Markov-Driven Graph Convolutional Networks for Social Spammer Detection

Leyan Deng, Chenwang Wu, Defu Lian, Yongji Wu, and Enhong Chen, *Senior Member, IEEE*

Abstract—With the growing popularity of social media, malicious users (spammers) unfairly overpower legitimate users with unwanted or fake content to achieve their illegal purposes, which encourages research on spammer detection. The existing spammer detection methods can be characterized into feature-based detection and propagation-based detection. However, feature-based methods (e.g., GCN) cannot capture the user's following relations, while propagation-based methods cannot utilize the rich text features. To this end, we consider combining these two methods and propose an Adaptive Reward Markov Random Field (ARMRF) layer. ARMRF layer models three intuitions on user label relations and assign them different learnable rewards. Besides, we learn the reward weights by stacking the ARMRF layer on top of GCN for end-to-end training, and we call the stacked model ARMGCN. To further improve the expressive power of ARMGCN, we propose the Markov-Driven Graph Convolutional Network (MDGCN), which integrates conditional random fields (CRF) and ARMGCN. CRF establishes the label joint probability distribution conditioned features for learning user dependencies, and the distribution can be optimized by a variational EM algorithm. We extensively evaluate the proposed method on two real-world Twitter datasets, and the experimental results demonstrate that MDGCN outperforms the state-of-the-art baselines. In addition, the ARMRF layer is model-independent, so it can be integrated with existing advanced detection methods to improve detection performance further.

Index Terms—Spammer detection, Markov random fields, graph convolutional networks

1 Introduction

Online Social Networks (OSNs) such as Twitter [1] and Facebook [2] have gained increasing popularity in recent years. It provides people with a platform to share unsatisfied common interests and needs across time and space, making these social networks gradually become part of people's daily routines. One global digital report[1] reveals that more than 60% of people worldwide use the internet, and the number of users on various social media platforms has increased by half a billion over the past 12 months.

However, coins have two sides. While the openness of OSN brings convenience to people, it also attracts criminal accounts (spammers) surface. Spammers often create fake accounts on social network sites, then spread advertisements to promote sales, post tweets containing pornographic sites, or send unwanted information to legitimate users to seize their privacy. On the one hand, we cannot deny the objective existence of these users, which will encourage researchers to explore and study the spammer detection algorithm and the robustness of OSN. On the other hand, these users significantly affect the experience of legitimate users and reduce their trust in the system. Therefore, no matter from theoretical research or practical application, designing an effective spammer detection method is essential.

The spammer detection problem can be formulated as a binary classification to classify legitimate users and malicious users. The extant methods can be divided into two main categories: feature-based models and propagation models. Feature-based models [3], [4], [5], [6], [7] generally learn users' embedding from tweets or behavior, then devise a classification module using learned features as inputs. However, these feature-based methods are supervised and can only utilize labeled users, i.e., they need a large number of labeled data to work successfully. Recently, [8] proposed a semi-supervised model to capture the task-relevant details effectively, which uses Node2Vec, Doc2Vec, and user-symbol network to extract features from social relation view, text view, and user view, respectively. However, it ignores the relations between identity labels of users. The propagation-based methods [9], [10] use Markov Random Field (MRF) to propagate the given label information among the social network based on some intuitions of correlations between user pairs. However, these methods simply compute the joint probability distributions of MRFs using a pre-trained pairwise influence weight, so they cannot learn knowledge from tweet text.

To address the aforementioned challenges, our preliminary work [11] proposed a novel end-to-end spammer

[1] https://wearesocial.com/blog/2021/04/60-percent-of-the-worlds-population-is-now-online

- *Leyan Deng and Chenwang Wu are with the School of Data Science, University of Science and Technology of China, Hefei, Anhui 230000, China. E-mail: {dleyan, wcw1996}@mail.ustc.edu.cn.*
- *Defu Lian and Enhong Chen are with the Anhui Province Key Laboratory of Big Data Analysis and Application, School of Computer Science and Technology, University of Science and Technology of China, Hefei, Anhui 230000, China. E-mail: {liandefu, cheneh}@ustc.edu.cn.*
- *Yongji Wu is with the Department of Computer Science, Duke University, Durham, NC 27708 USA. E-mail: yongji.wu769@duke.edu.*

*Manuscript received 30 Apr. 2021; revised 30 Dec. 2021; accepted 29 Jan. 2022.
Date of publication 14 Feb. 2022; date of current version 8 Nov. 2023.
This work was supported by the National Natural Science Foundation of China under Grants 61976198 and 62022077.
(Corresponding author: Defu Lian.)
Recommended for acceptance by A. Si.
Digital Object Identifier no. 10.1109/TKDE.2022.3150669*

detection approach, GCNwithMRF, to combine Graph Convolution Network (GCN) [12] and MRF. Specifically, based on three intuitions of the paired influence of social networks, we used MRF and two parameters representing reward and punishment to model the joint probability distribution of all users. Then we model it as an MRF layer of recurrent neural network structure for multi-step inference. Finally, the MRF layer uses the GCN prediction as the input probability to realize the effective combination of the two types of detection methods.

However, this study still has deficiencies. First, the weight matrix used by the MRF layer consists of only two parameters, that is, different pairwise relations get the same reward, which is unreasonable (e.g., the bidirectional follow reward of two legitimate users should be larger than the unidirectional ones). Second, GCN and MRF in GCNwithMRF are merged sequentially, and the posterior probabilities of MRF are initialized with GCN outputs. Therefore, the prerequisite for this step-by-step processing to be effective is that GCN can provide a high-confidence probability (i.e., GCN should have good performance); otherwise, it will degenerate into a basic MRF model in the worst case. However, GCN cannot model the dependencies between labels [13], so the performance in complex datasets is often not satisfactory [11].

To this end, based on the previous work of GCN with MRF, we further propose a Markov-driven Graph Convolution Network (MDGCN) for spammer detection. First, to solve the irrationality of equal reward and punishment in the previous work, we propose an Adaptive Reward MRF (ARMRF) layer. Specifically, the ARMRF layer uses an MRF to capture human insights of neighbor influences on user's identities (for instance, spammers tend to follow a large number of users). Then it assigns different learnable rewards and punishments of different neighbor relationships (e.g., for two legitimate users, the influence of bidirectional follow relation should be different from the ones of only unidirectional relation). We stack the ARMRF layer on top of GCN and call it ARMGCN, which can refine the detection performance of GCN. Furthermore, to enhance the learning ability of stacked structure ARMGCN, we propose to merge ARMGCN and conditional random field (CRF), which we call Markov-Drive Graph Convolutional Network (MDGCN). We use the EM algorithm to optimize it alternately between E-step and M-step. In E-step, we use an ARMGCN to estimate the posterior distribution of unknown user labels based on the mean-field approximation. In M-step, it is difficult to maximize the likelihood function directly, so we optimize a pseudo-likelihood function and use another ARMGCN to parameterize the local conditional distribution of users. In MDGCN, ARMGCN learns the user's feature representations and insights, and CRF models users label dependencies, so it can simultaneously use the user's features and neighbor information to perform spammer inferences.

Except for the contributions in the preliminary work [11] on spammer detection, we further deliver the following contributions.

- We propose an Adaptive Reward MRF (ARMRF) layer that uses MRF to model people insights in following relations as different reward and punishment to refine predictions. Furthermore, ARMRF is decoupled from the model, so it can be used as an independent refinement module to combine with the existing detection approach to improve the detection performance further.
- We propose a novel semi-supervised method called the Markov-Driven Graph Convolutional Networks (MDGCN) for spammer detection. MDGCN integrates the advantages of graph convolutional networks and Markov random field to model the dependency of relational users while learning feature representations.
- We conduct extensive experiments on two real-world Twitter datasets and report more performance metrics of spammer detection. The results show that the proposed method outperforms the baseline methods. In addition, through experiments, we proved that the integration of the ARMRF layer and the existing state-of-the-art methods could further improve their performance.

The remainder of this paper is organized as follows. Section 2 introduces the related work of social spammer detection. Section 3 illustrates some preliminaries, and Section 4 presents the detail of the proposed method. In Section 5, we evaluate our method on two real-world Twitter datasets. Finally, our conclusions are drawn in Section 6.

2 Related Work

Recently, the detection of spammer has become a hot issue in social network security. The methods for detecting spammer can be mainly divided into two categories [9]: feature-based models and propagation models. In the following, we will review the related work from these two aspects.

2.1 Feature-Based Methods

The existing feature-based methods [14], [15], [16], [17], [18], [19] generally exploit text features or behavior features from users. Matrix factorization is commonly used to model these features, and social graphs are used in regularization. Zhu *et al.* [3] proposed a joint optimization model that uses a matrix factorization to extract latent features of users, and the social relationship graph guides this latent feature learning process. Hu *et al.* [6] introduced a general framework for spammer detection, where a directed graph Laplacian is used to model social information, and a matrix factorization is used to model content information. Furthermore, they incorporated sentiment information by formulating a new constraint on the matrix factorization in [5]. Fu *et al.* [20] proposed a framework to measure the carefulness of users and incorporate the carefulness to improve the robustness of spammer detection. Shen *et al.* [7] considered multi-view information of users into matrix factorization instead of single view information. He *et al.* [21] proposed a content-based multi-factor attention model to extract potential patterns from spammers better. Obviously, these feature-based methods are supervised; that is, they need many labeled users, which is infeasible in practice. Therefore, Li *et al.* [8] designed a deep multi-view feature learning module that combines information from different views and then proposed a label inference module to predict labels for users.

Although feature-based methods have shown their effectiveness in experimental environments, they ignore the relations between user identities. In severe actual scenarios, malicious users will constantly change their Twitter content mode to prevent detection, and these methods often fail.

2.2 Propagation-Based Methods

The propagation-based methods propagate the information between nodes in the social graph [22], including random walk-based methods and MRF-based methods. Early works [23], [24], [25], [26], [27] mainly used random walks and then inferred the labels along the path. In recent years, more and more works [9], [10], [28], [29], [30] model the spammer detection problem as an MRF and use probability methods to estimate the identity of each user. For example, Gong *et al.* [28] adopted Loopy Belief Propagation [31], [32] to approximate the posterior probabilities. Wang *et al.* [9] applied a local rule to propagate the given label information among the social network, then predicts labels of remaining nodes by combining the influences from their neighbors and themselves. Besides, they extended the work to design a guilt-by-association method on directed graphs in [10], which utilizes pairwise Markov Random Field (pMRF) to model the joint probability distribution of random variables corresponding users. Zhang *et al.* [33] proposed an improved multi-label propagation strategy. First, an automatic filtering mechanism is proposed to filter uncertain nodes, and second, label propagation strength is introduced to propagate labels accurately. Compared to feature-based methods, the propagation-based methods can discover more suspicious users [34]. However, the design of the propagation-based method usually requires a large number of assumptions, and it is easy for spammers to establish relationships with legitimate users and cause assumptions to become invalid. For example, the spammer can easily simulate a real user account and inherit all the social relationships.

