2021 IEEE International Conference on Artificial Intelligence and Computer Applications (ICAICA) | 978-1-6654-1867-6/21/$31.00 ©2021 IEEE | DOI: 10.1109/ICAICA52286.2021.9498095

Detecting group shilling attacks in recommender systems based on maximum dense subtensor mining

Hongtao Yu, Haihong Zheng, Yishu Xu, Ru Ma, Dingli Gao, Fuzhi Zhang
School of Information Science and Engineering, Yanshan University
Qinhuangdao, Hebei, China
yu5771@163.com, 1299535254@qq.com, xuyishu1@126.com, 3023349202@qq.com, 951932919@qq.com, xjzfz@ysu.edu.cn

***Abstract*—Existing group shilling attack detection methods mainly depend on human feature engineering to extract group attack behavior features, which requires a high knowledge cost. To address this problem, we propose a group shilling attack detection method based on maximum density subtensor mining. First, the rating time series of each item is divided into time windows and the item tensor groups are generated by establishing the user-rating-time window data models of three-dimensional tensor. Second, the M-Zoom model is applied to mine the maximum dense subtensor of each item, and the subtensor groups with high consistency of behaviors are selected as candidate groups. Finally, a dual-input convolutional neural network model is designed to automatically extract features for the classification of real users and group attack users. The experimental results on the Amazon and Netflix datasets show the effectiveness of the proposed method.**

Keywords—recommender systems, group shilling attacks, maximum dense subtensor mining, dual-input convolutional neural network

III. INTRODUCTION

Recommender systems have been widely used in many fields to provide personalized recommendations for users according to their historical rating information [1]. However, Such systems are easily manipulated by malicious users. These users can inject a large number of fake profiles to make the system produce recommendation results which are beneficial to them. Such behaviors are generally termed shilling attacks [2]. In the traditional shilling attacks, attackers inject fake profiles into the recommender system separately to achieve the purpose of biasing recommendation results. In fact, attackers might corporately inject fake profiles into the recommender system according to a certain strategy. This behavior is termed a group shilling attack [3]. Compared with the single unorganized attacker, group attackers have a certain strategy and concealment. They are more harmful to recommender systems. Therefore, how to effectively detect group shilling attacks is the key issue in the field of recommender systems.

Over the past few years, research on shilling attack detection has made a great progress. However, existing methods mainly focus on detection of individual shilling attackers [4-6] and rarely take the collusion behaviors of attackers into consideration. Although a few methods are proposed to detect shilling groups in recommender systems, the performance of these methods still needs improving when detecting high diversity group attack profiles. Recently, some researchers use the concentration of item rating time to divide candidate groups. However, for the small attack groups, these detection methods are not optimal, and the detection recall is poor. There are also some methods [7-10] proposed for detecting spammer groups in consumer review websites, but these methods are not suitable for detecting shilling groups.

IV. RELATED WORK

In recent years, detection of shilling groups has become a research hotspot. Zhou et al [11] utilized two detection indicators to obtain suspicious user profiles and used target item analysis to get the final detection results. Wang et al. [12] used the AP clustering to divide users into groups, then they found target items and identified shilling groups. However, the detection performance of this method is limited when detecting group shilling profiles with high diversity. Wang et al. [13] first used frequent itemset mining to obtain candidate groups, then proposed attack group detection indicators, and finally used a principal component analysis-based ranking method to find attack groups. However, frequent itemset mining method is more sensitive to the threshold setting when generating candidate groups. Zhang et al. [14] divided candidate groups according to the concentration of rating time, and then used the bisecting K-Means clustering algorithm to identify shilling groups. However, this method is not effective for detecting small shilling groups.

V. DESIGN OF ALGORITHM FRAMEWORK

A. Dividing candidate groups

In this subsection, we divide candidate groups through finding each item's dense rating behavior. First, we give two definitions.

