Back to the Past: Source Identification in Diffusion Networks from Partially Observed Cascades

Mehrdad Farajtabar[*]   **Manuel Gomez-Rodriguez**[†]   **Nan Du**[*]
Mohammad Zamani[◇]   **Hongyuan Zha**[*]   **Le Song**[*]
[*]Georgia Institute of Technology   [†]MPI for Software Systems   [◇]Stony Brook University

Abstract

When a piece of malicious information becomes rampant in an information diffusion network, can we identify the *source* node that originally introduced the piece into the network and infer the time when it initiated this? Being able to do so is critical for curtailing the spread of malicious information, and reducing the potential losses incurred. This is a very challenging problem since typically only incomplete traces are observed and we need to unroll the incomplete traces into the past in order to pinpoint the source. In this paper, we tackle this problem by developing a two-stage framework, which first learns a continuous-time diffusion network model based on historical diffusion traces and then identifies the source of an incomplete diffusion trace by maximizing the likelihood of the trace under the learned model. Experiments on both large synthetic and real-world data show that our framework can effectively "go back to the past", and pinpoint the source node and its initiation time significantly more accurately than previous state-of-the-arts.

1 INTRODUCTION

On September 2014, a collection of hundreds of private pictures from various celebrities, mostly consisting of women and often containing nudity, were posted online, and later disseminated by users on websites and social networks such as Imgur[1], Reddit[2] and Tumblr[3] [Kedmey, 2014]. After quite some efforts in manual tracing of the diffusion paths, investigators found that the imageboard 4chan[4] was the culprit site where the photos were originally posted on August 31, even though the photos had been taken down from the site soon after their post. This leakage of private pictures has touched off a larger world-wide discussion and debate on the state of privacy and civil liberties on the Internet [Isaac, 2014].

Can we automatically pinpoint the identity of such malicious information sources, as well as the time when they first posted the malicious information, given historical incomplete diffusion traces? Solving this source identification problem is of outstanding interest in many scenarios [Lappas et al., 2010]. For example, finding people that originate rumors may reduce disinformation, identifying patient zeros in disease spreads may help to understand and control epidemics, or inferring where a trojan or computer worm is initially released may increase reliability of computer networks.

Related Work. The problem of finding the source of a diffusion trace, also called *cascade*, has not been studied until very recently [Lappas et al., 2010, Shah and Zaman, 2010, Aditya Prakash et al., 2012, Pinto et al., 2012]. However, most previous work assumes that a complete steady-state snapshot of the cascade is observed, in other words, we know *which* nodes got *infected* but *not when* they did so. Moreover, previous work uses discrete-time sequential propagation models such as the independent cascade model [Kempe et al., 2003] or the discrete version of the SIR model [Bailey, 1975], which are difficult to estimate accurately from real world data [Gomez-Rodriguez et al., 2011, Du et al., 2012, 2013b, Zhou et al., 2013a,b].

Only very recently, Pinto et al. [2012] consider a fairly general continuous-time model and assume that only a small fraction of sparsely-placed nodes are observed and, if infected, their infection time is observed. Unfortunately, their approach requires the distance between observed nodes to be large because they approximate the infection times by Gaussian distributions using the central limit theorem. Since this is easily violated in real social and information
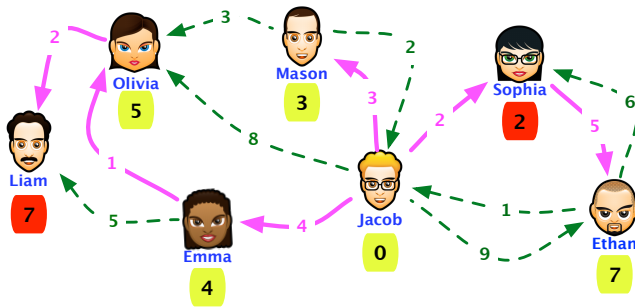
---

[1]http://imgur.com/

[2]http://www.reddit.com/

[3]https://www.tumblr.com/

[4]http://www.4chan.org/

Figure 1: Spread of a rumor in a social network. Each edge weight is the time it took for a rumor to pass along the edge. Solid magnet edges indicate the actual path through which the rumor spreads. Green dashed edges are alternative ways in which the rumor could have spread. The infection times of Sophia and Liam are observed (red squares); the infection times of the remaining nodes are hidden (yellow squares). How can an algorithm find that Jacob was the person who initiated the rumor?

networks [Backstrom et al., 2012], we find its performance on this type of networks underwhelming, as shown in Section 5.

Challenges. Previous approaches failed successfully address several challenges of the source identification problem, which we illustrate next using a toy example, shown in Figure 1.

— Partially observed infections. It has become difficult, if not impossible, to collect complete diffusion traces, and track each individual infection in online social and information networks. This problem is exacerbated by the need to develop methods that can provide outputs in (almost) real-time. For example, Spinn3r[5] crawls only a subset of the blogs periodically; Twitter's streaming API provides a small percentage (1%) of the full stream of tweets [Morstatter et al., 2013]; Facebook users typically keep their activity and profiles private [Sadikov et al., 2011]. It is thus necessary to develop methods that are robust to missing data [Chierichetti et al., 2011, Kim and Leskovec, 2011, Sadikov et al., 2011]. Our toy example illustrates this challenge by considering the infection times of Liam and Sophia to be observed and all other infection times to be missing (hidden or unobserved).

— Unknown infection start time. In most real-world scenarios, the exact time when a piece of malicious information starts spreading is unknown, and thus the observed infection times have only relative meaning. In our toy example, we know Liam got infected 5 time units later than Sophia but we do not actually observe how much time has passed between Jacob's infection, which triggered the spread, and Sophia's infection.

— Uncertain transmission delay. The spread of information over social and information networks is a stochastic process. Therefore, we need to consider probabilistic transmission models to capture the uncertainty. For example, our toy example illustrates the spread of a particular rumor and therefore considers a set of fixed edge delays (*e.g.*, the rumor took 5 time units to spread from Sophia to Ethan). However, the edge delays are stochastic and possibly different for every particular rumor (*e.g.*, a different rumor can take more or less than 5 time units to spread from Sophia to Ethan). The edge delay densities, or transmission densities, may depend on parameters like the content of the rumor or the users' influence.

