Hindawi
Security and Communication Networks
Volume 2022, Article ID 4354086, 13 pages
https://doi.org/10.1155/2022/4354086

Research Article

Network Embedding-Based Approach for Detecting Collusive Spamming Groups on E-Commerce Platforms

Jinbo Chao,[1,2] **Chunhui Zhao**,[1,2] **and Fuzhi Zhang**[1,2]

[1]*School of Information Science and Engineering, Yanshan University, Qinhuangdao, Hebei, China*
[2]*The Key Laboratory for Computer Virtual Technology and System Integration of Hebei Province, Qinhuangdao, Hebei, China*

Correspondence should be addressed to Fuzhi Zhang; xjzfz@ysu.edu.cn

Received 17 September 2021; Accepted 21 December 2021; Published 11 January 2022

Academic Editor: Chunhua Su

Information security is one of the key issues in e-commerce Internet of Things (IoT) platform research. The collusive spamming groups on e-commerce platforms can write a large number of fake reviews over a period of time for the evaluated products, which seriously affect the purchase decision behaviors of consumers and destroy the fair competition environment among merchants. To address this problem, we propose a network embedding based approach to detect collusive spamming groups. First, we use the idea of a meta-graph to construct a heterogeneous information network based on the user review dataset. Second, we exploit the modified DeepWalk algorithm to learn the low-dimensional vector representations of user nodes in the heterogeneous information network and employ the clustering methods to obtain candidate spamming groups. Finally, we leverage an indicator weighting strategy to calculate the spamming score of each candidate group, and the top-k groups with high spamming scores are considered to be the collusive spamming groups. The experimental results on two real-world review datasets show that the overall detection performance of the proposed approach is much better than that of baseline methods.

1. Introduction

With the development of Web 2.0, it has become fashionable for users to shop online [1]. E-commerce IoT platforms often collect and analyze users' shopping and review information from edge devices such as smartphones. This information, especially user reviews, directly affects the purchase decision behaviors of potential consumers, as users usually look at other users' reviews on the product before purchasing and then decide whether to buy it. However, consumers' dependence on product reviews has catalyzed the emergence of fake reviews. Driven by economic interests, some merchants tend to attract consumers to write favorable reviews for their products by presenting gifts or small vouchers. The reviews that are unrealistic advocacy or defamatory for the products of merchants to influence the purchase decision of consumers are called fake reviews [2]. Fake reviews are very deceptive and many consumers are misled by such reviews [1]. Therefore, identifying the authenticity of user reviews has become an important research topic in the field of e-commerce IoT information security.

To make higher profits, some merchants might hire a group of reviewers working together to promote their products by fabricating product reviews. A reviewer who writes fake reviews is termed a spammer, and a reviewer group whose members collude to fabricate product reviews is called a collusive spamming group [3, 4]. The collusive spamming groups are more harmful to e-commerce platforms than individual spammers because they can produce fake reviews in batches for a period of time and will affect or even control the reputation of the evaluated products [5]. Therefore, how to identify the collusive spamming groups on e-commerce platforms has become an urgent problem to be solved.

To detect the collusive spamming groups on e-commerce platforms, a variety of approaches have been proposed. These approaches can be roughly categorized as group content and behavior analysis-based methods and graph-based methods [5, 6]. Group content and behavior analysis-based detection methods [3, 4, 7–9] employ the frequent itemset mining (FIM) technique to discover candidate spamming groups, which are only applicable to detecting

tightly coupled spamming groups. Moreover, FIM-based candidate group discovery methods are sensitive to the setting of support threshold. Graph-based detection methods [10–16] need to construct a user relationship graph by mining the relationships between reviewers in the dataset. However, these methods do not deeply mine the hidden relationships between reviewers when constructing the user relationship graph, which cannot fully detect the collusive spamming behaviors between reviewers. Furthermore, these methods tend to discard users whose correlation strength is less than a certain threshold during the process of constructing the user relationship graph, thus causing a negative impact on the detection performance.

To address the limitations above, we propose an approach for collusive spamming group detection based on network embedding, which is named CSGD-NE. We first model the given user review dataset as a heterogeneous information network (HIN) based on the idea of meta-graph. Then, we use the modified DeepWalk algorithm to perform random walks on the HIN to learn the low-dimensional vector representations of user nodes and employ the Canopy and K-means clustering algorithms to generate candidate spamming groups. Finally, we obtain the top-k groups with high spamming scores as the collusive spamming groups.

The contributions of this paper are summarized below:

(1) Borrowing from the idea of meta-graph, we extract the key attributes such as users, products, and ratings required by the meta-graph from the given user review dataset to construct a HIN, which preserves more abundant structural and semantic information.

(2) We modify the DeepWalk algorithm to learn the low-dimensional, dense, and real-value vector representations of user nodes in the HIN. Based on this, we perform the Canopy and K-means clustering algorithms in the embedding space to obtain candidate spamming groups.

(3) We propose three indicators to measure the group spamming behaviors and combine the proposed indicators with two existing indicators to calculate the spamming score of each candidate group by using an indicator weighting strategy.

(4) To show the performance of the proposed approach, we conduct extensive experiments on two real-world review datasets and compare our approach with four baseline methods.

The remainder of this paper is organized as follows. Section 2 introduces the related work on collusive spamming group detection. Section 3 describes the proposed approach, which includes the construction of HIN, generation of candidate collusive spamming groups, and detection of collusive spamming groups. The experimental results are reported in Section 4, and conclusions are drawn in Section 5.

2. Related Work

To detect individual spammers on e-commerce platforms, researchers have done in-depth work [5, 17–20]. In recent years, some researchers have focused on the collusive spamming behaviors among reviewers and tried to detect collusive spamming groups on e-commerce platforms. In this section, we will discuss the related studies on collusive spamming group detection following the previously mentioned two categories. Table 1 summarizes the methods for collusive spamming group detection in line with these two categories.

2.1. Group Content and Behavior Analysis-Based Detection Methods. In the group content and behavior analysis-based detection methods, the correlation between group members is mined by analyzing the group review contents and behavior characteristics, and then the collusive spamming groups are identified through ranking or clustering methods. Mukherjee et al. [3] proposed the concept of the spamming group; they employed the FIM technique to discover candidate spamming groups and evaluated these candidate groups by analyzing the group members' review contents and the group members' unusual behaviors, and used a ranking method to obtain the collusive spamming groups. Xu et al. [4] calculated pairwise similarity between reviewers based on the review features and behavior features of reviewers and employed the k-nearest neighbors' method to predict top-k most similar reviewers. Xu and Zhang [7] presented an unsupervised framework to detect colluders, which used multiple heterogeneous pairwise features to capture the reviewers' collusive behaviors. In [8], a statistical model was proposed to model the collusive spamming behaviors among reviewers and an expectation maximization (EM) algorithm was employed to calculate the collusiveness of reviewers in the candidate groups that are obtained with the FIM technique. Zhang et al. [9] proposed a semisupervised method to identify spamming groups. They used the FIM technique to discover the candidate spamming groups and exploited the naive Bayesian model and EM algorithm to train a classifier for spamming group detection.

The abovementioned detection approaches mainly use the FIM technique to find candidate spamming groups, which are only suitable for tightly coupled spamming groups and are not applicable to loosely coupled spamming groups. Moreover, these approaches are sensitive to the setting of the support threshold. In addition, the semisupervised method needs a priori knowledge of spamming groups, which is often difficult to obtain in practice.

