Detecting shilling attacks in private environments

Ihsan Gunes[1] · Huseyin Polat[1]

Received: 20 January 2015 / Accepted: 25 July 2016 / Published online: 20 September 2016

Abstract Privacy-preserving collaborative filtering algorithms are successful approaches. However, they are susceptible to *shilling attacks*. Recent research has increasingly focused on collaborative filtering to protect against both privacy and shilling attacks. Malicious users may add fake profiles to manipulate the output of privacy-preserving collaborative filtering systems, which reduces the accuracy of these systems. Thus, it is imperative to detect fake profiles for overall success. Many methods have been developed for detecting attack profiles to keep them outside of the system. However, these techniques have all been established for non-private collaborative filtering schemes. The detection of shilling attacks in privacy-preserving recommendation systems has not been deeply examined. In this study, we examine the detection of shilling attacks in privacy-preserving collaborative filtering systems. We utilize four attack-detection methods to filter out fake profiles produced by six well-known shilling attacks on perturbed data. We evaluate these detection methods with respect to their ability to identify bogus profiles. Real data-based experiments are performed. Empirical outcomes demonstrate that some of the detection methods are very successful at filtering out fake profiles in privacy-preserving collaborating filtering schemes.

Keywords Detection · Shilling attack · Privacy · Collaborative filtering · Recommendation

1 Introduction

Electronic commerce has become widespread due to the rapid development of in Internet technologies. A large number of products are sold via the Internet. Recommender systems have been developed to help customers select appropriate products. One of the most

✉ Huseyin Polat
polath@anadolu.edu.tr

[1] Computer Engineering Department, Anadolu University, 26470 Eskisehir, Turkey

commonly used recommender systems is collaborative filtering (CF) (Bobadilla et al. 2013; Jia and Liu 2015). Customers may not want their product preferences and the products they rate to be known. To protect such personal preferences, privacy-preserving collaborative filtering (PPCF) methods have been developed (Bilge et al. 2013a; Casino et al. 2013, 2015; Jeckmans et al. 2013; Ozturk and Polat 2015). The goal of PPCF schemes is to provide recommendations with acceptable accuracy while preserving private data.

One common privacy-preserving technique utilized in PPCF algorithms is called *randomized perturbation* (Bilge et al. 2013a), which disguises data by adding noise data. Thus, data collectors storing the disguised data cannot learn actual rates but continue to produce accurate predictions. A Gaussian or uniform distribution with a zero mean (μ) and a standard deviation (σ) is used to produce random numbers. To hide the rated and/or unrated products, some of the uniformly randomly selected unrated items cells are filled with random numbers.

Malicious users who attempt to manipulate the outcomes of CF and PPCF systems can attack such systems by adding fake profiles in their databases. The purpose of these attacks is usually to increase the popularity of a target product (*push* attack) or reduce it (*nuke* attack). These types of attacks are known as *shilling attacks* (O'Mahony et al. 2004; Gunes et al. 2014). Mobasher et al. (2007) show that CF schemes are vulnerable to shilling attacks. Various PPCF systems are also vulnerable to such attacks, as shown by Gunes et al. (2013a, b) and Bilge et al. (2014a). Shilling attacks might significantly affect the accuracy of the estimated predictions in PPCF schemes. Thus, it is very important to detect these types of attacks and reduce their effects for recommendation systems to function correctly. Different detection methods have been developed and applied to CF algorithms to identify fake profiles (Li and Luo 2011; Zhang and Zhou 2014; Zhou et al. 2014a, b; Xia et al. 2015; Zhang and Zhou 2015). However, only a single study by Gunes and Polat (2015a) has focused on detecting shilling profiles in PPCF algorithms. The authors propose a hierarchical clustering-based scheme to detect fake profiles in private environments. They consider a single detection method. Hence, we apply the most commonly used four detection methods to PPCF schemes. We modify the current four detection methods so that they are applicable to PPCF methods and conduct experiments with real data sets. In practice, six attacking models developed previously for attacking PPCF algorithms are employed. Four of the most common detection schemes are utilized as detection technique, where two common data sets are used for empirical analysis.

The contributions of this article in general can be summarized as follows:

1. Four widely used shilling attack detection techniques are modified in such a way to detect shilling profiles generated by six shilling attacks (four push and two nuke attacks) in masked data in PPCF systems' databases.
2. Different sets of experiments are performed using two real data sets to evaluate the success of the detection techniques for PPCF systems.
3. The modified detection methods are compared with the one proposed by Gunes and Polat (2015a) with respect to detection performance. They are also compared with the ones used in non-private environments.

The rest of the paper is structured as follows. In Sect. 2, related studies are reviewed, and the differences between this work and existing work are briefly presented. Preliminary work is concisely explained in Sect. 3. In Sect. 4, we explain how to apply these four detection methods to PPCF algorithms and describe the modified detection techniques. We explain in detail real data-based trials and their outcomes in Sect. 5. Empirical results are discussed in Sect. 6. We conclude the paper and provide future research directions in Sect. 7.

2 Related work

Chirita et al. (2005) performed the first work on the detection of shilling profiles by checking profile properties in CF systems. They considered the simplest attacking models for random and average methods. Burke et al. (2006) studied the effectiveness of different characteristics derived from user profiles for attack detection. Their study demonstrated that a machine learning sorting method that included attributes derived from attack models was more successful than more widespread detection algorithms. To detect attack profiles in CF systems, variable selection based on principal component analysis (PCA) can be utilized (Mehta et al. 2007). This approach can only be applied to a dense user-item matrix because PCA cannot tolerate null values. Mehta (2007) and Mehta and Nejdl (2009) attempted to detect attack profiles using PLSA-based clustering algorithm. Attack profiles were distributed to the same clusters based on their similarities. Bhaumik et al. (2011) reported that the similarity of the values of the detection attributes depends on the similarity of the attack profiles. Based on these attributes, the profiles were separated into clusters using a *k*-means algorithm.

Bhaumik et al. (2006) studied two statistical operation control methods and proposed statistical process control-based methods as alternative solutions. Zhang et al. (2006b) suggested an attack detection method establishing a low dimensional linear model. Hurley et al. (2009) discussed the detection of standard attack models using statistical approaches. Using the Neyman–Pearson method, they distinguished real and attack profiles. Li and Luo (2011) established a probability model within the framework of a Bayesian network to detect any possible attack. Zhou et al. (2014a, b) reported the utilization of statistical metrics to determine the rating patterns of attackers and employed this metric to examine differences in rating configurations between shilling and genuine profiles in shilling attacks. Su et al. (2005) focused on detecting group attack users rather than individual attacks using a similarity algorithm that evaluated the consistency of a user's votes for similar products. O'Mahony et al. (2006) recommended a signal detection approach for determining shilling profiles. Zhang et al. (2006a) suggested the construction of a time series of the votes for a product, in which a frame is used to group successive votes for a product. In each frame, the sample average and entropy values are calculated, and the results are interpreted to identify suspicious attitudes. Zhang et al. (2009) suggested using trust values as a metric to protect systems based on trust.

Tang and Tang (2011) analyzed the time gaps between voting times to identify suspicious attitudes affecting top-*N* lists in prediction systems. Zhang (2011) focused on protecting trust-based recommendation systems from attacks. A data genealogical tree method was proposed to defend attacks by tracing the recommendation history and finding the victim nodes. Noh et al. (2014) proposed a novel robust recommendation algorithm, *RobuNec*, that provides admission control as a defense mechanism against shilling attacks. Cao et al. (2013) intended to utilize semi-supervised learning to identify attack profiles and described the application of semi-supervised learning to shilling attack detection. Zhang et al. (2013) proposed two methods, *CluTr* and *WCluTr*, for building robust CF system to prevent shilling attacks. *CluTr* filters out suspicious fake users in the formed clusters, and *WCluTr* uses trust information to strengthen similarities among genuine users. Morid et al.

(2013) proposed a new attack detection method that detects influential users instead of the whole user set to improve detection performance. Xia et al. (2015) proposed a novel detection scheme based on dynamic time interval segmentation. The method is able to detect fake profiles regardless of specific attack types.

Zhang and Zhou (2014) constructed rating series for each profile based on originality and product reputation. They employed an experimental mode decomposition method to decompose each rating series and extract Hilbert spectrum-based characteristics to describe shilling attacks. Zhang and Zhou (2015) also proposed a shilling attack detection method based on a back propagation neural network and ensemble learning method. Bilge et al. (2014b) recommended an original shilling attack identification technique for specific attacks based on bisecting *k*-means clustering method. The results indicated that the technique was exclusively successful against bandwagon, segment, and average attacks. Li (2014) proposed a method that discloses latent factors appealing missing ratings under the non-arbitrary-missing mechanism and further unites these hidden issues using a Dirichlet process in the framework of a probabilistic generative model. Zhuo and Kulkarni (2014) presented a technique to make CF systems resistant to shilling attacks. The maximum sub-matrix is explored by converting the issue into a graph and combining nodes by heuristic functions. Chung et al. (2013) suggested a Beta-protection approach to address the drawbacks of current detection techniques.

Our study differs from those described above. We attempted to detect attacks on PPCF schemes (*private environments*), whereas other studies have focused on attack detection on CF schemes without privacy (*non-private environments*). A literature review revealed that only one detection method has been used for attacks on PPCF. Gunes and Polat (2015a) proposed a hierarchical clustering-based shilling attack detection method in private environments. They scrutinized the ratings of target items to improve the overall performance of their scheme. Their empirical results revealed that the proposed method could identify shilling profiles with decent accuracy. We researched all of the detection methods applied to CF algorithms described above and selected four that are the most used and most applicable to PPCF schemes. We used each of these four methods to detect fake profiles in the databases of PPCF systems.

