Cybersecurity

RESEARCH

Open Access

Detecting fake reviewers in heterogeneous networks of buyers and sellers: a collaborative training-based spammer group algorithm

Qi Zhang[1], Zhixiang Liang[2], Shujuan Ji[1*], Benyong Xing[3] and Dickson K. W. Chiu[4]

Abstract

It is not uncommon for malicious sellers to collude with fake reviewers (also called spammers) to write fake reviews for multiple products to either demote competitors or promote their products' reputations, forming a gray industry chain. To detect spammer groups in a heterogeneous network with rich semantic information from both buyers and sellers, researchers have conducted extensive research using Frequent Item Mining-based and graph-based methods. However, these methods cannot detect spammer groups with cross-product attacks and do not jointly consider structural and attribute features, and structure-attribute correlation, resulting in poorer detection performance. Therefore, we propose a collaborative training-based spammer group detection algorithm by constructing a heterogeneous induced sub-network based on the target product set to detect cross-product attack spammer groups. To jointly consider all available features, we use the collaborative training method to learn the feature representations of nodes. In addition, we use the DBSCAN clustering method to generate candidate groups, exclude innocent ones, and rank them to obtain spammer groups. The experimental results on real-world datasets indicate that the overall detection performance of the proposed method is better than that of the baseline methods.

Keywords Spammer group, Heterogeneous network, Collaborative training, DBSCAN

Introduction

The convenience of e-commerce has made online shopping increasingly common. As the information in e-commerce is asymmetry and the lack of quality control centers, consumers tend to browse reviews related to products or services before purchasing them online. This makes online reviews an essential reference for consumers to make purchasing decisions. According to a Harvard University study, every 1-star increase in a product's rating on Yelp creates a 5–9% increase in revenue for that product (Luca 2016). Motivated by potential financial gain, some malicious sellers tend to collude with spammers, aiming to either demote competitors or promote their businesses by posting many fake reviews. The proliferation of fake reviews makes it impossible for consumers to judge the actual quality of products based on the review information. This seriously affects consumers' shopping experience and destroys the fair competition environment among merchants. It also has an extremely negative impact on the development of the e-commerce industry.

Spammers are constantly changing their spam strategies to escape the detection of the spammer identification model. Spammers often work in groups to camouflage their behavior and improve attack efficiency. A spammer

*Correspondence:
Shujuan Ji
jsjsuzie@sina.com
[1] Shandong Provincial Key Laboratory of Wisdom Mine Information Technology, Shandong University of Science and Technology, Qingdao 266590, Shandong, China
[2] Zhejiang University-University of Illinois at Urbana-Champaign Institute, Haining 314499, Zhejiang, China
[3] School of Civil Engineering, Beijing Jiaotong University, Beijing 100044, China
[4] Faculty of Education, The University of Hong Kong, Pok Fu Lam, Hong Kong, China

group is a group of reviewers who write fake reviews for one or several products in an organized and coordinated manner (Mukherjee et al. 2012). The spammer group is more covert, destructive, and influential than individual spammers. This is because spammer groups can evade the platform's detection by avoiding certain relationships with other group members or developing multiple relationships with genuine reviewers (Shehnepoor et al. 2022). In addition, they can mislead consumers by imitating the behavior and language of genuine reviewers and writing fake reviews about target products in a short period. Therefore, how to effectively identify spammer groups on e-commerce platforms and ensure the credibility of product reviews has become an urgent issue of network information security.

Since the pioneering work of Jindal and Liu (2008), most efforts have been aimed at detecting fake reviews (Jindal and Liu 2008; Ott et al. 2011; Li et al. 2011; Cao et al. 2020, 2022) or individual spammers (Wang et al. 2012, 2020; Mukherjee et al. 2013). In recent years, several researchers (Mukherjee et al. 2012; Shehnepoor et al. 2022, 2021; Ji et al. 2020; Xu et al. 2013; Zhang et al. 2021, 2022a, 2020a, 2022b; Wang et al. 2016, 2018; Li et al. 2017; Hu 2021; Akoglu et al. 2013; Ye and Akoglu 2015; Zheng et al. 2018; Zhu et al. 2019; Chao et al. 2022) have attempted to detect spammers with collusive fraudulent behaviors at the group level. The existing work for detecting spammer groups can be roughly divided into two categories, i.e., FIM-based and graph-based methods. FIM-based methods (Mukherjee et al. 2012; Shehnepoor et al. 2022, 2021; Xu et al. 2013; Zhang et al. 2021) typically identify candidate groups based on the co-review hypothesis, and graph-based methods (Wang et al. 2016, 2018; Li et al. 2017; Zhang et al. 2022a, 2020a, 2022b; Hu 2021; Akoglu et al. 2013; Ye and Akoglu 2015; Zheng et al. 2018; Zhu et al. 2019; Chao et al. 2022), such as graph partition or clustering on the constructed reviewer relationship network, to discover candidate groups. Both these two categories of methods utilize a set of spam indicators to measure the spamming behavior of each candidate group and from which to identify spammer groups. Though most existing methods have shown excellent performance, they have two major limitations. First, existing methods only detect spammer groups from the viewpoint of reviewers or single-product, ignoring the characteristic that spammer groups will implement cross-product (i.e., for multiple target products) attacks for camouflage to evade detectors. Secondly, most existing methods focus only on structural features of the review network or attribute features of nodes (e.g., features about the behavior of reviewers or products) when detecting spammer groups, without jointly considering structural and attribute features as well as the structure-attribute correlation. It is necessary to design a method that comprehensively discover all the available information to learn the feature representation of nodes.

Therefore, we propose a spammer group detection algorithm based on collaborative training for heterogeneous networks named SGDCTH. In particular, we first calculate the suspiciousness of each product based on the Network Footprint Score (NFS) metric to filter target products and then construct a heterogeneous induced sub-network based on all target products, in which we can detect spammer groups that commit cross-product attacks. Subsequently, we use a collaborative training method to model both the intra-partition and inter-partition proximity of a heterogeneous induced sub-network. In this process, we consider each node's structural and attribute information and the structure-attribute correlation to learn the feature representation of nodes effectively. Furthermore, we use the DBSCAN clustering method to generate candidate groups in the embedding space of reviewers. Finally, we obtain spammer groups by the group purification and ranking method. The contributions of this paper are summarized as follows:

(1) Unlike most existing methods that detect spammer groups based on the viewpoint of reviewers (Mukherjee et al. 2012; Shehnepoor et al. 2022, 2021; Xu et al. 2013; Zhang et al. 2021, 2022a, 2020a, 2022b; Wang et al. 2016, 2018; Li et al. 2017; Hu 2021; Akoglu et al. 2013; Ye and Akoglu 2015; Zheng et al. 2018; Zhu et al. 2019; Chao et al. 2022) or single-product (Ji et al. 2020), we propose a new heterogeneous network-based method for identifying spammer groups from the viewpoint of cross-product. We first filter target products, then construct a heterogeneous network based on the target product set, and finally discover spammer groups by learning feature representations of nodes in the heterogeneous network. This enables our method to detect groups that attack multiple target products more accurately.
(2) Unlike most existing methods that focus only on structural features of the review network (Shehnepoor et al. 2022; Ye and Akoglu 2015; Zheng et al. 2018; Zhu et al. 2019; Zhang et al. 2022b) or attribute features of nodes (Mukherjee et al. 2012; Ji et al. 2020; Xu et al. 2013; Li et al. 2017; Hu 2021;

Zhang et al. 2020a), we jointly consider structural and attribute features as well as the structure-attribute correlation to detect spammer groups. We first extract the raw structural and attribute features of nodes, then use the collaborative training method to model both the intra-partition and inter-partition proximity of a heterogeneous network. In the training process, we take all available information into account to capture suspicious spammer groups in terms of structure and attributes.

(3) We conduct experiments on real-world review datasets and make a comparison with four baseline methods. The experimental results indicate that our method can accurately and efficiently detect active spammer groups on e-commerce websites.

The rest of the paper is organized as follows. Section "Related work" reviews the related work on spammer group detection. Section "The spammer group detection algorithm based on collaborative training for heterogeneous networks" details our SGDCTH method. Section "Experiments" describes the experimental results. Finally, we summarize this paper in section "Conclusion".

Related work

According to different classification criteria, we can divide the existing work on spammer group detection into four dimensions. First, depending on the strategy used to generate candidate spammer groups, the existing work can be divided into two categories, i.e., FIM-based and graph-based methods. Secondly, according to the different features considered in group mining, the existing work can be divided into three categories, i.e., the methods based on group behavior and content analysis (B&C), the methods based on group structure analysis (S), and the methods combining group behavior and structure analysis (B&C+S). Thirdly, according to the different coupling degrees of the discovered group members, the existing work can be divided into two categories, i.e., tightly coupled and loosely coupled methods. Fourthly, according to the different concentrations of the attacked target products, the existing work can be divided into two categories, i.e., single-product and cross-product methods. It is worth noting that tightly coupled and loosely coupled methods are designed based on reviewers' viewpoints, and existing works are almost entirely from the viewpoint of reviewers to detect spammer groups. However, single-product and cross-product methods are designed based on products' viewpoints. To the best of our knowledge, Ji et al. (2020) were the first to propose detecting spammer groups from the viewpoint of products. The following subsections review existing work according to the strategy used to generate candidate spammer groups.

FIM-based methods

FIM-based methods first use the FIM method to generate candidate groups based on the co-review hypothesis and then rank or classify them to obtain spammer groups. Mukherjee et al. (2012) were the first to study the problem of spammer group detection. They use the FIM method to treat reviewers who co-review the same set of products as a candidate group and propose a relationship-based model to detect spammer groups. Later, Xu et al. (2013) proposed a KNN-based and a graph-based classification method to predict whether a candidate group's members are suspicious. Zhang et al. (2021) first use the FIM method to discover candidate groups and then propose a method that fuses behavioral and structural feature reasoning to detect spammer groups. After obtaining the candidate groups based on the concept of FIM, Shehnepoor et al. (2022, 2021) use deep learning methods to gradually refine the reviewers' representation and remove abnormal members from candidate groups based on the refined representation, and finally classify candidate groups. However, the FIM method may incorrectly classify some genuine reviewers who accidentally post reviews into the spammer groups in the process of mining groups. In addition, the method is very sensitive to the setting of support thresholds. Therefore, FIM methods are suitable for detecting tightly coupled groups (i.e., group members need to review all target products) but not for detecting loosely coupled groups (i.e., group members do not need to review every target product to conceal the group's spamming behavior) (Wang et al. 2016, 2018; Zhang et al. 2022b).

Graph-based methods

Graph-based methods use graph partition, clustering, community detection, and other methods to generate candidate groups on the review network and then rank or classify them to obtain spammer groups. Different graph construction methods can be further divided into homogeneous and heterogeneous graph-based methods.

Homogeneous graph-based methods

Among homogeneous graph-based methods, researchers generally detect spammer groups in the reviewer relationship network constructed based on the similarity between reviewers. Wang et al. (2016) adopt the divide-and-conquer idea to detect loosely coupled spammer groups on the reviewer projection network. On this basis, Wang et al. (2018) propose a top-down framework

GSBC, which uses the min-cut method to discover spammer groups on a bi-connected reviewer network. Li et al. (2017) use the graph clustering method to obtain spammer groups on the co-burst network. Zhang et al. (2022a) detect spammer groups in three steps. First, they construct a reviewer relationship network and use an improved label propagation method to discover candidate groups. Secondly, they adopt a combination of subjective and objective indicator weighting strategies to evaluate the spamicities of each candidate group. Finally, they rank the candidate groups according to spamicity scores to obtain spammer groups. Hu (2021) uses a community mining method based on network representation learning on the constructed reviewer similarity network to detect tightly connected groups and from which to identify true spammer groups. Zhang et al. (2020a) use an improved label propagation method to obtain candidate groups and propose a new ranking method to find collusive spammers. However, homogeneous graph-based methods do not deeply examine the implicit relationships among reviewers when constructing the reviewer relationship network, which fails to discover spammers with collusive fraudulent behaviors. Moreover, these methods cannot capture the highly non-linear relationship between nodes in the network (Zheng et al. 2018).

