Online Diffusion Source Detection in Social Networks

Haishuai Wang†, Peng Zhang†, Ling Chen†, Huan Liu†, Chengqi Zhang‡
† Centre for Quantum Computation & Intelligent Systems, FEIT, University of Technology, Sydney, Australia
‡ Computer Science and Engineering, Arizona State University, Tempe, AZ, USA


Abstract—In this paper we study a new problem of online diffusion source detection in social networks. Existing work on diffusion source detection focuses on offline learning, which assumes data collected from network detectors are static and a snapshot of network is available before learning. However, an offline learning model does not meet the needs of early warning, real-time awareness, and real-time response of malicious information spreading in social networks. In this paper, we combine online learning and regression-based detection methods for real-time diffusion source detection. Specifically, we propose a new $l_1$ non-convex regression model as the learning function, and an Online Stochastic Sub-gradient algorithm (OSS for short). The proposed model is empirically evaluated on both synthetic and real-world networks. Experimental results demonstrate the effectiveness of the proposed model.

I. INTRODUCTION

Information propagation via social networks has attracted much attention. The spread of malicious information such as viruses, spams and rumors has made social networks vulnerable to various privacy attacks and viral advertising. To stop the propagation of malicious information, researchers recently proposed several models to detect the diffusion sources (“culprits”) in networks.

Existing diffusion source detection models can be roughly categorized into two classes: the snapshot-based source detection [13], [3], [5], [10] and the detector-based detection [11], [12], [14], [18]. The snapshot-based methods are under the assumption that a snapshot of the entire network can be obtained and the source nodes can be estimated under stochastic propagation models such as the SI [4] and SIR [2] models. Although these methods have shown promising results in experiments, fetching a snapshot of the entire network is very expensive, if not impossible. The detector-based methods assume that only a small subset of nodes in a network can be monitored, and the source nodes can be inferred from the observations (samples) from these detectors. This group of methods has recently attracted increasing attention due to its potential usage in real-world monitoring applications.

However, to our knowledge, most existing work on the diffusion source locating problem falls into the category of offline detection, where the data are assumed to be static and available all the time. In fact, for time-critical security monitoring applications in social networks, it is necessary to unveil the diffusion source nodes as soon as an observation arrives. This way, it is important to detect diffusion source nodes as early as possible to enable early warning, real-time awareness, and real-time response of malicious information spreading in social networks.

Motivated by the urgent demand of real-time and continuous diffusion source detection in social networks, we propose to use regression learning as the basic detection model. The regression learning is favorable for real-time applications due to the freely available prior distribution. Compared to offline source node detection methods, online detection models have the following challenges:

- **Challenge 1: how to design an online detection model?** The online detection model needs to address five problems as a whole, the unknown number of diffusion sources $k$, the partially activated detectors, the unknown initial propagation time $t^*$, the uncertain propagation path, and the uncertain propagation propagation time delay. The unknown number of diffusion sources $k$ expects a sparse solution of $x$. The partially activated detectors lead to a non-convex objective function. The uncertain propagation path and propagation time delay demand an aggregate Gaussian distribution to describe the estimated propagation time delay.

- **Challenge 2: how to design a stochastic learning algorithm to solve the online detection model?** Because the detectors are activated sequentially while a decision needs to be made once an observation arrives, the algorithm needs to digest observed data in a continuous and converging way. Ideally, the results of the online algorithm can approximate those of the offline learning algorithms.

- **Challenge 3: how to evaluate the performance of the proposed online detection method?** Given the unique characteristics of the problem, both real-world and synthetic data sets, as well as benchmark methods, are demanded to evaluate the performance of the model.

In this paper, we propose a new online regression learning model to detect diffusion sources in large-scale social networks. To solve **Challenge 1**, we use an $l_1$ non-convex regression learning model built on top of the shortest path propagation and Gaussian propagation time delay. To tackle **Challenge 2**, we present an Online Stochastic Sub-gradient algorithm (OSS for short) that can converge to local minima. Also, we evaluate the model using both synthetic and real-world social network data to address **Challenge 3**.