In addition to detecting shilling attacks, robust PPCF schemes have been proposed (Bilge et al. 2013b). Also, there are various studies focusing on the robustness analysis of PPCF schemes, where the authors show how robust the schemes are against different shilling attacks (Gunes et al. 2013a, b; Bilge and Polat 2013b; Bilge et al. 2014a; Gunes and Polat 2015b; Yurekli and Kaleli 2016). Our study presented here is different from the abovementioned ones because we focus on attack detection while they focus on robustness analysis. Attack detection and robustness analysis require different approaches.

3 Preliminaries

3.1 Shilling attacks on disguised data

Applying traditional shilling attacks against PPCF systems is difficult due to disguised data. Therefore, attackers must modify conventional shilling attacks. Gunes et al. (2013a, b) redesigned well-known shilling attacks to enable their application to masked databases. Attackers must decide on either a uniform or Gaussian random number distribution to generate random numbers to mask their data. Moreover, standard deviations (σ

values) are uniformly randomly selected from the range (0, σ_{max}] for each attack profile prior to generating fake profiles, where σ_{max} is a privacy parameter. Modified attacks can be briefly explained as follows (Gunes et al. 2013a, b). In a typical shilling attack, one item is selected as a target item whose popularity is manipulated by the attackers. A set of items called filler items is selected to be filled with bogus ratings. Another set of items, referred to as selected items, are selected to enable specific shilling attacks by filling these items with values estimated from known knowledge about the attacked system. Finally, the remaining items in a profile are left unrated.

In the *random attack* model, the set of selected items is empty. Randomly selected filler items are filled with random values. The target item is assigned the highest possible random value. Similarly, in the *average attack* model, the set of selected items is also empty. Randomly selected filler items are filled with the item's mean. Recall that such mean values are estimated based on perturbed data due to privacy concerns in PPCF schemes. The target item is given the maximum random value to increase its popularity. In the *bandwagon attack* model, items are selected from popular items that are densely rated and have high means. The selected items are chosen from among popular items. The selected items and randomly chosen filler items are filled with random values. The selected items are given the largest random numbers because they are popular items and are expected to have higher ratings. Similarly, the target item is assigned the maximum possible random value to push its popularity. The *segment attack* model is designed to attack a specific group or segment of users. The selected items are chosen from among high average products in a specific segment. The selected items and the target item are filled in a similar manner as in the bandwagon attack model.

The r*everse bandwagon* and the *love/hate attack* models are nuke attacks. In a *reverse bandwagon attack*, selected items are chosen from among unpopular items that are densely rated and have low means. The selected items and randomly chosen filler items are filled with random values. In this case, the selected items are given the lowest values, whereas the target item is assigned the minimum random value so that the target item can be nuked. In the case of a *love/hate attack*, a set of selected items is empty for the same reasons. Randomly determined filler items are filled with high random values. The target item is appointed the minimum random value to decrease its popularity.

3.2 Shilling attack detection methods

We explain the Chirita algorithm, *kNN* classifier, *k*-means clustering, and PCA-based variable selection methods as shilling attack detection methods. Chirita et al. (2005) attempted to classify profiles using eight generic attributes described as follows:

1. *Number of prediction-differences* (TFS): A prediction is determined for each user. TFS describes the net difference after erasing the user from the system.
2. *Standard deviation in user's ratings*: This metric shows the selecting degree above the average of a user.
3. *Degree of agreement with other users*: This metric exhibits the difference degree of each of a user's selections from the average selecting degree of an item.
4. *Degree of similarity with top neighbors*: The weight of the similarities between a user and the closest *k* number of users of her.
5. *Rating deviation from mean agreement* (RDMA): This metric determines the deviation from the pre-given average values of some of specific items.

6. *Weighted deviation from mean agreement* (WDMA): The RDMA is weighted by the square of the number of votes for the item.
7. *Weighted degree of agreement* (WDA): This metric differs from the RDMA metric in that the former is not divided by the total number of votes given by the user.
8. *Length variance* (lengthVar): The metric measures the extent to which the length of the investigated profile differs from the average profile length.

In addition to generic attributes, Burke et al. (2006) subsequently used five model-specific attributes for classifying profiles described as follows:

1. *Filler mean variance* (FMV): FMV calculates the variation between the average value of the item and the value of each item (filler items I_F) assumed to exist in the item set of each profile.
2. *Filler mean difference* (FMD): The major difference between FMD and the model-based metric is the use of the absolute value of the difference between the vote of the user and the average of the votes instead of the square of the difference value.
3. *Filler average correlation*: This metric calculates the correlation between each item value and the average item value in the filler item set of the investigated profile.
4. *Filler mean target difference* (FMTD): FMTD calculates the difference between the average of the assumed filler item set and the average of the possible target item set.
5. *Profile variance*: This metric calculates the profile variance, which tends to be low compared to authentic users.

3.2.1 Chirita algorithm

Chirita et al. (2005) proposed an algorithm (referred to as the *Chirita algorithm*) on RDMA for detecting shilling attackers. The proposed algorithm is a two-step method as follows:

1. The algorithm computes the average similarity with the top neighbors for all users using the Pearson correlation coefficient. Thus, only a subset of the total users are used for computations. The algorithm then selects only those users with an average similarity less than 0.5 of the maximum average similarity in the system to compute the RDMA.
2. The algorithm associates with each value of RDMA a function that evaluates the probability (PA_u) that the respective user is a shilling attacker. The first s profiles, sorted based on PA_u, are considered attack profiles. PA_u is used to decide whether a profile is an attack profile or not. Higher PA_u values for RDMA mean that the related profiles are attack profiles.

3.2.2 kNN classifier-based detection algorithm

Mobasher et al. (2006, 2007) proposed a method based on classification that utilizes a total of 15 detection attributes: six generic attributes (WDMA, RDMA, WDA, LengthVar, DegSim with $k = 450$ and DegSim$'$ with $k = 2$, $d = 963$, where k is the number of neighbors and d is co-rate factor); six average attack model attributes (filler mean variance, filler mean difference and profile variance; computed for both push and nuke); two bandwagon attack model (FMTD; computed for both push and nuke); and one target detection model attribute (TMF). Class labels and detection attributes are generated for the entire data set, which is divided into two equal-sized sub-sections of training and test data sets. A *kNN* with $k = 9$ is used in the classifier. The *kNN* classifiers are implemented using Weka. For each test, the second half of the data is injected with attack profiles and then run through the classifier built on the augmented first half of the data.

3.2.3 k-means clustering-based detection algorithm

The attack profiles are similar to each other because they are generated by known algorithms. Consequently, when a *k*-means clustering algorithm is employed on the data set to which attack profiles are added, most of the attack profiles should be distributed to the same cluster. The most important issue at this stage is to identify the cluster in which the attack profiles will be gathered. Mehta and Nejdl (2009) aimed to find the tightest cluster (in which the elements are very similar to each other) in their study of clustered-based detection. Consequently, for each cluster, the distance of the profiles from the center is calculated. Then, the average distance is computed. The cluster with the shortest average distance to the center is defined as the attack cluster. The determined cluster is finally isolated.

3.2.4 PCA-based variable selection detection algorithm

In a recommendation system, if users are considered variables, there will be data with a similar number of dimensions. Thus, dimensionality reduction would discard these dimensions due to low covariance of the dimensions. Low covariance is observed not only between shilling users but also shilling users and normal users. PCA computes principal components that are oriented more toward real users, who exhibit the maximum variance of the data. Consequently, those users who display the least covariance with all other users should be selected. This quantity is used to select some variables from the original data to which PCA was applied. In the algorithm below (Algorithm 1), Mehta and Nejdl (2009) depicted their proposed approach for variable selection. The first *s* users are selected as the attack profiles and are isolated from the system, where *s* is the number of attack profiles added to the system.

```
Algorithm 1 PCA Select Users (D: user-item matrix & s: cut-off parameter)
1: $D \leftarrow$ z-scores (D)
2: $COV \leftarrow D^T D$                                   {Covariance of $D^T$}
4: $U \lambda U^T$ = Eigenvalue Decomposition (COV)
5: $PCA_1 \leftarrow U(:, 1)$                             {First Eigenvector of COV}
6: $PCA_2 \leftarrow U(:, 2)$                             {Second Eigenvector of COV}
7: $PCA_3 \leftarrow U(:, 3)$                             {Third Eigenvector of COV}
8: for all column id users in D do
9:     Distance (user) $\leftarrow PCA_1(user)^2 + PCA_2(user)^2 + PCA_3(user)^2$
10: end for
11: Sort Distance
Output: Return s users with smallest Distance
```

4 Shilling detection methods for PPCF schemes

The detection algorithms described above have been used for CF attacks. However, we use these algorithms for PPCF attacks by adapting them to allow their use to identify shilling attacks on masked data. There are two confidential data types in PPCF schemes: actual rating values and rated and/or unrated items. To protect these private data, random numbers are generated using either a uniform or Gaussian distribution with zero mean (μ) and σ, which is uniformly randomly selected from (0, σ_{max}]. These noise data are added to actual votes. Additionally, some of the uniformly randomly selected unrated items cells are filled with noise data. To select unrated cells, a β value is uniformly randomly selected from (0, β_{max}], where β_{max} is a privacy parameter and represents the upper bound of β values. Then, β percent of empty cells are filled with random numbers. The values of σ_{max} and β_{max} depend on the privacy and accuracy levels required by the CF users (Bilge et al. 2014a).

Recall that Chirita algorithm computes similarities using the Pearson correlation coefficient. Such similarities can also be estimated with decent accuracy based on perturbed data (Bilge et al. 2014a). Similarly, RDMA can be computed from masked data. Finally, the probability that a profile is an attack profile can be calculated using RDMA on masked data. Therefore, the Chirita algorithm can be employed to determine shilling profiles in PPCF schemes. The generic attribute values used in the Chirita algorithm are calculated for disguised data. Chirita et al. (2005) define $\alpha = 10$ in the formula to calculate the probability that a profile is an attack profile. When the Chirita algorithm is used on PPCF schemes, the best result is obtained when $\alpha = 1$, and this value is used in our trials.

