IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

SPECIAL SECTION ON ADVANCED SOFTWARE AND DATA ENGINEERING FOR SECURE SOCIETIES

Received February 27, 2019, accepted March 7, 2019, date of publication March 18, 2019, date of current version April 12, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2905862

Robust Model-Based Reliability Approach to Tackle Shilling Attacks in Collaborative Filtering Recommender Systems

SANTIAGO ALONSO[1], JESÚS BOBADILLA[1], FERNANDO ORTEGA[1], AND RICARDO MOYA[2]
[1]Departamento Sistemas Informáticos, ETSI de Sistemas Informáticos, Universidad Politécnica de Madrid, 28031 Madrid, Spain
[2]Telefónica Investigación y Desarrollo, 28050 Madrid, Spain

Corresponding author: Jesús Bobadilla (jesus.bobadilla@upm.es)

ABSTRACT As the use of recommender systems becomes generalized in society, the interest in varying the orientation of their recommendations is increasing. There are shilling attacks' strategies that introduce malicious profiles in collaborative filtering recommender systems in order to promote the own products or services or to discredit those of the competition. Academic research against shilling attacks has been focused in statistical approaches to detect the unusual patterns in user ratings. Nowadays, there is a growing research area focused on the design of robust machine learning methods to neutralize the malicious profiles inserted into the system. This paper proposes an innovative robust method, based on matrix factorization, to neutralize the shilling attacks. Our method obtains the reliability value associated with each prediction of a user to an item. By monitoring the unusual reliability variations in the items prediction, we can avoid promoting the shilling predictions to the erroneous recommendations. This paper openly provides more than 13 000 individual experiments involving a wide range of attack strategies, both push, and nuke, in order to test the proposed approach. The results show that the proposed method is able to neutralize most of the existing attacks; its performance only decreases in the not relevant situations: when the attack size is not large enough to effectively affect the recommendations provided by the system.

INDEX TERMS Recommender systems, shilling attacks, collaborative filtering, reliability, malicious profiles, matrix factorization.

I. INTRODUCTION

Recommender Systems (RS) [44] have been extensively used to deal with the overload information problem in Internet. Currently, there exists a variety of scopes where humans can choose from a large amount of options: film to watch, book to read, product to buy, service to use, song to listen, touristic spot to visit, restaurant to have dinner, news to read, etc. RS research has focused on providing accurate recommendations, and currently some other aims beyond accuracy have emerged: novelty, diversity, reliability, etc. RS are a powerful tool for personalization and they support some of the brand-new artificial intelligence services that learns from the user actions to adapt interfaces or to provide customized options to users. As a result of the success of RS, there are currently two trends that converge: a) The increasing dependence of users on these systems to make their decisions, and b) The enormous interest of the companies to obtain an advantageous position in the recommendations that RS users receive. A significant variation in the sense of recommendations about a selected set of items (news, products, etc.) can cause, for example, significant variations in the profits of companies or in the perception of public opinion about political issues.

Due to the economic, social and political importance of the recommendations received by users, there is a growing interest to influence them. While part of the efforts to influence recommendations may be lawful (positive publicity towards users, enrichment of the offered information on the products and services), there are other methods that are not legal or enriching, and whose objective is to get the modification of the recommendations by making use of tricking techniques. The most relevant attacks that RS can receive are a) *Hacking*, and b) *Profile injection attacks*. Whereas hacking attacks entirely falls within the scope of the systems security field and we do not cover it in this

The associate editor coordinating the review of this manuscript and approving it for publication was Raúl Lara-Cabrera.

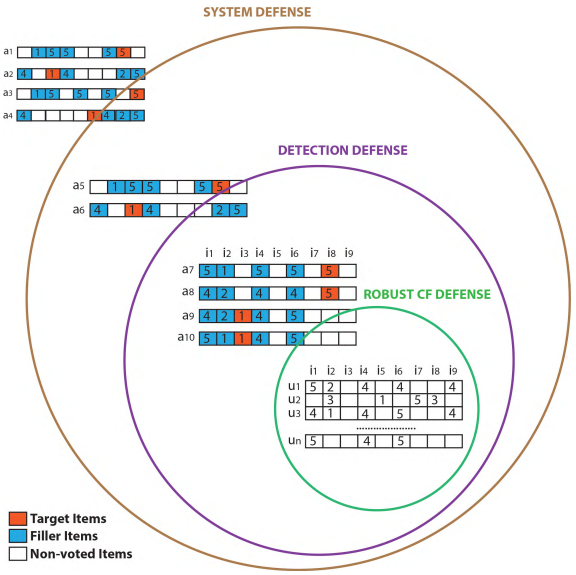

FIGURE 1. Shilling attacks and the three defense barriers to control de profile injections: retaining wall, detection defense and the CF RS machine learning designs to minimize the attack impact.

paper, profiling attacks can be handled from the machine learning area and we provide an innovative approach to tackle them.

