IET Information Security

Research Article

IET Journals
The Institution of Engineering and Technology

Unsupervised approach for detecting shilling attacks in collaborative recommender systems based on user rating behaviours

ISSN 1751-8709
Received on 13th April 2018
Revised 8th November 2018
Accepted on 28th November 2018
E-First on 21st January 2019
doi: 10.1049/iet-ifs.2018.5131
www.ietdl.org

Fuzhi Zhang[1,2] ✉, *Zhoujun Ling*[1,2], *Shilei Wang*[1,2]

[1]*School of Information Science and Engineering, Yanshan University, Qinhuangdao, Hebei Province, People's Republic of China*
[2]*The Key Laboratory for Computer Virtual Technology and System Integration of Hebei Province, Yanshan University, Qinhuangdao, Hebei Province, People's Republic of China*
✉ *E-mail: xjzfz@ysu.edu.cn*

Abstract: Collaborative recommender systems have been known to be extremely vulnerable to shilling attacks. To prevent such attacks, many detection approaches including supervised and unsupervised have been proposed. However, the supervised approaches are only suitable for detecting known types of attacks and the unsupervised approaches require a priori knowledge to ensure the detection performance. To address the limitations, the authors propose an unsupervised approach for detecting shilling attacks based on user rating behaviours. They first use Gibbs latent Dirichlet allocation model to extract latent topics of user preferences from user rating item sequences, then they use mixture transition distribution model to construct the user's preference model and present several metrics to capture the diversity between genuine and attack users in rating behaviours. In the case of unknown attack size, the number of attack users is obtained by analysing the critical point of rating behaviour suspicious degrees between genuine and attack users, and based on which the attack users are identified. The experimental results on the MovieLens 1 M dataset show that the proposed approach outperforms the baseline methods in terms of recall and precision metrics.

Nomenclature

U	set of users in the recommender system
I	set of items in the recommender system
M	number of users in the recommender system
N	number of items in the recommender system
K	number of latent topics
U^*	set whose elements are sets of users who rate items in set I, i.e. $U^* = \{U_{i_n} \mid n = 1, 2, \ldots, N\}$
U_{i_n}	set of users who rate item i_n
Z	set whose elements are sets of latent topic numbers corresponding to users who rate items in set I, i.e. $Z = \{Z_{i_n} \mid n = 1, 2, \ldots, N\}$
Z_{i_n}	set of latent topic numbers corresponding to users who rate item i_n
E	set of latent topics
e_k	kth latent topic in set E
$u_{n,m}$	mth user of set U_{i_n} who rates item i_n
$z_{n,m}$	latent topic number for the mth user who rates item i_n
$\neg_{n,m}$	excluding the user $u_{n,m}$
I_e	set of item-latent topics

1 Introduction

Collaborative recommender systems can provide personalised recommendations that suit a user's tastes by collecting the user's preferences, which have been widely used in e-commerce websites to deal with information overload problem. The fundamental assumption behind such systems is that users who had similar preferences with other users in the past are likely to have similar preferences in the future. However, this assumption could make a collaborative recommender system extremely vulnerable to shilling attacks [1] or profile injection attacks [2], which can be illustrated by an example of a shilling attack.

Consider a movie recommender system that uses the user-based collaborative filtering algorithm to generate recommendations, the system creates a profile for each user according to the user's ratings (in the scale of 1–5 and 1 indicates disliked) on the movies. Table 1 shows the profiles of Alice and seven genuine users (user1–7) and the attack profiles (attack1–3) inserted by an attacker, as well as the correlation similarity with Alice. According to the principle of user-based collaborative filtering algorithm, user6 is the most similar user to Alice when there are no attack profiles, and the system will predict Alice's rating on movie6 to be 2. This means Alice does not like movie6 and the system will not recommend movie6 to her. After the attack profiles are injected, however, attack1 becomes Alice's most similar user, and the system would yield a predicted rating of 5 for movie6 and movie6 will be recommended to Alice. This indicates that the recommendation of the collaborative recommender system can be manipulated by attackers.

In shilling attacks, malicious users inject a certain number of fake profiles in an attempt to bias the system's output to their advantage. The fake profiles are generally called attack profiles or shilling profiles. Depending on the purpose of attacks, shilling attacks can be categorised as either push attacks or nuke attacks, which promote or demote a particular item to be recommended [3]. The widely studied shilling attacks include random attack, average attack, bandwagon attack etc. [1, 4]. These attacks present a great challenge to the credibility of collaborative recommender systems.

To reduce the influence of shilling attacks on collaborative recommender systems, a variety of shilling attack detection approaches including supervised and unsupervised has been proposed. Meanwhile, the attackers may change their attack strategies to evade detection [5, 6]. Moreover, new types of attacks will continue to appear [7]. With the evolution of shilling attacks, the performance of existing detection approaches is restricted. On the one hand, the supervised detection approaches require labelling data samples and training classifiers, and they are only suitable for detecting known types of attacks. On the other hand, the unsupervised detection approaches can detect shilling attacks without considering the specific attack types, but they usually require a priori knowledge of attacks such as the number of attack users (i.e. attack size) and the candidate spam users labelled. Such prior knowledge is difficult to acquire in real collaborative recommender systems.

Table 1 Example of a shilling attack for promoting the target item movie6

	Movie1	Movie2	Movie3	Movie4	Movie5	Movie6	Correlation with Alice
Alice	5	4	3	2	—	?	—
user1	5	1	—	3	—	1	0.32
user2	2	4	1	2	—	2	0.31
user3	2	4	3	3	—	2	−0.32
user4	4	2	3	2	3	1	0.67
user5	5	1	3	2	5	—	0.51
user6	4	3	3	2	—	2	0.95
user7	—	5	2	2	3	1	0.74
attack1	5	4	3	2	—	5	1.00
attack2	5	4	3	2	1	5	0.91
attack3	5	3	3	2	—	5	0.92

Table 2 Attack models used in this paper

Attack models	I_S		I_F		I_T
	Items	Rating	Items	Rating	Rating
random attack	not used		randomly chosen	system's mean	r_{max}/r_{min}
average attack	not used		randomly chosen	each item's mean	r_{max}/r_{min}
bandwagon attack	popular items	r_{max}	randomly chosen	system's mean	r_{max}
AoP attack	not used		randomly chosen from top x% of the most popular items	each item's mean	r_{max}/r_{min}
power user attack	not used		randomly chosen from items rated by the power users	each item's mean	r_{max}/r_{min}
average-target shift attack	not used		randomly chosen	each item's mean	$r_{max}-1/r_{min}+1$
average-noise injected attack	not used		randomly chosen	each item's mean with Gaussian random noise	r_{max}/r_{min}
hybrid attacks (random attack)	not used		randomly chosen	system's mean	r_{max}/r_{min}
(average attack)	not used		randomly chosen	each item's mean	r_{max}/r_{min}
(bandwagon attack)	popular items	r_{max}	randomly chosen	system's mean	r_{max}

To address the above limitations, we propose an unsupervised approach for detecting shilling attacks based on user rating behaviours. The proposed approach assumes that the rating distributions of attack users differ from those of genuine ones, and thus making a difference in user preference sequences. On the basis of this hypothesis, we use Gibbs latent Dirichlet allocation (LDA) model to model user's history rating behaviours and use mixture transition distribution (MTD) model to construct the user's preference model in order to analyse the difference between genuine and attack users in rating behaviours. By calculating the sum of rating behaviour suspicious degree differences in each sliding window, the critical point of rating behaviour suspicious degrees between genuine and attack users is recognised, and thus the attack users are obtained. This approach neither does need to know the attack size in advance nor does it needs to label the candidate spam users.

The main contributions of this paper are as follows:

(i) To analyse the user rating behaviours, we use Gibbs LDA model to extract latent topics of user preferences from the user rating item sequences (ISs).
(ii) We use MTD model to construct user's preference model and propose metrics to capture the diversity between genuine and attack users in rating behaviours.
(iii) In the case of unknown attack size, the number of attack users is obtained by analysing the critical point of rating behaviour suspicious degrees between genuine and attack users, and based on which the attack users are identified.
(iv) To evaluate the proposed approach, we conduct experiments on the MovieLens 1 M dataset and compare it with the baseline methods.

The rest of this paper is organised as follows. The related work on shilling attack detection is introduced in Section 2. In Section 3, we propose our approach which includes the detection framework, extraction of latent topics, analysis of diversity in user rating behaviours, and shilling attack detection algorithm. Experimental results are reported in Section 4 and conclusions are drawn in Section 5.

2 Background and related work

2.1 Attack models

An attack model is the approach that attackers generate shilling profiles according to the knowledge of collaborative filtering (CF) recommender system, rating database, users, and items [8]. A shilling profile contains a set of biased ratings assigned by specific strategy including a rating for the target item to be promoted or demoted by the attacker. Depending on push or nuke attacks, the target item will be assigned either the maximum rating value r_{max} or the minimum rating value r_{min}. The general form of a shilling profile includes four kinds of items, i.e. the selected items, the filler items, the unrated items, and the target items, which can be represented by sets I_S, I_F, $I_\emptyset$, and I_t, respectively [4]. The strength of shilling attacks is usually specified by filler size and attack size [9].

The attack models used in this paper are summarised in Table 2, which include random attack, average attack, bandwagon attack, average over popular (AoP) items attack [6], power user attack [7], average-target shift, average-noise injected [5], and hybrid attacks. The hybrid attacks consist of three types of attacks: random attack, average attack, and bandwagon attack. The shilling profiles generated by the hybrid attack model are a mixture of shilling profiles generated by random attack, average attack, and bandwagon attack models.

As shown in Table 2, the set of selected items, I_S, is only used by bandwagon attack. The selected items are chosen from popular items (i.e. frequently rated by users) and their ratings are assigned with the maximum rating value r_{max}. As to the set of filler items,

I_F, the attackers tend to randomly choose the filler items in order to reduce the knowledge cost of attacks. Particularly, the filler items for AoP attack are randomly chosen from top x% of the most popular items, the filler items for power user attack are randomly chosen from items rated by the power users (i.e. users with the highest number of ratings), and the filler items for other attack models are randomly chosen from non-target items. Depending on the attack models, the ratings of items in set I_F are assigned either with normal distribution around the system's mean rating or with normal distribution around each item's mean rating.

2.2 Related work

To detect shilling attacks, a variety of approaches have been proposed over the past decade [5, 6, 8, 10–22]. From the perspective of machine learning, the existing attack detection approaches can be generally categorised as supervised and unsupervised approaches according to whether or not the training samples are needed. Table 3 summarises the approaches for shilling attack detection according to this classification.