Heterogeneous graph-based methods

Among heterogeneous graph-based methods, researchers detect spammer groups in a reviewer-object heterogeneous network constructed from the reviewer's review behavior. Akoglu et al. (2013) obtain spammer groups by the graph clustering method on an induced sub-graph containing highly suspicious reviewers and corresponding products. Ye et al. (2015) propose a two-step method to discover review spammer groups. They first identify target products vulnerable to spammer attacks and then use the agglomerative hierarchical clustering method to detect spammer groups on an induced sub-graph. Zheng et al. (2018) first utilize the deep network embedding method to jointly learn the feature representation of nodes in a bipartite review network and then use the DBSCAN clustering method to detect dense blocks in the latent space. Zhu et al. (2019) first embed explicit and implicit relations in a bipartite network to obtain the representation of reviewers and then use a k-dimensional tree-based fast-density sub-graph mining method to obtain multiple collaborative groups. Chao et al. (2022) first construct a heterogeneous network based on the idea of meta-graph and use the improved DeepWalk method to learn the feature representation of nodes. Then, they utilize the Canopy and K-means clustering method to generate candidate groups and treat the top k most suspicious groups as spammer groups. Zhang et al. (2022b) first construct a reviewer-product bipartite network as the agent's interactive environment and use an improved reinforcement learning method to generate candidate groups. Next, they exploit the Doc2Vec model to obtain the embedding vector of each candidate group and devise an adversarial autoencoder-based one-class classification model for detecting collusive spammers. The above heterogeneous graph-based methods neglected to exclude innocent individuals in candidate groups. Furthermore, these methods only utilize structural or attribute information when detecting spammer groups, which do not jointly consider structural and attribute information as well as the structure-attribute correlation.

Summary

Table 1 summarizes the existing work in these four dimensions. Although most existing FIM-based or graph-based methods for detecting spammer groups are generally effective, they have some limitations. Specifically, FIM-based methods are prone to misjudging genuine reviewers as spammers in the process of mining groups. Furthermore, the FIM methods are suitable for detecting tightly coupled spammer groups. For homogeneous graph-based and heterogeneous graph-based methods, the former does not take full advantage of the implicit relationships between reviewers when constructing the reviewer relationship network, while the latter ignores the step of group purification. Moreover, existing heterogeneous graph-based methods do not jointly consider the structural features of the review network and the attribute features of nodes as well as the structure-attribute correlation.

The spammer group detection algorithm based on collaborative training for heterogeneous networks

Aiming at the limitations of existing research methods, we propose a new unsupervised spammer group detection algorithm, SGDCTH, as shown in Algorithm 1. Figure 1 shows the overall framework of our method. In detail, our method consists of four steps. First, we filter target products based on the NFS metric and then construct a heterogeneous induced sub-network based on the set of target products (see Algorithms 2 and 3 for details). Secondly, we use a collaborative training method to model the intra-partition and inter-partition proximity of the heterogeneous induced sub-network to obtain low-dimensional vector representations of nodes (see Algorithm 4 for details). Thirdly, the candidate spammer groups are generated based on the DBSCAN clustering method (see Algorithm 5 for details). Fourthly, innocent reviewers are excluded from the candidate group and ranked to obtain spammer groups (see Algorithm 6 for details). An algorithm implements each step. We describe the preliminary in subsection "Preliminary", and the subsequent subsections describe the implementation details of each step.

Table 1 A summary of existing work on spammer group detection

Method	Classification dimension									
	The strategy used to generate candidate groups			Considered features			The coupling degrees of group members		The concentrations of the attacked target products	
	FIM-based	Graph-based		B&C	S	B&C+S	Tightly-coupled	Loosely-coupled	Single-product	Cross-product
		Homogeneous	Heterogeneous							
Mukherjee et al. (2012)	√									
Shehnepoor et al. (2022)	√				√		√			
Ji et al. (2020)				√					√	
Xu et al. (2013)	√			√			√			
Zhang et al. (2021)	√					√	√			
Shehnepoor et al. (2021)	√					√	√			
Wang et al. (2016)		√				√		√		
Wang et al. (2018)		√				√		√		
Li et al. (2017)		√		√				√		
Zhang et al. (2022a)		√				√		√		
Hu (2021)				√				√		
Zhang et al. (2020a)		√		√				√		
Akoglu et al. (2013)			√			√		√		
Ye et al. (2015)			√		√			√		
Zheng et al. (2018)			√		√			√		
Zhu et al. (2019)					√			√		
Chao et al. (2022)			√			√		√		
Zhang et al. (2022b)			√		√			√		
SGDCTH (Ours)			√			√		√		√

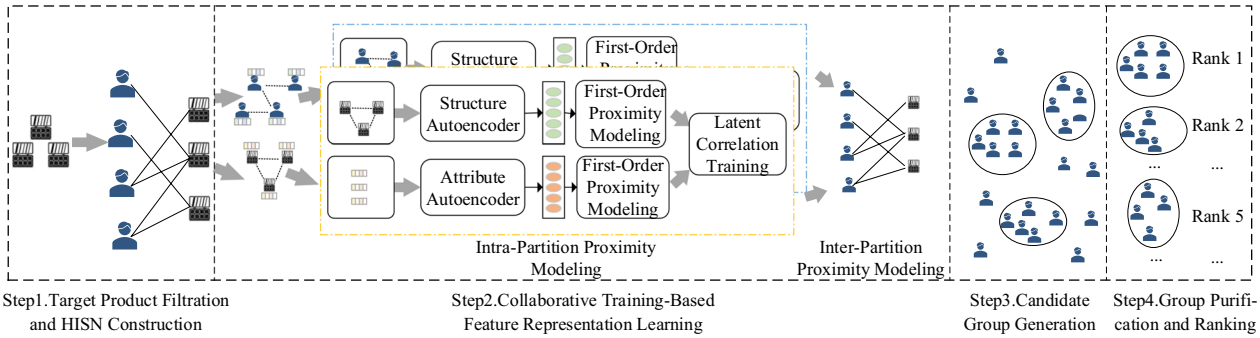

Fig. 1 The overall framework of SGDCTH

Algorithm 1 SGDCTH ($\mathcal{G}$, $\delta_{\tilde{p}}$, $\mathcal{SG}$, K, t, d, $\mathbf{H}$, ϵ, ϕ, $Candidate\_Group$, BiG)

Input:

$\mathcal{G}$: HIN

$\delta_{\tilde{p}}$: target product filtration threshold

$\tilde{P}$: target product set

$\mathcal{SG}$: HISN

K : number of sub-graphs

t : number of training epochs

d : embedding dimension

$\mathbf{H}$: feature representation of nodes

ϵ : neighborhood radius threshold

ϕ : minimum number of sample points threshold

$Candidate\_Group$: candidate group set

BiG : heterogeneous structural graph of candidate groups

Output:

$Spammer\_Group$: spammer group set

// Step1: Get the target product set and HISN

1. $\tilde{P}$ = **TargetProductFiltration** ($\mathcal{G}$, $\delta_{\tilde{p}}$)
2. $\mathcal{SG}$ = **HISNConstruction** ($\mathcal{G}$, $\tilde{P}$)

// Step2: Get the feature representation of nodes

3. $\mathbf{H}$ = **FeatureRepresentationLearning** ($\mathcal{SG}$, K, t, d)

// Step3: Get the candidate group set

4. $Candidate\_Group$ = **CandidateGroupGeneration** ($\mathbf{H}$, ϵ, ϕ)

//Step4: Get the spammer group set

5. $Spammer\_Group$ = **GroupPurification&Ranking** ($Candidate\_Group$, BiG)

Preliminary

This subsection defines several important concepts that are relevant to our work.

Definition 1 Heterogeneous Information Network (Wang et al. 2022). Heterogeneous Information Network (HIN) is defined as a network $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ and $\mathcal{E}$ denotes the set of nodes and the set of edges, respectively, and each node $v \in \mathcal{V}$ and each edge $e \in \mathcal{E}$ is associated with their node type mapping function $\phi(v) : \mathcal{V} \rightarrow \mathcal{A}$ and edge type mapping function $\varphi(e) : \mathcal{E} \rightarrow \mathcal{R}$, where $\mathcal{A}$ and $\mathcal{R}$ denotes the set of node types and edge types respectively, $|\mathcal{A}+\mathcal{R}| > 2$.

Definition 2 Meta-path (Wang et al. 2021a). A meta-path $\mathcal{P}$ is defined as a path in the form of $A_1 \xrightarrow{R_1} A_2 \xrightarrow{R_2} \cdots \xrightarrow{R_l} A_{l+1}$(abbreviated as $A_1A_2 \cdots A_{l+1}$), which describes a composite relation $R = R_1 \circ R_2 \circ \cdots \circ R_l$ between node types A_1 and A_{l+1}, where $\circ$ denotes the composition operator on relations.

The target product filtration method and the heterogeneous induced sub-network construction method

The target product filtration method

Inspired by Ji et al. (2020), who detect spammer groups based on review bursts from the products' viewpoints, we cite the NFS metric (Ye and Akoglu 2015) to quantify the likelihood of a product being attacked. NFS leverages two key observations relevant to real-world networks, i.e., neighbor diversity and self-similarity. The former means the local diversity of node importance within the neighborhood of a node, and the latter means the distributional similarity between node importance at the local and global levels. In this work, we use *degree* and *PageRank* centrality (Ye and Akoglu 2015) to measure node importance in a network.

(1) Neighbor diversity of nodes

In order to measure the diversity of neighborhood centralities of a given product $\widetilde{p} \in \mathcal{V}$ with *degree* $deg(\widetilde{p})$, we mainly divide it into three steps. First, a list of buckets $f = \{0, 1, ...\}$ is created so that the bucket boundary values grow exponentially as $a \cdot b^f$. Then, the reviewers are placed in F buckets, and the reviewers in each bucket are counted and normalized to obtain a discrete probability distribution $S^{(i)}$ with value $[s_1^{(i)}, ..., s_F^{(i)}]$. Finally, by calculating the Shannon entropy of $S^{(i)}$, the product $\widetilde{p}$ obtains two neighbor diversity scores $H_{deg}(\widetilde{p})$ and $H_{pr}(\widetilde{p})$ for *degree* and *PageRank*, respectively. The lower these scores are, the more suspicious the product is.

(2) Self-similarity in real-world network

To calculate the self-similarity for a given product $\widetilde{p}$, the histogram density $S^{(i)} = [s_1^{(i)}, ..., s_F^{(i)}]$ of the centrality of the reviewers and the *KL*-divergence between all reviewers in the network denoted by T is defined. T is calculated in the same way as S, except that T divides the centrality values of all reviewers in a network into buckets. Finally, the product $\widetilde{p}$ obtains two separate scores $KL_{deg}(\widetilde{p})$ and $KL_{pr}(\widetilde{p})$ from the difference in self-similarity. The higher these scores are, the more suspicious the product is.

(3) NFS metric

Finally, each product receives four suspiciousness scores, where two based on neighbor diversity, i.e., H_{deg} and H_{pr}, and two based on self-similarity, i.e., KL_{deg} and KL_{pr}. We use the Cumulative Distribution Function (CDF) to unify them into a standard scale score. Let $H = \{H(1), H(2), ...\}$ as a list of entropy values calculated for a set of products (based on *degree* or *PageRank* centrality). To quantify the extremes of $H(\widetilde{p})$, an empirical CDF is used on H and the probability that the set $H = \{H(1), H(2), ...\}$ is less than or equal to $H(\widetilde{p})$ is counted and calculated as follows (Ye and Akoglu 2015).

$$f(H(\widetilde{p})) = P(H \le H(\widetilde{p})) \tag{1}$$

On the other hand, for the *KL*-divergence, the statistical probability that the set $KL = \{KL(1), KL(2), ...\}$ is greater than $KL(\widetilde{p})$ is calculated as follows (Ye and Akoglu 2015).

$$f(KL(\widetilde{p})) = 1 - P(KL \le KL(\widetilde{p})) \tag{2}$$

Our ultimate goal is to take the low value in $H(\widetilde{p})$ and the high value in $KL(\widetilde{p})$, and obtain the NFS value of a product $\widetilde{p}$ by combining them. A higher value of $NFS(\widetilde{p}) \in [0, 1]$ indicates that a product is more suspicious, calculated as follows (Ye and Akoglu 2015).

$$NFS(\widetilde{p}) = 1 - \sqrt{\frac{f(H_{\text{deg}}(\widetilde{p}))^2 + f(H_{pr}(\widetilde{p}))^2 + f(KL_{\text{deg}}(\widetilde{p}))^2 + f(KL_{pr}(\widetilde{p}))^2}{4}} \tag{3}$$

The heterogeneous induced sub-network construction method

Based on the HIN and the set of target products, we first give the definition of the heterogeneous induced sub-network.