Compared to existing approaches, the proposed method has the following differences except for the ones[2,3] inherited from the preliminary work [11]. On the one hand, our model is the first fusion of GCN and CRF in spammer detection and uses the EM algorithm for optimization. On the other hand, we re-examine people's insights into social relationships and propose an Adaptive Reward MRF layer, which assigns different learnable rewards and punishments to different paired relations in social networks.

3 Preliminaries

3.1 Problem Definition

The problem of semi-supervised spammer detection considers a directed social graph $G=(V,E)$, including a set of nodes (users) $v \in V$ and a set of edges (follow relations) $e \in E$. The edge (u,v) represents user u follows the user v. We associate a binary random variable y_v for each user v to indicate whether the user is a spammer ($y_v=1$) or legitimate ($y_v=0$). Besides, each user v is associated with a feature $\boldsymbol{x}_v \in \mathbb{R}^m$, where m denotes the number of dimensions of the feature. Given a set of labeled users $L \subseteq V$, the goal of spammer detection is to detect malicious users in the unlabeled user set $U = V \backslash L$. Formally, spammer detection aims to learn the joint probability distribution of users' identities in social graph conditioned users' attributes and relations, i.e., $p(\boldsymbol{y}_V|\boldsymbol{x}_V, E)$, where $\boldsymbol{x}_V=\{\boldsymbol{x}_v|v\in V\}$ and $\boldsymbol{y}_V=\{y_v|v\in V\}$ (note that we omit the edge set E in the following parts). The distribution $p(\boldsymbol{y}_V|\boldsymbol{x}_V)$ can be modeled with conditional random fields, as follows:

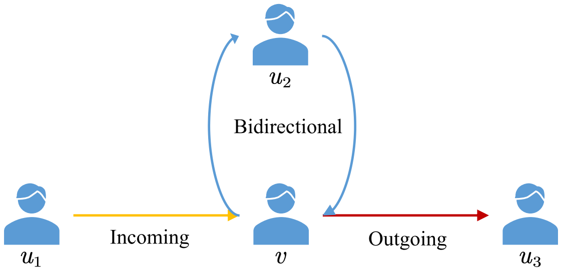


Fig. 1. Illustration of three types of neighbors. Here u_1, u_2, u_3 are bidirectional, unidirectional incoming, unidirectional outgoing neighbors of user v, respectively.

$$p(\boldsymbol{y}_V \mid \boldsymbol{x}_V) = \frac{1}{Z(\boldsymbol{x}_V)} \prod_{(u,v)\in E} \psi_{u,v}(y_u, y_v, \boldsymbol{x}_V), \tag{1}$$

where (u,v) is an edge in social graph G, $Z(\boldsymbol{x}_V)$ is the normalizing constant, and $\psi_{u,v}(\cdot)$ is the potential function. The specific definition of the potential function $\psi_{u,v}(\cdot)$ will be discussed later.

3.2 Graph Convolutional Network in Spammer Detection

There are three types of relations in social networks between two accounts: following, follower, and reciprocal follow. Therefore, we consider the three types of neighbors separately to capture different aspects of user's behaviors.

- *Bidirectional Edge*: We denote the set of bidirectional edges in the social graph as E_b, $(u,v) \in E_b$ means that the user u and the user v follow each other. Notice that for bidirectional edges (u,v) and (v,u), only one appears in E_b.
- *Unidirectional Incoming Edge*: We denote the set of unidirectional incoming edges in the social graph as E_i, $(u,v) \in E_i$ represents that the user v follows the user u but not vice versa.
- *Unidirectional Outgoing Edge*: Similarly, the set of unidirectional outgoing edges are denoted as E_o, $(u,v) \in E_o$ represents that the user u follows the user v but not vice versa.

Fig. 1 illustrates the three relations. For simplicity, we describe these three relations as three adjacency matrices, which are denoted as A_i, A_o, A_b, respectively.

The original forward propagation rule of GCN on a single directed graph $\boldsymbol{H}^{(l+1)} = \sigma(\boldsymbol{D}^{(-1)}\boldsymbol{A}\boldsymbol{H}^{(l)}\boldsymbol{W}^{(l)})$, where $\sigma(\cdot)$ denotes an activation function. Nevertheless, in spammer detection, we should treat the three types of neighbors separately when performing graph convolution. We assign separate weight matrices for each type of neighbors, and we

[2] Taking advantage of both feature-based and propagation-based methods.

[3] Using pairwise MRF to model the human intuitions.

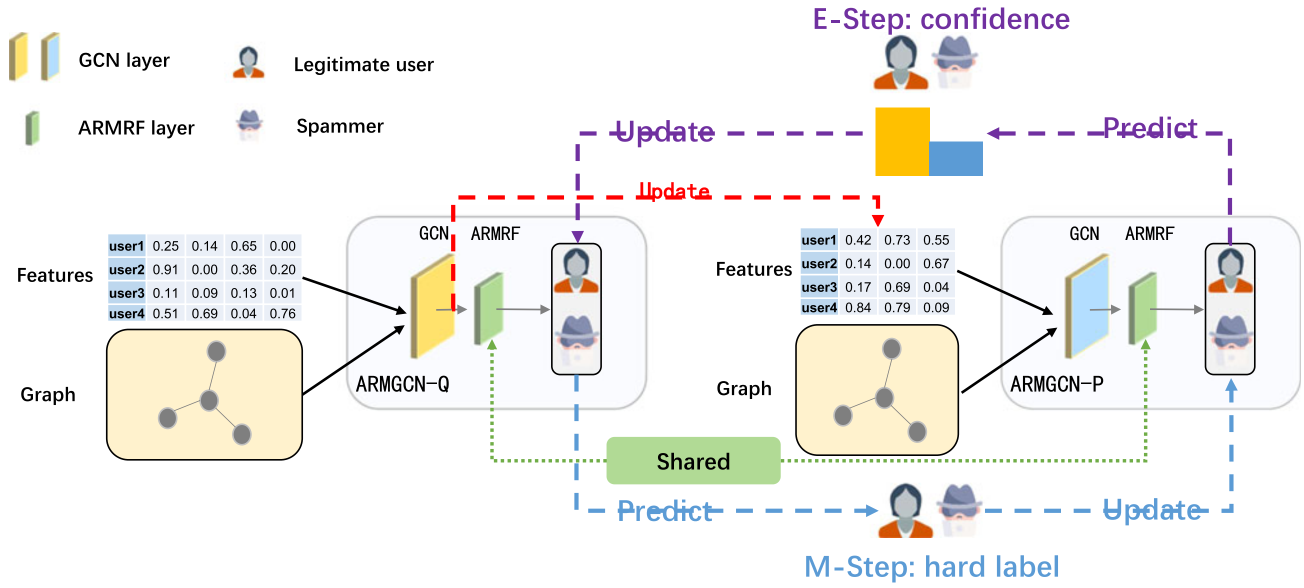


Fig. 2. The framework of our method using the variational EM algorithm.

have the following layer-wise propagation rule:

$$H^{(l+1)} = \sigma\Big(D_i^{-1}A_iH^{(l)}W_i^{(l)} + D_o^{-1}A_oH^{(l)}W_o^{(l)} + \tilde{D}_b^{-\frac{1}{2}}\tilde{A}_b\tilde{D}_b^{-\frac{1}{2}}H^{(l)}W_b^{(l)}\Big). \quad (2)$$

Here D_i, D_o are the degree matrices of A_i, A_o. $\tilde{A}_b = A_b + I_N$ is the adjacency matrix of bidirectional relations with added self-connections, and $\tilde{D}_b$ is its degree matrix. The feature matrix of users forms the input to the first GCN layer as $H^{(0)} = X$. In this paper, we use bag-of-words (BoW) features extracted from each user's tweets. We will show the importance of assigning different weights for different types of neighbors in the experimental section.

3.3 Markov Random Field in Spammer Detection

The GCN-based methods detect spammers through layer-wise neighbor message-passing. They learn how to aggregate information from each type of neighbor using weight matrices implicitly. However, there exist several explicit natural patterns entailed in user following relations that we can utilize to enhance the GCN model further. Different types of neighbors would have different impacts on a user's identity. Concretely, we have the following intuitions of pairwise influences in the social network:

- *Intuition I*: Bidirectional neighbors of a user u tend to have the same label as u. This property is known as the homophily of social networks.
- *Intuition II*: If a user u has many unidirectional incoming neighbors (i.e., many users follow u), u tends to be a legitimate user.
- *Intuition III*: If a user u has a lot of unidirectional outgoing neighbors (i.e., u follows many users), u tends to be a spammer.

The pairwise MRF (pMRF) can models the joint probability distribution of all users' identities to capture the three intuitions. A pMRF can be formulated in this Gibbs distribution form: $P(\boldsymbol{y}_V) = \frac{1}{Z}\exp(-E(\boldsymbol{y}_V))$ (here Z is a normalizing constant). The energy function E consists of unary potentials $\sum_v \phi(y_v)$ and pairwise potentials $\sum_{u,v}\varphi(y_u, y_v)$. Note that the lower the energy E (or potentials $\phi(y_v), \varphi(y_u, y_v)$) is, the higher the probability $P(y_V)$ becomes. These three intuitions correspond to the three types of neighbors, we can model them as the following energy function:

$$E(\boldsymbol{y}_V) = \sum\nolimits_{v\in V}\phi_v(y_v) + \sum\nolimits_{(u,v)\in E_b}\varphi_{bi}(y_u, y_v) + \sum\nolimits_{(u,v)\in E_{uni}}\varphi_{uni}(y_u, y_v). \quad (3)$$

Here E_{uni} denotes the unidirectional edges set, i.e., E_o and E_i. For now, we ignore the specific definition of the energy function, and we will discuss it in the following section.

4 Markov-Driven Graph Convolutional Network

4.1 Overview

To solve the deficiency that GCN cannot utilize people's insight of local label information, we stack an Adaptive Reward MRF (ARMRF) layer based on our previous work on top of GCN, which we call ARMGCN, to fix incorrect predictions made by GCN. In addition, to simultaneously use the user's features and neighbor identity information, we propose to merge ARMGCN and conditional random field (CRF), which we call MDGCN. We employ the EM algorithm to optimize MDGCN alternately. The E-step and M-step in the EM algorithm are parameterized as two ARMGCNs. ARMGCN could capture user' representations and label relations as well as the CRF models the dependency between local users. Fig. 2 shows the framework of MDGCN for spammer detection.

Note that we set the two ARMRF layers to share the same set of parameters in Fig. 2. This is because the ARMRF layer plays the role of learning the user label relations from human intuitions in both E-step and M-step, and the parameters of the same dataset modeling intuition should be consistent.

4.2 ARMRF

In this section, we describe the specific formulation of the MRF given by Eq. (3).

As shown in Eq. (3), we use pMRF to model the problem of spammer detection, but the exact posterior distribution $P(\boldsymbol{y}_V) = \frac{1}{Z}\exp(-E(\boldsymbol{y}_V))$ is infeasible to evaluate. So in this part, we introduce the computation of the MRF given by Eq. (3) through mean-field approximation.

