Multiple-Source Localization from a Single-Snapshot Observation Using Graph Bayesian Optimization

Zonghan Zhang, Zijian Zhang, Zhiqian Chen

Department of Computer Science and Engineering, Mississippi State University
zz239@msstate.edu, zz242@msstate.edu, zchen@cse.msstate.edu

Abstract

Due to the significance of its various applications, source localization has garnered considerable attention as one of the most important means to confront diffusion hazards. Multi-source localization from a single-snapshot observation is especially relevant due to its prevalence. However, the inherent complexities of this problem, such as limited information, interactions among sources, and dependence on diffusion models, pose challenges to resolution. Current methods typically utilize heuristics and greedy selection, and they are usually bonded with one diffusion model. Consequently, their effectiveness is constrained. To address these limitations, we propose a simulation-based method termed BOSouL. Bayesian optimization (BO) is adopted to approximate the results for its sample efficiency. A surrogate function models uncertainty from the limited information. It takes sets of nodes as the input instead of individual nodes. BOSouL can incorporate any diffusion model in the data acquisition process through simulations. Empirical studies demonstrate that its performance is robust across graph structures and diffusion models. The code is available at https://github.com/XGraph-Team/BOSouL.

Introduction

In recent decades, the world has become more interconnected thanks to the emergence of various networks. Consequently, we have become more vulnerable to network diffusion risks such as the spread of rumors, influenza-like viruses, and smart grid failures (Chowdhury, Srinivasan, and Getoor 2020; Ozili and Arun 2020; Amin and Schewe 2007). Source localization (SL), the reverse problem of information diffusion, has attracted significant attention from researchers as a necessary component of the confrontation against diffusion hazards (Prakash, Vreeken, and Faloutsos 2012; Zang et al. 2015; Wang et al. 2017; Zhu, Chen, and Ying 2017; Dong et al. 2019). It holds importance across various application domains such as medicine, security, large interconnected networks, social networks, and more(Shelke and Attar 2019; Li, Sun, and Chen 2007). Source localization can be leveraged to block negative influence (rumors and viruses), maintain infrastructure (power grid), determine accountability (propagators of rumors), and verify information reliability. For instance, negative news about an election candidate spread from the opponent's campaign office is less credible than one from a neutral third-party press.

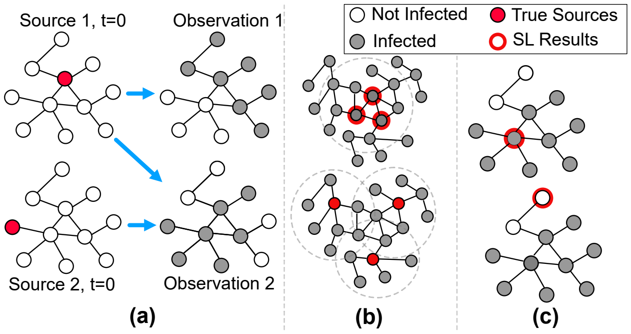

Figure 1: The challenges faced by heuristic methods: (a) Sources and spreads have many-to-many relationships. (b) Greedy algorithms result in sub-optimal solutions (top). (c) Different diffusion models (SI at the top and SIR at the bottom) affect source localization results.

The SL problems can be divided into three classes based on network observation types: complete observation, monitor observation, and snapshot observation (Shelke and Attar 2019). Among them, complete observation is rarely possible, given the massive scale of the real-world networks. Monitor observation, where part of the nodes are monitored for whether and when they are infected, is also not always available, especially for sudden outbreaks of unwanted propagation. Therefore, the most common scenario is single-snapshot observation, where only information about the network at a specific time is available (i.e., which nodes are infected at the snapshot). Early approach of single-snapshot source localization (Prakash, Vreeken, and Faloutsos 2012) identified single sources of infectious diseases. Later, multi-source methods were proposed (Wang et al. 2017; Zhu, Chen, and Ying 2017; Dong et al. 2019) to handle the wide existence of multiple diffusion sources in reality.

Although simulation-based methods seem to be legitimate to solve the SL problem, evaluating all node sets is infeasible due to the problem's combinatorial nature and the $\#P$-hardness of evaluating each set (Kempe, Kleinberg, and Tardos 2003). Therefore, heuristic methods dominate the problem of multi-source localization from a single network snapshot (Shah and Zaman 2010; Prakash, Vreeken, and Faloutsos 2012; Zang et al. 2015; Wang et al. 2017; Zhu, Chen, and

Ying 2017; Dong et al. 2019; Nie and Quinn 2019). However, this problem faces three major challenges that current heuristic methods cannot adequately address: **(1) Many-to-many relationship between sources and spreads.** Intuitively, one source set can result in various snapshots and vice versa. One snapshot may derive from multiple source sets with different probabilities. As shown in Figure 1(a), Source 1 can lead to two totally different spreads, Observation 1 and 2. At the same time, Observation 2 can initiate from either Source 1 or Source 2. It is difficult to properly model the many-to-many relationship using the limited information from a single snapshot due to the existence of this uncertainty (Cai, Xie, and Lui 2018). While heuristics efficiently approximate the localization, they only provide deterministic solutions without acknowledging the systematic uncertainty. **(2) Complicated interactions among the sources.** As a set of sources jointly spreads the influence through the network, considering their interrelationship is crucial for multi-source localization. Current heuristic methods approach the multi-source localization problem greedily (Wang et al. 2017; Dong et al. 2019; Nie and Quinn 2019). The highest-scoring nodes are selected as sources after scoring each node based on corresponding heuristics. Nevertheless, ignoring the interactions among the sources has significant consequences. Figure 1(b) illustrates the imperfection of localizing sources greedily. Given an infected sub-graph, nodes selected by a greedy algorithm lie at the center of the sub-graph (top), while the true sources might locate at the centers of multiple hidden communities (bottom). **(3) Heavy dependence on diffusion models.** Diffusion problems involve three key entities: sources, diffusion model, and spread. Given any two, the third can be inferred. Therefore, source localization requires a diffusion model as one of the inputs. The same spread combined with different diffusion models will result in different source localization results. As demonstrated in Figure 1(c), the same observation leads to different results when combined with SI (top) and SIR (bottom). In the SI model, the sources must be within the infected sub-graph. However, in SIR, the sources are likely to have already recovered since they have a higher recovery possibility than other nodes. Nearly all heuristic methods are designed for one specific diffusion model (Prakash, Vreeken, and Faloutsos 2012; Zang et al. 2015). Some claim to be model-free but make assumptions that implicitly constrain the diffusion model. For instance, LPSI (Wang et al. 2017) assumes that the sources are currently infected, indicating the diffusion model is like SI.