Supervised approaches require labelling training samples and use known types of samples to train a classifier to detect the attacks. Therefore, such approaches are only suitable for detecting known types of attacks. Under the hypothesis that the statistical signature of shilling profiles would differ greatly from that of genuine ones, Williams *et al.* [2] and Burke *et al.* [8] proposed several features by analysing the rating patterns of shilling profiles and trained three supervised classifiers for shilling profile classification. These classifiers can detect the standard attacks (i.e. random attack, average attack, and bandwagon attack) with various filler and attack sizes. Wu *et al.* [10] proposed a semi-supervised method to detect hybrid attacks, which exploited both labelled and unlabelled user profiles to classify attack users. This method is effective in detecting hybrid and obfuscated attacks. Li *et al.* [11] proposed a shilling attack detection method based on the improved iterative dichotomiser 3 decision tree algorithm, which extracted user's features based on item popularity distributions. This method is not easily affected by the obfuscated attack strategies. Zhou *et al.* [12] proposed a two-phase attack detection method, which used an SVM-based classifier to obtain a set of suspicious profiles and applied target item analysis method to remove genuine profiles from the set. Yang *et al.* [13] used a variant of boosting algorithm to detect shilling attacks based on 18 detection features, which is effective in detecting various shilling attacks. In [14], an online method for detecting shilling attacks was proposed by combining Hilbert–Huang transform and support vector machine. This method constructed user rating series according to the novelty and popularity of items and applied Hilbert–Huang transform method to extract user features and trained an SVM-based classifier to detect shilling profiles. In [15], an ensemble method for detecting shilling attacks was proposed, which extracted user features from both ordered popular ISs and ordered novelty ISs and exploited C4.5-based ensemble model to classify shilling profiles.

Unsupervised approaches do not require data samples to train classifiers, but they usually require a priori knowledge of attacks, e.g. the number of shilling profiles injected (i.e. attack size) and the candidate spam users to be labelled. Bryan *et al.* [16] proposed an unsupervised method to detect shilling attacks by introducing a mean square residue or H_v-*score* metric. This method performs well in detecting random and average attacks, but it works poorly in detecting bandwagon attack with small filler size. Mehta and Nejdl [17] exploited the high correlation between shilling profiles and presented an unsupervised method to detect shilling attacks based on principal component analysis. This method performs well in detecting the standard attacks, but it requires to know the attack size in advance. Bhaumik *et al.* [18] presented an attribute-based clustering method to detect shilling profiles. This method divided the user profiles into two groups according to several generic detection attributes of shilling profiles and H_v-*score* metric used in [16], the profiles in the smaller group were viewed as shilling ones. Lee and Zhu [19] assumed that the effective shilling profiles should be located at the centre of the distribution of genuine ones in order to influence most of genuine profiles and proposed a hybrid two-

Table 3 Summary of approaches for shilling attack detection

Category	Approaches	Techniques	Advantage	Disadvantage
supervised	Williams *et al.* [2] and Burke *et al.* [8]	supervised learning	effective for specific attack strategies	less effective for obfuscated attacks
	Wu *et al.* [10]	semi-supervised learning	effective for hybrid and obfuscated attacks	overall detection performance is not very high
	Li *et al.* [11]	supervised learning	not easily affected by the obfuscated attack strategies	detection precision is not very high for attacks with small filler and attack sizes
	Zhou *et al.* [12]	supervised learning target item analysis	high detection precision	low recall for attacks with small attack sizes
	Yang *et al.* [13]	ensemble learning	effective for specific attack strategies	much computational effort for classification features
	Zhang and Zhou [14]	supervised learning	individual profile-based feature extraction method	overall detection performance is not very high
	Zhang and Chen [15]	ensemble learning	it does not rely on the rating values of items	detection precision is not too high for attacks with small attack sizes
unsupervised	Bryan *et al.* [16]	variance adjusted mean square residue or H_V-score	it performs well in detecting random and average attacks	poor performance for bandwagon attack with small filler sizes
	Mehta and Nejdl [17]	principal component analysis	regardless of the specific attack strategy	poor performance for AoP attack and requires to know the attack size
	Bhaumik *et al.* [18]	*K*-means clustering	regardless of the specific attack strategy	detection of wrong cluster of users can result in filtering out of genuine users
	Lee and Zhu [19]	multidimensional scaling approach and hierarchical clustering	regardless of the specific attack strategy	less effective for attacks with low filler size
	Zhang and Kulkarni [20, 21]	graph-based method spectral clustering	it does not need to specify the exact number of shilling profiles	it requires the shilling profiles to be highly correlated
	Zhang *et al.* [22]	bipartite graph and label propagation algorithm	regardless of the specific attack strategy	it requires to label the candidate spam users and to know the attack size
	Yang *et al.* [23]	graph mining and target item analysis	it can detect various attacks with different filler sizes and attack sizes	less effective for small attack sizes

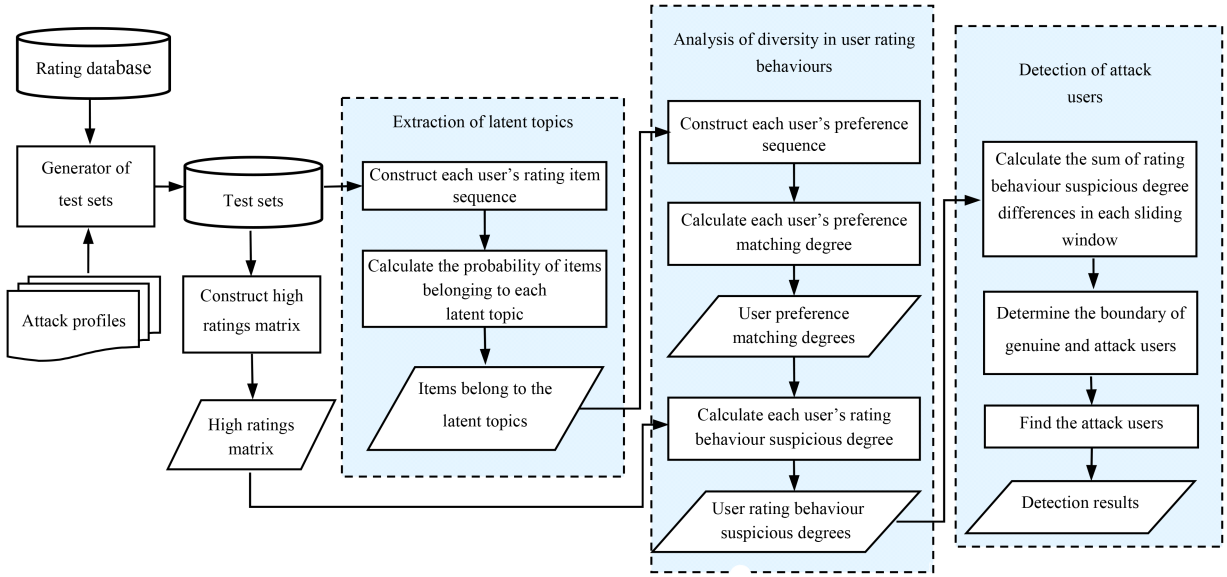

Fig. 1 *Framework of DSA-URB*

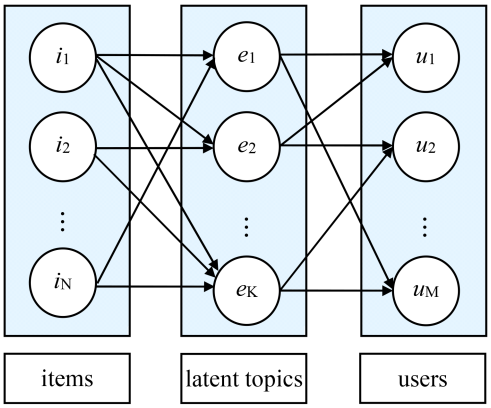

Fig. 2 *Graphical representation of item-latent topic-user*

phase method to detect shilling attacks based on multidimensional scaling and clustering technique. This method can effectively detect average attack with high filler size, but it is less effective in detecting random attack with small filler size. Zhang and Kulkarni [20, 21] assumed that the shilling profiles were highly correlated with each other, thus the problem of shilling attack detection could be formulated as finding a maximal sub-matrix in the similarity matrix and presented a graph-based algorithm and a spectral clustering-based algorithm to detect shilling attacks. Zhang *et al.* [22] started from constructing the user-item bipartite graph and proposed a unified framework for detecting shilling attacks based on the idea of label propagation. This method can detect attacks without considering the specific attack strategy. Nevertheless, it needs to label parts of attackers as the candidate spam users. Yang *et al.* [23] exploited the similarity of topological structure of attack users in graph model to find out the suspicious users and applied target item analysis method to identify attack users from the suspicious users.

3 Proposed approach

To identify the attack users effectively, we propose an unsupervised approach for detecting shilling attacks based on user rating behaviours, which is called detecting shilling attacks based on user rating behaviours (DSA-URB). The framework of DSA-URB is depicted in Fig. 1.

As shown in Fig. 1, the detection of shilling attacks can be divided into three stages: extraction of latent topics, analysis of diversity in user rating behaviours, and detection of attack users. In the first stage, each user's rating IS is constructed and converted into the user's preference sequence by Gibbs LDA model. In the second stage, each user's preference matching degree is calculated based on the user's preference sequence and combined with the users' high ratings matrix to calculate the user rating behaviour suspicious degrees. In the third stage, the boundary between genuine and attack users is determined by calculating the sum of rating behaviour suspicious degree differences in each sliding window and the attack users are detected. The details of DSA-URB will be discussed in the following sections.

To facilitate the discussion, 'Nomenclature' section gives the descriptions of notations used in this paper.

3.1 Extraction of latent topics

In this section, we use LDA model to extract latent topics from user rating ISs and propose an algorithm for extraction of latent topics. A latent topic is a category number that categorises items. LDA topic model is a document generation model, which considers that a document has multiple topics and each topic corresponds to different words. Probabilistic latent semantic analysis (PLSA) [http://en.wikipedia.org/wiki/Probabilistic_latent_semantic_analysis.] is another applicable approach for extraction of latent topics, which is a statistical technique for the analysis of two-mode and co-occurrence data. Compared with PLSA, LDA topic model has better representation ability for original data and the extracted latent topics can better represent item categories. A Gibbs LDA model is the LDA topic model that employs Gibbs sampling technique for parameter estimation and inference.

First of all, we give the definition of rating IS.

Definition 1: (rating IS): The rating IS of user $u_m \in U$ refers to a set of items rated by user u_m in the order of time, denoted by

$$\mathrm{IU}_m = \{i_{n1,\,t1},\, i_{n2,\,t2},\, \ldots,\, i_{ns,\,ts}\}$$

where $i_{n1,\,t1}, i_{n2,\,t2}, \ldots, i_{ns,\,ts}$ represent the items $i_{n1}, i_{n2}, \ldots, i_{ns}$ rated by user u_m at time $t1, t2, \ldots, ts$, respectively.

To extract latent topics from rating ISs using LDA model, we view an item in the collaborative recommender system as a document and the users who rate the item as words in the document. A document contains multiple topics and each topic contains multiple words. They are a many-to-many relationship. Therefore, we can use a three-tier representation mechanism (i.e. item-latent topic-user) to investigate the process of latent topic extraction. Fig. 2 illustrates the graphical representation of item-latent topic-user.

As shown in Fig. 2, each item corresponds to multiple latent topics (i.e. an item can belong to multiple categories) and each latent topic corresponds to multiple users (i.e. multiple users prefer to an item category). The probability of an item belonging to

different topics can be obtained through LDA model. The category with the highest probability is taken as the latent topic of the item.

