Multiple Rumor Source Detection with Graph Convolutional Networks

Ming Dong
Huazhong University of Science and Technology
vitsing@hust.edu.cn

Bolong Zheng*
Huazhong University of Science and Technology
bolongzheng@hust.edu.cn

Nguyen Quoc Viet Hung
Griffith University
henry.nguyen@griffith.edu.au

Han Su
University of Electronic Science and Technology of China
hansu@uestc.edu.cn

Guohui Li
Huazhong University of Science and Technology
guohuili@hust.edu.cn

ABSTRACT

Detecting rumor source in social networks is one of the key issues for defeating rumors automatically. Although many efforts have been devoted to defeating online rumors, most of them are proposed based an assumption that the underlying propagation model is known in advance. However, this assumption may lead to impracticability on real data, since it is usually difficult to acquire the actual underlying propagation model. Some attempts are developed by using label propagation to avoid the limitation caused by lack of prior knowledge on the underlying propagation model. Nonetheless, they still suffer from the shortcoming that the node label is simply an integer which may restrict the prediction precision. In this paper, we propose a deep learning based model, namely GCNSI (Graph Convolutional Networks based Source Identification), to locate multiple rumor sources without prior knowledge of underlying propagation model. By adopting spectral domain convolution, we build node representation by utilizing its multi-order neighbors information such that the prediction precision on the sources is improved. We conduct experiments on several real datasets and the results demonstrate that our model outperforms state-of-the-art model.

CCS CONCEPTS

• **Information systems → Data mining**.

KEYWORDS

Rumor, Source identification, Graph convoluational networks

ACM Reference Format:
Ming Dong, Bolong Zheng, Nguyen Quoc Viet Hung, Han Su, and Guohui Li. 2019. Multiple Rumor Source Detection with Graph Convolutional Networks. In *The 28th ACM International Conference on Information and Knowledge Management (CIKM '19), November 3–7, 2019, Beijing, China.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3357384.3357994

*Bolong Zheng is the corresponding author

CIKM '19, November 3–7, 2019, Beijing, China

ACM ISBN 978-1-4503-6976-3/19/11…$15.00
https://doi.org/10.1145/3357384.3357994

1 INTRODUCTION

The emergence and development of social networks have dramatically changed people's lifestyle. In the current social networks, the roles of users have changed from passive recipients to active publishers. However, due to the distributed and decentralised nature of social networks, content on social platforms is propagated without any type of moderation and may thus contain incorrect information. As proved in many studies [16, 17, 43], rumors spread rapidly in social networks, which may cause huge detriment to society. Consequently, the problem of automatically rumor defeating has attracted lots of attentions in recent years [3, 20–22, 24, 25, 32]. In general, most of the studies related to rumor defeating are divided into two categories. One is the content-based rumor identification, which mainly focuses on building multiple features of rumor content, such as rumor text, spreading model or user behavior etc. In this way, an automatic classifier is usually trained to distinguish whether a given text is a rumor or not. This line of studies is helpful for monitoring rumor spreading and mining the accounts that are suspicious to spread rumors. However, it is difficult to locate the propagation source of rumor, which is a significant task in rumor detection on social network. Once the rumor sources are detected, we are able to cut off the critical paths of rumor propagation such that the rumor spreading at early stage can be easily controlled. Therefore, another line of research has been conducted on the problem of rumor source detection in social network based on the network structure.

The methods of rumor source detection take the information propagation mode and the network structure into consideration, which can be classified into two types: (1) infection status based analysis [29, 37]; (2) observer node deployment [38, 40, 42]. In practice, the former has a broader application prospect since we cannot deploy observers or acquire related information in some real-world situations. However, in most of the methods on infection status based analysis, it is a prerequisite that the underlying propagation model is known in advance. Obviously, it is difficult to obtain the model information in practice. To our knowledge, only a few studies have been carried out for source identification without the requirement of knowing the underlying information propagation model. LPSI [34] is the first attempt on this problem, which propagates

integer labels in network and predicts the rumor sources based on the convergent node labels.

In this paper, we study the multiple rumor source detection (MSRD) problem, which is similar to the setting of LPSI. The input is an undirected social network, and the output is a set of rumor sources. The aim is to produce a set of rumor sources that is as close as possible to the set of ground-truth rumor sources. Although LPSI is able to locate rumor sources without any information of underlying propagation model, it still suffers from the shortcoming that the node label is simply an integer and thus restricts the prediction precision. To this end, we propose a supervised learning based model, namely Graph Convolutional Networks based Source Identification (GCNSI), to locate multiple rumor sources without prior knowledge of underlying propagation model. In GCNSI, we propose an input generation algorithm to produce training samples, and adopt GCN layers to capture features based on two assumptions, i.e., *source prominence* and *rumor centrality*, which describe the rules of rumor propagation. Since MSRD is a supervised learning based task, we modify the input and loss function of traditional GCN layers.

The major contributions of this work can be summarized as follows:

- To the best of our knowledge, this work is the first attempt that applies the Graph Convolutional Networks technique on MSRD problem.
- We proposed a supervised learning based model named GCNSI that modifies the input and loss function of traditional GCN in order to fit the supervised learning based task. In addition, we design an input generation algorithm for producing training data for GCNSI.
- We conduct extensive experiments on real-world datasets which demonstrates both the effectiveness and efficiency of our proposed model GCNSI.

The rest of this paper is organized as follows: Section 2 surveys the related work. Section 3 introduces the definition of MSRD and describes relevant techniques of our work. Section 4 shows the architecture and details of GCNSI. Experimental results are reported in Section 5. Finally, Section 6 concludes this paper.

2 RELATED WORK

2.1 Rumor source identification

The early work on rumor source identification is studied by Shah and Zaman [28, 29]. They propose the rumor centrality theory and design a likelihood function to calculate the rumor centrality score of each node according to the topology of the infection state. However, their method is applicable only for the situation that the underlying propagation model is SI (Susceptible-Infected) model and the network is a regular tree. Zhu et al. [41] study the rumor source detection problem with a underlying model as SIR (Susceptible-Infected Removed) model. They propose a notion named *Jordan center*, which is an extension of traditional diffusion kernel with sparse observations [42]. Shen et al. [31] consider the infection time of nodes as a sequence and use time window.

The above mentioned studies are all proposed for single source identification. Meanwhile, the problem of multiple sources detection are also developed. Lappas et al. [18] propose a model to obtain top-K sources under IC (Independent Cascade) model, in which K is set manually. They formally define the k-Effectors problem, which is similar to the multiple source identification problem. Prakash et al. [26] propose a method that automatically searches the multiple sources under SI model. Their model, namely NETSLEUTH, employs the Minimum Description Length principle to produce the best set of seed nodes (similar to the information sources). Zang et al. [37] study the same multiple source identification problem in SIR model. They propose an approximate multi-source locating algorithm by introducing a reverse propagation model to detect the recovered and unobserved infected nodes. In addition, their model is designed only for SIR model.

LPSI [34] is the first study that locates the multiple sources without any information of the underlying propagation model. LPSI is a label propagation based model, which is inspired by a semi-supervised learning method [39]. By setting an original label to each node and adopting an iterated processing, LPSI finally predicts the rumor sources based on the convergent node labels.