To address these issues, we propose a relatively efficient simulation-based method termed BOSouL. Specifically, we evaluate the candidate source set's likelihood of being the true source set via simulations. Bayesian optimization is employed for its sample efficiency to reduce the number of simulations. A surrogate function mimics the relationship between the candidate set and its likelihood. Sampling through clustering is conducted to guarantee the sampled instances in each Bayesian optimization iteration are evenly distributed in the search space. Our primary contributions include:

- **We propose an efficient simulation-based method utilizing Bayesian optimization**. BOSouL generates a Gaussian process that captures the uncertainty in the relationship between the set of sources and the observed snapshot. Furthermore, the Bayesian optimization paradigm significantly reduces the number of simulations, making the time cost of the algorithm acceptable.
- **The multiple sources are evaluated as a set instead of individually.** Thus, the interrelationship among the sources is included in BOSouL and its corresponding model. Comparing the greedy methods, the performance of BOSouL is more robust across different networks.
- **Our method can be combined with any diffusion model.** As long as we have a diffusion model that can well capture the diffusion pattern, we can adopt that model in the simulations to build the relationship between a source set and an observation.
- **We provide time complexity analysis. Extensive empirical experiments are conducted.** Real-world and synthetic datasets are employed to demonstrate BOSouL's superior performance. It is also displayed that BOSouL scales as well as most of the baselines, and the runtime is reasonably acceptable.

Related Work

Source localization, which aims to infer the origins of diffusion processes on networks given the diffused observation, has significant applications such as identifying rumor sources (Shelke and Attar 2019) and finding 'patient zero' in a pandemic (Scarpino and Petri 2019). It has attracted growing research interest in recent years (Shah and Zaman 2010; Prakash, Vreeken, and Faloutsos 2012; Zang et al. 2015; Wang et al. 2017; Zhu, Chen, and Ying 2017; Dong et al. 2019; Nie and Quinn 2019). Diffusion studies have presented multiple diffusion models, such as epidemic models like susceptible-infected (SI), susceptible-infected-recovered (SIR), and susceptible-infected-susceptible (SIS) (Brauer et al. 2019) and influence models like independent cascade (IC) and linear threshold (LT) (Kempe, Kleinberg, and Tardos 2003). However, early works focused on locating single sources under prescribed diffusion models. For instance, a few methods are designed specifically for SI model (Prakash, Vreeken, and Faloutsos 2012; Shah and Zaman 2010; Nie and Quinn 2019), and some others are designed for SIR (Zhu, Chen, and Ying 2017). Wang et al. (Wang et al. 2017) proposed a label propagation method named LPSI to detect multiple sources without knowing the underlying propagation model. Dong et al. (Dong et al. 2019) further enhanced LPSI by incorporating graph neural networks. However, since LPSI and its variants assume the sources are in the infected sub-graph, they implicitly suggest an SI-like diffusion model. Generally speaking, the current methods are bonded with certain diffusion models and lack generalizability. Also, most of the methods are simply greedily select sources based on single-source localization algorithms. The interrelationship among the sources is overlooked or intentionally ignored.

Bayesian Optimization is an approach for optimizing black-box functions that are expensive to evaluate. It constructs a probabilistic model of the objective function and uses this model to determine promising candidates to evaluate next (Frazier 2018). Bayesian optimization was first proposed by Mockus et al. (Mockus 1998) and has since become a popular methodology for hyperparameter tuning and optimization of complex simulations and models (Snoek, Larochelle, and Adams 2012). The key idea is to leverage Bayesian probability theory to model uncertainty about the objective function. A prior distribution is placed over the space of functions, often a Gaussian process, which is updated as observations are made. An acquisition function then uses this model to determine the next evaluation point by balancing exploration and exploitation. Some common acquisition functions include expected improvement, knowledge gradient, and upper confidence bound (Shahriari et al. 2015). There has been much work extending Bayesian optimization to handle constraints (Gelbart, Snoek, and Adams 2014), parallel evaluations (González et al. 2016), and high dimensions (de Freitas and Wang 2013). Overall, Bayesian optimization provides an elegant and principled approach to sample-efficient optimization of black-box functions. Bayesian optimization over a graph search space has emerged in the past decades. However, most of the works focus on node-level tasks and thus develop specific kernels for node smoothing (Ng, Colombo, and Silva 2018; Oh et al. 2019; Walker and Glocker 2019; Opolka and Liò 2020; Borovitskiy et al. 2021; Opolka et al. 2022). These works, while related, deal with a different task and the methods cannot be applied on our problem.

Method

We propose a Bayesian Optimization for Source Localization (BOSouL) approach that combines the Bayesian optimization paradigm with simulations to enable more precise inference of source sets from single-snapshot observations.

Problem Formulation

Definition 1 (Single-snapshot Multi-Source Localization). *Given **(1)** a size-N graph $G(V, E)$ where V and E represent vertices and edges, respectively. **(2)** one single observation of propagation snapshot represented by a vector $o^* = \{0, 1\}^N$, where 1 means the node is infected and 0 means otherwise, **(3)** the underlying diffusion model d, and **(4)** the source $s = \{0, 1\}^N$ with its cardinality $|s| > 1$. Note that any source node is not in the neighborhood of any other source node. The objective is to find the optimal s that maximizes its conditional probability $\mathbf{P}$:*

$$s = \arg\max_s \mathbf{P}(s|o^*, G, d), \tag{1}$$

where $\mathbf{P}$ is a conditional probability of s given o^*, G and d.

The difficulty presented by the aforementioned task is that the true source node set is unknown and cannot be retrieved during the source localization procedure. Therefore, it is impossible to compute the distance between the predicted and actual sources during the learning procedure. Extending Equation 1 with Bayes rule, we have:

$$\mathbf{P}(s|o^*, G, d) = \frac{\mathbf{P}(o^*|s, G, d)\mathbf{P}(s)}{\mathbf{P}(o^*)} \sim \mathbf{P}(o^*|s, G, d),$$

since no assumption is applied for $P(o)$ and $P(s)$, which will be set to uniform distribution as the prior probability. Then the task is changed to

$$s = \arg\max_s \mathbf{P}(o^*|s, G, d).$$

The probability of a candidate source set s is evaluated with the similarity between its simulated propagation spread o and the observed snapshot o^*. So the estimated source is:

$$\hat{s} = \arg\max_s \mathbf{P}(o^*|s, G, d) \sim \arg\max_s \mathrm{SIM}(o, o^*; G, d). \tag{2}$$

where o is the simulation result of s on G with d. SIM denotes a similarity metric between a pair of observations.