2.2. Graph-Based Detection Methods. Graph-based detection methods considered the specificity of collusive spamming groups in network structure and used the group structure features to identify the collusive spamming groups. Xu et al. [4] proposed a pairwise Markov network-based graph model to detect colluders, which took reviewers as nodes to construct a colluder graph model and used a probability graph model to predict the collusive spamming groups. Ye and Akoglu [10] used a 2-hop subgraph to detect the unusual behaviors of reviewers by analyzing their network footprints and employed the hierarchical clustering method to discover opinion spammer groups. Choo et al. [11] built user

Table 1: Summary of methods for collusive spamming group detection.

Category	Method	Data modeling	Candidate group discovery method	Spamming behavior evaluation	Detection method
Group content and behavior analysis-based methods	Mukherjee et al. [3]	Construct behavioral and relation models	Frequent itemset mining	The average of spam indicators	Ranking
	Xu et al. [4]	Measure the pairwise similarity of groups	Frequent itemset mining	No	KNN-based method
	Xu and Zhang [7]	Use pairwise features to model the relations among colluders	No	The weighted sum of all the pairwise features	Ranking
	Xu and Zhang [8]	Use homogeneity-based behavior features to model the collusive review fraud	Frequent itemset mining	Similarity-based measure	Unified probabilistic model
	Zhang et al. [9]	Use a number of features to model the group spamming behavior	Frequent itemset mining	No	Semi-supervised classification
Graph-based methods	Ye and Akoglu [10]	Model the dataset as a bipartite graph	No	Reviewers' network footprint scores	Hierarchical clustering
	Choo et al. [11]	Model the dataset as a user relationship graph	No	No	Community-based method
	Wang et al. [12]	Model the dataset as a bipartite graph	Graph partitioning	The average of spam indicators	Ranking
	Do et al. [13]	Model the dataset as a review graph	No	The average of spam indicators	k-means clustering
	Do et al. [14]	Model the dataset as a review graph	No	The average of spam indicators	Fuzzy k-means clustering
	Han et al. [15]	Model the dataset as a user relationship graph	No	No	Graph spectrum analysis
	Wang et al. [16]	Model the dataset as a user relationship graph	Graph partitioning	The average of spam indicators	Ranking
	Cao et al. [21]	Model the dataset as a bipartite graph	No	The average of spam indicators	Hierarchical agglomerative clustering
	Zhang et al. [22]	Model the dataset as a user relationship graph	Label propagation	Linearly weighted sum of spam indicators	Ranking

relationship graphs through the sentiment analysis of user interactions and used the strong positive communities of spammers to detect opinion spammer groups. Wang et al. [12] used a divide and conquer strategy to discover candidate spamming groups from the constructed reviewer relationship graph and introduced several spam indicators for detecting the review spammer groups. Do et al. [13] presented a review graph-based method to detect group spamming, which assigned a suspicious score to each user and employed the k-means algorithm to obtain the spamming groups. In [14], a network-based approach was utilized to identify individual spammers, and then a fuzzy k-means clustering algorithm was employed to find the group to which they belong. Han et al. [15] proposed an approach for detecting spammer groups based on graph spectrum analysis, which identified the anomaly structure in the constructed user relationship graph through analyzing the spectrum features. Wang et al. [16] used the minimum cut algorithm to split the user relationship graph to obtain candidate groups and employed a set of spam indicators to detect the spammer groups. Cao et al. [21] extracted the multiview anomalous features of collusive spammers and employed a hierarchical clustering algorithm to detect the spammer groups in location-based social networks. In our recent work [22], we used a Label propagation-based approach for detecting review spammer groups.

The above detection approaches are mainly based on the homogeneous graph. As these methods do not deeply mine the implicit relationship between reviewers in the review dataset when constructing the user relationship graph, they cannot fully detect the collusive spamming behaviors between reviewers. Moreover, these methods tend to discard some users during the construction of the user relationship graph, causing a negative impact on the detection performance. In addition, these methods take the average of all spam indicator values as the suspicious score of the candidate group, ignoring the difference among spam indicators.

3. The Proposed Detection Approach

The framework of the proposed detection approach is illustrated in Figure 1 and includes three phases. Figure 2 gives the scenario diagram for our detection approach. In the first phase, the key attributes such as users, products, and ratings required by the meta-graph are extracted from the user review dataset to construct a HIN. In the second phase, the modified DeepWalk algorithm is employed to perform equal probability

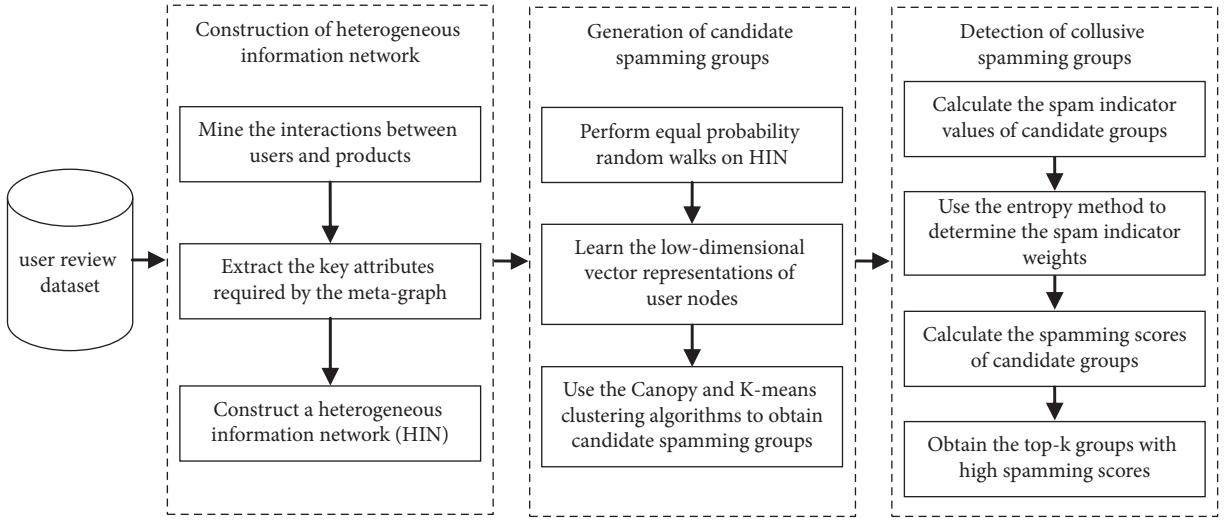

Figure 1: The framework of the proposed detection approach.