Definition 1. item tensor group: The item tensor group is a set of item tensors corresponding to all items in the dataset, that is:

$$A = \{A_i \mid i \in I\} \tag{1}$$

Definition 2. (*item rating time series, IRTS*). For any item $i \in I$, the item i's rating time series refers to timestamps when item i was rated in the ascending order, which is denoted as follows:

$$IRTS_i = \{t_1^i, t_2^i, ..., t_f^i\} \tag{2}$$

where t_f^i represents item *i* rated by users at time t_f.

The process of constructing item tensor groups is as follows. First, for each item in the dataset, the item rating time series is constructed. Second, the disjoint time windows are divided on item rating time series by the length of 30 days, and then the user-rating-time window tensor model is established. For blocks in the three-dimensional tensor, if a user rates an item at a specific time, the value of this block is assigned to 1, otherwise, the value is assigned to 0.

Let A_i represent the three dimensional tensor of item *i*, ITW_j^i represent the *j*th time window of item *i*, and *tw* represent the time window. The algorithm of constructing item tensor groups is described as follows.

Algorithm 1 Constructing item tensor groups

Input: the rating dataset *RD*=(*U*,*I*,*R*,*T*)
Output: set of item tensor groups *A*

$RD' \leftarrow \varnothing$, $A \leftarrow \varnothing$
for each item $i \in I$ **do**
  $TW \leftarrow \varnothing$
  $IRTS_i \leftarrow$ construct the time series of item i
  $l = 1$, $j = 1$
  while $l \le |IRTS_i|$ **do**
    $ITW_j^i \leftarrow \phi$
    for each time $t_m^i \in IRTS_i$ **do**
      if $t_m^i \ge t_l^i$ and $t_m^i - t_l^i \le 30$ **then**
        $ITW_j^i \leftarrow ITW_j^i \cup \{t_m^i\}$, $tw \leftarrow j$
        $TW \leftarrow TW \cup \{tw\}$, $RD' \leftarrow RD' \cup \{u,i,r,tw\}$
      end if
    end for
    $l \leftarrow l + |ITW_j^i|$, $j = j + 1$
  end while
end for
for each item $i \in T$ **do**
  $A_i \leftarrow 0_{|S_u| \times |S_r| \times |S_{tw}|}$
  for each (u,i,r,tw) in RD'
    $A_i[u][r][tw] \leftarrow 1$
  end for
  $A \leftarrow A \cup \{A_i\}$
end for
return A

Definition 3. Arithmetic average mass [16]: For any subtensor $S \in A$, M_S, D_S, N represent the sum of the weights, the size, and the number of dimensions, respectively. That is:

$$\rho_S = GAM_S = M_S / (D_S / N) \tag{3}$$

The dense subtensors of the tensor represents the synchronous behaviors of a group of users. Therefore, the generation of candidate groups can be formulated as the problem of finding the dense subtensors. First, suspicious user groups are generated using the M-zoom model [16] to mine the maximum dense subtensor of item tensor groups. Second, candidate groups are obtained by finding suspicious user groups whose density is greater than the given threshold. The algorithm of generating candidate groups is described as follows.

Algorithm 2 Generating candidate groups

Input: set of item tensor groups A, density threshold p
Output: set of Candidate groups G

$G \leftarrow \varnothing$
for each $A_i \in A$ **do**
  $subtensor_i \leftarrow do\ M - Zoom(A_i)$
  $G \leftarrow G \cup \{subtensor_i\}$
end for
for each $G_i \in G$ **do**
  calculate ρ_i according to Eq. (3)
  if $\rho_i < p$ **then**
    $G \leftarrow G - \{G_i\}$
  end if
end for
return G

B. Construction of user relevance rating matrices

In the section, we construct the user group relevance matrix and user item popularity rating matrix, respectively.