2.2 Deep Learning on Graph Data

The idea of Graph Convolutional Networks (GCN) is inspired by traditional neural network, and it extends traditional deep learning methods for non-Euclidean data. Gori et al. [9] introduce the neural networks on graph. Scarselli et al. [27] apply recurrent neural networks on graphs. Both of their models are iterative model and they repeat the iteration until convergence. Duvenaud et al. [5] introduce convolutional-like propagation rule on graphs, which is the earliest GCN. Kipf et al. [15] optimize the model [5] by using a single weight matrix per layer and deal with varying node degrees through an appropriate normalization of the adjacency matrix, which improves the efficiency of GCN and makes it more effective in practice. Li et al. [19] conduct a deep analysis of GCN and prove that GCN outperforms label propagation based method [39] on semi-supervised learning based tasks.

3 PRELIMINARIES

In this section, we introduce the problem statement and relevant techniques that are necessary for understanding GCNSI. Frequently used notations are summarized in Table 1.

Table 1: Notation Summarization

Notation	Definition
$\mathcal{G}$	the topological graph of social network
V	set of vertexes in $\mathcal{G}$ w.r.t. nodes in social network
E	set of edges in $\mathcal{G}$
$Y = (Y_1, ..., Y_{\|V\|})^T$	infection state of social network
$Y_i \rightarrow \{1, -1\}$	infection state of a single node i
R	set of predictable rumor sources
R^*	ground-truth set of rumor sources
$Y^{'} = [d_1, d_2, d_3, d_4]$	expanded infection state
λ	weight of L2 regularization
y'	output of GCNSI
y	ground-truth vector of rumor sources

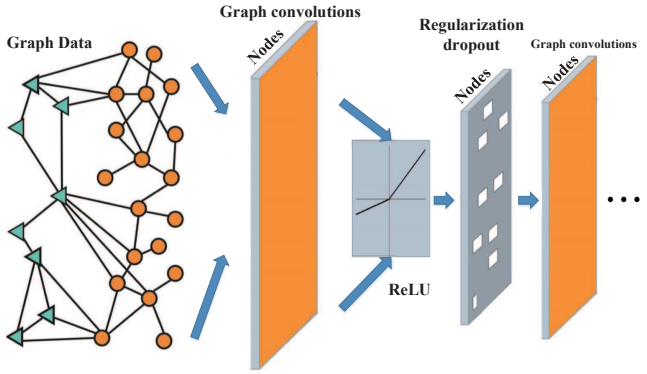


Figure 1: Overview of GCN

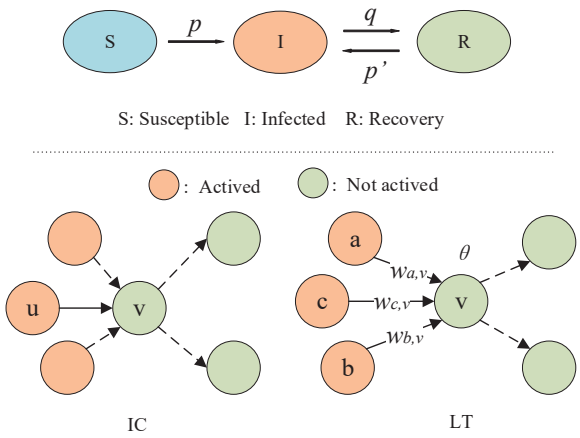


Figure 2: Infection Models and Influence Models

3.1 Problem Statement

Given an undirected social network $\mathcal{G} = (V, E, Y)$ where V is the node set, E is the edge set, and $Y = \{Y_1, ..., Y_{|V|}\}$ is an infection state of all nodes in $\mathcal{G}$, which describes that a subset of nodes in $\mathcal{G}$ have been infected. Each $Y_i \in \{1, -1\}$ denotes the infection state of node $v_i \in V$, where $Y_i = 1$ indicates that v_i is infected and otherwise $Y_i = -1$ indicates it is uninfected.

Let us denote a *ground-truth set* as $R^* \subset V$. The multiple rumor source detection (MRSD) problem is to find a label function $l : V \rightarrow \{1, 0\}$ to detect which nodes are rumor sources, such that the following Eq. 1 is maximized:

$$\frac{|R^* \cap R|}{|R^* \cup R|} \quad with \quad R = \{x \in V | \, l(x) = 1\}, \tag{1}$$

where R is the prediction results of our method.

Notably, the underlying information propagation model for generating Y is unknown in our model, which is the main contrast to the existing source identification problem.

3.2 Graph Convolutional Networks

As the traditional deep learning methods, such as convolutional neural networks (CNNs), have gained great success in grid-structured (Euclidean) data, the community turns to generalizing deep learning techniques for non-Euclidean data. Graph Convolutional Network (GCN) [4] is an extension of CNN on graph data, which generates local permutation-invariant aggregation on the neighborhood of a node in a graph such that the features of a graph can be efficiently captured.

However, the early GCN is time consuming in practice. To accelerate the training process of GCN, the approximate calculation with Chebyshev polynomial [11] is introduced in multi-layer GCN. The structure of multi-layer GCN is shown in Fig. 1. In each layer of multi-layer GCN, the layer-wise propagation rule is defined as follows :

$$H^{-1} = \sigma(\widetilde{D}^{-\frac{1}{2}}\widetilde{A}\widetilde{D}^{-\frac{1}{2}}H^{(l)}W^{(l)}), \tag{2}$$

where I is the identity matrix, $\widetilde{A} = A + I$ is the adjacency matrix of $\mathcal{G}$, and $\widetilde{D}_{ii} = \sum_j \widetilde{A}_{ij}$ represents the Laplacian matrix of $\mathcal{G}$. $W^{(l)}$ is a layer-specific trainable weight matrix. $\sigma(\cdot)$ is the activation function. $H^{(l)} \in \mathbb{R}^{N*D}$ is the matrix of hidden state in the l-th layer; $H^{(0)} = X$ is the input of the first layer.

From Eq. 2 we can see, only the first-order neighbors of each node are considered. In order to capture the features of multi-order neighbors, we stack a corresponding number of layers based on the neighborhood order. As we mainly focus on the application of GCN in this paper, interested readers may refer to [15] for more theoretical details.

As GCN is originally designed for semi-supervised learning, although it is able to effectively capture and properly express the features of a graph, it cannot be directly applied for the supervised leaning based source detection problem. In this paper, we modify GCN and apply it in our graph feature capturing task. The details can be found in Section 4.3.

3.3 Propagation Models

As described in [6], the propagation models can be divided into two categories, i.e., infection models and influence models. The infection model is originally proposed to describe disease infecting among individuals. There are three widely used infection models: SI (Susceptible-Infected) model [2], SIS (Susceptible-Infected-Susceptible) model and SIR (Susceptible-Infected-Recovery) model [1]. Generally, individuals in infection model have three states: *Suspected*, *Infected* and *Recovery*. As shown in Fig. 2, in SI model, a suspected individual has probability p to be infected. In SIR model, an infected individual has probability q to recovery and will never be infected again. As contrast, in SIS model, a recovered individual can be infected again with probability p'. Due to the high similarity of the propagation modes between disease and rumor, the infection models are widely adopted for describing rumor propagation in social networks [13] [33].