The contributions of the work are threefold:
We present a new online regression learning model to detect diffusion sources in social networks. The proposed model can handle the issues of the unknown number of diffusion sources $k$, the partially activated detectors, the unknown initial propagation time $t^*$, the uncertain propagation path, and the uncertain propagation time delay.

We present an online stochastic algorithm to solve the proposed online regression learning model. The algorithm uses a stochastic sub-gradient descent algorithm to continuously detect the source nodes.

We use both synthetic and real-world social network data to test the model. The results show that the proposed model outperforms the benchmark models.

II. RELATED WORK

Malicious information such as rumors and viruses has been observed recently propagating in social networks, which incurs privacy and security concerns [24] and motivates the research of diffusion source detection. To date, existing works on diffusion source detection focus on offline detection, where a snapshot of a large network or data harvested from detectors are assumed available in advance. In order to design an online detection algorithm, three technical questions need to be answered: 1) how to design stochastic propagation models, 2) how to design an objective function for online detection, and 3) how to design an online algorithm as the solution. In this section, we survey related work regarding the three aspects.

In terms of the stochastic propagation models in diffusion source locating, existing works simulate the spreading by using infection models such as the Susceptible-Infected-Recovered (SIR) [2], [3] model, the Susceptible-Infected SI [4], [5] model. These stochastic infection models can be regarded as a reverse engineering of the information maximization propagation models [27].

Based on the stochastic models, researchers proposed several models to infer the source nodes. The work [5] provided a systematic survey of locating rumor sources in a network, and presented a rumor centrality estimator to estimate the rumor sources by assigning a score to each infected node. The work [7] studied the problem of a single rumor source locating with priori knowledge. A recent work, Dynamic Message Passing (DMP) [13], presented a dynamic message-passing equation to estimate source nodes based on observed snapshots. Most existing estimators are based on either topological centrality measures [26] or distance measures between observed data. Then, maximum likelihood estimator can be used to infer the source nodes.

A limitation of the above works is that they all assume that the infection status of the nodes (i.e., labels) is known a priori. For example, the work [8], [10] considered the multiple infection sources estimation problem and assumed that the number of infection sources is unknown in advance. Some works [11], [9], [14] assumed that not all nodes are infected and only a subset of detectors are used for the source estimation.

In terms of the third question about online algorithm design, online learning algorithms have been extensively studied in machine learning [29], [30]. Typical methods include the Perceptron [31] and the Passive-Aggressive (PA) algorithms [32]. However, these online learning algorithms are based on linear and convex optimization, which do not fit our non-convex online learning problem. Moreover, the work [28], [33], [34], [35] studied a sparse online learning which considered limited size of active features. The work [35] proposed a FOBOS algorithm, which extended the Forward-Backward Splitting method. The work [28] proposed a general gradient-based method called truncated gradient to lead to sparsity. These works followed the idea of sub-gradient descent with truncation. The Regularized Dual Averaging (RDA) method follows the dual averaging method [34] and the learning variables are adjusted by solving a simple minimization problem that involves running average of all the past sub-gradients of the loss function.

None of the aforementioned works can directly address the online diffusion source node detection problem studied in this work.

III. PRELIMINARIES AND PROBLEM STATEMENT

In this section, we first introduce the diffusion process model, and then state the learning function in Eq. (11).

Diffusion Process Model: An underlying diffusion network can be modeled by a finite, undirected graph $G = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ is a set of $N$ vertices $\mathcal{V} = \{v_1, \ldots, v_N\}$ and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ is a set of $L$ edges. The sources $s^* \in \mathbb{R}^N$ is modeled as a random variable, and any node in the network is equally likely to be the source node.