In mean-field inference, we replace the exact posterior distribution $P(\boldsymbol{y}_V)$ with a factorizable distribution $Q(\boldsymbol{y}_V) = \prod_{v\in V} Q_v(y_v)$ that is used to minimize the KL-divergence between the two distributions: $D(Q \parallel P) = \mathbb{E}_{\boldsymbol{y}_V\sim Q}[\log Q(\boldsymbol{y}_V)] - \mathbb{E}_{\boldsymbol{y}_V\sim Q}[\log P(\boldsymbol{y}_V)]$. By substituting $P(\boldsymbol{y}_V)$ and $Q(\boldsymbol{y}_V)$, we get

$$
\begin{aligned}
D(Q \parallel P) = {} & \mathbb{E}_{\boldsymbol{y}_V\sim Q}[E(\boldsymbol{y}_V)] \\
& + \textstyle\sum_{v\in V} \mathbb{E}_{\boldsymbol{y}_v\sim Q_v}[\log Q_v(\boldsymbol{y}_v)] + \log Z.
\end{aligned}
$$

To optimize $Q_v(y_v)$, we define a Lagrangian composed of all terms involving $Q_v(y_v)$ in $D(Q \parallel P)$

$$
\begin{aligned}
L_v(Q) = {} & \mathbb{E}_{\boldsymbol{y}_V\sim Q}[E(\boldsymbol{y}_V)] + Q_v(y_v)\log Q_v(y_v) \\
& + \alpha(\textstyle\sum_{y_v} Q_v(y_v) - 1).
\end{aligned} \tag{4}
$$

Here the term involving Lagrange multiplier α assures that Q_v is a proper probability distribution. Now we take derivatives of Eq. (4) with respect to $Q_v(y_v)$ and set the derivative to 0, then we get optimal $Q_v(y_v)$

$$
\begin{aligned}
Q_v(y_v) = {} & \frac{1}{z}\exp\{-\phi_v(y_v) \\
& - \textstyle\sum_{u\in\mathcal{N}_b(v)} \mathbb{E}_{y_u\sim Q_u}[\varphi_b(y_u, y_v)] \\
& - \textstyle\sum_{u\in\mathcal{N}_i(v)} \mathbb{E}_{y_u\sim Q_u}[\varphi_{uni}(y_u, y_v)] \\
& - \textstyle\sum_{u\in\mathcal{N}_o(v)} \mathbb{E}_{y_u\sim Q_u}[\varphi_{uni}(y_v, y_u)]\},
\end{aligned} \tag{5}
$$

where $\mathcal{N}_i(u), \mathcal{N}_o(u), \mathcal{N}_b(u)$ denote the set of unidirectional incoming, unidirectional outgoing, bidirectional neighbors of a user u; $z = \exp\{\alpha + 1\}$ is the normalizing constant. The complete proof can be found in our previous work [11].

In our preliminary work [11], unary potentials $\phi_v(y_v) = -\log p_v(y_v)$ measure the prior probabilities of label assignments, where $p_v(y_v)$ is the output probability that user v has label y_v from GCN. For pairwise potentials, we reward the edge between nodes, i.e., $\varphi_b(y_u, y_v) = -w$ when y_u and y_v are the same (both spammers or both legitimate users), w' otherwise. $\varphi_{uni}(y_u, y_v) = -w$ when $y_u = 1$ or $y_v = 0$, $\varphi_{uni}(y_u, y_v) = w'$ only if $y_u = 0$ and $y_v = 1$ (legitimate users follow spammers). Here the reward and penalty factor w, $w' \geq 0$ are learnable parameters. It is easy to know that the bidirectional pairwise potentials capture the first intuition described in Section 3.3, while the two unidirectional pairwise potentials capture the second and third ones.

By modeling intuition as reward and punishment factors w and w', we can write Eq. (5) in the matrix form, where each row of Q corresponds to a user, and the two columns correspond to identity labels (spammer and legitimate user). the update rule as shown as follows:

$$
\begin{aligned}
\boldsymbol{Q} = {} & softmax\Bigg(\log \boldsymbol{H}^{(L)} - \boldsymbol{A}_i\boldsymbol{Q}\begin{bmatrix} -w & w' \\ -w & -w \end{bmatrix} \\
& - \boldsymbol{A}_o\boldsymbol{Q}\begin{bmatrix} -w & -w \\ w' & -w \end{bmatrix} - \boldsymbol{A}_b\boldsymbol{Q}\begin{bmatrix} -w & w' \\ w' & -w \end{bmatrix}\Bigg).
\end{aligned} \tag{6}
$$

More details on this conversion can be found in the preliminary work [11]. Here $\boldsymbol{H}^{(L)}$ is the softmax output from the last layer of GCN with the same shape as $\boldsymbol{Q}$, which is the predicted probabilities of user identities.

Algorithm 1. Forward Propagation of ARMGCN

```
Input: A Graph $G$; input features matrix $X$; GCN depth $K$;
    the ARMRF inference steps $T$; weight matrices
    $W_i^{(l)}, W_o^{(l)}, W_b^{(l)}, \forall l \in \{1, \ldots, L\}$; ARMRF weight matrices
    $W_1, W_2$; non-linearity $\sigma$.
Output: Predicted probability matrix $Q$, where the first column represents the probability of being a spammer, the
    second column represents that of being a legitimate user.
Construct $A_i, A_o, A_b$ from $G$.
$H^{(0)} \leftarrow X$
for $l = 1 \ldots L - 1$ do
    Compute $H^{(l)}$ from $H^{(l-1)}$ using Eq. (2) with non-linearity $\sigma$.
end for
Compute $H^{(L)}$ from $H^{(L-1)}$ using Eq. (2) with softmax nonlinearity.
$Q \leftarrow H^{(L)}$ Initialize posterior probabilities of the ARMRF layer with GCN outputs
  for $i = 1 \ldots T$ do
    Update $Q$ according to Eq. (7).
end for
return $Q$
```

However, different pairwise relations are given same reward or penalty, which is unreasonable. For example, for two legitimate users, the reward for a reciprocal following relation should be different from a unidirectional following relation, and intuitively should be greater. This kind of reward and punishment mechanism with the same weight will limit the expressive ability of MRF. Therefore, we propose an Adaptive Reward MRF (ARMRF), which is also based on the three user insights and assigns different learnable rewards and penalties to the influence of different neighbor relationships. Then the improved computation of MRF can be formulated as

$$
\begin{aligned}
\boldsymbol{Q} = {} & softmax\Bigg(\lambda\log \boldsymbol{H}^{(L)} - (1-\lambda)\Bigg(\boldsymbol{A}_i\boldsymbol{Q}\boldsymbol{W}_1 \odot \begin{bmatrix} -1 & 1 \\ -1 & -1 \end{bmatrix} \\
& + \boldsymbol{A}_o\boldsymbol{Q}\boldsymbol{W}_1^T \odot \begin{bmatrix} -1 & -1 \\ 1 & -1 \end{bmatrix} + \boldsymbol{A}_b\boldsymbol{Q}\boldsymbol{W}_2 \odot \begin{bmatrix} -1 & 1 \\ 1 & -1 \end{bmatrix}\Bigg)\Bigg),
\end{aligned} \tag{7}
$$

where λ is a learned parameter controlling the unary potentials and the pairwise potentials, $\boldsymbol{W}_1 \in \mathbb{R}_+^{2\times 2}, \boldsymbol{W}_2 \in \mathbb{R}_+^{2\times 2}$, and $\odot$ is the element-wise multiplication.

We notice from Eq. (7) that the computation of $\boldsymbol{Q}$ relies on $\boldsymbol{Q}$ itself; hence iterative computation is required. As RNNs compute the outputs of each time step based on the results from the previous time step, we use them as the building block to conduct this iterative computation using a fixed number of steps. The difference between our model and the RNNs used in natural language tasks is that there is no input in our model, only the cell state describing posterior probabilities $\boldsymbol{Q}$, which is initialized with GCN outputs $\boldsymbol{H}^{(L)}$. Using this RNN framework enables us to implement the iterative computation of Eq. (7) through multi-step inference.

4.3 MDGCN

The ARMRF layer is designed to correct the detection results of GCN. We propose ARMGCN to incorporate the

GCN and the MRF sequentially; the complete forward propagation procedure of ARMGCN is described in Algorithm 1. As mentioned above, the ARMRF layer fixes the prediction of GCN by modeling the user insights about social relations.

However, due to the lack of modeling of user label dependency, GCN cannot guarantee to provide accurate predictions with high confidence, which will affect the performance of ARMRF. Moreover, the designed potential functions for ARMRF are not enough to model the user identity dependencies. Therefore, we further merge ARMGCN and CRF to propose a Markov-driven Graph Convolutional Network (MDGCN).

We consider the basic problem of spammer detection defined by Eq. (1), which uses the conditional random field to simulate the joint probability distribution of user identity, i.e., $P_\phi(\boldsymbol{y}_L|\boldsymbol{x}_V)$, where ϕ is the model parameters. In semi-supervised learning, the parameters ϕ can be learned by maximizing the likelihood function of labeled users, i.e., $\arg\max_\phi P_\phi(\boldsymbol{y}_L|\boldsymbol{x}_V)$. However, it is intractable to directly maximize the likelihood. Therefore, we consider optimizing the lower bound of likelihood, where we use log likelihood

$$\begin{aligned}
&\log P_\phi(\boldsymbol{y}_L \mid \boldsymbol{x}_V)\\
&= \log \mathbb{E}_{Q_\theta(\boldsymbol{y}_U \mid \boldsymbol{x}_V)} \frac{P_\phi(\boldsymbol{y}_L, \boldsymbol{y}_U \mid \boldsymbol{x}_V)}{Q_\theta(\boldsymbol{y}_U \mid \boldsymbol{x}_V)}\\
&\geq \mathbb{E}_{Q_\theta(\boldsymbol{y}_U \mid \boldsymbol{x}_V)} \log \frac{P_\phi(\boldsymbol{y}_L, \boldsymbol{y}_U \mid \boldsymbol{x}_V)}{Q_\theta(\boldsymbol{y}_U \mid \boldsymbol{x}_V)}\\
&= \mathbb{E}_{Q_\theta(\boldsymbol{y}_U \mid \boldsymbol{x}_V)} \left[\log P_\phi(\boldsymbol{y}_V \mid \boldsymbol{x}_V) - \log Q_\theta(\boldsymbol{y}_U \mid \boldsymbol{x}_V)\right].
\end{aligned}$$

The equation holds when $Q_\theta(\boldsymbol{y}_U \mid \boldsymbol{x}_V) = P_\phi(\boldsymbol{y}_U \mid \boldsymbol{y}_L, \boldsymbol{x}_V)$. At this point, we can use the EM algorithm [35] to optimize the low bound alternately through an expectation step (E-step) and a maximization step (M-step). In the E-step, we fix P_ϕ and infer the posterior distribution via minimizing the KL divergence between $Q_\theta(\boldsymbol{y}_U \mid \boldsymbol{x}_V)$ and $P_\phi(\boldsymbol{y}_U \mid \boldsymbol{y}_L, \boldsymbol{x}_V)$. In the M-step, we fix Q_θ and learn the parameters by maximizing the log-likelihood.

