A Fuzzy Detection System for Rumors through Explainable Adaptive Learning

Zhiwei Guo, *Member, IEEE,* Keping Yu, *Member, IEEE*, Alireza Jolfaei, *Senior Member, IEEE*, Ali Kashif Bashir, *Senior Member, IEEE*, Alaa Omran Almagrabi, and Neeraj Kumar, *Senior Member, IEEE*

***Abstract*—Nowadays, rumor spreading has gradually evolved into a kind of organized behaviors, accompanied with strong uncertainty and fuzziness. However, existing fuzzy detection techniques for rumors focused their attention on supervised scenarios which require expert samples with labels for training. Thus they are not able to well handle unsupervised scenarios where labels are unavailable. To bridge such gap, this paper proposes a fuzzy detection system for rumors through explainable adaptive learning. Specifically, its core is a graph embedding-based generative adversarial network (Graph-GAN) model. First of all, it constructs fine-grained feature spaces via graph-level encoding. Furthermore, it introduces continuous adversarial training between a generator and a discriminator for unsupervised decoding. The two-stage scheme not only solves fuzzy rumor detection under unsupervised scenarios, but also improves robustness of the unsupervised training. Empirically, a set of experiments are carried out based on three real-world datasets. Compared with seven benchmark methods in terms of four metrics, the results of Graph-GAN reveal a proper performance which averagely exceeds baselines by 5% to 10%.**

***Index Terms*—Fuzzy detection system, graph embedding, generative adversarial learning, cyberspace security.**

This work was supported in part by the Chongqing Natural Science Foundation of China under grant cstc2019jcyj-msxmX0747, in part by the State Language Commission Research Program of China under grant YB135-121, in part by the Science and Technology Research Program of Chongqing Municipal Education Commission under Grant KJQN202000805, in part by the Japan Society for the Promotion of Science (JSPS) Grants-in-Aid for Scientific Research (KAKENHI) under Grant JP18K18044, and in part by the High-level Talents/Teams Research Project of Chongqing Technology and Business University under grant 1853013, grant ZDPTTD201917, and grant KFJJ2018071. (*Corresponding author: Keping Yu*)

Zhiwei Guo is with Chongqing Engineering Laboratory for Detection Control and Integrated System, National Research Base of Intelligent Manufacturing Service, Chongqing Technology and Business University, Chongqing 400067, China. (e-mail: zwguo@ctbu.edu.cn)

Keping Yu is with Global Information and Telecommunication Institute, Waseda University, Shinjuku, Tokyo 169-8050, Japan. (e-mail: keping.yu@aoni.waseda.jp)

Alireza Jolfaei is with Department of Computing, Macquarie University, Sydney, NSW 2113, Australia. (e-mail: alireza.jolfaei@mq.edu.au)

Ali Kashif Bashir is with Department of Computing and Mathematics, E-154, John Dolton, Chester Street, M15 6H, Manchester Metropolitan University, Manchester, United Kingdom, and with School of Electrical Engineering and Computer Science (SEECS), National University of Science and Technology, Islamabad (NUST), Pakistan, and with School of Information and Communication Engineering, University of Electronics Science and Technology of China (UESTC), Chengdu, China. (email: dr.alikashif.b@ieee.org)

Alaa Omran Almagrabi is with Department of Information Systems, Faculty of Computing and Information Technology (FCIT), King Abdul Aziz University (KAU), Jeddah, Saudi Arabia. (e-mail: aalmagrabi3@kau.edu.sa)

Neeraj Kumar is with the Department of Computer Science and Engineering, Thapar Institute of Engineering and Technology, Patiala 147004, India, and with Department of Computer Science and Information Engineering, Asia University, Taiwan, and with School of Computing, University of Petroleum and Energy Studies, Dehradun, Uttarakhand. (e-mail: neeraj.kumar@thapar.edu).

I. Introduction

With the development of Internet, cyberspace has been regarded as another important living space in daily life [1], [2]. Along with the prevalence of diverse social network applications, various rumor spreading events immensely affect cyberspace security and even social stability [3], [4]. For instance, they caused great panic to the public especially during the epidemic COVID-19 [5]. Although conventional artificial intelligence techniques have bred great progress in rumor detection, manual feature extraction usually makes them time-consuming and inefficient [7]. Especially during long-term confrontation against supervision, rumor events tend to evolve into more fuzzy activities to continuously gain concealment [8]. They are gradually becoming a kind of organized activities, accompanied with strong uncertainty and fuzziness [18]. As rumor spreaders have tended to become smarter, so as to avoid being easily identified their real intention [19]. For example, they may carry out social activities as normal users in most of times, yet give specific statements in necessary times. It remains really hard to discover such type of intention, which brings about the issue of uncertainty and fuzziness [25], [29]. The fuzziness makes it hard to efficiently extract hidden or latent patterns from original activity records [9]. To this end, it is expected to integrate deeper insight into feature spaces to establish rumors-oriented fuzzy detection systems [6]. In contrast, deep learning actively learns feature representation via multiple layers of abstraction [10]. Hence, feature components with stronger representative ability can be well extracted [11]. Given complexity of scenes, deep learning has been regarded as an ideal solution to build fuzzy detection systems for rumors [12].

The past few years have witnessed great progress of researches on deep learning-based fuzzy detection of rumors [13]. The research temporality can be viewed as two stages. At the first stage, approaches originated from two models: convolutional neural network (CNN) [14]-[17] and recurrent neural network (RNN) [20]-[23]. CNN-based approaches generally focused on global-level feature abstraction, capturing abstract features of activities from the perspective of integrity. RNN-based approaches emphasized on activity order and mined sequential patterns inside them. For instance, Li et al. [16] adopted multi-task learning to construct a classification-based rumor detection method. It is actually a neural network structure with a shared layer and two task layers. Bugueño et al. [22] learned sequential features from contents of contextual tweets via capturing sequential characteristics, and

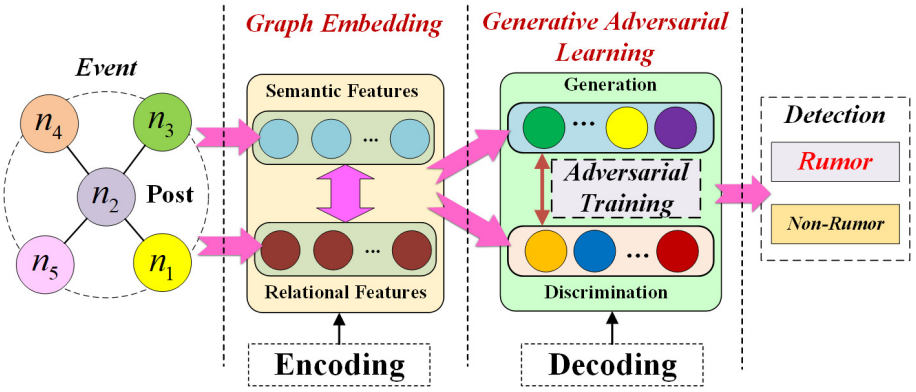


Fig. 1: Architecture of the proposed Graph-GAN model.

employed an RNN model for rumor detection. As for the aforementioned methods, the detection precision is able to reach 75% in average. At the second stage, researchers tried to explore combination of the two networks [26]-[28]. For example, Huang et al. [26] proposed a model captures both local features and long-term features inside social events to realize rumor detection. This type of integrated methods are able to reach detection precision of 80% in average, which possesses a little advancement compared with methods of the first stage. Nevertheless, almost all of them were developed to establish a mapping from features to identification results. They were highly dependent on the existence of expert samples with labels while training. Therefore, they cannot handle unsupervised situations where labels are unavailable. In reality, making annotations on rumor samples is extremely expensive and requires expert experience. Predictably, as the social applications constantly gain their popularity, situations where labels are missing tend to become more universal [30]. Therefore, adaptive learning with proper explainability for unsupervised scenarios will become an essential demand in the future.

The generative adversarial learning (GAL) [31], a deep learning technique newly proposed in recent years, is well suitable for resolving this challenge. The GAL contains two components: a generator and a discriminator [32]. The former generates unknown samples and the latter discriminates whether the generated samples are close to real ones. The adversarial training between them is expected to produce optimal outputs. In addition, the graph embedding (GE) theory [24] is introduced to improve fineness of feature spaces. It manages to construct more fine-grained feature spaces by simultaneously extracting entities and relations inside networked objects for encoding [33]. The combination of GE encoding and GAL decoding, constitutes an explainable adaptive learning framework. Thus, this paper proposes a fuzzy detection system for rumors through **graph** embedding-based **g**enerative **a**dversarial **n**etwork (Graph-GAN). Specifically, graph-level features are extracted and encoded from initial contents as well as associated contextual information. Based upon this, GAL is introduced as an adaptive decoder to actively learn rules of feature spaces. Then, a set of experiments are implemented on three real-world datasets to assess performance of the proposed Graph-GAN in terms of accuracy and robustness. Main contributions of this research are summarized as follows:

- In terms of fuzzy detection for rumors, this work recognizes the importance of situations where labels may be unavailable for model training.
- To deal with the problem raised here, the Graph-GAN model is put forward to establish a fuzzy detection system for rumors.
- A considerable number of experiments are conducted to assess efficiency and proactivity of the proposed Graph-GAN.

The rest of this paper is organized as follows. In Section II, problem situations are defined and overall workflow of the proposed Graph-GAN is presented. The detailed algorithmic procedures of Graph-GAN are illustrated in Section III. In Section IV, we set experimental scenarios and give experimental results as well as analysis. And we conclude this paper in Section V.

II. Problem Statement

It is assumed that $\mathcal{Q}_n\,(n=1,2,\cdots,\mathcal{N})$ denotes the set of events, and that that each event which consists of a set of posts is actually a graph network. In this paper, the main goal is to distinguish rumor events from a huge amount of social speech events. To begin with, basic items in this research are defined:

Definition 1 (Post): A post refers to a specific speech released by a social user, such as microblog, tweet, etc.

Definition 2 (Event): An event refers to a series of specific social activities that have happened or are happening. In this paper, it contains a collection of posts and is the object to be identified.

Fig. 1 presents workflow of the proposed Graph-GAN which contains three parts: graph-level encoding, GAL and detection. The posts belonging to an event are viewed as nodes, and implicit relations among posts are viewed as edges. Within an event $\mathcal{Q}_n$, let $\mathcal{P}_i\,(i=1,2,\cdots,|\mathcal{P}|)$ denote the set of $|\mathcal{P}|$ associated posts. For post $\mathcal{P}_i$, it is firstly expected to make graph-level encoding towards it. For one thing, its semantic features are modeled by developing a CNN operator to produce a representative vector $\boldsymbol{\mathcal{V}}_i$. For another, its relational features between others are modeled to produce a representative vector

$\mathcal{R}_i$. Extended into the event $\mathcal{Q}_n$, the two post-level vectors are transferred as event-level matrices $\boldsymbol{\mathcal{V}}_{(n)}$ and $\boldsymbol{\mathcal{R}}_{(n)}$, respectively. Then, the GAL is set up to decode the feature spaces of $\mathcal{Q}_n$. Particularly, the generator G is updated in each iterative round, and the discriminator D are trained at the beginning. The corresponding outputs of them are two sample vectors $\boldsymbol{\mathcal{G}}_n^{(t)}$ and $\boldsymbol{\mathcal{D}}_n'$, and the Wasserstein distance (WSD)-based objective function is built for adversarial training between them. Finally, all the speeches can be classified to identify rumor events. Intuitively, rumor detection can be abstracted as a binary classification problem of social events. Because single social speeches are generally short and not informative, it remains hard to precisely judge nature of an event through literal meaning or local features. As a result, extension of initial features is quite vital. Given above, this paper is also established on the basis of some assumptions:

Assumption 1: Inside an event, correlations among posts are assumed to be not sequential, and posts released earlier will not affect the follow-up posts. In a word, each event is independent and will not affect processes of others.

Assumption 2: Each post never exists independently, and is associated with an event. It is released by a user who has personal profiles and social relations.

Assumption 3: Social relations exist in a pair of users when they have common records of comments, response or reposts.

III. Methodology

This section describes mathematical processes of the proposed Graph-GAN through three main procedures. Firstly, GE theory is utilized to generate feature abstraction for posts and their relations, so that fine-grained feature spaces can be built. Then, the GAL is formulated to decode feature spaces. And within the last detection procedure, rumors can be detected naturally by learned parameters.

A. Graph-Level Encoding

As is illustrated in Fig. 2, graph-level encoding for event $\mathcal{Q}_n$ mainly contains two parts: modeling of contents and modeling of relations.