Let $\mathcal{V}(u)$ denote the nodes that are directly connected to the node $u$. At time point $t_u$, node $u$ receives information from neighbors and passes the information to its neighbor $v \in \mathcal{V}(u)$ at time $t_u + \tau_{uv}$, where $\tau_{uv}$ denotes the random propagation time delay associated with the edge between $u$ and $v$.

Suppose we budget a set of detectors, denoted by $\mathcal{D} = \{d_i\}_{i=1}^{m}$. If the detectors are activated during the time window $[t^*, t^* + T]$, then they are activated nodes, denoted by $D_a$, and the remaining detectors are inactivated, denoted by $D_{ai}$. The goal is to spot the diffusion source nodes $s^* \in \mathbb{R}^N$, which can be estimated by the status of all detectors $D = \{(d_1, t^* - t^*), \ldots, (d_m, t_m - t^*)\}$, where $t_m$ is the activated time point of detector $d_m$, and label $t_m - t^*$ denotes the time delay. We assume that $s^*$ is uniformly distributed in the network. Then, the location of the sources can be recovered by maximizing...
the likelihood function of the observed status of the detectors $D$ given $s \in \mathbb{R}^N$, as shown in Eq. (1),

$$s^* = \arg \max_{s \in \mathbb{R}^N} P(D|s) = \arg \max_{s \in \mathbb{R}^N} P(D_u|s)[1 - P(D_n|s)],$$

(1)

where $P(D_u|s)$ denotes the probability of observing $D_u$ given a set of diffusion sources $s$, $P(D_n|s)$ denotes the probability of observing $D_n$ given a set of diffusion sources $s$, and $1 - P(D_u|s)$ denotes the probability that $D_u$ is inactivated under sources $s$. Thus, $P(D_u|s)[1 - P(D_n|s)]$ denotes the joint probability of both observing the activated nodes $D_u$ and inactivated nodes $D_n$.

Let $\prod_u^s$ denote all possible paths $\{P_s, d_{su}\}_{i=1}^m$ from a source $s$ to an activated detector $d_{su} \in D_u$. For each activated detector $d_{su}$, the probability $P(d_{su}|s)$ includes two parts: a) $P(\prod_u^s|s)$, which denotes the propagation path from the source node $s$ to the activated detector $d_{su}$, and b) $P(d_{su}|\prod_u^s)$, which denotes the time delay under the path $\prod_u^s$. Similarly, $P(d_{su}|s)$ also has two parts, i.e., the propagation path $\prod_u^s$ and the the time delay $P(d_{su}|\prod_u^s)$ along the path. Therefore, Eq. (1) can be converted to Eq. (2).

$$s^* = \arg \max_{s \in \mathbb{R}^N} \left[ \sum_{\prod_u^s} P(\prod_u^s|s)P(D_u|\prod_u^s)[1 - \sum_{\prod_n^s} P(\prod_n^s|s)P(D_n|\prod_n^s)] \right].$$

(2)

However, there are two types of randomness in Eq. (2): a) the probability $P(\prod_u^s|s)$, which is the uncertainty in the propagation paths $\prod_u^s$ that the information takes to reach the activated (inactivated) detectors, and b) the probability $P(D_u|\prod_u^s)$, which is the uncertainty of the propagation time delay.

The propagation of rumors and virus generally follows the snowball phenomenon [17]. Hence, we can use the shortest-path, denoted by $P(u,v)$ between nodes $u, v \in V$, to approximately represent the propagation path in the network. In the worst case, the search for the shortest path takes time $O(N^2)$.

We consider that the random variable $\theta_{uv}$ of the propagation delay along edge $e_{uv} \in E$ is independent and identically distributed and $\theta_{uv} \sim N(\mu_{uv}, \sigma_{uv}^2)$. The time delay between any two nodes $u$ and $v$ can be calculated as the aggregate of all the delays $\theta_{uv}$ along the shortest path $e_{uv} \in P(u,v)$ under the central limit theorem, as shown in Eq. (3).

$$\theta_{uv} = \sum_{e_{uv} \in P(u,v)} \theta_{uv} \sim N(\mu T x, (\sigma^2)^T x),$$

where $\mu = (\mu_1, \cdots, \mu_E)^T$ is the mean vector, $x \in \mathbb{R}^N$ is a selector vector, which selects edges under the shortest path using the breadth-first graph search (BFS).