— Unknown infection path. In large real world networks, we will often encounter a large number of potential paths that may explain the spread of a rumor from a source node and any other node in the network. In fact, the set of potential paths increases exponentially with network size and network density and even simply counting the number of paths requires non-trivial methods [Gomez-Rodriguez and Schölkopf, 2012a,b, Du et al., 2013a]. For example, in our toy example, Liam can become aware of the rumor through either Olivia or Emma.

Our Approach. To tackle these challenges, we propose a two-stage scalable framework: we first learn a continuous-time diffusion network model based on historical diffusion traces and then identify the source of an incomplete diffusion trace by maximizing its likelihood under the learned model. The key idea of our framework is to view the problem from the perspective of graphical models, and cast the problem as a maximum likelihood estimation problem, for which we find optimal solutions very efficiently using an importance sampling approximation to the objective and an optimization procedure that exploits the structure of the problem. Additionally, for networks with exponentially distributed edge transmission densities, used previously for modeling information propagation [Gomez-Rodriguez et al., 2011], we show that the objective is a piece-wise unimodal function with respect to the source's infection time and develop a more efficient search procedure.

For both synthetic and real-world data, we show that the framework can effectively "travel back to the past", and pinpoint the source node and its infection time significantly more accurately than other methods.

2 OUR FRAMEWORK

Our framework for solving the source identification problem consists of two main stages: it first learns a continuous-time diffusion network model based on historical diffusion traces, and then identifies the source of an incomplete diffusion trace and its initiation time by maximizing the its likelihood under the learned model. We start our exposition by revisiting the continuous-time generative model

---

[5]http://spinn3r.com/

for cascade data in social networks introduced in Gomez-Rodriguez et al. [2011], Du et al. [2013a].

2.1 Continuous-Time Model for Cascades

Given a *directed* contact network, $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with N nodes, a diffusion process begins with an infected source node s initially adopting certain *contagion* (idea, rumor or malicious piece of information) at time t_s. The contagion is transmitted from the source along her out-going edges to their direct neighbors. Each transmission through an edge entails a *random* spreading time, τ, drawn from a density over time, $f_{ji}(\tau; \alpha_{ji})$, parametrized by a transmission rate α_{ji}. Then, the infected neighbors transmit the contagion to their respective neighbors, and the process continues. We assume transmission times are independent and nonnegative, in other words, a node cannot be infected by a node infected later in time; $f_{ji}(\tau; \alpha_{ji}) = 0$ if $\tau < 0$. Moreover, an infected nodes remain infected for the entire diffusion process. Thus, if a node i is infected by multiple neighbors, only the neighbor that first infects node i will be the *true parent*.

The temporal traces left by diffusion processes are often called *cascades*. A cascade $\mathbf{t}$ is an N-dimensional vector $\mathbf{t} := (t_1, \ldots, t_N)$ recording the times when nodes are infected, if so, *i.e.*, $t_i \in [0, \infty]$, where T is the observation window cut-off and ∞ denotes nodes that did not get infected during the observation window. However, as noted above, in many scenarios, we only observe a subset of the infected nodes, $\mathcal{O}$, while the state of all other nodes, $\mathcal{H}$, is hidden (we assume the source node $s \in \mathcal{H}$). Our aim is then to find the source of a cascade s from the infection times $\{t_j\}_{j \in \mathcal{O}}$ of the subset of infected nodes $\mathcal{O}$. Figure 1 illustrates the observed data.

2.2 Cascade Likelihood

According to the conditional independence relation proposed in the continuous-time model for cascades, the complete likelihood of a cascade $\mathbf{t}$ (for both observed and hidden nodes) factorizes as

$$p(\mathbf{t}|t_s) = \prod_{i \in \mathcal{O} \cup \mathcal{H}} p\left(t_i | \{t_j\}_{j \in \pi_i}\right) \quad (1)$$

where π_i is the set of parents of i defined by the directed graph $\mathcal{G}$. For the infected nodes, Gomez-Rodriguez et al. [2011] showed that the likelihood can be further written as

$$p\left(t_i | \{t_j\}_{j \in \pi_i}\right) = \prod_{j \in \pi_i} S(t_i - t_j; \alpha_{ji}) \sum_{l \in \pi_i} H(t_i - t_l; \alpha_{li}),$$

where $S_{ji}(\tau; \alpha_{ji}) = 1 - F_{ji}(\tau; \alpha_{ji})$ is the survival function, $F_{ji}(\tau; \alpha_{ji}) = \int_0^\tau f_{ji}(t; \alpha_{ji}) dt$ is the cumulative distribution function, and $H_{ji}(\tau; \alpha_{ji}) = \frac{f_{ji}(\tau; \alpha_{ji})}{S_{ji}(\tau; \alpha_{ji})}$ is the hazard function, or instantaneous infection rate. We will focus on the Weibull family of distributions $f_{ji}(\tau; \alpha_{ji})$ since they have been shown to fit well real world diffusion data [Du

et al., 2013b]. In this case,

$$f_{ji}(\tau; \alpha_{ji}) = \frac{k\tau^{k-1}}{\alpha_{ji}^k} e^{-\left(\frac{\tau}{\alpha_{ji}}\right)^k}, \ S_{ji}(\tau; \alpha_{ji}) = e^{-\left(\frac{\tau}{\alpha_{ji}}\right)^k},$$

where k is a hyperparameter controlling the shape of the density. This family includes many well-known special cases, such as the exponential or Rayleigh distributions, which have also been used to model information propagation over information networks [Gomez-Rodriguez et al., 2010, Du et al., 2013b].