On the other hand, IC (Independent Cascade) model [8] and LT (Linear Threshold) model [10] are two widely-adopted influence models. IC model is an iterative model, where in each iteration, each node has a chance to active its neighbors. As shown in Fig. 2, if a neighbor v is not activated by the node u successfully, it will never be activated by u again, however, v can still be activated by other neighbors. In IC model, this iteration is repeated until the model converges. In LT model, each node is initialized with

a threshold θ, which describes the probability of the node to be infected. Each edge is initialized with a weight w. As shown in Fig. 2, in each iteration, for a node v, the total influence of its neighbors (a, b, c) will be summarized with $v = w_{a,v} \times a + w_{b,v} \times b + w_{c,v} \times c$. If $v > \theta$ the node v will be activated. The iteration is repeated until the convergence is reached.

As introduced in [12], all of above propagation models are designed for generating infection states of networks. Given a set of nodes R^* as sources, the infection state Y of $\mathcal{G}$ will be generated. In this work, we use these models to generate both the ground truth and the training data for the experiment, and the details can be found in Section 4.2.

3.4 Baseline: Label Propagation based Source Identification

Wang et al. [34] propose Label Propagation based Source Identification (LPSI), which is inspired by a label propagation based semi-supervised learning method [39]. As LPSI studies the same problem with our work, we consider it as a baseline method for performance comparison.

In LPSI, the iteration formulation (Eq. 3) and convergent state (Eq. 4) are defined as follows:

$$\mathcal{G}_i^{t+1} = \alpha \sum_{j:j\in\mathcal{N}(i)} S_{ij}\mathcal{G}_j^t + (1-\alpha)Y_i, \tag{3}$$

$$\mathcal{G}^* = (1-\alpha)(I-\alpha S)^{-1}Y. \tag{4}$$

In Eq. 3, $\alpha \in (0, 1)$ is the parameter used to control the influence that node i gets from its neighbors. $\mathcal{N}(i)$ represents the set of neighbors of node i. $\mathcal{G}_i^t$ is the infection state of node i at iteration t. S_{ij} is the (i, j)-th element of the regularized Laplace matrix S of $\mathcal{G}$. Y_i is the given infection state of node i.

It is proved that the iteration will finally converge and the convergent state formulation is described in Eq. 4, where $\mathcal{G}^*$ is the convergence state of the network. I is the identity matrix. $S = D^{-1/2}WD^{-1/2}$ is the regularized Laplace matrix of $\mathcal{G}$, where D is a diagonal matrix with its (i,i)-th element equal to the sum of the i-th row of W, and W is the adjacency matrix of graph $\mathcal{G}$.

The output of LPSI is a label for each node. The node label is simply an integer, and it indicates the probability of a node being an infection source. In other words, a larger value indicates that the node is more likely to be a rumor source.

However, the information contained in the node label is insufficient to express the complicated structural information. Therefore, in this work, we build a multi-dimensional vector instead of an integer for each node. In addition, as LPSI is able to capture the feature of rumor centrality, we adopt LPSI for generating inputs of GCNSI and the details can be found in Section 4.2.

4 GRAPH CONVOLUTIONAL NETWORKS BASED SOURCE IDENTIFICATION

4.1 Architecture of GCNSI

The input of multiple rumor source detection (MRSD) is a number of different infection states of a given network, and the output is the actual rumor sources. According to the nature of MRSD, it can be considered as a variance of the multi-label classification problem that aims to assign a two-valued label $(0, 1)$ to each node in $\mathcal{G}$.

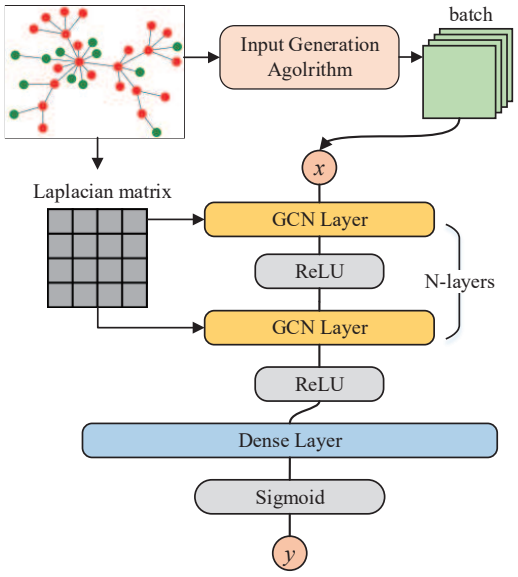


Figure 3: Architecture of GCNSI

Our model proposed for solving MRSD is mainly based on two assumptions:

(1) **Source prominence.** The nodes surrounded by a larger proportion of infected nodes are more likely to be rumor sources.
(2) **Rumor centrality** [30]. The nodes far from the rumor source are less likely to be infected than those near the rumor source.

By taking these two assumptions into consideration, we believe that designing a method to represent nodes with the information of its neighbors will effectively promote the precision of the source detection task.

As GCN can effectively capture the features of vertex domain and spectral domain for a graph, it is therefore proper to adopt GCN for expressing complicated neighborhood information. As mentioned before, GCN uses first-order Chebyshev polynomial for approximate calculation, and a single GCN layer only captures information from the first-order neighbors in a graph. In order to capture the features of multi-order neighbor nodes, we stack multiple GCN layers, and the particular number of the layers is chosen corresponding to the orders of neighbors, which varies for different dataset and will be introduced later. In addition, since the output of GCN layer is a matrix, we adopt a dense (full connected) layer to transform the output matrix into a vector.

The architecture of our model GCNSI is shown in Fig. 3. First, the Laplacian matrix is generated from $\mathcal{G}$. Next, a set of training samples are generated by Algorithm 1. Then, each training sample x flows via multiple GCN layers and a single dense layer. The active functions are ReLU for GCN layer and Sigmoid for dense layer, respectively. Finally, the prediction output of GCNSI is y.

Algorithm 1: Input Generation Algorithm

Input: The infected network $\mathcal{G} = (V, E, Y)$ where $Y = (Y_1, ..., Y_{|v|})^T$; parameter α;

Output: Infection state matrix $Y^{'} = [d_1, d_2, d_3, d_4]$

1 Form the weight matrix W defined by $W_{ij} = 1$ if there exists an edge connecting nodes i and j;
2 Construct the matrix $S = D^{-1/2}WD^{-1/2}$, where D is a diagonal matrix with its (i,i)-element equal to the sum of the i-th row of W;
3 Initialize the $V_3 = Y$, $V_4 = Y$;
4 **for** $i < len(Y)$ **do**
5 **if** Y_i == -1 **then**
6 $V_{3i} = 0$;
7 **else**
8 $V_{4i} = 0$;
9 **end**
10 **end**
11 $d_1 = Y$;
12 $d_2 = (1-\alpha)(I - \alpha S)^{-1}Y$;
13 $d_3 = (1-\alpha)(I - \alpha S)^{-1}V_3$;
14 $d_4 = (1-\alpha)(I - \alpha S)^{-1}V_4$;
15 $Y^{'} = concatenate(d_1, d_2, d_3, d_4)$

4.2 Input Generation

The existing work such as LPSI treats the original infection state Y as labels of nodes. However, in our point of view, the elements of Y are simply integers, which is not informative enough to express the complicated connection structure between nodes. Therefore, we propose an input generation algorithm to expand the integer label into a multi-dimensional vector for each node.