RS operation is based on filtering. Mainly, we can find the following types of filtering [44]: a) *Content-based* [48], where products and services (items) recommendations are based on the similarity of contents (to recommend a historical novel to a user that has previously read some historical novels), b) *Demographic-based* [46], where recommendations to a user are based on the consumed items by users demographically similar (similar age, same gender, geographical proximity), c) *social-based* [47], where RS take as information the followers and followed users graphs, d) *context aware* [49], exploiting geographical proximity. Finally, e) *Collaborative Filtering* (CF) [45] recommends, to each active user, the set of her unknown items that are the best rated ones by the most similar users to each active user. Content, collaborative and their hybrid filtering [50] approaches are the most used by current commercial RS. Content-based filtering can be attacked by means of hacking, but it is not vulnerable to profile injection attacks since items contents (products and services descriptions) are only filled by the RS operators. Conversely, CF is vulnerable to profile injection attacks since user profiles are created by users and updated with each of their implicit (e.g: songs listened to) or explicit ratings (e.g.: explicit votes to movies). In summary: a) RS have a deep impact on economic, social and political issues b) CF is the most used approach to design current RS, and c) CF RS are susceptible to be attacked by means of profile injection techniques. Due to the growing importance of this issue, CF RS researchers are increasingly focusing on the CF shilling attacks field, where the consequences of the profile injection attacks are diminished by using different defence approaches. Figure 1 shows several important concepts about the shilling attacks issue: a) An RS can protect its dataset knowledge by reducing the inclusion of malicious profiles. The first defence barrier is to interpose elements to make difficult the automated inclusion of malicious profiles; e.g.: captcha components or email authentication: this is the contention wall. Figure 1 labels it as ''System defense'', b) The second barrier against profile attacks is to detect them and to remove the malicious detected profiles. Figure 1 labels it as ''Detection defense''. *Detection* [4] is based on statistical methods [4] that search for unusual profiles: unusual rating patterns for individual users. This is possible because attackers do not know the CF dataset details, and c) Because the first two lines of defence cannot filter all the malicious profiles, it is important to minimize the consequences of the shilling attacks by using a last barrier: labeled ''*Robust CF defense*'' [20] in Figure 1. This shilling attack shield is based on machine learning methods designed to obtain recommendations by reducing the malicious profiles impact. This paper proposes an innovative CF model-based method to minimize shilling attacks impact.

The most internal circle in Figure 1 shows a usual CF dataset: a user-item sparse matrix containing the items ratings (in the interval [1,...,5]); empty cells mean not voted items. In this example we can state that item i_9 should be recommended to user u_n, since users u_1 and u_3 profiles are similar to the u_n one, and both u_1 and u_3 like item i_9. Figure 1 also shows some avoided profiles in barrier one (''System defence'') and some filtered profiles in barrier two (''Detection defence''). Barrier three has to handle some malicious profiles. In this example profiles a_7 and a_8 perform a *push* attack: they try to promote item i_8, whereas a_9 and a_{10} perform a *nuke* attack: they try to harm i_3. In this data-toy, the pushed or nuked red color items are ''target items''. The blue color items are the ''filler items''; filler items and selected items, that we will see later, are selected to maximize the attack impact (e.g.: choosing popular items).

Traditional research on shilling attacks has focused on the detection defense stage. This approach makes sense, since ''prevention is better than cure''. The aim is to detect and to filter malicious profiles before running the RS CF methods. Nevertheless, this approach has its own limitations: a) It cannot filter all the malicious profiles, b) It can wrongly filter genuine profiles, and c) It just uses statistical information coming from ratings rather than highly abstract information coming from the CF machine learning methods. The typical attack strategies are characterized in [5]; our section IV explains in detail some representative profile attacks. Attacks can be categorized attending to the RS dataset required knowledge to perform the attack: *high-knowledge attacks* are harder to detect, since they have similar profiles to the genuine ones. *Low-knowledge attacks* are easy to design, and they are also easy to detect since the malicious profile patterns are different to the genuine ones. As we have seen, shilling attacks can aim to promote items (*push attacks*) or to demote items (*nuke attacks*). Finally, shilling attackers consider the *cost/benefit balance*.