1) Modeling of Contents: Word sequence of $\mathcal{P}_i$ can be denoted as a word vector $\boldsymbol{\Omega}_i = [w_1, w_2, \cdots, w_\gamma]$, where $w_j\,(j = 1, 2, \cdots, \gamma)$ denotes all the γ words in it. For numerical calculation, each word is transformed into a vector through one-hot encoding. Specifically, dimension of such a vector is the size of word set which is assumed to be τ. And each element of the vector corresponds to a word in the word set. The one-hot encoding results for $\boldsymbol{\Omega}_i$ are denoted as:

$$\boldsymbol{\Omega}_i^{(1)} = \left[\boldsymbol{v}_1^T, \boldsymbol{v}_2^T, \cdots, \boldsymbol{v}_\gamma^T\right] \tag{1}$$

where $\boldsymbol{v}_j (j = 1, 2, \cdot, \gamma)$ denotes all the γ encoding vectors for γ words,and each $\boldsymbol{v}_j$ is a τ-dimensional vector. The j-th word vector $\boldsymbol{v}_j^T$ is semantically related to adjacent k vectors before and after it. Merging the $\boldsymbol{v}_j^T$ and its adjacent $2k$ vectors produces feature matrix for word w_j, which is denoted as:

$$\boldsymbol{d}_j = \left[\boldsymbol{v}_{j-k}^T, \boldsymbol{v}_{j-k+1}^T, \cdots, \boldsymbol{v}_j^T, \cdots, \boldsymbol{v}_{j+k-1}^T, \boldsymbol{v}_{j+k}^T\right] \tag{2}$$

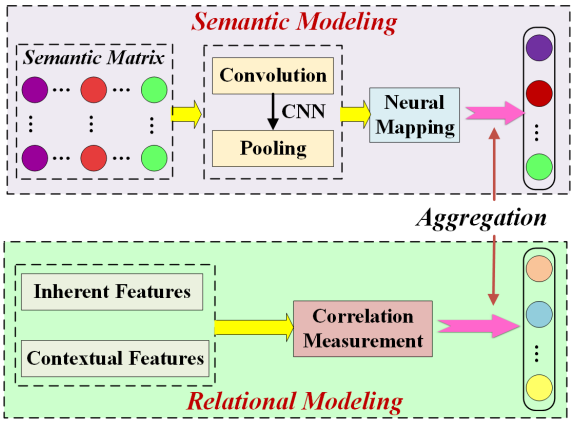


Fig. 2: Flowchart of the graph-level encoding.

As for semantic modeling, CNN operator is selected for two reasons. Firstly, posts are generally short text, so that long-term dependence can be ignored. Secondly, CNN is able to reduce dimensions while extracting features because dimension of the obtained vector $\boldsymbol{\Omega}_i^{(1)}$ is too high. The CNN operator in this research is defined as a series of convolution and pooling operations. For the former, word matrix $\boldsymbol{d}_j$ is fed into Ψ-core filters for convolutional calculation. For the latter, dimension of the feature matrix obtained after convolution is further reduced via subsampling. Each combination of convolution and pooling is viewed as a filter group. Index number of groups is assumed as θ which ranges from 1 to η.

In the θ-th filtering, a feature matrix $\boldsymbol{\mathcal{B}}_j^{(\theta)}$ is obtained after convolutional transformation, and the process is denoted as:

$$\boldsymbol{\mathcal{B}}_j^{(\theta)} = \sigma_1\left\{\sum_{\psi=1}^{\Psi}\left[\boldsymbol{W}_{\mathcal{V}1}^{(\theta)(\psi)} \otimes \mathcal{F}_1\left(d_j\right) + \boldsymbol{b}_{\mathcal{V}1}^{(\theta)(\psi)}\right]\right\} \tag{3}$$

where $\sigma_1(\cdot)$ denotes the ReLU activation function, $\otimes$ denotes convolution operation, $\boldsymbol{W}_{\mathcal{V}1}^{(\theta)(\psi)}$ and $\boldsymbol{b}_{\mathcal{V}1}^{(\theta)(\psi)}$ are parameters, ψ is index number of filter cores, and $\mathcal{F}_1(\cdot)$ defines a nonlinear mapping as follows:

$$\mathcal{F}_1\left(\boldsymbol{d}_j\right) = \boldsymbol{W}_{\mathcal{V}2} \cdot \boldsymbol{d}_j^T + \boldsymbol{b}_{\mathcal{V}2} \tag{4}$$

where $\boldsymbol{W}_{\mathcal{V}2}$ and $\boldsymbol{b}_{\mathcal{V}2}$ are parameters, and the $\mathcal{F}_1\left(\boldsymbol{d}_j\right)$ transforms $\boldsymbol{d}_j$ into a $\tau \times \tau$-dimensional matrix. Note that the ReLU activation function is represented as:

$$\sigma_1(x) = \max(0, x) \tag{5}$$

In pooling operation, every 4×4 matrix is set as a pooling block and only the maximum value is selected to represent the block. Word matrix $\boldsymbol{d}_j$ is transformed into a matrix $\boldsymbol{\mathcal{B}}_j$ through all the η filter groups. It is further transferred to a Φ-core full connection filter, producing another representative vector $\boldsymbol{\mathcal{B}}_j'$:

$$\boldsymbol{\mathcal{B}}_j' = \sigma_1\left\{\sum_{\phi=1}^{\Phi}\left[\boldsymbol{W}_{\mathcal{V}3}^{(\phi)} \cdot \boldsymbol{\mathcal{B}}_j + \boldsymbol{b}_{\mathcal{V}3}^{(\phi)}\right]\right\} \tag{6}$$

where $\boldsymbol{W}_{\mathcal{V}3}^{(\phi)}$ and $b_{\mathcal{V}3}^{(\phi)}$ are parameters, and ϕ is the index number of cores. Note that $\boldsymbol{\mathcal{B}}_j'$ is a normalized vector. Considering diverse roles of different words, an attention layer is further introduced. We firstly denote initial semantic matrix as the concatenation of all the $\boldsymbol{\mathcal{B}}_j'$, where j ranges from 1 to γ:

$$\boldsymbol{\mathcal{V}}_i' = \left[\boldsymbol{\mathcal{B}}_1'^{T}, \boldsymbol{\mathcal{B}}_2'^{T}, \cdots, \boldsymbol{\mathcal{B}}_\gamma'^{T}\right] \tag{7}$$

Then we denote the attention weight of the j-th word as the following two formulas:

$$\alpha_j' = \sigma_2\left[\boldsymbol{W}_{\mathcal{V}4} \cdot \boldsymbol{\mathcal{V}}_i' + \boldsymbol{b}_{\mathcal{V}4}\right] \cdot \boldsymbol{\beta}_j^T \tag{8}$$

$$\alpha_j = \frac{\exp\left(\alpha'_j\right)}{\sum_{j=1}^{\gamma} \exp\left(\alpha'_j\right)} \tag{9}$$

where $\boldsymbol{W}_{\mathcal{V}4}$ and $\boldsymbol{b}_{\mathcal{V}4}$ are parameters, $\boldsymbol{\beta}_j^T$ is weight factor for the j-th word, and $\sigma_2(\cdot)$ denotes the tanh activation function:

$$\sigma_2(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \tag{10}$$

The post-level representative vector for $\mathcal{P}_i$ is obtained as:

$$\boldsymbol{\mathcal{V}}_i = \sum_{j=1}^{\gamma} \alpha_j \cdot \boldsymbol{\mathcal{B}}_j' \tag{11}$$

Accordingly, the event-level representative matrix for $\mathcal{Q}_n$ is denoted as:

$$\boldsymbol{\mathcal{V}}_{(n)} = \left[\boldsymbol{\mathcal{V}}_1^T, \boldsymbol{\mathcal{V}}_2^T, \cdots, \boldsymbol{\mathcal{V}}_{|\mathcal{P}|}^T\right] \tag{12}$$

2) Modeling of Relations: Relations among posts are divided into two types inherent relations and contextual relations. All the relations between post $\mathcal{P}_i$ and $\mathcal{P}_l\,(l = 1, 2, \cdots, |\mathcal{P}|; l \neq i)$ are initially denoted as:

$$\begin{aligned}\boldsymbol{\mathcal{R}}_{i,l} =& \lambda_1 \cdot \omega_{i,l}^{(inh)} \cdot \boldsymbol{W}_{\mathcal{R}1} \cdot \boldsymbol{\Lambda}_{i,l}^{(inh)} \\ &+ (1-\lambda_1) \cdot \omega_{i,l}^{(con)} \cdot \boldsymbol{W}_{\mathcal{R}2} \cdot \boldsymbol{\Lambda}_{i,l}^{(con)} + \boldsymbol{b}_{\mathcal{R}1}\end{aligned} \tag{13}$$

where $\boldsymbol{W}_{\mathcal{R}1}$, $\boldsymbol{W}_{\mathcal{R}2}$ and $\boldsymbol{b}_{\mathcal{R}1}$ are parameters, $\omega_{i,l}^{(inh)}$ and $\omega_{i,l}^{(con)}$ are the weight factors from inherent information and contextual information, $\boldsymbol{\Lambda}_{i,l}^{(inh)}$ and $\boldsymbol{\Lambda}_{i,l}^{(con)}$ are two similarity vectors between $\mathcal{P}_i$ and $\mathcal{P}_l$, and λ_1 is the trade-off parameter.

The $\omega_{i,l}^{(inh)}$ is mainly decided by similarity of their words, and it is computed as:

$$\omega_{i,l}^{(inh)} = \begin{cases} \dfrac{\sum_{j=1}^{\gamma} \varphi_j(i \cap l)/\varphi_j(l)}{\sum_{p=1}^{|P|}\sum_{j=1}^{\gamma} \varphi_j(i \cap p)/\varphi_j(p)}, & i \neq l \\ 0, & i = l \end{cases} \tag{14}$$

where $\varphi_j(l)$ and $\varphi_j(p)$ count the number of word w_j in post $\mathcal{P}_l$ and $\mathcal{P}_p$ respectively, $\varphi_j(i \cap l)$ counts the cooccurrence number of word w_j in $\mathcal{P}_i$ and $\mathcal{P}_l$, $\varphi_j(i \cap p)$ counts the cooccurrence number of word w_j in $\mathcal{P}_i$ and $\mathcal{P}_p$, and p is another index number of posts that satisfies $p \neq i$. The similarity vector $\boldsymbol{\Lambda}_{i,l}^{(inh)}$ is measured as:

$$\boldsymbol{\Lambda}_{i,l}^{(inh)} = \left\|\boldsymbol{\mathcal{V}}_i - \boldsymbol{\mathcal{V}}_l\right\| \tag{15}$$

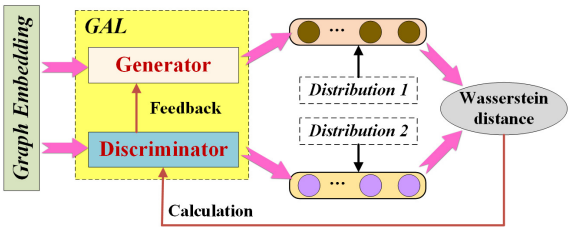


Fig. 3: Flowchart of the Generative Adversarial Learning.

It is assumed that posts $\mathcal{P}_i$ and $\mathcal{P}_l$ are released by users $\mathcal{U}_i$ and $\mathcal{U}_l$. Although i is inequal to l, it is possible for $\mathcal{U}_i$ and $\mathcal{U}_l$ to be the same user. The $\omega_{i,l}^{(con)}$ is mainly decided by correlations between their users, and is computed as:

$$\omega_{i,l}^{(con)} = \begin{cases} \dfrac{\mathcal{F}_2(i,l)\cdot[\varphi_u(\mathcal{U}_i \cap \mathcal{U}_l)/\varphi_u(\mathcal{U}_l)]}{\sum_{u=1}^{|\mathcal{U}|} F_2(i,l)\cdot[\varphi_u(\mathcal{U}_i \cap \mathcal{U}_u)/\varphi_u(\mathcal{U}_u)]}, & \mathcal{U}_i \neq \mathcal{U}_l \\ 0, & \mathcal{U}_i = \mathcal{U}_l \end{cases} \tag{16}$$

where u is the index number of users that ranges from 1 to $|\mathcal{U}|$, $\mathcal{U}_u$ is another user different from $\mathcal{U}_i$, $\varphi_u(\mathcal{U}_l)$ and $\varphi_u(\mathcal{U}_u)$ count number of friends concerning $\mathcal{U}_l$ and $\mathcal{U}_u$, $\varphi_u(U_i \cap U_l)$ counts number of common friends between $\mathcal{U}_i$ and $\mathcal{U}_l$, and $\varphi_u(\mathcal{U}_i \cap \mathcal{U}_u)$ counts number of common friends between $\mathcal{U}_i$ and $\mathcal{U}_u$. In addition, $\mathcal{F}_2(i,l)$ is used to judge whether $\mathcal{U}_i$ and $\mathcal{U}_l$ are friends. $\mathcal{F}_2(i,l)$ equals to 1 if they are friends, and 0 otherwise.