LDA model can be decomposed into two physical processes:

(i) Generate the latent topic number $z_{n,m}$ by Dirichlet distribution θ_n that represents the item-latent topic.
(ii) In the K Dirichlet distribution $\varphi_{e_k}(k = 1, 2, \ldots, K)$ of the latent topic-user, generate the user $u_{n,m}$ when selecting $e_k = z_{n,m}$.

On the basis of the above two processes, the generative probability of user $u_{n,m}$ can be expressed as follows:

$$p(u_{n,m}|i_n) = \sum_{k=1}^{K} (p(u_{n,m}|z_{n,m} = e_k)p(z_{n,m} = e_k|i_n)) \tag{1}$$

where $z_{n,m}$ is a latent variable indicating that the user $u_{n,m}$ is generated by $\varphi_{e_k = z_{n,m}}$, $p(u_{n,m}|z_{n,m} = e_k)$ denotes the probability of generating user $u_{n,m}$ when $z_{n,m} = e_k$, $p(z_{n,m} = e_k|i_n)$ represents the probability of item i_n belonging to the latent topic e_k.

Thus, the probability of items rated by the users is

$$p(U^*|I) = \prod_{n=1}^{N} \prod_{m=1}^{M} \left(\sum_{k=1}^{K} p(u_{n,m}|z_{n,m} = e_k)p(z_{n,m} = e_k|i_n) \right) \tag{2}$$

For simplicity, we write that $\varphi_{e_k u_{n,m}} = p(u_{n,m}|z_{n,m} = e_k)$ and $\theta_{ne_k} = p(z_{n,m} = e_k|i_n)$. Thus, (2) can be simplified as

$$p(U^*|I) = \prod_{n=1}^{N} \prod_{m=1}^{M} \left(\sum_{k=1}^{K} \varphi_{e_k u_{n,m}} \theta_{ne_k} \right) \tag{3}$$

Since we cannot accurately obtain the latent topics by LDA model, in this paper we use Gibbs LDA model to extract the latent topics. We first give each user a random latent topic, and then we can obtain the stable latent topics of each user by iterations. In the process of iterations, we use Gibbs sampling method to calculate the probability that the current user prefers to each latent topic when the current latent topic distribution of other users is known, and reselect the user's latent topics based on the probability value.

According to the requirements of Gibbs sampling algorithm, we can obtain the probability of user $u_{n,m}$ preferring to each latent topic as follows:

$$p\left(z_{n,m} = e_k \middle| Z_{\neg_{n,m}}, U^*\right) \propto \frac{C_{i_n, \neg_{n,m}}^{e_k} + \alpha}{\sum_{k=1}^{K} C_{i_n, \neg_{n,m}}^{e_k} + K\alpha} \cdot \frac{C_{e_k, \neg_{n,m}}^{u_f} + \beta}{\sum_{f=1}^{M} C_{e_k, \neg_{n,m}}^{u_f} + M\beta} \tag{4}$$

where α and β represent the hyperparameters of Dirichlet distribution θ_n and φ_{e_k}, respectively, $C_{i_n, \neg_{n,m}}^{e_k}$ represents the number of users who rate item i_n and prefer to the latent topic e_k, $\sum_{k=1}^{K} C_{i_n, \neg_{n,m}}^{e_k}$ represents the number of users who rate item i_n, $a_1 < a_2 < \cdots < a_{h-1} < a_h$ represents the number of times that user u_f prefers to the latent topic e_k, $\sum_{f=1}^{M} C_{e_k, \neg_{n,m}}^{u_f}$ represents the number of users who rate items in set I and prefer to the latent topic e_k. The derivation process of (4) is given in the Appendix.

Equation (4) is the basis for LDA to extract the latent topics. Under the condition that a user's rating IS and the current latent topic distribution of other users are known, the user's latent topic distribution can be obtained by Bayesian law.

On the basis of the analysis above, the latent topic extraction algorithm is described as follows.

Algorithm 1 (see Fig. 3) mainly includes three parts. The first part (lines 2–10) initialises each user's latent topics randomly. The second part (lines 11–23) trains the LDA model by (4); this process is repeated until the latent topics of all users no longer change. Particularly, lines 15–17 calculate the probability that user $u_{n,m}$ prefers to each latent topic when the current latent topic distribution of other users is known. Lines 18, 19, and 22 select the latent topic with the maximum probability value as the current user's latent topic and update the corresponding sets at the same time. The last part (Lines 24–30) constructs the set of item-latent topics I_e. The key to this part is to count the number of users who prefer to each latent topic in set $Z_{i_n}(n = 1, 2, \ldots, N)$ and select the latent topic liked by the largest number of users as item i_n's latent topic.

3.2 Analysis of diversity in user rating behaviours

In this section, we transform the user rating ISs into user preference sequences and use MTD model to construct the user preference model. On the basis of that we introduce preference matching degree to analyse the difference between genuine and attack users in rating behaviours.

Definition 2: (user preference sequence): The preference sequence of user $u_m \in U$ refers to the latent topics corresponding to the items rated by user u_m in the order of time, which is denoted by

$$EU_m = \{e_{k1,t1}, e_{k2,t2}, \ldots, e_{ks,ts}\}$$

where $e_{k1,t1}, e_{k2,t2}, \ldots, e_{ks,ts}$ represent the latent topics $e_{k1}, e_{k2}, \ldots, e_{ks}$ corresponding to the items rated by user u_m at time $t1, t2, \ldots, ts$, respectively.

The basic idea of high-order Markov chain model is that the current state q_o depends only on its previous h states and not any other historical states, which satisfies

$$p(q_o|q_{o-1}q_{o-2}\cdots q_2 q_1) = p(q_o|q_{o-1}q_{o-2}\cdots q_{o-h}) \tag{5}$$

To prevent the parameter explosion in the high-order Markov chain model, the MTD model is used, which satisfies

$$p(q_o|q_{o-1}q_{o-2}\cdots q_{o-h}) = \sum_{j=1}^{h} p(q_o|q_{o-j}) \tag{6}$$

In the user preference model, we also assume that the user's current preference state depends only on his previous h preference states and not any other historical preference states. Thus, we can use MTD model to construct the user preference model, which is depicted in Fig. 4.

As shown in Fig. 4, the first three preference states not only affect the user's current preference state $e_{k5,t5}$, but also their influence is not the same. The closer the preference state to $e_{k5,t5}$, the greater the impact on it. Therefore, we can use different weights $\{a_1, a_2, \ldots, a_{h-1}, a_h\}$ to represent the degree of influence on the user's current preference state, where the weights satisfy $a_1 < a_2 < \cdots < a_{h-1} < a_h$ and $\sum_{j=1}^{h} a_j = 1$.

On the basis of the constructed user preference model, we can analyse the difference between genuine and attack users in rating behaviours.

Definition 3: (transition probability matrix): The transition probability matrix is an $\boldsymbol{E}(\theta_{ne_k})$ matrix, which is represented as follows:

$$\boldsymbol{B} = \begin{bmatrix} b_{e_1,e_1} & b_{e_1,e_2} & \cdots & b_{e_1,e_K} \\ b_{e_2,e_1} & b_{e_2,e_2} & \cdots & b_{e_2,e_K} \\ \vdots & \vdots & \ddots & \vdots \\ b_{e_K,e_1} & b_{e_K,e_2} & \cdots & b_{e_K,e_K} \end{bmatrix}$$

where the element $b_{e_x,e_y}(x = 1, 2, \ldots, K; y = 1, 2, \ldots, K)$ represents the probability value transferring from latent topic e_x to latent topic e_y.

Definition 4: (preference matching degree): The preference matching degree of user $u_m \in U$ refers to the probability that user u_m's preference sequence occurs under the transition probability matrix $\boldsymbol{B}$, which is calculated as follows:

$$\text{PreMat}D_m = \sqrt[|IU_m|-h]{\prod_{l=h+1}^{|IU_m|}\left(\sum_{j=1}^{h} a_j b_{e_{k(l+j-h-1)},e_{kl}}\right)} \tag{7}$$

where $|IU_m|$ represents the length of user u_m's rating IS (i.e. the number of items rated by user u_m), h represents the order of MTD model, which is the same as the number of history preference states that affect user u_m's current preference state.

Equation (7) is the basis for calculating preference matching degrees. According to the high-order Markov model, we can calculate the total transfer degree of user rating IS, which is used to capture the diversity of genuine and attack users in rating behaviours.

Fig. 5 depicts the preference matching degrees of 800 users including 400 genuine users and 400 attack ones. The genuine users are chosen randomly from the MovieLens 1 M dataset and the attack profiles are generated by the attack models described in Table 2. The attack size is 3% and the filler size is 5%. We select 50 attack profiles from each type of attack profiles as the attack users.

Input: set I, set U^*, and set E
Output: set of item latent topics I_e

$I_e \leftarrow \phi$, $Z \leftarrow \phi$
for $\forall i_n \in I$ **do**
  $Z_{i_n} \leftarrow \phi$
  $U_{i_n} \leftarrow$ Get the set of users who rate item i_n from set U^*
  for $\forall u_{n,m} \in U_{i_n}$ **do**
    $z_{n,m} \leftarrow$ Select latent topic from set E, randomly
    $Z_{i_n} \leftarrow Z_{i_n} \cup \{z_{n,m}\}$
  end for
  $Z \leftarrow Z \cup \{Z_{i_n}\}$
end for
for j=1 to *loops* **do** /* *loops* is the number of iterations */
  for $\forall i_n \in I$ **do**
    $Z_{i_n} \leftarrow$ Get the set of latent topics corresponding to users who rate item i_n from set Z
    for $\forall z_{n,m} \in Z_{i_n}$ **do**
      for $\forall e_k \in E$ **do**
        Compute probability $p\left(z_{n,m}=e_k \middle| Z_{\neg_{n,m}}, U^*\right)$ by Eq. (4)
      end for
      $z_{n,m} \leftarrow$ Select latent topic e_k with the maximum probability value
      Update the corresponding latent topic number in set Z_{i_n} with $z_{n,m}$
    end for
  end for
  Update the corresponding element in set Z with Z_{i_n}
end for
for $\forall Z_{i_n} \in Z$ **do**
  for $\forall e_k \in E$ **do**
    Count the number of users who prefer to latent topic e_k in set Z_{i_n}
  end for
  $e_{kn,i_n} \leftarrow$ Select the latent topic liked by the largest number of users as item i_n's latent topic
  $I_e \leftarrow I_e \cup \{e_{kn,i_n}\}$
end for
return I_e