4.3.1 E-Step

The E-step aims to minimize the KL divergence between the variational posterior distribution $Q_\theta(\boldsymbol{y}_U \mid \boldsymbol{x}_V)$ and the true posterior distribution $P_\phi(\boldsymbol{y}_L \mid \boldsymbol{x}_V)$ when ϕ is fixed. Due to the intractable of exact inference, we propose to approximate the variational posterior $Q_\theta(\boldsymbol{y}_U \mid \boldsymbol{x}_V)$ with a mean-field distribution, in which each user's identity is inferred independently, then we can get the posterior probability of each user u [13]

$$\begin{aligned}
&\log Q_\theta(y_u \mid \boldsymbol{x}_V)\\
&= \mathbb{E}_{Q_\theta\left(\boldsymbol{y}_{\mathrm{NB}(u)\cap U} \mid \boldsymbol{x}_V\right)}\left[\log P_\phi\Big(y_u \mid \boldsymbol{y}_{\mathrm{NB}(u)}, \boldsymbol{x}_V\Big)\right] + \text{const}, \qquad (8)
\end{aligned}$$

where the $\mathrm{NB}(u)$ is the neighbor set of user u.

However, the optimization of Eq. (8) involves the expectation with respect to $P_\phi(y_u \mid \boldsymbol{y}_{\mathrm{NB}(u)}, \boldsymbol{x}_V)$, so we estimate the expectation by sampling $\hat{\boldsymbol{y}}_{\mathrm{NB}(u)} = \{\hat{y}_{u'}\}_{u' \in \mathrm{NB}(u)}$ from $Q_\theta(\boldsymbol{y}_{\mathrm{NB}(u)\cap U} \mid \boldsymbol{x}_V)$ following [36]. Then we have

$$\begin{aligned}
&\mathbb{E}_{Q_\theta\left(\boldsymbol{y}_{\mathrm{NB}(u)\cap U} \mid \boldsymbol{x}_V\right)}\left[\log P_\phi\Big(y_u \mid \boldsymbol{y}_{\mathrm{NB}(u)}, \boldsymbol{x}_V\Big)\right]\\
&\simeq \log p_\phi\Big(y_u \mid \hat{\boldsymbol{y}}_{\mathrm{NB}(u)}, \boldsymbol{x}_V\Big). \qquad (9)
\end{aligned}$$

Based on Eqs. (8) and (9), we have

$$P_\theta(y_u \mid \boldsymbol{x}_V) \approx Q_\phi\Big(y_u \mid \hat{\boldsymbol{y}}_{\mathrm{NB}(u)}, \boldsymbol{x}_V\Big).$$

So the optimization goal is

$$\arg\max_\theta \sum_{u \in U} \mathbb{E}_{P_\phi\left(y_u \mid \hat{\boldsymbol{y}}_{\mathrm{NB}(u)}, \boldsymbol{x}_V\right)}[\log Q_\theta(y_u \mid \boldsymbol{x}_V)].$$

We also add a supervised learning objective to enhance the inference network, as follow:

$$\arg\max_\theta \sum_{u \in L} \log Q_\theta(y_u \mid \boldsymbol{x}_V). \qquad (10)$$

So the overall objective function becomes

$$\begin{aligned}
\arg\max_\theta &\sum_{u \in U} \mathbb{E}_{P_\phi\left(y_u \mid \hat{\boldsymbol{y}}_{\mathrm{NB}(u)}, \boldsymbol{x}_V\right)} \log Q_\theta(y_u \mid \boldsymbol{x}_V)\\
&+ \sum_{u \in L} \log Q_\theta(y_u \mid \boldsymbol{x}_V). \qquad (11)
\end{aligned}$$

Besides, we notice that the number of spammers is far less than legitimate users in real-world social networks. The imbalance of data will cause the model to be over-fitted to legitimate users in the early training stage, making it difficult for the E-step to provide accurate parameter estimates for the M-step. Therefore, We design a simple random oversampling strategy to upsample spammers until the number of spammers is equal to the legitimate users. The overall E-step objective function in each iteration becomes

$$\begin{aligned}
\arg\max_\theta &\sum_{u \in V_{s,U}} \mathbb{E}_{P_\phi\left(y_u \mid \hat{y}_{\mathrm{NB}(u)}, \boldsymbol{x}_V\right)}[-\log Q_\theta(y_u \mid \boldsymbol{x}_V)]\\
&- \sum_{u \in V_{s,L}} \log Q_\theta(y_u \mid \boldsymbol{x}_V). \qquad (12)
\end{aligned}$$

Here, $V_{s,L}$ is the labeled subset, and $V_{s,U}$ is the unlabeled subset; $V_s = V_{s,L} \cup V_{s,U}$ is the new dataset after sampling, where the negative samples (legitimate users) are the same as the ones in V, and the positive samples (spammers) are oversampled to the same number as the negative samples.

Based on the optimization function described in Eq. (12), we parameterize the variational posterior distribution Q_θ with an ARMGCN. Through ARMGCN, we can infer spammer by learning users' representation from their Twitter content, social relation graph, and intuitions.

4.3.2 M-Step

In the M-step, our goal is to learn the parameters ϕ when θ is fixed. However, it is intractable to maximize the likelihood function directly. To tackle this problem, we instead optimize a pseudo-log-likelihood optimization function $\mathcal{L}(\phi)$ proposed in [37], which is defined as

$$\begin{aligned}
\mathcal{L}(\phi) &:= \mathbb{E}_{Q_\theta(\boldsymbol{y}_U \mid \boldsymbol{x}_V)}\left[\sum_{u \in V} \log P_\phi\Big(y_u \mid \boldsymbol{y}_{V \setminus u}, \boldsymbol{x}_V\Big)\right]\\
&= \mathbb{E}_{Q_\theta(\boldsymbol{y}_U \mid \boldsymbol{x}_V)}\left[\sum_{u \in V} \log P_\phi\Big(\boldsymbol{y}_u \mid \boldsymbol{y}_{\mathrm{NB(u)}}, \boldsymbol{x}_V\Big)\right]. \qquad (13)
\end{aligned}$$

For the M-step, we employ another AMRGCN to parameterize the local conditional distribution of users. We fix the Q_θ and optimize the P_ϕ by maximizing Eq. (13). We estimate the expectation in Eq. (13) by drawing a sample $\hat{\boldsymbol{y}}_{\mathrm{NB}(u)}$ and $\hat{y}_u$ from $Q_\theta(y_u \mid \boldsymbol{x}_V)$. If user u is a label user, we use the true label. Therefore, the objective function for M-step is formulated as

$$\arg\max_{\phi} \sum_{u \in U} \log P_\phi\Big(\hat{y}_u \mid \hat{\boldsymbol{y}}_{\mathrm{NB}(u)}, \boldsymbol{x}_V\Big). \tag{14}$$

Therefore, P_ϕ can treat the output $\hat{\boldsymbol{y}}_{\mathrm{NB}(u)}$ of model Q_θ surrounding the user u as features. In practice, the hidden feature of Q_θ can retain more information compared to outputs, and thus P_ϕ uses the average of hidden features surrounding the user u as features. Here the hidden features are the outputs of the first layer in Q_θ.

Algorithm 2. MDGCN

```
Input: A Graph $G$; Number of iterations $K$; some labeled users $(L, \boldsymbol{y}_L)$; input features matrix $\boldsymbol{X}$; two ARMGCNs $Q_\theta, P_\phi$.
Output: Predicted probability matrix $\boldsymbol{Q}$
Pre-train $Q_\theta$ with $\boldsymbol{y}_L$ according to Eq. (10).
  for $k = 1 \ldots K$ do
    M-step
    Sample $\hat{\boldsymbol{y}}_U$ from $Q_\theta(\boldsymbol{y}_U \mid \boldsymbol{x}_V)$.
    Update input features of $P_\phi$ with $Q_\theta$'s hidden features.
    Set $\hat{\boldsymbol{y}}_V = (\boldsymbol{y}_L, \hat{\boldsymbol{y}}_U)$ and update $P_\phi$ with Eq. (14).
    E-step
    Set $\hat{\boldsymbol{y}}_U = \{P_\phi\Big(y_u \mid \hat{\boldsymbol{y}}_{\mathrm{NB}(u)}, \boldsymbol{x}_V\Big) \mid u \in U\}$.
    Over-sample $\hat{\boldsymbol{y}}_{V_s}$ from $\hat{\boldsymbol{y}}_V = (\boldsymbol{y}_L, \hat{\boldsymbol{y}}_U)$.
    Update $Q_\theta$ with Eq. (12) based on $\hat{\boldsymbol{y}}_{V_s}$.
end for
return $\boldsymbol{Q} \leftarrow Q_\theta(\boldsymbol{y}_U \mid \boldsymbol{x}_V)$
```

The detailed procedure of MDGCN is summarized in Algorithm 2. We pre-train Q_θ with the labeled user set L. Then we alternately train Q_θ and P_ϕ with a fixed number of iterations. Afterward, both Q_θ and P_ϕ can be used to detect spammers. In our experiments, we find that the performance of the model Q_θ is better than that of the P_ϕ, so we use Q_θ as the final detection model.

5 Experiments

5.1 Experimental Setup

5.1.1 Datasets

We use two public datasets to evaluate our method: Twitter Social Honeypot Dataset (TwitterSH) [38] and Twitter 1KS-10KN dataset (1KS-10KN) [26]. The two datasets collected from Twitter contain labeled spammers and legitimate users, along with their corresponding tweets. In addition, considering the lack of social network information, we use an external Twitter social graph dataset [39] to extract the users' social relations in the TwitterSH dataset. For each tweet content, we perform preprocessing operations such as deleting non-English words, URLs, numbers, mentions and stop words, etc. Then we select the 500 most frequent words of the filtered tweets to construct the vocabulary tables and then construct each user's bag-of-words feature. Furthermore, we filter out users whose non-zero feature is less than 5 and the number of following relations and follower relations are less than 2 to get the experimental dataset. Among them, TwitterSH left 1921 legitimate users and 2953 spammers, and 1KS-10KN has 9441 legitimate users and 625 spammers. The specific statistics of the two datasets are shown in Table 1. It is worth noting that the 1KS-10KN dataset is highly imbalanced, and the ratio of spammers to legitimate users is 1:15. We divide each data set into training set, validation set, and test set according to the ratio of 4:1:5.

TABLE 1
Statistics of Processed Datasets

	TwitterSH	1KS-10KN
Spammers	2953	625
Legitimate users	1921	9441
Tweets	704152	1030828
Social relations	312619	2294116
Sparsity of relations	98.69%	97.74%

Remarks. In this paper, we mainly focus on the spam detection of relational Twitter users. For other public datasets, such as [40], [41], due to the lack of social relations between users, they are not suitable for our experimental environment, so they are omitted here.