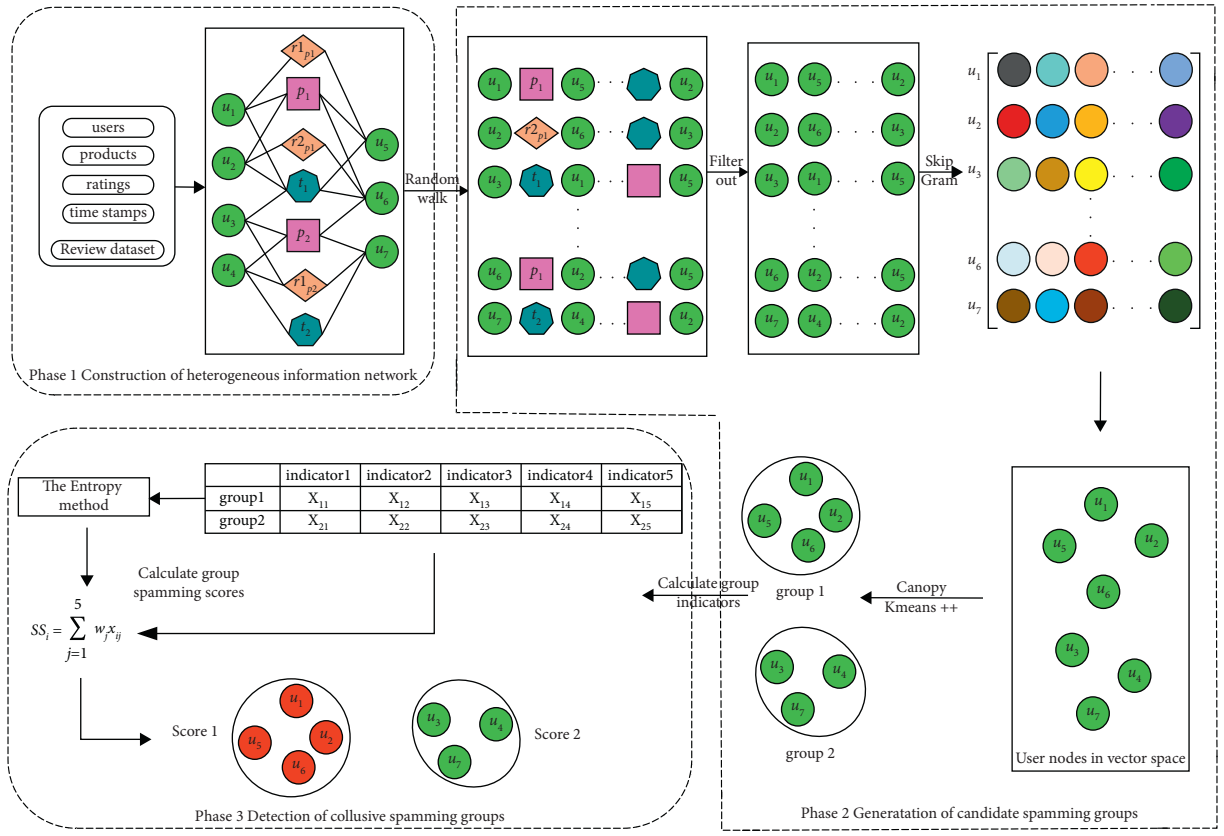

Figure 2: The scenario diagram of the proposed detection approach.

random walks on the HIN to learn the low-dimensional vector representations of user nodes, and the Canopy and K-means clustering algorithms are applied to generate candidate spamming groups. In the last phase, the spamming score of each candidate group is calculated by weighting all spam indicator values, and the top-k groups with high spamming scores are treated as the collusive spamming groups.

3.1. Construction of Heterogeneous Information Network. In this section, we model the given user review dataset as a HIN. Particularly, we extract the key attributes, including users, products, ratings, and timestamps required by the meta-graph from the user review dataset, and construct the HIN based on the idea of meta-graph. First, we introduce some concepts related to the HIN.

Definition 1 (heterogeneous information network [23]). Let V, E, A, and $\mathscr{R}$ denote sets of nodes, edges, node types, and edge types, respectively; a HIN is defined as a graph $G = (V, E)$ with a node type mapping function $\phi: V \longrightarrow A$ and an edge type mapping function $\psi: E \longrightarrow \mathscr{R}$, where each node $v \in V$ belongs to one particular node type $\phi(v) \in A$; each edge $e \in E$ belongs to one particular relation $\psi(e) \in \mathscr{R}$, and at least one of $|A|$ and $|\mathscr{R}|$ is greater than 1.

Definition 2 (network schema [24]). Given a HIN $G = (V, E)$ with two mapping functions $\phi: V \longrightarrow A$ and $\psi: E \longrightarrow \mathscr{R}$, the network schema of G is a directed graph defined over node types A and edge types $\mathscr{R}$, which is denoted as $T_G = (A, \mathscr{R})$.

The network schema of a HIN serves as a template for the HIN, indicating how many objects are in the HIN and whether links may exist between these objects.

Definition 3 (meta-path [24]). Given a HIN $G = (V, E)$ with a network schema $T_G = (A, \mathscr{R})$, a meta-path P is a path represented in the form of $P = A_1 \longrightarrow^{R_1} A_2 \longrightarrow^{R_2} \cdots \longrightarrow^{R_l} A_{l+1}$, where $A_i \in$ A and R_i denotes the relation between node types A_i and A_{i+1}.

A meta-path represents the relationship between nodes in a HIN, and different types of meta-paths represent different meanings between nodes.

Definition 4 (meta-graph [25]). Given a HIN $G = (V, E)$ with a network schema $T_G = (A, \mathscr{R})$, a meta-graph $S = (N, M, n_s, n_t)$ is a directed acyclic graph defined on T_G, where $N \subseteq V$ and $M \subseteq E$, n_s and n_t are the source and target nodes, respectively.

Unlike meta-paths, meta-graphs have a more complex and flexible structure, which can be used to represent more complex relationships. Based on the idea of meta-graph, the steps for constructing the HIN from the given user review dataset are as follows.

First, we extract the key attributes and mine the interactions between users and products from the review dataset to construct a meta-graph. The extracted attributes include users, products, ratings, and timestamps, which are treated as user nodes, product nodes, rating nodes, and time nodes in the HIN. To avoid ambiguity, the four types of nodes are renumbered uniformly in this paper. The rating node contains the product ID information because the product reviewed by the user may not be the same when the user gives the same rating. Based on the extracted attributes and interactions, we can construct three meta-paths, i.e., user-product-user, user-rating-user, and user-time-user, which represent that two users reviewed the same product, gave the same rating for a product, and reviewed a product at the same time, respectively. A meta-graph can be built by connecting the three meta-paths together.

Second, we construct the HIN by extending the meta-graph. Data objects in the given user review dataset can be connected through the meta-graph and not just a single path. Even if two users have not reviewed the same product, there may be a correlation between them if their review time is the same. Therefore, the meta-graph can be used to explore the potential relationship between users from three products, rating, and review time.

3.2. Generation of Candidate Spamming Groups. In this section, we modify the DeepWalk algorithm to learn the low-dimensional vector representations of user nodes in the HIN and perform clustering in the embedding space to obtain candidate spamming groups.

3.2.1. The Modified DeepWalk Algorithm. DeepWalk [26] is a network embedding method based on random walks and skip-gram modeling. Based on the idea of Word2vec [27], it takes the sequence of nodes obtained by random walks as a sentence, obtains the local information of the network from the truncated random walk sequence, and then learns the latent representations of nodes through the local information.

DeepWalk is applicable to learning the representations of nodes in the homogeneous graph, while the HIN we construct in this paper belongs to a heterogeneous graph that includes four types of nodes (i.e., user node, product node, rating node, and time node). Therefore, we need to modify the DeepWalk algorithm to make it suitable for learning the representations of user nodes in the constructed HIN in order to divide candidate spamming groups. The main steps of the modified DeepWalk algorithm are as follows.

Step 1. Each user node is taken as the starting node in turn, and any node is selected from the set of adjacent nodes of the current node to walk randomly with equal probability.

Step 2. At the end of each random walk, the node type is judged. If the node belongs to a user node, it will be added to the random walk sequence, and then the next random walk will be conducted.

Step 3. Repeat the above steps until the length of the random walk reaches a given threshold. Put the obtained sequence of user nodes into the skip-gram model for training and generate the user node vector representation.