For user $\mathcal{U}_i$, his social relation vector is denoted as $\boldsymbol{\mathcal{S}}_i^{(fri)}$. It is a $(|\mathcal{U}|-1)$-dimensional vector, in which the $(|\mathcal{U}|-1)$ elements correspond to relationships between him and others. If corresponding friendship exists, an element equals to 1; otherwise, it equals to 0. Besides, his profile vector is denoted as $\boldsymbol{\mathcal{S}}_i^{(pro)}$. Of all the attributes in user profiles, only structural attributes are selected, which refer to those whose values are one of several fixed options, such as sex, location, etc.. In one-hot encoding, the selected option is set to 1 and other options are set to 0. This scheme is well suitable for structural attributes, and concatenation of all the encoded attributes constitute profile vector $\boldsymbol{\mathcal{S}}_i^{(pro)}$. When $\mathcal{U}_i = \mathcal{U}_l$, the similarity vector $\boldsymbol{\Lambda}_{i,l}^{(con)}$ equals to 0. When $\mathcal{U}_i \neq \mathcal{U}_l$, the similarity vector $\boldsymbol{\Lambda}_{i,l}^{(con)}$ is measured as:

$$\boldsymbol{\Lambda}_{i,l}^{(inh)} = \left\|\boldsymbol{\mathcal{S}}_i^{(fri)} + \boldsymbol{\mathcal{S}}_i^{(pro)} - \boldsymbol{\mathcal{S}}_j^{(fri)} - \boldsymbol{\mathcal{S}}_j^{(pro)}\right\| \tag{17}$$

The final relational value between $\mathcal{P}_i$ and $\mathcal{P}_l$ is calculated as:

$$\mathcal{R}_{i,l}' = \boldsymbol{\mathcal{R}}_{i,l} \cdot \boldsymbol{\mathcal{R}}_{i,l}^T \tag{18}$$

All the $\boldsymbol{\mathcal{R}}_{i,l}'$ are concatenated into a $(|P|-1)$-dimensional representative vector. And it is further transformed into $\boldsymbol{\mathcal{R}}_i$ after normalization, which is expressed as:

$$\boldsymbol{\mathcal{R}}_i = \left[\mathcal{R}_{i,1}', \cdots, \mathcal{R}_{i,i-1}', \mathcal{R}_{i,i+1}', \cdots, \mathcal{R}_{i,|\mathcal{P}|}'\right] \tag{19}$$

The event-level representative matrix for $\mathcal{Q}_n$ is denoted as:

$$\boldsymbol{\mathcal{R}}_{(n)} = \left[\boldsymbol{\mathcal{R}}_1^T, \boldsymbol{\mathcal{R}}_2^T, \cdots, \boldsymbol{\mathcal{R}}_{|\mathcal{P}|}^T\right] \tag{20}$$

Algorithmic process of graph-level encoding is illustrated in Algorithm 1.

Algorithm 1 Graph-Level Encoding

INPUT: $\mathcal{Q}_n$, $\mathcal{P}_i\,(i=1,2,\cdots,|\mathcal{P}|)$, $\mathbf{\Omega}_i$, k, η, Ψ, τ, Φ, λ_1
OUTPUT: $\boldsymbol{\mathcal{V}}_{(n)}$ and $\boldsymbol{\mathcal{R}}_{(n)}$
for $n=1\rightarrow\mathcal{N}$ **do**
  for $i=1\rightarrow|\mathcal{P}|$ **do**
    Encode $\mathbf{\Omega}_i$ into $\mathbf{\Omega}_i^{(1)}$ as Eq. (1)
    for $j=1\rightarrow\gamma$ **do**
      Set up feature matrix for w_j as Eq. (2)
      for $\theta=1\rightarrow\eta$ **do**
        for $\psi=1\rightarrow\Psi$ **do**
          Compute $\boldsymbol{\mathcal{B}}_j^{(\theta)}$ as Eq. (3), (4) and (5)
        end for
      end for
      for $\phi=1\rightarrow\Phi$ **do**
        Compute $\boldsymbol{\mathcal{B}}_j'$ as Eq. (6)
      end for
      Concatenate all the $\boldsymbol{\mathcal{B}}_j'$ into $\boldsymbol{\mathcal{V}}_i'$ as Eq. (7)
      Compute $\boldsymbol{\mathcal{V}}_i$ from Eq. (8) to Eq. (11)
    end for
    for $l=1\rightarrow|\mathcal{P}|$ and $l\neq i$ **do**
      Compute $R_{i,l}$ from Eq. (13) to (17)
      Compute $R_{i,l}'$ as Eq. (18)
      Concatenate all the $R_{i,l}'$ into R_i as Eq. (19)
    end for
  end for
end for

B. Generative Adversarial Learning

As is shown in Fig. 3, architecture of GAL is essentially the adversarial training between a generator G and a discriminator D. During each round of iterations, the G projects distribution of original feature spaces $\boldsymbol{\mathcal{V}}_{(n)}$ and $\boldsymbol{\mathcal{R}}_{(n)}$ into a simulated factor of label. At the same time, the D provides feedback for G by estimating WSD between real and simulated label factors according to some distributions.

1) Generator: To better fit sequential iterative multiple-round updating, the gated recurrent unit (GRU) model is embedded into G. The fusion of two representative vectors $\boldsymbol{\mathcal{V}}_{(n)}$ and $\boldsymbol{\mathcal{R}}_{(n)}$ into generated prediction results undergoes an iterative process which is denoted as t and ranges from 1 to $\mathcal{T}$. At the t-th round, the hidden state vector is updated via GRU operator according to hidden state of the previous round. The GRU consists of two gates: update gate (UG) and reset gate (RG). UG controls the degree where state information of the previous round is brought into the current round, and RG controls the degree where state information of the previous round is neglected. Therefore, hidden state at such timestamp is represented as:

$$\boldsymbol{\mathcal{H}}_n^{(t)}=GRU\left[\boldsymbol{\mathcal{H}}_n^{(t-1)},\boldsymbol{\mathcal{A}}_n^{(t)}\right] \tag{21}$$

where $\boldsymbol{\mathcal{A}}_n^{(t)}$ is the aggregation state and denoted as:

$$\boldsymbol{\mathcal{A}}_n^{(t)}=\boldsymbol{W}_{G1}\cdot\boldsymbol{\mathcal{H}}_n^{(t-1)}+\boldsymbol{b}_{G1} \tag{22}$$

where $\boldsymbol{W}_{G1}$ is weight parameter, and $\boldsymbol{b}_{G1}$ is bias parameter. Values of UG $\boldsymbol{z}_{n-up}^{(t)}$ and $\boldsymbol{z}_{n-fo}^{(t)}$ are formulated as:

$$\boldsymbol{z}_{n-up}^{(t)}=\sigma_3\left[\boldsymbol{W}_{z1}\cdot\boldsymbol{\mathcal{A}}_n^{(t)}+\boldsymbol{\mathcal{Z}}_1\cdot\boldsymbol{\mathcal{H}}_n^{(t-1)}+\boldsymbol{b}_{z1}\right] \tag{23}$$

$$\boldsymbol{z}_{n-fo}^{(t)}=\sigma_3\left[\boldsymbol{W}_{z2}\cdot\boldsymbol{\mathcal{A}}_n^{(t)}+\boldsymbol{\mathcal{Z}}_2\cdot\boldsymbol{\mathcal{H}}_n^{(t-1)}+\boldsymbol{b}_{z2}\right] \tag{24}$$

where $\boldsymbol{W}_{z1}$, $\boldsymbol{W}_{z2}$, $\boldsymbol{\mathcal{Z}}_1$, $\boldsymbol{\mathcal{Z}}_2$ are weight parameters, $\boldsymbol{b}_{z1}$ and $\boldsymbol{b}_{z2}$ are bias parameters, and $\sigma_3(\cdot)$ denotes sigmoid activation function which is represented as:

$$\sigma_3(x)=\frac{1}{1+e^{-x}} \tag{25}$$

Thus, Eq. (21) can be rewritten as:

$$\boldsymbol{\mathcal{H}}_n^{(t)}=\tilde{\boldsymbol{\mathcal{H}}}_n^{(t)}\odot\boldsymbol{z}_{n-up}^{(t)}+\boldsymbol{\mathcal{H}}_n^{(t)}\odot\left[1-\boldsymbol{z}_{n-up}^{(t)}\right] \tag{26}$$

where $\odot$ denotes element-wise multiplication, and $\tilde{\boldsymbol{\mathcal{H}}}_n^{(t)}$ is a transition vector denoted as:

$$\begin{aligned}&\tilde{\boldsymbol{\mathcal{H}}}_n^{(t)}\\&=\sigma_2\left\{\boldsymbol{W}_{z3}\cdot\boldsymbol{\mathcal{H}}_n^{(t)}+\boldsymbol{\mathcal{Z}}_3\cdot\left[\boldsymbol{z}_{n-fo}^{(t)}\odot\boldsymbol{\mathcal{H}}_n^{(t-1)}\right]+\boldsymbol{b}_{z3}\right\}\end{aligned} \tag{27}$$

where $\boldsymbol{W}_{z3}$ and $\boldsymbol{\mathcal{Z}}_3$ are weight parameters, and $\boldsymbol{b}_{z3}$ is bias parameter. Note that $\boldsymbol{\mathcal{H}}_n^{(t)}$ of initial state is expressed as:

$$\boldsymbol{\mathcal{H}}_n^{(0)}=\left[\boldsymbol{\mathcal{V}}_{(n)}\oplus\boldsymbol{\mathcal{R}}_{(n)}\right] \tag{28}$$

As the $\boldsymbol{\mathcal{H}}_n^{(t)}$ captures global latent information via mixture of posts and their relations, its elements naturally play different roles. The attention mechanism is added to output original distributions. The hidden state vector is firstly transformed into another vector $\boldsymbol{h}_n^{(t)}$ through a neural network mapping procedure:

$$\boldsymbol{h}_n^{(t)}=\sigma_1\left[\boldsymbol{W}_{G2}\cdot\boldsymbol{\mathcal{H}}_n^{(t)}+\boldsymbol{b}_{G2}\right] \tag{29}$$

where $\boldsymbol{W}_{G2}$ is weight parameter, and $\boldsymbol{b}_{G2}$ is bias parameter. Introducing attention weight parameter a_n for $\boldsymbol{h}_n^{(t)}$, simulated factors from the G is represented as:

$$\boldsymbol{\mathcal{G}}_n^{(t)}=\sigma_1\left\{a_n\cdot\left[\boldsymbol{h}_n^{(t)}\right]^T\right\} \tag{30}$$

2) Discriminator: The discriminator D projects initial samples into real factors of labels. Different from G, the D produces real factors for labels at the beginning via multiple rounds of propagations. Index number of propagations is denoted as m that ranges from 1 to $\mathcal{M}$.

At the m-th step, vector $\boldsymbol{\mathcal{V}}_{(n)}^{(m)}$ is updated through the following formula:

$$\boldsymbol{\mathcal{V}}_{(n)}^{(m)}=\sigma_3\left[\boldsymbol{\mathcal{V}}_{(n)}^{(m-1)}+\boldsymbol{\mathcal{C}}_{v1}^{(m)}\right] \tag{31}$$

where $\boldsymbol{\mathcal{C}}_{v1}^{(m)}$ is state variable denoted as:

$$\boldsymbol{\mathcal{C}}_{v1}^{(m)}=\sigma_1\left\{\boldsymbol{W}_{D1}\cdot\left[\boldsymbol{\mathcal{V}}_{(n)}^{(m-1)}\oplus\boldsymbol{\mathcal{R}}_{(n)}^{(m-1)}\right]+\boldsymbol{b}_{D1}\right\} \tag{32}$$

where $\boldsymbol{W}_D1$ is weight parameter, and $\boldsymbol{b}_{D1}$ is bias parameter. Besides, vector $\boldsymbol{\mathcal{R}}_{(n)}^{(m)}$ is updated via the following formula:

$$\boldsymbol{\mathcal{R}}_{(n)}^{(m)}=\sigma_3\left[\boldsymbol{\mathcal{R}}_{(n)}^{(m-1)}+\boldsymbol{\mathcal{C}}_{v2}^{(m)}\right] \tag{33}$$

TABLE I: Brief description of main symbols with respect to indices.