Definition 4. (*user relevance series, URS*). For any $u \in U$, the *u's* relevance series refers to the set of all users who belong to the same candidate group with user *u*, that is:

$$URS(u) = \{v_1, v_2, ..., v_m\} \tag{4}$$

Definition 5. (*user relevance matrix, URM*). For any $u \in U$, the user *u's* relevance matrix reflects the relationship between user *u* and other users in the dataset.

The elements in *URM* represent the relevance of users. If two users belong to at least one candidate group, the relevance is 1; otherwise, the relevance is 0.

Definition 6. (*user item popularity rating series, UIPRS*). For any $u \in U$, the item popularity rating series of user *u* is the series of item-rating pairs between user *u* and all items in the dataset sorted on the item's popularity in ascending order, denoted as

$$UIPRS(u) = \{(i_1, r_{i_1}^u), (i_2, r_{i_2}^u), ..., (i_f, r_{r_f}^u)\} \tag{5}$$

where i_j is the *j*th item, $r_{i_j}^u$ is the rating of user *u* to item i_j. In this paper, we use the number of ratings of all users in the dataset to calculate the popularities of items [17].

Definition 7. (*user item popularity rating matrix, UIPRM*). For any $u \in U$, the user *u's* item popularity rating matrix reflects the preference of user *u* for items.

The elements in *UIPRM* represent the ratings of users to items, which are determined as follows:

$$UIPRM(u)_{j,k} = \begin{cases} UIPRS(u)^r_{j \times n + (k+1)}, & j \times n + (k+1) \le |I| \\ 0, & j \times n + (k+1) > |I| \end{cases} \tag{6}$$

where $n = \max\left(\left\lceil \sqrt{|U|} \right\rceil, \left\lceil \sqrt{|I|} \right\rceil\right)$ represents the number of

rows of user relevance matrix, $j=v/n$ and $k=v\ \%\ n$ represent the row and column of the matrix, respectively, $UIPRS(u)^{r}_{j\times n+(k+1)}$ is the rating of the $j\times n+(k+1)$ th element in *UIPRS* of user u.

The algorithm of constructing user relevance rating matrix is as follows.

Algorithm 3 Constructing user relevance rating matrix

Input: dataset *RD* , set of candidate groups *G*

Output: relevance rating matrices *URM*, *UIPRM*

1. $IPop \leftarrow \varnothing$, $n=\max\left(\left\lceil\sqrt{|U|}\right\rceil,\left\lceil\sqrt{|I|}\right\rceil\right)$
2. **for** each user $i\in I$ **do**
3. $IPop_i \leftarrow$ Calculate the popularity of item i
4. $IPop \leftarrow IPop\cup\{IPop_i\}$
5. **end for**
6. **for** each user $u\in U$ **do**
7. $UIPRS(u)\leftarrow\varnothing$
8. **for** each item $i\in I$ **do**
9. **if** user u has rated item i **then**
10. $UIPRS(u)\leftarrow UIPRS(u)\cup\{i,r\}$
11. **end if**
12. **end for**
13. sort $UIPRS(u)$ on popularity in descending order
14. $UIPRM(u)\leftarrow 0_{|n|\times|n|}$
15. **for** each element in $UIPRS(u)$
16. calculate $UIPRM(u)_{j,k}$ according to Eq. (6)
17. **end for**
18. $URS(u)\leftarrow$ construct the relevance series of user u
19. $URM(u)\leftarrow 0_{|n|\times|n|}$
20. **for** each element in $URS(u)$
21. calculate $URM(u)_{j,k}$
22. **end for**
23. **end for**
24. **return** *URM*, *UIPRM*

C. Detection of shilling groups

Based on the above work, we devise a dual-input convolutional neural network model to detect shilling groups. The designed dual-input convolutional neural network model, which is called DIC, is shown in Fig.1. The model includes input layer, hidden layer, and output layer. The parameters of hidden layer in DIC are shown in Table I.

a) Design of the Input layer

The DIC has two parallel input layers, which uses equal-sized user relevance matrix and user item popularity rating matrix as the input.

