v_j}(1-p_{ki}s_k^\ell)p_{ji}. \end{aligned} \tag{9}$$

Then, solving the optimization problem in Eq. (5) can be transformed into solving the following equations.

$$\begin{aligned} &c_i^\ell=0, && s_i^\ell\in S^{mis};\\ &\frac{\partial\mathcal{L}}{\partial s_j^\ell}=0, && s_j^\ell\in S^{mis};\\ &\frac{\partial\mathcal{L}}{\partial p_{ji}}=0, && v_j\in F_i^{(t+1)}. \end{aligned} \tag{10}$$

Specifically, starting with randomly initialized $p_{ji}\in(0,1)$, Eq. (10) can be solved by the following three iterative steps:

Step 1. Substituting current p_{ji} into Eq. (6), we have a system of equations w.r.t. $s_i^\ell\in S^{mis}$:

$$1-s_i^\ell-\textstyle\prod_{v_j\in F_i^{(t+1)}}(1-p_{ji}s_j^\ell)=0,\ \ s_i^\ell\in S^{mis}. \tag{11}$$

As each historical diffusion process is independent to each other, for each $\ell\in\{1,...,\beta\}$, we have a system of equations:

$$1-s_i^\ell-\prod_{v_j\in F_i^{(t+1)}}(1-p_{ji}s_j^\ell)=0,\ i\in\{i\mid s_i^\ell\in S^{mis}\}. \tag{12}$$

To solve Eq. (12), if the condition that for each $v_j\in F_i^{(t+1)}$, s_j^ℓ is observed or has been calculated already, holds for node v_i in the ℓ-th diffusion process, then the corresponding $s_i^\ell\in S^{mis}$ can be calculated as

$$s_i^\ell = 1-\textstyle\prod_{v_j\in F_i^{(t+1)}}(1-p_{ji}s_j^\ell). \tag{13}$$

We repeatedly find which nodes in the ℓ-th diffusion process satisfy the above condition, and calculate corresponding s_i^ℓ by Eq. (13), until no node can be found. If there are still some unobserved s_i^ℓ left that cannot be directly calculated by Eq. (13), it indicates that there exist cyclic dependencies among the calculation of these unobserved s_i^ℓ. Without loss of generality, let $s_1^\ell,...,s_k^\ell$ be this kind of unobserved infection statuses, then we can calculate their values jointly by solving the following equations.

$$1-s_i^\ell-\textstyle\prod_{v_j\in F_i^{(t+1)}}(1-p_{ji}s_j^\ell)=0,\ i\in\{1,...,k\}. \tag{14}$$

The above Eq. (14) is a polynomial system of equations w.r.t. $s_1^\ell,...,s_k^\ell$, of which the numerical solution can be efficiently obtained by existing tools, such as the `fsolve` and `root` functions in SciPy package for Python.

Step 2. Substituting current p_{ji} and s_j^ℓ into equations $\frac{\partial\mathcal{L}}{\partial s_j^\ell}=0$ ($s_j^\ell\in S^{mis}$), we have a system of equations w.r.t. the Lagrange multipliers λ_i^ℓ. According to Eq. (9), this system of equations is linear and can be solved directly.

Step 3. Substituting current p_{ji}, s_j^ℓ and λ_i^ℓ into Eq. (8), we have the latest gradient $\frac{\partial\mathcal{L}}{\partial p_{ji}}$ of the Lagrangian $\mathcal{L}$ w.r.t. p_{ji}, then we update p_{ji} with the latest gradient as follows.

$$p_{ji}=p_{ji}+\theta^{(\tau)}\frac{\partial\mathcal{L}}{\partial p_{ji}}, \tag{15}$$

where $\theta^{(\tau)}>0$ is the updating step size used in the τ-th iteration of Steps 1–3.

Note that using variable step sizes helps strike a trade-off between convergence guarantee and speed. Large step sizes are first adopted to quickly reach the neighborhood of the optimum; then, small steps are adopted to avoid overshooting. If the optimum is static, the gradient descent approach is guaranteed to converge to a stationary point when the step sizes satisfy the relationship below [Kushner and Yin, 2003].

$$\lim_{\tau \to +\infty} \theta^{(\tau)} = 0 \quad \text{and} \quad \sum_{\tau=1}^{\infty} \theta^{(\tau)} = \infty \tag{16}$$

Inspired by this, we set $\theta^{(\tau)} = \frac{\theta_0}{\sqrt{\tau}}$, where $\theta_0 = 0.001$ works as the initial step size.