Indices:	
n	Index number of all the events ($n = 1, 2, \cdot, \mathcal{N}$)
i	Index number of all the posts ($i = 1, 2, \cdot, \lvert\mathcal{P}\rvert$)
j	Index number of words in a post ($j = 1, 2, \cdot, \gamma$)
θ	Index number of operation rounds in CNN ($\theta = 1, 2, \cdot, \eta$)
ψ	Index number of convolutional cores in CNN ($\psi = 1, 2, \cdot, \Psi$)
t	Index number of all iterative rounds ($t = 1, 2, \cdot, \mathcal{T}$)
k	Index number of adjacent words ($k > 1$)
λ_1, λ_2, λ_3	Trade-off parameters ($0 < \lambda_1, \lambda_2, \lambda_3 < 1$)

TABLE II: Brief description of symbols with respect to main parameters.

Parameters:	
$\mathcal{Q}_n$	The set of events whose size
$\mathcal{P}_i$	The set of posts associated with each event
$\boldsymbol{\Omega_i}$	The word vector associated with post $\mathcal{P}_i$
w_j	The set of words inside post $\mathcal{P}_i$
$\boldsymbol{\Omega_i^{(1)}}$	Encoding vector for $\boldsymbol{\Omega_i}$
$\boldsymbol{d}_j$	Feature matrix for word w_j
θ	Index number of convolutional transformation rounds
ψ	Index number of convolutional cores
$\omega_{i,l}^{inh}$, $\omega_{i,l}^{com}$	Similarity weight between post $\mathcal{P}_i$ and post $\mathcal{P}_l$
$\boldsymbol{\Lambda}_{i,l}^{(inh)}$, $\boldsymbol{\Lambda}_{i,l}^{(con)}$	Similarity vector between post $\mathcal{P}_i$ and post $\mathcal{P}_l$
$\mathcal{R}'_{i,l}$	Final relation value between post $\mathcal{P}_i$ and post $\mathcal{P}_l$
$\boldsymbol{\mathcal{R}}_i$	Normalized relation vector for post $\mathcal{P}_i$
$\boldsymbol{\mathcal{E}}(a, b)$	WSD between two distributions a and b
$J_{\mathcal{G}}$	Distribution corresponding to $\boldsymbol{\mathcal{G}}_n^{(t)}$
$J_{\mathcal{D}}$	Distribution corresponding to $\boldsymbol{\mathcal{D}}_n^{(t)}$
$\mathcal{L}_{\mathcal{D}}$, $\mathcal{L}_{\mathcal{G}}$	Objective function corresponding to D and G
$\mathcal{L}$	Final objective function
$\mathcal{F}_1$, $\mathcal{F}_2$, $\mathcal{F}_3$	Non-linear transformation functions

where state variable $\boldsymbol{\mathcal{C}}_{v2}^{(m)}$ is denoted as:

$$\boldsymbol{\mathcal{C}}_{v2}^{(m)} = \sigma_1\left[\boldsymbol{W}_{D2} \cdot \boldsymbol{\mathcal{V}}_i^{(m-1)} + \boldsymbol{b}_{D2}\right] \tag{34}$$

Given above, real factor of label is expressed with the aid of a neural mapping as follows:

$$\boldsymbol{\mathcal{D}}'_n = \sigma_1\left\{\boldsymbol{W}_{D3} \cdot \left[\boldsymbol{\mathcal{V}}_{(n)}^{(m)} \oplus \boldsymbol{\mathcal{R}}_{(n)}^{(m)}\right] + \boldsymbol{b}_{D3}\right\} \tag{35}$$

During each round t, D measures distance between real and simulated factors of labels. The WSD of discrete form measures distance between two sample distributions by counting work amount of changing one into the other [34]. It is assumed that two vectors $\boldsymbol{\mathcal{G}}_n^{(t)}$ and $\boldsymbol{\mathcal{D}}'_n$ are drawn from two distributions $\mathcal{J}_{\mathcal{G}}$ and $\mathcal{J}_{\mathcal{D}}$ which come from the Dirac delta [35]. The expressions are represented as:

$$\mathcal{E}\left(\mathcal{J}_{\mathcal{G}} \left| \mathcal{J}_{\mathcal{D}} \right.\right) = \min_{\delta_{G,D}^{(t)} \geq 0}\left\{\sum_{n=1}^{\mathcal{N}} \delta_{G,D}^{(t)} \cdot EUC\left[\boldsymbol{\mathcal{G}}_n^{(t)}, \boldsymbol{\mathcal{D}}'_n\right]\right\} \tag{36}$$

where $EUC\left[\boldsymbol{\mathcal{G}}_n^{(t)}, \boldsymbol{\mathcal{D}}'_n\right]$ denotes Euclidean distance between two samples. When extending discrete distributions into continuous ones, WSD of general forms can be set up as:

$$\begin{aligned}&\mathcal{E}'\left(\mathcal{J}_{\mathcal{G}} \left| \mathcal{J}_{\mathcal{D}} \right.\right)\\ &= \inf_{\mu \in \Gamma(\mathcal{J}_{\mathcal{G}}, \mathcal{J}_{\mathcal{D}})}\left\{\sum_{n=1}^{\mathcal{N}} \int EUC\left[\boldsymbol{\mathcal{G}}_n^{(t)}, \boldsymbol{\mathcal{D}}'_n\right] d\mu\left[\boldsymbol{\mathcal{G}}_n^{(t)}, \boldsymbol{\mathcal{D}}'_n\right]\right\}\end{aligned} \tag{37}$$

where $\Gamma(\cdot)$ denotes the joint probabilistic distribution. Inspired by typical Wasserstein distance-based GAL, learning goal of the D is to minimize the following formula:

$$\begin{aligned}&\mathcal{D}\left(\mathcal{J}_{\mathcal{G}} \left| \mathcal{J}_{\mathcal{D}} \right.\right)\\ &= \inf_{\mu \in \Gamma(J_{\mathcal{G}}, J_{\mathcal{D}})}\left\{\sum_{n=1}^{\mathcal{N}} E_{\left[\boldsymbol{\mathcal{G}}_n^{(t)}, \boldsymbol{\mathcal{D}}'_n\right] \sim \mu} \cdot \mathcal{E}'\left[\boldsymbol{\mathcal{G}}_n^{(t)}, \boldsymbol{\mathcal{D}}'_n\right]\right\}\end{aligned} \tag{38}$$

where $E(\cdot)$ denotes the expectation.

TABLE III: Brief description of symbols with respect to main decision variables.

Decision Variables:	
v_j	Encoding vectors for words
$\boldsymbol{\mathcal{B}}_j^{(\theta)}$	Feature matrix of the θ-th round of iteration
$\boldsymbol{\mathcal{B}}_j'$	Final feature matrix after transformation
$\boldsymbol{\mathcal{V}}_i$	Representative vector for $\mathcal{P}_i$
$\boldsymbol{\mathcal{V}}_{(n)}$	Event-level representative matrix for event $\mathcal{Q}_n$
$R_{i,l}$	Initial relation vector between post $\mathcal{P}_i$ and post $\mathcal{P}_l$
$\boldsymbol{\mathcal{R}}_{(n)}$	Event-level representative matrix for event $\mathcal{Q}_n$
$\boldsymbol{\mathcal{H}}_n^{(t)}$	Hidden state for event $\mathcal{Q}_n$ at such timestamp
$\boldsymbol{\mathcal{A}}_n^{(t)}$	Aggregation state for event $\mathcal{Q}_n$ at such timestamp
$\boldsymbol{h}_n^{(t)}$	Aggregation state for event $\mathcal{Q}_n$ at the t-th timestamp
$\boldsymbol{\mathcal{G}}_n^{(t)}$	Representation of G at the t-th timestamp
a_n	Attention weight parameter for $\boldsymbol{h}_n^{(t)}$
$\boldsymbol{\mathcal{D}}_n'$	Representation of D at the t-th timestamp
$\{\boldsymbol{W}, \boldsymbol{\mathcal{Z}}\}$	Weight parameters
$\{\boldsymbol{b}\}$	Bias parameters

3) Training and Optimization: As the aforementioned probabilistic distribution in Eq. (37) and (38) is discrete, a mapping function is required to transform it into continuous form. And the WSD can be defined as supreme of a continuous function. Eq. (38) can be rewritten as:

$$\begin{aligned}&\mathcal{D}\left(\mathcal{J}_{\mathcal{G}}\left|\mathcal{J}_{\mathcal{D}}\right.\right)\\&=\frac{1}{K}\sup_{\|\mathcal{F}_3\|\leq K}\left\{\sum_{n=1}^{\mathcal{N}}E_{\boldsymbol{\mathcal{G}}_n^{(t)}\sim\mathcal{J}_G}\left[\mathcal{F}_3\left(\boldsymbol{\mathcal{G}}_n^{(t)}\right)\right]-E_{\boldsymbol{\mathcal{D}}_n'\sim\mathcal{J}_{\mathcal{D}}}\left[F_3\left(\boldsymbol{\mathcal{D}}_n'\right)\right]\right\}\end{aligned}\tag{39}$$

where $\mathcal{F}_3(\cdot)$ denotes the K-Lipschitz continuous function. Thus, objective of the D is to estimate parameters of $\mathcal{F}_3(\cdot)$ and find its supreme, leading to the following formula:

$$\max_D\sum_{n=1}^{\mathcal{N}}\mathcal{L}_D\tag{40}$$

where

$$\mathcal{L}_D=\sum_{n=1}^{\mathcal{N}}E_{\boldsymbol{\mathcal{G}}_n^{(t)}\sim\mathcal{J}_{\mathcal{G}}}\left[\mathcal{F}_3\left(\boldsymbol{\mathcal{G}}_n^{(t)}\right)\right]-E_{\boldsymbol{\mathcal{D}}_n'\sim\mathcal{J}_{\mathcal{D}}}\left[\mathcal{F}_3\left(\boldsymbol{\mathcal{D}}_n'\right)\right]\tag{41}$$

Theoretically, if all the parameters are learned, its result should be the WSD. Accompanied with the fact that D estimates WSD, the G manages to minimize the estimated WSD. The objective of G is formulated as:

$$\min_G\sum_{n=1}^{\mathcal{N}}\mathcal{L}_G\tag{42}$$

where

$$\mathcal{L}_G=-\sum_{n=1}^{\mathcal{N}}E_{\boldsymbol{\mathcal{G}}_n^{(t)}\sim\mathcal{J}_{\mathcal{G}}}\left[\mathcal{F}_3\left(\boldsymbol{\mathcal{G}}_n^{(t)}\right)\right]\tag{43}$$

Considering that WSD-based GAL still suffers from estimation quality problem, a penalty item needs to be introduced into the final loss function. Interpolation sampling is utilized to randomly pick a pair of true samples $\nabla\left(\boldsymbol{\mathcal{D}}_n'\right)$ and generates samples $\nabla\left(\boldsymbol{\mathcal{G}}_n^{(t)}\right)$ from corresponding vectors, yielding:

$$\boldsymbol{\chi}_n^{(t)}=\lambda_2\cdot\nabla\left(\boldsymbol{\mathcal{G}}_n^{(t)}\right)+(1-\lambda_2)\cdot\nabla\left(\boldsymbol{\mathcal{D}}_n'\right)\tag{44}$$

where $\nabla\left(\boldsymbol{\mathcal{G}}_n^{(t)}\right)$ and $\nabla\left(\boldsymbol{\mathcal{D}}_n'\right)$ are sampled from $\boldsymbol{\mathcal{G}}_n^{(t)}$ and $\boldsymbol{\mathcal{D}}_n'$, and λ_2 is the trade-off parameter. Accordingly, the $\boldsymbol{\chi}_n^{(t)}$ is drawn from the following probabilistic distribution:

$$\mathcal{J}_\chi=\lambda_2\cdot\mathcal{J}_{\mathcal{G}}+(1-\lambda_2)\cdot\mathcal{J}_{\mathcal{D}}\tag{45}$$

Adding the $\boldsymbol{\chi}_n^{(t)}$ as adaptive penalty item, the total objective function can be established as:

$$\min_G\max_D\mathcal{L}\tag{46}$$

$$\mathcal{L}=\mathcal{L}_G-\mathcal{L}_D+\lambda_3\cdot E_{\chi_n^{(t)}\sim J_\chi}\left[\mathcal{F}_3\left(\boldsymbol{\chi}_n^{(t)}\right)\right]\tag{47}$$

where λ_3 is the penalty coefficient. Then, the Adam [36] is selected as learning method for above objective function to estimate the set of parameters, in which learning rate is denoted as e. The algorithmic process of GAL is illustrated in Algorithm 2.