5.1.2 Compared Baselines

We compare our proposed MDGCN with the following baselines, including state-of-the-art spammer detection approaches and variants of GCN:

- *SMFSR* [3]: It is a supervised social spammer detection method based on matrix factorization to learn latent user and text features representations. Undirected social graphs are used as a regularization mechanism.
- *OSSD* [6]: This method further extends SMFSR to consider directed social networks.
- *SybilBelief* [28]: It is a Loop Belief Propagation (LBP)-based method using a pairwise Markov Random Field, based on the intuition that neighbor users tend to have the same label.
- *SybilSCAR* [9]: Considering the non-robustness of the random walk to noise and the non-scalability of the LBP method, this algorithm unifies the two methods to overcome their shortcomings.
- *GANG* [10]: An improved version of SybilSCAR, which considers the directed social graph composed of three types of neighbors.
- *RF* [42]: It is a random forest baseline utilizing the bag-of-words features we constructed.
- *GCN* [12]: It is a basic graph convolution neural network, where each layer aggregates the features of three types of neighbors.
- *GCNsg*: It is a variant of the GCN model. It operates on a single directed social network instead of modeling three types of neighbors separately.
- *CARE-GNN* [43]: Considering the forged features of abnormal data, this method uses three modules: label perception, reinforcement learning, and cross-relational neighbor aggregation to enhance GNN's anomaly detection performance.
- *GIN* [44]: A powerful GNN under the neighborhood aggregation framework, which preserves the graph isomorphism.
- *DCI* [45]: This method decouples the node representation learning and the anomaly detection. It adopts

TABLE 2
Comparison Results on TwitterSH Dataset

ALG	20%			40%			60%			80%			100%		
	ACC	PRAUC	F_1	ACC	PRAUC	F_1	ACC	PRAUC	F_1	ACC	PRAUC	F_1	ACC	PRAUC	F_1
SMFSR	0.5253	0.6684	0.5060	0.5366	0.6841	0.5000	0.5381	0.6830	0.5065	0.5414	0.6824	0.5154	0.5424	0.6885	0.5123
OSSD	0.6353	0.6813	0.7559	0.6293	0.6861	0.7424	0.6370	0.6802	0.7566	0.6513	0.7108	0.7666	0.6466	0.7222	0.7579
SybilBelief	0.6763	0.7824	0.7463	0.6937	0.8100	0.7546	0.6810	0.8097	0.7569	0.6758	0.8150	0.7561	0.6787	0.8161	0.7599
SybilScar	0.6758	0.7823	0.7460	0.6947	0.8101	0.7545	0.6849	0.8128	0.7551	0.6758	0.8149	0.7561	0.6783	0.8156	0.7597
GANG	0.6758	0.7502	0.7460	0.6889	0.7399	0.7525	0.6849	0.7232	0.7551	0.6758	0.7106	0.7561	0.6783	0.7034	0.7597
RF	0.7866	0.7479	0.8411	0.7776	0.7391	0.8357	0.7792	0.7413	0.8363	0.7784	0.7390	0.8368	0.7731	0.7345	0.8332
GCNsg	0.7932	0.8670	0.8355	0.7968	0.8751	0.8390	0.7955	0.8800	0.8391	0.7973	0.8822	0.8392	0.7966	0.8873	0.8396
CARE-GNN	0.8162	0.8226	0.7846	0.7950	0.8242	0.7731	0.8137	0.8252	0.7845	0.8102	0.8207	0.7816	0.8089	0.8198	0.7809
GIN	0.7751	0.8716	0.8234	0.7636	0.8735	0.8195	0.7745	0.8775	0.8211	0.7764	0.8803	0.8195	0.7567	0.8771	0.8109
Decoupled-DGI	0.7774	0.8646	0.8273	0.7743	0.8682	0.8253	0.7881	0.8801	0.8392	0.7905	0.8863	0.8326	0.7931	0.8884	0.8356
DCI	0.7739	0.8575	0.8254	0.7883	0.8778	0.8356	0.7829	0.8820	0.8235	0.7799	0.8899	0.8353	0.7542	0.8702	0.8450
GCN	0.7872	0.8675	0.8296	0.7864	0.8649	0.8314	0.7901	0.8639	0.8339	0.7961	0.8659	0.8387	0.7996	0.8671	0.8417
GCNwithMRF	0.7944	0.8668	0.8334	0.7940	0.8664	0.8380	0.7967	0.8683	0.8405	0.7950	0.8682	0.8412	0.7995	0.8711	0.8431
MDGCN-CRF	0.8039	**0.8890*****	0.8422	0.8053	0.8889	0.8434	0.8129	0.8937	0.8482	0.8134	0.8925	0.8494	0.8164	0.8910	0.8518
ARMGCN	0.7942	0.8692	0.8371	0.7944	0.8666	0.8385	0.8025	0.8715	0.8438	0.8040	0.8726	0.8451	0.8090	0.8770	0.8488
MDGCN	**0.8096*****	0.8875	**0.8471*****	**0.8106*****	**0.8943*****	**0.8473*****	**0.8198*****	**0.9026*****	**0.8531*****	**0.8235*****	**0.9027*****	**0.8563*****	**0.8265*****	**0.9004*****	**0.8593*****

, **, and * indicate that the improvements are statistically significant for* $p < 0.05$, $p < 0.01$, *and* $p < 0.001$, *respectively.*

GIN to instantiate the GNN encoder, and then equips the decoupled training with a self-supervised learning (SSL) scheme DCI.

- *Decoupled-DGI* [45]: Similar to DCI, it utilizes a representative graph SSL scheme DGI [46] for decoupled training.

In addition, we introduce two proposed MDGCN variants for ablation experiments.

- *MDGCN-CRF*: This variant removes the ARMRF layer and directly uses the GCN outputs optimized by the EM algorithm as the prediction result, which can be used to evaluate the effectiveness of conditional random fields.
- *ARMGCN*: In this variant, we stack an ARMRF layer on a single GCN. It can be compared with GCN to evaluate the effectiveness of ARMRF layer.

5.1.3 Evaluation Metrics

First, we use accuracy (ACC) as a metric, which measures how many unlabeled users detected correctly. Besides, considering the extreme imbalance of the 1KS-10KN dataset, we use Area Under the Precision-Recall Curve (PRAUC) instead of Area under the ROC Curve (ROCAUC). The reason is that precision-recall curves are better in evaluating performance with class imbalance datasets, while ROC curves can be deceptive in this circumstance [47]. Lastly, we use the most commonly used F_1 score in anomaly detection as an important evaluation metric, which considers both the precision and recall rate of detection.

5.1.4 Parameter Settings

We optimize hyperparameters on the validation set, then the settings after hyperparameter tuning are reported. Both model P and model Q use a two-layer graph convolutional network [12], where the number of hidden units is set to 64. We use ReLU as the activation function of the hidden layer and the output layer followed by a softmax function. In addition, we use Dropout [48] with a ratio of 0.5 by default. All models are trained using Adam optimizer [49]. We set the learning rate to 0.01 in TwitterSH and a slightly lower learning rate of 0.002 in 1KS-10KN. This is because a large learning rate is easy to oscillate under the highly unbalanced 1KS-10KN, which poses challenges for training. A similar treatment is reflected in the number of training epochs. In TwitterSH, both networks are trained for 50 epochs in each iteration, while 1KS-10KN uses a larger number of training epochs of 100. Each model in MDGCN is trained for 2000 rounds, which means 40 iterations in TwitterSH, and 20 iterations for 1KS-10KN. The source code and datasets are available at https://github.com/dleyan/MDGCN.

For the compared state-of-the-art methods, if the original paper provides parameters, we use the default settings. For some unknown parameters, the grid search is utilized to obtain the optimal parameters under the validation set.

5.2 Comparisons Results

In this part, we compare MDGCN with existing advanced spam detection methods and the GCN variants (Note that variants of MDGCN will be discussed in Section 5.3). We evaluate the performance of our model and the baselines listed above on the two datasets using $20\%\sim100\%$ of data from the 40% semi-supervised training set (so the models will only see $8\%\sim40\%$ data of the entire dataset). The results are shown in Tables 2 and 3.

From these results, we can see that MDGCN consistently outperforms all the baselines compared with the increasing training data used, especially on the extremely class-imbalanced 1KS-10KN dataset where other methods yield significantly lower PRAUC values and F_1 scores. This result is encouraging because the dataset is more in line with the realistic case; that is, most of the users in the social network are legitimate users. Next, we compare MDGCN with SOTA non-GCN-based methods and GCN-based baselines.

5.2.1 Comparison With State-of-the-Art Methods

The matrix factorization-based models SMFSR and OSSD are fully supervised models, which can only exploit labeled parts of the social networks, hence performing poorly in the real-world semi-supervised setting. Still, we can observe

TABLE 3
Comparison Results on 1KS-10KN Dataset

ALG	20%			40%			60%			80%			100%		
	ACC	PRAUC	F_1	ACC	PRAUC	F_1	ACC	PRAUC	F_1	ACC	PRAUC	F_1	ACC	PRAUC	F_1
SMFSR	0.9372	0.0695	0.0360	0.9381	0.0550	0.0156	0.9398	0.0553	0.0136	0.9398	0.0503	0.0143	0.9398	0.0514	0.0095
OSSD	0.8257	0.1304	0.1157	0.8765	0.1503	0.1218	0.9275	0.1399	0.0860	0.9374	0.1437	0.0656	0.9374	0.1297	0.0515
SybilBelief	0.9388	0.1064	0.1149	0.9387	0.0808	0.0612	0.9386	0.0795	0.0596	0.9386	0.0750	0.0063	0.9387	0.0764	0.0063
SybilScar	0.9388	0.1144	0.1154	0.9386	0.0914	0.0611	0.9380	0.0823	0.0420	0.9384	0.0783	0.0246	0.9386	0.0785	0.0063
GANG	0.9387	0.0706	0.0955	0.9386	0.0675	0.0609	0.9380	0.0651	0.0420	0.9384	0.0619	0.0246	0.9386	0.0620	0.0063
RF	0.9412	0.0655	0.0133	0.9450	0.1255	0.1317	0.9446	0.1192	0.1199	0.9456	0.1350	0.1491	0.9466	0.1508	0.1774
GCNsg	0.9408	0.1067	0.1364	0.9408	0.1175	0.1332	0.9408	0.1211	0.1509	0.9410	0.1178	0.1431	0.9409	0.1257	0.1648
CARE-GNN	0.7895	0.5662	0.2931	0.7885	0.5677	0.2947	0.7899	0.5690	0.2948	0.7867	0.5646	0.2923	0.7852	0.5591	0.2914
GIN	0.9536	0.5973	0.5388	0.9559	0.6532	0.5683	0.9581	0.6655	0.564	0.9531	0.6533	0.5732	0.9654	0.725	0.6429
Decoupled-DGI	0.9572	0.6909	0.4992	0.9547	0.6666	0.5185	0.9624	0.681	0.608	0.9555	0.7244	0.518	0.96	0.6613	0.5466
DCI	0.9587	0.6433	0.564	0.9585	0.6905	0.5251	0.9646	0.7322	0.6248	0.9634	0.7302	0.5888	0.9566	0.6725	0.5473
GCN	0.9748	0.7811	0.7747	0.9798	0.8735	0.8191	0.9818	0.8737	0.8408	0.9854	0.9132	0.8736	0.9849	0.8985	0.8722
GCNwithMRF	0.9746	0.7689	0.7645	0.9802	0.8676	0.8228	0.9826	0.8868	0.8455	0.9859	0.9220	0.8767	0.9837	0.8968	0.8688
MDGCN-CRF	0.9777	0.8596	0.7927	0.9806	0.9002	0.8320	0.9829	0.9057	0.8536	0.9857	0.9318	0.8805	0.9868	0.9392	0.8910
ARMGCN	0.9773	0.8255	0.7982	0.9821	0.8903	0.8432	0.9845	0.9130	0.8627	0.9868	0.9362	0.8849	0.9872	0.9276	0.8892
MDGCN	**0.9787***	**0.8682**	**0.8068**	**0.9837***	**0.9155***	**0.8620****	**0.9860**	**0.9310****	**0.8816***	**0.9878***	**0.9449****	**0.8906**	**0.9879**	**0.9498*****	**0.9014***