Unfortunately, to use Eq. 1, all infected nodes in a cascade need to be fully observed. If we only observe a subset $\mathcal{O}$ of the infected nodes, the likelihood of the incomplete cascade is computed as follows,

$$\begin{aligned} p(\{t_i\}_{i \in \mathcal{O}} | t_s) &= \int_\Omega p(\mathbf{t}|t_s) \prod_{j \in \mathcal{H}} dt_j \\ &= \int_\Omega \prod_{i \in \mathcal{O} \cup \mathcal{H}} p(t_i | \{t_j\}_{j \in \pi_i}) \prod_{j \in \mathcal{H}} dt_j, \quad (2) \end{aligned}$$

which essentially marginalize out the time for all hidden nodes $\mathcal{H}$ over a product space $\Omega := [t_s, \infty)^{|\mathcal{H}|}$. For simplicity of notation, we will omit the domain of the integration in the remainder of the paper.

The computation of the incomplete likelihood is a difficult high dimensional integration problem for continuous variables. We will address this technical challenge using importance sampling in Section 3.2.

2.3 Learning Diffusion Networks

Our framework relies on the assumption that it is possible to record a sufficiently large number of historical cascades, $\mathcal{C}$, in order to discover the existence of all nodes in the network, to infer the network structure as well as the model parameters, $\{\alpha_{ji}\}$. We note that it is not necessary to record cascades that cover all nodes and edges, but each cascade has to be *fully* observed to a sufficiently large time period. Furthermore, all cascades collectively need to cover the entire diffusion network. Under the precise conditions stated in [Daneshmand et al., 2014], one can infer the parameters of the continuous time model using an ℓ_1-regularized maximum likelihood estimation procedure.

2.4 Cascade Source Identification Problem

Given a learned diffusion model, our aim is to find the source node s of an incomplete cascade, such that the log-likelihood of the incomplete cascade is maximized. Thus, we aim to solve

$$s^* = \underset{s \in \mathcal{H}}{\operatorname{argmax}} \max_{t_s \in (-\infty, \min_{i \in \mathcal{O}} t_i)} p(\{t_i\}_{i \in \mathcal{O}} | t_s), \quad (3)$$

where $p(\{t_i\}_{i \in \mathcal{O}} | t_s)$ is defined in Eq. 2, and we assume that $t_s < \min_{i \in \mathcal{O}} t_i$. If we observe several independent incomplete cascades $\mathcal{D}$, all triggered by the same source

node, we will maximize their joint likelihood

$$s^* = \underset{s \in \mathcal{H}}{\operatorname{argmax}} \prod_{c \in \mathcal{D}} \left(\max_{t_s^c \in (-\infty, \min_{j \in \mathcal{O}} t_j^c)} \mathcal{L}_c \right), \quad (4)$$

where $\mathcal{L}_c := p\left(\{t_j^c\}_{j \in \mathcal{O}} | t_s\right)$. In the following sections, we will design algorithms to efficiently optimize the above objective and present experimental evaluations.

3 APPROXIMATE OBJECTIVE FUNCTION

There remain two technical challenges to solve to make our framework useful in practice. First, the likelihood of incomplete cascades, given by Eq. 2, is a difficult high dimensional integration problem over a continuous domain. We overcome this difficulty by an approximation algorithm, based on importance sampling, which will greatly simplify the integration. Second, the inner-loop maximization over the source timing in Eq. 4 is non-convex. We solve this by designing an efficient algorithm, which finds the global maximum by exploiting the piece-wise structure of the problem.

3.1 Importance Sampling

Since an analytical evaluation of the integral in Eq. 2 is intractable, we turn to a Monte Carlo approximation. To do so, in principle, we need to draw samples from the posterior distribution of latent variables, $p(\{t_i\}_{i \in \mathcal{H}} | t_s, \{t_i\}_{i \in \mathcal{O}})$, given the source time t_s and the times of the observed nodes, $\{t_i\}_{i \in \mathcal{O}}$. However, it is very challenging to sample from this posterior distribution, and we will instead address the problem by designing an efficient importance sampling approach.

More specifically, we first introduce a set of *auxiliary random variables* $\{\eta_i\}_{i \in \mathcal{O}}$, where each variable corresponds to one observed infected node, with an arbitrary joint probability distribution $\tilde{q}(\{\eta_i\}_{i \in \mathcal{O}})$. In the next section, we will briefly discuss how $\tilde{q}$ is chosen. Then, given the auxiliary distribution we have

$$p(\{t_i\}_{i \in \mathcal{O}} | t_s) = \int p(\{t_i\}_{i \in \mathcal{O} \cup \mathcal{H}} | t_s) \prod_{i \in \mathcal{H}} dt_i$$

$$= \int p(\{t_i\}_{i \in \mathcal{O} \cup \mathcal{H}} | t_s) \tilde{q}(\{\eta_i\}_{i \in \mathcal{O}}) \prod_{i \in \mathcal{H}} dt_i \prod_{i \in \mathcal{O}} d\eta_i. \quad (5)$$

Second, we introduce the *proposal distribution* for importance sampling on the auxiliary and hidden variables, $q(\{\eta_i\}_{i \in \mathcal{O}}, \{t_i\}_{i \in \mathcal{H}})$. Then, the integral becomes

$$p(\{t_i\}_{i \in \mathcal{O}} | t_s) = \int \frac{p(\{t_i\}_{i \in \mathcal{O} \cup \mathcal{H}} | t_s) \tilde{q}(\{\eta_i\}_{i \in \mathcal{O}})}{q(\{\eta_i\}_{i \in \mathcal{O}}, \{t_i\}_{i \in \mathcal{H}})}$$

$$q(\{\eta_i\}_{i \in \mathcal{O}}, \{t_i\}_{i \in \mathcal{H}}) \prod_{i \in \mathcal{H}} dt_i \prod_{i \in \mathcal{O}} d\eta_i$$

$$\approx \frac{1}{L} \sum_{l=1}^{L} \frac{p(\{t_i\}_{i \in \mathcal{O}}, \{t_i^l\}_{i \in \mathcal{H}} | t_s) \tilde{q}(\{\eta_i^l\}_{i \in \mathcal{O}})}{q(\{\eta_i^l\}_{i \in \mathcal{O}}, \{t_i\}_{i \in \mathcal{H}})}$$

$$\triangleq \phi_L(t_s), \quad (6)$$

where we draw L samples from $q(\{\eta_i\}_{i \in \mathcal{O}}, \{t_i\}_{i \in \mathcal{H}})$ to approximate the integral. Now, we have an approximation to $\mathcal{L}_c$. Next, we explain how to choose the *proposal* and the *auxiliary* distributions.