After learning the optimal $\{p_{ji} \mid (v_j, v_i) \in E^{(t+1)}\}$ and corresponding $\{s_i^\ell \in S^{mis}\}$, LIDO executes sampling based on the learned $\{s_i^\ell \in S^{mis}\}$, which describes the probability distribution of unobserved data.

Complexity analysis. In topology inference, performing TENDS algorithm on complete data sampled by LIDO takes about $O(m\beta n^2)$ time, where n is the number of nodes in target diffusion network, β is the number of the historical diffusion processes, and m is the number of sampling rounds for unobserved data.

To learn probability distribution for sampling, the most computationally expensive process is solving the equations Eq. (10) by our proposed three steps. The time complexities of the three steps are all bounded by $O(\beta n^3)$, and thus learning probability distribution takes about $O(\tau\beta n^3)$ time, where τ is the iteration number of the three steps.

In brief, the time complexity of LIDO is $O(tm\beta n^2 + t\tau\beta n^3)$, where t refers to the number of iterations of LIDO.

4 Experimental Evaluation

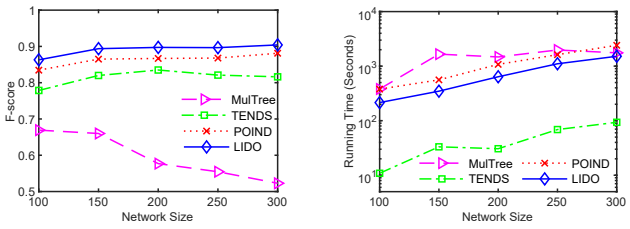
In this section, we first introduce experimental setup, and then evaluate LIDO on synthetic and real-world networks with simulated infection data by investigating the effects of network size, network's average degree, unobserved data ratio, the amount of diffusion processes, and the iterations of LIDO on the accuracy and running time performance of LIDO. Furthermore, we also carry out a case study on a real-world network with real infection data. All algorithms in the experiments are implemented in Python, running on a desktop PC with Intel Core i3-6100 CPU at 3.70GHz and 8GB RAM.

Experimental setup. (1) Network: We adopt LFR benchmark graphs [Lancichinetti *et al.*, 2008] as the synthetic networks, and generate two series of LFR benchmark graphs with properties summarized in Table 1. In addition, we adopt two commonly used real-world networks: NetSci [Newman, 2006], a co-authorship network containing 379 scientists and 1602 co-authorships, and DUNF [Wang *et al.*, 2014], a microblogging network containing 750 users and 2974 following relationships.

(2) Infection data: The infection status results S or cascades can be obtained 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 an infection propagation probability that follows a Gaussian distribution with mean of 0.3 and standard deviation of 0.05, to make about 95% of infection propagation probabilities within a range from 0.2 to 0.4 [Gan *et al.*, 2021]. We randomly remove partial data from S as the unobserved data S^{mis} (γ denotes the unobserved data ratio), and use the remaining observation data S^{obs} for diffusion network reconstruction.

Graphs	Number of Nodes	Average Degree
LFR1-5	100,150,200,250,300	4
LFR6-10	200	2,3,4,5,6

Table 1: Properties of LFR benchmark graphs



(a) F-score (b) Running time

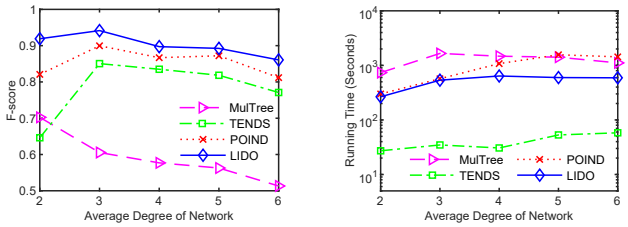
Figure 1: Effect of diffusion network size

(3) Performance criterion: To evaluate the accuracy performance of LIDO algorithm, we report the F-score (the harmonic mean of precision and recall) of its inferred directed edges, which is computed as $F\text{-}score = \frac{2 \cdot Precision \cdot Recall}{Precision + Recall}$, where $Precision = \frac{N_{TP}}{N_{TP}+N_{FP}}$, $Recall = \frac{N_{TP}}{N_{TP}+N_{FN}}$, N_{TP} is the number of the true edges that are correctly inferred by the algorithm; N_{FP} is the number of the wrong inferred edges that are not in the real network; and N_{FN} is the number of the true edges that are not correctly inferred by the algorithm.