Definition 3 Heterogeneous Induced Sub-network. The heterogeneous induced sub-network (HISN) is defined as a network $\mathcal{SG} = (\mathcal{V}_{\mathcal{SG}}, \mathcal{E}_{\mathcal{SG}})$, where $\mathcal{V}_{\mathcal{SG}}$ and $\mathcal{E}_{\mathcal{V}}$ denotes the set of nodes and edges, respectively. The sub-network consists of target product set $\widetilde{P}$, all reviewers R who reviewed target products in $\widetilde{P}$, and all products $P \supseteq \widetilde{P}$ reviewed by these reviewers. In other words, this sub-network is an induced sub-network of the network $\mathcal{G}$ at nodes within two hops of target products in $\widetilde{P}$.

Based on the above description, we design a method for filtering target products in Algorithm 2. For each product in the review network, its NFS value is calculated. If the NFS value exceeds a given threshold $\delta_{\widetilde{p}}$, and then it is added to the target product set $\widetilde{P}$. In addition, we design a method for constructing HISN in Algorithm 3. In Algorithm 3, we construct the HISN using all of the target products.

The collaborative training-based feature representation learning method

In real life, a spammer often has a close relationship with a series of manipulated target products, i.e., a reviewer-product relationship. To increase the concealment of a group, its members often collaborate to co-review multiple target products, i.e., a reviewer-reviewer relationship. The target products under attack will have overlapped

Algorithm 2 TargetProductFiltration ($\mathcal{G}$, $\delta_{\widetilde{p}}$)

Input:
$\mathcal{G}$: HIN
$\delta_{\widetilde{p}}$: target product filtration threshold
Output:
$\widetilde{P}$: target product set
1. $\widetilde{P} = \varnothing$
2. **for** each $\widetilde{p}$ in $\mathcal{V}$ **do**
3. **if** $NFS(\widetilde{p}) \geq \delta_{\widetilde{p}}$ **then** // via Eq.(3)
4. $\widetilde{P} \cup = \widetilde{p}$
5. **end if**
6. **end for**
7. **return** $\widetilde{P}$

Algorithm 3 HISNConstruction ($\mathcal{G}$, $\widetilde{P}$)

Input:
$\mathcal{G}$: HIN
$\widetilde{P}$: target product set
Output:
$\mathcal{SG}$: HISN
1. $\mathcal{V}_{\mathcal{SG}} = \varnothing$, $\mathcal{E}_{\mathcal{SG}} = \varnothing$
2. find all reviewers R who reviewed $\widetilde{P}$ in $\mathcal{G}$
3. find all products P who is reviewed by R in $\mathcal{G}$
4. $\mathcal{V}_{\mathcal{SG}} = R \cup P$
5. **for** each $p_j \in P$ **do**
6. **for** each $r_i \in R$ **do**
7. **if** p_j is reviewed by r_i **then**
8. $\mathcal{E}_{\mathcal{SG}} \leftarrow \mathcal{E}_{\mathcal{SG}} \cup \{(r_i, p_j)\}$
9. **end if**
10. **end for**
11. **end for**
12. $\mathcal{SG} = (\mathcal{V}_{\mathcal{SG}}, \mathcal{E}_{\mathcal{SG}})$
13. **return** $\mathcal{SG}$

spammers, i.e., a product-product relationship. To capture these relationships, we first use a collaborative training method to model both the intra-partition proximity and inter-partition proximity of HISN. Then, we model the structure-attribute correlation using a latent correlation training strategy to learn the feature representation of nodes.

Intra-partition proximity modeling

The intra-partition proximity captures the relationships between nodes within the same partition in terms of both structure and attributes (i.e., implicit relationships, including reviewer-reviewer relationship and product-product relationship). On the one hand, the nodes with similar "interaction" behaviors with nodes in the other partition should have high proximity (i.e., structural proximity). On the other hand, nodes sharing similar attributes tend to exhibit similar behaviors in the network (i.e., attribute proximity) (Huang et al. 2020). Specifically, we first extract the raw structural features and attribute features of nodes in HISN. Subsequently, we partition HISN and input the structural and attribute features of nodes within the same partition into two independent autoencoders to learn their compact representations. Finally, we jointly model the structural and attribute proximity to preserve the first-order proximity of nodes.

The raw feature extraction method

(1) The raw structural feature extraction method

The NFS measures how unusually suspiciously similar reviewers target a product, and such groups of highly similar reviewers are likely to work together in spam campaigns (Ye and Akoglu 2015). Therefore, the behavior of reviewers within each sub-graph (i.e., a bipartite sub-graph consisting of a target product and its corresponding reviewer) is highly consistent. However, Wang et al. (2021b) found an inconsistency between a node's behavior and its label semantics. Inspired by Wang et al. (2021b), we propose to adopt the contrast between node representation and sub-graph representation to reduce the impact of inconsistency caused by different behaviors across sub-graphs.

We first perform feature decomposition for the normalized adjacency matrix to obtain the initial feature vector X, where $x_i \in \mathcal{R}^{d_0}$ represents the d_0-dimensional initial feature vector of a node v_i. Then, it is fed into GNN to learn the structural features of nodes. In our work, we adopt GIN (Xu et al. 2018) in Eq. (4), a state-of-the-art graph neural network, to learn the structural features of each node by means of a sum-like neighborhood aggregation function.

$$\mathbf{x}_i^{(l)} = \mathrm{MLP}^{(l)}\left(\left(1+\lambda^{(l)}\right)\cdot \mathbf{x}_i^{(l-1)} + \sum_{v_j \in N(v_i)} \mathbf{x}_j^{(l-1)}\right) \quad (4)$$

where $\mathbf{x}_i^{(l)} \in \mathcal{R}^d$ is the embedding of node v_i at l-th layer, and $\mathbf{x}_i^{(0)}=x_i$. $N(v_i)$ is the set of neighbors of node v_i. MLP denotes a multi-layer perceptron. $\lambda^{(l)}$ is either a learnable parameter or a fixed scalar. We stack L layers to obtain the higher-order structural features $\mathbf{x}_i^{(L)}$ of each node in HISN.

For each sub-graph C_k, we compute a sub-graph level representation $\boldsymbol{s}_k$ to summarize most nodes' behavior.

$$\boldsymbol{s}_k = \sigma\left(\frac{1}{n_k}\sum_{v_i \in C_k} \mathbf{x}_i^{(L)}\right) \quad (5)$$

where n_k denotes the number of nodes in C_k.

The sub-graph level representation is encoded as the node representation by optimizing the loss function $\mathcal{L}_{Stru}^k$, and the final loss function $\mathcal{L}_{Stru}$ is the average of K sub-graph losses.

$$\begin{aligned}\mathcal{L}_{Stru}^k = -\frac{1}{2n_k}\sum_{v_i \in V_k}\Big(&\mathbb{E}_{C_k}\log \mathcal{D}\left(\mathbf{x}_i^{(L)}, \boldsymbol{s}_k\right)\\ &+\mathbb{E}_{\widetilde{C_k}}\log\left(1-\mathcal{D}\left(\widetilde{\mathbf{x}}_i^{(L)}, \boldsymbol{s}_k\right)\right)\Big)\end{aligned} \quad (6)$$

$$\mathcal{L}_{Stru} = \frac{1}{K}\sum_{k=1}^{K}\mathcal{L}_{Stru}^k \quad (7)$$

where $\mathcal{D}$ is a discriminator that outputs the affinity score for each node-sub-graph pair. A sub-graph $\widetilde{C}_k$ is generated by a row-wise shuffling of the initial feature matrix of C_k, providing that node representation $\widetilde{\mathbf{x}}_i^{(L)}$ can be paired with sub-graph representation $\boldsymbol{s}_k$ as a negative sample.

(2) The raw attribute feature extraction method

Each node in HISN is associated with a set of attributes. In this paper, we extract 23 behavioral features from the literature (Zhang et al. 2020b) as the raw attribute features of a reviewer. In addition, we extract 6 behavioral features from the literature (Rayana and Akoglu 2015) and the proportion of each rating level of a product, a total of 11 behavioral features as the raw attribute features of a product. In particular, the numerical attributes are normalized, the categorical attributes are coded using one-hot, and they are all concatenated as the raw attribute features of a node.

Partitioning and compact representation learning method After obtaining the raw structural features $\mathbf{x}$ and attribute features $\mathbf{z}$, we divide HISN into reviewer partition and product partition based on the meta-path "R-P-R" and "P-R-P", where two nodes connected within a partition are each other's intra-partition neighbors. The same applies to product partition as to reviewer partition.

We feed the features $\mathbf{x}$ and $\mathbf{z}$ into two independent autoencoders to obtain encodings $\mathbf{x}\prime$ and $\mathbf{z}\prime$ as well as the reconstructed vectors $\hat{\mathbf{x}}$ and $\hat{\mathbf{z}}$. We capture the attribute information and structural information of a reviewer v_i by minimizing the following reconstruction loss function.

$$\mathcal{L}_1 = \sum_i \left\|\hat{\mathbf{x}}_i - \mathbf{x}_i\right\|^2 + \sum_i \left\|\hat{\mathbf{z}}_i - \mathbf{z}_i\right\|^2 \tag{8}$$

Joint modeling To bring two nodes with similar review behavior closer together in the embedding space, after obtaining the encodings $\mathbf{x}\prime$ and $\mathbf{z}\prime$, we perform a joint model of attribute encoding and structure encoding to preserve the first-order proximity between nodes by optimizing the following loss function.

$$\begin{aligned}\mathcal{L}_2 = &\sum_{a_{mn}>0} \log\sigma\left(\mathbf{x}\prime_m^{\mathrm{T}} \cdot \mathbf{x}\prime_n\right) - \sum_{n\prime=1} E_{v_{n\prime}\cdot\Omega_n(v)} \log\sigma\left(-\mathbf{x}\prime_m^{\mathrm{T}} \cdot \mathbf{x}\prime_{n\prime}\right)\\ &- \sum_{a_{mn}>0} \log\sigma\left(\mathbf{z}\prime_m^{\mathrm{T}} \cdot \mathbf{z}\prime_n\right) - \sum_{n\prime=1} E_{v_{n\prime}\cdot\Omega_n(v)} \log\sigma\left(-\mathbf{z}\prime_m^{\mathrm{T}} \cdot \mathbf{z}\prime_{n\prime}\right)\end{aligned} \tag{9}$$

where a_{mn} denotes the adjacency matrix elements of the synthesized intra-partition network, $\Omega_n(v)$ denotes the negative sampling distribution.

Finally, $\mathbf{x}\prime$ and $\mathbf{z}\prime$ are concatenated to obtain the final embedding $\mathbf{h}$, which is used for inter-partition proximity modeling.

Inter-partition proximity modeling

Inter-partition proximity captures the relationship between reviewers and products (i.e., the explicit relationship). For edges $\mathcal{E}_{\mathcal{SG}_{mn}}$ between r_m and p_n, consider the joint probability as the inter-partition proximity between them.

$$p(m,n) = \sigma\left(\mathbf{h}_m^{\mathrm{T}} \cdot \mathbf{h}_n\right) \tag{10}$$

where σ denotes the sigmoid function. $\mathbf{h}_m$ and $\mathbf{h}_n$ denotes the final embedding of r_m and p_n, respectively.