3.2 Choice of Proposal Distributions

We define our proposal distribution using the forward-generative process of the cascades. Our proposal distribution $q(\{\eta_i\}_{i \in \mathcal{O}}, \{t_i\}_{i \in \mathcal{H}})$ will sample cascades from the learned continuous diffusion network model with s as the source set. One of the interesting properties of this proposal distribution is that many terms involving the latent variables in Eq. 6 will be canceled out and hence the formula will become simpler.

We remind the reader that the independent cascade model has a useful shortest-path property [Du et al., 2013a], which allows us to sample the parents' infection times, $\{t_i\}_{i \in \pi_j}$, for each node j efficiently for different source infection times t_s. More specifically, we first sample a set of transmission times $\{\tau_{uv}\}_{(u,v) \in \mathcal{E}}$, one per edge, independent of each other. Then, the time t_i taken to infect a node i is simply the length of the shortest path in $\mathcal{G}$ from the source s to node i, where the edge weights correspond to the associated transmission times. Let $\mathcal{Q}_i(s)$ be the collection of directed paths in $\mathcal{G}$ from the source s to node i, where each path $q \in \mathcal{Q}_i(s)$ contains a sequence of directed edges (j, m), and assume the source node is infected at time t_s, then we obtain variable t_i via

$$t_i = g_i \left(\{\tau_{jm}\}_{(j,m) \in \mathcal{E}} | s\right) := \min_{q \in \mathcal{Q}_i(s)} \sum_{(j,m) \in q} \tau_{jl}, \quad (7)$$

where $g_i(\cdot)$ is the value of the shortest-path.

This above relation is key to speed up the evaluation of the sampled likelihood in Eq. 6 for different t_s values. First, the sampled transmission times τ_{uv} are independent and thus can be sampled in parallel. Second, we can reuse the sampled transmission times τ_{uv} for different t_s values and sources s, since the transmission times are independent of t_s and s. We only need to compute the infection time t_i for each node using $t_s = 0$, and then for a different value of t_s, the infection time is just an offset by t_s. Third, the likelihood of a sampled cascade $(\{\eta_i^l\}_{i \in \mathcal{O}}, \{t_i^l\}_{i \in \mathcal{H}})$ for $l = 1, \ldots, L$ can be simply computed using Eq. 1 as $p(\{\eta_i^l\}_{i \in \mathcal{O}}, \{t_i^l\}_{i \in \mathcal{H}} | t_s)$, which is independent of the actual value of t_s and depends only on the identity of the source node s.

3.3 Choice of Auxiliary Distribution

The auxiliary distribution $\tilde{q}(\{\eta_i\}_{i \in \mathcal{O}})$ is chosen to be equal to $p(\{\eta_i\}_{i \in \mathcal{O}} | \{t_i\}_{i \in \mathcal{H}})$. In other words, our auxiliary distribution will simply sample cascades from the learned continuous diffusion network model with $\mathcal{H}$ as the source set. Here, it is is easy to see that

---

Algorithm 1 Our source detection algorithm

Require: $\mathcal{C}, \mathcal{D}, L$

    Infer transmission rates $\mathbf{A}$ from $\mathcal{C}$ using [Daneshmand et al., 2014, Algorithm 1].

    Sample L sets of transmission times $\{\tau_{ij}\}_{(i,j)\in\mathcal{E}}$.

    Compute infection times $\hat{t}^l_{i\in\mathcal{V}}$, $l = 1, \ldots, L$ assuming $t_s = 0$ using Eq. 7.

    Compute change points: $t_i - \hat{t}^l_j, i \in \mathcal{D}, j \in \pi_i, l = 1, \ldots, L$ and $t_i - \hat{t}^l_j, j \in \mathcal{M}, i \in \pi_j \cap \mathcal{O}, l = 1, \ldots, L$.

    for $i \in \mathcal{M}$ **do**

        $t^*_i = \text{argmax}_{t_s} \phi_L(t_s)$ (using line search method or Lemma 2 in each piece)

    end for

    $s^* = \text{argmax}_{i\in\mathcal{M}} \phi_L(t^*_i)$

    $t_{s^*} = \max_{i\in\mathcal{M}} \phi_L(t^*_i)$

$\int p(\{\eta_i\}_{i\in\mathcal{O}} \,|\, \{t_i\}_{i\in\mathcal{H}}) \prod_{i\in\mathcal{O}} d\eta_i = 1$. With the above choices for the proposal and auxiliary distribution, we can greatly simplify the approximate likelihood in Eq. 6 into

$$\phi_L(t_s) = \frac{1}{L} \sum_{l=1}^{L} \prod_{i\in\mathcal{O}} p(t_i | \{t^l_j\}_{j\in\pi_i\setminus\mathcal{O}}, \{t_j\}_{j\in\pi_i\cap\mathcal{O}}) \quad (8)$$

$$\prod_{i\in\mathcal{M}} \frac{p(t^l_i | \{t^l_j\}_{j\in\pi_i\setminus\mathcal{O}}, \{t_j\}_{j\in\pi_i\cap\mathcal{O}})}{p(t^l_i | \{t^l_j\}_{j\in\pi_i\setminus\mathcal{O}}, \{\eta^l_j\}_{j\in\pi_i\cap\mathcal{O}})},$$

where $\mathcal{M}$ is the set of hidden nodes with observed variables as parents,

$$\mathcal{M} := \{i \in \mathcal{H} | \pi_i \cap \mathcal{O} \neq \emptyset\}, \quad (9)$$

which is typically much smaller than the overall set of hidden nodes. It is noteworthy that, under mild regularity conditions, the Monte Carlo approximation of the integral will converge to the true value with sufficient number of samples. However, a clever choice of the proposal distribution makes the convergence faster and the computation more efficient.