Fig. 3 *Algorithm 1: extraction of latent topics*

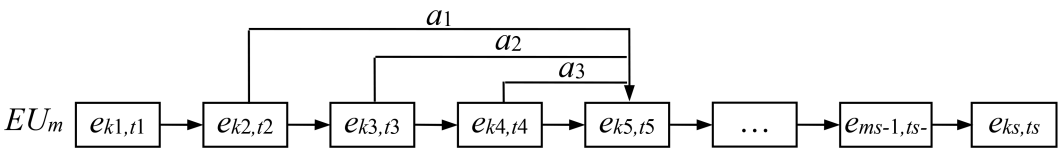

Fig. 4 *User preference model*

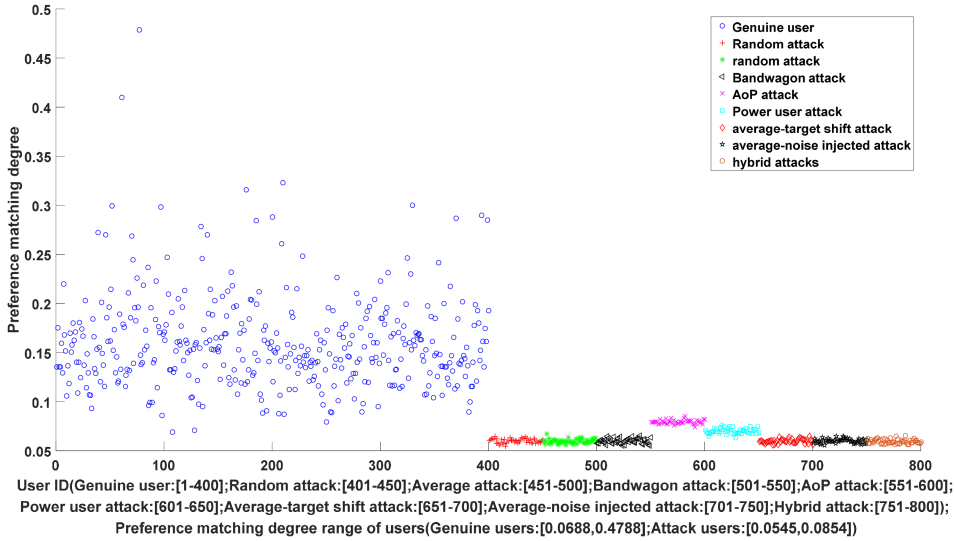

Fig. 5 *Preference matching degrees of genuine and attack users*

Input: user rating dataset D, set U, and set E
Output: list of preference matching degrees *PreMatD*

1: $I_e \leftarrow$ Call Algorithm 1
2: **for** $\forall u_m \in U$ **do**
3: $IU_m \leftarrow$ Construct user u_m's rating item sequence by D
4: $EU_m \leftarrow$ Construct u_m's preference sequence by IU_m and I_e
5: **end for**
6: **for** $j = 1$ to *loop* **do** /* *loop* is the number of iterations */
7: **if** $j = 1$ **then**
8: $U_s \leftarrow$ Select all users in set U as the samples
9: **else**
10: $U_s \leftarrow$ Select the first c% users from U_{sort} as the samples
11: **end if**
12: **for** $\forall e_x \in E$ **do**
13: $sume_x \leftarrow 0$
14: **for** $\forall u_m \in U_s$ **do**
15: $s_1 \leftarrow$ Count the number of e_x transferring to latent topics in EU_m
16: $sume_x \leftarrow sume_x + s_1$
17: **end for**
18: **for** $\forall e_y \in E$ **do**
19: $sume_{xy} \leftarrow 0$
20: **for** $\forall u_m \in U_s$ **do**
21: $s_2 \leftarrow$ Count the number of e_x transferring to e_y in EU_m
22: $sume_{xy} \leftarrow sume_{xy} + s_2$
23: **end for**
24: $b_{e_x,e_y} \leftarrow sume_{xy} / sume_x$
25: **end for**
26: **end for**
27: **for** $\forall u_m \in U$ **do**
28: Compute user u_m's preference matching degree, $PreMatD_m$, by Eq. (7) and add it to list *PreMatD*
29: **end for**
30: $U_{sort} \leftarrow$ Sort users in set U in descending order according to the preference matching degree
31: **end for**
32: **return** *PreMatD*

Fig. 6 *Algorithm 2: analysis of diversity in user rating behaviours*

As shown in Fig. 5, the preference matching degrees of genuine users are different from those of attack ones. Moreover, almost all of them are greater than those of attack users. The greater a user's preference matching degree is, the more likely the user is to be a genuine one.

The main steps for analysing the diversity of user rating behaviours are as follows:

Step 1: Count the transfer frequency of preference states in user preference sequences and obtain the transition probability matrix.
Step 2: Construct the user preference model through MTD model, calculate each user's preference matching degree, and sort all users in descending order according to preference matching degree.
Step 3: To enlarge the difference between genuine and attack users in preference matching degree, we use iterative method to recalculate the transition probability matrix of latent topics. In each iteration, we select a part of users who have the larger preference matching degree to calculate the next topic transition probability.

On the basis of the steps above, the algorithm for analysing the diversity of user rating behaviours is described as follows.

Algorithm 2 (see Fig. 6) mainly includes three parts. The first part (lines 1–5) constructs the preference sequence of each user in set U according to the set of item-latent topics I_e and the constructed rating ISs, where I_e can be obtained by Algorithm 1 (Fig. 3). The second part (lines 7–26) selects the user samples and calculates the transition probability matrix. Particularly, lines 7–11 select the user samples from set U or U_{sort}. Lines 12–26 calculate the transition probability matrix $\boldsymbol{B}$ by Definition 3 based on the user samples. The last part (lines 27–30) calculates each user's preference matching degree by (7) and obtains an ordered set of users (i.e. U_{sort}) by sorting all users in set U in descending order according to the preference matching degree.

3.3 Detection of attack users

In this section, we detect attack users based on rating behaviour suspicious degrees which are calculated according to the user preference matching degrees and user rating matrix. The underlying premise of our approach is that a user with smaller preference matching degree gives high or low ratings for the more suspicious items, the more likely the user is to be an attacker. On the other hand, a target item should be rated by a certain number of attack users; otherwise, the desired attack effect cannot be produced. Therefore, the same target item must have a certain number of ratings provided by the attackers who give it extremely high ratings for push attacks or give it extremely low ratings for nuke attacks.

Definition 5: (preference deviation degree): The preference deviation degree of user $u_m \in U$, $\text{PreDev}D_m$, is defined as follows:

$$\text{PreDev}D_m = \frac{1/\text{PreMat}D_m - 1/\text{PreMat}D_{\max}}{1/\text{PreMat}D_{\min} - 1/\text{PreMat}D_{\max}} \tag{8}$$

where $\text{PreMat}D_{\max}$ and $\text{PreMat}D_{\min}$ represent the maximum and minimum values of user u_m's preference matching degree,

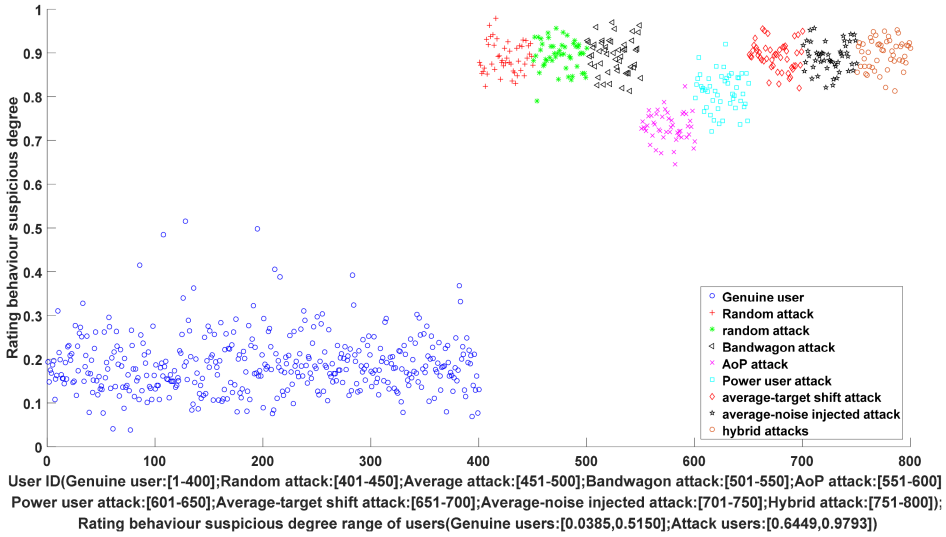

Fig. 7 *Rating behaviour suspicious degrees of genuine and attack users*

respectively. The greater the user u_m's preference deviation degree is, the more suspicious the user u_m is.

Definition 6: (item suspicious cumulative score): The item i_n's suspicious cumulative score, ItSusCumS_n, is defined as follows:

$$\text{ItSusCum}S_n = \frac{\sum_{u_m \in U_\text{H}} (\text{PreDev}D_m \times r_{m,n})}{|U_\text{H}|} \tag{9}$$

where $r_{m,n}$ denotes the user u_m's rating on item i_n, U_H denotes the set of users who give item i_n high ratings (e.g. for the MovieLens 1 M dataset, the high rating refers to the rating value is >3), PreDevD_m denotes the user u_m's preference deviation degree.

Definition 7: (rating behaviour suspicious degree): Let I_H denote the set of items given high ratings by user $u_m \in U$, the user u_m's rating behaviour suspicious degree, RBSusD_m, is defined as the linear weighted combination of user u_m's preference deviation degree and the maximum value of the standardised suspicious cumulative score of items in set I_H, that is

$$\begin{aligned}\text{RBSus}D_m &= \varphi \text{PreDEV}D_m \\ &+ \gamma \max_{i_n \in I_\text{H}} \frac{\text{ItSusCum}S_n - \text{ItSusCum}S_{\min}}{\text{ItSusCum}S_{\max} - \text{ItSusCum}S_{\min}}\end{aligned} \tag{10}$$

where φ and γ are weight factors, PreDevD_m denotes the user u_m's preference deviation degree, ItSusCumS_n denotes the item i_n's suspicious degree, ItSusCum$S_{\max}$ and ItSusCum$S_{\min}$ denote the maximum and minimum values of item suspicious cumulative score, respectively.

Equation (10) consists of two parts: the user preference deviation degree and the maximum value of the standardised suspicious cumulative score of items in set I_H. Since the attackers have a co-rated target item, the item suspicious cumulative score is enlarged by using the label propagation algorithm, thus increasing the rating behaviour suspicious degrees of attackers.

Fig. 7 illustrates the rating behaviour suspicious degrees of 800 users including 400 genuine users and 400 attack ones. The method for selecting genuine users and generating attack users is the same as that used in Section 3.2.

As shown in Fig. 7, the rating behaviour suspicious degree of attack users is greater than that of genuine ones. The greater a user's rating behaviour suspicious degree is, the more likely the user is to be an attacker.

Definition 8: (sequence of rating behaviour suspicious degree differences): Let RBSusD_sort be a list of rating behaviour suspicious degrees sorted in ascending order, the sequence of rating behaviour suspicious degree differences is a sequence whose elements are the differences between two adjacent rating behaviour suspicious degrees in RBSusD_sort, which is represented as

$$S\_\text{RBSDD} = \{\text{Rbsdd}_1, \text{Rbsdd}_2, \ldots, \text{Rbsdd}_{M-1}\}$$

where Rbsdd denotes rating behaviour suspicious degree difference and $\text{Rbsdd}_j = \text{RBSus}D_\text{sort}[j+1] - \text{RBSus}D_\text{sort}[j], j = 1, 2, \ldots, M-1$.