The details are introduced in Algorithm 1. The input is $\mathcal{G} = (V, E, Y)$ with an infection state vector $Y = (Y_1, ..., Y_{|V|})^T$. The output is $Y^{'} = [d_1, d_2, d_3, d_4]$, which is a matrix generated by superimposing the vectors of $d_1 \sim d_4$. In Y', the four dimensions in row i can be treated as different features of node v_i. The construction of the regularization Laplacian matrix is introduced in Step 1 and Step 2 of Algorithm 1. From Step 3 to 10, vectors V_3 and V_4 are defined and initialized, which are used for generating d_3 and d_4. The remaining part of Algorithm 1 shows the procedure of generating $d_1 \sim d_4$.

Although deep learning method is able to learn the different weights of infected and uninfected nodes. To obtain converged results as soon as possible, we intuitively select some suitable features, i.e., $d_1 \sim d_4$. We denote d_1 as the infection state value Y where the infected nodes are set as 1 and the uninfected nodes are set as -1. As stated in [34], the feature of *source prominence* can be captured by LPSI. Since in each iteration of LPSI, the label of a node is updated based on its neighbors' labels according to Eq. 3, we consider d_2 as the feature of *source prominence* which is the output of LPSI with Y as input.

In order to capture the feature of *rumor centrality*, we further build d_3 and d_4 which are the outputs of LPSI with modified inputs. As introduced in Step 4 ~ 10, the input of LPSI w.r.t. d_3 is a vector generated by changing all -1 to 0 in Y, and the input w.r.t. d_4 is a vector generated by changing all 1 to 0 in Y. To some extent, d_2 also captures the feature of *rumor centrality*. However, during the iteration process in LPSI, both the influences of infected nodes and uninfected nodes spread and resist each other until convergence. In other words, the propagation of infected nodes is counteracted by that of uninfected nodes. To remedy this issue, we remove the influence of one type of nodes by setting the label as 0. In this way, the other type of nodes can propagate its influence as far as possible in $\mathcal{G}$. As a result, feature d_3 only concerns the infected nodes while feature d_4 only concerns the uninfected nodes. Essentially, d_3 and d_4 are extensions of d_1 in high dimensional domain.

4.3 Modified GCN layer

The traditional GCN layer [15] is designed for semi-supervised learning, which is not suitable for the supervised learning problem in this paper. To make use of it, we modify it to fit the input and output data of MRSD.

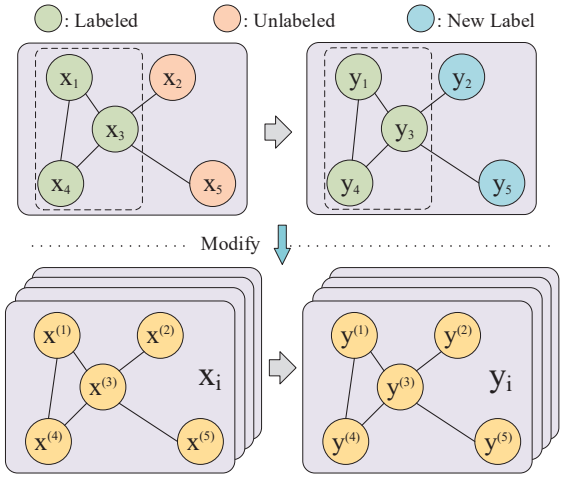


Figure 4: Traditional and modified GCN layer
The upper half represents semi-supervised learning based GCN layer while the lower half represents the supervised learning based GCN layer.

In the existing semi-supervised based GCN tasks, the training set is a graph $\mathcal{G}$, and each node in $\mathcal{G}$ is an entry for an input sample. At the beginning of learning phase, most of the nodes are assigned with labels and a small portion of nodes are unlabeled. The training data (labeled) and the test data (unlabeled) are processed at the same time. As shown in Fig. 4, nodes X_1, X_3, X_4 are labeled samples and X_2, X_5 are unlabeled samples. In the training process, each unlabeled sample is assigned with a new label according to its neighbors. For the semi-supervised based GCN, only the loss of labeled nodes are computed by the loss function. Finally, when the training finished, all of the unlabeled nodes output new labels for prediction or evaluation.

However, MRSD is a supervised learning based task. Therefore, there are two main differences between the traditional and the modified GCN layer, i.e., input and loss. In GCNSI, the input is generated

by Algorithm 1, denoted as $x_i = Y'$, which is a complete infection state of network. As shown in Fig. 4, each node in $\mathcal{G}$ receives a component of x_i. When a batch of samples is acquired, GCNSI calculates the batch-sized loss between inputs and outputs to update the weight-matrix. After all epochs of data are trained and the model reaches convergence, the GCN layer can make predictions for test samples.

4.4 Loss Function

We denote a training sample for GCNSI as (x, y), where the input is x and the output is y. For a given input $x = Y$, the corresponding output y is the actual network state that described rumor sources. As mentioned before, MRSD is variance of the multi-label classification problem and we need to predict whether each node is a rumor source at the same time. Therefore, we adopt a sigmiod cross-entropy loss as the loss function. In addition, we use L2 regularization in the loss function to reduce overfitting. The loss function is described as follows:

$$L(y', y) = -\log \sigma(y') \times y - \log(1 - \sigma(y')) \times (1 - y) + \lambda ||w||_2, \quad (5)$$

where y' is the output of GCNSI with the input x, y is the true label and σ is the Sigmoid function. In addition, w represents all of the weights in GCNSI and $||w||_2$ is the L2 regularization item with a weight coefficient as λ.

5 EXPERIMENT

In this section, we report the experimental results of our proposed method on real-world datasets.

5.1 Datasets and Baselines

As shown in Table 2, we choose five datasets[1] for the evaluation on the performance of our method:

- **Karate** [36] is a social network of friendships between 34 members of a karate club at an US university in the 1970s. It's a classic dataset that is widely used in many work for fast verification.
- **Dolphin** [23] is an undirected social network of frequent associations between 62 dolphins in a community living off Doubtful Sound, New Zealand.
- **Jazz** [7] is a network of Jazz bands performing from 1912 to 1940.
- **US Power grid** [35] is an undirected unweighted network representing the topology of the Western States Power Grid of the United States.
- **Ego-Facebook** contains social circles from Facebook.

The following methods are our model and baselines to be compared in the experiment:

- **GCNSI**: Graph Convolutional Networks based source identification model, which is our proposed model in this paper.
- **LPSI** [34]: Label propagation based source identification model, which is state-of-the-art model for the MRSD.
- **NetSleuth** [26]: A method for multiple sources detection under SI model.

[1]All of these datasets can be acquired from public websites http://www-personal.umich.edu/~mejn/netdata/ and http://snap.stanford.edu/.