The second detection method, the *kNN* classifier, is based on detection attributes. The values of such attributes can be determined from disguised data. The modified classifier utilizes 14 detection attributes: six generic attributes (WDMA, RDMA, WDA, LengthVar, DegSim with $k = 450$ and DegSim$'$ with $k = 2, d = 963$); six average attack models (filler mean variance, filler mean difference, and profile variance; computed for both push and nuke); and two bandwagon attack models (FMTD; computed for both push and nuke). As in a non-private environment, class labels and detection attributes are generated for the whole data set. A *kNN* with $k = 9$ is used as our classifier. This is the same value used in the study by Mobasher et al. (2007) and allows the results of our proposed method to be compared with the result obtained by Mobasher et al. (2007). All experiments in the present study were conducted using both the proposed method and the one introduced by Mobasher et al. (2007).

The k-means clustering-based detection method utilizes the Pearson correlation coefficient to group users into k clusters. As shown by Bilge and Polat (2013a), k-means clustering can group users into clusters with decent accuracy using disguised data. The success of this method mainly depends on how accurately users are clustered. The similarity of each profile in the clusters to the cluster center is calculated to determine the attack cluster. The similarity between the attack profiles is higher than that of the other profiles. The cluster with the highest average similarity is isolated from the system. The selection of the cluster number is important for the performance of the application. The results of the trials revealed that the ideal number of clusters is 12. The choice of initial cluster centers can affect the results slightly. The steps of the k-means clustering-based detection algorithm employed on perturbed data in PPCF schemes are defined as follows:

Algorithm 2 k-means clustering-based detection method for masked data

Let $U' = \{u_1, u_2, \ldots, u_n\}$ be a set of disguised data vectors and $C = \{c_1, c_2, \ldots, c_k\}$ be a set of cluster centers.

```
1: Randomly select the 'k' cluster centers.
2: Estimate the similarity between each data vector and cluster centers.
3: Assign the data vector to the closest cluster.
4: Recalculate the new cluster center.
5: Recalculate the similarity between each data vector and new obtained cluster centers.
6: If no data vector is reassigned then stop, otherwise, repeat from step 3.
7: Determine the cluster with the highest average similarity as the shilling cluster.
```

The steps defined in Algorithm 1 for the PCA-based variable selection detection method are also used in PPCF. However, in PPCF schemes, disguised z-score data are used as input data. Although the data are disguised, random numbers with an average value of 0 are added to z-scores, and masked values are obtained $(z'_{uj} = z_{uj} + r_{uj})$. Similarly, the average of the z-score data is expected to be 0. Bilge and Polat (2013a) state that during the scalar product and sum process, the effect of random numbers can be neglected because the average of random numbers is 0. However, the same random numbers must be multiplied while calculating the diagonal components of the matrix obtained as a result of the COV $\leftarrow D^T D$ process described in the third line of Algorithm 1. Multiplying the same random numbers, r_{uj}^2, will create an excess value. To reduce these effects, the $n\sigma_r^2$ value is extracted from the diagonal components, in which n indicates the number of random numbers and σ_r indicates the standard deviation of the random numbers. After modifying Algorithm 1 as described above, we utilize it as a detection method for filtering out shilling profiles in PPCF schemes' databases.

5 Experiments

To demonstrate the ability of the four modified shilling attack detection methods on disguised databases in PPCF schemes for six shilling attack models, different sets of experiments were performed on real data sets. The success of shilling attacks depends on two control parameters: *filler size* and *attack size*. *Filler size* is the percentage of empty cells that are filled in the attacker's profile. *Attack size* denotes the number of attack profiles to inject. In this sense, attack size is directly proportional to the number of users in the system. For instance, a five percent *attack size* indicates that there are 50 attack profiles in a system initially holding 1000 users. Privacy-preserving control parameters are first kept constant, $\beta_{max} = 25\ \%$ and $\sigma_{max} = 2$. Then, to demonstrate how detection performance changes with varying values of these parameters, different trials are performed by changing their values.

5.1 Data sets and evaluation criteria

The MovieLens public data set (MLP) and Jester were used in the experiments. The GroupLens research team collected MLP (http://www.grouplens.org). MLP comprises 100 K ratings of 1682 movies by 943 users. Within the set, the ratings are discrete values from 1 to 5. Each user rated at least 20 movies. Jester data set was released from the Jester Joke Recommender System (http://eigentaste.berkeley.edu/dataset/). Jester includes numeric continuous ratings ranging from −10 to 10. It is relatively dense (around 56 %). There are 73,496 users and 100 jokes in the set. Although we used all users' data in MLP, we randomly selected 1000 users from Jester.

To measure the performance of the detection methods, the standard measurements of *precision* and *recall* are used. The basic definition of such metrics is given as follows:

Precision = *Number of true positives/(Number of true positives* + *Number of false positives)*

Recall = *Number of true positives/(Number of true positives* + *Number of false negatives)*

Number of true positives is the number of correctly classified attack profiles, while *number of false positives* is the number of authentic profiles misclassified as attack profiles, and *number of false negatives* is the number of attack profiles misclassified as authentic profiles.

5.2 Methodology

Our experimental methodology was as follows. Two distinct target item sets were first formed for each real data set. In MLP, each target item set includes 50 movies for push and nuke attacks. Due to the limited number of jokes in Jester, target item sets for push and nuke attacks include 25 jokes. Target items were randomly selected by stratified sampling. Intuitively, attempting to push a popular item or nuke an unpopular one is considered unreasonable. Thus, the push and nuke attack sets comprised unpopular and popular items, respectively. During the trials, attack profiles were created to manipulate the outcomes of the target items. We did not perform segment attack in Jester because there is no joke category in Jester. The tests were repeated 100 times due to randomization in the perturbation process.

5.3 Empirical results

5.3.1 Effects of filler size parameter

Experiments were performed to illustrate the performance of the detection methods with varying *filler size* values while detecting fake profiles in masked databases. *Filler size* was varied from 5 to 50 %, while *attack size*, β_{max}, and σ_{max} were maintained constant at 15, 25, and 2 %, respectively. The overall averages of the precision and recall values for the Chirita, *kNN* classifier, *k*-means clustering-based, and PCA-based detection algorithms with varying filler size values are presented in Tables 1 and 2 for MLP and Jester, respectively, where *RB* refers to the reverse bandwagon attack model.

As indicated in Tables 1 and 2, the empirical outcomes for precision and recall were equal for the Chirita algorithm for both data sets. The profiles are listed from top to bottom according to PA_u, and the first *s* profiles are classified as attack profiles. Since *s* number of attack profiles added to the system, the precision and recall values are equal. An increase in the filler size value did not significantly change the precision and recall values for all attack models. Therefore, the Chirita algorithm exhibited weak detection operation performance in private environments. The best outcomes were usually observed when the filler size was 25 and 50 % for MLP and Jester, respectively. The most successful results were obtained for a random attack and love/hate attack for MLP and Jester, respectively. For an average attack, all filler size values received a value 0 because the Chirita algorithm performs the classification operation by specifically considering the RDMA attribute value. The RDMA values will be higher for attack profiles than real profiles. However, while forming average attack profiles, because the filler items are filled with the item mean, the RDMA value decreases. Compared to the outcomes for a non-private environment published by Chirita et al. (2005), the Chirita algorithm provides lower results for private environments. There are couple of reasons why our results are lower. First, in the report by Chirita et al. (2005), there were simultaneous attacks to three target items. Consequently, the RDMA values were higher. Second, our values might be lower because of the use of disguised data in PPCF schemes.

The *kNN* classifier algorithm was quite successful in the detection operation of the PPCF attack models. As the filler size value increased from 5 to 50 %, nearly all of the precision and recall values varied between 0.800 and 1.000 for all attack models for both

Table 1 Performance of detection algorithms with varying *filler size* (MLP)

	Precision				Recall			
Filler size	5	10	25	50	5	10	25	50
Chirita								
Random	0.206	0.214	**0.221**	0.217	0.206	0.214	**0.221**	0.217
Average	0.000	0.000	**0.000**	0.000	0.000	0.000	**0.000**	0.000
Bandwagon	0.143	0.181	**0.209**	0.207	0.143	0.181	**0.209**	0.207
Segment	0.169	0.141	**0.175**	0.138	0.169	0.141	**0.175**	0.138
RB	0.145	0.175	**0.195**	0.192	0.145	0.175	**0.195**	0.192
Love/hate	0.135	0.177	0.204	**0.205**	0.135	0.177	0.204	**0.205**
KNN classifier								
Random	**0.987**	0.884	0.872	0.987	0.987	0.987	0.974	**0.987**
Average	**1.000**	0.927	0.938	0.987	0.987	0.987	0.987	**0.987**
Bandwagon	0.776	0.800	0.817	**0.987**	0.987	0.987	0.987	**0.987**
Segment	**1.000**	0.925	0.873	0.987	0.987	0.961	0.805	**0.987**
RB	**0.987**	0.920	0.962	0.987	0.974	0.896	0.987	**0.987**
Love/hate	0.917	0.936	0.884	**0.987**	0.286	0.948	**0.987**	0.984
k-means								
Random	**0.501**	0.361	0.245	0.192	1.000	0.997	1.000	**1.000**
Average	**0.349**	0.347	0.308	0.255	0.838	1.000	1.000	**1.000**
Bandwagon	**0.358**	0.285	0.260	0.232	1.000	1.000	1.000	**1.000**
Segment	**0.436**	0.362	0.297	0.225	0.953	1.000	1.000	**1.000**
RB	**0.350**	0.313	0.290	0.248	1.000	1.000	1.000	**1.000**
Love/hate	**0.340**	0.275	0.231	0.229	0.995	1.000	1.000	**1.000**
PCA								
Random	0.300	0.330	**0.340**	0.270	0.300	0.330	**0.340**	0.270
Average	0.440	0.610	**0.650**	0.300	0.440	0.610	**0.650**	0.300
Bandwagon	0.090	0.080	**0.090**	0.090	0.090	0.080	**0.090**	0.090
Segment	0.090	0.080	0.090	**0.100**	0.090	0.080	0.090	**0.100**
RB	0.076	0.079	0.082	**0.088**	0.076	0.079	0.082	**0.088**
Love/hate	**0.060**	0.058	0.057	0.057	**0.060**	0.058	0.057	0.057