Definition 9: (sum of rating behaviour suspicious degree differences in the sliding window): Let W_s denote the size of sliding window, the sequence of rating behaviour suspicious degree differences $S_$RBSDD can be divided into $M-W_\text{s}$ overlapped windows by sliding one element at a time. The sum of rating behaviour suspicious degree differences in the sliding window w is calculated as follows:

$$\text{Srbsdd}_w = \sum_{j=1}^{W_\text{s}} \text{Rbsdd}_{w+j-1} \tag{11}$$

where $w = 1, 2, \ldots, M-W_\text{s}$ and W_s is set to 10 in this paper.

Definition 10: (sequence of the sum of rating behaviour suspicious degree differences): Let W_s denote the size of sliding window, $M-W_\text{s}$ is the number of overlap windows into which the sequence $S_$RBSDD is partitioned. The sequence of the sum of rating behaviour suspicious degree differences refers to the sequence whose elements are sums of the differences of rating behaviour suspicious degrees in each sliding window, which is represented as

$$S\_\text{SRBSDD} = \{\text{Srbsdd}_1, \text{Srbsdd}_2, \ldots, \text{Srbsdd}_{M-W_\text{s}}\}$$

where Srbsdd_w ($w = 1, 2, \ldots, M-W_s$) denotes the sum of rating behaviour suspicious degree differences in the sliding window w.

Fig. 8 depicts the curve of rating behaviour suspicious degrees sorted in ascending order, which includes 6040 genuine users and 181 attack users. The genuine users are chosen from the MovieLens 1 M dataset and the attack users are generated by the random attack model with 3% filler size and 3% attack size.

As shown in Fig. 8, the change of rating behaviour suspicious degrees is relatively stable before the 6000th user. After that, the rating behaviour suspicious degrees change dramatically. This indicates that there is a great difference between genuine and attack users in rating behaviours. Therefore, the change of rating

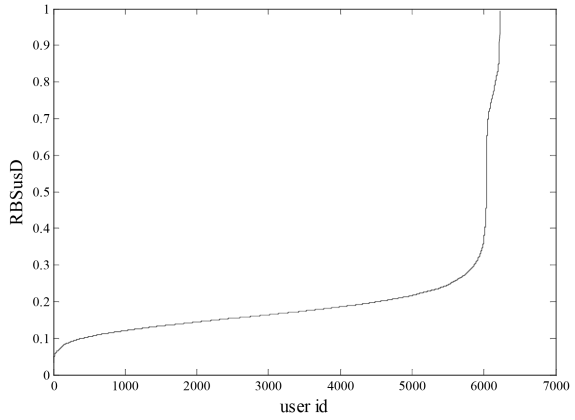

Fig. 8 *Curve of rating behaviour suspicious degrees sorted in ascending order*

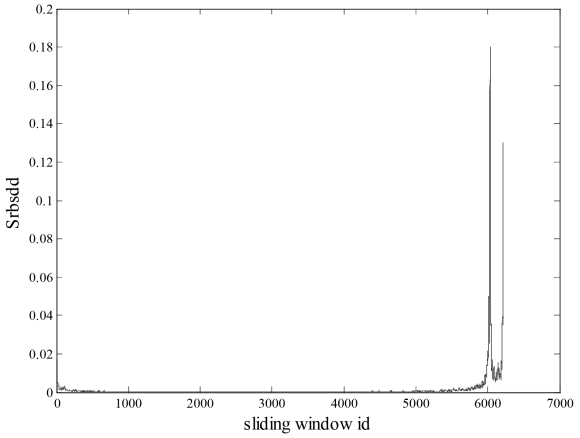

Fig. 9 *Change of the sum of rating behaviour suspicious degree differences in the sliding windows*

behaviour suspicious degrees is significant at the boundary of genuine and attack users. To highlight the boundary of genuine and attack users, we calculate the sum of rating behaviour suspicious degree differences in each sliding window, as depicted in Fig. 9.

In Fig. 9, as the sliding window id increases, the sum of rating behaviour suspicious degree differences of sliding window tends to be stable after a slight decrease. This is because the rating behaviour suspicious degree differences between a small number of genuine users are larger while those between a large number of genuine users are smaller. After that, the sum of rating behaviour suspicious degree differences in the sliding window increases gradually and then becomes smaller after reaching the peak. The reason is that the differences of rating behaviour suspicious degrees between genuine and attack users are larger especially at the boundary of them. As the sliding window id continues to increase, the sum of rating behaviour suspicious degree differences in the sliding window has a sharp increase after a slight change. The reason for this phenomenon is that the rating behaviour suspicious degree differences between some attack users are smaller while those between others are larger.

As shown in Fig. 9, the change of the sum of rating behaviour suspicious degree differences is significant at the boundary between genuine and attack users in the sliding windows, which is the first extreme point. Moreover, the sum of rating behaviour suspicious degree differences in the later sliding window is also larger, which is the second extreme point. To avoid selecting the sliding window of attack users as the boundary, the following two conditions should be satisfied:

(i) The sum of rating behaviour suspicious degree differences in the sliding window at the boundary should be larger.
(ii) The number of sliding windows after the boundary is proportional to the attack size, so the sliding window at the boundary has a larger attack size.

Input: user-item rating matrix R, set U, and set I
Output: set of attack users $AttUs$

1: $S\_SRBSDD \leftarrow \phi$
2: $PreMatD \leftarrow$ Call Algorithm 2
3: **for** $\forall\, u_m \in U$ **do**
4: $PreDevD_m \leftarrow$ Compute the preference deviation degree of user u_m by Eq. (8)
5: **end for**
6: **for** $\forall\, i_n \in I$ **do**
7: $U_H \leftarrow$ Get the users who give item i_n high ratings from R
8: $ItSusCumD_n \leftarrow$ Compute the item i_n's suspicious cumulative score by Eq. (9)
9: **end for**
10: **for** $\forall\, u_m \in U$ **do**
11: $I_H \leftarrow$ Get the items given high ratings by user u_m from R
12: $RBSusD_m \leftarrow$ Compute the user u_m's rating behaviour suspicious degree by Eq. (10)
13: **end for**
14: $S\_RBSDD \leftarrow$ Construct the sequence of rating behaviour suspicious degree differences by Definition 8
15: **for** w=1 to M-W_s **do**
16: $Srbsdd_w \leftarrow$ Compute the sum of rating behaviour suspicious degree differences in window w by Eq. (11)
17: $S\_SRBSDD \leftarrow S\_SRBSDD \cup \{ Srbsdd_w \}$
18: **end for**
19: $w_b \leftarrow getBoundaryWindow(S\_SRBSDD)$
20: $Boundary \leftarrow$ Get the order in S_RBSDD whose element corresponds to the largest difference of rating behaviour suspicious degrees in sliding window w_b
21: $NumofAttUs \leftarrow M$-$Boundary$
22: $U_{sorted} \leftarrow$ Sort all users in set U in descending order according to the rating behaviour suspicious degree
23: $AttUs \leftarrow$ Select the first $NumofAttUs$ users from U_{sorted}
24: **return** $AttUs$

Fig. 10 *Algorithm 3: detecting the attack users*

According to the above conditions, we can determine the sliding window in which the first extreme point appears, and then select the order in $S_$RBSDD whose element corresponds to the largest difference of rating behaviour suspicious degrees in this sliding window as the boundary of genuine and attack users.

On the basis of the above analysis, the algorithm for detecting attack users is described below.

Algorithm 3 (see Fig. 10) mainly includes three parts. The first part (lines 2–13) calculates each user's preference deviation degree (lines 2–5), each item's suspicious cumulative score (lines 6–9), and each user's rating behaviour suspicious degree (lines 10–13). The second part (lines 14–21) determines the number of attack users. Particularly, the sequence of rating behaviour suspicious degree differences is first constructed according to the rating behaviour suspicious degrees (line 14), then the sequence of the sum of rating behaviour suspicious degree differences is constructed (lines 15–18). On the basis of which, the sliding window w_b is determined by function getBoundaryWindow ($S_$SRBSDD) (line 19) and the boundary between genuine and attack users is obtained (line 20). Moreover, finally the number of attack users is calculated (line 21). The third part (lines 22–23) is to obtain the set of attack users.

4 Experimental evaluation

4.1 Experimental dataset and setting

We use the MovieLens 1 M dataset [http://grouplens.org/datasets/movielens/1m/.] as experimental data. This dataset consists of 1,000,209 ratings on 3952 movies by 6040 users, which contains the user's number, the movie's number, the rating for the movie,

and the rating timestamp. All the ratings are integers between 1 and 5, where 1 and 5 indicate disliked and most liked, respectively.

In the experiment, we assume all the profiles in the MovieLens 1 M dataset are genuine ones. The shilling profiles are generated by the attack models described in Table 2 and injected into the MovieLens 1 M dataset. All the shilling profiles are used for promoting a target item (i.e. push attacks). The filler size is set to 3 and 5%, the attack size is set to 3, 5, 7, and 10%. The target item is randomly selected from unpopular items (i.e. the items rated by a few users) for push attacks. Nevertheless, if we change the high ratings matrix in Fig. 1 to the low ratings matrix (i.e. each rating value is <3), our approach can also be used for detecting nuke attacks (i.e. to demote a popular item). The rating timestamps of attack users for the items are randomly selected from those of genuine users for the same items. In the experiments, the average values of ten experiments are reported as the final evaluation results for each filler size and attack size.

4.2 Evaluation metrics

We use precision, recall, and F1-measure metrics to evaluate the performance of the proposed approach, which are defined as follows:

$$\text{precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{12}$$

$$\text{recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{13}$$

$$\text{F1} - \text{measure} = \frac{2 \times \text{precision} \times \text{recall}}{\text{precision} + \text{recall}} \tag{14}$$

where TP denotes the number of attack users identified correctly, FN denotes the number of attack users misidentified as genuine ones, and FP denotes the number of genuine users misidentified as attack ones.

4.3 Experimental results and analysis

To illustrate the effectiveness of the proposed approach, we compare DSA-URB with the following three baseline methods:

(i) Principal component analysis (PCA)-VarSelect [17]: A typical unsupervised method for shilling attack detection, which performs well in detecting standard attacks when the attack size is known. In the experiments, we allow it to know the attack size in advance.
(ii) Catch the black sheep (CBS) [22]: An unsupervised method for shilling attack detection, which requires a priori knowledge for the candidate spam users. In our experiments, we allow it to know the attack size in advance and 10% attack users of each attack size are labelled as the candidate spam users.
(iii) Estimating user behaviour toward detecting anomalous ratings (EUB-DAR) [23]: An unsupervised method for shilling attack detection, which detects the attack users by using the similarity of topological structure of attack users in the graph. This method requires setting several thresholds, and we allow it to satisfy the requirements of setting these thresholds.