Thus, the probability $P(D_u|s)$ in Eq. (1) can be rewritten as follows,

$$P(D_u|s) = \prod_{i=1}^m P(\prod_i^s|s)P(D_u|\prod_i^s) = \prod_{i=1}^m P(\theta_{su,i,d}),$$

(4)

where $m_s$ is the number of observers during the time window, and $\theta_{su,i,d}$ is the time delay. Based on Eq. (3), the mean and variance of the random variable $\theta_{su,i,d}$ are $\mu_{su,i}^2$ and $\sigma_{su,i}^2$ respectively. Let $\Lambda = (\sigma_{su,i}^2)^T x$, the Eq. (4) can be converted to Eq. (5).

$$P(D_u|s) = \prod_{i=1}^m (2\pi\Lambda)^{-1/2} e^{-(\mu_{su,i}^2 x - \sigma_{su,i}^2)^2/(2\Lambda)}.$$
IV. THE PROPOSED ALGORITHMS

In this section, we present an Online Stochastic Sub-gradient algorithm to solve the regression model in Eq.(11). The proposed regression model in Eq.(11), compared to the classical regression learning, has several new challenges: 1) the dependent variable $t$ is implicit, because the initial propagation time is unknown, 2) the $l_1$ non-convex objective function expects sparse and fast convergent algorithm, and 3) data collected from detectors arrive continuously. To solve the challenges, we use the Relative Time Difference of Arrivals (part A), Non-convex to Convex Approximation (part B), and Sub-gradient (part C) as the solutions. The Online Stochastic Sub-gradient algorithm is summarized in part D.

A. Relative Time Difference of Arrivals

The dependent variable $t$ in Eq. (11) is implicit, because the initial propagation time $t^*$ is unknown. To solve this challenge, we can use an “anchor node” to cancel out the initial time $t^*$.

Assume the $\alpha^{th}$ detector $d_\alpha$ is the “anchor node”, its activated time is

$$t_\alpha = t^* + \sum_{e_i \in \mathcal{P}(s^*,d_\alpha)} \theta_i,$$

where $\theta_i$ is the time delay along edges $e_i \in \mathcal{P}(s^*,d_\alpha)$. Then, the Relative Time Difference of Arrivals (RTDA) between $d_\alpha$ and the $k^{th}$ ($k \neq \alpha, 1 \leq k \leq m_\alpha$) detector $d_k$ is

$$T_k := t_k - t_\alpha = \sum_{e_i \in \mathcal{P}(s^*,d_k)} \theta_i - \sum_{e_i \in \mathcal{P}(s^*,d_\alpha)} \theta_i.$$

Based on RTDA, Eq. (11) has an elementary column change, i.e.,

$$\bar{A}(;c_\alpha) := A(;c_\alpha) - A(;c_\alpha),$$

$$\bar{B}(;c_\alpha) := B(;c_\alpha) - B(;c_\alpha).$$

Thus, the dimension of $T$, $\bar{A}$ and $\bar{B}$ are $\bar{T} \in \mathbb{R}^{m_\alpha - 1}$, $\bar{A} \in \mathbb{R}^{N \times (m_\alpha - 1)}$ and $\bar{B} \in \mathbb{R}^{N \times (m_\alpha - 1)}$ respectively. Then, Eq. (11) can be rewritten to

$$s^*(x) = \arg \min_{x \geq 0} \frac{1}{2} \| \bar{T} - \bar{A}^T x \|^2_2 - \lambda \| \bar{B}^T x \|^2_2 + \rho \| x \|_1.$$  \hspace{1cm} (13)