The best outcomes are given in bold

Table 2 Performance of detection algorithms with varying *filler size* (Jester)

	Precision				Recall			
Filler size	5	10	25	50	5	10	25	50
Chirita								
Random	0.135	0.137	0.153	**0.156**	0.135	0.137	0.153	**0.156**
Average	0.000	0.000	0.000	**0.000**	0.000	0.000	0.000	**0.000**
Bandwagon	0.012	0.036	0.100	**0.146**	0.012	0.036	0.100	**0.146**
RB	0.010	0.023	0.053	**0.068**	0.010	0.023	0.053	**0.068**
Love/hate	**0.227**	0.216	0.182	0.181	**0.227**	0.216	0.182	0.181
kNN classifier								
Random	**1.000**	1.000	0.974	0.835	**0.987**	0.987	0.987	0.987
Average	**1.000**	1.000	0.962	0.817	**0.987**	0.987	0.987	0.987
Bandwagon	**1.000**	0.974	0.894	0.894	**0.987**	0.987	0.987	0.987
RB	**1.000**	1.000	1.000	1.000	**0.987**	0.987	0.987	0.987
Love/hate	**1.000**	1.000	1.000	0.962	**0.987**	0.987	0.987	0.987
k-means								
Random	**0.493**	0.344	0.228	0.227	1.000	1.000	1.000	**1.000**
Average	**0.385**	0.301	0.257	0.236	1.000	1.000	1.000	**1.000**
Bandwagon	**0.306**	0.222	0.204	0.205	1.000	1.000	1.000	**1.000**
RB	**0.282**	0.253	0.240	0.235	1.000	1.000	1.000	**1.000**
Love/hate	**0.408**	0.356	0.266	0.218	1.000	1.000	1.000	**1.000**
PCA								
Random	**0.929**	0.879	0.783	0.681	**0.929**	0.879	0.783	0.681
Average	**0.967**	0.945	0.871	0.674	**0.967**	0.945	0.871	0.674
Bandwagon	**0.359**	0.306	0.247	0.220	**0.359**	0.306	0.247	0.220
RB	**0.367**	0.318	0.254	0.217	**0.367**	0.318	0.254	0.217
Love/hate	**0.879**	0.762	0.408	0.228	**0.879**	0.762	0.408	0.228

The best outcomes are given in bold

data sets. In other words, they are very close to each other for both data sets. The change in the precision value depicts the variability as a function of the filler size for the push attacks. At precision values less than 1.0, some of the real profiles were classified as attack profiles. However, in general, the *kNN* classifier algorithm was also successful in the PPCF algorithm, as in the CF algorithm. The disguise operation in the PPCF algorithm does not significantly affect the detection algorithm's performance. Because the *kNN* classifier can also create a model using training data, which were disguised in the PPCF schemes, the attack profiles on masked data can be detected easily in the test set using this model. As shown in Tables 1 and 2, like the precision values, the recall values of the *kNN* classifier algorithm indicate high success rates. For all filler size values, the algorithm performs very well with respect to recall. The recall value of the segment attack, which differs in purpose from the other attacks, might be slightly lower than those of the other attacks for MLP. Our results on MLP are similar to those calculated for the CF algorithm by Mobasher et al. (2007).

As indicated in Tables 1 and 2, the recall values were very close to each other for all attack models in MLP and they are the same for Jester. The precision values decreased with increasing filler size values for all attack models. Recall that the *k*-means clustering-based detection method performs the clustering operation by considering the similarities between profiles. Consequently, the attack model type is not significant. Because all of the attack models are formed using defined algorithms, they are all naturally similar to each other. Based on this similarity, the *k*-means clustering-based detection method identifies the tightest cluster and isolates that cluster from the system. As the filler size value increases, the attack profiles become more similar to the real profiles. Thus, there will be more real profiles in the cluster with the attack profiles. In this situation, more real profiles were isolated from the system, leading to lower precision values. The increase in filler size values only increased the number of real profiles in the cluster under search; and thus, the recall value was unaffected. Consequently, the *k*-means clustering-based detection method classifies nearly 100 % of the attack profiles correctly but pushes many of the real profiles to the outside of the system.

As shown in Table 1, the best results for the average attack model were obtained when the PCA-based detection method was utilized. Because this method generates average attack profiles by filling the filler item set with values around the item mean, a small covariance value of the attack profiles among each other is expected. Mehta et al. (2007) reported that the covariance value among attack profiles is smaller than that among real profiles. Consequently, the attack profiles might be detected by PCA-based variable selection technique. As shown in Table 1, both the precision and recall values for the average attack model reached 0.670. When establishing the other attack models, the filler item set is filled with random numbers generated with a known standard deviation. In this set of experiments, because σ_{max} was set to 2, σ was randomly selected from the range (0, 2]. If σ is high, the covariance among the profiles will be high. In this case, the PCA-based detection algorithm did not yield successful results. For Jester, we observed more successful results, as shown in Table 2. This phenomenon can be explained the larger rating range in Jester. That leads to higher covariance value of the genuine profiles. On the other hand, covariance value of the attack profiles is smaller because similar ratings are used to create attack profiles. Hence, PCA-based detection method successfully determine the attack profiles and separate them from genuine ones.

5.3.2 *Effects of the attack size parameter*

Various sets of experiments were conducted to scrutinize the success of shilling attack detection methods with changing attack size values on private environments. The influence of attack size emphasizes the impact of determining the number of bogus profiles to be inserted into a database as well as the utility perspective of an attack. The attack size establishes a compromise between the detectability and the impact of the applied attack model. Therefore, we performed experiments while varying the attack size from 1 to 15 % with a constant filler size of 25 %. The overall averages of precision and recall with varying attack size values for the Chirita, *kNN* classifier, *k*-means clustering-based, and PCA-based detection schemes are presented in Tables 3 and 4 for MLP and Jester, respectively.

The Chirita algorithm successfully detected shilling attacks with dense attacker profiles but was unsuccessful against attacks with small size and high sparsity (Williams et al. 2007). As shown in Table 3, as the attack size increased, this algorithm was more successful toward attacks, excluding the average attack. Because the RDMA values of the random attack profiles were higher, the most successful precision and recall values were obtained for random attack. By contrast, the average attack profiles had lower RDMA values because of the established methodology and because the Chirita algorithm could not detect these attack profiles. Although the detection performance of the algorithms improved with increasing attack size, they were not successful in detecting shilling profiles. As seen from Table 4, the detection performances of all methods for all attack size values are smaller than the ones for MLP. This phenomenon can be explained the density of Jester data set. As stated by Williams et al. (2007), the Chirita algorithm does not perform well for dense data sets.

As shown in Tables 3 and 4, the precision and recall values usually ranged between 0.8 and 1.0 for the *kNN* classifier for both data sets. The success of the algorithm increased

Table 3 Performance of detection algorithms with varying *attack size* (MLP)

	Precision				Recall			
Attack size	1	3	10	15	1	3	10	15
Chirita								
Random	0.018	0.051	0.157	**0.221**	0.018	0.051	0.157	**0.221**
Average	0.000	0.000	0.000	**0.000**	0.000	0.000	0.000	**0.000**
Bandwagon	0.015	0.049	0.148	**0.209**	0.015	0.049	0.148	**0.209**
Segment	0.013	0.041	0.127	**0.175**	0.013	0.041	0.127	**0.175**
RB	0.014	0.043	0.132	**0.195**	0.014	0.043	0.132	**0.195**
Love/hate	0.017	0.049	0.144	**0.204**	0.017	0.049	0.144	**0.204**
kNN classifier								
Random	0.000	0.750	0.847	**0.872**	0.000	0.706	0.962	**0.974**
Average	**1.000**	0.842	0.895	0.938	0.857	0.941	0.981	**0.987**
Bandwagon	0.000	0.375	**0.850**	0.817	0.000	0.176	0.981	**0.987**
Segment	0.000	0.750	0.872	**0.873**	0.000	0.706	0.788	**0.805**
RB	**1.000**	0.938	0.927	0.962	0.143	0.882	0.981	**0.987**
Love/hate	**1.000**	1.000	0.836	0.884	0.571	0.941	0.981	**0.987**
k-means								
Random	0.095	0.152	**0.248**	0.245	0.883	0.915	0.927	**1.000**
Average	0.116	0.229	**0.357**	0.308	0.874	0.902	0.879	**1.000**
Bandwagon	0.108	0.226	**0.316**	0.260	0.885	0.913	0.937	**1.000**
Segment	0.103	0.179	**0.329**	0.297	0.910	0.920	0.922	**1.000**
RB	0.109	0.243	**0.371**	0.290	1.000	0.999	0.975	**1.000**
Love/hate	0.110	0.212	**0.264**	0.231	1.000	0.991	0.999	**1.000**
PCA								
Random	0.120	0.180	0.290	**0.340**	0.120	0.180	0.290	**0.340**
Average	0.030	0.220	0.560	**0.650**	0.030	0.220	0.560	**0.650**
Bandwagon	0.040	0.050	0.070	**0.090**	0.040	0.050	0.070	**0.090**
Segment	0.110	**0.150**	0.060	0.090	0.110	**0.150**	0.060	0.090
RB	0.078	0.067	0.057	**0.082**	0.078	0.067	0.057	**0.082**
Love/hate	0.006	0.015	0.038	**0.057**	0.006	0.015	0.038	**0.057**

The best outcomes are given in bold

when the attack size was 15 %. The number of attack profiles in the training and test data was not sufficient for stable classification when the attack size was low. Thus, zero precision and recall values were obtained for an attack size of 1 % for random, bandwagon, and segment attacks for MLP, whereas better results were obtained for other attack types. For Jester, the detection performance of the *kNN* classifier method is zero for all attack types. Because the training data set was used in all attack models for this method, it does not matter which attack model is used. As long as there are sufficient training data, this method will be successful. Therefore, the attack size parameter plays an important role in the success of this method. Moreover, since similar training data was used for both data sets, similar results were observed.