4 MAXIMIZE OBJECTIVE FUNCTION

Our objective function, given by Eq. 4, consists of an inner and an outer maximization. In the inner maximization, we leverage the Monte Carlo sample approximation and solve

$$\max_{t_s} \phi_L(t_s). \quad (10)$$

In the outer maximization, we rank all possible source nodes, s, in terms of their best starting time t_s, which is the solution to the inner maximization, and then select the top source node in the ranking as our optimal source, s^*.

The outer maximization is straightforward, however, the inner maximization, which consists of finding the optimal t_s that maximizes $\phi_L(t_s)$, defined in Eq. 8, may seem difficult at first. Although it is a 1-dimensional problem, the objective function is piece-wise continuous and non-convex with respect to t_s. This is because by increasing (or decreasing) t_s, the parent-child relation between nodes may change. However, there are two key properties of $\phi_L(t_s)$, which allow us to carry out the optimization efficiently. First, $\phi_L(t_s)$ is piece-wise continuous and the number of such

pieces increases as $O(L\Delta N)$, *i.e.*, linearly in the number of Monte Carlo samples, the number of observed nodes, and the maximum in-degree, Δ, of the observed nodes. Second, within each piece, the maximum of the function can be found efficiently.

4.1 Finding Each Continuous Piece

In this section, we aim to efficiently find all the change points t_{s_i} in the approximated likelihood $\phi_L(t_s)$, given by Eq. 8. In other words, we will efficiently find the left and right end points of each of its continuous pieces. Here, we assume there is a directed path in $\mathcal{G}$ from the source s to each of the observed infected nodes $\mathcal{O}$, otherwise, it cannot be a source for those nodes, trivially.

The key idea to finding all change points is realizing that each piece in Eq. 8 corresponds to a different *feasible* parents-child configuration. Here, by feasible parents, we mean parents that get infected earlier than the child and thus are temporally plausible. More specifically, given a source s, Eq. 8 is composed of three types of terms: $p(t_i | \{t^l_j\}_{j\in\pi_i\setminus\mathcal{O}}, \{t_j\}_{j\in\pi_i\cap\mathcal{O}})$ and $p(t^l_i | \{t^l_j\}_{j\in\pi_i\setminus\mathcal{O}}, \{t_j\}_{j\in\pi_i\cap\mathcal{O}})$, which depend on the source time value t_s, as we will realize shortly, and thus are responsible for the change point values t_{s_i}, and $p(t^l_i | \{t^l_j\}_{j\in\pi_i\setminus\mathcal{O}}, \{\eta^l_j\}_{j\in\pi_i\cap\mathcal{O}})$, which does not depend on t_s, because both $\{\eta^l_i\}_{i\in\mathcal{O}}$ and $\{t^l_i\}_{i\in\mathcal{M}}$ are sampled, t_s equally shifts all sampled times and its likelihood is time shift invariant. Based on the structure of the first two type of terms, it is easy to show that at each change point t_{s_i}, there is a node $j \in \mathcal{O} \cup \mathcal{M}$, observed or hidden, that changes its set of feasible temporally plausible parents, *i.e.*, a parent of one observed or hidden node becomes (stops being) a feasible parent at time t_{s_i}. Therefore, it is clear that there are $O(L\Delta N)$ change points, where Δ is the maximum in-degree of nodes. Next, we describe a procedure to find all change points efficiently.

Efficient Change Point Enumeration. We start by setting $t_s = 0$ and computing the infection time, denoted as $\hat{t}^l_j$, for each hidden node $j \in \mathcal{M}$ and realization l using the
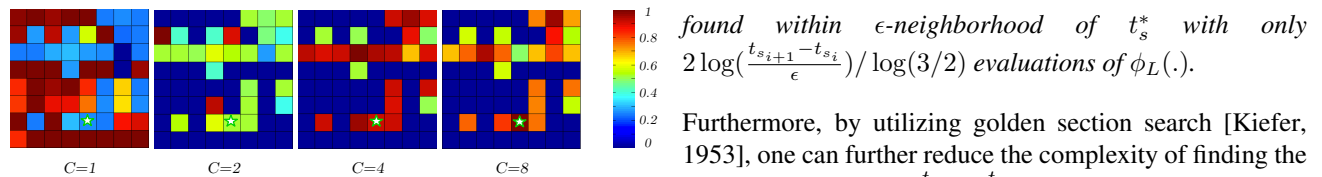
Figure 2: Evolution of the proposed method with respect to the number of cascades.

shortest path property described in Section 3.2. Then, we find the change points in which an observed node $i \in \mathcal{O}$ looses feasible parents by computing the time difference $t_i - \hat{t}_j^l, j \in \pi_i \backslash \mathcal{O}, l = 1, \ldots, L$, and the change points in which a hidden node $j \in \mathcal{M}$ earns feasible parents by computing the time differences $t_i - \hat{t}_j^l, i \in \pi_j \cap \mathcal{O}, l = 1, \ldots, L$. If a time difference is negative, we skip it, since the associated parent will never (always) be feasible, independently of the t_s value.

Additionally, we can compute $\phi_L(.)$ efficiently for each change point t_{s_i}, since at each change point t_{s_i}, we will only need to revaluate the corresponding terms to the node $i \in \mathcal{O} \cup \mathcal{M}$ that changes its set of feasible parents. In the case of exponential transmission likelihoods, once we have computed the likelihood at each change point t_{s_i}, we can re-evaluate it at any time $t \in [t_{s_i}, t_{s_{i+1}})$, by multiplying the corresponding terms in the approximated likelihood by $e^{t-t_{s_i}}$.

4.2 Maximizing within Each Piece

Once we have delimited each piece of the approximate likelihood given by Eq. 8, we can find the times t_s that maximize the likelihood in each piece efficiently, using well-known line-search procedures for one-dimensional continuous function, such as the forward-backward method, the golden section method or the Fibonacci method [Luenberger, 1973]. However, in the case of exponential transmission likelihoods, we can perform the maximization step even more efficiently.