B. Non-convex to Convex Approximation

To address the non-convex challenge, we wish to find a sequence of convex programs, through which the non-convex function can be approximated and converge to a local minimum. To obtain a convergent sequence, at step $k+1$, the concave part $-\lambda \| \bar{B}^T x \|^2_2$ is linearly approximated by using the differential at the previous iterative point $x_k$, i.e.,

$$< \frac{\partial}{\partial x} (\| \bar{B}^T x \|^2_2), x_k, x > \geq -2\lambda \bar{B}^T \bar{B} x_k, x.$$  \hspace{1cm} (14)

At step $k+1$, we only solve a convex optimization as follows:

$$x_{k+1} \left\{ \min_{x \geq 0} \frac{1}{2} \| \bar{T} - \bar{A}^T x \|^2_2 - 2\lambda \bar{B}^T \bar{B} x_k + \rho \| x \|_1 \right\}.$$  \hspace{1cm} (15)

C. Sub-gradient

We use sub-gradient method to solve the $l_1$ regularization problem. Let $\mathcal{J}(x) = f(x) + g(x) = \frac{1}{2} \| \bar{T} - \bar{A}^T x \|^2_2 - 2\lambda \bar{B}^T \bar{B} x_k, x,$

$$\nabla_j \mathcal{J}(x) = f_j(x) + g'_j(x) = \frac{1}{2} \{ \sum_{i=1}^{m_\alpha} (\bar{T}_i - a_i^T x)_j \} + [-2\lambda \sum_{i=1}^{m_\alpha} (b_i^T b_i x_k)]_j$$

$$= \sum_{i=1}^{m_\alpha} (\bar{T}_i - a_i^T x)_j (\bar{T}_i - a_i^T x) + [-2\lambda \sum_{i=1}^{m_\alpha} b_i^T b_i x_k]$$

$$= (\bar{T} - \bar{A}^T \bar{A} x)_j - 2\lambda \bar{A}^T \bar{B} \bar{B} x_k.$$  \hspace{1cm} (16)

So $s^*(x) = \arg \min_{x \geq 0} \mathcal{J}(x) + \rho \| x \|_1$ and this objective function is non-differential because of the $l_1$ norm term. Assume $X = (\bar{x}^1, \cdots, \bar{x}^n)^T$ is the global optimal point. Consider the $j^{th}$ variable $\bar{x}_j$. The first-order optimality conditions are:

$$\left\{ \begin{array}{l} \nabla_j \mathcal{J}(x) + \rho \text{sign}(\bar{x}_j) = 0, \quad \text{s.t. } |\bar{x}_j| > 0 \\
\nabla_j \mathcal{J}(x) + \rho e : c \in [-1, 1], \quad \text{s.t. } \bar{x}_j = 0 \end{array} \right\}$$

These conditions can be used to define a sub-gradient for each $\bar{x}_j$:

$$\nabla \mathcal{J}(x) + \rho \text{sign}(\bar{x}_j), \quad |\bar{x}_j| > 0$$

$$\nabla \mathcal{J}(x) + \rho, \quad \bar{x}_j = 0, \quad \nabla \mathcal{J}(x) < \rho$$

$$\nabla \mathcal{J}(x) - \rho, \quad \bar{x}_j = 0, \quad \nabla \mathcal{J}(x) > \rho$$

Thus, based on the three solutions, the sub-gradient method uses iterations:

$$x_{k+1} = x_k - \eta_t (\nabla \mathcal{J}(x)),$$  \hspace{1cm} (19)

where the parameter $\eta_t > 0$ is the learning rate. In our analysis, we only consider constant learning rate with a fixed $\eta_t > 0$.