The Proposed Method: BOSouL

Overview. As shown in Figure 2, we propose a Bayesian optimization-based learning framework for one-shot multi-source localization. First, a kernel for the Gaussian Process (GP) is devised to measure the distance between source nodes, and then its validity is proven through theoretical analysis. Using the kernel, the output of the GP model is derived as a surrogate to predict the probability of a given source node set. Next, we initialize GP with multiple actual simulations and select sites with expected improvement (EI) iteratively. Ultimately, the optimal solution is determined by traversing all candidates within the designated range.

Gaussian Process Design. Consider a graph with N nodes. Node sets are typically associated with a binary vector, labeled as 1 if they are sources and 0 otherwise. This vector is represented with $s = \{0, 1\}^N$. With k sources, the total possible source configurations is $\binom{N}{k}$. Recognizing that not all nodes are equally significant in diffusion, like major cities in transport networks or key influencers in social networks, we focus on the top a nodes by degree. This reduces potential source combinations to $\binom{a}{k}$ where $a \ll N$.

Meanwhile, s is only a one-hot vector and lacks graph structure information. To illustrate, consider two 3-node sets: one original and the other formed by shifting each node by one hop based on the original one. Although the final observations are anticipated to be similar for these two sets, their similarity with the binary representations is quite low (0 in this case). This binary representation inadequately characterizes the similarity between two sets of nodes and violates the smoothness assumption imposed by the Gaussian process. Previous work for graph kernels prioritizes structural comparisons, often ignoring attributes over the graphs (Vishwanathan et al. 2010; Kriege, Johansson, and Morris 2020; Nikolentzos, Siglidis, and Vazirgiannis 2021; Siglidis et al. 2020). To overcome this constraint, we propose the introduction of a novel kernel that effectively combines structure information with theoretical validity. First, the source vector s) is transformed into its Fourier counterpart $\tilde{s}$ such that:

$$\tilde{s} = U^\top s, \quad \tilde{s}(i) = \sum_{i=1}^{n} s_i U^\top(i), \tag{3}$$

where $U^\top$ is the inverse eigenvectors of the graph Laplacian and serves a graph Fourier transformer. Combining the graph Fourier transform and RBF kernel, we have a new kernel termed as graph spectral Gaussian (GSG) kernel:

$$\mathcal{K}(x, x'; l) = exp(-\frac{||U^\top x - U^\top x'||^2}{2l^2}), \tag{4}$$

where l is a hyperparameter corresponding to the length-scale of the RBF kernel. Mercer kernels are essential for

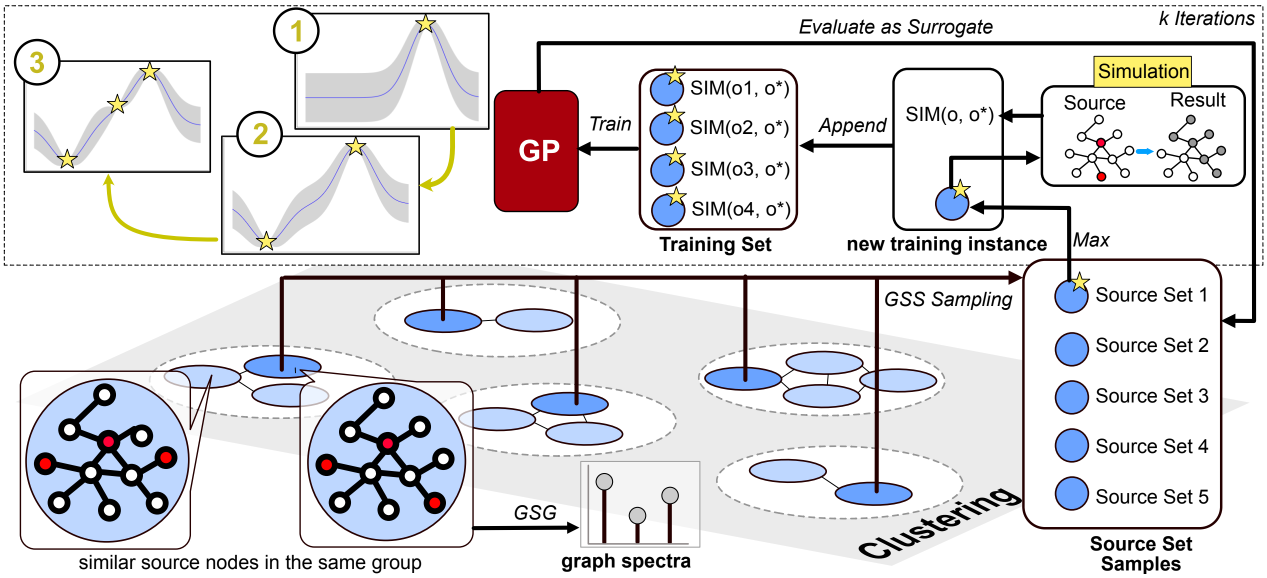

Figure 2: Illustration of the proposed BOSouL. Based on their graph Fourier representations, all possible candidates for source node sets are clustered in the bottom half of this figure. As training sets for GP, the model samples instances from each group utilizing GSS. In the upper half, one optimal instance is chosen by the surrogate GP (Max), and its likelihood of being a genuine source is determined through simulation. Sampling and training will be performed iteratively, and in each iteration, the GP model will be updated from a prior to a posterior (for example, from 1 to 2 on the upper left).

Gaussian Processes (GPs) as they ensure valid covariance matrices and enable implicit high-dimensional data mapping. Additionally, they offer computational benefits through the "kernel trick" in expansive spaces. Therefore, we analyze if the proposed kernel is a valid Mercer kernel.