C. Detection

Having estimated the set of parameters Θ, simulated factor $\boldsymbol{\mathcal{G}}_n^{(\mathcal{T})}$ and real factor $\boldsymbol{\mathcal{D}}_n'$ of GAL can be accordingly learned. They are respectively mapped into two vectors $\boldsymbol{f}_1$ and $\boldsymbol{f}_2$ through two multi-layer perception (MLP) networks [37]:

$$\boldsymbol{f}_1=MLP_1\left(\boldsymbol{\mathcal{G}}_n^{(\mathcal{T})}\right)\tag{48}$$

Algorithm 2 Generative Adversarial Learning

INPUT: $\mathcal{Q}_n$, $\mathcal{M}$, $\boldsymbol{\mathcal{V}}_{(n)}$, $\boldsymbol{\mathcal{R}}_{(n)}$, λ_2, λ_3, e
OUTPUT: $\boldsymbol{\mathcal{G}}_n^{(t)}$, $\boldsymbol{\mathcal{D}}'_n$, Θ
for $n = 1 \rightarrow \mathcal{N}$ **do**
  for $t = 1 \rightarrow |\mathcal{T}|$ **do**
    while G-steps **do**
      Update hidden state $\boldsymbol{\mathcal{H}}_n^{(t)}$ via GRU operator as Eq. (21) and (22)
      Define GRU operator from Eq. (23) to (27)
      Initialize $\boldsymbol{\mathcal{H}}_n^{(t)}$ at $t = 0$ as Eq. (28)
      Compute $\boldsymbol{\mathcal{G}}_n^{(t)}$ as Eq. (30) and (31)
    end while
    while D-steps **do**
      for $m = 1 \rightarrow \mathcal{M}$ **do**
        Update vector $\boldsymbol{\mathcal{V}}_{(n)}^{(m)}$ as Eq. (31) and (32)
        Update vector $\boldsymbol{\mathcal{R}}_{(n)}^{(m)}$ as Eq. (33) and (34)
        Compute $\boldsymbol{\mathcal{D}}'_n$ as Eq. (35)
      end for
      Draw $\boldsymbol{\mathcal{G}}_n^{(t)} \sim \mathcal{J}_G$ and $\boldsymbol{\mathcal{D}}'_n \sim \mathcal{J}_{\mathcal{D}}$
      Define WSD of discrete form and continuous form as Eq. (36) and Eq. 37
      Set up learning goal of D as Eq. (38) and (39)
    end while
    while training steps **do**
      Set up objective function of D and G as Eq. (40) to (43)
      Sample $\boldsymbol{\chi}_n^{(t)}$ from $\boldsymbol{\mathcal{G}}_n^{(t)}$ and $\boldsymbol{\mathcal{D}}'_n$ as Eq. (44), and draw $\boldsymbol{\chi}_n^{(t)} \sim J_\chi$ as Eq. (45)
      Set up total objective functions as Eq. (46)
      Update $\boldsymbol{\mathcal{G}}_\Theta^{(t)} \leftarrow \frac{1}{N} \cdot \sum_{n=1}^{N} \frac{\partial \mathcal{L}}{\partial \boldsymbol{\mathcal{G}}} \cdot \frac{\partial \boldsymbol{\mathcal{G}}_n^{(t)}}{\partial \Theta}$
      Update $\Theta \leftarrow \Theta - e \cdot Adam\left(\Theta, \boldsymbol{\mathcal{G}}_\Theta^{(t)}\right)$
    end while
  end for
end for

$$\boldsymbol{f}_2 = MLP_1\left(\boldsymbol{\mathcal{D}}'_n\right) \tag{49}$$

where $MLP_1(\cdot)$ and $MLP_2(\cdot)$ are the MLP network whose parameters are well known. In this research, the rumor detection is viewed as a 0-1 binary classification process, in which 0 denotes non-rumor and 1 denotes rumor. The final detection results can be output as:

$$\mathcal{O}_n = \sigma_3\left\{\frac{1}{y}\sum_{y}\left(\lambda_3 \cdot \boldsymbol{f}_1\right) \cdot \left[\left(1-\lambda_3\right) \cdot \boldsymbol{f}_2\right]^T\right\} \tag{50}$$

where y is the dimension of $\boldsymbol{f}_1$ and $\boldsymbol{f}_2$.

In all, all the notations involved in this paper are briefly introduced in TABLE I, TABLE II and TABLE III. All of the variables are divided into three types according to their roles: indices, parameters, and decision variables. The indices refer to those that are used to denote value ranges of some variables. The parameters refer to those that can be known directly based on initial datasets or manual settings. The decision variables refer to those that cannot be obtained directly and need to be estimated during the process of model training. TABLE I, TABLE II and TABLE III briefly illustrate descriptions for indices, process parameters, and decision variables, respectively.

TABLE IV: Metadata included in the experimental dataset

Attribute	***PHEME***	***Weibo***	***Twitter***
# of users	215217	2246374	24328
# of posts	198872	1415983	48621
# of events	901	2582	1676
# of true rumors	458	1280	842
# of false rumors	443	1302	834
average # of posts per event	221	870	29
maximum # of posts per event	2675	58350	289
minimum # of posts per event	63	12	15

TABLE V: Precision results on PHEME dataset

Algorithms	Different Sizes of Training Data			
	50%	60%	70%	80%
Random-LR	0.3275	0.3493	0.3668	0.3930
Random-MLP	0.3493	0.3384	0.3886	0.4017
C-Means	0.3384	0.3537	0.3755	0.3886
Twitter-LDA	0.3428	0.3755	0.3843	0.4061
HDP	0.3624	0.3646	0.4127	0.4192
BPI	0.3690	0.3908	0.4083	0.4323
GAN	0.3974	0.4258	0.4541	0.4672
Graph-GAN	**0.4301**	**0.4498**	**0.4760**	**0.4869**

IV. Experiments and Analysis

This section presents the detailed process for evaluating performance of the proposed Graph-GAN. Firstly, three real-world datasets that are commonly used for such purpose, are selected as the experimental scenarios. Secondly, parameter settings, baseline methods, and evaluation metrics of experiments are described separately. Thirdly, the obtained experimental results are displayed through various of figures and tables. And corresponding reasons for observed phenomenon are also analyzed.

A. Datasets

The simulative scenarios in this research are constructed with the aid of three publicly available benchmark datasets in this field. Among which, two of them are initial datasets named PHEME and Weibo respectively, and the third one is named Twitter in this research, which is the synthesis of two similar datasets named *Twitter 15* and *Twitter 16*. In terms of applicability, they are all nearly standard datasets that can be used for evaluation of general data mining problems in terms of social networks. Although there exists some distance between these datasets and realistic different scenes, they still act as proper approximation towards realistic scenarios. And the utilizing these datasets to assess ordinary social network analysis problems has been widely recognized by many researchers. Detailed information of these three datasets is described as follows:

No.	*Model*	*Description*
1	***Random-LR***	Labels are randomly generated, and logistic regression is trained for classification.
2	***Random-MLP***	Labels are randomly generated, and the MLP is trained for classification.
3	***C-Means***	Vectorized representation for the texts is extracted by the TF-IDF, and all of texts are clustered into two classes: rumor and non-rumor.
4	***Twitter-LDA***	It is a probabilistic method that utilizes Gibbs sampling to infer topic indicators.
5	***HDP***	It is a probabilistic method clusters events into two classes: rumor and non-rumor.
6	***BPI***	It is a probabilistic method that utilizes Bayesian theory to infer nature of events.
7	***GAN***	It is the initial GAN model without graph embedding.
8	***Graph-GAN***	It is the proposal that combines GAL with graph embedding.

Fig. 4: Summary of baseline methods and the proposed Graph-GAN.

TABLE VI: Recall results on PHEME dataset

Algorithms	Different Sizes of Training Data			
	50%	60%	70%	80%
Random-LR	0.3238	0.3472	0.3639	0.3947
Random-MLP	0.3503	0.3512	0.3885	0.4027
C-Means	0.3471	0.3564	0.3839	0.4002
Twitter-LDA	0.3492	0.3736	0.3824	0.4062
HDP	0.3747	0.3778	0.4138	0.4292
BPI	0.3653	0.3907	0.4227	0.4348
GAN	0.3982	0.4252	0.4591	0.4704
Graph-GAN	**0.4335**	**0.4475**	**0.4827**	**0.4996**

PHEME—It was firstly collected by Zubiaga *et al.* [38] from five breaking news. As for each news, it has many claims that are released by different users. For each claim, other users are allowed to express their opinions by making comments. Publisher of the claim is able to respond to comments, forming conversations among this user and reviewers. Generalized into our experiments, the claims and comments correspond to events and posts, respectively. Furthermore, events having less than 63 posts have been filtered out to avoid sparsity.

Weibo—It was firstly published by Ma *et al.* [39] and collected from the most popular Chinese social platform Sina Weibo . It has many topics which can be discussed by users through releasing speeches. As the response information is too sparse in this dataset, some response records are randomly generated to enrich the dataset. Generalized into this research, the topics and usersÃÂ¡ÃÂ¯ speeches correspond to events and posts, respectively. Furthermore, events having less than 12 posts have been filtered out to avoid sparsity.

Twitter—It was merged from two standard datasets whose initial names are *Twitter 15* and *Twitter 16*. The two datasets were published by Ma *et al.* [40] and were crawled from a most prevalent international social platform Twitter . Similar to the PHEME, users can express themselves and others are allowed to participate in discussions via giving comments. Besides, the two datasets also possess records of responses or reposts. In our experiments, claims and comments correspond to events and posts, respectively. Moreover, events having less than 15 posts have been filtered out to avoid sparsity.

Two users are assumed to have social relations if they have direct records of comments, responses or reposts. In each dataset, some meaningless comments or posts, such as stop words, have been removed. As for labels, events of PHEME and Weibo have been labeled as Rumor or Non-rumor. However, the Twitter has four types of labels: Non-rumor, True Rumor, False Rumor, and Unverified Rumor. Among them, the last two labels are viewed as uncertain labels. We hired some graduates from Chongqing Technology and Business University to verify nature for some events. They dealt with it by accessing Wikipedia or event information. The verification work lasted for about two months, and identified 102 true rumors and 90 false rumors. After preprocessing, the statistical characteristics of the three datasets are listed in TABLE IV.

B. Experimental Settings

As we are the first to investigate unsupervised rumor detection, direct baselines are not available. To highlight performance superiority of the proposed Graph-GAN, seven methods that can be used for the research problem are selected as indirect baselines. The first two methods try to add labels for each event, and two different supervised learning methods are utilized for rumor detection. The third method selects a famous text clustering method named C-Means. The next three methods are probabilistic inference-based approaches. The last method is the pure GAN model without graph embedding. They are described as follows:

Random-LR—It is a two-stage combination method. Firstly, labels are randomly generated for training samples.

TABLE VII: Precision results on Weibo dataset

Algorithms	Different Sizes of Training Data			
	50%	60%	70%	80%
Random-LR	0.3242	0.3383	0.3570	0.3766
Random-MLP	0.3281	0.3398	0.3664	0.3797
C-Means	0.3414	0.3625	0.3813	0.3883
Twitter-LDA	0.3500	0.3758	0.3883	0.3867
HDP	0.3484	0.3797	0.3867	0.4016
BPI	0.3742	0.3914	0.3953	0.4133
GAN	0.3859	0.4086	0.4250	0.4320
Graph-GAN	**0.3961**	**0.4227**	**0.4367**	**0.4461**

TABLE VIII: Recall results on Weibo dataset

Algorithms	Different Sizes of Training Data			
	50%	60%	70%	80%
Random-LR	0.3257	0.3341	0.3530	0.3762
Random-MLP	0.3253	0.3384	0.3650	0.3741
C-Means	0.3415	0.3622	0.3734	0.3896
Twitter-LDA	0.3498	0.3726	0.3890	0.3863
HDP	0.3451	0.3810	0.3869	0.3970
BPI	0.3688	0.3921	0.3951	0.4120
GAN	0.3811	0.4067	0.4204	0.4266
Graph-GAN	**0.3992**	**0.4216**	**0.4358**	**0.4441**

TABLE IX: Precision results on Twitter dataset

Algorithms	Different Sizes of Training Data			
	50%	60%	70%	80%
Random-LR	0.3824	0.4109	0.4216	0.4252
Random-MLP	0.3967	0.4240	0.4430	0.4442
C-Means	0.4097	0.4204	0.4347	0.4513
Twitter-LDA	0.4371	0.4430	0.4525	0.4656
HDP	0.4228	0.4418	0.4572	0.4739
BPI	0.4145	0.4644	0.4679	0.4822
GAN	0.3859	0.4086	0.4250	0.4320
Graph-GAN	**0.4656**	**0.4893**	**0.4941**	**0.5024**

TABLE X: Recall results on Twitter dataset

Algorithms	Different Sizes of Training Data			
	50%	60%	70%	80%
Random-LR	0.3788	0.4057	0.4213	0.4326
Random-MLP	0.3953	0.4247	0.4422	0.4413
C-Means	0.4169	0.4216	0.4424	0.4605
Twitter-LDA	0.4426	0.4433	0.4589	0.4699
HDP	0.4266	0.4464	0.4626	0.4807
BPI	0.4269	0.4625	0.4706	0.4915
GAN	0.4432	0.4791	0.4874	0.4988
Graph-GAN	**0.4604**	**0.4881**	**0.4988**	**0.5073**

Then, the logistic regression model is trained to classify rumors from all the events. After generation of labels, this rumor detection is actually transformed into a supervised learning process.