D. The Online Stochastic Sub-gradient Algorithm

In this part, we design an Online Stochastic Sub-gradient detection algorithm to continuously infer the diffusion sources. Algorithm 1 summarizes the solution to Eq. (13), where the sparse non-convex regression is solved by iteratively calculating the convex program in Eq. (15) and the non-smooth $l_1$ program in Eq. (19). In terms of online learning, we use stochastic sub-gradient iterations to calculate $\nabla \mathcal{J}(x)$ during the time window $T$, i.e.,

$$\nabla \mathcal{J}(x) = (\bar{T}_i - a_i^T x)^T (-a_i)$$  \hspace{1cm} (20)

As shown in Algorithm 1, the proposed online algorithm calls two functions for continuous learning. Function 1 corresponds to the online computation within the time window $T$ where the detectors are activated one by one. Function 2 corresponds
Algorithm 1 The Online Detection Algorithm

Input:
\( \mathcal{G} \): Network Graph; \\
\( \mathcal{D} \): Detectors \( \{ D_a \cup D_b \} \);
\( \mu \): The propagation mean parameters \( \mu = (\mu_1, \ldots, \mu_{|E|})^T \);
\( \sigma \): The propagation variance parameters \( \sigma = (\sigma_1, \ldots, \sigma_{|E|})^T \);
\( \epsilon \): Stop criteria;

Output:
\( \mathbf{s}^* = \{ s_1, \ldots, s_n \} \): A set of diffusion sources \( \mathbf{s}^* \);
1: \( d_{s_i}\) \( \leftarrow \) anchor(\( D_{s_i}() \)) ; // randomly pick anchors
2: if \( t < T \) then
3: for each \( d_i \in D_a \) do
4: for each \( v_j \in \mathcal{V} \) do
5: \( \mathcal{P}(d_j, v_j) \leftarrow \text{BFS} (\mathcal{G}, d_i, v_j) ; // \text{breath first} \)
6: \( A_{ij} = \sum_{c_k \in \mathcal{P}(d_i, v_j)} \mu_k ; // \text{matrix A} \)
7: \( B_{ij} = \sum_{c_k \in \mathcal{P}(d_i, v_j)} \mu_k ; // \text{matrix B} \)
8: if \( i > \alpha \) then
9: for each \( k \leq m_a \) do
10: \( \vec{A}(:, c_k) \leftarrow A(:, c_k) - A(:, \hat{c}_k) ; // \text{matrix } \vec{A} \)
11: \( \vec{t} = t_k - t_a ; // \text{build } \vec{t} \)
12: end for
13: \( \lambda^* \leftarrow \text{Function 1}(\vec{A}, \vec{t}, \epsilon, \eta_i) ; // \text{Call Function 1} \)
14: end if
15: end for
16: end for
17: else
18: for each \( k \leq m_a \) do
19: \( \vec{B}(:, c_k) \leftarrow B(:, c_k) - B(:, \hat{c}_k) ; // \text{matrix } \vec{B} \)
20: end for
21: \( \lambda^* \leftarrow \text{Function 2}(\vec{A}, \vec{B}, \vec{t}, \epsilon, \eta_i) ; // \text{Call Function 2} \)
22: end if
23: return \( \mathbf{s}^* = \mathcal{G}(\lambda^*) \);

The corresponding online solutions of \( x \) approximate the offline optimal solution when the time window \( T \) ends.

V. EXPERIMENTS

In this section, we report experimental results on both synthetic and real-world social network data sets. The experiments are designed to validate 1) the optimal parameters for the new model, 2) the superiority of the proposed model compared with benchmark methods, and 3) the performance on real-world applications. The algorithms are implemented in Java with Matlab as the convex solution package. The system is a Linux Ubuntu server with 8*2.9GHz CPU and 32G memory.

A. Data sets

We use four synthetic data sets and two real-world data sets(Facebook\(^1\) and Twitter\(^2\)) data sets for parameter study, performance testing and algorithm comparison.