4.3.1 Selection of parameters: In this section, we select the parameters used in our approach. The parameters to be set include the hyperparameters α and β, the number K of latent topics in Gibbs LDA model, the order h of MTD model, the weights $\{a_1, a_2, \ldots, a_{h-1}, a_h\}$ of preference matching degree, the ratio of user samples c%, the weight factors φ and γ in (10).

According to the criterion of parameters setting of Gibbs LDA model, the hyperparameters α and β are set to 0.5 and 0.1, respectively. The parameters K and h can be selected by experiment. To show the influence of parameters K and h on DSA-URB, we inject the shilling profiles generated by the previous eight attacks with 3% attack size and 3% filler size into the MovieLens 1 M dataset. Fig. 11 illustrates the influence of parameters K and h on the F1-measure of DSA-URB under eight attacks.

As shown in Fig. 11, DSA-URB can obtain better performance under eight attacks in terms of F1-measure metric when $K = 10$ and $h = 3$. Therefore, we set parameters K and h to 10 and 3, respectively, in the experiments.

The weights $\{a_1, a_2, \ldots, a_{h-1}, a_h\}$ of preference matching degree represent the influence of a preference state on the current preference state, which have little impact on the experimental results. In our experiments, the weights of preference matching degree are set to $\{0.20, 0.35, 0.45\}$. In general, the number of attack users is less than that of genuine users and the transition probability matrix does not require a large number of users as samples, so we set c% to 20% when updating the transition probability matrix $\boldsymbol{B}$.

The weight factors φ and γ are selected by experiment. Fig. 12 illustrates the influence of weight factors φ and γ on the sum of rating behaviour suspicious degree differences in the sliding windows. As shown in Fig. 12, the boundary between genuine and attack users is easy to distinguish when $\varphi = 0.1$ and $\gamma = 0.9$ and difficult to distinguish when $\varphi = 0.9$ and $\gamma = 0.1$. Therefore, we set the weight factors φ and γ to 0.1 and 0.9, respectively, in the experiments.

4.3.2 Results and discussion: To evaluate the effectiveness of DSA-URB, we conduct experiments on the MovieLens 1 M dataset with eight attacks at various filler sizes across various attack sizes and compare it with three baseline methods. Figs. 13 and 14 show the comparison of precision and recall for four methods under eight attacks, respectively.

As shown in Fig. 13, the precision of PCA-VarSelect under eight attacks is between 0 and 0.9495. PCA-VarSelect detects the attack users by calculating the principal components of user rating matrix, which is effective in detecting random attack, average attack, bandwagon attack, average-target shift attack, average-noise injected attack, and hybrid attacks. However, the precision of PCA-VarSelect is poor when detecting AoP attack and power user attack. This is because the AoP attack profiles and power user attack profiles are very similar with the genuine profiles, so that a number of genuine profiles are misclassified as attack ones by PCA-VarSelect, thus resulting in a significant decline in precision especially for AoP attack. The precision of EUB-DAR under eight attacks is between 0.3465 and 0.7471, which indicates that a certain number of genuine profiles are misclassified as attack ones by EUB-DAR. Also, the precision of EUB-DAR is poor under average-target shift attack. The reason is that EUB-DAR identifies the target item according to the number of users who give the highest rating for it in the filtering phase. Moreover, the precision of EUB-DAR tends to increase with the increase of attack size, which means EUB-DAR is suitable for detecting attacks with large attack sizes. The precision of CBS under eight attacks is between 0.8521 and 0.9645, which is much better than that of EUB-DAR. CBS is based on the idea of label propagation and calculates the user suspicious degrees iteratively by passing the suspicious degrees of the known attack users, thus its detection precision is not affected by the types of attacks. For example, the precision of CBS is very high under AoP attack and power user attack. This is because all the attack users rate the popular items under two attacks, so that the suspicious degrees of popular items are greater, thus increasing the suspicious degrees of other attack users. The precision of DSA-URB under eight attacks is between 0.9628 and 1, which is better than that of CBS. The precision of DSA-URB is above 98% when detecting random attack, average attack, and bandwagon attack. For the AoP attack and power user attack, the precision of DSA-URB is also above 96%. Although the precision of DSA-URB slightly decreases in detecting average-target shift attack, it is still above 96%. As to average-noise injected attack and hybrid attacks, the precision of DSA-URB is all above 99%, which is much better than that of PCA-VarSelect, CBS, and EUB-DAR. These results show the superiority of DSA-URB in detecting eight attacks. Also, the precision of DSA-URB has little change when detecting eight attacks with various attack sizes, which means the attack size has little influence on the precision of DSA-URB. Therefore, we can conclude that DSA-URB outperforms PCA-

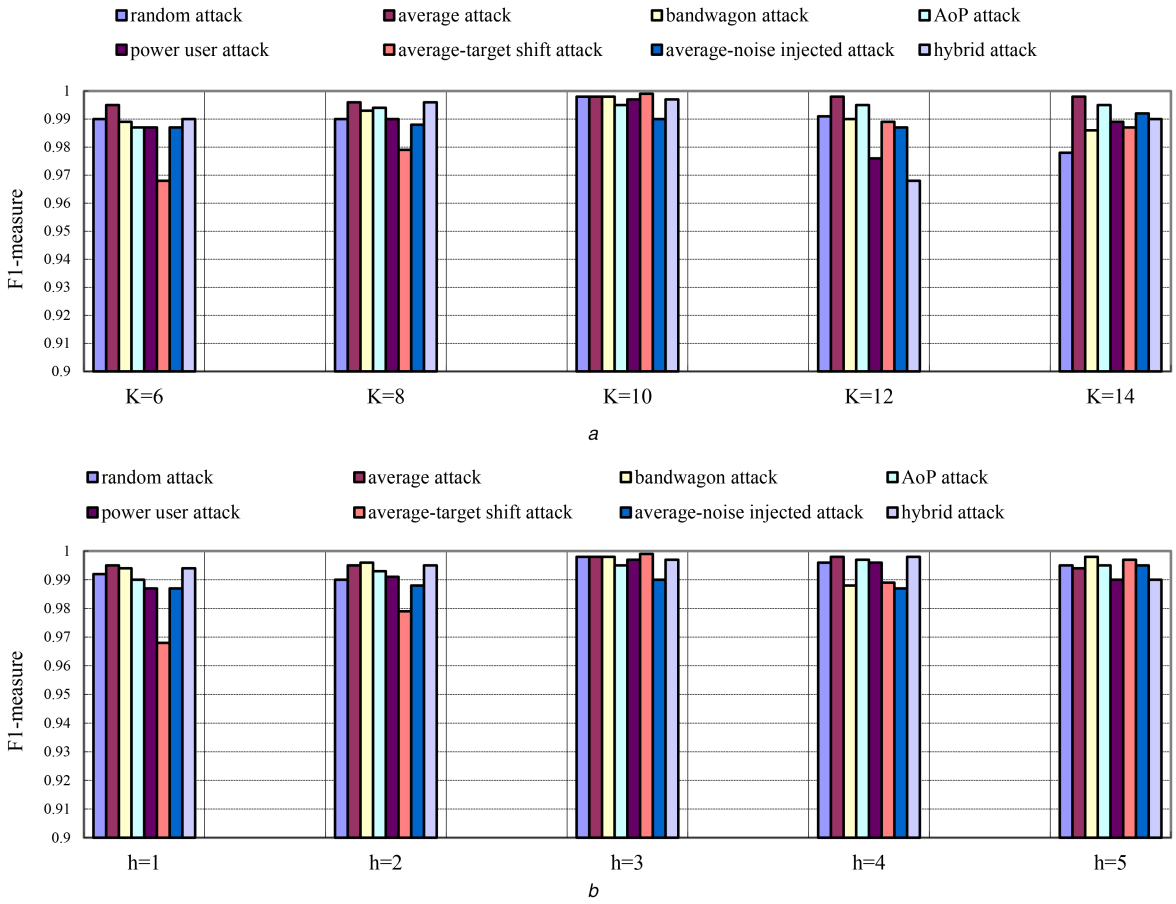

Fig. 11 *Influence of parameters K and h on F1-measure of DSA-URB*
(a) Parameter *K*, ***(b)*** Parameter *h*

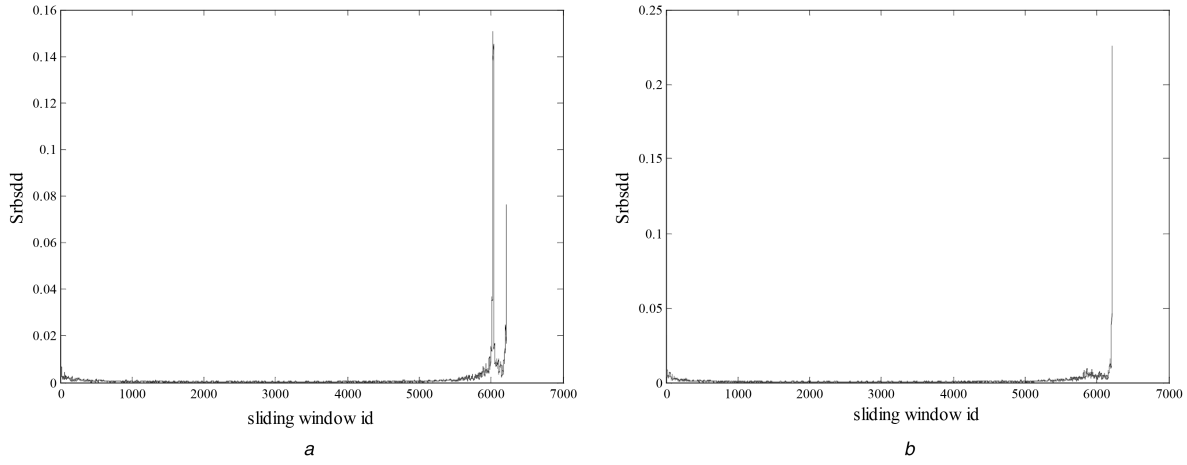

Fig. 12 *Influence of φ and γ on the sum of rating behaviour suspicious degree differences in the sliding windows*
(a) $\varphi = 0.1$ and $\gamma = 0.9$, ***(b)*** $\varphi = 0.9$ and $\gamma = 0.1$

VarSelect, CBS, and EUB-DAR in terms of precision metric when detecting eight attacks.

As shown in Fig. 14, the recall of PCA-VarSelect is above 90% under eight attacks, except for AoP attack and power user attack, which means most of the attack users can be spotted by PCA-VarSelect. As to AoP attack, the recall of PCA-VarSelect is below 50%. This indicates that a certain number of AoP attack profiles are misclassified as genuine ones by PCA-VarSelect. The reason is that AoP attack model selects a part of popular items as the filler items so that the similarity between AoP attack profiles and genuine ones is very high. CBS maintains very high recall in detecting eight attacks, which means most of the attack users can be detected by CBS and only a few attack users are misclassified as genuine ones. The recall of EUB-DAR is not as good as that of CBS in detecting eight attacks because a part of attack users are misclassified as genuine ones by EUB-DAR. As to DSA-URB, most of its recall values are 1 in detecting eight attacks. These results indicate that almost all of the attack users can be identified by DSA-URB and few attack users are misidentified as genuine ones in detecting eight attacks. Therefore, the recall of DSA-URB is better than that of PCA-VarSelect, CBS, and EUB-DAR in detecting eight attacks.