Random-MLP—Similarly, the unsupervised situation is transformed into a supervised learning issue. It is a two-stage combination method. Different from the Random-LR, the logistic regression is replaced as an MLP network.

C-Means [41]—It is a famous clustering method that can be used for numerical data and texts. At the beginning, vectorized representation for the texts is extracted by the TF-IDF. Top-100 high frequency words are selected as the word dictionary of TF-IDF. All the events are finally divided into two clusters corresponding to rumor events and non-rumor events.

Twitter-LDA [42]–It is a classical topic model that can be used to identify a topic indicator of each piece of text. It assigns prior distributions for generative processes of texts, and realizes text classification via Gibbs sampling-based posterior probabilistic inference. In effect, all the events are finally divided into two clusters corresponding to rumor events and non-rumor events.

HDP [43]—It refers to the hierarchical Dirichlet process (HDP) model which is also a probabilistic generative model. It is developed to cluster similar samples together, so that different categories can be distinguished. When the number of classes is set to two, all the events can be divided into two clusters corresponding to rumor events and non-rumor events.

BPI—It refers to the Bayesian probabilistic inference (BPI), a method uniquely proposed by us. It originates from the inference model in the study [44]. It assumes that the labels are drawn from some prior distributions which are further drawn from some hyper parameters. And all the parameters can be learned while training.

GAN [45]—It is the initial generative adversarial network (GAN) model without graph embedding. At the beginning, all the events are transformed into feature vectors with the aid of TF-IDF. Its procedures of generation process and discrimination process are similar to corresponding parts in Graph-GAN.

And the seven selected baselines and the proposed Graph-GAN are briefly described in Fig. 4.

To quantify performance of the proposed Graph-GAN, several evaluation metrics are introduced for measurement. In such a binary classification situation, a positive event sample indicates that it is labeled as rumor, and a negative event sample indicates that it is labeled as non-rumor. The scenarios where positive samples and negative samples are correctly discriminated, are defined as true positive (TP) and true negative (TN), respectively. Similarly, scenarios where positive samples and negative samples are incorrectly discriminated, are defined as false positive (FP) and false negative (FN), respectively. Above definitions lead to four metrics: precision,

recall, accuracy, and F-score, which are defined as:

$$Precision = \frac{\zeta(TP)}{\zeta(TP) + \zeta(FP)} \tag{51}$$

$$Recall = \frac{\zeta(TP)}{\zeta(TP) + \zeta(FN)} \tag{52}$$

$$Accuracy = \frac{\zeta(TP) + \zeta(TN)}{\zeta(TP) + \zeta(FP) + \zeta(TN) + \zeta(FN)} \tag{53}$$

$$F - score = \frac{2 \cdot Precision \cdot Recall}{Precision + Recall} \tag{54}$$

where $\zeta(x)$ counts the number of x.

All the experiments are carried out in a deep learning working station with 28-core CPU, 256-GB RAM and a GPU (RTX-2080-Ti). The proposed Deep-PR is implemented with the assistance of TensorFlow [1]. Because all the texts in Weibo dataset are Chinese texts, the tool jieba is employed for word segmentation and stop words are removed. In Algorithm 1, the k in Eq. (2) is set to 5, θ in Eq. (3) is set to 6, Φ in Eq. (6) and Ψ in Eq. (3) are both set to 8, and λ_1 in Eq. (13) is set to 0.5. In Algorithm 2, the λ_2 in Eq. (45), $\mathcal{M}$ is set to 10, λ_3 in Eq. (50) is set to 0.3, and the number of iterative rounds $\mathcal{T}$ is set to 20. Its learning rate is initially set to 0.001 and will be changed multiple times during experiments. Parameters in baselines are set to their default values, and are left out here due to the limitation of textual length. Size of training data is set to 70% in default and it is fluctuating in experiments.

C. Results and Analysis

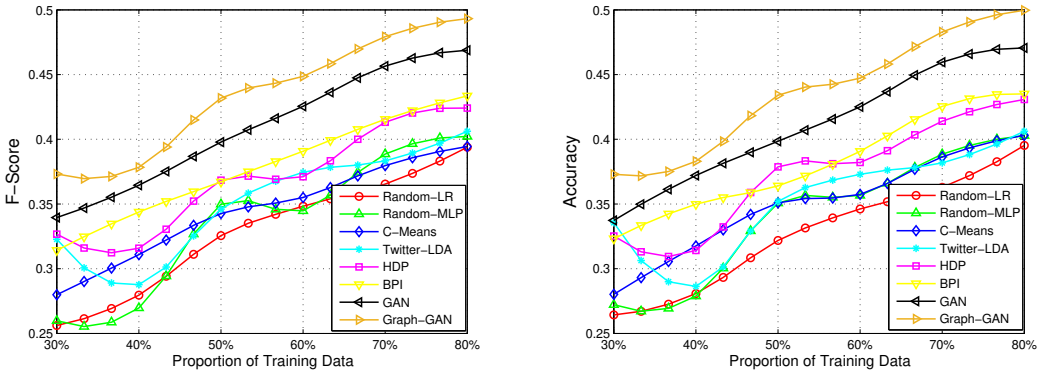
1) Performance Efficiency: The obtained precision and recall results on three datasets are listed from TABLE V to TABLE X. From the view of overall trend, almost all of the methods perform better when the proportions of training data is increasing. When the proportion reaches 60%, the ascending trend tends to be gentle. In terms of performance comparison, the first two methods are relatively weaker and followed by the middle four, while the last two GAN-based methods are stronger. Of all the methods, the proposed Graph-GAN obtains the best performance. We specialize two representative situations to illustrate this view. As for precision results on Twitter dataset, it is about 18% better than Random-LR, 16% better than Random-MLP, 12% better than C-Means, 9% better than Twitter-LDA, 11% than HDP, 8% better than BPI, and 5% better than GAN. And for recall results on Weibo dataset, it is about 22% better than Random-LR, 19% better than Random-MLP, 15% better than C-Means, 12% better than Twitter-LDA, 13% better than HDP, 9% better than BPI, and 4% better than GAN. The achievements of above observations can be attributed to two possible reasons. Firstly, the Graph-GAN constructs a more fine-grained feature space from the perspective of graph learning. Then, the Graph-GAN overcomes the barrier of label unavailability through an adversarial learning-based adaptive optimization process. The successive effect of the two parts contributes to a more precise detector under unsupervised scenarios.

[1]http://tensorflow.google.cn/

The F-Score and accuracy results of three datasets are demonstrated in Fig. 5, Fig. 6 and Fig. 7. Each of them has two subfigures, in which X-axis denotes proportion of training data and Y-axis denotes values of evaluation metrics. The overall performance trends are similar to precision and results where two GAN-related methods perform more excellently than other probabilistic inference-based models. Performance of these methods tends to be better with risen proportion of training data.When the proportion of training data is risen, the performance advantage of Graph-GAN is more remarkable. It is evident that the Graph-GAN needs sufficient training samples to learn a quite comprehensive feature space. We also give two examples to specialize proper performance of the Graph-GAN. As for F-Score on Twitter dataset, it is about 19% better than Random-LR, 14% better than Random-MLP, 15% better than C-Means, 12% better than Twitter-LDA, 11% better than HDP, 7% better than BPI, and 5% better than GAN. And for accuracy on PHEME dataset, it is 25% better Random-LR, 21% better than Random-MLP, 19% better than C-Means, 16% better than Twitter-LDA, 13% better than HDP, 9% better than BPI, and 7% better than GAN. Two possible explanations may be deduced for above phenomenons. For one thing, the idea of GAL is more robust than general unsupervised learning methods because it improves the understanding of knowledge during constant adversarial training inside itself. For another, the graph embedding is a more fine-grained feature abstraction scheme compared with general ones, hence, a more comprehensive feature space also ensures model performance.

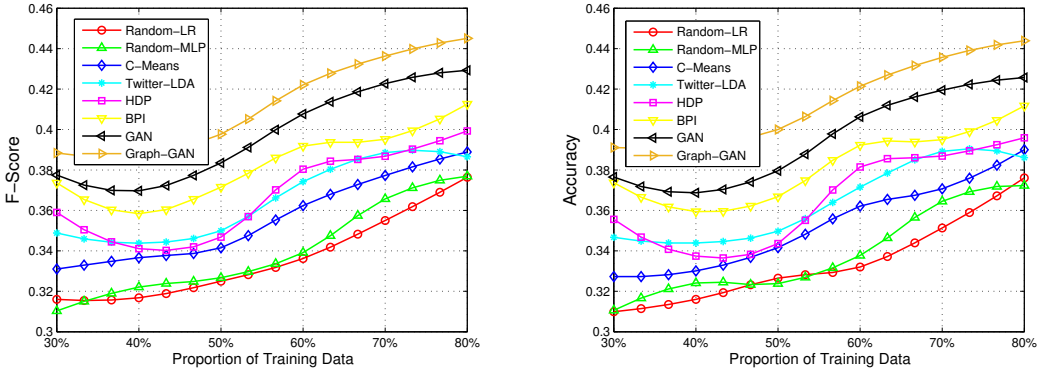
To sum up, both of two above aspects jointly contribute to the promotion the Graph-GAN. Note that results of the four metrics on the Weibo dataset are worse than other two datasets. A possible explanation for this lies in that semantic modeling part of Graph-GAN is more suitable for English texts and Chinese texts, as effect of word segmentation is likely to bring about some influence. But no matter how the experimental scenarios change, the proposed Graph-GAN always performs better than baselines with respect to precision and recall.

2) Parameter Sensitivity: Having evaluated efficiency of the proposed Graph-GAN, another set of experiments are conducted to explore its stability by testing its sensitivity to parameter changes. In this group of experiments, only performance fluctuation of Graph-GAN itself under different parameter settings is visualized, without comparing with baselines. In detail, performance tendency in terms of four metrics is analyzed with the changing of two groups of parameters: learning rate and proportion of training data. Considering the relatively weak performance of these methods when proportion of training data is too small, only four proportions are selected: 50%, 60%, 70% and 80%. And learning rate is also set to three values: 0.001, 0.002 and 0.003. To this end, parameter sensitivity results on three datasets are illustrated in Fig. 8, Fig. 9 and Fig. 10. Each of them contains four subfigures, in which X-axis denotes different learning rates and Y-axis denotes different proportions of training data. The smaller the chromaticity difference among blocks, the less the performance of the algorithm is affected by the change of parameters. It can be observed from these figures that performance generally tends to become quite stable when the proportion of training