Synthetic networks were generated using four network models: Erdos-Renyi\([19]\), Barabasi Albert\([21]\), Forest Fire\([22]\) and Small world\([20]\). Facebook and Twitter network\([23]\) are online social networking and microblogging services that enable users to post and read text messages. These data sets are publicly available at http://snap.stanford.edu/data which contains of ‘circles’ (or ‘lists’) from Facebook (10 networks) and Twitter (973 networks). These statistics are compiled by combining the ego-networks.

The parameter settings are listed in Table I.

B. Experimental Setup

In our experiments, we use both synthetic and real-world data sets to test our proposed online algorithm. Because we do not know the actually diffusion sources of the Facebook and Twitter data sets, we choose sources randomly in these data sets. We conduct experiments at different detecting time, and test different learning rates of \( \eta_t \) for the online locating algorithm.

We assess our methods w.r.t the average distance (hops) between actual locations of the diffusion sources and the estimated locations. Smaller hops indicate that the algorithms have higher accuracy. Let \( S^* \) denote the actual source, and the estimated source set as \( \hat{S} \). Since the size of \( S^* \) may not be equal to that of \( \hat{S} \), we measure their distance \( h(S^*, \hat{S}) \) by calculating the average number of hops between each element \( s_i \in \hat{S} \),

\[
h(S^*, \hat{S}) = \frac{1}{|S^*|} \sum_{s_i \in S^*} ||s_i - f(S^*, s_i)||^2, \tag{22}
\]

\(^1\)http://snap.stanford.edu/data/egonets-Facebook
\(^2\)http://snap.stanford.edu/data/egonets-Twitter.
Fig. 2. Parameter study on synthetic data sets by using the proposed regression learning model and Online Stochastic Sub-gradient OSS algorithm. The number of hops (error rate) w.r.t. (A) the diffusion sources number \(k\); (B) the detector number \(m\); (C) the monitoring time window \(T\); (D) the parameter \(\lambda\); (E) the parameter \(\rho\). (F) the online detection. The average distance on synthetic data sets w.r.t propagation time.

Table I. List of synthetic and real-world datasets in our experiments.

<table>
<thead>
<tr>
<th>DataSet</th>
<th>Nodes</th>
<th>Edges</th>
<th>Avg. Degree</th>
<th>Avg. Path length</th>
<th>Avg. Clustering Coefficient</th>
<th>Other parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Albert-Barabasi</td>
<td>1000</td>
<td>3990</td>
<td>7.98</td>
<td>3.172</td>
<td>0.037</td>
<td>(n = 4)</td>
</tr>
<tr>
<td>Forest Fire</td>
<td>1000</td>
<td>3428</td>
<td>6.86</td>
<td>4.736</td>
<td>0.536</td>
<td>(f = 0.35)</td>
</tr>
<tr>
<td>Small World</td>
<td>1000</td>
<td>3999</td>
<td>7.99</td>
<td>5.079</td>
<td>0.473</td>
<td>(\alpha = 4, p = 0.32)</td>
</tr>
<tr>
<td>Erdos-Renyi</td>
<td>1000</td>
<td>6000</td>
<td>12.00</td>
<td>4.214</td>
<td>0.012</td>
<td>default parameters</td>
</tr>
<tr>
<td>Twitter</td>
<td>32985</td>
<td>763713</td>
<td>30.54</td>
<td>5.702</td>
<td>0.627</td>
<td>(\mu = 0.01, \sigma = 0.01)</td>
</tr>
<tr>
<td>Facebook</td>
<td>4039</td>
<td>176468</td>
<td>19.78</td>
<td>3.892</td>
<td>0.671</td>
<td>(\mu = 0.01, \sigma = 0.01)</td>
</tr>
</tbody>
</table>

Fig. 3. Parameter \(\eta_t\) in the proposed Online Stochastic Sub-gradient OSS algorithm.
and \|s_i - f_i(S^*, s_i)\|^2 denotes the hops between nodes \(s_i\) and \(f_i(S^*, s_i)\).