Exponential Transmission. We start by realizing that, in the case of exponential transmission functions, the approximate likelihood given by Eq. 8 can be expressed as

$$\phi_L(t_s) = \sum_{l=1}^{L} \gamma_l e^{\beta_l t_s}, \tag{11}$$

where $\gamma_l > 0$ and β_l are independent of t_s. Then, we can prove that each piece of the approximate likelihood is uni-modal (proven in Appendix A):

Lemma 1 $\phi_L(t_s)$ *is uni-modal in* $t_{s_i} < t_s < t_{s_{i+1}}$.

Now, we can find the maximum of $\phi_L(.)$ by only evaluation of the function on a sequence of points (proven in Appendix B):

Lemma 2 *The maximum point of* $\phi_L(.)$ *can be*

found within ϵ-neighborhood of t_s^* with only $2 \log(\frac{t_{s_{i+1}} - t_{s_i}}{\epsilon}) / \log(3/2)$ *evaluations of* $\phi_L(.)$.

Furthermore, by utilizing golden section search [Kiefer, 1953], one can further reduce the complexity of finding the optimum point to $\log(\frac{t_{s_{i+1}} - t_{s_i}}{\epsilon}) / \log(1.618)$ evaluations.

We summarize the overall algorithm in Algorithm 1.

5 EXPERIMENTS

We evaluate the performance of our method on: *(i)* synthetic networks that mimic the structure of social networks and *(ii)* real networks inferred from a large cascade dataset, using a well-known state-of-the-art network inference method [Gomez-Rodriguez et al., 2011]. We show that our approach discovers the true source of a cascade or set of cascades with surprisingly high accuracy in synthetic networks and quite often in real networks, given the difficulty of the problem, and significantly outperforms several baselines and two state of the art methods [Aditya Prakash et al., 2012, Pinto et al., 2012]. Appendix C provides additional experimental results.

5.1 Experiments on Synthetic Data

Experimental Setup. We generate three types of Kronecker networks [Leskovec et al., 2010]: *(i)* core-periphery networks (parameter matrix: [0.9 0.5; 0.5 0.3]), which mimic the information diffusion traces in real world networks [Gomez-Rodriguez et al., 2010], *(ii)* random networks ([0.5 0.5; 0.5 0.5]), typically used in physics and graph theory [Easley and Kleinberg, 2010], and *(iii)* hierarchical networks ([0.9 0.1; 0.1 0.9]) [Clauset et al., 2008]. We then set the pairwise transmission rates of the edges of the networks by drawing samples from $\alpha \sim U(10, 5)$. For each type of Kronecker network, we generate 10 networks with 256 nodes and 512 edges. Finally, for each network, we generate a set of cascades from ten different random sources s^*. Since we are interested in detecting source nodes of *large* cascades, we only consider source nodes that triggered at least ten *large* cascades out of 100 simulated cascades. Given the size of the networks we experiment with, we consider a cascade to be *large* if it contains more than 40 nodes. Our aim is then to find the source of a *large* cascade or small set of *large* cascades from the infection times of a small (unknown) fraction of all infected nodes. In all the following experiments the sample size is 400 and 10% of the infected nodes are observed except when it is explicitly mentioned.

A Toy Example. We first consider a small 64-node hierarchical Kronecker network and visualize the approximate likelihood given by Eq. 8 against the number of observed cascades for each node in the network. We use 150 Monte Carlo samples. Figure 2 summarizes the results, where
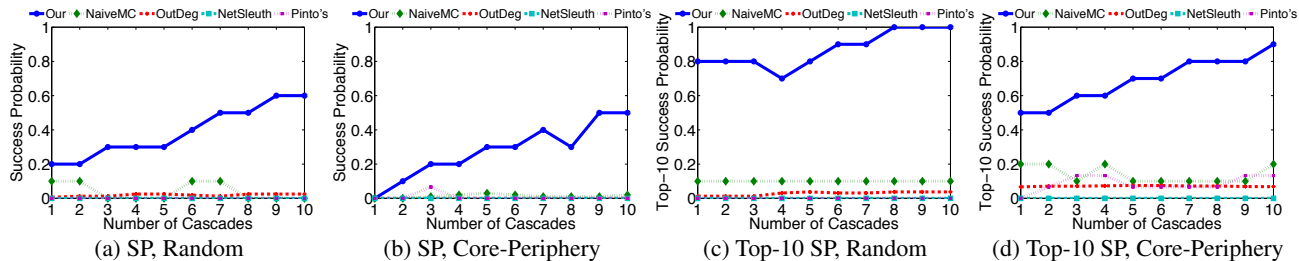
Figure 3: Success Probability (SP) and Top-10 Success Probability (Top-10 SP) for two types of Kronecker networks.

each square represents a node, the true source is marked with a star and the heat map represents normalized likelihoods in $[0, 1]$. In this toy example, a single cascade is insufficient to detect the true source, since it has a relatively low likelihood. However, once more cascades are observed, the likelihood of the true source increases and ultimately become higher than all other nodes for 8 cascades.

Accuracy. Next, we evaluate the accuracy of our method in comparison with two state of the art methods, NETSLEUTH [Aditya Prakash et al., 2012] and Pinto's method [Pinto et al., 2012], and two baselines in larger synthetic networks. The first baseline runs Montecarlo from each potential source and ranks them by counting the average maximum number of observed infected nodes that get infected in a time window equal to the length of observation window. Then, it ranks the potential sources according to the average value of this quantity, where the node with the highest value is the top node. The second baseline first finds all potential sources that can reach all observed infected nodes and then ranks them by decreasing out-degree, where the node with the highest out-degree is the top node. NETSLEUTH assumes the same infection probability β over all the edges, which we set to 0.1, following Aditya Prakash et al. [2012]. Pinto's method similarly assumes that all pairwise transmission times come from the same Gaussian distribution and they require its mean to be much larger than its standard deviation in order to guarantee nonnegative transmission times. In their work, they set $\mu/\sigma = 4$, where μ and σ are the mean and standard deviation, respectively. Since our fitted diffusion network contains edges with different transmission rates and thus different expected transmission time, we set the parameter μ to be the minimum expected value over all the edges.