, **, and * indicate that the improvements are statistically significant for* $p < 0.05$, $p < 0.01$, *and* $p < 0.001$, *respectively.*

that OSSD is relatively better than SMFSR since it considers directed social networks. The MRF based methods SybilBelief, SybilSCAR, and GANG perform poorly on both datasets, even their F_1 scores are as low as 0.0063 on the 1KS-10KN dataset. On the TwitterSH datasets, they produce low accuracy values since the social network is quite sparse, the posterior probabilities of most nodes remain unaffected in the belief propagation process (20.5% of users have the same prior and posterior probabilities of 0.5 given 100% training data using SybilSCAR). On the 1KS-10KN dataset, most users' posterior probabilities of being legitimate users collapse to 1.0. Clearly, these three methods are not suitable for such imbalanced datasets. For RF, although it is in the lead compared with the non-GCN method, the performance in 1KS-10KN is not satisfactory. Its F_1 score is only 0.1774 in the case of 100% training set, and even as low as 0.0133 in the case of 20% training set. We suspect that on extreme imbalanced datasets, social relations are critical for detection, and RF cannot explicitly utilize user interaction, limiting its performance. Besides, the excellent performance of GCN on 1KS-10KN further verifies our conjecture. Finally, comparing MDGCN with the best non-GCN-based methods, the improvements are visible to the naked eye, and the improvement with respect to F_1 score even reaches an amazing 1000% on the 1KS-10KN dataset.

5.2.2 Comparison With GCN-Based Baselines

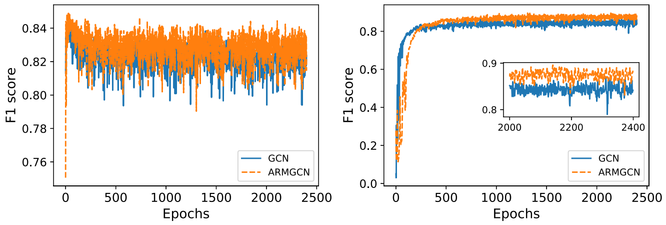
First, for three basic GNNs, by comparing GCNsg and GIN, we can observe that GIN significantly outperforms GCNsg on 1KS-10KN since it can map different neighborhoods to different node representations. Besides, comparing GCNsg and GIN with GCN, it can be seen that treating three types of neighbors separately in GCN is essential for better performance as each type of neighbors implies different kinds of information, and different weight matrices should be learned. Second, for CARE-GNN, compared to the performance on TwitterSH, it is significantly worse on 1KS-10KN. This is because CARE-GNN increases the detection confidence by aggregating information-rich adjacent users, but the features on 1KS-10KN are more sparse (the non-zero features of each user are about 25%, while TwitterSH has about 40%), making it difficult to find rich information neighbors. Third, the decoupled GNN-based anomaly detection models both utilize GIN as a representation learning network, and their improvements over GIN prove that decoupled training copes with the inconsistency between the behavior patterns and the label semantics. However, the node representation learned in the imbalanced dataset limits the performances of their classifiers. Finally, compared with GCN, the proposed MDGCN has an average increase of 5% with respect to PRAUC and an average increase of 3.1% concerning F_1 score. In particular, the proposed MDGCN with its two variants significantly outperforms GCN on the imbalanced dataset (1KS-10KN). One possible reason is the oversampling strategy used in E-step. The key reason is that the ARMRF layer and the CRF both alleviate the overfitting of GCN by modeling user label dependency. These improvements are precious under such high performance of the comparison algorithms.

5.3 Performance Analysis

5.3.1 The Effectiveness of the ARMRF Layer

One key idea of this paper is to use Markov Random Field to model spammer detection intuitions. To demonstrate the refining effect of the MRF layer, let us review Tables 2 and 3. First, We find that GCNwithMRF is slightly better than GCN. This indicates that the MRF layer, which models the three intuitions, indeed helps improve the performance of GCN. However, as we discussed earlier, modeling the rewards and penalties of all relationships as the same does not conform to the actual situation and will inevitably limit its performance. Therefore, the average improvement in F1 scores is only 0.2%. In contrast, comparing its improved version ARMGCN with GCN, we can see significant improvements. PRAUC and F_1 scores are mostly increased above 2% in the 1KS-10KN dataset, and the highest improvement reaches 6%. This comparison shows the rationality of assigning different rewards and punishments to different relations.

In addition, we show the improvement of F_1 scores in GCN and ARMGCN during training, as shown in Fig. 3. On



(a) TwitterSH

(b) 1KS-10KN

Fig. 3. The change curve of F_1 score of GCN and ARMGCN with the progress of training.

the TwitterSH, ARMGCN is better than GCN in terms of F_1 score at the beginning, and continues until the end of training. On the 1KS-10KN, GCN performed better in the initial 200 epochs, but ARMGCN was better than GCN afterwards. This is because the highly imbalanced data makes GCN difficult to provide convincing probability (most of the output probability of GCN are 0 in 1KS-10KN), but as the poster probability credibility increases in GCN, the ARMRF layer gradually shows its superiority.

Furthermore, the ARMRF layer is decoupled from the algorithm, so we can add the ARMRF layer to the baselines (e.g., we define adding the ARMRF layer to SMFSR as SMFSRwithARMRF). At this time, the prediction of the baseline method will be used as the input probability of the ARMRF layer. In this experiment, we used ARMGCN to learn the parameter weights of the ARMRF layer, and the specific results are shown in Table 4. It can be seen that after adding the ARMRF layer, the performance of most approaches has been dramatically improved. In TwitterSH, the average improvement with respect to PRAUC is 14.0%, while the average improvement in 1KS-10KN is 159.3%. Similar results can be observed for F_1 score. These heartening results indicate that the proposed ARMRF layer can be combined with existing detection approaches to improve their detection performance further.

5.3.2 The Effectiveness of the Conditional Random Field

Another core of this paper is to combine GCN with CRF to simultaneously model features and related neighbors. It can be seen from Tables 2 and 3 that MDGCN-CRF, which combines GCN and CRF, has different degrees of improvement compared to GCN. In 1KS-10KN, PRAUC and F1 have an average increase of 4.7% and 1.7%, respectively, while in TwitterSH, PRAUC and F_1 score have an average increase of 2.9% and 1.5%, respectively. Similar performance gaps are also reflected in the comparison between MDGCN and ARMGCN, and their difference also lies in whether conditional random fields are used.

We also plot the changes in F1 scores of MDGCN-CRF and GCN during the training process, as shown in Fig. 4. In the first 400 epochs, MDGCN-CRF also uses GCN for pretraining, so the performance is similar to GCN. However, when MDGCN-CRF uses CRF in 400 epochs, the F_1 score has a sharp increase in both data sets, and after that, they remain better than GCN until the training end. This abrupt performance improvement intuitively proves the effectiveness of CRF modeling in spammer detection.

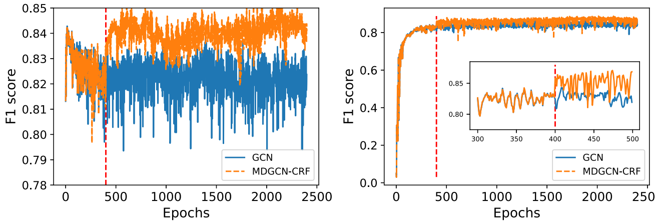
Furthermore, we report the optimization curves of the MDGCN in E-step and M-step, as shown in Fig. 5. It can be seen that the EM algorithm converges within 10 iterations, i.e., 500 epochs in TwitterSH, and 1000 epochs in 1KS-10KN.

5.4 Parameter Sensitivity

To evaluate the robustness of our model, we conduct sensitivity analysis with respect to the number of hidden units,

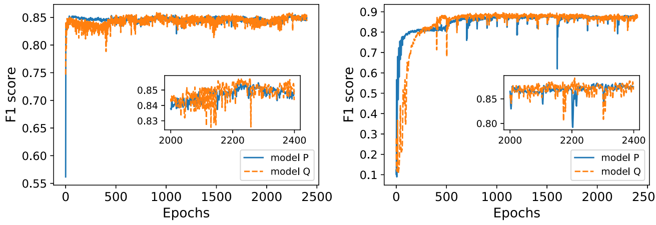
TABLE 4
Performance Comparison Before and After Using the ARMRF Layer on the Baselines

Metric	Alg	TwitterSH					1KS-10KN				
		20%	40%	60%	80%	100%	20%	40%	60%	80%	100%
PRAUC	SMFSR	0.6684	0.6841	0.6830	0.6824	0.6885	0.0695	0.0550	0.0553	0.0503	0.0514
	SMFSRwithARMRF	**0.8274**	**0.8281**	**0.8271**	**0.8296**	**0.8275**	**0.2217**	**0.2725**	**0.2685**	**0.2839**	**0.3017**
	OSSD	0.6813	0.6861	0.6802	0.7108	0.7222	0.1304	0.1503	0.1399	0.1438	0.1297
	OSSDwithARMRF	**0.8263**	**0.8272**	**0.8248**	**0.8286**	**0.8262**	0.0413	0.0413	0.0413	0.0413	0.0413
	RF	0.7479	0.7391	0.7413	0.7390	0.7345	0.0655	0.1255	0.1192	0.1350	0.1508
	RFwithARMRF	**0.8499**	**0.8491**	**0.8509**	**0.8496**	**0.8477**	**0.2217**	**0.2725**	**0.2685**	**0.2839**	**0.3017**
	Sybilbelief	0.7824	0.8100	0.8097	0.8150	0.8161	0.1064	0.0808	0.0795	0.0750	0.0764
	SybilbeliefwithARMRF	**0.8175**	**0.8225**	**0.8220**	**0.8227**	**0.8227**	**0.2098**	**0.1752**	**0.1760**	**0.1775**	**0.1750**
	GANG	0.7502	0.7399	0.7232	0.7106	0.7034	0.0706	0.0675	0.0651	0.0619	0.0620
	GANGwithARMRF	**0.8137**	**0.8178**	**0.8192**	**0.8207**	**0.8214**	**0.2002**	**0.2002**	**0.2122**	**0.2101**	**0.2133**
F_1	SMFSR	0.5060	0.5000	0.5065	0.5154	0.5123	0.0360	0.0156	0.0136	0.0143	0.0095
	SMFSRwithARMRF	**0.7609**	**0.7645**	**0.7610**	**0.7616**	**0.7662**	**0.1066**	**0.1123**	**0.1120**	**0.1123**	**0.1122**
	OSSD	0.7559	0.7424	0.7566	0.7666	0.7579	0.1157	0.1218	0.0860	0.0656	0.0515
	OSSDwithARMRF	0.7442	0.7456	0.7450	0.7477	0.7450	0.1123	0.1123	**0.1123**	**0.1120**	**0.1124**
	RF	0.8411	0.8357	0.8363	0.8368	0.8332	0.0133	0.1317	0.1199	0.1491	0.1774
	RFwithARMRF	0.8387	0.8312	0.8326	**0.8376**	**0.8364**	**0.1066**	**0.1981**	**0.1893**	**0.2153**	**0.2406**
	Sybilbelief	0.7463	0.7546	0.7569	0.7561	0.7599	0.1149	0.0612	0.0596	0.0063	0.0063
	SybilbeliefwithARMRF	**0.7525**	**0.7570**	**0.7593**	**0.7580**	**0.7603**	**0.2021**	**0.1379**	**0.1343**	**0.1060**	**0.1016**
	GANG	0.7460	0.7545	0.7551	0.7561	0.7597	0.0955	0.0609	0.0420	0.0246	0.0063
	GANGwithARMRF	**0.7512**	0.7545	**0.7562**	**0.7567**	0.7592	**0.1349**	**0.1349**	**0.1271**	**0.1108**	**0.1022**