C. Experimental Results

**Parameter study.** We first test the parameters in Eq. (13) \(w.r.t\) the number of diffusion sources \(k\), the number of detectors \(m\), the length of monitoring time window \(T\), and the parameters \(\lambda\) and \(\rho\). The default parameters are: the number of diffusion sources \(k = 5\), the number of detectors \(d = 20\% \times N\), the monitoring time window \(T = 5\) minutes. The selection of sources and detectors are random. The propagation time delay along each edge follows a Gaussian \(\theta_i \sim N(1, 0.01)\). Figure 2 demonstrates the model performance \(w.r.t\) different parameters on the four synthetic datasets.

**The number of diffusion sources \(k\).** From Figure 2(A), we can see that the error rate (hops) decreases \(w.r.t\) the number of diffusion sources. We can conclude that the more diffusion sources, the higher probability to be found out.

**The number of detectors \(m\).** Figure 2(B) shows the average distance (hops) \(w.r.t\) the number of detectors. The result demonstrates that the accuracy improves \(w.r.t\) the number of detectors.

**The monitoring time window \(T\).** The monitoring time window indicates the detection time span. Figure 2(C) shows that the error hops drop with the time window.

**The parameter \(\lambda\).** The parameter \(\lambda\) controls the preference between the activated nodes (convex part) and the inactivated nodes (concave part). As shown in Figure 2(D), the parameter \(\lambda\) should weigh the convex part and the concave part. If \(\lambda\) is selected too large, it overfits the inactivated nodes; otherwise, it overfits the activated nodes.

**The parameter \(\rho\).** \(\rho\) is the regularization parameter, and restricts the size of \(x\). If \(\rho\) is too large, the algorithm surfers from time cost, especially on large scale networks. From Figure 2(E), we observe the minimal hops when \(\rho = 6\).

**Continuous detecting test.** We test the proposed online algorithm on the synthetic date sets with parameter \(\lambda = 2, \rho = 6\). Also, we empirically study the learning rate parameter \(\eta_t\) in the sub-gradient algorithm. Figure 2(F) shows the performance of the algorithm OSS. The hops reduce along with the propagation time because more activated detectors lead to better performance. At the end of the time window, the hops shrink as the inactivated detectors are obtained. Figure 3 demonstrates the \(OSS\) algorithm with learning rates \(\eta_t\) on the synthetic and real-world data sets. The algorithm usually reaches the least average distance when \(\eta_t\) is 0.1.

**Propagation time delay and path test.** We also test the propagation path and propagation time delay by real-world data harvested from Sina Weibo (http://weibo.com), which is a popular Chinese microblogging website. We crawl data from a movie community with 570 users. A user is labeled infected if s/he posts or forwards a target topic/keyword, e.g., a rumor. We randomly choose a subset of detectors, where page scripts ran continuously during the monitoring time window to crawl and analyze the texts of the detected nodes. A node can be taken as an activated detector if it contains a target keyword.

**Propagation time delay test.** We randomly test the weibo time delay in the last half year of ten pairs of nodes. We totally collected 24,760 tweets after removing noise. From Figure 4(A), we can observe that the average time a tweet takes to be forwarded is about 15 minutes in this community, and the distribution approximately complies with Gaussian. Figures 4(B) and 4(C) show the shortest path test and the number of tweets propagated on Sina Weibo respectively. As shown in Figure 4(A), we can observe that the average time a tweet takes to be forwarded is about 15 minutes in this community, and the distribution approximately complies with Gaussian.

**OSS algorithm running time test.** Figure 4(A) shows the average time delay (min) \(w.r.t\) the number of tweets. The OSS algorithm takes about 2000s out of 40,000 (77.6\%) tweets pass through the shortest path. Thus, we can use the shortest path approximation in our model and algorithms.

**ACKNOWLEDGMENTS**

This work was supported by the Australia ARC Discovery Project (DP140102206).
REFERENCES