(4) Benchmark algorithms: We compare LIDO with a classical cascade-based approach MulTree [Gomez-Rodriguez and Schölkopf, 2012], a high-performance infection status-based approach TENDS [Han *et al.*, 2020], and POIND [Gan *et al.*, 2021] which is the only existing approach to diffusion network reconstruction with incomplete data. In LIDO, the number m of sampling round is set to 6, the maximum number t of iterations is set to 5, and the stop condition for updating p_{ji} in each iteration is that the variation of each p_{ji} is less than 0.01. As TENDS requires complete data, we complete the S^{obs} with the following ad-hoc method for them: we estimate the average infection probability of each node in S^{obs}, and sample for the unobserved data 6 times based on the average infection probabilities.

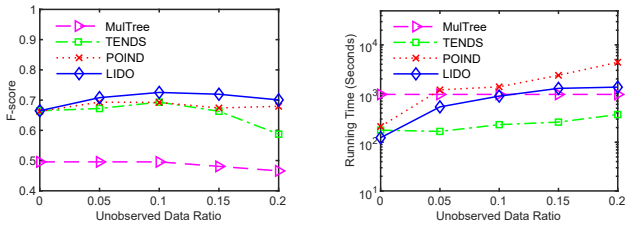
Effect of diffusion network size. To study the effect of diffusion network size, we adopt five synthetic networks, LFR1–5, whose sizes vary from 100 to 300. We simulate 200 times of diffusion processes on each network ($\beta = 200$), and randomly remove 15% of infection status observations as unobserved data ($\gamma = 0.15$).

Fig. 1 illustrates the F-score and running time of each algorithm, from which we can have the following observations:

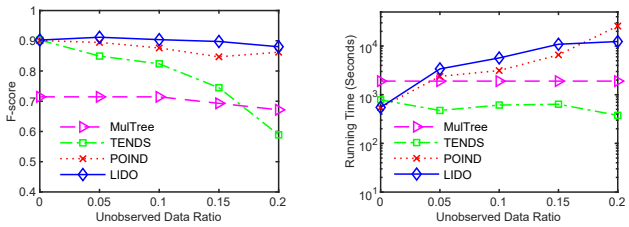


(a) F-score (b) Running time

Figure 2: Effect of average degree of diffusion network



(a) F-score on NetSci (b) Running time on NetSci



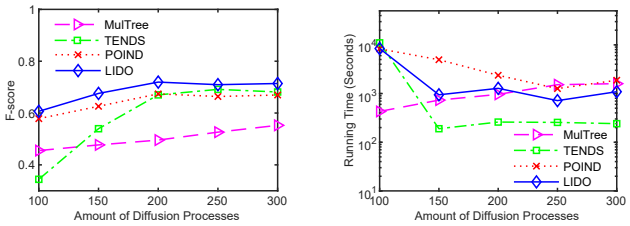
(c) F-score on DUNF (d) Running time on DUNF

Figure 3: Effect of unobserved data ratio

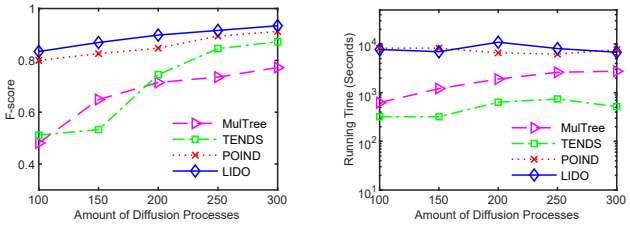
(1) LIDO outperforms the three benchmark algorithms in terms of accuracy. (2) A larger diffusion network size tends to degrade the accuracy of MulTree, while the accuracy of LIDO, TENDS, and POIND are reasonably insensitive to the size of diffusion network. TENDS's insensitivity to diffusion network size has also been verified in existing study [Han *et al.*, 2020]. This is one of the reasons for why LIDO adopts TENDS for topology inference with completed data. (3) The running time of each tested algorithm tends to increase with the growth of diffusion network size, and TENDS is faster than LIDO as there is no iteration in TENDS.

Effect of average degree of diffusion network. To study the effect of the average degree of diffusion network, we test the algorithms on five synthetic networks, LFR6–10, whose average degree of each node varies from 2 to 6. We simulate 200 times of diffusion processes on each network ($\beta = 200$), and randomly remove 15% of infection status observations as unobserved data ($\gamma = 0.15$).