Theorem. *GSG is a valid Mercer Kernel for GP.*

Proof. The kernel in Equation 4 can be transformed as follows:

$$\begin{aligned}
&\mathcal{K}(x, x'; l) \\
&= exp(-\frac{||U^\top x - U^\top x'||^2}{2l^2}) \\
&= exp(-\frac{[(U^\top x)^\top U^\top x + (U^\top x')^\top U^\top x' - 2(U^\top x)^\top U^\top x']}{2l^2}) \\
&= exp(-\frac{[x^\top x + x'^\top x' - 2x^\top x']}{2l^2}) = exp(-\frac{||x - x'||^2}{2l^2}).
\end{aligned}$$

Hence, $\mathcal{K}(x, x'; l)$ can be considered equivalent to the RBF kernel, which is widely recognized as a valid Mercer kernel. □

Next, we set up a Gaussian process (GP) with GSG kernel to realize Equation 2. This GP aims to estimate the existence probability of the provided source by evaluating the similarity between its corresponding and real observation.

$$GP : s \to \tau(o, o^*), \quad (5)$$

where o is one observation by simulation from s.

Data acquisition. The surrogate GP needs to be initialized and trained iteratively, which both resort to sampling techniques. Initialization requires sampling multiple data points, while each iteration selects another data point from a new set of samples by maximizing an acquisition function, which uses the GP posterior to balance exploration and exploitation. Due to the discrete property of the graph data, traditional sampling methods, such as the Sobol sequence (Sobol' 1967), do not fit the source localization problem. As a replacement, we propose a graph stratified sampling (GSS), which clusters the candidate and sample uniformly from each group. Specifically, GSS performs clustering over graph Fourier signals of candidate sources (Equation 3), and samples equal-size candidates from each cluster.

Theorem. *GSS has lower variance than random sampling.*

Proof. Simple random sampling randomly draws m samples from the entire population. The variance of its mean estimator is:

$$\mathrm{Var}(\bar{Y}_{\mathrm{rs}}) = \mathrm{Var}\left(\frac{\sum_{i=1}^{m} Y_i}{m}\right) = \frac{1}{m^2}\mathrm{Var}\left(\sum_{i=1}^{m} Y_i\right) = \frac{\sigma^2}{m},$$

where $\sigma^2 = \mathrm{Var}(Y_i)$ is the population variance. To set up GSS, we divide all candidates into κ non-overlapping equal-sized groups based on similarity. N is the population, and N_i is the population in i-th group. From the i^{th} group, m_i samples are drawn, with a total of $m = m_1 + m_2 + \cdots + m_\kappa$ samples. The variance of this GSS mean estimator is given by: $\mathrm{Var}(\bar{Y}_{\mathrm{gss}}) = \mathrm{Var}\left(\sum_{i=i}^{\kappa} \bar{Y}_i\right) = \sum_{i=1}^{\kappa} \left(\frac{N_i}{N}\right)^2 \frac{\sigma_i^2}{m_i}$, where σ_i is the sample mean of the i^{th} group. To demonstrate the variance reduction of GSS compared to simple random sampling, we compare $\mathrm{Var}(\bar{Y}_{\mathrm{gss}})$ and $\mathrm{Var}(\bar{Y}_{\mathrm{rs}})$. Note that the within-group similarity exists, so the variances within each group are smaller than the overall population variance, i.e., $\forall i, \sigma^2 \geq \sigma_i^2$. In addition, the sample size of each group is the same (i.e., $m_1 = m_2 = \ldots = m_c = \tilde{m}$, and $\kappa \cdot \tilde{m} = m$), the size of each group is the same ($\frac{N}{N_i} = \kappa$). So:

$$\mathrm{Var}(\bar{Y}_{\mathrm{gss}}) = \sum_{i=1}^{\kappa} \left(\frac{1}{\kappa}\right)^2 \frac{\sigma_i^2}{\tilde{m}} \leq \sum_{i=1}^{\kappa} \left(\frac{1}{\kappa}\right)^2 \frac{\sigma^2}{\tilde{m}} = \mathrm{Var}(\bar{Y}_{\mathrm{rs}}).$$

□

GSS clusters similar items within each group, thereby reducing within-group variance and, consequently, the estimator's overall variance. This would aid Bayesian Optimization in minimizing overall variance and drawing precise conclusions about actual sources. Note that the expected sample mean by GSS is identical to the sample mean by random sampling, which is the population mean. Consequently, a decrease in variance reduces inference errors.

Expected improvement (EI) is used to estimate the potential improvement of samples over the current best observation. Suppose the model clusters all candidates into b groups $C = \{c_1, c_2, \dots, c_b\}$, γ sets are sampled from each group such that $\{s_{ij}\}_{j=1}^{\gamma} \sim c_i$. We optimize EI over the sample set $[s_{11}, s_{12}, ..., s_{b\gamma}]$ such that:

$$\tilde{s}^* = \underset{\tilde{s}_{ij}\in[\tilde{s}_{11},\tilde{s}_{12},\dots,\tilde{s}_{b\gamma}]}{\text{argmax}} \text{EI}(\tilde{s}_{ij}) = \underset{\tilde{s}_{ij}}{\text{argmax}}\,\mathbb{E}[\delta(s_{ij}, s+)\cdot I(s_{ij})],$$

where $s+$ is the best set so far, s_{ij} is the node set that corresponds to the graph Fourier transform signal $\tilde{s}$, and $\delta(s, s+) = f(s; o^*) - f(s+; o^*)$. $I(s_{ij})$ is an indicator function that equals to 1 when $f(s_{ij}; o^*) > f(s+; o^*)$ and 0 when otherwise. Although the search space in our problem is finite, enumerating all node sets in each iteration violates our principle of efficiency. Thus, we strategically sample a few sets with GSS and use EI to pick the maximizer.