Table 2: Datasets

Name	#Nodes	#Edges	# Avg(degree)
Karate	34	78	4.6
Dolphin	62	159	4.97
Jazz	198	2742	27.7
Power grid	4941	6594	1.33
Ego-Facebook	4039	88234	43.7

- **Zang** [37]: A model for locating multiple sources under SIR model.

5.2 Settings and Optimization

In this section, we first introduce some settings of our experiment and then some optimization measures of GCNSI. For underlying propagation model, we choose two infection models and an influence model as the underlying propagation model to test the effectiveness of GCNSI and baselines. Considering that the SIS model is similar to SI and SIR model, we only use SI and SIR model for comparison. For influence model, we choose IC (Independent Cascade) model.

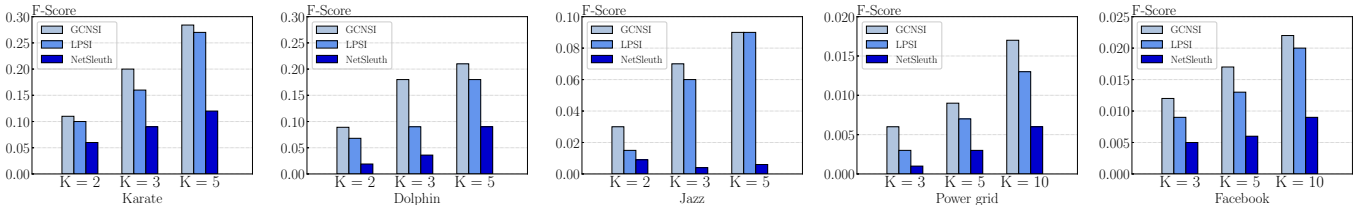
As shown in Table 3, GCNSI and LPSI are tested under all the propagation models. Since NetSleuth is specially designed for SI model and Zang is for SIR model, NetSleuth is only tested under SI model and Zang is only tested under SIR model. As shown in many existing work [34, 42], the infection probability p in SI and SIR is sampled from uniform distribution $U(0, 1)$ and the recovery probability q in SIR is sampled from uniform distribution $U(0, p)$. In the IC model,the infection probability p' is chosen from uniform distribution $U(0, 1)$.

Table 3: Testing Methods under Underlying Models

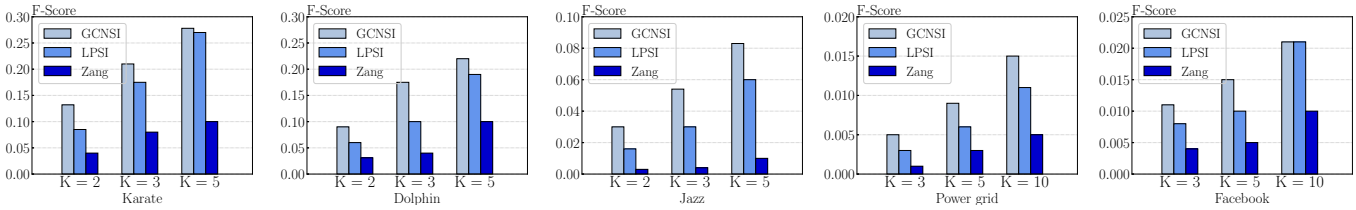
Methods	SI	SIR	IC
GCNSI	✓	✓	✓
LPSI	✓	✓	✓
NetSleuth	✓		
Zang		✓	

The parameter α in LPSI is set as 0.5, and we choose the convergent state (see in Eq. 4) to obtain the best performance (proved in [34]). All the introduced results are under over 500 independent runs to ensure the credibility (as the same in [26, 34]). What's more, for each situation (one dataset with one underlying propagation model), we set different numbers of infection sources. For Karate, Jazz and Dolphin, the source numbers K are 2, 3, 5 and for power grid and Facebook dataset with a larger scale, the source numbers K are set as 5, 8, 10. In addition, we restrict the lowest infection rate of $\mathcal{G}$ as 30% (as the same in [34]) and this setting is the same for all datasets.

By following the strategy introduced in Section 4.2, we produce training, validation and test data for GCNSI. In addition, the training samples are generated from different underlying propagation models with different numbers of sources. For each dataset, we mix training samples generated under SI, SIR and IC models with the



(a) Results under SI



(b) Results under SIR

Figure 5: Results under Infection Model

proportion of 1 : 1 : 1. For each propagation model, the proportion of training samples generated from different source numbers is also the same. In addition, all samples are shuffled before training. As for the test data, it is distinguished by different underlying models and different source numbers to obtain different evaluation results.

Since GCNSI is a deep learning based model, we adopt some optimization measures to promote the performance. As introduced above, L2 regularization and dropout are adopted in every GCN layer to avoid overfitting, and the dropout rate is different for each dataset. The activation function between GCN layers is ReLU, which helps to avoid gradient vanishing problem. Furthermore, our model is trained through back-propagation algorithm and we choose the Adam optimization algorithm [14] to achieve it. We iterate the whole training procedure until the value of loss function converges.

Evaluation Metrics. We use two metrics to evaluate the performance of our proposed model:

- **F-score** is a suitable metric for measuring multi-labeled classification problems, since it is a combination of precision and recall. Therefore, by following [34], we choose F-score as one of our evaluation metrics. The β of F-socre is set as 1 (means the precision and recall are treated equally). It can be proved that the Eq.1 is proportional to F1-score, thus, maximizing F1-score is equivalent to maximizing Eq.1.
- **Error distance** is adopted by some existing work [12] on source identification problem as the evaluation metric. Even if stated in [34], the results with small error distance are usually the nodes nearby true sources rather than the true sources. To adequately display the effectiveness of GCNSI, we still demonstrate the results (See in Table 5) under SI model.

The definition of error distance is following:

$$\Delta = \frac{1}{N_{Src}} \left(\sum_{i \in Src} \min_{j \in Gt} dist(i,j) + \eta |N_{Src} - N_{Gt}| \right) \quad (6)$$

where Src is the set of the ground-truth sources and Gt is the set of predicted sources. N_{Src} and N_{Src} is number of nodes in Src and Gt. The shortest path from node i to node j is denoted as $dist(i,j)$, and weight η is set as 0.5. In addition, N_{Gt} is set to be the same as N_{Src} in this paper.

Table 4: Default settings of important hyper-parameters

Dataset	learning rate	GCN layers	hidden size	dropout rate	batch size	epoch
Karate	0.001	5	512	0.1	1000	900
Dolphin	0.002	6	512	0.1	1000	900
Jazz	0.001	6	512	0.1	1000	1000
Power grid	0.005	7	1024	0.1	1500	2000
Ego-Facebook	0.005	10	1024	0.3	1500	4000

5.3 Self Evaluations

GCNSI is implemented with Tensorflow[2], and all the values of hyper-parameters are set using cross-validation. Some important validation processes of the key hyper-parameters are shown in Section 5.5, such as the learning rate, number of GCN layers, hidden unit size and the rate of dropout. Meanwhile, the validation processes of the other hyper-parameters such as batch size and epoch are omitted for the sake of brevity. Default settings of hyper-parameters are shown in Table 4.