The likelihood function of the joint probability is maximized by minimizing the following loss function.

$$\mathcal{L}_3 = -\sum_{\varepsilon_{\mathcal{SG}_{mn}}\in\mathbb{E}} \log\sigma\left(\mathbf{h}_m^{\mathrm{T}} \cdot \mathbf{h}_n\right) - \sum_{n\prime=1} \mathbb{E}_{v_{n\prime}\cdot\Omega_n(v)} \log\sigma\left(-\mathbf{h}_m^{\mathrm{T}} \cdot \mathbf{h}_{n\prime}\right) \tag{11}$$

where $\Omega_n(v)$ denotes the negative sampling distribution.

Latent correlation training strategy

Since structural information and attribute information are two different modalities, they provide complementary information. Moreover, they both describe the same network, implying that they have potential consistency. Therefore, we comprehensively consider their complementarity and consistency, which is called structure-attribute correlation (Huang et al. 2020).

To effectively preserve attribute-structure correlation, two auxiliary space transformation kernels are used to transform encodings to a new latent space and project it to obtain the latent representations $\widetilde{\mathbf{x}}$ and $\widetilde{\mathbf{z}}$ (Huang et al. 2020). The attribute-structure correlation of any two nodes is defined as the joint probability of their latent representations.

$$\widetilde{p}(m,n) = \sigma\left(\widetilde{\mathbf{x}}_m^{T} \cdot \widetilde{\mathbf{z}}_n\right) \tag{12}$$

The likelihood function of the joint probability is maximized by minimizing the following loss function.

$$\begin{aligned}\mathcal{L}_4 = &-\sum_{\substack{m=n\\ or\\ r_m,r_n\sim\widetilde{p}(m,n)}} \log\sigma\left(\widetilde{\mathbf{x}}_m^{\mathrm{T}} \cdot \widetilde{\mathbf{z}}_n\right)\\ &- \sum_{n\prime=1} \mathbb{E}_{v_{n\prime}\sim\Omega_n(v)} \log\sigma\left(-\widetilde{\mathbf{x}}_m^{\mathrm{T}} \cdot \widetilde{\mathbf{z}}_{n\prime}\right)\end{aligned} \tag{13}$$

where $\widetilde{p}(m,n)$ denote the dynamic positive sampling distribution.

Ultimately, we combine all the optimization functions as a final objective function to optimize the embedding vector jointly.

$$\mathcal{L} = \mathcal{L}_1 + \mathcal{L}_2 + \mathcal{L}_3 + \mathcal{L}_4 \tag{14}$$

We summarize the process of collaborative training in Algorithm 4. Lines 1–19 model the intra-partition proximity. In particular, lines 1–9 extract the raw structural and attribute features of nodes. Line 10 divides HISN into two partitions based on meta-path. Lines 13–14 perform compact feature learning. Line 15 performs a joint model to preserve the first-order proximity between nodes within the same partition. Lines 16–19 model the correlation between attribute and structural information. Lines 20–21 model the inter-partition proximity.

Algorithm 4 FeatureRepresentationLearning ($\mathcal{SG}$, K, t, d)

Input:

$\mathcal{SG}$: HISN
K : number of sub-graphs
t : number of epochs
d : embedding dimension

Output:

$\mathbf{H} \in \mathbb{R}^d$: feature representation of nodes

1. obtain K sub-graphs $[C_1, C_2, \cdots, C_K]$
2. Initialize the parameters θ_1 and ω for the encoder g and the discriminator $\mathcal{D}$
3. **for** $epoch \leftarrow 1$ to t **do**
4. $\mathbf{x} = g(\mathcal{SG}, \theta_1)$
5. $\mathcal{L}_{Stru} = \frac{1}{K}\sum_{k=1}^{K} \mathcal{L}_{Stru}^{k}(H, C_k, \omega)$
6. $\theta_1, \omega \leftarrow Adam(\mathcal{L}_{Stru})$
7. **end for**
8. obtain the raw structural feature $\mathbf{x}$
9. obtain the raw attribute feature $\mathbf{z}$
10. divide HISN into two partitions P and R based on meta-path “R-P-R” and “P-R-P”
11. initialize model parameters θ_2
12. **while** not converged **do**
13. feed $\mathbf{x}$ and $\mathbf{z}$ into autoencoders to obtain $\mathbf{x}'$, $\mathbf{z}'$, $\hat{\mathbf{x}}$ and $\hat{\mathbf{z}}$
14. update autoencoders parameters via Eq. (8)
15. update encoders parameters via Eq. (9)
16. transform to $\mathbf{x}'$ and $\mathbf{z}'$ to $\tilde{\mathbf{x}}$ and $\tilde{\mathbf{z}}$
17. build up HNSW indexes based on $\tilde{\mathbf{x}}$ and $\tilde{\mathbf{z}}$
18. perform positive sampling for each $\tilde{\mathbf{x}}$ and $\tilde{\mathbf{z}}$
19. update encoders and kernels parameters via Eq. (13)
20. concatenate $\mathbf{x}'$ and $\mathbf{z}'$ to obtain embedding $\mathbf{h}$
21. update encoders parameters via Eq. (11)
22. **end while**
23. **return** $\mathbf{H}$

The DBSCAN-based candidate group generation method

After obtaining the feature representation of nodes, we utilize the DBSCAN clustering method to find candidate spammer groups in the reviewers' embedding space. The reasons for choosing the DBSCAN clustering method are that: (1) it can generate groups adaptively without the need for an artificially predefined number of groups; (2) it can discover groups of arbitrary shapes; and (3) it can find abnormal points in the process of mining groups (Ester et al. 1996). Algorithm 5 describes the specific process of the method.

Algorithm 5 CandidateGroupGeneration (H , ϵ , ϕ)

Input:

H : feature representation of nodes
ϵ : neighborhood radius threshold
ϕ : minimum number of sample points threshold

Output:

Candidate _ Group : candidate group set

```
Group ← ∅ , Candidate _ Group ← ∅
find ϵ neighbors of every node and identify n core nodes with more than ϕ neighbors
find the connected components of core nodes, ignoring all non-core nodes
assign each non-core node to a nearby group if the group is an ϵ neighbor; otherwise, assign it to noise
Group ← Group ∪ {group}
for each g in Group do
    if g is not in Candidate _ Group and size ( g )>=2 then
        Candidate _ Group ← Candidate _ Group ∪ {g}
    end if
end for
return Candidate _ Group
```

The group purification and ranking method

As some genuine reviewers who coincidently post reviews may be mixed in the detected candidate groups and are misjudged as spammers, we should filter out these innocent individuals. Therefore, we use the group purification method adopted by Zhang et al. (2022a) that can be used for HIN. The basic steps of the method are as follows. First, we calculate the contrast suspiciousness metric. Specifically, we extract the reviewer-product bipartite graph of each candidate group from the original review dataset. Based on the heterogeneous structure graph of candidate groups, we calculate the contrast suspiciousness metric in terms of structural characteristics of groups, rating time characteristics, and rating distribution characteristics. Secondly, we purify and rank the candidate groups. In particular, we define the spamicity (degree of spam) of an individual reviewer and the spamicity of a group according to the contrast suspiciousness metric. And we rank the candidate groups according to their spamicities to obtain spammer groups.

Contrast suspiciousness metric calculation method

Based on the generated candidate groups, we first construct a heterogeneous structure graph of candidate groups, which is defined as follows.

Definition 4 Heterogeneous structure graph of candidate groups.

The heterogeneous structure graph of candidate groups is defined as $BiG = (U, V, E)$, where U denotes all members of the candidate groups and V denotes the set of products reviewed by these members from the original review dataset. Notably, if a member writes multiple reviews on a product, there are multiple edges between them, each of which is associated with a rating and a timestamp.

In real life, to reduce the cost of attacks (e.g., time), a group of suspicious reviewers $A \subset U$ tends to collectively and actively write reviews on a set of products $B \subset V$ in a short period. Therefore, the density score $D(A, B)$ can be used to measure the extent to which A collective reviews the set of products B reviewed (Liu et al. 2018).

$$\begin{cases} D(A,B) = \frac{\sum_{v_i \in B} f_A(v_i)}{|A|+|B|} \\ f_A(v_i) = \sum_{(u_j,v_i)\in E \wedge u_j \in A} \sigma_{ji} e_{ji} \end{cases} \tag{15}$$

where $f_A(v_i)$ denotes the total edge frequency from A to a product v_i in B, σ_{ji} denotes the global suspiciousness of an edge, e_{ji} refers to the number of edges between (u_j, v_i).

To maximize $D(A, B)$, A and B are mutually dependent. As a result, we introduce the definition of contrast suspiciousness.

Definition 5 Contrast suspiciousness (Liu et al. 2018).

The contrast suspiciousness denoted as $P(v_i|A)$ is defined as the conditional probability of a node v_i that belongs to B, given A. The values of contrast suspiciousness are proportional to $q(\alpha_i)$, $q(\beta_i)$ and $q(\gamma_i)$. These values are calculated as follows.

(1) Topology

A product is suspicious if it is only reviewed by members in A and rarely by other members (Liu et al. 2018). From the topology perspective, the contrast suspiciousness satisfies Eq. (16).

$$\begin{cases} P(v_i|A) \propto q(\alpha_i) \\ \alpha_i = \frac{f_A(v_i)}{f_U(v_i)} \end{cases} \tag{16}$$

where $\alpha_i \in [0, 1]$ is the involvement ratio of members in A in the spam activity of a product v_i, $f_U(v_i)$ is the weighted indegree of v_i similar to $f_A(v_i)$, the edges are weighted by global suspiciousness and $q(\cdot)$ is a scale function chosen in the exponential form $q(x) = b^{x-1}$, where $b = 32$.

(2) Temporal bursts and drops

Let the time series of a product v_i as $\mathrm{T} = \{(t_0, c_0), (t_1, c_1), ..., (t_e, c_e)\}$, where c_i is the number of timestamps in the time box $[t_i - \Delta t/2, t_i + \Delta t/2)$ and Δt is the box size. The point with the maximum value c_m is set as the bursting point, i.e., (t_m, c_m). The awakening point of the burst is defined as the point along the time series T, to which the distance from l (the auxiliary straight line from the start point to the bursting point) is greatest. This paper uses the MultiBurst method (Liu et al. 2018) to find the sub-burst points and associated awakening points of multiple bursts. From the perspective of rating time, the contrast suspiciousness satisfies Eq. (17).

$$\begin{cases} P(v_i|A) \propto q(\beta_i) \\ \beta_i = \frac{\Phi(T_A)}{\Phi(T_U)} \\ \Phi(T) = \sum\limits_{(t_a, t_m)} \Delta c_{am} s_{am} \sum\limits_{t \in T} \mathrm{I}(t \in [t_a, t_m]) \end{cases} \tag{17}$$

where $\beta_i \in [0, 1]$ is the involvement ratio of members in A in multiple bursts, T_A is the collection of timestamp from members in A to v_i, T_U is the collection of timestamps from all members to v_i, Δc_{am} is the height difference of burst-awakening point pair, and s_{am} is the slope of burst-awakening point pair.

To capture the sudden drop pattern after attacking, we draw another auxiliary straight line from the highest point (t_m, c_m) to the last point (t_e, c_e). The point of death (t_d, c_d) (i.e., the end of the drop) is found by maximizing the distance to this straight line. We use the MaxDrop method (Liu et al. 2018) to find the maximum drop and slope. A product of the maximum drop and slope is used in Eq. (15) to measure the global suspiciousness of an edge.

$$\sigma = \Delta c_{bd} \cdot s_{bd} \tag{18}$$

where Δc_{bd} is the fall of maximum drop, and s_{bd} is the slope of the maximum drop.