4.3.3 Comparison of performance with and without latent topics: To further show the superiority of DSA-URB in detecting various attacks, we make a comparison for the detection methods with and without latent topics in terms of F1-measure metric. The detection method without latent topics is denoted as without LDA, which builds the user preference sequences directly from the item (movie) category provided by the MovieLens 1 M dataset instead

of extracting the item-latent topics. Table 4 shows the comparison of F1-measure for DSA-URB and without LDA under eight attacks.

As shown in Table 4, the F1-measure of without LDA is very poor in detecting eight attacks, which means the user preference sequences constructed directly from the item category information cannot reflect the difference between genuine and attack users in rating behaviours. By contrast, the F1-measure of DSA-URB is very good and all the F1-measure values are above 0.97, which illustrates the superiority of DSA-URB with latent topic analysis.

5 Conclusion

Shilling attacks present a great challenge to the security of CF recommender systems. With the evolution of shilling attacks, the performance of existing detection approaches is restricted. This paper starts from analysis of diversity in user rating behaviours and develops an unsupervised approach for detecting shilling attacks from the perspective of rating behaviours. We use Gibbs LDA model to extract the latent topics of user preferences from user rating ISs. On the basis of that, we use MTD model to construct the user preference model and propose several metrics to capture the diversity between genuine and attack users in rating behaviours. The attack size is obtained by calculating the sum of differences of rating behaviour suspicious degrees in each sliding window and analysing the critical point of rating behaviour suspicious degrees between genuine and attack users, thus the attack users are detected. The experimental results on the MovieLens 1 M dataset indicate that DSA-URB outperforms the baseline methods in terms of precision and recall metrics when detecting various attacks.

While DSA-URB has an advantage over the baseline methods in detecting attacks, it still has limitations. One limitation for DSA-URB is that some parameters are set by experiment. However, how to set these parameters on real-world datasets without ground truth is a challenging problem. In our future work, we will explore the effective ways to choose these parameters for real application. In addition, DSA-URB calculates the attack size by identifying the critical point of rating behaviour suspicious degrees between genuine and attack users, and based on which the attack users are detected. If attacks are distributed in such a way that the critical point may be hardly identified, DSA-URB may fail in detecting such attacks. In our next work, we will explore the feasibility of such attacks and propose an effective method to detect the attacks.

6 Acknowledgments

This work was supported by the National Natural Science Foundation of China (Nos. 61772452 and 61379116) and the Natural Science Foundation of Hebei Province, China (No. F2015203046).

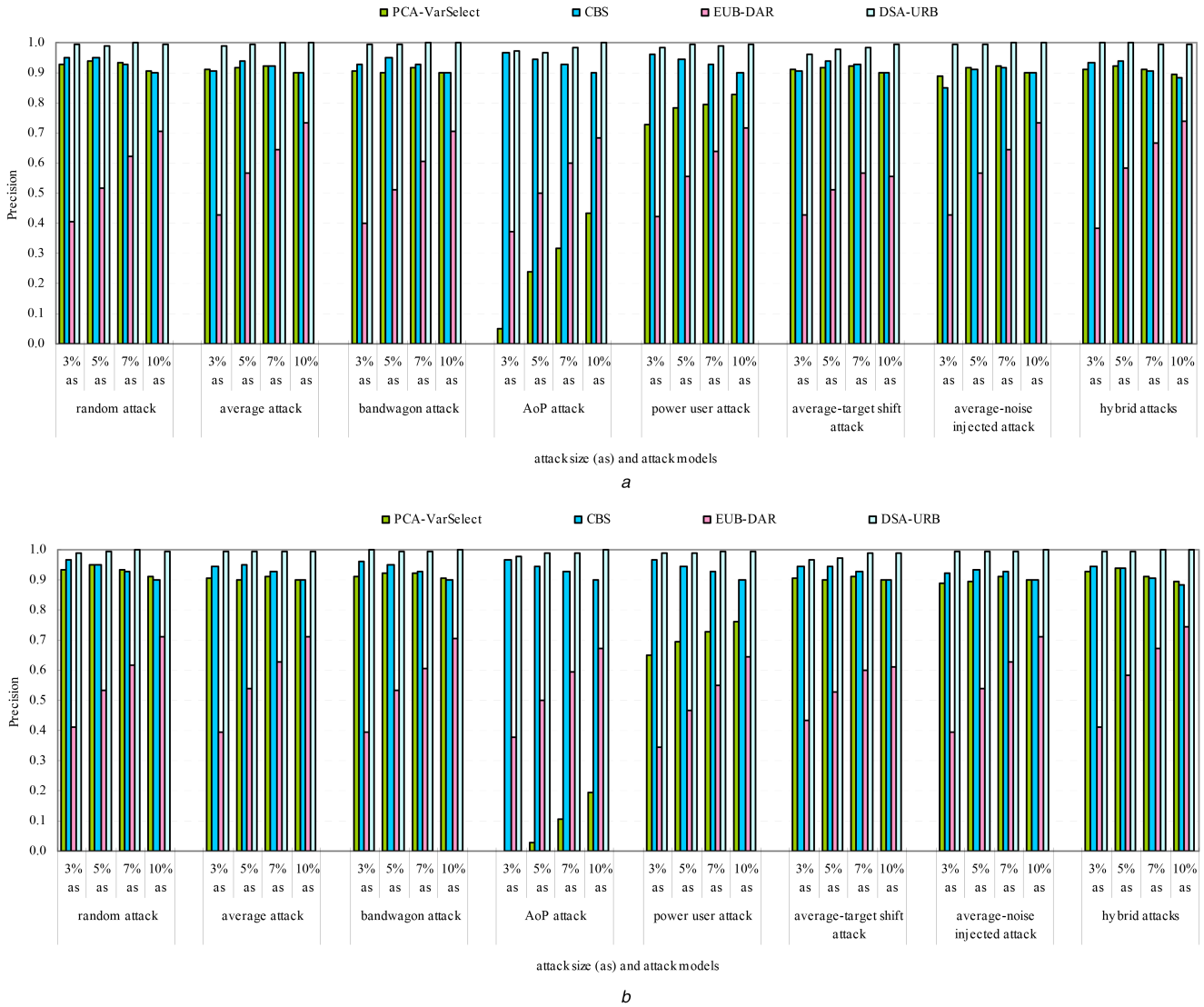

Fig. 13 *Comparison of precision for four methods under eight attacks. Note that the precision of PCA-VarSelect is 0 in detecting AoP attack with 5% filler size and 3% attack size*

(a) 3% Filler size, ***(b)*** 5% Filler size

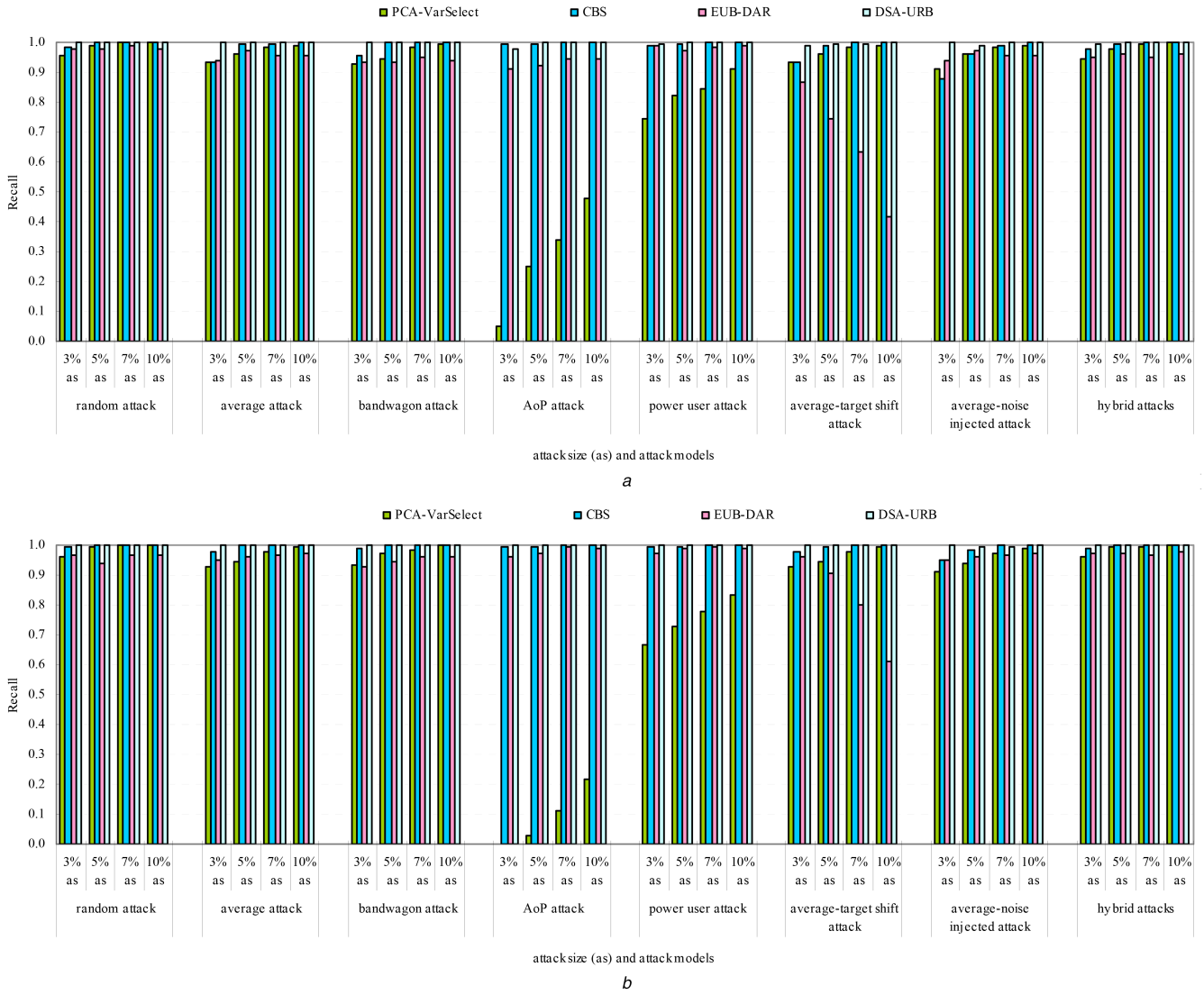

Fig. 14 *Comparison of recall for four methods under eight attacks*
(a) 3% Filler size, ***(b)*** 5% Filler size