5.4 Results

Fig. 5(a) shows the results under SI model. As we can see, GCNSI significantly outperforms LPSI and NetSleuth for all datasets. For instance, in Karate dataset, GCNSI achieves the best performance among all baseline methods, no matter how K is set. It can be seen that GCNSI outperforms LPSI by almost 10% relatively on average. LPSI is the second best result, which nearly outperforms NetSleuth

[2]https://www.tensorflow.org/

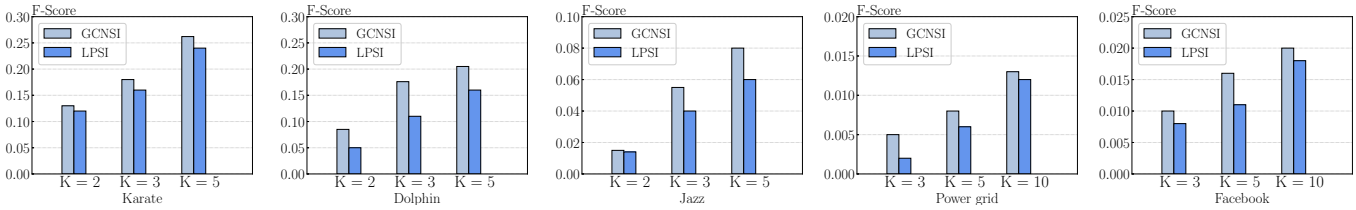
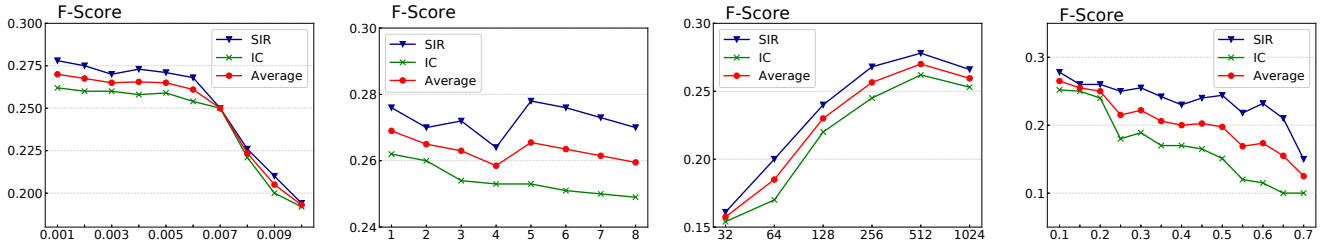


Figure 6: Results under Influence Model



(a) Learning rate (b) GCN layers (c) Hidden unit size (d) Dropout rate

Figure 7: Impact of parameters

Table 5: Error Distance under SI model

	K=5			K=10	
	Karate	Dolphin	Jazz	Power grid	Facebook
GCNSI	**0.63**	**0.82**	**0.71**	**2.61**	**0.61**
LPSI	0.65	0.87	0.74	2.67	0.64
NetSleuth	0.92	1.02	1.2	3.15	0.97
	K=3			K=8	
	Karate	Dolphin	Jazz	Power grid	Facebook
GCNSI	**1.08**	**1.49**	**1.29**	**3.16**	**0.7**
LPSI	1.13	1.51	1.33	3.18	0.74
NetSleuth	1.36	1.86	1.52	4.74	1.22
	K=2			K=5	
	Karate	Dolphin	Jazz	Power grid	Facebook
GCNSI	**1.48**	**2.28**	**1.87**	**5.33**	**1.17**
LPSI	1.5	2.35	1.89	5.54	1.26
NetSleuth	1.65	2.51	2.03	6.86	1.53

by 50%. The result in Dolphin dataset demonstrates that GCNSI outperforms LPSI and NetSleuth for all the values of K especially when K is 3. For the situations that $K = 2$ and $K = 5$, GCNSI outperforms LPSI by nearly 30% on average. As for Jazz dataset, GCNSI outperforms LPSI by almost 13% on average. It is shown in results w.r.t Power grid and Facebook, when the network size increases, performance drops for all methods. The results reveal that GCNSI outperforms LPSI and NetSleuth in both Power grid and Ego-Facebook. Notably, GCNSI still achieves the best performance on these two datasets with larger scales. Generally speaking, we have evidenced that GCNSI achieves the best performance than any other baselines on under SI model.

Next, we explore the results under SIR model. In Fig. 5(b), it shows that GCNSI outperforms the other baselines under SIR model on different datasets no matter how K is set. Meanwhile, LPSI obtains the second best performance and NetSleuth gets the worst performance. In datasets with small scales, such as Karate, Dolphins and Jazz, GCNSI and LPSI substantially outperform Zang. As stated in [37], Zang tends to locate the nodes nearby sources rather than real sources, which leads to bad performance. It is shown that GCNSI outperforms LPSI and NetSleuth in Power grid, meanwhile GCNSI and NetSleuth nearly get a same performance in Ego-Facebook dataset.

All of the results are consistent with the notion that GCNSI obtains the best performance among these baselines under infection model including SI model and SIR model. The results demonstrate the effectiveness of GCNSI under infection model. Furthermore, the results reveal that the methods aiming to locate the real sources (GCNSI and LPSI) outperform those aiming to return the most likely source (NetSleuth and Zang) under infection model. In addition, the similar performance of GCNSI under SI and SIR shows its effectiveness on different infection models.

To demonstrate the effectiveness of GCNSI under influence model, we further conduct experiment under IC model. As NetSleuth is designed for SI model and Zang is designed for SIR model, we only test GCNSI and LPSI for IC model. We examine the performance of these methods under IC model and Fig. 6 shows the results. Overall, we observe that the GCNSI still achieves the best performance with varied values on different datasets no matter how K is set. LPSI is the second best result under IC model. Notably, it reveals that the superiority of GCNSI drops compared to those under infection model. In our point of view, compared to infection models, the source prominence and rumor centrality are not so significant under IC model. Maybe it's the main reason for

this phenomenon. Conclusively, the results under IC model further demonstrate the effectiveness of GCNSI under various information propagation models.

Finally, we calculate the error distance under SI model and the results are shown in Table 5. Since error distance measures the average shortest path between predicted rumor sources and true rumor sources, the smaller error distance indicates the better performance. We test GCNSI, LPSI and NetSleuth under SI model on all the five datasets, and the settings of K are same as those introduced previously. As we can see, GCNSI gains the best performance in each instance while LPSI achieves the second best performance. Notably, the average performance on Ego-Facebook is better than that on US Power grid, and we think the reason is that Power grid is a sparse graph while Ego-Facebook is an extreme dense graph.

5.5 Impact of Parameters

Lastly, the impact of some important parameters in GCNSI are examined and the validation results are shown in Fig. 7. Four parameters are chosen to be validated, i.e., number of learning rate, GCN layers, hidden unit size, dropout rate. Since the validation of all datasets is massive, we only demonstrate the validation on Karate dataset as a case study. For each parameter, an infection model (SIR) and an influence model (IC) are chosen as the underlying propagation models. Actually, after the cross-validation, we finally obtain the best settings of these four parameters. Therefore, when we adjust a parameter, the other three are fixed with the best settings, and the impact of each parameter is introduced separately in the rest of this section.