Fig. 2 illustrates the F-score and running time of each algorithm, from which we can have the following observations: (1) Among the tested algorithms, LIDO has the best accuracy. (2) With the increase of the average degree, the accuracy of LIDO decreases slightly. This is because a greater average degree often brings more complicated influence relationships between nodes, and thus adds complexity to diffusion network reconstruction. (3) The running time of each tested algorithm increases with the growth of average degree.



(a) F-score on NetSci (b) Running time on NetSci



(c) F-score on DUNF (d) Running time on DUNF

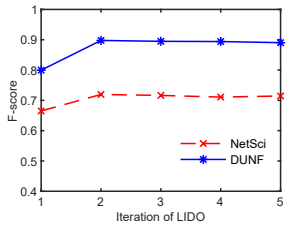
Figure 4: Effect of amount of diffusion processes

Effect of unobserved data ratio. To study the effect of the ratio of unobserved data, we test the algorithms on two real-world networks, NetSci and DUNF, with $\beta = 200$, varying unobserved data ratio γ from 0 to 0.2.

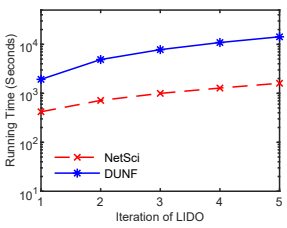
Fig. 3 illustrates the F-score and running time of each algorithm, from which we can have the following observations: (1) Compared with MulTree and POIND, LIDO often achieves higher accuracy. (2) LIDO and TENDS have the same accuracy in the setting of $\gamma = 0$; when there is more unobserved data, LIDO shows a greater advantage on accuracy than TENDS. (3) The increase of unobserved data ratio has a rather mild effect on the running time of MulTree and TENDS, but results in significantly longer running time for LIDO and POIND. This is because LIDO and POIND require longer running time to infer more unobserved data, while MulTree and TENDS use completed data directly.

Effect of amount of diffusion processes. To study the effect of the amount β of diffusion processes, we test the algorithms on NetSci and DUNF with different β, which varies from 100 to 300. In the diffusion result obtained with each β, we randomly remove 15% of infection status observations as unobserved data ($\gamma = 0.15$).

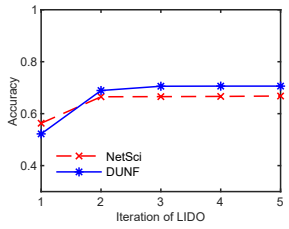
Fig. 4 illustrates the F-score and running time of each algorithm, from which we can have the following observations: (1) With more diffusion processes, the tested algorithms obtain more accurate results, and LIDO leads its competitors in accuracy. (2) To analyze the cascades observed from more diffusion processes, MulTree requires more running time, while a greater amount of diffusion processes does not always increase the running time of LIDO, TENDS and POIND.



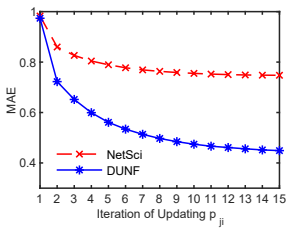
(a) F-score



(b) Running time



(c) Accuracy of estimated S^{mis}



(d) MAE of learned p_{ji}

Figure 5: Effect of iterations of LIDO

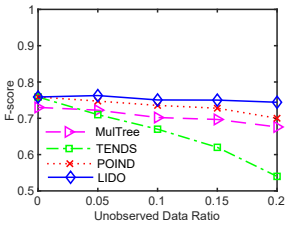
Effect of iterations of LIDO. To study the effect of the iterations of LIDO, we test LIDO on NetSci and DUNF with $\beta = 200$ and $\gamma = 0.15$, and report the accuracy and accumulated running time of LIDO at the end of each iteration.

Fig. 5(a) & (b) illustrate the testing result, from which we can observe that (1) LIDO shows a fast convergence property; (2) more iterations result in longer running time for LIDO.

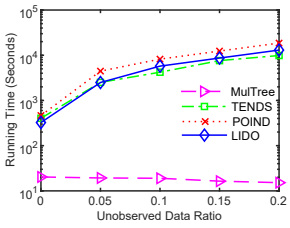
The main reason behind the fast convergence property is that LIDO can accurately estimate the values of unobserved data. To verify this point, Fig. 5(c) illustrates the *accuracy* (i.e., the rate of correct estimated values) of LIDO's estimated values for unobserved data S^{mis} in each iteration, from which we can observe that the accuracy of estimated values increases and converges to a stable level quickly. Another important reason is due to the effectiveness of LIDO on revealing the underlying infection propagation probabilities, which helps LIDO to estimate the probability distribution and the values of of unobserved data in turn. To demonstrate this effectiveness, Fig. 5(d) illustrates the MAE (Mean Absolute Error) of infection propagation probabilities $\{p_{ji} \mid (v_j, v_i) \in E^{(t+1)}\}$ learned in the first iteration of LIDO (i.e., $t = 1$). A lower MAE indicates a higher accuracy. From Fig. 5(d), we can observe that the accuracy of learned p_{ji} is significantly improved by iteratively performing our proposed learning approach. Similar results can be observed when $t > 1$.