For the initial node sets and the one node set in each iteration, we need to query the true value of the objective function $\tau(o, o^*)$. The evaluation is achieved by simulations based on the given diffusion models, such as SI, SIS, or SIR. The proposed method BOSouL does not require the diffusion step as one of the inputs. Instead, our algorithm finds a simulation step t that maximizes the similarity between the diffusion spread from s and the given observation o^* in each simulation round. The similarity SIM, an integer evaluated by the Hamming distance between the two vectors, keeps updating as the time step grows. We expect the similarity to grow first as diffusion time step t grows. It peaks after a few steps and starts to decrease. On the one hand, with diffusion models like SI, the similarity decreases as the simulated spread suppresses the observed snapshot. On the other hand, with diffusion models like SIR and SIS, the similarity decreases when the diffusion waves of the simulation stagger the infected sub-graph of the observation. The simulation stops when the similarity shows a monotonically decreasing pattern and SIM is set to its historically high. After multiple rounds of simulations, we have:

$$\tau(o, o^*) = \mathbb{E}[\max_t \text{SIM}(o_t, o^*; G, d, t, s)]. \tag{6}$$

Algorithm

BOSouL is demonstrated in Algorithm 1. Initiated with graph G with n source nodes, one-shot observation o^*, given diffusion model d, budget k, and sample size γ, it aims to produce an n-sized node set s that approximates the true diffusion source. The algorithm selects the top a nodes as the candidate pool based on degree centrality. A graph Fourier transform is applied on all n-sized subsets of s^{pool} (lines 3-7). These transformed sets are clustered into c groups for later stratified sampling (line 8). One graph Fourier transform signal is randomly sampled from each cluster and evaluated by the peak similarity between the diffusion spread and the observation achieved during the simulations as discussed in the data acquisition section (line 12). The c pairs of Fourier representation of sources and similarities are used to train the GP model as an initialization (line 9-16). In each following iteration, a new group of data points is sampled by GSS, and one of them is picked by the EI acquisition function. After evaluation, the GP model is updated with the new signal-similarity pair, and the process repeats until convergence or the iteration budget is used up (line 17-25). After that, all candidate sets are evaluated with the model, and the maximizer is the estimated source set $\hat{s}$ (line 26).

Algorithm 1: **BOSouL**

Input: Graph G, source number n, observed snapshot o^*, diffusion model d, budget k, sample size γ
Output: A n-sized node set $\hat{s}$

```
1:  set $\tilde{S} \leftarrow \emptyset$, set $\Phi \leftarrow \emptyset$, simulation step $t \leftarrow 0$
2:  $s^{pool} \leftarrow$ top $a$ nodes by degree centrality
3:  $S \leftarrow$ all $n$-size node sets $\subset s^{pool}$
4:  for $s \in S$ do
5:      $\tilde{s} \leftarrow U^\top s$ as in Eq. 3
6:      $\tilde{S} \leftarrow \tilde{S} + \tilde{s}$
7:  end for
8:  cluster $\tilde{S}$ into $b$ groups: $C = \{c_1, c_2, \dots, c_b\}$
9:  sample 1 sets from each set group, $\{\tilde{s}_i\}_{i=1}^{b} \sim c_i$
10: for $\tilde{s}_i \in [\tilde{s}_1, \tilde{s}_2, ..., \tilde{s}_b]$ do
11:     $s_i \leftarrow S[loc(\tilde{s}_i)]$ where $loc(\cdot)$ is the index of $\cdot$ in $\tilde{S}$
12:     $o_t \leftarrow$ simulate $d$ on $G$ with $s_i$ and increasing $t$
13:     $\tau_i \leftarrow \mathbb{E}[\max_t$ SIM$(o^*, o_t)]$ as in Eq. 6
14:     $\Phi \leftarrow \Phi + (\tilde{s}_i, \tau_i)$
15: end for
16: train GP (as surrogate) with $\Phi$: $\tilde{s} \xrightarrow{GP} \tau$
17: while $z \neq 0$ ($z = k - b$) do
18:     sample $\gamma$ sets from each set group, $\{\tilde{s}_{ij}\}_{j=1}^{\gamma} \sim c_b$
19:     $\tilde{s}^* \leftarrow \arg\max_{\tilde{s}_{ij}}$ EI$(\tilde{s}_{ij})$, and $\tilde{s}_{ij} \in [\tilde{s}_{11}, \tilde{s}_{12}, ..., \tilde{s}_{b\gamma}]$
20:     $s^* \leftarrow S[loc(\tilde{s}^*)]$
21:     $o_t \leftarrow$ simulate $d$ on $G$ with $s$ and increasing $t$
22:     $\tau^* \leftarrow \mathbb{E}[\max_t$ SIM$(o^*, o_t)]$
23:     $\Phi \leftarrow \Phi + (\tilde{s}^*, \tau^*)$ and re-train GP with $\Phi$
24:     $z \leftarrow z - 1$
25: end while
26: Evaluate $S$ with GP: $\hat{s} = \arg\max_{s \in S}$ GP$(s)$
```

Time complexity

We analyze the time complexity of BOSouL based on Algorithm 1 and compare it with popular multi-source localization methods. Selecting a nodes with the highest degree centralities (line 2) is $\mathcal{O}(|V| + |E|) = \mathcal{O}(N^2)$ using BFS traversal. This complexity can be further reduced to $\mathcal{O}(N)$ for sparse graphs. Calculating the graph Fourier transform operator is $\mathcal{O}(N^3)$ (Merris 1994). Generating all n-sized node sets from s (line 3) requires $\mathcal{O}(a^n)$. Looping through all combinations has a time complexity of $\mathcal{O}(a^n)$, and the operations inside the loop are multiplications between $1 * N$ vectors and $N * N$ matrices, which are $\mathcal{O}(N^2)$. Thus, the time complexity for the whole block (line 4-7) is $\mathcal{O}(a^n N^2)$.

Methods	Jordan Centrality	LPSI	NetSleuth	LISN	BOSI_prep	BOSI_opti
Time Complexity	$\mathcal{O}(N^3)$	$\mathcal{O}(N^3)$	$\mathcal{O}(\lvert V_I\rvert + \lvert E_I\rvert + \lvert E\rvert)$	$\mathcal{O}(N^3)$	$\mathcal{O}(N^3)$	$\mathcal{O}(k^4)$

Table 1: Time complexities of BOSouL and other popular alternatives.

Clustering the graph Fourier signals is $\mathcal{O}(a^n)$. The operations above are the preparations for the GP training and have a combined time complexity of

$$\mathcal{O}(N^2 + N^3 + a^n + a^n N^2 + a^n) = \mathcal{O}(N^3), \tag{7}$$

when $a \ll N$ and n is a very small integer.