Based on the above steps, the modified DeepWalk algorithm is described below.

Algorithm 1 generates the low-dimensional vector representations of user nodes in the HIN through equal probability random walks. Particularly, it first takes each user node as the starting node for random walks and obtains the walk sequence composed of all types of nodes (Lines 4–6). Then the user nodes are extracted from the walk sequence W_{vi} and the user node sequence S_u is generated (Lines 7–12). Finally, the generated user node sequence S_u is used as the input of the skip-gram model and the vector representation of each user node is obtained (Line 13).

Input: HIN G, window size ws, embedding size d, the number of walks per node n, walk length l, set of user nodes V_u
Output: matrix of user node representations $\Phi \in \mathbb{R}^{|V_u| \times d}$
(1) Initialization: Sample Φ from $U^{|V_u| \times d}$
(2) Build a binary tree from V_u
(3) **for** $i = 0$ to n **do**
(4) $O \leftarrow$ Shuffle (V_u)
(5) **for** each $v_i \in O$ **do**
(6) $W_{vi} \leftarrow$ RandomWalk (G, v_i, l)
(7) $S_u \leftarrow \varnothing$
(8) **for** each node $v \in W_{vi}$ **do**
(9) **if** $v \in V_u$ **then**
(10) add v to S_u
(11) **end if**
(12) **end for**
(13) SkipGram (Φ, S_u, ws)
(14) **end for**
(15) **end for**
(16) **return** Φ

Algorithm 1: The modified DeepWalk algorithm.

3.2.2. Generating Candidate Spamming Groups Based on the Canopy and K-Means Clustering. Based on the user vector representations obtained with the modified DeepWalk algorithm, we employ a combination of the Canopy clustering and the K-means clustering to generate candidate spamming groups. Canopy clustering [28] is a fast approximate clustering algorithm, which does not need to specify the number of clusters in advance. While this clustering algorithm cannot produce accurate clustering results, it can give the optimal number of clusters. The K-means clustering is a simple, efficient, and effective clustering algorithm, but it has limitations such as sensitivity to the selection of initial cluster centers and need to specify the number of clusters in advance. Therefore, we can use the Canopy clustering algorithm as the preprocessing of the K-means clustering algorithm to determine the optimal number of clusters and the best initial cluster centers for the K-means clustering.

The basic idea of generating candidate spamming groups based on the Canopy and K-means clustering algorithms is as follows. First, we treat the user vector representations as a set of sample objects and employ the Canopy clustering algorithm to produce a number of Canopy sets (or clusters). Second, we take the number of clusters and the cluster centers obtained with the Canopy clustering algorithm as the cluster number K and initial cluster centers of the K-means algorithm to perform clustering on the set of sample objects to generate the candidate spamming groups. The specific algorithm is described as follows.

Algorithm 2 first calculates the average Euclidean distance between sample objects (Lines 2–7), which is used as a distance threshold of the Canopy algorithm, where Function calculateEuclideanDistance () is used to calculate the Euclidean distance between two objects. Then it performs the Canopy clustering algorithm to obtain the canopy sets and cluster centers (Lines 8–23), where Function getClusterCenter () is used to obtain the center of a cluster. Finally, the number of clusters $|C|$ and the cluster centers are used as the cluster number and the initial cluster centers of the K-means clustering algorithm to generate the candidate spamming groups (Line 24).

3.3. Detection of Collusive Spamming Groups. In this section, we propose three spamming group detection indicators and combine them with two existing detection indicators to calculate the spamming scores of the candidate groups. Based on this, we rank the candidate groups according to their spamming scores and consider the top-k most suspicious groups as the collusive spamming groups.

3.3.1. The Spamming Group Detection Indicators. For a product, if most of the product's review information comes from the same group, it is more likely that the merchant hires a spamming group for profit. If there exists a significant difference in the review ratio of products within and outside a group, then it is likely that the group is hired by a merchant, and the more likely it is to be a spamming group.

Definition 5. (review ratio of products within and outside a group, *RRP*): The review ratio of products within and outside a group g refers to the mean of review ratio values of products within and outside group g, whose normalized value is calculated as follows:

$$RRP(g) = \left(\frac{2}{1 + e^{-1/\left|P_g\right| \sum_{p \in P_g} \left|R_{p,g}\right| / \left|R_{p,D}\right|}} - 1 \right) \times L(g), \quad (1)$$

where P_g denotes the set of products that are reviewed by the reviewers in group g, $R_{p,D}$ and $R_{p,g}$ denote the sets of reviewers who have reviewed product p in the dataset D and in group g, respectively. $L(g)$ is used to reduce the possibility

```
Input: set of sample objects S
Output: candidate spamming groups CPG
(1) dis←0, C←∅, Cluster Centers←∅
(2) for each object o_i ∈ S do
(3)    for each object o_j ∈ S do
(4)       dis←dis + calculateEuclideanDistance(o_i, o_j)
(5)    end for
(6) end for
(7) T_2←dis÷(|S| × (|S| − 1)÷2)
(8) while S is not empty do
(9)    choose a sample object s from S to initialize a canopy set c
(10)   S←S − {s}
(11)   for each object o in S do
(12)      dis←calculateEuclideanDistance(s, o)
(13)      if dis < T_2 + 1 then
(14)         c←c ∪ {o}
(15)      end if
(16)      if dis < T_2 then
(17)         S←S − {o}
(18)      end if
(19)   end for
(20)   C←C ∪ c
(21)   Center←getClusterCenter(c)
(22)   Cluster Centers←Cluster Centers ∪ {Center}
(23) end while
(24) CPG←doK-means(S, |C|, Cluster Centers)
(25) return CPG
```

Algorithm 2: Generating the candidate spamming groups.

that group g is formed by accident, which is calculated as follows [12]:

$$L(g) = \frac{1}{1 + e^{-\left(\left|R_g\right| + \left|P_g\right| - 3\right)}}, \tag{2}$$

where R_g denotes the set of reviewers in group g and P_g has the same meaning as in equation (1).

In order to promote the target product (s), the members in a spamming group are often required to review the product (s) in a short time. Therefore, the reviews of the spamming group have the characteristics of time concentration or outburst. We can use a single-day review outburst indicator to measure this spamming behavior of a group. The higher the single-day review outburst of a group, the more likely it is to be a spamming group.

Definition 6 (single-day review outburst, SRO). The single-day review outburst of a group g refers to the average ratio of the highest number of reviews written by a user in a single day to the total number of reviews written by users in group g, whose normalized value is calculated as follows:

$$SRO(g) = \frac{1}{1 + e^{-\text{avg}_{u \in R_g} R_u^h / R_g^t}} \times L(g), \tag{3}$$

where R_u^h denotes the highest number of reviews written by user u in a single day, R_g^t denotes the total number of reviews written by the users in group g, R_g and $L(g)$ have the same meaning as in equations (2) and (1), respectively.

Generally, users in a spamming group tend to give extreme ratings for the target products, while normal reviewers tend to give relatively scattered ratings. If there is a significant difference in the rating of the target product between the users in a group and the normal reviewers, it indicates that the users in the group have strong intentionality in the product reviewed. Thus it can be inferred that the group is likely to be a spamming group. We can use the rating deviation indicator to measure this spamming behavior of a group.