We used two measures of accuracy: success probability and top-10 success probability. We define success probability as $P(\hat{s} = s^*)$ and top-10 success probability as the probability that the true source s^* is among the top-10 in terms of maximum likelihood or ranking. For each network type, we estimated both measures by running our method on 10 different random source sets. Since NETSLEUTH and Pinto's method can only accept one observed cascade at a time, we run the methods independently for each individual cascade
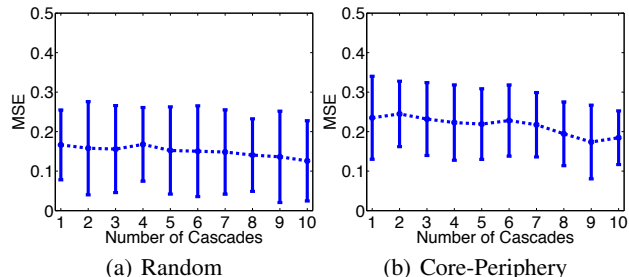

Figure 4: Mean-squared error (MSE) on the estimation of t_s for two types of Kronecker networks.

and then compute the top-1 and top-10 success probability based on all outputs for all cascades. Figure 3 summarizes the results for two types of Kronecker networks against the number of observed cascades. Our method outperforms dramatically all others, achieving a success probability as high as 0.6 and top-10 success probability of almost 1. The low performance that state-of-the-art methods exhibit, in comparison with the validation within the corresponding papers, may be explained as follows: in both cases, the authors validated their algorithms with synthetic and real networks with large diameters, without long-range connections, such as 2-D grids [Aditya Prakash et al., 2012] and spatial (geographical) networks [Pinto et al., 2012], where the source identification problem is much easier.

Source Infection Time Estimation. We also evaluate how accurately our method infers the infection time of the true source by computing the mean square error (MSE), $E_{s^*}\left[(t_{s^*} - \hat{t}_{s^*})^2\right]$, estimated by running our method on 10 different random sources. Here, we do not compare with other competitive methods since they do not provide an estimate of the infection time of the true source. Figure 4 shows the MSE of the estimated infection times of the true source for the same networks as above against the number of cascades.

5.2 Experiments on Real Data

Experimental Setup. We focus on the spread of memes, which are a short textual phrases (like, "lipstick on a pig") that travel almost intact through the Web [Leskovec et al.,
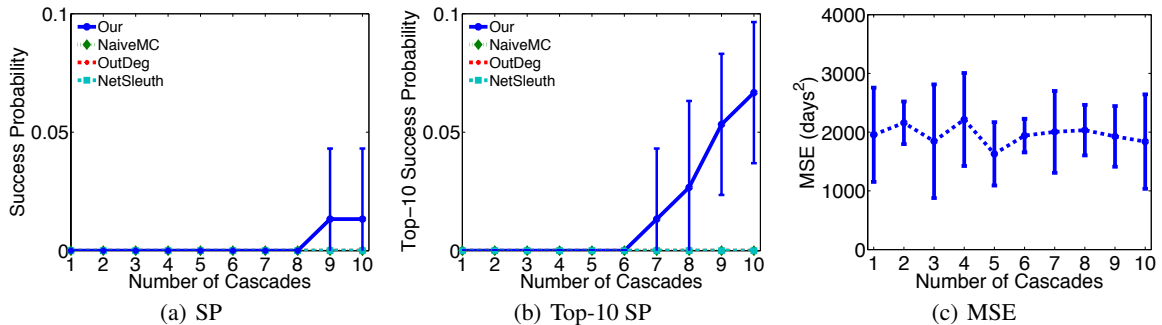
Figure 5: Success Probability (SP), Top-10 Success Probability (Top-10 SP) and mean-squared error (MSE) on the estimation of t_s for real cascade data.

2009]. We experiment with a large meme dataset[6], which traces the spread of memes across 1,700 popular mainstream media sites and blogs [Gomez-Rodriguez et al., 2013]. The dataset classifies memes per topic, and associates each meme m to an information cascade $\mathbf{t}_m$, which is simply a record of times when sites first mentioned meme m. We proceed as follows. We first infer an underlying diffusion network per topic using NETRATE, a well-known network inference method [Gomez-Rodriguez et al., 2011], using all observed information cascades. We then use these inferred networks along with a percentage of the infections of large cascades to infer the source of these cascades. We select 15 sources, each of them having at least 10 long cascades. Here, by long cascade we mean possessing more than 27 nodes. The results are averaged over 5 runs, randomizing the selection of the observed nodes, we consider that 10% of the infected nodes are observed and utilize 500 samples to approximate the likelihood.

Accuracy. We evaluate the accuracy of our method in comparison with NETSLEUTH and the same baselines as in the synthetic experiments, using success probability and top-10 success probability. Unfortunately, we cannot compare to Pinto's method because it requires the identity of the true parent for each observed node in each cascade, and this is not available in real cascade data. Figure 5 summarizes the results. Surprisingly, neither NETSLEUTH nor the baselines succeed at detecting cascade sources in real data, even with 10 observed cascades; they output solutions with an (almost) zero (top-10) success probabilities. In contrast, our method achieves a non-zero (top-10) success probability as long as we observe more than 8 and 6 cascades respectively, a fairly low number of cascades in this scenario. Even then, the performance of our method in terms of success and top-10 success probability may seem low at first, however, we would like to highlight how difficult the problem we are trying to solve is, by considering the performance of two simple random guessers. A first random guesser who chooses the source uniformly at

random from all nodes in the network would succeed with probability $1/1700 = 5.8 \times 10^{-4}$, almost 20 times less accurate than our method. A second random guesser that chooses the source uniformly at random among the nodes from whom the observed nodes are reachable would succeed with probability $1/425 = 2.4 \times 10^{-3}$, almost 5 times less accurate. The same argument for top-10 success probability shows 12 times improvement in accuracy compared to naive guesser and 3 times improvement in comparison to the more clever one. Finally, our method's MSE values indicate that our method is able to find the source infection time within an accuracy of $\sqrt{2000} \approx 45$ days. We find this quite remarkable given that the cascades we considered typically unfold during a 1-year period.