Learning rate is an important hyper-parameter in deep learning, which controls the speed to convergence of the training procedure. As shown in Fig. 7(a), the performance of GCNSI drops when the learning rate increases from 0.001 to 0.01, and it achieves the best performance with learning rate as 0.001 on average. The result reveals that it is important to select a proper initial value for it (The optimizer is Adam, which adjusts learning rate during training). What's more, it shows that the smaller initial learning rate promotes the performance on Karate.

The number of GCN layers decides the orders of neighborhoods that a node can reach in $\mathcal{G}$, which has a great impact on the performance of the model. We vary the number of GCN layers from 1 to 8 and record the corresponding F-Score of GCNSI under SIR and IC model. The result is shown in Fig. 7(b), and it reveals that when the number of GCN layers is set as 1 and 5, the GCNSI achieves the best performance on average. Moreover, when the number of GCN layers is set as 4, it achieves the worst performance, and the performance is also bad when the number of GCN layers is 8. We think there are two reasons for that: (1) When the number of GCN layers is larger than 5, the network structure is too deep to train on the Karate dataset with a small scale. (2) Too many GCN layers lead to a huge number of parameters, which usually causes overfitting.

Then, we discuss the impact of the hidden unit size. By varying the values of hidden units from 32 to 1024, we finally get the corresponding results shown in Fig. 7(c). It can be seen, the results show that GCNSI achieves the best performance when the hidden unit size is 512 and gets the worst performance when the hidden unit size is 32. The input of GCNSI uses 4-dimensional vector to embed each node, which is expanded into a new vector with the size of hidden unit. Furthermore, if the size of hidden unit is too small, it will not represent the features well and if the size of hidden unit is too large, it leads to overfitting. All reasons above may explain the result w.r.t hidden unit size.

Finally, we explain the impact of the dropout rate. In order to avoid overfitting, this parameter is adopted in each layer of GCNSI. The dropout rate controls the rate of randomly abandoning information in GCN layers. As we can see in Fig. 7(d), the result suggests that the GCNSI achieves the best performance when the dropout rate is 0.1, which is the most appropriate rate for the Karate dataset. Moreover, it shows that the Karate dataset is less affected by overfitting.

5.6 Scalability

As GCNSI is a deep learning based model, it spends time to train and tune parameters. After the training process finished, the prediction procedure consumes only a little time for a single sample. According to Eq. 2, it can be easily proved that the overall complexity of GCNSI prediction procedure is $O(k \times N^3)$, where k is the number of layers in GCNSI. In contrast, as proved in [34], the overall complexity of LPSI is $O(N^3)$.

The experiment results demonstrate the time complexity above, it takes 0.007 seconds on average to make a prediction on Karate under SI model with a 3-layers GCNSI. With the same conditions, LPSI consumes 0.008 seconds on average. In our point of view, the time consuming of the GCNSI prediction procedure is acceptable in practice.

5.7 Discussion

In Section 5.4, we show that GCNSI achieves the best performance among all baselines. In most experiment results, we can see GCNSI always outperforms other baselines no matter how K is set. Then, considering the comparison with LPSI, we believe that an integer feature is too simple to represent a node in graph and representing node with a vector promotes the performance effectively for MRSD. If we make a comparison among the results under different underlying models, we figure out that the performance of GCNSI almost does not change. It reveals that if a dataset and topological graph (network structure) are fixed, the performance of GCNSI is stable. We think it is because the rule that GCNSI has learnt from dataset is a common attribute of information propagation models. Once the topological graph is given, the rumor infection states under different propagation models have similar distributions, which also supports the assumptions (such as *source prominence* and *rumor centrality*). Moreover, if we compare the results under different datasets, for instance, in Fig. 5, we find that the performance of all methods drops when the scale of dataset increases. This phenomenon is well understood that the larger quantity of nodes brings more difficulty for all methods. Nevertheless, we can see that the GCNSI is proved with the less reduction than other methods. It shows the potential of GCNSI on larger networks.

Although GCNSI achieves good performance in different types of experiments, there are also limitations. Essentially, GCNSI is a deep learning based model, that is to say, it'is necessary to retrain

the model when the topological graph changes. Moreover, most of the parameters need fine adjustment, and it takes time.

6 CONCLUSION AND FUTURE WORK

As source detection is an important task in defeating rumors in social network, in this paper, we study the multiple rumor source detection (MRSD) problem. To solve MRSD with high precision, we propose a deep learning based model GCNSI to locate multiple rumor sources without prior knowledge of underlying propagation model. Furthermore, an input generation algorithm is developed to extend the integer label into vector for node representation. Then, we conduct experiment by comparing our model with the state-of-the-art baselines on five real-world networks, and the results demonstrate the effectiveness of GCNSI where it outperforms the state-of-the-art method by about 15%. To our knowledge, GCNSI is the first model adopting GCN in MRSD problem and we hope it may bring some inspirations. In the future, we plan to apply GCN into other rumor defeating problems such as content-based rumor identification etc. Due to the complex features of rumor content (including text features and users' behaviors), we believe it is of great potential for applying GCN.

ACKNOWLEDGMENTS

This research is partially supported by the Fundamental Research Funds for the Central Universities (HUST: Grants No. 2019kfyXKJC021, 2019kfyXJJS091), and NSFC (Grants No. 61902134, 61572215).