Definition 7 (rating deviation, RD). The rating deviation of a group g is the mean of the products' average rating differences between users within and outside group g, whose normalized value is calculated as follows:

$$RD(g) = \frac{\text{avg}_{p \in P_g} |i\text{avg}(p) - o\text{avg}(p)|}{r_{\max} - r_{\min}} \times L(g), \tag{4}$$

where $i$avg (p) and $o$avg (p) denote the average ratings of users within and outside group g for product p, respectively, $r_{\max}$ and $r_{\min}$ denote the maximum and minimum ratings in the dataset, respectively, and P_g and $L(g)$ have the same meaning as in equation (1).

Considering the purpose of spamming group, the members in a spamming group often review the preplanned products to be promoted or demoted and rarely or never review other products. Therefore, if the members in a group review too closely on the products, the group is likely to be a

spamming group. We can use the product tightness indicator to measure this spamming behavior of a group. The greater the product tightness of a group, the more likely it is to be a spamming group.

Definition 8 (product tightness, PT [16]). The product tightness of a group g refers to the ratio of the number of products co-reviewed by the reviewers in group g to the total number of products reviewed by the reviewers in group g, that is,

$$PT(g) = \frac{\left|\cap_{u \in R_g} P_u\right|}{\left|\cup_{u \in R_g} P_u\right|}, \tag{5}$$

where P_u denotes the set of products reviewed by reviewer u and R_g has the same meaning as in equation (2).

Normal reviewers usually make an objective evaluation according to the actual situation of a product, so they may give different ratings for the same products. At the same time, reviewers in a spamming group tend to have consistent ratings on the target products to be promoted or demoted. Therefore, we can use the variance of the target product's ratings provided by the reviewers in a group to measure the spamming behavior of the group.

Definition 9 (rating variance, RV [16]). The rating variance of a group g refers to the average of rating variances of the target products and is calculated as follows:

$$RV(g) = 2\left(1 - \frac{1}{1 + e^{-avg_{p \in P_g} S^2(p,g)}}\right), \tag{6}$$

where $S^2(p, g)$ denotes the variance of product p's ratings provided by the reviewers in group g.

3.3.2. Obtaining the Collusive Spamming Groups. Based on the spamming group detection indicators described in Section 3.3.1, we can calculate a spamming score for each candidate group. Considering the differences in the importance of these indicators when measuring the spamming behavior of a group, we employ the entropy method [29,30] to determine the weights of these indicators and take the weighted sum of these indicator values as the group's spamming score.

Suppose we have m candidate groups and n detection indicators, x_{ij} $(i = 1, 2, \ldots, m;\ j = 1, 2, \ldots, n)$ denotes the j-th indicator value of the i-th group, the steps that employ the entropy method to determine the weights of these indicators are as follows.

Step 1. Calculate the normalized value of x_{ij}

$$z_{ij} = \frac{x_{ij} - \min(x_j)}{\max(x_j) - \min(x_j)}, \quad i = 1, 2, \ldots, m;\ j = 1, 2, \ldots, n, \tag{7}$$

where x_j represents the list of the j-th indicator values for each group.

Step 2. Calculate the ratio of the i-th group in the j-th indicator

$$r_{ij} = \frac{z_{ij}}{\sum_{i=1}^{m} z_{ij}}, \quad i = 1, 2, \ldots, m;\ j = 1, 2, \ldots, n. \tag{8}$$

Step 3. Calculate the j-th indicator's entropy

$$ie_j = \frac{-\sum_{i=1}^{m} r_{ij} \ln r_{ij}}{\ln m},\ j = 1, 2, \ldots, n. \tag{9}$$

Step 4. Calculate each indicator's entropy weight

$$w_j = \frac{1 - ie_j}{\sum_{j=1}^{n}(1 - ie_j)},\ j = 1, 2, \ldots, n. \tag{10}$$

The greater the entropy weight of an indicator, the more important it is in measuring the group spamming behavior.

Definition 10 (spamming score of group, SS). The spamming score of a group i is defined as the weighted sum of all indicator values of group i, that is,

$$SS_i = \sum_{j=1}^{n} w_j x_{ij},\ i = 1, 2, \ldots, m, \tag{11}$$

where x_{ij} denotes the j-th indicator value of the i-th group and w_j denotes the entropy weight of the j-th indicator. The greater the spamming score of a group, the more likely it is to be a spamming group.

We can use equation (11) to calculate the spamming scores of the candidate groups. We rank the candidate groups according to their spamming scores and take top-k groups with high spamming scores as the collusive spamming groups. The algorithm of obtaining the collusive spamming groups is described below.

Algorithm 3 consists of three parts. The first part (Lines 1–6) calculates all indicator values of the candidate spamming groups and the entropy weights of five indicators. The second part (Lines 7–12) calculates the spamming scores of the candidate groups. The third part (Line 13) obtains the top-k groups with high spamming scores.

4. Experimental Evaluation

4.1. Experimental Datasets. We use two real-world datasets as the experimental data to evaluate our approach.

(1) Amazon review dataset [4, 31]: this dataset was crawled from Amazon.cn till August 20, 2012. It includes 1205125 ratings/reviews from 645072 reviewers on 136785 products, in which 5055 reviewers have labels indicating normal users or spammers. The ratings in this dataset are integers between 1 and 5, indicating how much users like the products they have evaluated, where 1 indicates disliked and 5 indicates most liked. For the convenience of experimental evaluation, we extract these 5055 reviewers with labels, their ratings on products, and

```
Input: CSG: the candidate spamming groups
Output: TKG: the top-k groups with high spamming scores
(1)  for each group i = 1 to |CSG| do
(2)    for each indicator j = 1 to 5 do
(3)      x[i][j] ← calculate the j-th indicator value of the i-th group
(4)    end for
(5)  end for
(6)  w ← calculate the entropy weights of five indicators using equations (7)–(10)
(7)  for each group i = 1 to |CSG| do
(8)    SS[i] ← 0
(9)    for each indicator j = 1 to 5 do
(10)     SS[i] ← SS[i] + w[j] * x[i][j]
(11)   end for
(12) end for
(13) TKG ← getTopkGroups(CSG, SS)
(14) return TKG
```

Algorithm 3: Obtaining the collusive spamming groups.

their rating timestamps from the dataset to construct a sampled dataset. The sampled dataset contains 53777 ratings from 5055 reviewers on 17610 products, which includes 3118 normal users and 1937 spammers.

(2) Yelp_Miami review dataset: in [32], the authors constructed a dataset to classify fake reviews by crawling business reviews in the consumer electronics domain in four American cities (i.e., New York, Los Angeles, Miami, and San Francisco) from Yelp.com till 2017. The reviews in this dataset were labeled as fake or not by Yelp's antifraud filter, which can be treated as near ground truth. The reviewers corresponding to the fake or not reviews are considered as spammers or normal reviewers. Yelp_Miami review dataset is a subset of this dataset, which includes 3463 ratings from 3311 reviewers on 549 electronic businesses in the city of Miami. The ratings in the Yelp_Miami review dataset are integers between 1 and 5, where 1 indicates disliked and 5 indicates most liked. In the Yelp_Miami review dataset, there are 1394 spammers and 1917 normal reviewers.

4.2. Evaluation Metrics. We use precision@k (P@k) and recall@k (R@k) metrics to evaluate the performance of our approach, which are defined as follows [33]:

$$P@k = \frac{\left|\mathcal{U}_{\text{spam}} \cap \mathcal{U}_D\right|}{\left|\mathcal{U}_{spam}\right|},$$
$$R@k = \frac{\left|\mathcal{U}_{\text{spam}} \cap \mathcal{U}_D\right|}{\left|\mathcal{U}_D\right|}, \tag{12}$$

where $\mathcal{U}_{\text{spam}}$ denotes the set of top-k reviewers who are considered to be spam users and $\mathcal{U}_D$ denotes the set of spam users in the dataset D.