Breaking down the GP training process and the final prediction (line 9 - 26), we get all the operations carried out. There are $(b + b\gamma(k - b))$ samplings from the cluster, k simulations to evaluate the true similarity between the simulated spread and the true observation, $(k - b + 1)$ rounds of GP model training, and 1 evaluation for each candidate node sets using the trained GP. Assuming each simulation takes a long but constant time, the time complexities of sampling, simulation, GP training, and evaluation are $\mathcal{O}(1), \mathcal{O}(1), \mathcal{O}(|\Phi|^3)$, and $\mathcal{O}(1)$ (Rasmussen, Williams et al. 2006), respectively. Thus, the time complexity for the whole training and predicting period is

$$\mathcal{O}(b + bk\gamma - b^2\gamma + k + (k - b + 1)|\Phi|^3 + a^n) = \mathcal{O}(k^4) \tag{8}$$

since $|\Phi| \leq k$, b and g are constants, and a^n is ignorable compared to the problem size N. The overall time complexity of BOSouL is $\mathcal{O}(N^3 + k^4)$ where N is the graph size and k is the evaluation budget. This complexity is compared with other methods in Table 1. We can see that NetSleuth, proposed as an efficient algorithm, still has a better scalability. But BOSouL, as a simulation-based method, has the same time complexity as the other heuristics when the budget is relatively small. Also, note that the only operation bonded with the $\mathcal{O}(N^3)$, namely the eigendecomposition of the Laplacian matrix, runs only once in BOSouL; we expect its running time to grow slower than the other baselines.

Experiment

The experiments are carried out with 32 AMD EPYC 7302P 16-Core processors and 32GB RAM. Simulations are performed by NDLib (Rossetti, Milli, and Rinzivillo 2018), an open-source toolkit for diffusion dynamics. The baselines are realized by Cosasi (McCabe 2022), a Python package for graph diffusion source localization. The Bayesian optimization paradigm is implemented by BOTorch (Balandat et al. 2020) and gPyTorch (Gardner et al. 2018).

Configurations

We adopt SI, SIR, SIS, and IC as diffusion models. The infection rate is set to be 0.1 in the epidemic models, and the recovery rate in SIR and SIS is set to be 0.1. Each candidate source set is evaluated by an average of 100 simulation rounds. The Bayesian optimization paradigm includes 50 iterations to train the final model. **Datasets:** Three real-world datasets, namely Cora, CiteSeer, and PubMed (Yang, Cohen, and Salakhudinov 2016), reproduce the complex social network structure. Since the source localization problem is traditionally studied on connected graphs, we take the largest connected component of these graphs as the studied network. Two synthetic graphs are generated using NetworkX to represent pseudo social networks. They include *connected Watts-Strogatz small-world graphs* (SW) (Watts and Strogatz 1998) and *Erdős–Rényi random graphs* (ER) (Gilbert 1959). Each synthetic graph has $1,000$ nodes for effectiveness evaluation, and the average degree is around 10. We use SW graphs with sizes ranging from $1,000$ to $5,000$ for runtime analysis. **Baselines:** BOSouL is compared to three popular baselines. (1) Jordan centrality (JC) (Shah and Zaman 2010) greedily selects the nodes with the smallest maximum distances to other nodes in the infected sub-graph. (2) NetSleuth (Net) (Prakash, Vreeken, and Faloutsos 2012) is a highly efficient algorithm to identify the number and the location of sources under the SI model. (3) LISN (Nie and Quinn 2019) scores nodes based on shortest distance and maximum likelihood and selects nodes with the highest scores. **Metrics:** The result is evaluated by its distance from the true sources. In our experiment, the identified and true source sets are the same size. Thus, the distance between the two sets is calculated by $\mathcal{D}\{a, b\} = \min\{\Sigma_i \Delta(a_i, \hat{b}_i)\}$, where $\Delta(a_i, \hat{b}_i)$ represents the shortest distance between the nodes a_i and b_i, and $\hat{b}$ stands for a permutation of list b.

Results

The empirical study includes (1) Performance: the effectiveness of BOSouL is compared against the baselines; (2) Runtime Analysis: the time cost of BOSouL and the baselines are demonstrated to verify the time complexity analysis; and (3) Ablation Test: we compare BOSouL with two variants to evaluate the utility of our proposed GSG and GSS.

Performance. To demonstrate the compatibility of BOSouL, we test it along with the baselines with two significantly different diffusion models, SIR and SI. The infection rate for both models is 0.1, and the recovery rate for SIR is 0.1. For BOSouL, the budget of simulation is 70. The number of candidate nodes is 50, thus there are $\binom{50}{3} = 19,600$ candidate sets. They are clustered into 20 groups. Each method runs 10 times on each graph with different true source sets of size 3. The mean and standard deviation are reported as the final results. In SIR models where the source nodes might already recover and do not distinguish from the nodes that have never been infected, identifying the source nodes is much more challenging.

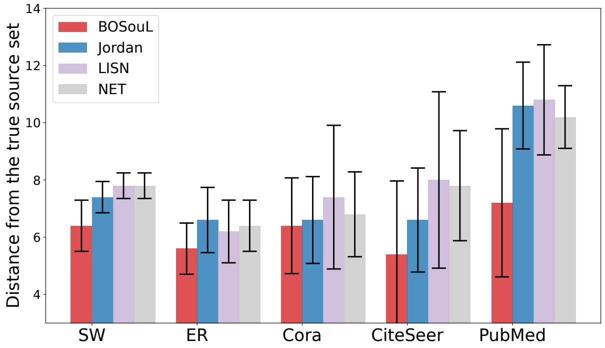

Figure 3: The distance toward the true source set with SIR.

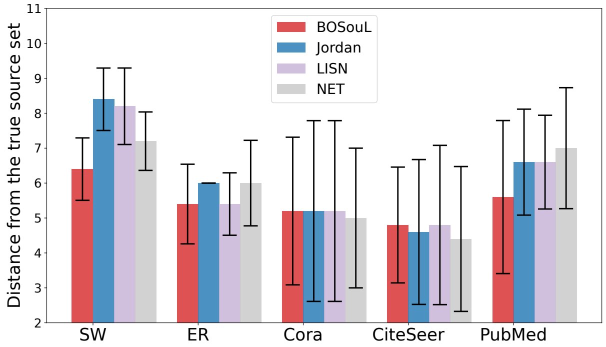

Figure 4: The distance toward the true source set with SI.

Figure 3 clearly illustrates the superiority of BOSouL when solving multi-source localization problems with the SIR model. It achieves the lowest localization error on all five datasets, demonstrating its effectiveness and diffusion model adaptability. Compared to other methods, the performance advance is most significant on SW, CiteSeer, and PubMed, with an enhancement of up to 29%. Understandably, the baseline methods do not perform well enough with the SIR model since they explicitly or implicitly assume the SI model. Thus, they only select nodes in the infected sub-graph. Figure 4 demonstrates that BOSouL also achieves competitive performance on all five graphs with the SI model. It outperforms the other four methods on four datasets. On CiteSeer, NetSleuth is the best performer but only leads BOSouL by 0.4. Additional experiments show that BOSouL surpasses all baselines across all five graphs under the IC model and outperforms baselines on four datasets except for Cora with the SIS model.