REFERENCES

[1] Linda JS Allen. 1994. Some discrete-time SI, SIR, and SIS epidemic models. *Mathematical biosciences* 124, 1 (1994), 83–105.
[2] Roy M Anderson and Robert M May. 1992. *Infectious diseases of humans: dynamics and control*. Oxford university press.
[3] Tong Chen, Xue Li, Hongzhi Yin, and Jun Zhang. 2018. Call Attention to Rumors: Deep Attention Based Recurrent Neural Networks for Early Rumor Detection. In *PAKDD (Workshops) (Lecture Notes in Computer Science)*, Vol. 11154. Springer, 40–52.
[4] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. 2016. Convolutional Neural Networks on Graphs with Fast Localized Spectral Filtering. In *NIPS*. 3837–3845.
[5] David K. Duvenaud, Dougal Maclaurin, Jorge Aguilera-Iparraguirre, Rafael Gómez-Bombarelli, Timothy Hirzel, Alán Aspuru-Guzik, and Ryan P. Adams. 2015. Convolutional Networks on Graphs for Learning Molecular Fingerprints. In *NIPS*. 2224–2232.
[6] David A. Easley and Jon M. Kleinberg. 2010. *Networks, Crowds, and Markets - Reasoning About a Highly Connected World*. Cambridge University Press.
[7] Pablo M. Gleiser and Leon Danon. 2003. Community Structure in Jazz. *Advances in Complex Systems* 6, 4 (2003), 565–574.
[8] Jacob Goldenberg, Barak Libai, and Eitan Muller. 2001. Talk of the network: A complex systems look at the underlying process of word-of-mouth. *Marketing letters* 12, 3 (2001), 211–223.
[9] Marco Gori, Gabriele Monfardini, and Franco Scarselli. 2005. A new model for learning in graph domains. In *IJCNN*, Vol. 2. IEEE, 729–734.
[10] Mark Granovetter. 1978. Threshold models of collective behavior. *American journal of sociology* 83, 6 (1978), 1420–1443.
[11] David K. Hammond, Pierre Vandergheynst, and Rémi Gribonval. 2009. Wavelets on Graphs via Spectral Graph Theory. *CoRR* abs/0912.3848 (2009).
[12] Jiao Jiao Jiang, Sheng Wen, Shui Yu, Yang Xiang, and Wanlei Zhou. 2017. Identifying Propagation Sources in Networks: State-of-the-Art and Comparative Studies. *IEEE Communications Surveys and Tutorials* 19, 1 (2017), 465–481.
[13] Nikhil Karamchandani and Massimo Franceschetti. 2013. Rumor source detection under probabilistic sampling. In *ISIT*. IEEE, 2184–2188.
[14] Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *ICLR*.
[15] Thomas N. Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *ICLR*.
[16] Jan Kostka, Yvonne Anne Oswald, and Roger Wattenhofer. 2008. Word of Mouth: Rumor Dissemination in Social Networks. In *SIROCCO (Lecture Notes in Computer Science)*, Vol. 5058. Springer, 185–196.
[17] David Burth Kurka, Alan Godoy, and Fernando J. Von Zuben. 2015. Online Social Network Analysis: A Survey of Research Applications in Computer Science. *CoRR* abs/1504.05655 (2015).
[18] Theodoros Lappas, Evimaria Terzi, Dimitrios Gunopulos, and Heikki Mannila. 2010. Finding effectors in social networks. In *KDD*. ACM, 1059–1068.
[19] Qimai Li, Zhichao Han, and Xiao-Ming Wu. 2018. Deeper Insights Into Graph Convolutional Networks for Semi-Supervised Learning. In *AAAI*. AAAI Press, 3538–3545.
[20] Guanfeng Liu, Yan Wang, and Mehmet A. Orgun. 2010. Optimal Social Trust Path Selection in Complex Social Networks. In *AAAI*.
[21] Guanfeng Liu, Yan Wang, Mehmet A. Orgun, and Ee-Peng Lim. 2013. Finding the Optimal Social Trust Path for the Selection of Trustworthy Service Providers in Complex Social Networks. *IEEE Trans. Services Computing* 6, 2 (2013), 152–167.
[22] Yang Liu and Yi-fang Brook Wu. 2018. Early Detection of Fake News on Social Media Through Propagation Path Classification with Recurrent and Convolutional Networks. In *AAAI*. AAAI Press, 354–361.
[23] David Lusseau, Karsten Schneider, Oliver J Boisseau, Patti Haase, Elisabeth Slooten, and Steve M Dawson. 2003. The bottlenose dolphin community of Doubtful Sound features a large proportion of long-lasting associations. *Behavioral Ecology and Sociobiology* 54, 4 (2003), 396–405.
[24] Jing Ma, Wei Gao, Prasenjit Mitra, Sejeong Kwon, Bernard J. Jansen, Kam-Fai Wong, and Meeyoung Cha. 2016. Detecting Rumors from Microblogs with Recurrent Neural Networks. In *IJCAI*. IJCAI/AAAI Press, 3818–3824.
[25] Federico Monti, Fabrizio Frasca, Davide Eynard, Damon Mannion, and Michael M. Bronstein. 2019. Fake News Detection on Social Media using Geometric Deep Learning. *CoRR* abs/1902.06673 (2019).
[26] B. Aditya Prakash, Jilles Vreeken, and Christos Faloutsos. 2012. Spotting Culprits in Epidemics: How Many and Which Ones?. In *ICDM*. IEEE Computer Society, 11–20.
[27] Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. 2009. The Graph Neural Network Model. *IEEE Trans. Neural Networks* 20, 1 (2009), 61–80.
[28] Devavrat Shah and Tauhid Zaman. 2010. Detecting sources of computer viruses in networks: theory and experiment. In *SIGMETRICS*. ACM, 203–214.
[29] Devavrat Shah and Tauhid Zaman. 2011. Rumors in a Network: Who's the Culprit? *IEEE Trans. Information Theory* 57, 8 (2011), 5163–5181.
[30] Devavrat Shah and Tauhid Zaman. 2012. Rumor centrality: a universal source detector. In *SIGMETRICS*. ACM, 199–210.
[31] Zhesi Shen, Shinan Cao, Wen-Xu Wang, Zengru Di, and H Eugene Stanley. 2016. Locating the source of diffusion in complex networks by time-reversal backward spreading. *Physical Review E* 93, 3 (2016), 032301.
[32] Nguyen Thanh Tam, Matthias Weidlich, Bolong Zheng, Hongzhi Yin, Nguyen Quoc Viet Hung, and Bela Stantic. 2019. From Anomaly Detection to Rumour Detection using Data Streams of Social Platforms. In *PVLDB*.
[33] Zhaoxu Wang, Wenxiang Dong, Wenyi Zhang, and Chee Wei Tan. 2014. Rumor source detection with multiple observations: fundamental limits and algorithms. In *SIGMETRICS*. ACM, 1–13.
[34] Zheng Wang, Chaokun Wang, Jisheng Pei, and Xiaojun Ye. 2017. Multiple Source Detection without Knowing the Underlying Propagation Model. In *AAAI*. AAAI Press, 217–223.
[35] Duncan J Watts and Steven H Strogatz. 1998. Collective dynamics of 'small-world'networks. *nature* 393, 6684 (1998), 440.
[36] Wayne W Zachary. 1977. An information flow model for conflict and fission in small groups. *Journal of anthropological research* 33, 4 (1977), 452–473.
[37] Wenyu Zang, Peng Zhang, Chuan Zhou, and Li Guo. 2015. Locating multiple sources in social networks under the SIR model: A divide-and-conquer approach. *J. Comput. Science* 10 (2015), 278–287.
[38] Sabina Zejnilovic, João Pedro Gomes, and Bruno Sinopoli. 2013. Network observability and localization of the source of diffusion based on a subset of nodes. In *Allerton*. IEEE, 847–852.
[39] Dengyong Zhou, Olivier Bousquet, Thomas Navin Lal, Jason Weston, and Bernhard Schölkopf. 2003. Learning with Local and Global Consistency. In *NIPS*. MIT Press, 321–328.
[40] Kai Zhu, Zhen Chen, and Lei Ying. 2017. Catch'Em All: Locating Multiple Diffusion Sources in Networks with Partial Observations. In *AAAI*. AAAI Press, 1676–1683.
[41] Kai Zhu and Lei Ying. 2013. Information source detection in the SIR model: A sample path based approach. In *ITA*. IEEE, 1–9.
[42] Kai Zhu and Lei Ying. 2014. A robust information source estimator with sparse observations. In *INFOCOM*. IEEE, 2211–2219.
[43] Arkaitz Zubiaga, Ahmet Aker, Kalina Bontcheva, Maria Liakata, and Rob Procter. 2018. Detection and Resolution of Rumours in Social Media: A Survey. *ACM Comput. Surv.* 51, 2 (2018), 32:1–32:36.