6 CONCLUSIONS

We propose a two-stage framework for detecting the source of a cascade in continuous-time diffusion networks, which improves dramatically over previous state-of-the-arts in terms of detection accuracy. Our framework cast the problem as a maximum likelihood estimation problem and then find optimal solutions very efficiently using an importance sampling approximation to the objective and an optimization procedure that exploits the structure of the problem. Our work opens many interesting venues for future work. For example, it would be useful to extend our method to support cascades with multiple sources and other continuous-time models different than the continuous-time independent cascade model [Gomez-Rodriguez et al., 2011]. Also, a theoretical analysis of our importance sampling scheme is also interesting. Finally, it would be interesting to apply the current framework to other real-world datasets.

---

[6]Data is available at http://snap.stanford.edu/infopath/

References

B. Aditya Prakash, J. Vrekeen, and C. Faloutsos. Spotting culprits in epidemics: How many and which ones? In *ICDM*, 2012.

L. Backstrom, P. Boldi, M. Rosa, J. Ugander, and S. Vigna. Four degrees of separation. In *WebSci*, 2012.

N. T. J. Bailey. *The Mathematical Theory of Infectious Diseases and its Applications*. Hafner Press, 2nd edition, 1975.

F. Chierichetti, J. Kleinberg, and D. Liben-Nowell. Reconstructing Patterns of Information Diffusion from Incomplete Observations. In *NIPS*, 2011.

A. Clauset, C. Moore, and M. E. J. Newman. Hierarchical Structure and the Prediction of Missing Links in Networks. *Nature*, 453(7191):98–101, 2008.

H. Daneshmand, M. Gomez-Rodriguez, L. Song, and B. Schölkopf. Estimating diffusion network structures: Recovery conditions, sample complexity & soft-thresholding algorithm. In *ICML*, 2014.

N. Du, L. Song, A. Smola, and M. Yuan. Learning Networks of Heterogeneous Influence. In *NIPS*, 2012.

N. Du, L. Song, M. Gomez-Rodriguez, and H. Zha. Scalable influence estimation in continuous-time diffusion networks. In *NIPS*, 2013a.

N. Du, L. Song, H. Woo, and H. Zha. Uncover topic-sensitive information diffusion networks. In *Artificial Intelligence and Statistics (AISTATS)*, 2013b.

D. Easley and J. Kleinberg. *Networks, Crowds, and Markets: Reasoning About a Highly Connected World*. Cambridge University Press, 2010.

M. Gomez-Rodriguez and B. Schölkopf. Influence Maximization in Continuous Time Diffusion Networks. In *Proceedings of the 29th International Conference on Machine Learning*, 2012a.

M. Gomez-Rodriguez and B. Schölkopf. Submodular Inference of Diffusion Networks from Multiple Trees. In *Proceedings of the 29th International Conference on Machine Learning*, 2012b.

M. Gomez-Rodriguez, J. Leskovec, and A. Krause. Inferring Networks of Diffusion and Influence. In *KDD*, 2010.

M. Gomez-Rodriguez, D. Balduzzi, and B. Schölkopf. Uncovering the Temporal Dynamics of Diffusion Networks. In *ICML*, 2011.

M. Gomez-Rodriguez, J. Leskovec, and B. Schölkopf. Structure and Dynamics of Information Pathways in Online Media. In *WSDM*, 2013.

M. Isaac. Nude Photos of Jennifer Lawrence Are Latest Front in Online Privacy Debate. *New York Times*, 2014.

D. Kedmey. Hackers Leak Explicit Photos of More Than 100 Celebrities. *Time*, 2014.

D. Kempe, J. M. Kleinberg, and E. Tardos. Maximizing the Spread of Influence Through a Social Network. In *KDD*, 2003.

J. Kiefer. Sequential minimax search for a maximum. *Proceedings of the American Mathematical Society*, 4(3): 502–506, 1953.

M. Kim and J. Leskovec. The network completion problem: Inferring missing nodes and edges in networks. In *SDM*, 2011.

T. Lappas, E. Terzi, D. Gunopulos, and H. Mannila. Finding Effectors in Social Networks. In *KDD*, 2010.

J. Leskovec, L. Backstrom, and J. Kleinberg. Meme-tracking and the dynamics of the news cycle. In *KDD*, 2009.

J. Leskovec, D. Chakrabarti, J. Kleinberg, C. Faloutsos, and Z. Ghahramani. Kronecker Graphs: An Approach to Modeling Networks. *JMLR*, 2010.

D. G. Luenberger. *Introduction to linear and nonlinear programming*, volume 28. Addison-Wesley Reading, MA, 1973.

F. Morstatter, J. Pfeffer, H. Liu, and K. Carley. Is the sample good enough? comparing data from twitters streaming api with twitters firehose. *ICWSM*, 2013.

P. C. Pinto, P. Thiran, and M. Vetterli. Locating the source of diffusion in large-scale networks. *Physical review letters*, 109(6):068702, 2012.

S. Sadikov, M. Medina, J. Leskovec, and H. Garcia-Molina. Correcting for Missing Data in Information Cascades. In *WSDM*, 2011.

D. Shah and T. Zaman. Detecting sources of computer viruses in networks: theory and experiment. In *ACM SIGMETRICS Performance Evaluation Review*, volume 38, pages 203–214, 2010.

K. Zhou, L. Song, and H. Zha. Learning social infectivity in sparse low-rank networks using multi-dimensional hawkes processes. In *AISTATS*, 2013a.

K. Zhou, H. Zha, and L. Song. Learning triggering kernels for multi-dimensional hawkes processes. In *ICML*, 2013b.