b) Hidden layers with the feature pyramid networks

The details of hidden layer are as follows. First, three feature maps with different scales are obtained. Two of them are from two convolution blocks (Conv1 and Conv2) and the remaining one is just the original matrix (user relevance matrix or user item popularity rating matrix). Second, the feature pyramid networks use two 1×1 convolution blocks (Conv3, Conv4) to unify the number of channels of feature maps to 64. The feature maps are fused through two upsampling and zooming operations and the results of the two feature pyramids are concatenated. Finally, two consecutive maximum pooling layers are used for dimensionality reduction.

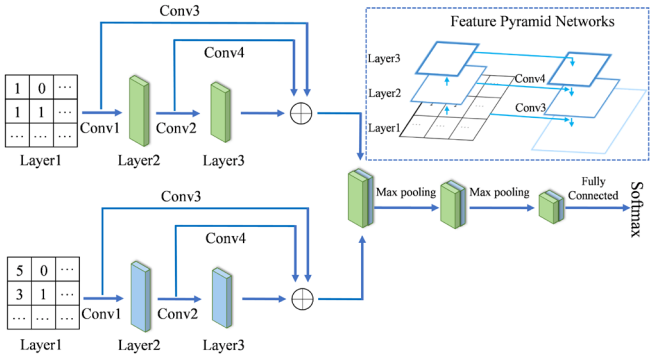


Fig. 1. Dual-input convolutional neural network model

TABLE I. NETWORK STRUCTURE PARAMETERS

	layer	size	stride/padding	kernel
Conv1	convolutional	5×5	1/0	32
	Max pooling	2×2	2/0	null
Conv2	convolutional	5×5	1/0	64
	Max pooling	2×2	2/0	null
pooling	Max pooling	2×2	1/0	null
pooling	Max pooling	2×2	2/0	null

c) Design of the Output layer

We set the output category of the fully connected layer to 2. The final output results (y_0,y_1) represent the probability that a user is an attacker or a genuine user, which are determined as follows:

$$pro_u=\begin{cases}0\ , & y_0\geq y_1\\ 1\ , & y_1\geq y_0\end{cases} \tag{7}$$

where pro_u represents the final label of user u, 0 represents the attacker, and 1 represents the genuine user.

d) Model implementation details

To train the DIC model, the user relevance rating matrices are randomly divided into 60% and 40%, which are used as the training set and test set, respectively. Parameters epoch, minibatch, and learning rate are set to 150, 128, and 0.0001, respectively. The cross entropy loss is used as model loss function.

VI. EXPERIMENTS AND ANALYSIS

A. Experimental datasets

1) Netflix dataset. This dataset consists of 103297638 ratings generated by 480186 users' evaluations of 1770 movies. All ratings are integer values between 1 and 5. In this paper, 229686 ratings of 3998 movies from 2000 users were randomly selected as the experimental dataset. We use the group attack models *GSAGen*$_1$*Ran* and *GSAGen*$_1$*Avg* [18]

to generate 10 attack groups that include 137 attackers and inject into the Netflix sampled dataset to construct a synthetic dataset.

2) Amazon dataset. This dataset was crawled from Amazon.cn. The rating range is an integer from 1 to 5. In this paper, some labeled users are randomly selected as the experimental dataset. The dataset is consisting of 53777 ratings that 5055 users gave to 17610 products, which includes 1937 attack users and 3118 genuine users.

B. Evaluation Metrics

We use the precision, recall, and F1-measure metrics to evaluate the performance of GAD-MDST, which are below:

$$Precision = \frac{TP}{TP + FP} \tag{9}$$

$$Recall = \frac{TP}{TP + FN} \tag{10}$$

$$F1\text{-}measure = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{11}$$

where TP is the number of attackers correctly detected, FN is the number of attackers misclassified as genuine users, and FP is the number of genuine users misclassified as attackers.