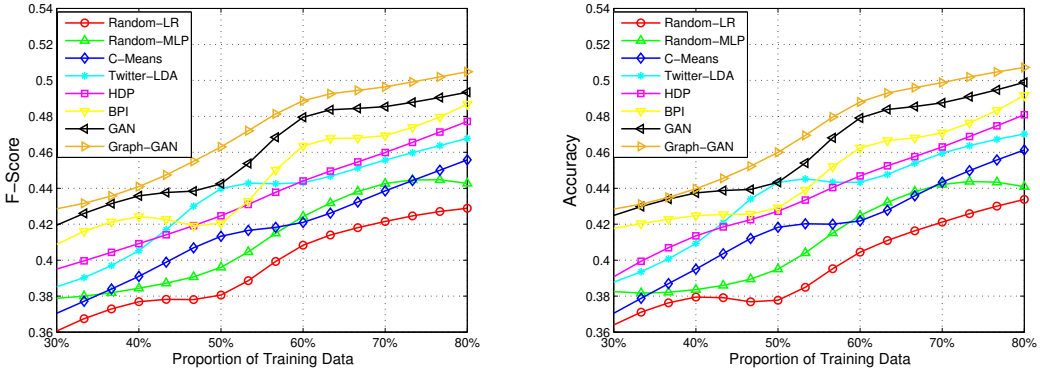
(a) F-score (b) Accuracy

Fig. 5: F-Score and accuracy results on PHEME dataset



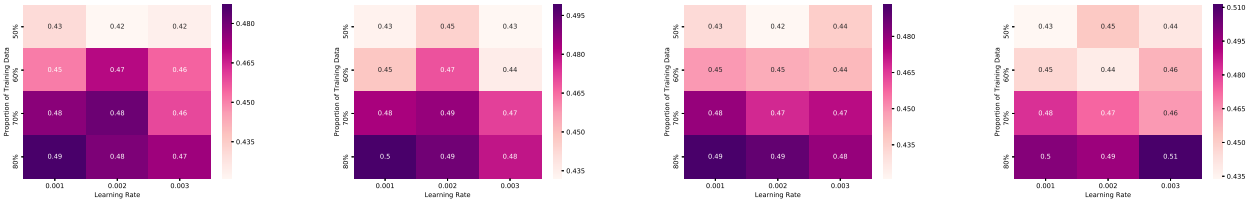
(a) F-score (b) Accuracy

Fig. 6: F-Score and accuracy results on Weibo dataset



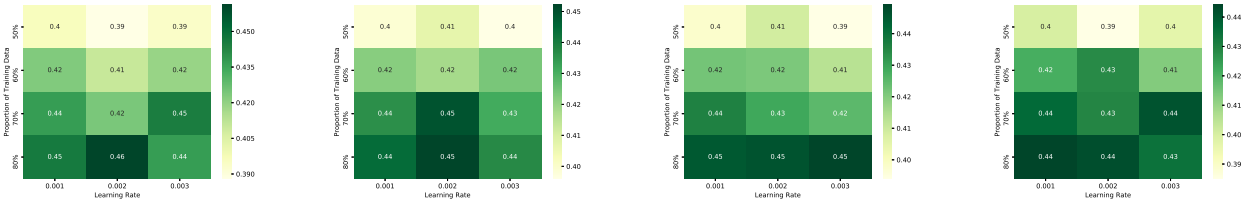
(a) F-score (b) Accuracy

Fig. 7: F-Score and accuracy results on Twitter dataset



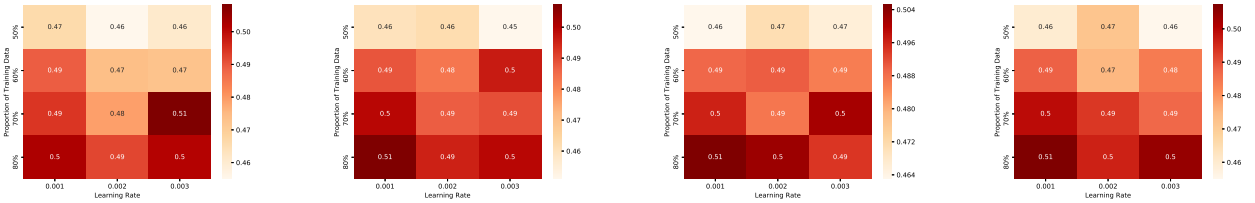
(a) Precision (b) Recall (c) F-score (d) Accuracy

Fig. 8: Parameter sensitivity results with respect to four metrics on PHEME dataset.



(a) Precision (b) Recall (c) F-score (d) Accuracy

Fig. 9: Parameter sensitivity results with respect to four metrics on Weibo dataset.



(a) Precision (b) Recall (c) F-score (d) Accuracy

Fig. 10: Parameter sensitivity results with respect to four metrics on Twitter dataset.

data reaches 60%, especially on Twitter dataset. We analyze all the results and summarize two possible reasons for the phenomenons. Firstly, as feature extraction part undergoes multiple layers of neural computation procedures, the obtained feature space has been quite robust. Secondly, the GAL tries to tackle with deviation issue of initial values by active gaming learning. Due to the two reasons, the proposed Graph-GAN is not susceptible to parameter changes. Only within PHEME dataset, the change of F-Score and accuracy results is a little obvious when the proportion of training set is switched from 60% to 70%. It may be caused by the fact that Graph-GAN is still not well trained in some scenarios where training samples are not sufficient enough. Thus, it never influences the stability assessment of the Graph-GAN.

Based on the results and analysis concerning above two groups of experiments, the proposed Graph-GAN is a both efficient and robust fuzzy detection system for rumor events with respect to unsupervised situations.

V. Conclusions

The rumor spreading gradually become a major security threat in cyberspace. In consequence, deep learning-based fuzzy detection for rumors has attracted much research attention in recent years. Although much progress has been achieved, almost all of existing studies focused on supervised scenarios where expert samples with labels are available. As far as we are concerned, fuzzy detection for rumors under unsupervised scenarios, have not been noticed. In fact, they are quite common as the annotation of rumor samples is expensive and complicated. To deal with such challenge, this paper proposes a fuzzy detection system for rumors named as Graph-GAN. It employs the idea of GAL to construct an adaptive classifier and sets up graph-level feature spaces to further improve robustness. Such design is able to tackle with the scenarios where labels are absent for training.

And two groups of experiments are implemented on three real-world datasets to assess the proposed Graph-GAN. Seven methods are selected as baselines and four typical metrics are adopted for evaluation. Experimental results show that the proposal improves performance about 5% to 10% compared with baseline methods, and that it possesses proper robustness. Therefore, the proposed solution not only realizes adaptive fuzzy detection for rumor events under unsupervised situations, but also possesses proper efficiency as well as stability. This research distinguishes itself from others by constructing

novel rumor detection models without prior expertise labels, so as to realize adaptive rumor detection. It is also noted that the proposed Graph-GAN is not a completely adaptive detection method, because the neural network structures inside it are empirically set. The complete adaption needs to possess the ability to automatically modify network structures to fit for different types of social contexts. The transfer learning may provide some promising ideas for this point, which is certainly the future working direction of our research team.

REFERENCES

[1] Z. Guo and H. Wang, "A Deep Graph Neural Network-based Mechanism for Social Recommendations," *IEEE Trans. Ind. Informatics*, doi: 10.1109/TII.2020.2986316.

[2] S. Kuter, G. W. Weber, and Z. Akyürek, "A progressive approach for processing satellite data by operational research," *Oper. Res.*, vol. 17, no. 2, pp. 371-393, 2017.

[3] R. Lotfi, Y. Z. Mehrjerdi, M. S. Pishvaee, A. Sadeghieh and G. W. Weber, "A robust optimization model for sustainable and resilient closed-loop supply chain network design considering conditional value at risk," *Numerical Algebra, Control & Optimization*, doi: 10.3934/naco.2020023.

[4] K. Yu et al., "A Key Management Scheme for Secure Communications of Information Centric Advanced Metering Infrastructure in Smart Grid," *IEEE Trans. Instrum. Meas.*, vol. 64, no. 8, pp. 2072-2085, 2015.

[5] J. Grobelny, R. Michalski, and G. W. Weber, "Modeling human thinking about similarities by neuromatrices in the perspective of fuzzy logic," *Neural Comput. Appl.*, doi: 10.1007/s00521-020-05363-y.

[6] X. Zhou, Y. Hu, W. Liang, J. Ma, and Q. Jin, "Variational LSTM Enhanced Anomaly Detection for Industrial Big Data," *IEEE Trans. Ind. Informatics*, Sep. 2020, doi: 10.1109/TII.2020.3022432.

[7] A. Jolfaei et al., "Guest Editorial Special Issue on Privacy and Security in Distributed Edge Computing and Evolving IoT," *IEEE Internet Things J.*, vol. 7, no. 4, pp. 2496-2500, 2020.

[8] G. W. Weber, İ. Batmaz, G. Gülser, P. Taylan, and F. Yerlikaya-Özkurt, "CMARS: a new contribution to nonparametric regression with multivariate adaptive regression splines supported by continuous optimization," *Inverse Problems in Science and Engineering*, vol. 20, no. 3, pp. 371-400, 2012.

[9] Y. Z. Mehrjerdi and R. Lotfi, "Development of a Mathematical Model for Sustainable Closed-loop Supply Chain with Efficiency and Resilience Systematic Framework," *International Journal of Supply and Operations Management*, vol. 6, no. 4, pp. 360-388, 2019.

[10] S. Kuter, Z. Akyurek, and G. W. Weber, "Retrieval of fractional snow covered area from MODIS data by multivariate adaptive regression splines," *Remote Sensing of Environment*, vol. 2015, pp. 236-252, 2018.

[11] K. Yu, L. Lin, M. Alazab, L. Tan, B. Gu, "Deep Learning-Based Traffic Safety Solution for a Mixture of Autonomous and Manual Vehicles in a 5G-Enabled Intelligent Transportation System", *IEEE Trans. Intell. Transp. Syst.*, doi: 10.1109/TITS.2020.3042504.

[12] J. P. Chávez, B. Gürbüz and C. M. A. Pinto, "The Effect of Aggressive Chemotherapy in a Model for HIV/AIDS-cancer Dynamics," *Commun. Nonlinear Sci. Numer. Simul.*, vol. 75, pp. 109-120, 2019.

[13] Z. Guo et al., "Robust Spammer Detection Using Collaborative Neural Network in Internet of Thing Applications," in *IEEE Internet Things J.*, doi: 10.1109/JIOT.2020.3003802.

[14] T. Bian et al., "Rumor Detection on Social Media with Bi-Directional Graph Convolutional Networks," in *Proc. of the Thirty-Fourth AAAI Conference on Artificial Intelligence*, New York, NY, USA, 2020, pp. 549-556.

[15] A. E. Fard et al., "Computational Rumor Detection Without Non-Rumor: A One-Class Classification Approach," *IEEE Trans. Comput. Soc. Syst.*, vol. 6, no. 5, pp. 830-846, 2019.

[16] Q. Li, Q. Zhang and L. Si, "Rumor Detection by Exploiting User Credibility Information, Attention and Multi-task Learning," in *Proc. of the 57th Annual Meeting of the Association for Computational Linguistics*, Florence, Italy, 2019, pp. 1173-1179.

[17] Z. Guo, L. Tang, T. Guo, K. Yu, M. Alazab, A. Shalaginov, "Deep Graph neural network-based spammer detection under the perspective of heterogeneous cyberspace", Future Gener. Comput. Syst., vol. 117, pp. 205-218, 2021.

[18] R. Lotfi, Z. Yadegari, S. H. Hosseini, A. H. Khameneh, E. B. Tirkolaee, and G. W. Weber, "A Robust Time-Cost-Quality-Energy-Environment Trade-off with Resource-Constrained in Project Management: A Case Study for a Bridge Construction Project," *Journal of Industrial & Management Optimization*, vol. 13, no. 5, pp. 1-22, 2017.

[19] E. B. Tirkolaee, P. Abbasian, G. W. Weber, "Sustainable fuzzy multi-trip location-routing problem for medical waste management during the COVID-19 outbreak," *Science of The Total Environment*, doi: 10.1016/j.scitotenv.2020.143607

[20] D. Lin et al., "Chinese Microblog Rumor Detection Based on Deep Sequence Context," *Concurr. Comput. Pract. Exp.*, vol. 31, no. 23, pp. 1-12, 2019.

[21] S. Belen, E. Kropat and G. W. Weber, "On the classical Maki-Thompson rumour model in continuous time," *Cent. Eur. J. Oper. Res.* vol. 19, pp. 1Â¨C17, 2011.

[22] M. Bugueño, G. Sepulveda and M. Mendoza, "An Empirical Analysis of Rumor Detection on Microblogs with Recurrent Neural Networks," in *Proc. of 21st International Conference on Human-Computer Interaction*, Orlando, FL, USA, 2019, pp. 293-310.

[23] Z. Wang et al., "Research on Microblog Rumor Events Detection via Dynamic Time Series Based GRU Model," in *Proc. of the 2019 IEEE International Conference on Communications*, Shanghai, China, 2019, pp. 1-6.

[24] X. Zhou, W. Liang, K. Wang, R. Huang, and Q. Jin, "Academic Influence Aware and Multidimensional Network Analysis for Research Collaboration Navigation Based on Scholarly Big Data," *IEEE Transactions on Emerging Topics in Computing*, 2018, doi: 10.1109/TETC.2018.2860051

[25] E. B. Tirkolaee; A. Goli; G. W. Weber, "Fuzzy Mathematical Programming and Self-Adaptive Artificial Fish Swarm Algorithm for Just-in-Time Energy-Aware Flow Shop Scheduling Problem With Outsourcing Option," *IEEE Trans. Fuzzy Syst.*, vol. 28, no. 11, pp. 272-2783, 2020.

[26] Q. Huang et al., "Deep Structure Learning for Rumor Detection on Twitter," in *Proc. of the 2019 International Joint Conference on Neural Networks*, Budapest, Hungary, 2019, pp. 1-8.

[27] Y. Chen, L. Hu and J. Sui, "Text-Based Fusion Neural Network for Rumor Detection," in *Proc. of the 12th International Conference on Knowledge Science, Engineering and Management*, Athens, Greece, 2019, pp. 105-109.