As indicated in Tables 3 and 4, there was a direct correlation between attack size and the precision value of the *k*-means clustering-based detection algorithm toward the attacks for both data set. As the attack size increased, the number of attack profiles in the cluster of interest increased, thus improving the precision value. Because the tightest cluster was identified and isolated from the database, many real profiles were omitted from this cluster.

Table 4 Performance of detection algorithms with varying *attack size* (Jester)

	Precision				Recall			
Attack size	1	3	10	15	1	3	10	15
Chirita								
Random	0.015	0.039	0.109	**0.153**	0.015	0.039	0.109	**0.153**
Average	0.000	0.000	0.000	**0.000**	0.000	0.000	0.000	**0.000**
Bandwagon	0.008	0.025	0.074	**0.100**	0.008	0.025	0.074	**0.100**
RB	0.005	0.015	0.040	**0.053**	0.005	0.015	0.040	**0.053**
Love/hate	0.017	0.049	0.137	**0.182**	0.017	0.049	0.137	**0.182**
kNN classifier								
Random	0.000	0.000	**1.000**	0.974	0.000	0.000	0.987	**0.987**
Average	0.000	**1.000**	0.987	0.962	0.000	0.987	0.987	**0.987**
Bandwagon	0.000	0.000	0.550	**0.692**	0.000	0.000	0.165	**0.195**
RB	0.000	1.000	1.000	**1.000**	0.000	0.013	0.325	**0.714**
Love/hate	0.000	0.000	1.000	**1.000**	0.000	0.000	0.987	**0.987**
k-means								
Random	0.051	0.107	0.203	**0.242**	**0.972**	0.972	0.947	0.969
Average	0.051	0.111	0.248	**0.287**	**1.000**	0.996	0.992	0.994
Bandwagon	0.051	0.102	0.188	**0.223**	0.972	1.000	1.000	**1.000**
RB	0.049	0.104	0.204	**0.259**	1.000	1.000	1.000	**1.000**
Love/hate	0.047	0.098	0.205	**0.292**	1.000	1.000	1.000	**1.000**
PCA								
Random	0.528	0.642	0.753	**0.783**	0.528	0.642	0.753	**0.783**
Average	0.473	0.687	0.841	**0.871**	0.473	0.687	0.841	**0.871**
Bandwagon	0.153	0.202	0.231	**0.247**	0.153	0.202	0.231	**0.247**
RB	0.167	0.214	0.243	**0.254**	0.167	0.214	0.243	**0.254**
Love/hate	0.292	0.369	0.409	**0.408**	0.292	0.369	0.408	**0.408**

The best outcomes are given in bold

Because real profiles were omitted from the user-item matrix, the precision value was lower than the recall value. Since the attack profiles were so similar, they were located in the same cluster. Therefore, the recall value was successful in all attack models. As shown in the tables, the recall values varied between approximately 0.9 and 1.0 for all attacks for both data sets. Differences in attack size can only increase the number of attacks in a cluster and thus did not significantly affect the recall value.

As in the case of the application based on the filler size parameter describe above, the application based on the attack size parameter yielded the best results for the average attack for the PCA-based detection scheme, as shown in Tables 3 and 4. Since fake profiles are generated using mean values in the average attack, the covariance is smaller for this attack. This leads to better results. The increase in the attack size value resulted in an increase in the precision and recall values of nearly all of the attack models. For MLP, the precision and recall values of the average attack were as high as 0.65, an acceptable and successful value. Better results were obtained for Jester due to the higher covariance of the genuine profiles. Shilling profiles have lower covariance. Thus, it becomes easy to distinguish fake profiles from genuine ones. This method was unsuccessful for the other attack models because the covariance values of the profiles were higher due to the generating algorithms.

5.3.3 *Effects of the β_{max} parameter*

We performed another set of experiments to illustrate how changing the values of β_{max} affects the success of detection schemes. We fixed the filler size, attack size, and σ_{max} at 25, 15, and 2 %, respectively, while β_{max} parameter was varied from 5 to 25 %. The overall averages of the precision and recall values for the four detection algorithms on six attack models are presented in Tables 5 and 6 for MLP and Jester, respectively.

As shown in Tables 5 and 6, in general, the success of the detection algorithms decreased with increasing β_{max} values. The only exception was the recall values for the *k*-means clustering-based detection method. The recall values for this algorithm slightly improved with increasing β_{max} values due to increasing similarity among shilling profiles. Although the detection performance of the algorithms decreased with increasing β_{max} values, the outcomes remained close to each other for both data sets. The value of β_{max} had a greater effect on the Chirita algorithm than the other algorithms. The best outcomes for precision and recall were observed for the Chirita and *kNN* classifier algorithms for a β_{max} value of 5. For the *k*-means clustering-based method, $\beta_{max} = 25$ provided the best recall values. However, the best precision values differed for the same algorithm with varying β_{max} values. Similar trends in precision and recall were observed for the PCA-based detection scheme. As indicated in Tables 5 and 6, the precision and recall values for varying β_{max} values were closer to each other for the *k*-means clustering-based scheme compared to the other algorithms.

Increasing β_{max} values significantly affected the ratings distribution of genuine user ratings profiles due to the sparse nature of such profiles. In this case, it might become difficult to differentiate fake profiles from genuine profiles because the filled profiles will become more similar to each other. Creating fake profiles by filling some filler items does not change the general rating distribution of the filled profiles. The detection schemes, in general, then will have difficulties detecting bogus profiles due the increase in β_{max} values. For example, in the Chirita algorithm, increasing β_{max} increases the number of random values inserted into genuine profiles. These noise data increase RDMA values due to larger dispersion. As stated by Chirita et al. (2005), due to the high standard deviation among the

Table 5 Performance of detection algorithms with varying β_{max} (MLP)

	Precision				Recall			
β_{max}	5	10	15	25	5	10	15	25
Chirita								
Random	**0.516**	0.448	0.314	0.221	**0.516**	0.448	0.314	0.221
Average	**0.000**	0.000	0.000	0.000	**0.000**	0.000	0.000	0.000
Bandwagon	**0.514**	0.440	0.370	0.209	**0.514**	0.440	0.370	0.209
Segment	**0.387**	0.349	0.288	0.175	**0.387**	0.349	0.288	0.175
RB	**0.516**	0.446	0.381	0.195	**0.516**	0.446	0.381	0.195
Love/hate	**0.515**	0.436	0.379	0.204	**0.515**	0.436	0.379	0.204
kNN classifier								
Random	**1.000**	1.000	0.962	0.872	**0.987**	0.987	0.987	0.974
Average	**1.000**	1.000	0.962	0.938	**0.987**	0.987	0.987	0.987
Bandwagon	**1.000**	1.000	0.950	0.817	**0.987**	0.987	0.987	0.987
Segment	**1.000**	1.000	0.960	0.873	**0.987**	0.987	0.935	0.805
RB	**1.000**	1.000	0.836	0.962	**0.987**	0.974	0.987	0.987
Love/hate	**1.000**	1.000	0.641	0.884	**0.987**	0.987	0.993	0.987
k-means								
Random	0.189	0.263	**0.378**	0.245	0.892	0.962	0.946	**1.000**
Average	**0.421**	0.351	0.378	0.308	0.993	0.993	0.993	**1.000**
Bandwagon	0.443	0.382	**0.502**	0.260	0.993	0.994	0.993	**1.000**
Segment	0.214	**0.333**	0.177	0.297	0.900	0.986	0.957	**1.000**
RB	0.424	0.420	**0.523**	0.290	0.987	0.993	0.993	**1.000**
Love/hate	0.213	0.268	**0.258**	0.231	0.982	0.992	0.989	**1.000**
PCA								
Random	0.130	0.135	0.123	**0.340**	0.130	0.135	0.123	**0.340**
Average	**0.761**	0.754	0.750	0.650	**0.761**	0.754	0.750	0.650
Bandwagon	0.089	**0.091**	0.088	0.090	0.089	**0.091**	0.088	0.090
Segment	**0.095**	0.088	0.090	0.090	**0.095**	0.088	0.090	0.090
RB	0.089	**0.090**	0.084	0.082	0.089	**0.090**	0.084	0.082
Love/hate	**0.126**	0.115	0.100	0.057	**0.126**	0.115	0.100	0.057

The best outcomes are given in bold

rating dispersions in attack profiles, the RDMA attribute value will be higher for attack profiles than for real profiles. Larger RDMA values make it difficult to detect fake profiles using the Chirita algorithm. Hence, the detection rate of the Chirita algorithm diminishes with increasing β_{max}.

Noise data is inserted into genuine profiles for privacy protection. Moreover, shilling profiles are created using noise data. This leads to very similar results for *k*-means and PCA-based detection algorithms for both data sets, as seen from Tables 5 and 6. Therefore, it is difficult to determine the optimum values of β_{max} for *k*-means and PCA-based detection algorithms for both data sets.