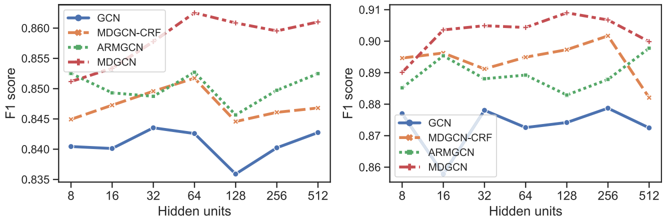
(a) TwitterSH (b) 1KS-10KN

Fig. 4. The change curve of F_1 score of GCN and MDGCN-CRF with the progress of training.



(a) TwitterSH (b) 1KS-10KN

Fig. 5. The change curve of F_1 score of MDGCN in E-step (model Q) and M-step (model P).



(a) TwitterSH (b) 1KS-10KN

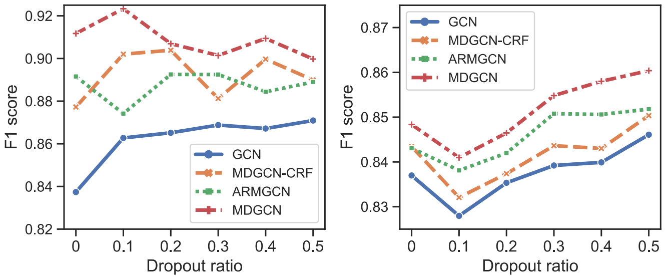
Fig. 6. The effect of the hidden units on F_1 score.

the dropout ratio, and the number of epochs in each iteration of EM algorithm. All models here are trained using 100% training data.

First, we consider the effect of hidden units of GCN, as shown in Fig. 6. As the hidden layer unit goes from 8 to 512, GCN variants have slight fluctuations in the performance concerning F_1 score, mostly within 0.01. It is worth mentioning that the approximate best level can be reached when the hidden layer unit is only 8, which fully reflects the powerful feature expression ability of GCN.

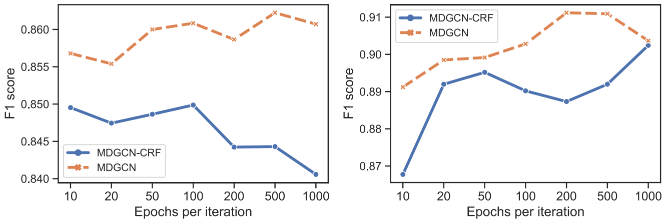
Second, we consider the sensitivity of Dropout ratio on performance, as shown in Fig. 7. When Dropout ratio increases from 0 to 0.5, there is no obvious rule in the performance of TwitterSH, but the fluctuation is small. In 1KS-10KN, the F_1 score first decreases and then increases, and it performs best when the Dropout is 0.5. This shows that Dropout can alleviate overfitting to the mode in unbalanced datasets and improve detection performance.

Third, we analyze the impact of the number of epochs in each iteration of the EM algorithm, as shown in Fig. 8. Here we still fix the total epochs of training to 2000. First, we focus on MDGCN-CRF. As epochs increase from 10 to 1000, the number of iterations of EM algorithm decreases from 200 to 2, making the performance in TwitterSH gradually decline. For 1KS-10KN, which is more difficult to train, it is necessary to ensure sufficient training in each iteration. Therefore, with the increase of epochs, its performance can be improved gradually. The two different trends reflect the difficulty of dataset training. Second, for MDGCN, it can be clearly observed that the corresponding trend is small compared to MDGCN-CRF, because the ARMRF layer is helpful for training, which alleviates the effect of dataset.



(a) 1KS-10KN (b) TwitterSH

Fig. 7. The effect of the Dropout ratio on F_1 score.



(a) TwitterSH (b) 1KS-10KN

Fig. 8. The effect of the number of iterations of the EM algorithm on the F_1 score.

5.5 Time Complexity and Run-Time Analysis

We first analyze the time complexity of Algorithm 1 (ARMGCN). The time complexity of GCN is $O(|E|m + |V|mh + L|E|h + L|V|h^2)$, where $|V|$ and $|E|$ are the numbers of nodes and edges, m and h are the dimensions of inputs and hidden states, L is the number of GCN layers. The time complexity of the proposed ARMRF with T iterations is $O(T|E|)$. In practice, the imapct of the ARMRF layer is insignificant since T is small. Second, we analyze the network Q (E-step) and network P (M-step) in Algorithm 1 (MDGCN), respectively. For each iteration, the E-step cost $O(|E|m + |V|mh + L|E|h + L|V|h^2 + T|E|)$ and the M-step costs $O(L|E|h + L|V|h^2 + T|E|)$. Therefore, the per-iteration time complexity of MDGCN is that twice of GCN.

Furthermore, we compare the proposed MDGCN and its variants with GCN in terms of training speed. Table 5 shows the running times of different models training 2400 epochs. First, by comparing the GCN and ARMGCN, we can observed that the impact of the ARMRF layer on running time is small and positively associated with the number of nodes. As analyzed in Section 4, network P (M-step) and network Q (E-step) are two different ARMGCNs; thus the per-iteration time complexity of the EM algorithm is that twice of GCN. As shown in Table 5, comparing GCN and MDGCN-CRF, ARMRF and MDGCN, it can be seen that the EM algorithm results in double the running time.

5.6 Visualization

To intuitively demonstrate the quality of the user embeddings, we use the t-SNE tool [50] to visualize the learned latent user representations of different models on the TwitterSH dataset. All models are trained using 60% of all data.

TABLE 5
Comparison of Running Time(s)

Dataset	GCN	MDGCN-CRF	ARMGCN	MDGCN
TwitterSH	42.8	65.9	44.5	85.4
1KS-10KN	123.1	245.5	180.8	372.8

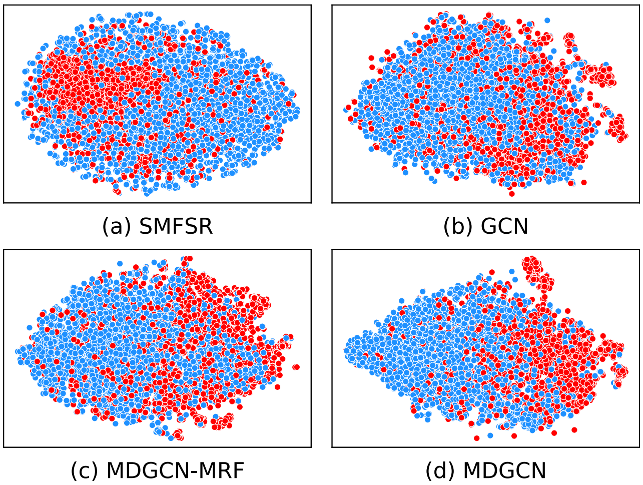

(a) SMFSR
(b) GCN
(c) MDGCN-MRF
(d) MDGCN

Fig. 9. The t-SNE visualization of latent representations.

For GCN, ARMGCN, and MDGCN, we use the hidden layer outputs as the features, while for SMRSR, we use the user embedding of matrix factorization to perform the t-SNE visualization. For a fair comparison, the hidden units of the GCN variants are all 16, as same as the baseline SMFSR. We use the sklearn module for this experiment, and all parameters use the default settings. The results are shown in Fig. 9, where the red points represent spam users and the blue points represent legitimate users. We can observe that although the spam embedding of SMRSR can be effectively clustered, it cannot be distinguished in the legitimate embeddings, which leads to its poor performance. Relatively speaking, the spam embeddings generated by various GCN variants are closer to the boundary, which have more potential to distinguish. Among them, the embedding produced by GCN is slightly weaker than ARMGCN and MDGCN, and MDGCN performs best among them.

6 Conclusion

In this paper, we study how to combine GCN and MRF for social spammer detection. First, we propose an adaptive reward MRF (ARMRF) layer to use MRF to capture users' insights on different social relationships. Stack the ARMRF layer on the top of GCN, which we call ARMGCN, to correct GCN prediction errors. Furthermore, we merge ARMGCN and CRF to propose a Markov-driven Graph Convolutional Network (MDGCN). MDGCN uses CRF to model the joint probability distribution of user label conditioned features to further learn label dependence. This method incorporates the GCN and CRF to learn the users' feature and neighbor information. The extensive evaluations on two real-world Twitter datasets show that the proposed method outperforms state-of-the-art methods. In addition, the ARMRF layer is decoupled from the model, so it can be used as an independent refinement module for the existing detection model, which is also proved by the experimental results.

References

[1] I. Anger and C. Kittl, "Measuring influence on Twitter," in *Proc. 11th Int. Conf. Knowl. Manage. Knowl. Technol.*, 2011, pp. 1–4.

[2] A. Nadkarni and S. G. Hofmann, "Why do people use Facebook?," *Pers. Indiv. Differ.*, vol. 52, no. 3, pp. 243–249, 2012.

[3] Y. Zhu, X. Wang, E. Zhong, N. N. Liu, H. Li, and Q. Yang, "Discovering spammers in social networks," in *Proc. 26th AAAI Conf. Artif. Intell.*, 2012, pp. 171–177.

[4] X. Hu, J. Tang, Y. Zhang, and H. Liu, "Social spammer detection in microblogging," in *Proc. 23rd Int. Joint Conf. Artif. Intell.*, 2013, pp. 2633–2639.

[5] X. Hu, J. Tang, H. Gao, and H. Liu, "Social spammer detection with sentiment information," in *Proc. IEEE Int. Conf. Data Mining*, 2014, pp. 180–189.

[6] X. Hu, J. Tang, and H. Liu, "Online social spammer detection," in *Proc. 28th AAAI Conf. Artif. Intell.*, 2014, pp. 59–65.

[7] H. Shen, F. Ma, X. Zhang, L. Zong, X. Liu, and W. Liang, "Discovering social spammers from multiple views," *Neurocomputing*, vol. 225, no. C, pp. 49–57, Feb. 2017. [Online]. Available: https://doi.org/10.1016/j.neucom.2016.11.013

[8] C. Li, S. Wang, L. He, P. S. Yu, Y. Liang, and Z. Li, "SSDMV: Semi-supervised deep social spammer detection by multi-view data fusion," in *Proc. IEEE Int. Conf. Data Mining*, 2018, pp. 247–256.