Runtime Analysis. As expected, BOSouL has longer running times than the baseline methods on SW graphs with increasing sizes due to the time spent on simulations. This difference is most significant when graph size $N = 1,000$. BOSouL takes 396.69 seconds, about six times slower than the slowest baseline LISN. Comparatively, Jordan centrality only takes 28.10 seconds, and NetSleuth needs 40.42 seconds. Those are 7.08% and 10.19% of BOSouL's running time, respectively. But this difference shrinks as the graph size grows. BOSouL almost scales linearly in the empirical experiment, evidenced by the steady increase in mean runtime. Jordan centrality spends the most time on eigendecomposition, which is part of BOSouL. Thus, it shows a similar pattern with a slightly higher rate of increase. NetSleuth, despite its lower time complexity, consistently takes more time than Jordan centrality as the graph size grows from $1,000$ to $5,000$. Also, it has a relatively large variance due to the term representing the size of the infected subgraph in its time complexity. Lastly, LISN's running time grows fastest as the problem size scales because it involves several matrix multiplications. On a size-$5,000$ connected small-world graph, BOSouL takes $2,018.98$ seconds, while LISN takes $1,989.67$ seconds. We can expect that on a larger graph, the running time of the latter will surpass that of the former. The running times for Jordan centrality and NetSleuth are 580.86 seconds and 704.72 seconds, respectively. The percentages compared to BOSouL raise to 28.77% and 34.90%. This trend is illustrated in Figure 5. Overall, BOSouL shows a stable scalability. The adaptation of Bayesian optimization makes the simulation-based approach tractable. Although its running time is longer than the faster baseline methods like Jordan centrality and NetSleuth, it remains competitive due to its superior performance. At the same time, the efficiency gap shrinks as the problem size grows.

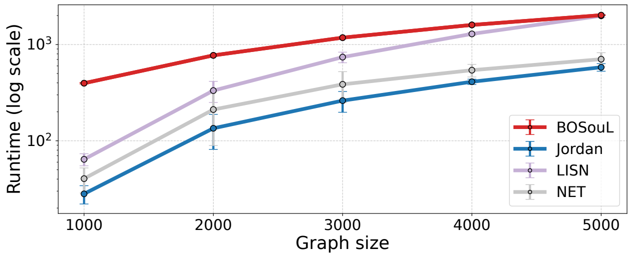

Figure 5: Runtime analysis on SW graph.

	SW	ER	Cora	CiteSeer	PubMed
raw	7.4±0.6	6.0±1.2	8.6±2.3	7.6±2.5	7.6±2.3
(GSG)	-10.8%	-3.3%	-23.2%	-18.4%	0%
half	6.6±1.5	5.8±0.5	6.6±1.1	6.2±1.8	7.6±2.2
(GSS)	-2.7%	-3.3%	-2.4%	-10.5%	-5.2%
full	6.4±0.9	5.6±0.9	6.4±1.7	5.4±2.7	7.2±2.6

Table 2: Ablation tests with SIR model.

Ablation Study. Table 2 compares the performance of our proposed BOSouL method (full) against two ablated versions: using random sampling (RS) instead of GSS for data acquisition and using an RBF kernel instead of the proposed GSG kernel. Column (GSG) shows the percentage decrease in localization error after substituting the RBF kernel with the GSG kernel. And Column (GSS) demonstrates the further performance increase brought by GSS. We can observe that RS+GSG (half) always performs at least as well as RS+RBF (raw). More specifically, except for PubMed, GSG brings performance enhancement ranging from 0.2 to 2.0, which is a $3.3\% - 23.2\%$ decrease in the localization error. This shows the benefits of the graph spectral Gaussian kernel for effective adaptation to the graph-structured data and the source localization problem. It is also demonstrated that GSS+GSG (full) outperforms RS+GSG on all five datasets. This shows the benefits of graph stratified sampling for uniform data acquisition. It explores the search space better than random sampling. In sum, our ablation study verifies the proposed components each provides significant gains over variants without those techniques. GSG kernel consistently assists in graph-structured data adaptation, fulfilling the smoothness assumption. Graph stratified sampling is crucial for handling more complex search spaces.

Conclusion

This study presents a simulation-based method BOSouL for multi-source localization from a one-shot observation. Bayesian optimization is adopted to foster efficiency and reveal a relationship between the node set and the observation. We theoretically prove that GSG, a graph-level kernel for the Gaussian process, is a valid Mercer kernel, and GSS, a stratified sampling method based on graph clustering, reduces variance better than random sampling.

Acknowledgements

This work is supported by NSF IIS award #2153369.

References

Amin, M.; and Schewe, P. F. 2007. Preventing blackouts. *Scientific American*, 296(5): 60–67.

Balandat, M.; Karrer, B.; Jiang, D.; Daulton, S.; Letham, B.; Wilson, A. G.; and Bakshy, E. 2020. BoTorch: A framework for efficient Monte-Carlo Bayesian optimization. *Advances in neural information processing systems*, 33: 21524–21538.

Borovitskiy, V.; Azangulov, I.; Terenin, A.; Mostowsky, P.; Deisenroth, M.; and Durrande, N. 2021. Matérn Gaussian processes on graphs. In *International Conference on Artificial Intelligence and Statistics*, 2593–2601. PMLR.

Brauer, F.; Castillo-Chavez, C.; Feng, Z.; et al. 2019. *Mathematical models in epidemiology*, volume 32. Springer.

Cai, K.; Xie, H.; and Lui, J. C. 2018. Information spreading forensics via sequential dependent snapshots. *IEEE/ACM Transactions on Networking*, 26(1): 478–491.

Chowdhury, R.; Srinivasan, S.; and Getoor, L. 2020. Joint Estimation of User And Publisher Credibility for Fake News Detection. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, 1993–1996.

de Freitas, N.; and Wang, Z. 2013. Bayesian Optimization in High Dimensions via Random Embeddings.