C. Experiment Results

We compare GAD-MDST with the following four baseline methods.

1) GD-BKM [14]: A group attack detection method based on bisecting K-means clustering, which divides candidate groups according to the rating tracks of items and uses the bisecting K-means algorithm to obtain attack groups.

2) CBS [15]: A shilling attack detection method, which ranks users according to their spam probability values. In the experiment, we randomly selected 27 and 5 attack users as the seeds of CBS on the Amazon and Netflix datasets, respectively.

3) URGSSGD [19]: A method for spammer group detection, which constructs the user relationship graph and detects the abnormal structure of the graph by analyzing the spectrum features. On the Amazon dataset, we set $l = 100,\ k = 1,\ c = \sqrt{3/2},\ \gamma = 1.3$, and $\gamma_{sg} = 0.1$. On the Netflix dataset, we set $l = 100,\ k = 40,\ c = \sqrt{3/2},\ \gamma = 2$, and $\gamma_{sg} = 0.01$.

4) IRM-TIA [20]: An unsupervised approach for shilling attack detection, which extracts the behavior characteristics by analyzing item relationships and generates the set of suspicious users by using PCA and the k-means clustering algorithm and identifies attackers from suspicious user set according to target items. For the Amazon dataset, we set $\varepsilon = 0.05,\ \theta = 3$, and $\delta = 0.9$. For the Netflix dataset, we set $\varepsilon = 0.3,\ \theta = 10$, and $\delta = 0.5$.

a) Performance comparison of five methods on the Netflix dataset

Fig. 2 shows the precision, recall, and F1-measue of five methods on the Netflix dataset. The recall of GAD-MDST, GD-BKM, CBS, and IRM-TIA on the Netflix dataset is above 0.8, which shows that these four methods can effectively detect the group attack on the Netflix dataset. The detection performance of URGSSGD is poor because users in the same group have no more than three co-rated movies in the Netflix dataset and most attackers are filtered by this method. The precision, recall, and F1-measure of GAD-MDST are about 0.978, 0,99, and 0.986, respectively. Therefore, GAD-MDST has better detection than four baseline methods.

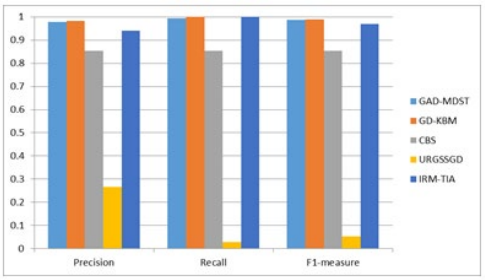


Fig. 2. Compare the precision, recall and F1-measure of the five methods on the Netflix datasets

b) Comparison of the five methods on the Amazon dataset

Fig. 3 shows that the precision, recall, and F1-measure of GAD-MDST on the Amazon dataset are 0.86, 0.82, and 0.84, respectively. The GD-BKM has high detection precision on the Amazon dataset, but the recall is only 0.67. The precision, recall, and F1-measure of CBS is about 0.46, 0.30, and 0.36, respectively. The precision of CBS cannot reach 0.5 because CBS is limited by the initial number of seeds. Although the precision of URGSSGD can reach 0.8, its recall and F1-measure are only 0.2 and 0.3, respectively. Therefore, the overall detection effect is poor. The main reason is that a large number of attackers are filtered out. The precision, recall, and F1-measure of IRM-TIA are 0.82, 0.68, and 0.74 respectively, which can detect attack groups on the Amazon dataset to some extent. Hence, the performance of GAD-MDST is better than that of four baseline methods.