[9] B. Wang, L. Zhang, and N. Z. Gong, "SybilSCAR: Sybil detection in online social networks via local rule based propagation," in *Proc. IEEE Conf. Comput. Commun.*, 2017, pp. 1–9.

[10] B. Wang, N. Z. Gong, and H. Fu, "GANG: Detecting fraudulent users in online social networks via guilt-by-association on directed graphs," in *Proc. IEEE Int. Conf. Data Mining*, 2017, pp. 465–474.

[11] Y. Wu, D. Lian, Y. Xu, L. Wu, and E. Chen, "Graph convolutional networks with Markov random field reasoning for social spammer detection," in *Proc. AAAI Conf. Artif. Intell.*, 2020, pp. 1054–1061.

[12] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *Proc. Int. Conf. Learn. Representations*, 2016, pp. 1–14.

[13] M. Qu, Y. Bengio, and J. Tang, "GMNN: Graph Markov neural networks," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 5241–5250.

[14] H. Zheng *et al.*, "Smoke screener or straight shooter: Detecting elite sybil attacks in user-review social networks," 2017, *arXiv: 1709.06916*.

[15] K. Thomas, F. Li, C. Grier, and V. Paxson, "Consequences of connectivity: Characterizing account hijacking on Twitter," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, 2014, pp. 489–500.

[16] K. Thomas, C. Grier, D. Song, and V. Paxson, "Suspended accounts in retrospect: An analysis of Twitter spam," in *Proc. ACM SIGCOMM Conf. Internet Meas. Conf.*, 2011, pp. 243–258.

[17] G. Wang, T. Konolige, C. Wilson, X. Wang, H. Zheng, and B. Y. Zhao, "You are how you click: Clickstream analysis for sybil detection," in *Proc. 22nd USENIX Secur. Symp.*, 2013, pp. 241–256.

[18] Q. Cao, X. Yang, J. Yu, and C. Palow, "Uncovering large groups of active malicious accounts in online social networks," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, 2014, pp. 477–488.

[19] B. Viswanath *et al.*, "Towards detecting anomalous user behavior in online social networks," in *Proc. 23rd USENIX Secur. Symp.*, 2014, pp. 223–238.

[20] H. Fu, X. Xie, Y. Rui, N. Z. Gong, G. Sun, and E. Chen, "Robust spammer detection in microblogs: Leveraging user carefulness," *ACM Trans. Intell. Syst. Technol.*, vol. 8, no. 6, pp. 83:1–83:31, Aug. 2017.

[21] X. He, Q. Gong, Y. Chen, Y. Zhang, X. Wang, and X. Fu, "DatingSec: Detecting malicious accounts in dating apps using a content-based attention network," *IEEE Trans. Dependable Secure Comput.*, vol. 18, no. 5, pp. 2193–2208, Sep./Oct. 2021.

[22] Y. Feng, J. Li, L. Jiao, and X. Wu, "Towards learning-based, content-agnostic detection of social bot traffic," *IEEE Trans. Dependable Secure Comput.*, vol. 18, no. 5, pp. 2149–2163, Sep./Oct. 2021.

[23] Q. Cao and X. Yang, "SybilFence: Improving social-graph-based sybil defenses with user negative feedback," 2013, *arXiv:1304.3819*.

[24] G. Danezis and P. Mittal, "SybilInfer: Detecting sybil nodes using social networks," in *Proc. Netw. Distrib. Syst. Secur. Symp.*, 2009, pp. 1–15.

[25] Q. Cao, M. Sirivianos, X. Yang, and T. Pregueiro, "Aiding the detection of fake accounts in large scale social online services," in *Proc. 9th USENIX Symp. Netw. Syst. Des. Implementation*, 2012, pp. 197–210.

[26] C. Yang, R. Harkreader, J. Zhang, S. Shin, and G. Gu, "Analyzing spammer's social networks for fun and profit: A case study of cyber criminal ecosystem on twitter," in *Proc. 21st Int. Conf. World Wide Web*, 2012, pp. 71–80.

[27] Z. Gyongyi, H. Garcia-Molina , and J. Pedersen, "Combating web spam with trustrank," in *Proc. 30th Int. Conf. Very Large Data Bases*, 2004, pp. 576–587.

[28] N. Z. Gong, M. Frank, and P. Mittal, "SybilBelief: A semi-supervised learning approach for structure-based sybil detection," *IEEE Trans. Inf. Forensics Secur.*, vol. 9, no. 6, pp. 976–987, Jun. 2014.

[29] B. Wang, J. Jia, and N. Z. Gong, "Graph-based security and privacy analytics via collective classification with joint weight learning and propagation," 2018, *arXiv:1812.01661*.

[30] S. Rayana and L. Akoglu, "Collective opinion spam detection: Bridging review networks and metadata," in *Proc. 21th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2015, pp. 985–994.

[31] J. Pearl, *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Amsterdam, The Netherlands: Elsevier, 2014.

[32] K. Murphy, Y. Weiss, and M. I. Jordan, "Loopy belief propagation for approximate inference: An empirical study," 2013, *arXiv:1301.6725*.

[33] F. Zhang, X. Hao, J. Chao, and S. Yuan, "Label propagation-based approach for detecting review spammer groups on e-commerce websites," *Knowl.-Based Syst.*, vol. 193, 2020, Art. no. 105520.

[34] G. Wang, S. Xie, B. Liu, and P. S. Yu, "Review graph based online store review spammer detection," in *Proc. IEEE 11th Int. Conf. Data Mining*, 2011, pp. 1242–1247.

[35] R. M. Neal and G. E. Hinton, "A view of the EM algorithm that justifies incremental, sparse, and other variants," in *Learn. in Graphical Models*. Berlin, Germany: Springer, 1998, pp. 355–368.

[36] M. D. Hoffman, D. M. Blei, C. Wang, and J. Paisley, "Stochastic variational inference," *J. Mach. Learn. Res.*, vol. 14, pp. 1303–1347, 2013.

[37] J. Besag, "Statistical analysis of non-lattice data," *J. Roy. Statist. Soc. Ser. Statistician*, vol. 24, no. 3, pp. 179–195, 1975.

[38] K. Lee, B. D. Eoff, and J. Caverlee, "Seven months with the devils: A long-term study of content polluters on twitter," in *Proc. 5th Int. AAAI Conf. Weblogs Soc. Media*, 2011, pp. 185–192.

[39] H. Kwak, C. Lee, H. Park, and S. Moon, "What is twitter, a social network or a news media?," in *Proc. 19th Int. Conf. World Wide Web*, 2010, pp. 591–600.

[40] C. Chen, J. Zhang, X. Chen, Y. Xiang, and W. Zhou, "6 million spam tweets: A large ground truth for timely twitter spam detection," in *Proc. IEEE Int. Conf. Commun.*, 2015, pp. 7065–7070.

[41] S. Sedhai and A. Sun, "HSpam14: A collection of 14 million tweets for hashtag-oriented spam research," in *Proc. 38th Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2015, pp. 223–232.

[42] M. Mccord and M. Chuah, "Spam detection on twitter using traditional classifiers," in *Proc. Int. Conf. Autonomic Trusted Comput.*, 2011, pp. 175–186.

[43] Y. Dou, Z. Liu, L. Sun, Y. Deng, H. Peng, and P. S. Yu, "Enhancing graph neural network-based fraud detectors against camouflaged fraudsters," in *Proc. 29th ACM Int. Conf. Inf. Knowl. Manage.*, 2020, pp. 315–324.

[44] K. Xu, W. Hu, J. Leskovec, and S. Jegelka, "How powerful are graph neural networks?" 2018, *arXiv:1810.00826*.

[45] Y. Wang, J. Zhang, S. Guo, H. Yin, C. Li, and H. Chen, "Decoupling representation learning and classification for GNN-based anomaly detection," in *Proc. 44th Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2021, pp. 1239–1248.

[46] P. Veličković, W. Fedus, W. L. Hamilton, P. Liò, Y. Bengio, and R. D. Hjelm, "Deep graph infomax," in *Proc. Int. Conf. Learn. Representations*, 2019, pp. 1–17.

[47] J. Davis and M. Goadrich, "The relationship between precision-recall and ROC curves," in *Proc. 23rd Int. Conf. Mach. Learn.*, 2006, pp. 233–240.

[48] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 1929–1958, 2014.

[49] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. Int. Conf. Learn. Representations*, 2015, pp. 1–15.

[50] L. van der Maaten and G. Hinton, "Visualizing data using t-SNE," *J. Mach. Learn. Res.*, vol. 9, no. Nov., pp. 2579–2605, 2008.

Leyan Deng received the BS degree in mathematics from Shandong University, Jinan, China, in 2019. She is currently working toward the ME degree with the School of Data Science, University of Science and Technology of China, Hefei, China. Her current research interests include anomaly detection and spatial-temporal data mining.

Chenwang Wu received the BS degree from the College of Mathematics and Computer Science, Fuzhou University, Fuzhou, China, in 2018. He is currently working toward the PhD degree with the School of Data Science, University of Science and Technology of China. His current research interests include recommender systems and deep learning.

Defu Lian received the PhD degree in computer science from the University of Science and Technology of China (USTC), Hefei, China, in 2014. He is currently a professor with the School of Computer Science and Technology, USTC. He has authored or coauthored prolifically in referred journals and conference proceedings, such as *ACM Transactions on Intelligent Systems and Technology*, *ACM Transactions on Information Systems*, and *IEEE Transaction on Knowledge and Data Engineering*, AAAI Conference on Artificial Intelligence, IEEE International Conference on Data Mining (ICDM), ACM SIGKDD Conference on Knowledge Discovery and Data Mining, ACM International Conference on Research on Development in Information Retrieval, International Joint Conferences on Artificial Intelligence, and ACM International World Wide Web Conferences. His current research interests include spatial data mining, recommender systems, and learning to hash.

Yongji Wu received the bachelor's degree from the University of Science and Technology of China in 2020. He is currently working toward the PhD degree with the Department of Computer Science, Duke University. His main research interests include machine learning, data mining, recommender system, security and privacy issues in recommender systems and social networks.

Enhong Chen (Senior Member, IEEE) received the PhD degree in computer science from the University of Science and Technology of China (USTC), Hefei, China, in 1996. He is currently a professor and the vice dean of the School of Computer Science, USTC. He has authored or coauthored more than 200 papers in refereed conferences and journals, including *IEEE Transactions on Knowledge and Data Engineering*, *IEEE Transactions on Mobile Computing*, ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD), IEEE International Conference on Data Mining (ICDM), Conference on Neural Information Processing Systems, and ACM International Conference on Information and Knowledge Management. His current research interests include data mining and machine learning, social network analysis, and recommender systems. Dr. Chen was the recipient of the Best Application Paper Award on KDD2008, Best Research Paper Award on ICDM-2011, and Best of SIAM International Conference on Data Mining (SDM)-2015. He was on the program committees of numerous conferences, including KDD, ICDM, and SDM.