4.3. Experimental Results and Analysis. To illustrate the effectiveness of the proposed approach (CSGD-NE), we conduct experiments on two real-world review datasets and compare CSGD-NE with the following four methods.

(1) GSBC [16]: a graph-based spammer group detection method, which models spammer groups as bi-connected graphs and treats the bi-connected components whose spamming scores exceed the given threshold as the spammer groups. In the experiments, the parameters τ, δ, *MP*, and *MINSPAM* for GSBC are set to 30, 0.1, 1000, and 0.49 on the Amazon review dataset and 10, 0.4, 10000, and 0.59 on the Yelp_Miami review dataset, respectively.

(2) FAP [33]: a fraudulent action propagation-based method for spammer detection, which employs the label propagation method to iteratively calculate the spamming probability values of users based on the labeled seed spam users. In the experiments, we randomly select 5 spam users as the seed users. To reduce the randomness of the experimental results, the average values of 10 experiments are used as the final results.

(3) CONSGD [34]: a spammer group detection method that employs cosine pattern mining to find candidate spammer groups and models the candidate groups and their relations as an HIN. CONSGD uses the label propagation method to calculate the spamming probabilities of the candidate groups based on the labeled instances and treats the groups with high spamming probabilities as the spammer groups. In the experiments, the parameters τ_{AR}, τ_{MS} , α, and k for CONSGD are set to 0.2, 0.01%, 0.3, and 20, respectively.

(4) GSDB [6]: a burst-based method to detect review spammer groups, which discovers candidate spammer groups in review bursts using the Kernel Density

Estimation algorithm, and then exploits both individual and group spam indicators to classify spammer groups. In the experiments, parameters θ, δ, δ_{TP}, δ_l, δ_G, k, h, and *ISIZE* used in GSDB are set to 30, 30, 0.1, 0.2, 0.3, 0.5, 0.075, and 7, respectively.

In the experiments, the window size and embedding size for CSGD-NE are set to 5 and 64, respectively. The number of walks per node and walk length for CSGD-NE are set to 100 and 10 on the Amazon review dataset and 10 and 20 on the Yelp_Miami review dataset, respectively.

4.3.1. Comparison of Detection Performance for Five Methods on the Amazon Review Dataset. Figures 3 and 4 show the comparison of *P@k* and *R@k* for CSGD-NE, GSBC, CONSGD, FAP, and GSDB with various numbers of top-*k* reviewers on the Amazon review dataset.

As shown in Figure 3, on the Amazon review dataset, the *P@k* of CONSGD is the worst on the whole among the five methods and tends to decrease as the number of top-k reviewers increases. FAP has good *P@k* before the top-330 reviewers. This is because FAP uses some seed spam users to drive the fraudulent action propagation, which leads to very high initial detection precision. However, the *P@k* of FAP shows a downward trend with the number of top-k reviewers increasing. This indicates that an increasing number of normal reviewers are misjudged as spam users. The *P@k* of GSDB and GSBC is overall better than that of CONSGD and FAP and gradually becomes stable at about 0.76 and 0.71, respectively, after the top-600 reviewers. Furthermore, the *P@k* value of GSBC keeps above 0.7 at the top-2000 reviewers. However, the *P@k* curve of GSDB ends before top-1450, indicating that the total number of spammers obtained by GSDB is less than 1450. The *P@k* of CSGD-NE increases gradually before the top-330 reviewers and subsequently tends to be stable. Moreover, the *P@k* value of CSGD-NE still keeps 0.88 at the top-2000 reviewers, which is obviously higher than that of the baseline methods. Therefore, the *P@k* of CSGD-NE is overall better than that of GSBC, CONSGD, FAP, and GSDB on the Amazon review dataset.

As shown in Figure 4, the *R@k* of CONSGD and FAP on the Amazon review dataset shows an upward trend before the top-800 reviewers. Subsequently, it has little change as the number of top-k reviewers increases. This phenomenon shows that the number of spam users identified by CONSGD and FAP is basically unchanged after the top-800 reviewers. In contrast, the *R@k* of CSGD-NE, GSDB, and GSBC tends to increase as the number of top-k reviewers increases, which means more and more spam users are detected by the three methods. Moreover, the *R@k* value of CSGD-NE and GSBC at the top-2000 reviewers still maintains at about 0.92 and 0.73, respectively. Clearly, the *R@k* of CSGD-NE is much better on the whole than that of GSBC, CONSGD, FAP, and GSDB on the Amazon review dataset.

Based on the above analysis, we can conclude that CSGD-NE outperforms GSBC, CONSGD, FAP, and GSDB in terms of *P@k* and *R@k* metrics when detecting collusive spamming groups in the Amazon review dataset.

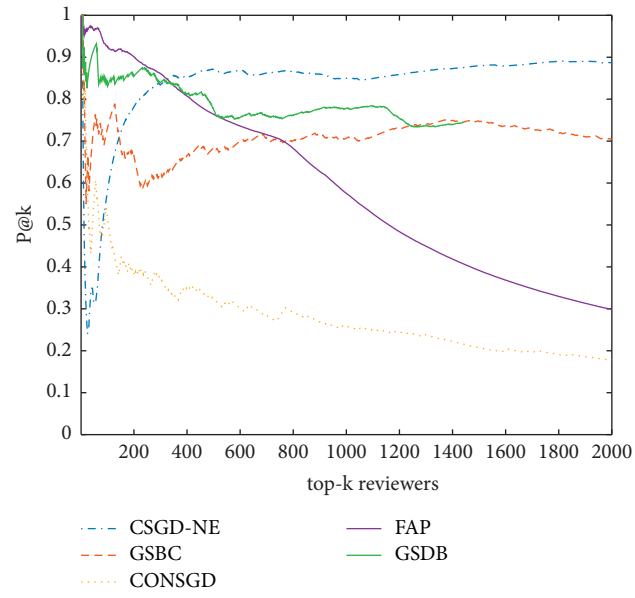

Figure 3: Comparison of P@k for five methods on the Amazon review dataset.

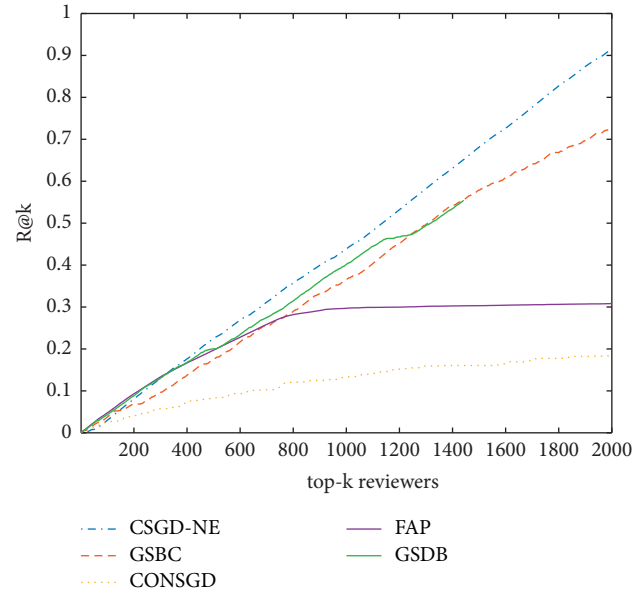

Figure 4: Comparison of R@k for five methods on the Amazon review dataset.