(3) Rating deviation and aggregation

For each product, we use the *KL*-divergence from the distribution between members in A and other members in $U \backslash A$ to calculate the rating deviation and weight it by a balancing factor. From the perspective of rating distribution, the contrast suspiciousness satisfies Eq. (19).

$$\begin{cases} P(v_i|A) \propto q(\gamma_i) \\ \gamma_i = \sum\limits_{k \le K} F_k(v_i) \log \frac{F_k(v_i)}{F_k^{'}(v_i)} \\ \gamma_i = \min\left\{ \frac{f_A(v_i)}{f_{U \backslash A}(v_i)}, (v_i) \right\} \cdot \gamma_i \end{cases} \tag{19}$$

where k denotes the rating category, $F_k(v_i)$ denotes the frequency with which members in A rated product v_i with category k scores, and $F_k^{'}(v_i)$ denotes the frequency with which other members $U \backslash A$ rated product v_i with category k scores.

Ultimately, we use joint probability to aggregate the three signals above to obtain the contrast suspiciousness metric.

$$P(v_i|A) = q(\alpha_i) q(\beta_i) q(\gamma_i) = b^{\alpha_i + \beta_i + \gamma_i - 3} \tag{20}$$

The candidate group purification and ranking method

Based on the contrast suspiciousness metric, the spamicity for a reviewer can be calculated according to Eq. (21).

$$S(u_j \in A) = \sum_{v_i : (u_j, v_i) \in E} \sigma_{ji} e_{ji} P(v_i|A) \tag{21}$$

where σ_{ji} is the global suspiciousness on an edge, e_{ji} is the number of edges between (u_j, v_i), and $P(v_i|A)$ is the contrast suspiciousness.

To increase the association of a candidate group A with a set of products reviewed B, we use the expectation of the density score $D(A,B)$ over the probability $P(v_i|A)$ as the spamicity of a group. The objective function is defined according to Eq. (22).

$$\begin{aligned}\max_A Obj(A) &= \mathbb{E}[D(A,B)] \\ &= \frac{1}{|A| + \sum_{k\in V} P(v_k|A)} \sum_{i\in V} f_A(v_i) P(v_i|A)\end{aligned} \tag{22}$$

We describe the specific steps for group purification and ranking in Algorithm 6. The algorithm takes one candidate group A in *Candidate_Group* at a time as the input, and uses a priority tree to efficiently find the reviewer with the lowest spamicity in A and remove it. Then, the contrast suspiciousness changes, and the reviewer's spamicities are updated. The algorithm decreases A until A is empty, obtaining A^* that maximizes the value of the objective function. The A^* with a group size greater than or equal to 2 is placed into *Spammer_Group*. Based on spamicities, we rank the groups in *Spammer_Group*. Finally, the algorithm returns the top 300 most suspicious spammer groups.

Experiments

Dataset and human labeling

As there is no ground truth for spammer groups in the e-commerce field, we need to label the datasets to compare the performance of the spammer group detection methods. In this subsection, we first introduce the dataset used in the experiments and then detail the method for manually labeling the dataset.

Dataset

In our experiments, we use the unlabeled AmazonBooks review dataset. AmazonBooks is a dataset of book reviews from 1993 to 2014, which includes 22,507,155 reviews, 8,026,324 reviewers, and 2,330,066 products. Due to the large amount of review data, we only extracted data in 2013 for experiments according to the GSDB method (Ji et al. 2020). Finally, we got 6,990,316 reviews, 2,998,380 reviewers, and 1,079,741 products. Table 2 shows the statistics of the dataset.

Human labeling

The problem of spammer group detection is very challenging because of no available standard datasets with group labels for model building or method evaluation. Although our SGDCTH method is completely unsupervised and does not require any labels in its

Algorithm 6 GroupPurification&Ranking (*Candidate_Group* , *BiG* **)**

Input:

Candidate_Group : candidate group set

BiG : heterogeneous structural graph of candidate groups

Output:

Spammer_Group : spammer group set

```
Spammer_Group = ∅
for each A in Candidate_Group do
    P=calculate contrast suspiciousness given A via Eq.(20)
    S=calculate reviewer's spamicities in A via Eq.(21)
    MT=build priority tree of A with scores S
    while A is not empty do
        u =pop the reviewer of the minimum spamicity from MT
        A = A \ u , delete u from A
        update P with respect to new A
        update MT with respect to new P
    end while
    A* =the group that has the best objective obj(A*) so far via Eq.(22)
    if size ( A* )>=2 then
        Spammer_Group ← Spammer_Group ∪ {A*}
    end if
end for
rank Spammer_Group according to spamicities and obtain most suspicious top 300 spammer groups
return Spammer_Group
```

Table 2 Statistics of the dataset

Dataset	The whole AmazonBooks data	Data in 2013
#Review	22,507,155	6,990,316
#Reviewer	8,026,324	2,998,380
#Product	2,330,066	1,079,741

implementation, we need to obtain labels for the final groups to analyze the impact of parameter values on group detection performance and to achieve performance comparisons with baseline methods. Therefore, we hired three graduate student experts in the e-commerce environment to manually label the resulting top 300 spammer groups that are generated by each detection method and take these labels as ground truth.

Specifically, we use five individual spam indicators used by Ji et al. (2020) to label groups output by Algorithm 5, including Rating deviation (RD), Ratio of Extreme Rating (EXR), The Most Reviews One-day (MRO), Account Duration (AD), and Active time interval reviews (ATR). The human labeling method is divided into three steps. First, each group member is assigned 1 point for each spam judgment, 0.5 points for each borderline judgment, and 0 for non-spam judgment. Secondly, we calculate each group's total spamicity and average spamicity score according to the labels of its members. Thirdly, if the average spamicity score of a group is greater than or equal to 2/3, then the group will be labeled as a spammer group.

Baselines, evaluation metrics, and experimental setting

Baselines

To evaluate the performance of our method, we compare it with four classical unsupervised spammer group detection methods.

(1) GSDB (Ji et al. 2020). A review burst-based spammer group detection method. From the viewpoint of single-product, GSDB uses the Kernel Density Estimation (KDE) method to generate candidate groups from the review bursts of target products and further purify and classify the candidate groups to obtain spammer groups. The similarity between SGDCTH and GSDB is that both detect spammer groups from the viewpoint of products. The difference is that the SGDCTH method detects spammer groups for cross-product attacks and considers the structural-attribute correlation.

(2) GSBC (Wang et al. 2018). A graph-based spammer group detection method that introduces a top-down computational framework to identify spammer groups using the topology of a reviewer graph. The similarity between SGDCTH and GSBC is that both are graph-based methods. The difference is that the SGDCTH method takes products as the entry point and constructs a heterogeneous network to detect spammer groups.

(3) GroupStrainer (Ye and Akoglu 2015). A two-step method for discovering target products and spammer groups in a heterogeneous network. The similarity between SGDCTH and GroupStrainer is that both detect spammer groups in heterogeneous networks. The difference is that the SGDCTH method considers the structure-attribute correlation and uses the group purification method to further improve the performance of the spammer group detection method.

(4) HoloScope (Liu et al. 2018). A method that uses the Singular Value Decomposition (SVD) method in a heterogeneous network to detect dense subgraphs. The similarity between SGDCTH and HoloScope is that both detect spammer groups in heterogeneous networks. The difference is that the SGDCTH method is from the viewpoint of products and considers the structure-attribute correlation.

Evaluation metrics

As in previous work (Ji et al. 2020; Zhang et al. 2021, 2022a; Wang et al. 2016, 2018), we use *precision*, *recall*, and *F1* values as evaluation criteria, which are defined according to Eqs. (23), (24), and (25).

$$Precision = \frac{TP}{TP + FP} \tag{23}$$

$$Recall = \frac{TP}{TP + FN} \tag{24}$$

$$F1 = \frac{2 \cdot Precision \cdot Receall}{Precision + Recall} \tag{25}$$

where *TP*(True Positive) represents the number of spammer groups that are correctly detected, *FP* (False Positive) represents the number of true groups that are misjudged as a spammer group, and *FN* (False Negative) represents the number of spammer groups that are not accurately identified.

Table 3 The neurons in each layer for structural autoencoder and attribute autoencoder

	Structural autoencoder	Attribute autoencoder
Reviewer partition	128–64–128	23–64–23
Product partition	128–64–128	11–64–11

Precision reflects the number of correctly detected spammer groups as a percentage of the total number of groups predicted to be spammer groups, with larger values indicating better detection *precision*. *Recall* indicates the number of correctly detected spammer groups as a proportion of the total number of spammer groups in practice, with larger values indicating better detection performance. The *F1* value reconciles and averages the test *precision* and *recall*, reflecting the overall performance of the spammer group detection algorithm.

Experimental settings

We designed three sets of experiments based on the AmazonBooks dataset in 2013. The first set of experiments aims to analyze the impact of parameter values on the group detection performance of our method. The second set of experiments aims to evaluate the performance of our method by comparison with baseline methods, including two analyses. Specifically, the first analysis is the analysis of the *precision*, *recall*, and *F1* values for group detection methods, and the second is a comparative analysis of the time complexity of SGDCTH with baseline methods, which aims to verify the effectiveness and efficiency of our method. In the third set of experiments, we designed four variants of SGDCTH to verify the necessity of considering all available information and the step of group purification.

The SGDCTH method involves three parameters that need to be verified, i.e., the target product filtration threshold, the neighborhood radius threshold, and the minimum number of sample points threshold. The target product filtration threshold $\delta_{\tilde{p}}$ is between 0.5 and 0.7. To generate a suitable number of candidate groups, we select the neighborhood radius threshold ϵ is {0.4, 0.5, 0.6, 0.7, 0.8} and the minimum number of sample points threshold ϕ is {2, 3, 4} for experimental verification. In addition, we randomly initialize model parameters with a standard Xavier normal distribution (Glorot and Bengio 2010) and optimize the model with Adam (Kingma and Ba 2014). The number of GNN layers L is 2, the learning rate lr is 0.01, λ is set as 0 in Eq. (4), and the dimension of the raw structural features $\mathbf{x}$ is set as 128. We list the parameters for structural autoencoder and attribute autoencoder for two partitions in Table 3. The transformation kernel is set as 64–16. The number of epochs t is set as 50. The number of dynamic samples is set as 5. The dimension of the final embedding vector d is set as 128.

For the GSDB method, we set the target product filtration threshold δ_p as 0.1, the individual spamicity threshold δ_I as 0.43, and the group spamicity threshold δ_G as 0.57 to obtain a total of 320 groups. For the GSBC method, we set the co-review time window size τ as 30, the edge weight threshold δ as 0.1, the user-specified parameter MP as 1000, and the group spamicity threshold δ_G as 0.53 to obtain a total of 325 groups. For the GroupStrainer method, the target product filtration threshold δ_p is set as 0.75 to filter out target products with high suspicion. For the HoloScope method, the scaling base b is set to 32. In summary, we detail the parameter settings for each method in Table 4.

Results and analysis of parameter selection

Based on the parameter settings listed in Table 4, we perform the first set of experiments for parameter selection. In this section, we mainly explore the impacts of the target product filtration threshold, the neighborhood radius threshold, and the minimum number of sample points threshold on the detection performance of our method. Notably, when discussing one parameter, other parameters will be set to their best value.

Results and analysis of the target product filtration threshold

We draw a histogram of the NFS value distribution of products, as shown in Fig. 2. The frequencies of products with different NFS values show a skewed distribution, with most products having a concentrated distribution of NFS values between about 0.5 and 0.7. Since the Amazon dataset is relatively dense with reviews, too small a target product filtration threshold $\delta_{\tilde{p}}$ will increase the time complexity of our method, and too large a value will discard some reviewers in the process of constructing the graph, which negatively affects the algorithm's detection performance. In our experiments, we obtain through interpolation analysis that when the target product's filtration threshold $\delta_{\tilde{p}} > 0.65$, the product is vulnerable to attack. Therefore, we will set it as 0.65 and finally obtain 7027 target products.

Results and analysis of the neighborhood radius threshold

Our experiments found that the number of detected spammer groups decreases gradually as the value of ϵ increases. To generate a comparable number of groups as the baseline methods, the neighborhood radius threshold ϵ is set as {0.4, 0.5, 0.6, 0.7, 0.8}. The impact of ϵ on the detection performance of our method will be further explored.