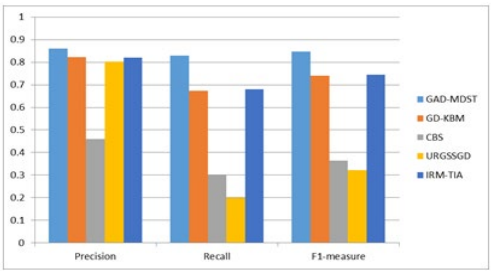
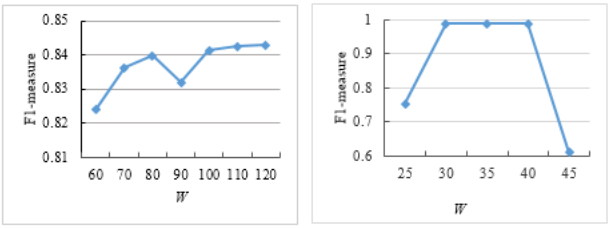


Fig. 3. The influence of W on F1-measure in GAD-MDST on the Amazon and Netflix datasets.

D. Parameter Analysis

a) Selection of W.

Fig. 4 shows the influence of the upper bound W of the maximum density subtensor size output in the M-Zoom model on the F1-measure of GAD-MDST on the Amazon and Netflix datasets.



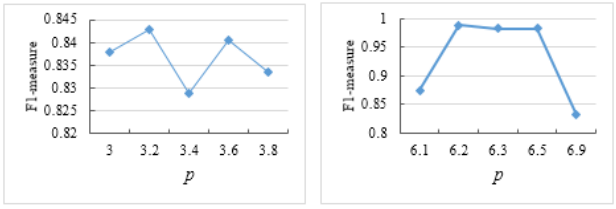
(a)Amazon (b)Netflix

Fig. 4. The influence of W on F1-measure in GAD-MDST on the Amazon and Netflix datasets.

In Fig. 4 (a), when W =120, the F1-measure of GAD-MDST on the Amazon dataset is the best. Therefore, we set W to 120 on the Amazon dataset. In Fig. 4 (b), when W is 30, the F1-measure of GAD-MDST is the best, which is close to 0.99. Thus, we set W to 30 on the Netflix dataset.

b) Selection of P.

Fig. 5 shows the impact of parameter p on the F1-measure of GAD-MDST on the Amazon and Netflix datasets.



(a) Amazon (b) Netflix

Fig. 5. The influence of parameter p on F1-measure of GAD-MDST on the Amazon and Netflix datasets

As shown in Fig. 5 (a), on the Amazon dataset, when p is between 3 and 3.8, the F1-measure fluctuates between 0.825 and 0.845, and when p = 3.2, the F1-measure reaches the best. Therefore, we set p to 3.2 on the Amazon dataset. Similarly, we set p to 6.2 on the Netflix dataset according to Fig. 5 (b).

VII. Conclusion

Detecting group shilling attacks is the key problem to ensure the credibility of recommender systems. We propose a detection method based on maximum density subtensor mining. The main contributions of this paper are below: 1) We propose a method to generate candidate groups, which is based on the tensor data modeling of item time series and the M-Zoom model. 2) We propose a dual-input convolutional neural network model, which fuses individual user rating features and user relevance features. 3) We propose a method to generate multi-scale user features automatically by fusing convolutional neural network and feature pyramid network. The experimental results on the Netflix and Amazon datasets show that the proposed method is more effective than the baseline methods.

Acknowledgment

This work was supported by the National Natural Science Foundation of China (No. 61772452).

References

[1] Q. Zhang, J. Lu, D. Wu, G. Zhang, "A cross-domain recommender system with kernel-induced knowledge transfer for overlapping entities," IEEE trans. Neural Netw. Learn. Syst., vol.30, no.7, pp. 1998-2012, Jul. 2019.

[2] S. K. Lam, J. Riedl. "Shilling recommender systems for fun and profit," Thirteenth Int. World Wide Web Conf. Proc. USA, 2004, pp. 393-402.

[3] X. F. Su, H. J. Zeng, Z. Chen, "Finding group shilling in recommendation system," Special interest tracks and posters of the 14th Int. Conf. World Wide Web, 2005, pp. 960–961.