4.3.2. Comparison of Detection Performance for Five Methods on the Yelp_Miami Review Dataset. Figures 5 and 6 show the comparison of *P@k* and *R@k* for CSGD-NE, GSBC, CONSGD, FAP, and GSDB with various numbers of top-*k* reviewers on the Yelp_Miami review dataset.

It can be seen from Figure 5 that the *P@k* curve of GSBC on the Yelp_Miami review dataset end at the top-70 reviewers, indicating that the number of spamming groups that GSBC can detect on this dataset is very limited. This phenomenon is not difficult to explain. GSBC employs the minimum cut algorithm to divide candidate groups, which easily produces isolated nodes. Moreover, the candidate groups with spamming scores

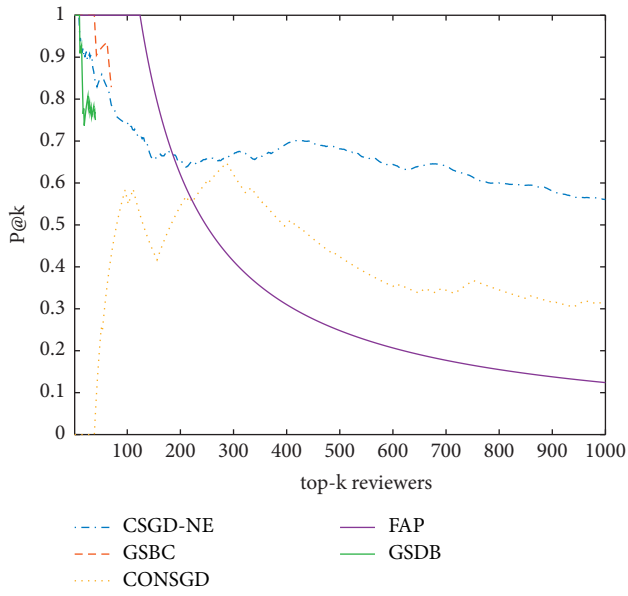

Figure 5: Comparison of P@k for five methods on the Yelp_Miami review dataset.

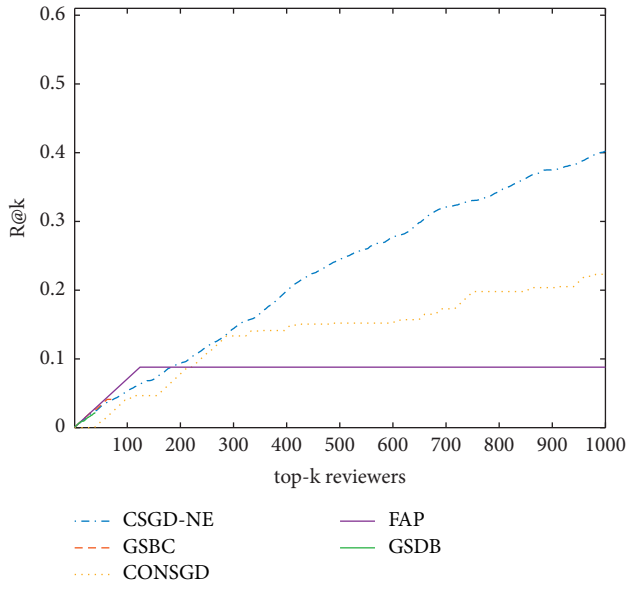

Figure 6: Comparison of R@k for five methods on the Yelp_Miami review dataset.

below a given threshold are discarded. GSDB has similar phenomena on this dataset. The *P@k* of FAP on the Yelp_Miami review dataset is always 1 before the top-125 reviewers. Subsequently, it gradually decreases as the number of top-*k* reviewers increases and it is only 0.124 at the top-1000 reviewers, which means more and more normal reviewers are misjudged as spam users by FAP on this dataset. The *P@k* of CONSGD on the Yelp_Miami review dataset is lower than that of FAP before the top-225 reviewers, and subsequently, it is better than that of FAP on this dataset. While CONSGD can detect more spam users than GSBC on the Yelp_Miami review dataset, its *P@k* is lower than that of GSBC. The *P@k* of CSGD-NE on the Yelp_Miami review dataset shows a downward trend before the top-150 reviewers and subsequently tends to be stable. Clearly, the *P@k* of CSGD-NE is, on the whole, better than that of GSBC, CONSGD, FAP, and GSDB on the Yelp_Miami review dataset.

As shown in Figure 6, the *R@k* curve of GSDB and GSBC on the Yelp_Miami review dataset end before top-71 reviewers because they cannot spot more spamming groups in this dataset. The *R@k* of FAP on the Yelp_Miami review dataset increases gradually before the top-125 reviewers. Subsequently, it tends to be stable and maintains at about 0.088, which means no spamming groups are detected by FAP after the top-125 reviewers. The *R@k* of CONSGD on the Yelp_Miami review dataset tends to increase as more and more spam users are spotted by CONSGD and reaches 0.22 at the top-1000 reviewers, which is, on the whole, better than that of GSBC and FAP. The *R@k* of CSGD-NE on the Yelp_Miami review dataset also increases gradually as an increasing number of spam users are detected and reaches about 0.4 at the top-1000 reviewers. Therefore, the *R@k* of CSGD-NE is overall better than that of GSBC, CONSGD, FAP, and GSDB on the Yelp_Miami review dataset.

Based on the above analysis, we can conclude that CSGD-NE performs better than GSBC, CONSGD, FAP, and GSDB when detecting collusive spamming groups in the Yelp_Miami review dataset.

5. Conclusions

The collusive spamming groups can produce fake product reviews in batches, which poses a serious challenge to the credibility of e-commerce platforms. In this paper, we propose a network embedding based approach for detecting collusive spamming groups. Borrowing from the idea of meta-graph, we model the given user review dataset as a HIN. We modify the DeepWalk algorithm to learn the low-dimensional vector representations of user nodes in the HIN and obtain the candidate spamming groups by combining the Canopy and K-means clustering algorithms. We put forward three spamming group detection indicators and combined them with two existing detection indicators to measure the spamming behavior of each candidate group. By ranking the candidate groups according to their spamming scores, we obtain top-k groups with high spamming scores as the collusive spamming groups. The experimental results on two real-world review datasets demonstrate the superiority of the proposed method in detecting the collusive spamming groups.

Data Availability

The data used to support the findings of this study are available from the first author (xjcjb@ysu.edu.cn) upon request.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (no. 62072393).

References

[1] R. K. Dewang and A. K. Singh, "State-of-art approaches for review spammer detection: a survey," *Journal of Intelligent Information Systems*, vol. 50, no. 2, pp. 231–264, 2018.

[2] A. Heydari, M. a. Tavakoli, N. Salim, and Z. Heydari, "Detection of review spam: a survey," *Expert Systems with Applications*, vol. 42, no. 7, pp. 3634–3642, 2015.

[3] A. Mukherjee, B. Liu, and N. Glance, "Spotting fake reviewer groups in consumer reviews," in *Proceedings of the 21st International Conference on World Wide Web*, pp. 191–200, Lyon, France, April 2012.

[4] C. Xu, J. Zhang, K. Chang, and C. Long, "Uncovering collusive spammers in Chinese review websites," in *Proceedings of the 22nd ACM International Conference On Information & Knowledge Management*, pp. 979–988, San Francisco, CA, USA, November 2013.