Figure 3 shows the *precision* and *F1* values of our method are gradually improving as the value of ϵ

Table 4 Parameter settings

Method	Parameter	Description	Value
SGDCTH (Ours)	$\delta_{\bar{p}}$	Target product filtration threshold	0.5–0.7
	ϵ	Neighborhood radius threshold	0.4–0.8
	ϕ	Minimum number of sample points threshold	2, 3, 4
	L	Number of GNN layers	2
	lr	Learning rate	0.01
	λ	Learnable parameter or fixed scalar	0
	t	Number of epochs	50
	d	Embedding dimension of nodes	128
GSDB	δ_p	Target product filtration threshold	0.1
	δ_l	Individual spamicity threshold	0.43
	δ_G	Group spamicity threshold	0.57
GSBC	τ	Co-review time window size	30
	δ	Edge weight threshold	0.1
	MP	User-specified parameter	1000
	δ_G	Group spamicity threshold	0.53
GroupStrainer	δ_p	Target product filtration threshold	0.75
HoloScope	b	Scaling base	32

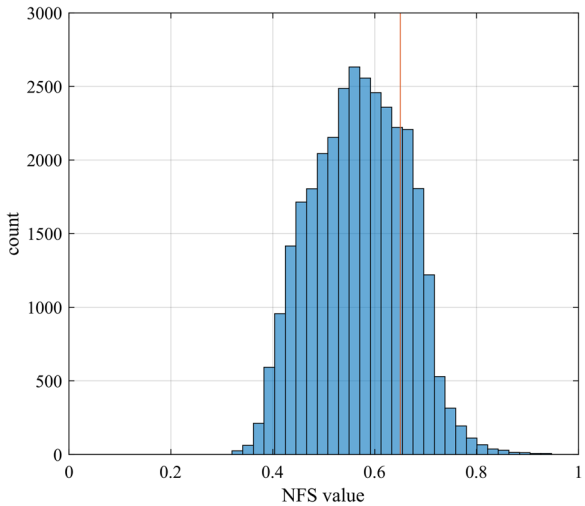

Fig. 2 A histogram of the NFS value distribution for products

increases. When the neighborhood radius ϵ is set to 0.6, the *precision* and *F1* values of the SGDCTH method are the highest for the top 300 groups, and the *recall* curve changes more gently. Following that, the *precision* of our method gradually decreases, as does the number of groups generated. When the neighborhood radius ϵ is set to 0.8, only 232 spammer groups are generated.

Results and analysis of the minimum number of sample points threshold

Consistent with the neighborhood radius threshold ϵ, the larger the value of the minimum number of sample points threshold, the smaller the number of discovering groups. Therefore, we set the minimum number of sample point thresholds ϕ as {2, 3, 4} to further explore the impact of parameter ϕ on the detection performance of our method.

Figure 4 shows the *precision* of SGDCTH gradually decreases as the value of ϕ increases. The *precision* of SGDCTH is lowest when ϕ is set as 4, and only 215 spammer groups are obtained. Although the *F1* value is lower for approximately the top 160 groups, after that, the *F1* value is higher than when ϕ is set as 3 or 4. On the whole, the detection performance of our method for the top 300 groups is better when the minimum number of sample points threshold ϕ is set as 2.

Results and comparison analysis for the group detection method

We implemented a second set of experiments to compare the performance of our method with baseline methods. The analysis is mainly carried out in two aspects, i.e., the comparative analysis of the *precision*, *recall,* and *F1* values and the time complexity.

Results and comparison analysis of the precision, recall, and F1 values

Based on the manual labeling of the top 300 groups detected by the GSDB, GSBC, GroupStrainer, HoloScope, and SGDCTH methods, we analyze the *precision*, *recall,* and *F1* values for SGDCTH with baseline methods, as shown in Fig. 5.

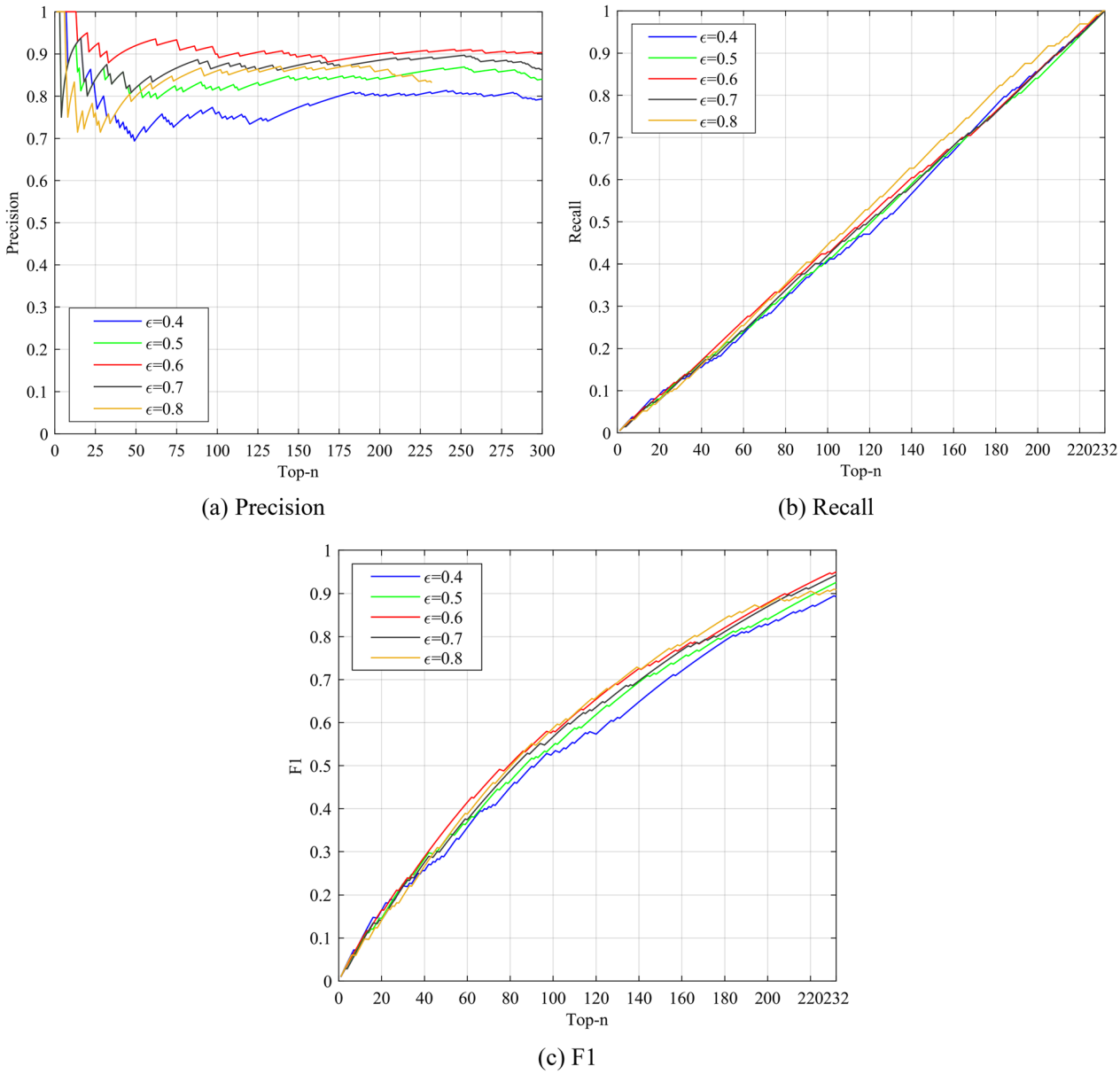

Fig. 3 The impact of the neighborhood radius threshold ϵ on SGDCTH

Figure 5a shows the *precision* curves of SGDCTH, GroupStrainer, and HoloScope consistently outperform the GSBC method, indicating that the heterogeneous graph-based method is capable of digging deeper into the implicit relationships among reviewers than the homogeneous graph-based method to capture spam groups with suspected collusion. Moreover, the *precision* curve of SGDCTH consistently outperformed that of GroupStrainer and HoloScope, indicating the effectiveness of comprehensively considering structural and attribute features of nodes. The *precision* curves of SGDCTH and GSDB cross at about the 185th group, before which GSDB outperforms SGDCTH, but after which SGDCTH outperforms GSDB. This is because GSDB only detects spammer groups in review bursts of a single target product, while SGDCTH can detect spammer groups that attack across multiple products, which is closer to the attack method of spammer groups in real life.

As can be seen in Fig. 5b, for about the top 130 groups, the *recall* curve of SGDCTH is slightly lower than that of GroupStrainer, which may be because our method did

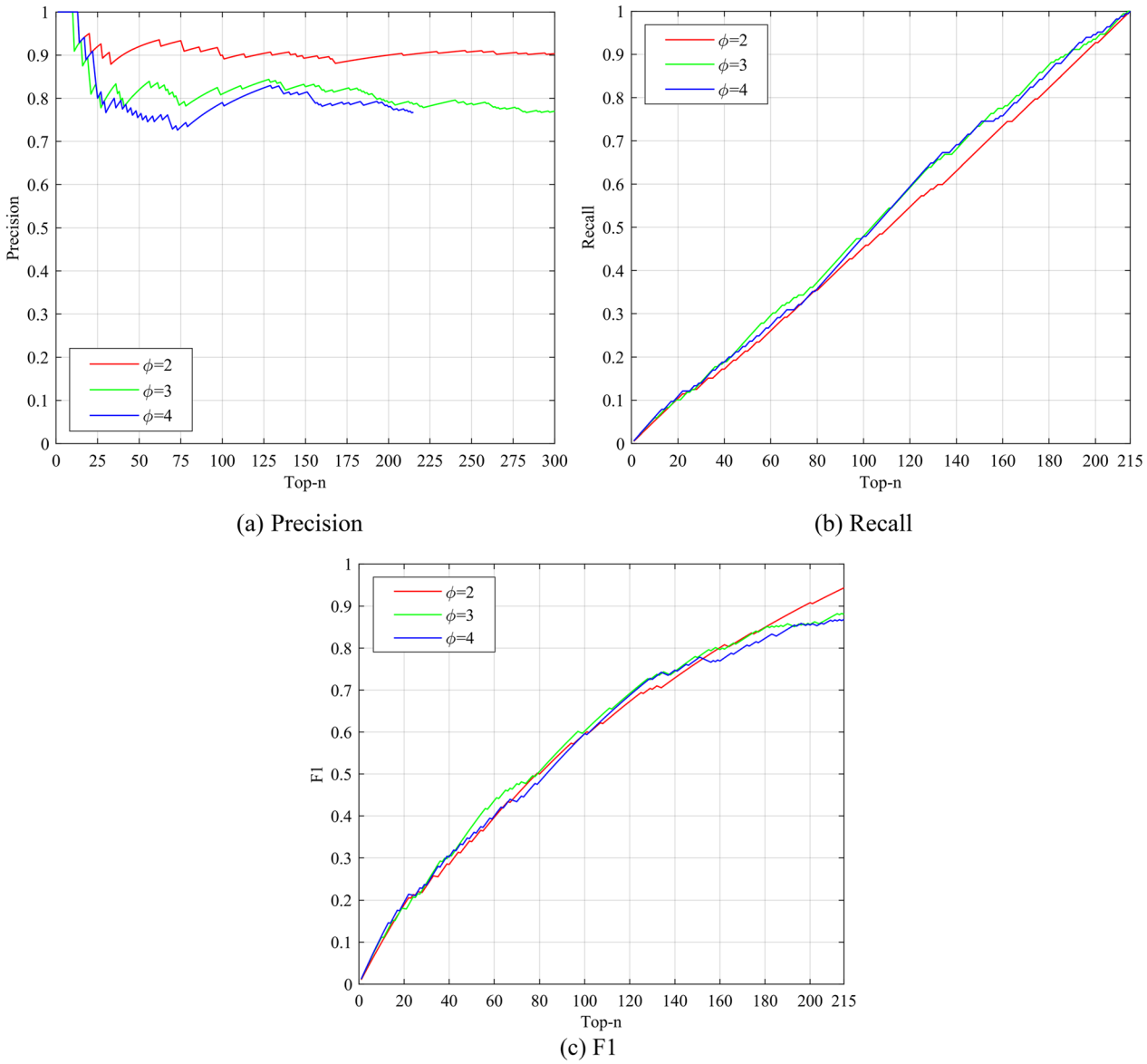

Fig. 4 The impact of the minimum number of sample points threshold φ on SGDCTH

not detect some spam groups that evaded detection for the purpose of camouflage. However, after that, the *recall* curve of SGDCTH is better than that of GroupStrainer, HoloScope. Overall, the SGDCTH method seems to have the smoothest *recall* curve fluctuations.