Case study on real infection data. To verify the effectiveness of LIDO in real scenarios, we adopt a commonly used real-world data set MemeTracker (http://memetracker.org), which contains more than 96 million news articles and blogs from 1 million sources over a period of nine months from August 1 2008 till April 31 2009.

As the ground truth of network topology is unknown on MemeTracker data set, we generate a MemeTracker network having 1000 media sites and blogs with 4000 directed hyperlinks as follows. We first extract the top 1000 media



(a) F-score



(b) Running time

Figure 6: Case study on MemeTracker data set

sites and blogs with the largest number of documents as 1000 network nodes, and then add directed edge (v_i, v_j) if a post on network node v_i linked to a post on a network node v_j. Hyperlinks between blog posts can be used to trace diffusion processes [Leskovec *et al.*, 2007]. When a blog publishes a piece of information and uses hyperlinks to refer to posts published by other blogs, we consider this kind of activities as events of information propagation. By tracing these information propagation events between the nodes of our generated MemeTracker network, we identify 13000 infection results S and cascades of corresponding diffusion processes. Furthermore, to create incomplete data, we randomly remove partial data from S as the unobserved data S^{mis} (the unobserved data ratio γ varies from 0 to 0.2), and use the remaining observation data S^{obs} for diffusion network reconstruction.

Fig. 6 illustrates the F-score and running time of each tested algorithm, from which we can have the following observations: (1) A greater unobserved data ratio often degrades the accuracy of each tested algorithm, while LIDO leads its competitors in accuracy. (2) The increase of unobserved data ratio results in longer running time for LIDO, TENDS and POIND, while having a rather mild effect on the running time of MulTree. (3) MulTree tends to be relatively more efficient than the other tested algorithm. This is because most of the diffusion processes identified from the MemeTracker data set start from a single media or blog site, with tree-shape propagation paths, which are more applicable to MulTree.

5 Conclusion

In this paper, we have investigated the problem of how to reconstruct the topology of a diffusion network with incomplete observations of node infection statuses, and presented an effective approach called LIDO to solve this problem in an iterative way. In each iteration, LIDO infers the topology of target diffusion network based on observed data and estimated values of unobserved data, and learns the optimal infection propagation probabilities and corresponding probability distributions of unobserved data w.r.t. the inferred topology. The learned probability distributions are used by LIDO to update the estimate of unobserved data for the next iteration of topology inference. Extensive experimental results on both synthetic and real-world networks show that LIDO can properly handle incomplete observation data and reconstruct diffusion network topologies accurately.

Acknowledgments

This work was supported in part by NSFC Grants 61976163 and 61902284. Ting Gan is the corresponding author.

References

[Amin *et al.*, 2014] Kareem Amin, Hoda Heidari, and Michael Kearns. Learning from contagion(without timestamps). In *ICML 2014*, pages 1845–1853, 2014.

[Daneshmand *et al.*, 2014] Hadi Daneshmand, Manuel Gomez-Rodriguez, Le Song, and Bernhard Schölkopf. Estimating diffusion network structures: Recovery conditions, sample complexity & soft-thresholding algorithm. In *ICML 2014*, pages 793–801, 2014.

[Du *et al.*, 2012] Nan Du, Le Song, Alex Smola, and Ming Yuan. Learning networks of heterogeneous influence. In *NIPS 2012*, pages 2780–2788, 2012.

[Gan *et al.*, 2021] Ting Gan, Keqi Han, Hao Huang, Shi Ying, Yunjun Gao, and Zongpeng Li. Diffusion network inference from partial observations. In *AAAI 2021*, pages 7493–7500, 2021.

[Gomez-Rodriguez and Schölkopf, 2012] Manuel Gomez-Rodriguez and Bernhard Schölkopf. Submodular inference of diffusion networks from multiple trees. In *ICML 2012*, pages 489–496, 2012.

[Gomez-Rodriguez *et al.*, 2010] Manuel Gomez-Rodriguez, Jure Leskovec, and Andreas Krause. Inferring networks of diffusion and influence. In *KDD 2010*, pages 1019–1028, 2010.