Dong, M.; Zheng, B.; Quoc Viet Hung, N.; Su, H.; and Li, G. 2019. Multiple rumor source detection with graph convolutional networks. In *Proceedings of the 28th ACM international conference on information and knowledge management*, 569–578.

Frazier, P. I. 2018. A tutorial on Bayesian optimization. *arXiv preprint arXiv:1807.02811*.

Gardner, J.; Pleiss, G.; Weinberger, K. Q.; Bindel, D.; and Wilson, A. G. 2018. Gpytorch: Blackbox matrix-matrix gaussian process inference with gpu acceleration. *Advances in neural information processing systems*, 31.

Gelbart, M. A.; Snoek, J.; and Adams, R. P. 2014. Bayesian optimization with unknown constraints. *arXiv preprint arXiv:1403.5607*.

Gilbert, E. N. 1959. Random graphs. *The Annals of Mathematical Statistics*, 30(4): 1141–1144.

González, J.; Dai, Z.; Hennig, P.; and Lawrence, N. 2016. Batch Bayesian optimization via local penalization. In *Artificial intelligence and statistics*, 648–657. PMLR.

Kempe, D.; Kleinberg, J.; and Tardos, É. 2003. Maximizing the spread of influence through a social network. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, 137–146.

Kriege, N. M.; Johansson, F. D.; and Morris, C. 2020. A survey on graph kernels. *Applied Network Science*, 5(1): 1–42.

Li, C.; Sun, Y.; and Chen, X. 2007. Analysis of the blackout in Europe on November 4, 2006. In *2007 International Power Engineering Conference (IPEC 2007)*, 939–944. IEEE.

McCabe, L. H. 2022. cosasi: Graph Diffusion Source Inference in Python. *Journal of Open Source Software*, 7(80): 4894.

Merris, R. 1994. Laplacian matrices of graphs: a survey. *Linear algebra and its applications*, 197: 143–176.

Mockus, J. 1998. The application of Bayesian methods for seeking the extremum. *Towards global optimization*, 2: 117.

Ng, Y. C.; Colombo, N.; and Silva, R. 2018. Bayesian semi-supervised learning with graph gaussian processes. *Advances in Neural Information Processing Systems*, 31.

Nie, G.; and Quinn, C. 2019. Localizing the Information Source in a Network. In *TrueFact 2019: KDD 2019 Workshop on Truth Discovery and Fact Checking: Theory and Practice*.

Nikolentzos, G.; Siglidis, G.; and Vazirgiannis, M. 2021. Graph kernels: A survey. *Journal of Artificial Intelligence Research*, 72: 943–1027.

Oh, C.; Tomczak, J.; Gavves, E.; and Welling, M. 2019. Combinatorial bayesian optimization using the graph cartesian product. *Advances in Neural Information Processing Systems*, 32.

Opolka, F.; Zhi, Y.-C.; Lio, P.; and Dong, X. 2022. Adaptive gaussian processes on graphs via spectral graph wavelets. In *International Conference on Artificial Intelligence and Statistics*, 4818–4834. PMLR.

Opolka, F. L.; and Liò, P. 2020. Graph convolutional gaussian processes for link prediction. *arXiv preprint arXiv:2002.04337*.

Ozili, P. K.; and Arun, T. 2020. Spillover of COVID-19: impact on the Global Economy. *Available at SSRN 3562570*.

Prakash, B. A.; Vreeken, J.; and Faloutsos, C. 2012. Spotting culprits in epidemics: How many and which ones? In *2012 IEEE 12th international conference on data mining*, 11–20. IEEE.

Rasmussen, C. E.; Williams, C. K.; et al. 2006. *Gaussian processes for machine learning*, volume 1. Springer.

Rossetti, G.; Milli, L.; and Rinzivillo, S. 2018. NDlib: a python library to model and analyze diffusion processes over complex networks. In *Companion Proceedings of the The Web Conference 2018*, 183–186.

Scarpino, S. V.; and Petri, G. 2019. On the predictability of infectious disease outbreaks. *Nature communications*, 10(1): 898.

Shah, D.; and Zaman, T. 2010. Detecting sources of computer viruses in networks: theory and experiment. In *Proceedings of the ACM SIGMETRICS international conference on Measurement and modeling of computer systems*, 203–214.

Shahriari, B.; Swersky, K.; Wang, Z.; Adams, R. P.; and De Freitas, N. 2015. Taking the human out of the loop: A review of Bayesian optimization. *Proceedings of the IEEE*, 104(1): 148–175.

Shelke, S.; and Attar, V. 2019. Source detection of rumor in social network–a review. *Online Social Networks and Media*, 9: 30–42.

Siglidis, G.; Nikolentzos, G.; Limnios, S.; Giatsidis, C.; Skianis, K.; and Vazirgiannis, M. 2020. Grakel: A graph kernel library in python. *The Journal of Machine Learning Research*, 21(1): 1993–1997.

Snoek, J.; Larochelle, H.; and Adams, R. P. 2012. Practical bayesian optimization of machine learning algorithms. *Advances in neural information processing systems*, 25.

Sobol', I. M. 1967. On the distribution of points in a cube and the approximate evaluation of integrals. *Zhurnal Vychislitel'noi Matematiki i Matematicheskoi Fiziki*, 7(4): 784–802.

Vishwanathan, S. V. N.; Schraudolph, N. N.; Kondor, R.; and Borgwardt, K. M. 2010. Graph kernels. *Journal of Machine Learning Research*, 11: 1201–1242.

Walker, I.; and Glocker, B. 2019. Graph convolutional Gaussian processes. In *International Conference on Machine Learning*, 6495–6504. PMLR.

Wang, Z.; Wang, C.; Pei, J.; and Ye, X. 2017. Multiple source detection without knowing the underlying propagation model. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 31.

Watts, D. J.; and Strogatz, S. H. 1998. Collective dynamics of 'small-world'networks. *nature*, 393(6684): 440–442.

Yang, Z.; Cohen, W.; and Salakhudinov, R. 2016. Revisiting semi-supervised learning with graph embeddings. In *International conference on machine learning*, 40–48. PMLR.

Zang, W.; Zhang, P.; Zhou, C.; and Guo, L. 2015. Locating multiple sources in social networks under the SIR model: A divide-and-conquer approach. *Journal of Computational Science*, 10: 278–287.

Zhu, K.; Chen, Z.; and Ying, L. 2017. Catch'em all: Locating multiple diffusion sources in networks with partial observations. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 31.