Table 6 Performance of detection algorithms with varying β_{max} (Jester)

	Precision				Recall			
β_{max}	5	10	15	25	5	10	15	25
Chirita								
Random	**0.189**	0.185	0.172	0.153	**0.189**	0.185	0.172	0.153
Average	**0.000**	0.000	0.000	0.000	**0.000**	0.000	0.000	0.000
Bandwagon	**0.129**	0.122	0.120	0.100	**0.129**	0.122	0.120	0.100
RB	**0.163**	0.153	0.145	0.053	**0.163**	0.153	0.145	0.053
Love/hate	**0.205**	0.195	0.192	0.182	**0.205**	0.195	0.192	0.182
kNN classifier								
Random	**1.000**	0.962	0.974	0.974	**0.987**	0.987	0.981	0.987
Average	**1.000**	0.987	0.962	0.962	**0.987**	0.987	0.987	0.987
Bandwagon	**0.925**	0.607	0.781	0.652	**0.961**	0.221	0.325	0.195
RB	0.926	0.952	**0.961**	1.000	**0.325**	0.441	0.961	0.714
Love/hate	**1.000**	1.000	0.987	1.000	**0.987**	0.987	0.987	0.987
k-means								
Random	0.241	**0.260**	0.234	0.228	0.816	0.876	0.746	**1.000**
Average	0.362	0.356	**0.366**	0.257	0.981	0.986	0.984	**1.000**
Bandwagon	**0.357**	0.318	0.320	0.204	1.000	1.000	1.000	**1.000**
RB	0.292	**0.306**	0.303	0.240	1.000	1.000	1.000	**1.000**
Love/hate	0.285	0.271	**0.292**	0.266	0.999	1.000	1.000	**1.000**
PCA								
Random	0.776	0.775	**0.786**	0.783	0.776	0.775	**0.786**	0.783
Average	0.851	0.852	0.869	**0.871**	0.851	0.852	0.869	**0.871**
Bandwagon	0.239	0.243	**0.248**	0.247	0.239	0.243	**0.248**	0.247
RB	0.252	0.242	0.247	**0.254**	0.252	0.242	0.247	**0.254**
Love/hate	0.394	0.397	0.408	**0.408**	0.394	0.397	0.408	**0.408**

The best outcomes are given in bold

5.3.4 Effects of the σ_{max} parameter

To illustrate how the detection methods perform with varying values of σ_{max}, we performed another set of experiments. We fixed filler size, attack size, and β_{max} at 25, 15, and 25 %, respectively, while varying the value of σ_{max} from 0.5 to 2. The overall averages of precision and recall values for all detection methods on the six attack models after running the trials 100 times are presented in Tables 7 and 8 for MLP and Jester, respectively.

The results in Tables 7 and 8 show that the performance of the Chirita algorithm with respect to both precision and recall improved with increasing σ_{max}. Smaller σ_{max} values resulted in smaller RDMA values. Smaller RDMA values increase the difficulty of detecting attack profiles using the Chirita algorithm. Therefore, with decreasing σ_{max} values, the RDMA values become smaller; and the performance of the Chirita algorithm consequently decreases.

In contrast to the Chirita algorithm, in general, the performance of the PCA-based detection scheme with respect to both precision and recall increased with decreasing σ_{max}

Table 7 Performance of detection algorithms with varying σ_{max} (MLP)

	Precision				Recall			
σ_{max}	0.25	0.50	1.00	2.00	0.25	0.50	1.00	2.00
Chirita								
Random	0.000	0.000	0.014	**0.221**	0.000	0.000	0.014	**0.221**
Average	0.000	0.000	0.000	**0.000**	0.000	0.000	0.000	**0.000**
Bandwagon	0.000	0.000	0.012	**0.209**	0.000	0.000	0.012	**0.209**
Segment	0.000	0.000	0.087	**0.175**	0.000	0.000	0.087	**0.175**
RB	0.000	0.000	0.012	**0.195**	0.000	0.000	0.012	**0.195**
Love/hate	0.000	0.000	0.013	**0.204**	0.000	0.000	0.013	**0.204**
kNN classifier								
Random	**1.000**	0.800	0.750	0.872	0.941	0.941	0.857	**0.974**
Average	**1.000**	0.842	0.750	0.938	0.941	0.941	0.857	**0.987**
Bandwagon	**1.000**	0.833	0.714	0.817	0.941	0.882	0.714	**0.987**
Segment	**1.000**	0.790	0.750	0.873	**0.941**	0.882	0.857	0.805
RB	0.941	0.857	0.714	**0.962**	0.941	0.857	0.714	**0.987**
Love/hate	**0.889**	0.857	0.667	0.884	0.941	0.857	0.857	0.987
k-means								
Random	**0.470**	0.436	0.403	0.245	0.992	0.980	0.970	**1.000**
Average	**0.532**	0.458	0.387	0.308	0.993	0.990	0.980	**1.000**
Bandwagon	**0.485**	0.451	0.404	0.260	0.992	0.982	0.960	**1.000**
Segment	**0.499**	0.412	0.352	0.297	0.993	0.980	0.960	**1.000**
RB	**0.467**	0.419	0.384	0.290	0.993	0.990	0.970	**1.000**
Love/hate	**0.485**	0.425	0.350	0.231	0.990	0.990	0.980	**1.000**
PCA								
Random	**0.853**	0.705	0.454	0.340	**0.853**	0.705	0.454	0.340
Average	0.591	0.656	**0.750**	0.650	0.591	0.656	**0.750**	0.650
Bandwagon	**0.498**	0.257	0.139	0.090	**0.498**	0.257	0.139	0.090
Segment	**0.839**	0.668	0.388	0.090	**0.839**	0.668	0.388	0.090
RB	**0.682**	0.257	0.117	0.082	**0.682**	0.257	0.117	0.082
Love/hate	**0.066**	0.062	0.061	0.057	**0.066**	0.062	0.061	0.057

The best outcomes are given in bold

values for both data sets. As σ_{max} increases, the covariance value among profiles also increases; therefore, the PCA-based variable selection-based detection method may not be able to detect these attack profiles. In studies by Mehta (2007), Mehta et al. (2007), and Mehta and Nejdl (2009), the authors state that attack profiles are expected to have lower covariance values than real profiles because the filler items set is generally completed with the item mean or system overall mean when creating attack profiles. Hence, the success of the PCA-based detection algorithm can be improved if smaller σ_{max} values are used.

The *kNN* classifier and *k*-means clustering-based methods behaved similarly with varying σ_{max} values. The best precision values were observed when σ_{max} was 0.25 for both detection algorithms for both data sets. In contrast to the precision values, both schemes produced the most promising outcomes with respect to recall when σ_{max} was 2. Although

Table 8 Performance of detection algorithms with varying σ_{max} (Jester)

	Precision				Recall			
σ_{max}	0.25	0.50	1.00	2.00	0.25	0.50	1.00	2.00
Chirita								
Random	0.000	0.000	0.018	**0.153**	0.000	0.000	0.018	**0.153**
Average	0.000	0.000	0.000	**0.000**	0.000	0.000	0.000	**0.000**
Bandwagon	0.000	0.000	0.001	**0.100**	0.000	0.000	0.001	**0.100**
RB	0.000	0.000	0.002	**0.053**	0.000	0.000	0.002	**0.053**
Love/hate	0.000	0.000	0.032	**0.182**	0.000	0.000	0.032	**0.182**
kNN classifier								
Random	**1.000**	1.000	0.985	0.974	0.987	0.987	0.987	**0.987**
Average	**1.000**	1.000	0.938	0.962	0.987	0.987	0.987	**0.987**
Bandwagon	**1.000**	0.974	0.916	0.894	0.987	0.987	0.987	**0.987**
RB	0.962	0.938	0.950	**1.000**	0.987	0.987	0.987	**0.987**
Love/hate	0.987	1.000	0.987	**1.000**	0.987	0.987	0.987	**0.987**
k-means								
Random	**0.411**	0.333	0.330	0.242	0.730	**0.895**	0.834	0.763
Average	**0.465**	0.351	0.415	0.373	0.820	0.975	0.969	**0.983**
Bandwagon	**0.503**	0.403	0.359	0.354	0.975	1.000	1.000	**1.000**
RB	**0.483**	0.424	0.363	0.302	1.000	1.000	1.000	**1.000**
Love/hate	**0.461**	0.323	0.306	0.310	0.971	**0.984**	0.978	0.981
PCA								
Random	**0.999**	0.975	0.914	0.772	**0.999**	0.975	0.914	0.772
Average	**0.939**	0.939	0.921	0.861	**0.939**	0.939	0.921	0.861
Bandwagon	**0.963**	0.848	0.521	0.243	**0.963**	0.848	0.521	0.243
RB	**0.968**	0.858	0.550	0.251	**0.968**	0.858	0.550	0.251
Love/hate	**0.986**	0.937	0.751	0.406	**0.986**	0.937	0.751	0.406

The best outcomes are given in bold

the recall values seemed to decrease with decreasing σ_{max} values for MLP, this change was very stable, especially for *k*-means clustering-based method. The recall values for Jester are almost the same for both methods. Smaller σ_{max} values decrease the amount of noise data inserted into profiles during data disguising. The reduced amount of noise does not weaken the effects of filler items during fake profile generation and facilitates the successful retrieval of fake profiles by *k*-means clustering and the *kNN* classifier.