[28] H. Zhang et al., "Multi-modal Knowledge-aware Event Memory Network for Social Media Rumor Detection," in *Proc. of the 27th ACM International Conference on Multimedia*, New York, NY, USA, 2019, pp. 1942-1951.

[29] E. B. Tirkolaee, A. Mardani, Z. Dashtian, M. Soltani, and G. W. Weber, "A novel hybrid method using fuzzy decision making and multi-objective programming for sustainable-reliable supplier selection in two-echelon supply chain design," *Journal of Cleaner Production*, vol. 250, 119517, 2020. doi: 10.1016/j.jclepro.2019.119517.

[30] Z. Wu et al., "hPSD: A Hybrid PU-Learning-Based Spammer Detection Model for Product Reviews," *IEEE Trans. Cybern.*, vol. 50, no. 4, pp. 1595-1606, 2020.

[31] X. Chen et al., "One-Shot Generative Adversarial Learning for MRI Segmentation of Craniomaxillofacial Bony Structures," *IEEE Trans. Med. Imaging*, vol. 39, no. 3, pp. 787-796, 2020.

[32] S. Qiu et al., "Referring Image Segmentation by Generative Adversarial Learning," *IEEE Trans. Multimedia*, vol. 22, no. 5, pp. 1333-1344, 2020.

[33] Y. Wang et al., "A Heterogeneous Graph Embedding Framework for Location-Based Social Network Analysis in Smart Cities," *IEEE Trans. Ind. Informatics*, vol. 16, no. 4, pp. 2747-2755, 2020.

[34] Y. Liu, Y. Liu and L. Ding, "Scene Classification by Coupling Convolutional Neural Networks With Wasserstein Distance," *IEEE Geosci. Remote. Sens. Lett.*, vol. 16, no. 5, pp. 722-726, 2019.

[35] H. Yang, J. Guo and J. Jung, "Schwartz Duality of the Dirac Delta Function for the Chebyshev Collocation Approximation to the Fractional Advection Equation," *Appl. Math. Lett.*, vol. 64, pp. 205-212, 2017.

[36] S. Bock and M. Weiß, "A Proof of Local Convergence for the Adam Optimizer," in *Proc. of the 2019 International Joint Conference on Neural Networks*, Budapest, Hungary, 2019, pp. 1-8.

[37] B. Gürbüz, M. Sezer and C. Güler, "Laguerre Collocation Method for Solving Fredholm Integro-Differential Equations with Functional Arguments," *J. Appl. Math.*, vol. 2014, pp. 682398:1–682398:12, 2014.

[38] A. Zubiaga, M. Liakata and R. Procter, "Exploiting Context for Rumour Detection in Social Media," in *Proc. of the 9th International Conference of Social Informatics*, Oxford, UK, 2017, pp. 109-123.

[39] J. Ma et al., "Detecting Rumors from Microblogs with Recurrent Neural Networks," in *Proc. of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, New York, NY, USA, 2016, pp. 3818-3824.

[40] J. Ma et al., "Detect Rumors in Microblog Posts Using Propagation Structure via Kernel Learning," in *Proc. of the 55th Annual Meeting of the Association for Computational Linguistics*, Vancouver, Canada, 2017, pp. 708-717.
[41] W. L. Chang, K. M. Tay and C. P. Lim, "Enhancing an Evolving Tree-based text document visualization model with Fuzzy c-Means clustering," in *Proc. of 2013 IEEE International Conference on Fuzzy Systems*, Hyderabad, India, 2013, pp. 1-6.
[42] M. C. Yang and H. C. Rim, "Identifying interesting Twitter contents using topical analysis," *Expert Syst. Appl.*, vol. 41, no. 9, pp. 4330-4336, 2014.
[43] A. Karlsson et al., "Evaluation of the Dirichlet Process Multinomial Mixture Model for Short-Text Topic Modeling," in *Proc. of 2018 6th International Symposium on Computational and Business Intelligence*, Basel, Switzerland, 2018, pp. 79-83.
[44] B. Gürbüz and M. Sezer, "Laguerre Polynomial Approach for Solving Lane-Emden Type Functional Differential Equations," *Appl. Math. Comput.*, vol. 242, pp. 255-264, 2014.
[45] T. Wang et al., "Joint Character-Level Convolutional and Generative Adversarial Networks for Text Classification," *Complexity*, vol. 2020, pp. 1-11, 2020.

Zhiwei Guo (Member, IEEE) received the B.E. degree in Communications Engineering from Zhengzhou University, Zhengzhou, China, in 2013, and the Ph.D. degree in Communications and Information Systems from Chongqing University, Chongqing, China, in 2018. He was a lecturer with Chongqing Technology and Business University, China, from 2018 to 2020. Since the year of 2021, he has been working as an Associate Professor with Chongqing Technology and Business University, Chongqing, China.

Dr. Guo has hosted and participated in more than ten projects, and is currently involved in some national projects of China, such as National Key Research and Development Program of China, and National Natural Science Foundation of China. He has authored more than 20 publications including papers in prestigious journal/conferences such as the IEEE Transactions on Industrial Informatics, IEEE Internet of Things Journal, IEEE Transactions on Fussy Systems, IEEE Transactions on Network Science and Engineering, Future Generation Computer Systems, Neural Computing and Applications, IET Electronics Letters, IEEE GLOBECOM, etc.

In 2017, he gave an oral presentation in the International Joint Conference on Artificial Intelligence (IJCAI 2017). His research interests focus on data mining and pattern recognition.

Keping Yu (Senior Member, IEEE) received the M.E. and Ph.D. degrees from the Graduate School of Global Information and Telecommunication Studies, Waseda University, Tokyo, Japan, in 2012 and 2016, respectively. He was a Research Associate and a Junior Researcher with the Global Information and Telecommunication Institute, Waseda University, from 2015 to 2019 and 2019 to 2020, respectively, where he is currently a Researcher.

Dr. Yu has hosted and participated in more than ten projects, is involved in many standardization activities organized by ITU-T and ICNRG of IRTF, and has contributed to ITU-T Standards Y.3071 and Supplement 35. He received the Best Paper Award from ITU Kaleidoscope 2020, the Student Presentation Award from JSST 2014. He has authored 100+ publications including papers in prestigious journal/conferences such as the IEEE NetMag, CEM, T-ITS, TVT, TNSE, IoTJ, TII, GLOBECOM, etc. He is an Editor of IEEE Open Journal of Vehicular Technology (OJVT). He has been a Lead Guest Editor for Sensors, Peer-to-Peer Networking and Applications, Energies and Guest Editor for IEICE Transactions on Information and Systems, Computer Communications, IET Intelligent Transport Systems, Wireless Communications and Mobile Computing. He served as general co-chair and publicity co-chair of the IEEE VTC2020-Spring 1st EBTSRA workshop, general co-chair of IEEE ICCC2020 2nd EBTSRA workshop, session chair of IEEE ICCC2020, TPC co-chair of SCML2020, local chair of MONAMI 2020, Session Co-chair of CcS2020, and session chair of ITU Kaleidoscope 2016. His research interests include smart grids, information-centric networking, the Internet of Things, artificial intelligence, blockchain, and information security. He is a Member of the IEEE.

Alireza Jolfaei (Senior Member, IEEE) received the Ph.D. degree in applied cryptography from Griffith University, Gold Coast, Southport, QLD, Australia, in 2015.

He is currently a Lecturer with the Department of Computing, Macquarie University, Macquarie Park, NSW, Australia. Before this appointment, he was an Assistant Professor with Federation University Australia, Ballarat, VIC, USA and Temple University, Philadelphia, PA, USA. He has authored more than 60 peer-reviewed articles on topics related to cybersecurity. His current research areas include cybersecurity, Internet of Things security, human-in-the-loop CPS security, cryptography, AI and machine learning for cybersecurity.

Dr. Jolfaei was the recipient of multiple awards for Academic Excellence, University Contribution, and Inclusion and Diversity Support, the prestigious IEEE Australian council award for his research paper published in the IEEE Transactions on Information Forensics and Security, and a recognition diploma with a cash award from the IEEE Industrial Electronics Society for his publication at the 2019 IEEE IES International Conference on Industrial Technology. He is a founding member of the Federation University IEEE Student Branch. He was the Chairman of the Computational Intelligence Society in the IEEE Victoria Section and also as the Chairman of Professional and Career Activities for the IEEE Queensland Section. He was the Guest Associate Editor for the IEEE journals and transactions, including the IEEE Internet of Things Journal, IEEE Sensors Journal, IEEE Transactions on Industrial Applications, IEEE Transactions on Intelligent Transportation Systems, and IEEE Transactions on Emerging Topics in Computational Intelligence. He has served more than ten conferences in leadership capacities including program Co-Chair, track Chair, session Chair, and Technical Program Committee member, including IEEE International Conference on Trust, Security and Privacy in Computing and Communications and IEEE Conference on Computer Communications. He is a Senior Member of an ACM Distinguished Speaker on the topic of Cyber-Physical Systems Security.

Ali Kashif Bashir (Senior Member, IEEE) is a Senior Lecturer at the Department of Computing and Mathematics, Manchester Metropolitan University, United Kingdom. He is also holding Adjunct Professor Position at National University of Science and Technology, Pakistan. He is a senior member of IEEE, invited member of IEEE Industrial Electronic Society, member of ACM, and Distinguished Speaker of ACM. His past assignments include Associate Professor of ICT, University of the Faroe Islands, Denmark; Osaka University, Japan; Nara National College of Technology, Japan; the National Fusion Research Institute, South Korea; Southern Power Company Ltd., South Korea, and the Seoul Metropolitan Government, South Korea. He has worked on several research and industrial projects of South Korean, Japanese and European agencies and Government Ministries.

He received his Ph.D. in computer science and engineering from Korea University South Korea. He has authored over 140 research articles and is supervising/co-supervising several graduate (MS and PhD) students. His research interests include internet of things, wireless networks, distributed systems, network/cyber security, cloud/network function virtualization, machine learning, etc. He is serving as the Editor-in-chief of the IEEE FUTURE DIRECTIONS NEWSLETTER. He is also serving as area editor of KSII Transactions on Internet and Information Systems; associate editor of IEEE Access, IET Quantum Computing. He is leading many conferences as a chair (program, publicity, and track) and had organized workshops in flagship conferences like IEEE Infocom, IEEE Globecom, IEEE Mobicom, etc.

Alaa Omran Almagrabi received the B.Sc. degree in computer science from Jeddah Teacher's College, Jeddah, Saudi Arabia, in 2003, the master's degree in information technology and the Ph.D. degree in computer science, both from La Trobe University, Melbourne, VIC, Australia, in 2009 and 2014, respectively. In 2014, he was an Assistant Professor of Computer Science and Information Technology with the Department of Information System, University of King Abdulaziz, Jeddah, where he became an Associate Professor in 2019. His research interests include pervasive computing, networking, data mining, system analysis and design, and ontology domains.

Neeraj Kumar (Senior Member, IEEE) received the Ph.D. degree in computer science engineering from Shri Mata Vaishno Devi University, Katra, India, in 2009. He was a Postdoctoral Research Fellow with Coventry University, Coventry, U.K. He is currently an Associate Professor with the Department of Computer Science and Engineering, Thapar University, Patiala, India. He is a Visiting Professor with Coventry University, Coventry, U.K. He has authored or coauthored more than 300 technical research papers in leading journals and conferences from IEEE, Elsevier, Springer, Wiley, and others. Some of his research findings are published in top-cited journals such as the IEEE Transactions on Industrial Electronics, the IEEE Transactions on Dependable and Secure Computing, the IEEE Transactions on Intelligent Transportation Systems, the IEEE Transactions on Cloud Computing, the IEEE Transactions on Knowledge and Data Engineering, the IEEE Transactions on Vehicular Technology, the IEEE Transactions on Consumer Electronics, the IEEE Network, the IEEE Transactions on Communications, the IEEE Transactions on Wireless Communications, the IEEE Internet of Things Journal, the IEEE Systems Journal, the Future Generation Computer Systems, the Journal of Network and Computer Applications, and the Computer Communications. He has guided many Ph.D. and M.E./M.Tech. students. His research was supported by fundings from Tata Consultancy Service, the Council of Scientific and Industrial Research, and the Department of Science and Technology. He is leading the research group ¡°Sustainable Practices for Internet of Energy and Security,¡± where group members are working on the latest cutting-edge technologies.

Dr. Kumar was the recipient of the Best Research Paper awards from the 2018 IEEE International Conference on Communications and the IEEE Systems Journal in 2018. He is a Technical Program Committee Member and Reviewer of many international conferences across the globe. He is on the editorial board of the Journal of Network and Computer Applications, the IEEE Communication Magazine, the International Journal of Communication Systems, and Security and Privacy.