Table 4 Comparison of F1-measure for DSA-URB and without LDA under eight attacks

Filler size		3%				5%			
Attack size		3%	5%	7%	10%	3%	5%	7%	10%
random attack	without LDA	0.0072	0.0234	0.0659	0.0587	0.0168	0.0314	0.0101	0.0354
	DSA-URB	0.9972	0.9951	1.0000	0.9983	0.9945	0.9983	1.0000	0.9983
average attack	without LDA	0.0116	0.0248	0.0460	0.0418	0.0132	0.0334	0.0316	0.0023
	DSA-URB	0.9945	0.9967	0.9988	0.9991	0.9972	0.9983	0.9976	0.9983
bandwagon attack	without LDA	0.0048	0.0300	0.0357	0.0119	0.0084	0.0312	0.0181	0.0494
	DSA-URB	0.9972	0.9983	0.9988	0.9991	1.0000	0.9983	0.9976	0.9991
AoP attack	without LDA	0.0138	0.0239	0.0476	0.0597	0.0104	0.0358	0.0020	0.0081
	DSA-URB	0.9752	0.9821	0.9918	0.9991	0.9891	0.9951	0.9941	0.9991
power user attack	without LDA	0.0172	0.0102	0.0582	0.0024	0.0199	0.0216	0.0087	0.0210
	DSA-URB	0.9891	0.9967	0.9953	0.9967	0.9945	0.9951	0.9976	0.9975
average-target shift attack	without LDA	0.0038	0.0204	0.0429	0.0444	0.0182	0.0279	0.0574	0.0190
	DSA-URB	0.9757	0.9853	0.9894	0.9950	0.9837	0.9869	0.9953	0.9942
average-noise injected attack	without LDA	0.0141	0.0047	0.0459	0.0170	0.0227	0.0237	0.0079	0.0422
	DSA-URB	0.9972	0.9934	0.9988	0.9983	0.9972	0.9950	0.9941	0.9991
hybrid attacks	without LDA	0.0195	0.0195	0.0434	0.0363	0.0040	0.0058	0.0419	0.0164
	DSA-URB	0.9972	1.0000	0.9976	0.9983	0.9972	0.9967	0.9988	0.9991

7 References

[1] Gunes, I., Kaleli, C., Bilge, A., *et al.*: 'Shilling attacks against recommender systems: a comprehensive survey', *Artif. Intell. Rev.*, 2014, **42**, (4), pp. 767–799

[2] Williams, C.A., Mobasher, B., Burke, R.: 'Defending recommender systems: detection of profile injection attacks', *Serv. Oriented Comput. Appl.*, 2007, **1**, (3), pp. 157–170

[3] Burke, R., O'Mahony, M.P., Hurley, N.J.: 'Robust collaborative recommendation', in (Eds.) Ricci, F., Rokach, L., Shapira, B., *et al.* (Eds.): '*Recommender systems handbook*' (Springer, Boston, MA, 2011), pp. 805–835

[4] Mobasher, B., Burke, R., Bhaumik, R., *et al.*: 'Toward trustworthy recommender systems: an analysis of attack models and algorithm robustness', *ACM Trans. Internet Technol.*, 2007, **7**, (4), pp. 1–38

[5] Williams, C., Mobasher, B., Burke, R., *et al.*: 'Detection of obfuscated attacks in collaborative recommender systems'. Proc. 17th European Conf. Artificial Intelligence, Riva del Garda, Italy, August 2006, pp. 19–23

[6] Hurley, N.J., Cheng, Z.P., Zhang, M.: 'Statistical attack detection'. Proc. Third Int. Conf. Recommender Systems, New York, NY, USA, October 2009, pp. 149–156

[7] Wilson, D.C., Seminario, C.E.: 'Evil twins: modeling power users in attacks on recommender systems'. Proc. Int. Conf. User Modeling, Adaptation, and Personalization, Aalborg, Netherlands, July 2014, pp. 231–242

[8] Burke, R., Mobasher, B., Williams, C., *et al.*: 'Classification features for attack detection in collaborative recommendation systems'. Proc. 12th Int. Conf. Knowledge Discovery and Data Mining, Philadelphia, PA, USA, August 2006, pp. 542–547

[9] Mehta, B., Nejdl, W.: 'Attack resistant collaborative filtering'. Proc. 31st Annual Int. ACM SIGIR Conf. Research and Development in Information Retrieval, Singapore, July 2008, pp. 75–82

[10] Wu, Z., Wu, J., Cao, J., *et al.*: 'HySAD: a semi-supervised hybrid shilling attack detector for trustworthy product recommendation'. Proc. 18th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining, Beijing, China, August 2012, pp. 985–993

[11] Li, W.T., Gao, M., Li, H., *et al.*: 'An shilling attack detection algorithm based on popularity degree features', *Zidonghua Xuebao/Acta Autom. Sin.*, 2015, **41**, (9), pp. 1563–1575 (in Chinese)

[12] Zhou, W., Wen, J., Xiong, Q., *et al.*: 'SVM-TIA a shilling attack detection method based on SVM and target item analysis in recommender systems', *Neurocomputing*, 2016, **210**, pp. 197–205

[13] Yang, Z., Xu, L., Cai, Z., *et al.*: 'Re-scale AdaBoost for attack detection in collaborative filtering recommender systems', *Knowl. Based Syst.*, 2016, **100**, pp. 74–88

[14] Zhang, F., Zhou, Q.: 'HHT-SVM: an online method for detecting profile injection attacks in collaborative recommender systems', *Knowl.-Based Syst.*, 2014, **65**, pp. 96–105

[15] Zhang, F., Chen, H.: 'An ensemble method for detecting shilling attacks based on ordered item sequences', *Secur. Commun. Netw.*, 2016, **9**, (7), pp. 680–696

[16] Bryan, K., O'Mahony, M., Cunningham, P.: 'Unsupervised retrieval of attack profiles in collaborative recommender systems'. Proc. Second ACM Conf. Recommender Systems, Lausanne, Switzerland, October 2008, pp. 155–162

[17] Mehta, B., Nejdl, W.: 'Unsupervised strategies for shilling detection and robust collaborative filtering', *User Model. User-Adapt. Interact.*, 2009, **19**, (1/2), pp. 65–97

[18] Bhaumik, R., Mobasher, B., Burke, R.D.: 'A clustering approach to unsupervised attack detection in collaborative recommender systems'. Proc. Seventh IEEE Int. Conf. Data Mining, Las Vegas, NV, USA, 2011, pp. 181–187

[19] Lee, J., Zhu, D.: 'Shilling attack detection – a new approach for a trustworthy recommender system', *Informs J. Comput.*, 2012, **24**, (1), pp. 117–131

[20] Zhang, Z., Kulkarni, S.R.: 'Graph-based detection of shilling attacks in recommender systems'. Proc. IEEE Int. Workshop on Machine Learning for Signal Processing, Southampton, UK, September 2013, pp. 1–6

[21] Zhang, Z., Kulkarni, S.R.: 'Detection of shilling attacks in recommender systems via spectral clustering'. Proc. 17th Int. Conf. Information Fusion, Salamanca, Spain, July 2014, pp. 1–8

[22] Zhang, Y., Tan, Y., Zhang, M., *et al.*: 'Catch the black sheep: unified framework for shilling attack detection based on fraudulent action propagation'. Proc. 24th Int. Conf. Artificial Intelligence, Buenos Aires, Argentina, 2015, pp. 2408–2414

[23] Yang, Z., Cai, Z., Guan, X.: 'Estimating user behavior toward detecting anomalous ratings in rating system', *Knowl.-Based Syst.*, 2016, **111**, pp. 144–158

8 Appendix

The derivation process of (4) is as follows:

$$
\begin{aligned}
p\left(z_{n,m}=e_k \middle| Z_{\neg_{n,m}}, U^*\right) &\propto p\left(z_{n,m}=e_k, u_{n,m} \middle| Z_{\neg_{n,m}}, U^*_{\neg_{n,m}}\right) \\
&= \int p\left(z_{n,m}=e_k, u_{n,m}, \boldsymbol{\theta}_n, \boldsymbol{\varphi}_{ek} \middle| Z_{\neg_{n,m}}, U^{*|}_{\neg_{n,m}}\right) \mathrm{d}\boldsymbol{\theta}_n \mathrm{d}\boldsymbol{\varphi}_{ek} \\
&= \int p(z_{n,m}=e_k, |\boldsymbol{\theta}_n) p\left(\boldsymbol{\theta}_n \middle| Z_{\neg_{n,m}}, U^*_{\neg_{n,m}}\right) \\
&\quad \cdot p(u_{n,m}|\boldsymbol{\varphi}_{ek}) p\left(\boldsymbol{\varphi}_{ek} \middle| Z_{\neg_{n,m}}, U^*_{\neg_{n,m}}\right) \mathrm{d}\boldsymbol{\theta}_n \mathrm{d}\boldsymbol{\varphi}_{ek} \\
&= \int p(z_{n,m}=e_k, |\boldsymbol{\theta}_n) p\left(\boldsymbol{\theta}_n \middle| Z_{\neg_{n,m}}, U^*_{\neg_{n,m}}\right) \mathrm{d}\boldsymbol{\theta}_n \\
&\quad \cdot \int p(u_{n,m}|\boldsymbol{\varphi}_{ek}) p\left(\boldsymbol{\varphi}_{ek} \middle| Z_{\neg_{n,m}}, U^*_{\neg_{n,m}}\right) \mathrm{d}\boldsymbol{\varphi}_{ek} \\
&= E\left(\theta_{ne_k}\right) E\left(\varphi_{e_k u_{n,m}}\right)
\end{aligned}
$$

where $E\left(\theta_{ne_k}\right)$ and $E\left(\varphi_{e_k u_{n,m}}\right)$ represent expectation of θ_{ne_k} and $\varphi_{e_k u_{n,m}}$, respectively.

From two physical processes of LDA model, we can obtain the posterior distribution of Dirichlet distribution of θ_n and φ_k as follows:

$$
p(\theta_n|U^*, \alpha) = \mathrm{Dirichlet}\left(\theta_n \middle| C_{i_n} + \alpha\right) \tag{15}
$$

$$
p\left(\varphi_{e_k} \middle| Z, \beta\right) = \mathrm{Dirichlet}\left(\varphi_{e_k} \middle| C_{e_k} + \beta\right) \tag{16}
$$

where C_{i_n} is a K-dimensional vector whose components represent the number of users who rate item i_n and prefer to each latent topic, C_{e_k} is an M-dimensional vector whose components represent the number of times that each user prefers to latent topic e_k.

Under the Bayesian framework, take the mean of the corresponding posterior distribution of θ_{ne_k} and $\theta_{e_k u_{n,m}}$ as the estimated value of $E\left(\theta_{ne_k}\right)$ and $E\left(\varphi_{e_k u_{n,m}}\right)$, we obtain

$$
\hat{\theta}_{ne_k} = \frac{C^{e_k}_{i_n, \neg_{n,m}} + \alpha}{\sum_{k=1}^{K} C^{e_k}_{i_n, \neg_{n,m}} + K\alpha} \tag{17}
$$

$$
\hat{\varphi}_{e_k u_{n,m}} = \frac{C^{u_f}_{e_k, \neg_{n,m}} + \beta}{\sum_{f=1}^{M} C^{u_f}_{e_k, \neg_{n,m}} + M\beta} \tag{18}
$$

Thus, we obtain the desired (4).