[4] Z. Yang, L. Xu, Z. Cai, Z. Xu, "Re-scale AdaBoost for attack detection in collaborative filtering recommender systems," Knowl Based Syst, vol.100, pp. 74-88, June 2015.

[5] F. Zhang, H. Chen. "An ensemble method for detecting shilling attacks based on ordered item sequences," Secur. Commun. Netw, vol. 9, no.9, pp. 680-696, May 2016.

[6] Tong C, Yin X, Li J. "A shilling Attack Detector Based on Convolutional Neural Network for Collaborative Recommender System in Social Aware Network," Comput J, 2vol. 61, no.7, pp. 949-958, July 2018.

[7] A. Mukherjee, B. Liu, J. H. Wang, N. Glance, N. Jindal, "Detecting group review spam", Proc. Int. Conf. Companion World Wide Web. USA, 2011: 93-94.

[8] C. Xu, J. Zhang, K. Chang, C. Long, "Uncovering collusive spammers in Chinese review websites," Int Conf Inf Knowledge Manage. USA, 2013, pp. 979-988.

[9] M. Allahbakhsh, A. Ignjatovic, B. Benatallah, et al. "Collusion detection in online rating systems," Lect. Notes Comput. Sci. Australia, 2013, pp. 96-207.

[10] Z. Wang, S. Gu, X. Zhao, X. Xu, "Graph-based review spammer group detection", Knowl. Inf. Syst., vol. 55, pp. 571-597, June 2018.

[11] W. Zhou, Y. S. Koh, J. Wen, S. Alam, G. Dobbie, "Detection of abnormal profiles on group attacks in recommender systems," SIGIR - Proc. Int. ACM SIGIR Conf. Res. Dev. Inf. Retr. Australia, 2014, pp. 955-958.

[12] Q. Wang, Y. Ren, N. He, M. Wan, G. Lu, "A group attack detector for collaborative filtering recommendation," Proc. Int. Computer Conf. Wavelet Active Media Technology and Information Processing. China, 2014, pp. 454-457.

[13] Y. Wang, Z. Wu, Z. Bu, "Discovering shilling groups in a real e-commerce platform," Online Inf. Rev., vol. 40, pp.62-78, Feb 2016.

[14] F. Zhang, S. Wang, "Detecting group shilling attacks in online recommender systems based on bisecting k-means clustering," IEEE Trans. Comput. Soc. Syst., vol. 7, pp. 1189-1199, 2020.

[15] Y. Zhang, Y. Tan, M. Zhang, Y. Liu, C. Tat, S. Ma, "Catch the black sheep: Unified framework for shilling attack detection based on fraudulent action propagation," JCAI Int. Joint Conf. Artif. Intell. Argentina, 2015, pp. 2408-2414.

[16] K. Shin, B. Hooi, C. Faloutsos, "Fast, accurate, and flexible algorithms for dense subtensor mining," ACM Trans. Knowl. Discov. Data, vol. 12, no.3, pp: 1-30, January 2018.

[17] F. Zhang, H. Chen, "An ensemble method for detecting shilling attacks based on ordered item sequences," Secur. Commun. Netw., vol.9, no.7, pp. 680-696, May 2016.

[18] Y. Wang, Z. Wu, J. Cao, Fang. C, "Towards a tricksy group shilling attack model against recommender systems," Lect. Notes Comput. Sci., China, 2012, pp. 675-688.

[19] Z. Han, K. Yang, X. Tan, "Analyzing spectrum features of weight user relation graph to identify large spammer groups in online shopping websites," Jisuanji Xuebao, vol.40, pp. 939-954, April 2017.

[20] H. Cai, F. Zhang "An unsupervised method for detecting shilling attacks in recommender systems by mining item relationship and identifying target items," Comput J, vol.62, no.4, pp. 579-597, April 2019.