Figure 5c shows the *F1* value obtained by combining *precision* and *recall*. Also, the *F1* value curve of each method maintains a monotonically increasing state. After about the 120th group, the *F1* value of SGDCTH is better than that of the GSBC, GroupStrainer, and HoloScope methods. Finally, it surpasses the GSDB method, which illustrates the superiority of cross-product detection and collaborative training methods.

From this, we can draw two conclusions. (1) The SGDCTH, GroupStrainer, and HoloScope methods based on the heterogeneous network are better than the GSBC method based on the homogeneous network. (2) The SGDCTH method for detecting spammer groups in cross-product attacks is more consistent with real-life attacks on spammer groups than the GSDB method for detecting spammer groups from the review bursts of a single product.

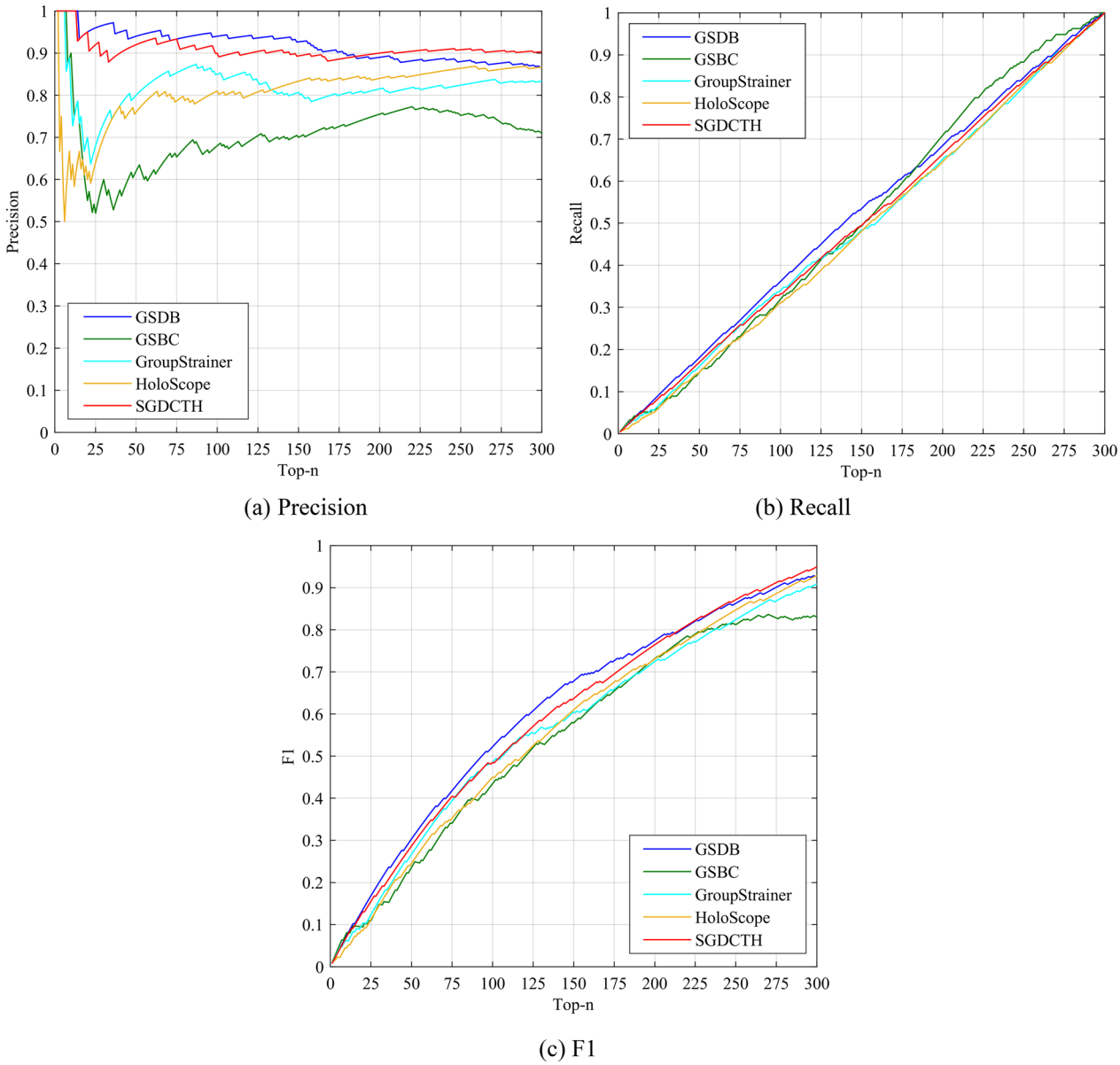

Fig. 5 The *precision*, *recall*, and *F1* values of the top 300 groups for SGDCTH with baseline methods

Comparison analysis of the time complexity

We compare and analyze the time complexity of the GSDB, GSBC, GroupStrainer, HoloScope, and SGDCTH methods, as shown in Table 5.

The GSDB method uses the KDE method to generate candidate groups in the review bursts of single-product with time complexity $O(n^2)$. The time complexity of the target product filtration and the group purification and classification are both $O(n)$, resulting in the total time complexity $O(n^2)$. The GSBC method uses three loop levels to construct a reviewer relationship graph with time complexity $O(n^3)$. The group generation and detection stage uses the min-cut method in one level of loops with a time complexity of $O(n^3)$, so the total time complexity is $O(n^3)$. The GroupStrainer method consists of two stages, i.e., target product detection and spammer group generation. Each stage has a time complexity $O(n^2)$, so the total time complexity is $O(n^2)$. For the HoloScope method, the time complexity of constructing the heterogeneous graph is $O(n^2)$, and the time complexity of detecting dense blocks using SVD is $O(n \log n)$, so the total time complexity is $O(n^2)$. For the SGDCTH method, the time complexity of the target product filtration is $O(n^2)$, the time complexity of constructing heterogeneous graph is $O(n^2)$,

Table 5 The time complexity of SGDCTH with baseline methods

Method	Target product filtration	Construct graph	Feature representation learning	Candidate groups generation	Group purification and ranking (or classification)	Total of time complexity
GSDB	$O(n)$	×	×	$O(n^2)$	$O(n)$	$O(n^2)$
GSBC	×	$O(n^3)$	×	$O(n^3)$	$O(n)$	$O(n^3)$
GroupStrainer	$O(n^2)$	$O(n^2)$	×	$O(n^2)$	×	$O(n^2)$
HoloScope	×	$O(n^2)$	×	$O(n \log n)$	×	$O(n^2)$
SGDCTH	$O(n^2)$	$O(n^2)$	$O(n \log n)$	$O(n^2)$	$O(n)$	$O(n^2)$

the time complexity of node feature representation learning is $O(n \log n)$, the time complexity of candidate groups generation is $O(n^2)$, and the time complexity of the group purification and ranking stage is $O(n)$, so the total time complexity is $O(n^2)$.

Overall, the total time complexity of GSBC is $O(n^3)$, while the total time complexity of the GSDB, GroupStrainer, HoloScope, and SGDCTH methods are all $O(n^2)$. In addition, since the SGDCTH method first filters target products vulnerable to attack by spammers, it makes SGDCTH focus on the review data closely related to target products, which greatly improves the algorithm's efficiency.

Analysis of ablation

We conduct an ablation analysis to evaluate our method and configure SGDCTH to the following settings.

(1) SGDCTH_TP. A variant of our method, which utilizes the behavioral metric combining the abnormal distributions of product rating and product average rating used by Ji et al. (2020) in filtering target products.
(2) SGDCTH_Stru. A variant of our method that only utilizes structural features, but ignores attribute and structure-attribute correlation features.
(3) SGDCTH_Attr. A variant of our method that only utilizes attribute features, but ignores structural and structure-attribute correlation features.
(4) SGDCTH_Stru+Attr. A variant of our method that utilizes structural and attribute features, but ignores structure-attribute correlation features.
(5) SGDCTH_DPC. A variant of our method that utilizes the Density Peaks Clustering (DPC) method to discover candidate groups in the vector space of reviewers.
(6) SGDCTH_K-Means. A variant of our method that utilizes the K-Means clustering method to discover candidate groups in the vector space of reviewers.
(7) SGDCTH_No purification. A variant of our method that ignores the step of group purification.

We analyze the *precision*, *recall,* and *F1* values for SGDCTH with seven variants, as shown in Fig. 6. The *precision* curve in Fig. 6a shows that SGDCTH achieves the best performance. This is because it comprehensively considers structure, attribute, and structure-attribute correlation features when detecting spammer groups. Furthermore, it uses a more robust NFS metric to filter target products and a group purification method to filter innocent members of candidate groups generated by the DBSCAN method, which further improves the performance of SGDCTH. SGDCTH_TP shows inferior performance to SGDCTH, indicating the NFS metric is more robust to evasion than behavioral metrics (Ye and Akoglu 2015). In the *precision* curve between the 115th and 240th groups, SGDCTH_Stru and SGDCTH_Attr show inferior performance to SGDCTH_Stru+Attr, which indicates the necessity of considering structure and attribute features comprehensively. The performance of SGDCTH_Stru+Attr is inferior to that of SGDCTH, indicating the importance of considering the structure-attribute correlation features. The detection performance of SGDCTH_DPC and SGDCTH_K-Means is inferior to that of SGDCTH, which indicates that DBSCAN is better than DPC and K-Means at discovering spammer groups that attack target products separately. In addition, SGDCTH_No purification shows inferior performance to SGDCTH, indicating the necessity of group purification.

The *recall* curve in Fig. 6b shows that SGDCTH_Attr and SGDCTH_Stru+Attr have a higher *recall*, while SGDCTH_Stru has a lower *recall*. This indicates that the attribute features of nodes are somewhat adversarial. Spammers are prone to disguising their relationship with other members of groups (i.e., structural features) to evade the detector.

The *F1* curve in Fig. 6c shows that SGDCTH_Attr, SGDCTH_Stru, and SGDCTH_K-Means have the worst performance, indicating that all available information and a better clustering method should be considered when designing the detector to enhance its robustness.