[5] L. Li, B. Qin, and T. Liu, "Survey on fake review detection research," *Chinese Journal of Computers*, vol. 41, no. 4, pp. 946–968, 2018, in Chinese.

[6] S.-j. Ji, Q. Zhang, J. Li et al., "A burst-based unsupervised method for detecting review spammer groups," *Information Sciences*, vol. 536, pp. 454–469, 2020.

[7] C. Xu and J. Zhang, "Combating Product Review Spam Campaigns via Multiple Heterogeneous Pairwise Features," in *Proceedings of the Siam International Conference On Data Mining*, pp. 172–180, Vancouver, BC, Canada, April 2015.

[8] C. Xu and J. Zhang, "Towards collusive fraud detection in online reviews," in *Proceedings of the 15th IEEE International Conference On Data Mining*, pp. 1051–1056, Atlantic City, NJ, USA, November 2015.

[9] L. Zhang, Z. Wu, and J. Cao, "Detecting spammer groups from product reviews: a partially supervised learning model," *IEEE Access*, vol. 6, pp. 2559–2568, 2018.

[10] J. Ye and L. Akoglu, "Discovering Opinion Spammer Groups by Network Footprints," in *Proceedings of the Joint European Conference On Machine Learning And Knowledge Discovery In Databases*, pp. 267–282, Porto, Portugal, September 2015.

[11] E. Choo, T. Yu, and M. Chi, "Detecting opinion spammer groups through community discovery and sentiment analysis," in *Proceedings of the 29th Annual IFIP Conference On Data And Applications Security And Privacy*, pp. 170–187, Fairfax, VA, USA, July 2015.

[12] Z. Wang, T. Hou, D. Song, Z. Li, and T. Kong, "Detecting review spammer groups via bipartite graph projection," *The Computer Journal*, vol. 59, no. 6, pp. 861–874, 2016.

[13] Q. N. T. Do, A. Zhilin, C. Z. P. Junior, G. Wang, and F. K. Hussain, "A network-based approach to detect spammer groups," in *Proceedings of the International Joint Conference On Neural Networks*, pp. 3642–3648, Vancouver, BC, Canada, July 2016.

[14] Q. N. T. Do, F. K. Hussain, and T. N. Bang, "A fuzzy approach to detect spammer groups," in *Proceedings of the IEEE International Conference on Fuzzy Systems*, pp. 1–6, Naples, Italy, July 2017.

[15] Z. Han, K. Yang, and X. Tan, "Analyzing spectrum features of weight user relation graph to identify large spammer groups in online shopping websites," *Chinese Journal of Computers*, vol. 40, no. 4, pp. 939–954, 2017, in Chinese.

[16] Z. Wang, S. Gu, X. Zhao, and X. Xu, "Graph-based review spammer group detection," *Knowledge and Information Systems*, vol. 55, no. 3, pp. 571–597, 2018.

[17] E. Serra, A. Shrestha, F. Spezzano, and A. Squicciarini, "DeepTrust: an automatic framework to detect trustworthy users in opinion-based systems," in *Proceedings of the Tenth ACM Conference On Data And Application Security And Privacy*, pp. 29–38, New Orleans, LA, USA, March 2020.

[18] Y. Dou, G. Ma, P. S. Yu, and S. Xie, "Robust spammer detection by nash reinforcement learning," in *Proceedings of the 26th ACM SIGKDD Conference On Knowledge Discovery And Data Mining*, pp. 924–933, California, CA, USA, June 2020.

[19] Z. Guo, L. Tang, T. Guo, K. Yu, M. Alazab, and A. Shalaginov, "Deep Graph neural network-based spammer detection under the perspective of heterogeneous cyberspace," *Future Generation Computer Systems*, vol. 117, pp. 205–218, 2021.

[20] Z. Song, F. Bai, J. Zhao, and J. Zhang, "Spammer detection using graph-level classification model of graph neural network," in *Proceedings of the IEEE 2nd International Conference On Big Data, Artificial Intelligence And Internet Of Things Engineering*, pp. 531–538, Nanchang, China, March 2021.

[21] J. Cao, R. Xia, Y. Guo, and Z. Ma, "Collusion-aware detection of review spammers in location based social networks," *World Wide Web*, vol. 22, no. 6, pp. 2921–2951, 2019.

[22] F. Zhang, X. Hao, J. Chao, and S. Yuan, "Label propagation-based approach for detecting review spammer groups on e-commerce websites," *Knowledge-Based Systems*, vol. 193, pp. 1–19, 2020.

[23] Y. Sun, J. Han, C. C. Aggarwal, and N. V. Chawla, "When will it happen?-relationship prediction in heterogeneous information networks," in *Proceedings of the 5th ACM International Conference on Web Search and Data Mining*, pp. 663–672, Seattle, WA, USA, February 2012.

[24] Y. Sun, J. Han, X. Yan, P. S. Yu, and T. Wu, "PathSim," *Proceedings of the VLDB Endowment*, vol. 4, no. 11, pp. 992–1003, 2011.

[25] Z. Huang, Y. Zheng, R. Cheng, Y. Sun, N. Mamoulis, and X. Li, "Meta structure: computing relevance in large heterogeneous information networks," in *Proceedings of the 22nd ACM SIGKDD International Conference On Knowledge Discovery And Data Mining*, pp. 1595–1604, San Francisco, CA, USA, August 2016.

[26] B. Perozzi, R. Al-Rfou, and S. Skiena, "Deepwalk: online learning of social representations," in *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 701–710, New York, NY, USA, August 2014.

[27] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Proceedings of the Advances In Neural Information Processing Systems*, pp. 3111–3119, Lake Tahoe, NV, USA, December 2013.

[28] A. McCallum, K. Nigam, and L. H. Ungar, "Efficient clustering of high-dimensional data sets with application to reference matching," in *Proceedings of the 6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 169–178, Boston, MA, USA, August 2000.

[29] X. Liu, K. Shi, and Z. Yao, "Evaluation of the M&A effectiveness based on grey systems theory and entropy method," in *Proceedings of the 4th IEEE International Conference On Information Science And Technology*, pp. 268–271, Shenzhen, China, April 2014.

[30] M. Song, Q. Zhu, J. Peng, E. D. R. Santibanez Gonzalez, and D. R. Ernesto, "Improving the evaluation of cross efficiencies:

a method based on Shannon entropy weight," *Computers & Industrial Engineering*, vol. 112, pp. 99–106, 2017.

[31] W. Li, M. Gao, H. Li, J. Zeng, Q. Xiong, and S. Hirokawa, "Shilling attack detection in recommender systems via selecting patterns analysis," *IEICE - Transactions on Info and Systems*, vol. E99.D, no. 10, pp. 2600–2611, 2016.

[32] R. Barbado, O. Araque, and C. A. Iglesias, "A framework for fake review detection in online consumer electronics retailers," *Information Processing & Management*, vol. 56, no. 4, pp. 1234–1244, 2019.

[33] Y. Zhang, Y. Tan, M. Zhang, Y. Liu, T. Chua, and S. Ma, "Catch the black sheep: unified framework for shilling attack detection based on fraudulent action propagation," in *Proceedings of the 24th International Joint Conference On Artificial Intelligence*, pp. 2408–2414, Buenos Aires, Argentina, July 2015.

[34] L. Zhang, G. He, J. Cao, H. Zhu, and B. Xu, "Spotting review spammer groups: a cosine pattern and network based method," *Concurrency and Computation: Practice and Experience*, vol. 30, no. 20, pp. 1–15, 2018.