6 Discussion

An examination of the empirical results presented in the tables above revealed that the *kNN* classifier method is the most successful method for all attack models. The disguise operation in private environments does not have a significant effect on detection algorithm performance. The *kNN* classifier detection algorithm calculates a number of generic and model-specific attribute values for each profile, creates a new data table, and performs classifications using this new data table. The *kNN* classifier detection algorithm divides this attribute table into two groups under the headings of training and test data. Because it creates a model using training data, data masking does not have significant effects for the *kNN* classifier. The use of a training set generated from perturbed data enables the creation of a new model to detect PPCF attack profiles in the test set.

The performance of the Chirita algorithm against attacks in private environments was not highly successful. Chirita et al. (2005) stated that due to the high standard deviation among the rating dispersions in attack profiles, the RDMA attribute values of attach profiles will be higher than those of real profiles. The attack profiles generated on the PPCF schemes were filled with random numbers. When the σ_{max} value was small, the RDMA value for the attack profiles was small, and the Chirita algorithm could not detect these PPCF attacks successfully. We hypothesized that at higher σ_{max} values, the Chirita algorithm may become more successful. This hypothesis was verified by the empirical outcomes presented in Tables 7 and 8. Larger σ_{max} values significantly enhanced the performance of the Chirita algorithm. Because the item mean was used to fill the profiles in the average attack, the RDMA value was low even when larger σ_{max} values were used. In this case, as shown in Tables 7 and 8, the Chirita algorithm will be unsuccessful for the average attack.

Although the precision value of the *k*-means algorithm was not very good, the recall value was highly successful. Our *k*-means algorithm performs clustering by considering the similarities between profiles. A certain profile is created for each attack model in non-private and private environments. Therefore, these profiles are similar to each other and are expected to be dispersed in the same cluster. Moreover, shilling profiles mimic real profiles to effectively manipulate the outcomes. Consequently, in *k*-means clusters, many real profiles may cluster together with the attack profiles and be omitted form the database upon isolation of the defined attack clusters, reducing the precision.

The PCA-based detection method was successful for the average attack only when σ_{max} was 2 due to smaller covariance values among the profiles because the filler items set was completed with the item mean when creating the average attack profiles. In other attack models, the filler items specified in the profiles are completed with random numbers generated with a certain σ_{max} value. If the σ_{max} values are high, the covariance value among profiles also becomes high, and the PCA algorithm may not be able to detect these attack profiles. Consequently, the success of the PCA-based detection algorithm might be improved using smaller σ_{max} values. It performs better for Jester due to higher covariance of the fake profiles.

We compared the detection methods used for PPCF with those used in CF schemes under the same conditions. In Table 9, the precision and recall values are compared for corresponding algorithms in non-private and private environments. The results of the experiments conducted for CF schemes were compiled from related studies, where MLP was used. The comparison with the results of the study by Burke et al. (2006) of the Chirita algorithm reveals a similar precision value as for our results; however, the CF algorithm is more successful for the recall value. Our results for the *kNN* classifier algorithm and those of Mobasher et al. (2007) differ but are similar at a higher filler size and attack size. Bhaumik et al. (2011) utilized the *k*-means algorithm on CF in a different manner than in the present study. They defined generic attribute values and performed the cluster process using these values. A comparison of the results indicates that their results were more successful than ours. The precision and recall values achieved for the PCA method in the study by Mehta and Nejdl (2009) are considerably higher than the values we obtained because we selected higher σ_{max} values to generate attack profiles.

Table 9 Comparison of detection algorithms in non-private and private environments

	Non-private environment				Private environment			
Algorithm	Chirita	kNN	*k*-means	PCA	Chirita	kNN	k-means	PCA
Precision								
Random	0.020	0.350	0.980	0.960	0.018	0.000	0.095	0.120
Average	0.020	0.330	0.920	0.900	0.000	1.000	0.116	0.030
Bandwagon	0.020	0.350	0.900	0.960	0.015	0.000	0.108	0.040
Segment	0.050	0.280	0.980	–	0.013	0.000	0.103	0.110
RB	–	–	–	–	0.014	1.000	0.109	0.093
Love/hate	0.010	0.350	–	–	0.017	1.000	0.110	0.058
Recall								
Random	0.680	1.000	1.000	1.000	0.018	0.000	0.883	0.120
Average	0.620	1.000	1.000	1.000	0.000	0.857	0.874	0.030
Bandwagon	0.650	1.000	1.000	1.000	0.015	0.000	0.885	0.040
Segment	0.650	0.920	1.000	–	0.013	0.000	0.910	0.110
RB	–	–	–	–	0.014	0.143	1.000	0.078
Love/hate	0.670	1.000	–	–	0.017	0.571	1.000	0.006

We finally compared the precision and recall of the four detection algorithms with those of the algorithm proposed by Gunes and Polat (2015a) under the same conditions. The profiles in a perturbed user-item matrix are clustered using hierarchical clustering, and the cluster that probably contain fake profiles is identified as the attacked cluster. The authors also scrutinized the ratings of target items to enhance the performance of their scheme. They slightly improved their method by investigating the ratings of target items. The results of the four detection methods presented in this study and the hierarchical clustering-based method are compared in Table 10.

As shown in Table 10, the *kNN* classifier performs better than the hierarchical clustering-based method in terms of precision for the random, average, and love/hate attack models. However, the hierarchical clustering-based scheme provides more promising results than our methods with respect to precision for the bandwagon, segment, and reverse bandwagon attack models. In terms of recall, our *k*-means clustering-based scheme provides the best outcomes. The hierarchical clustering-based method performs very similar to our *k*-means clustering-based method for average, bandwagon, segment, and reverse bandwagon attacks. All of our algorithms achieve better outcomes than the hierarchical clustering-based method for the random attack model.

7 Conclusions and future work

Detecting shilling profiles is important in privacy-preserving collaborative filtering methods. We modified four widely used detection algorithms, proposed for detecting shilling profiles in non-private environments, in such a way to determine fake profiles created using six shilling attacks in private environments. We compared the modified methods with their correspondence for non-private environments. Also, we compared them

Table 10 Comparison of detection algorithms for PPCF

Algorithm	Chirita	kNN	*k*-means	PCA	Hierarchical
Precision					
Random	0.221	**0.872**	0.245	0.340	0.003
Average	0.000	**0.938**	0.308	0.650	0.918
Bandwagon	0.209	0.817	0.260	0.090	**1.000**
Segment	0.175	0.873	0.297	0.090	**1.000**
RB	0.195	0.962	0.290	0.082	**1.000**
Love/hate	0.204	**0.884**	0.231	0.057	0.640
Recall					
Random	0.221	0.974	**1.000**	0.340	0.002
Average	0.000	0.987	**1.000**	0.650	0.830
Bandwagon	0.209	0.987	**1.000**	0.090	0.999
Segment	0.175	0.805	**1.000**	0.090	0.967
RB	0.195	0.987	**1.000**	0.082	1.000
Love/hate	0.204	0.987	**1.000**	0.057	0.329

The best outcomes are given in bold

with the hierarchical clustering-based detection method, which is proposed for private environments. We evaluated the schemes in terms of precision and recall by conducting experiments on two real data sets.

Our key findings can be summarized as follows:

1. The most successful detection methods are the *kNN* classifier and *k*-means methods. However, the *kNN* classifier requires a training data set. The *k*-means method might isolate a great number of real profiles, which will negatively affect system accuracy.
2. The PCA algorithm performs better for the data set including ratings with larger range. It classifies profiles according to the covariance value. When the rating range is larger, the covariance value of genuine profiles becomes larger. Shilling profiles have smaller covariance value. Thus, PCA then can successfully differentiate them.
3. Although the *kNN* classifier requires a training set for detection, it might be considered the best algorithm of the presented detection algorithms for both non-private and private environments according to the empirical outcomes.
4. The success of the Chirita algorithm is generally low compared to other algorithms, particularly for attacks with a smaller attack size.
5. With increasing filler size values, in general, the detection performance of the detection method for both data sets decreases.
6. The methods, in general, perform better for larger attack size values due to increasing number of attacks.
7. Increasing β_{max} values negatively affect the performance of the algorithms.
8. Smaller σ_{max} values improve the detection performance of all algorithms except the Chirita method due to smaller randomness.

We are planning to develop new detection methods to reduce the disadvantages of these algorithms. We also want to investigate how to further improve the success of the existing detection algorithms. In addition to numeric ratings-based recommendation schemes, there are binary ratings-based prediction algorithms. Therefore, we are planning to develop detection algorithms that can filter out binary ratings-based shilling profiles. Another important future research is to study how shilling attacks affect the top-*N* recommendation lists. In other words, we are planning to scrutinize how shilling attacks change the position or rank of the targeted items in top-*N* recommendation lists.

Acknowledgments This work was supported by Grant 108E221 and 108E221 from TUBITAK.

References

Bhaumik, R., Mobasher, B., & Burke, R. D. (2011). A clustering approach to unsupervised attack detection in collaborative recommender systems. In *Proceedings of the 7th IEEE international conference on data mining*, (pp. 181–187). Las Vegas, NV, USA.

Bhaumik, R., Williams, C. A., Mobasher, B., & Burke, R. D. (2006). Securing collaborative filtering against malicious attacks through anomaly detection. In *Proceedings of the 4th workshop on ıntelligent techniques for web personalization*, Boston, MA, USA.

Bilge, A., Gunes, I., & Polat, H. (2013b). A robust privacy-preserving recommendation algorithm, *The 2nd Asian conference on ınformation systems*, Phuket, Thailand.

Bilge, A., Gunes, I., & Polat, H. (2014a). Robustness analysis of privacy-preserving model-based recommendation schemes. *Expert Systems with Applications, 41*(8), 3671–3681.

Bilge, A., Kaleli, C., Yakut, I., Gunes, I., & Polat, H. (2013a). A survey of privacy-preserving collaborative filtering schemes. *International Journal of Software Engineering and Knowledge Engineering, 23*(8), 1085–1108.