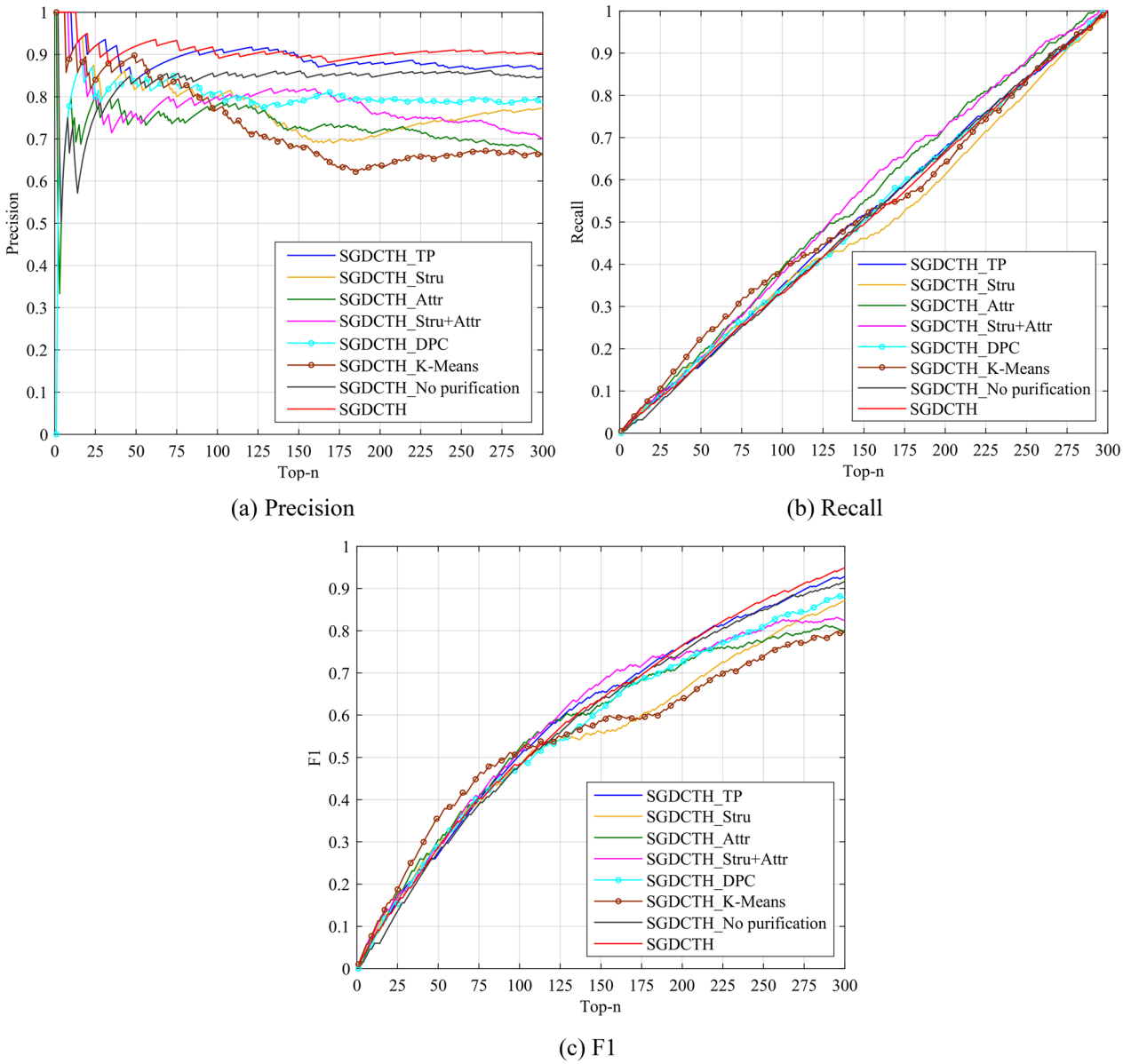

Fig. 6 The *precision*, *recall*, and *F1* values of the top 300 groups for SGDCTH with its variants

Conclusion

Online fake reviews have increasingly become a real threat to e-commerce evaluation and reputation systems, and detecting spammer groups is key to ensuring the credibility of review information on e-commerce websites. This paper proposes a collaborative training-based algorithm for detecting spammer groups in a heterogeneous network called SGDCTH. It greatly reduces the algorithm's time complexity by filtering target products vulnerable to spammer attacks from the products' viewpoint. To effectively learn low-dimensional vector representations of nodes, the SGDCTH method uses a collaborative training method to model the intra-partition and inter-partition proximity of a heterogeneous network, which considers the structure-attribute correlation. This makes our method detect suspicious spammer groups in e-commerce in terms of structure and attributes. Since genuine reviewers are easily mixed into the detected candidate groups and misjudged by the detector as spammers, SGDCTH utilizes the group purification method to filter the innocent individuals in candidate groups. This further improves the performance of our SGDCTH method.

Although our SGDCTH method achieves good performance, there is still room for improvement. For example, we use the DBSCAN clustering method to generate candidate groups, but we need to manually set two thresholds. We will explore a method to automatically learn these two thresholds to generate higher-quality groups. Future work also includes designing methods to learn node features more efficiently in heterogeneous networks, as well as simulating the attack patterns of spammer groups to write fake reviews and injecting these data into real datasets to evaluate the performance of the detection method.

Abbreviations

FIM	Frequent item mining
NFS	Network footprint score
B&C	Group behavior and content
S	Group structure
B&C+S	Group behavior and structure
HIN	Heterogeneous information network
HISN	Heterogeneous induced sub-network
KDE	Kernel density estimation
SVD	Singular value decomposition
RD	Rating deviation
EXR	Ratio of extreme rating
MRO	Most reviews one-day
AD	Account duration
ATR	Active time interval reviews

Acknowledgements
The authors would like to thank the editor and anonymous referees for their constructive comments.

Author contributions
QZ completed the writing and experiments for the manuscript, ZL and BX examined and validated experiments, and SJ and DKWC provided guidance and suggestions for revision. All authors read and approved the final manuscript.

Funding
This paper is supported in part by the Natural Science Foundation of China (No. 71772107, 62072288), Shandong Nature Science Foundation of China [Grant No. ZR2019MF003, ZR2020MF044].

Availability of data and materials
When certain data sharing requirements are met, the data is available upon request. Such requests should be sent to the first author of this paper.

Declarations

Competing interests
The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Received: 22 February 2023 Accepted: 13 April 2023
Published online: 02 October 2023

References

Akoglu L, Chandy R, Faloutsos C (2013) Opinion fraud detection in online reviews by network effects. In: Proceedings of the international AAAI conference on web and social media, vol 7, 1st edn. pp 2–11
Cao N, Ji S, Chiu DK, He M, Sun X (2020) A deceptive review detection framework: combination of coarse and fine-grained features. Expert Syst Appl 156:113465
Cao N, Ji S, Chiu DK, Gong M (2022) A deceptive reviews detection model: separated training of multi-feature learning and classification. Expert Syst Appl 187:115977
Chao J, Zhao C, Zhang F (2022) Network embedding-based approach for detecting collusive spamming groups on E-commerce platforms. In: Security and communication networks, pp 1–13
Ester M, Kriegel HP, Sander J, Xu X (1996) A density-based algorithm for discovering clusters in large spatial databases with noise. In: KDD, vol 96, 34th edn. pp 226–231
Glorot X, Bengio Y (2010). Understanding the difficulty of training deep feedforward neural networks. In: Proceedings of the thirteenth international conference on artificial intelligence and statistics. JMLR Workshop and Conference Proceedings, pp 249–256
Hu Y (2021) Unsupervised learning for spammer group detection based on network representation. Univ Electron Sci Technol China. https://doi.org/10.27005/d.cnki.gdzku.2021.000829
Huang W, Li Y, Fang Y, Fan J, Yang H (2020) BiANE: Bipartite attributed network embedding. In: Proceedings of the 43rd international ACM SIGIR conference on research and development in information retrieval. pp 149–158
Ji SJ, Zhang Q, Li J, Chiu DK, Xu S, Yi L, Gong M (2020) A burst-based unsupervised method for detecting review spammer groups. Inf Sci 536:454–469
Jindal N, Liu B (2008) Opinion spam and analysis. In: Proceedings of the 2008 International Conference on Web Search and Data Mining. pp 219–230
Kingma DP, Ba J (2014) Adam: a method for stochastic optimization. arXiv preprint arXiv:1412.6980
Li FH, Huang M, Yang Y, Zhu X (2011) Learning to identify review spam. In: Twenty-second international joint conference on artificial intelligence
Li H, Fei G, Wang S, Liu B, Shao W, Mukherjee A, Shao J (2017) Bimodal distribution and co-bursting in review spam detection. In: Proceedings of the 26th international conference on World Wide Web. pp 1063–1072
Liu S, Hooi B, Faloutsos C (2018) A contrast metric for fraud detection in rich graphs. IEEE Trans Knowl Data Eng 31(12):2235–2248
Luca M (2016) Reviews, reputation, and revenue: the case of Yelp. Com. (March 15, 2016). Harvard Business School NOM Unit Working Paper, (12-016)
Mukherjee A, Liu B, Glance N (2012). Spotting fake reviewer groups in consumer reviews. In: Proceedings of the 21st International Conference on World Wide Web. pp 191–200
Mukherjee A, Kumar A, Liu B, Wang J, Hsu M, Castellanos M, Ghosh R (2013) Spotting opinion spammers using behavioral footprints. In: Proceedings of the 19th ACM SIGKDD international conference on knowledge discovery and data mining. pp 632–640
Ott M, Choi Y, Cardie C, Hancock JT (2011) Finding deceptive opinion spam by any stretch of the imagination. arXiv preprint arXiv:1107.4557
Rayana S, Akoglu L (2015) Collective opinion spam detection: Bridging review networks and metadata. In: Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining. pp 985–994
Shehnepoor S, Togneri R, Liu W, Bennamoun M (2021) HIN-RNN: a graph representation learning neural network for fraudster group detection with no handcrafted features. In: IEEE transactions on neural networks and learning systems. pp 1–14
Shehnepoor S, Togneri R, Liu W, Bennamoun M (2022) Spatio-temporal graph representation learning for fraudster group detection. In: IEEE transactions on neural networks and learning systems. pp 1–15
Wang G, Xie S, Liu B, Yu PS (2012) Identify online store review spammers via social review graph. ACM Trans Intell Syst Technol (TIST) 3(4):1–21
Wang Z, Hou T, Song D, Li Z, Kong T (2016) Detecting review spammer groups via bipartite graph projection. Comput J 59(6):861–874
Wang Z, Gu S, Zhao X, Xu X (2018) Graph-based review spammer group detection. Knowl Inf Syst 55(3):571–597
Wang J, Guo Y, Wen X, Wang Z, Li Z, Tang M (2020) Improving graph-based label propagation algorithm with group partition for fraud detection. Appl Intell 50(10):3291–3300
Wang X, Liu N, Han H, Shi C (2021a) Self-supervised heterogeneous graph neural network with co-contrastive learning. In: Proceedings of the 27th ACM SIGKDD conference on knowledge discovery & data mining. pp 1726–1736

Wang Y, Zhang J, Guo S, Yin H, Li C, Chen H (2021b) Decoupling representation learning and classification for GNN-based anomaly detection. In: Proceedings of the 44th international ACM SIGIR conference on research and development in information retrieval. pp 1239–1248
Wang X, Bo D, Shi C, Fan S, Ye Y, Philip SY (2022) A survey on heterogeneous graph embedding: methods, techniques, applications and sources. IEEE Trans Big Data 9(2):415–436
Xu C, Zhang J, Chang K, Long C (2013) Uncovering collusive spammers in Chinese review websites. In: Proceedings of the 22nd ACM international conference on information & knowledge management. pp 979–988
Xu K, Hu W, Leskovec J, Jegelka S (2018) How powerful are graph neural networks? arXiv preprint arXiv:1810.00826
Ye J, Akoglu L (2015) Discovering opinion spammer groups by network footprints. In: Machine learning and knowledge discovery in databases: European conference, ECML PKDD 2015, Porto, Portugal, September 7–11, 2015, Proceedings, Part I 15. pp 267–282
Zhang F, Hao X, Chao J, Yuan S (2020a) Label propagation-based approach for detecting review spammer groups on e-commerce websites. Knowl-Based Syst 193:105520
Zhang Y, Li Y, Gu X, Ji S (2021) A group spam detection algorithm combining behavior and structural feature reasoning. Comput Eng Sci 43(05):926–935
Zhang Q, Ji S, Zhang W et al (2022a) Group spam detection algorithm considering structure and behavior characteristics. Appl Res Comput 39(05):1374–1379
Zhang F, Yuan S, Wu J, Zhang P, Chao J (2022b) Detecting collusive spammers on e-commerce websites based on reinforcement learning and adversarial autoencoder. Expert Syst Appl 203:117482
Zhang S, Yin H, Chen T, Hung QVN, Huang Z, Cui L (2020b) GCN-based user representation learning for unifying robust recommendation and fraudster detection. In: Proceedings of the 43rd international ACM SIGIR conference on research and development in information retrieval. pp 689–698
Zheng M, Zhou C, Wu J, Pan S, Shi J, Guo L (2018) FraudNE: a joint embedding approach for fraud detection. In: 2018 international joint conference on neural networks (IJCNN). IEEE, pp 1–8
Zhu C, Zhao W, Li Q, Li P, Da Q (2019) Network embedding-based anomalous density searching for multi-group collaborative fraudsters detection in social media. Comput Mater Continua 60(1):317–333

Publisher's Note