[Gomez-Rodriguez *et al.*, 2011] Manuel Gomez-Rodriguez, David Balduzzi, and Bernhard Schölkopf. Uncovering the temporal dynamics of diffusion networks. In *ICML 2011*, pages 561–568, 2011.

[Gomez-Rodriguez *et al.*, 2013] Manuel Gomez-Rodriguez, Jure Leskovec, and Bernhard Schölkopf. Modeling information propagation with survival theory. In *ICML 2013*, pages 666–674, 2013.

[Han *et al.*, 2020] Keqi Han, Yuan Tian, Yunjia Zhang, Ling Han, Hao Huang, and Yunjun Gao. Statistical estimation of diffusion network topologies. In *ICDE 2020*, pages 625–636, 2020.

[He *et al.*, 2016] Xinran He, Ke Xu, David Kempe, and Yan Liu. Learning influence functions from incomplete observations. In *NIPS 2016*, pages 2065–2073, 2016.

[Huang *et al.*, 2019] Hao Huang, Qian Yan, Ting Gan, Di Niu, Wei Lu, and Yunjun Gao. Learning diffusions without timestamps. In *AAAI 2019*, pages 582–589, 2019.

[Huang *et al.*, 2021] Hao Huang, Qian Yan, Lu Chen, Yunjun Gao, and Christian S. Jensen. Statistical inference of diffusion networks. *IEEE Transactions on Knowledge and Data Engineering*, 33(2):742–753, 2021.

[Kalimeris *et al.*, 2018] Dimitris Kalimeris, Yaron Singer, Karthik Subbian, and Udi Weinsberg. Learning diffusion using hyperparameters. In *ICML 2018*, pages 2420–2428, 2018.

[Kempe *et al.*, 2003] David Kempe, Jon M. Kleinberg, and Éva Tardos. Maximizing the spread of influence through a social network. In *KDD 2003*, pages 137–146, 2003.

[Kushner and Yin, 2003] Harold J. Kushner and G. George Yin. *Stochastic approximation and recursive algorithms and applications*, volume 35. Springer, 2003.

[Lancichinetti *et al.*, 2008] Andrea Lancichinetti, Santo Fortunato, and Filippo Radicchi. Benchmark graphs for testing community detection algorithms. *Physical Review E*, 78(4), 2008.

[Leskovec *et al.*, 2007] Jure Leskovec, Mary McGlohon, Christos Faloutsos, Natalie S. Glance, and Matthew Hurst. Patterns of cascading behavior in large blog graphs. In *SDM 2007*, pages 551–556, 2007.

[Lokhov, 2016] Andrey Y. Lokhov. Reconstructing parameters of spreading models from partial observations. In *NIPS 2016*, pages 3467–3475, 2016.

[Myers and Leskovec, 2010] Seth A. Myers and Jure Leskovec. On the convexity of latent social network inference. In *NIPS 2010*, pages 1741–1749, 2010.

[Narasimhan *et al.*, 2015] Harikrishna Narasimhan, David C. Parkes, and Yaron Singer. Learnability of influence in networks. In *NIPS 2015*, pages 3186–3194, 2015.

[Newman, 2006] M. E. J. Newman. Finding community structure in networks using the eigenvectors of matrices. *Physical Review E*, 74(3):036104, 2006.

[Pouget-Abadie and Horel, 2015] Jean Pouget-Abadie and Thibaut Horel. Inferring graphs from cascades: A sparse recovery framework. In *ICML 2015*, pages 977–986, 2015.

[Rong *et al.*, 2016] Yu Rong, Qiankun Zhu, and Hong Cheng. A model-free approach to infer the diffusion network from event cascade. In *CIKM 2016*, pages 1653–1662, 2016.

[Sefer and Kingsford, 2015] Emre Sefer and Carl Kingsford. Convex risk minimization to infer networks from probabilistic diffusion data at multiple scales. In *ICDE 2015*, pages 663–674, 2015.

[Wang *et al.*, 2014] Senzhang Wang, Xia Hu, Philip S. Yu, and Zhoujun Li. MMRate: Inferring multi-aspect diffusion networks with multi-pattern cascades. In *KDD 2014*, pages 1246–1255, 2014.

[Wilinski and Lokhov, 2020] Mateusz Wilinski and Andrey Y. Lokhov. Scalable learning of independent cascade dynamics from partial observations. *arXiv preprint*, arXiv:2007.06557, 2020.