Bilge, A., Ozdemir, Z., & Polat, H. (2014b). A novel shilling attack detection method. *Procedia Computer Science, 31*, 165–174.

Bilge, A., & Polat, H. (2013a). A comparison of clustering-based privacy-preserving collaborative filtering schemes. *Applied Soft Computing, 13*(5), 2478–2489.

Bilge, A. & Polat, H. (2013b). Robustness of bisecting *k*-means clustering-based collaborative filtering algorithm. In *Proceedings of the 5th international conference on advances in information mining and management*, (pp. 7–13). Brussels, Belgium.

Bobadilla, J., Ortega, F., Hernando, A., & Gutiérrez, A. (2013). Recommender systems survey. *Knowledge-Based Systems, 46*, 109–132.

Burke, R. D., Mobasher, B., Williams, C. A, & Bhaumik, R. (2006). Classification features for attack detection in collaborative recommender systems. In *Proceedings of the 12th ACM SIGKDD international conference on knowledge discovery and data mining,* (pp. 542–547). Philadelphia, PA, USA.

Cao, J., Wu, Z., Mao, B., & Zhang, Y. (2013). Shilling attack detection utilizing semi-supervised learning method for collaborative recommender system. *World Wide Web, 16*(5–6), 729–748.

Casino, F., Domingo-Ferrer, J., Patsakis, C., Puig, D., & Solanas, A. (2015). A *k*-anonymous approach to privacy preserving collaborative filtering. *Journal of Computer and System Sciences, 81*(6), 1000–1011.

Casino, F., Patsakis, C., Puig, D., & Solanas, A. (2013). On privacy preserving collaborative filtering: Current trends, open problems, and new issues. In *Proceedings of the IEEE 10th international conference on e-business engineering*, (pp. 244–249). Coventry, United Kingdom.

Chirita, P. A., Nejdl, W., & Zamfir, C. (2005). Preventing shilling attacks in online recommender systems. In *Proceedings of the 7th annual ACM international workshop on web information and data management*, (pp. 67–74). Bremen, Germany.

Chung, C.-Y., Hsu, P.-Y., & Huang, S.-H. (2013). *βP*: A novel approach to filter out malicious rating profiles from recommender systems. *Decision Support Systems, 55*(1), 314–325.

Gunes, I., Bilge, A., Kaleli, C., & Polat, H. (2013a). Shilling attacks against privacy-preserving collaborative filtering. *Journal of Advanced Management Science, 1*(1), 54–60.

Gunes, I., Bilge, A., & Polat, H. (2013b). Shilling attacks against memory-based privacy-preserving recommendation algorithms. *KSII Transactions on Internet and Information Systems, 7*(5), 1272–1290.

Gunes, I., Kaleli, C., Bilge, A., & Polat, H. (2014). Shilling attacks against recommender systems: A comprehensive survey. *Artificial Intelligence Review, 42*(4), 767–799.

Gunes, I. & Polat, H. (2015a). Hierarchical clustering-based shilling attack detection in private environments. In *Proceedings of the 3rd international symposium on digital forensics and security*, (pp. 1–7). Ankara, Turkey.

Gunes, I., & Polat, H. (2015b). Robustness analysis of privacy-preserving hybrid recommendation algorithm. *International Journal of Information Security Science, 4*(1), 13–25.

Hurley, N. J., Cheng, Z., & Zhang, M. (2009). Statistical attack detection. In *Proceedings of the 3rd ACM conference on recommender systems*, (pp. 149–156). New York, NY, USA.

Jeckmans, A. J. P., Beye, M. R. T., Erkin, Z., Hartel, P. H., Lagendijk, R. L., & Tang, Q. (2013). Privacy in recommender systems. *Social media retrieval. Computer communications and networks* (pp. 263–281). London: Springer.

Jia, C.-X., & Liu, R.-R. (2015). Improve the algorithmic performance of collaborative filtering by using the interevent time distribution of human behaviors. *Physica A: Statistical Mechanics and its Applications, 436*, 236–245.

Li, M. (2014). Shilling attack detection algorithm based on non-random-missing mechanism. *International Journal of Security and Its Applications, 8*(6), 115–126.

Li, C. & Luo, Z. (2011). Detection of shilling attacks in collaborative filtering recommender systems. In *Proceedings of the international conference of soft computing and pattern recognition*, (pp. 190–193). Dalian, China.

Mehta, B. (2007). Unsupervised shilling detection for collaborative filtering. In *Proceedings of the 22nd national conference on artificial intelligence* (Vol. 2, pp. 1402–1407). Vancouver, Canada.

Mehta, B., Hofmann, T., & Fankhauser, P. (2007). Lies and propaganda: Detecting spam users in collaborative filtering. In *Proceedings of the 12th international conference on intelligent user interfaces*, (pp. 14–21). Honolulu, HI, USA.

Mehta, B., & Nejdl, W. (2009). Unsupervised strategies for shilling detection and robust collaborative filtering. *User Modeling and User-Adapted Interaction, 19*(1–2), 65–97.

Mobasher, B., Burke, R. D., Bhaumik, R., & Williams, C. A. (2007). Toward trustworthy recommender systems: An analysis of attack models and algorithm robustness. *ACM Transactions Internet Technology, 7*(4), 23.

Mobasher, B., Burke, R. D., Williams, C. A., & Bhaumik, R. (2006). Analysis and detection of segment-focused attacks against collaborative recommendation. *Lecture Notes in Computer Science, 4198*, 96–118.

Morid, A. M., Shajari, M., & Hashemi, A. R. (2013). Defending recommender systems by influence analysis. *Information Retrieval, 17*(2), 137–152.

Noh, G., Kang, Y.-M., Oh, H., & Kim, C.-K. (2014). Robust sybil attack defense with information level in online recommender systems. *Expert Systems with Applications, 41*(4 Part 2), 1781–1791.

O'Mahony, M. P., Hurley, N. J., Kushmerick, N., & Silvestre, G. C. M. (2004). Collaborative recommendation: A robustness analysis. *ACM Transactions Internet Technology, 4*(4), 344–377.

O'Mahony, M. P., Hurley, N. J., & Silvestre, G. C. M. (2006). Detecting noise in recommender system databases. In *Proceedings of the 11th international conference on intelligent user interfaces*, (pp. 109–115). Sydney, Australia.

Ozturk, A., & Polat, H. (2015). From existing trends to future trends in privacy-preserving collaborative filtering. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, 5*(6), 276–291.

Su, X. -F., Zeng, H. -J., & Chen, Z. (2005). Finding group shilling in recommendation system. In *Proceedings of the 14th international conference on world wide web*, (pp. 960–961). Chiba, Japan.

Tang, T. & Tang, Y. (2011). An effective recommender attack detection method based on time SFM factors. In *Proceedings of the 3rd international conference on communication software and networks*, (pp. 78–81). Xi'an, China.

Williams, C. A., Mobasher, B., & Burke, R. D. (2007). Defending recommender systems: Detection of profile injection attacks. *Service Oriented Computing and Applications, 1*(3), 157–170.

Xia, H., Fang, B., Gao, M., Ma, H., Tang, Y., & Wen, J. (2015). A novel item anomaly detection against shilling attacks in collaborative recommendation systems using the dynamic time interval segmentation technique. *Information Sciences, 306*, 150–165.

Yurekli, B. Y., & Kaleli, C. (2016). Robustness analysis of arbitrarily distributed data-based recommendation methods. *Expert Systems with Applications, 44*, 217–229.

Zhang, F.-G. (2011). Preventing recommendation attack in trust-based recommender systems. *Journal of Computer Science and Technology, 26*(5), 823–828.

Zhang, S., Chakrabarti, A., Ford, J., & Makedon, F. (2006a). Attack detection in time series for recommender systems. In *Proceedings of the 12th ACM SIGKDD international conference on knowledge discovery and data mining*, (pp. 809–814). Philadelphia, PA, USA.

Zhang, X.-L., Lee, T., & Pitsilis, G. (2013). Securing recommender systems against shilling attacks using social-based clustering. *Journal of Computer Science and Technology, 28*(4), 616–624.

Zhang, Q., Luo, Y., Weng, C., & Li, M. (2009). A trust-based detecting mechanism against profile injection attacks in recommender systems. In *Proceedings of the 3rd IEEE international conference on secure software integration and reliability improvement*, (pp. 59–64). Shanghai, China.

Zhang, S., Ouyang, Y., Ford, J., & Makedon, F. (2006b). Analysis of a low-dimensional linear model under recommendation attacks. In *Proceedings of the 29th annual international ACM SIGIR conference on research and development in information retrieval*, (pp. 517–524). Seattle, WA, USA.

Zhang, F., & Zhou, Q. (2014). HHT-SVM: An online method for detecting profile injection attacks in collaborative recommender systems. *Knowledge-Based Systems, 65*, 96–105.

Zhang, F., & Zhou, Q. (2015). Ensemble detection model for profile injection attacks in collaborative recommender systems based on BP neural network. *IET Information Security, 9*(1), 24–31.

Zhou, W., Koh, Y. S., Wen, J., Alam, S., & Dobbie, G. (2014a). Detection of abnormal profiles on group attacks in recommender systems. In *Proceedings of the 37th international ACM SIGIR conference on research and development in information retrieval*, (pp. 955–958). Gold Coast, Australia.

Zhou, W., Wen, J., Koh, Y. S., Alam, S., & Dobbie, G. (2014b). Attack detection in recommender systems based on target item analysis. In *Proceedings of the 2014 international joint conference on neural network*, (pp. 332–339). Beijing, China.

Zhuo, Z. & Kulkarni, S. R. (2014). Detection of shilling attacks in recommender systems via spectral clustering. In *Proceedings of the 17th international conference on information fusion*, (pp. 1–8). Salamanca, Spain.