Currently, there is an open research field that tries to minimize shilling attacks impact by designing innovative machine learning approaches. We can divide these CF approaches in: a) *Memory-based*, and b) *Model-based*. Model-based approaches are the more promising ones, since they benefit from the abstract parameter values from CF models: mainly the ''hidden factors'' obtained in the Matrix Factorization (MF) machine learning method. Furthermore, they fit with the current commercial CF RS designs, usually based on MF.

This paper proposes an innovative model-based approach to face shilling attacks. It is based on the ''reliability'' concept, where each prediction and recommendation value are complemented with a prediction reliability or prediction recommendation value. Currently, reliability on CF RS has been obtained by using different strategies: an approach is to get reliability measures based on trust-aware information; Moradi and Ahmadian [43] make a trust network for each user, and then they evaluate each predicted rating quality. Following the hiphotesys ''the more reliable a prediction, the less liable to be wrong'' [41] defines a general reliability measure suitable for any arbitrary CF RS. It also proposes a method for obtaining specific reliability measures. The confidence concept and several algorithms to retrieve confidence are shown in [42]. Prediction uncertainty can be used to improve product ranking recommendations [13] by retrieving confidence values from predictions and memory-based data. In order to measure the *reliability* quality, the confidence curve has been used [42], and recently a couple of reliability quality measures have been proposed [40]: RPI to measure 'reliability prediction improvement', and RRI to measure 'reliability recommendation improvement'.

In our proposed approach, each prediction of the item i to the user u ($p_{u,i}$) is complemented with its reliability ($l_{u,i}$). We can define the tuple $< p_{u,i}, l_{u,i} >$, and read: ''prediction of the item i to the user u has the value $p_{u,i}$ and its reliability is $l_{u,i}$''. The model-based machine learning method has been taken from our current published paper [39]. In section III we provide a simplified explanation of it. In this paper we exploit the potential of the chosen innovative approach to discard not relevant predictions and recommendations. Our hypothesis is: *High temporal raising of the prediction reliability in an item can be caused from injected profiles; then, the prediction reliability raising of an item, compared to the rest of items, can be used as a shilling attack estimator*. The underlying idea behind our hypothesis (our motivation) is the assumption that malicious injected profiles will raise prediction reliabilities in an anomalous way. Our hypothesis is in the line of the time-based detection techniques, assuming that fake ratings are injected in short intervals of time.

The rest of the paper is structured as follows: section II summarizes the related work, section III explains the robust CF proposed method, section IV defines the experiments design, section V shows results, section VI outlines the main conclusions and the related work, finally, section ''References'' contains this paper's bibliography.

II. RELATED WORK

As explained in the previous section, the artificial intelligence approaches to oppose or to counteract CF shilling attacks are: a) *Detection* [4], and b) *Robust CF* [20]. Most of the research has focused on identifying the possible types of attacks and on detecting each of these types, in order to eliminate the malicious profiles from the knowledge source (the dataset). Efforts have also been made to design robust CF methods that allow accurate recommendations for datasets subject to shilling attacks. Detection approach research has been adequately surveyed. The shilling attack detection strategies have been categorized in [4] and [5]:

- *Detectors based on Classification*
  - *Generic statistical attributes*: these are measures where ratings are not grouped in partitions (filler and target items, Figure 1). The most used measures are: *Rating Deviation from Mean Agreement* (RDMA) [1], *Weighted Degree of Agreement* (WDA) [2], *Weighted Deviation from Mean Agreement* (WDMA) [2], *Degree of Similarity with top neighbors* (DegSim) [1], *Length Variance* (LengthVar) [2], *Entropy* [3], *Target Model Focus* [2], RD-TIA and DegSim' target item analysis detection structures [28], [32]. Multicriteria CF is tested in [37], showing it is particularly vulnerable to attacks. By using statistical attributes, QoS variation have been detected when a Web services RS is attacked [38].
  - *Model specific attributes*: these are measures based on the characteristics of the partitions. *Mean Variance* (MeanVar) and *Length Variance* (LengthVar) [2] focuses on target items, while *Filler Mean Target Difference* (FMTD) [2] incorporates information from both target and filler items.
  - *Model-based approaches*: where different types of models, without restrictions, are proposed to improve shilling attacks detection: A mapping model between rating behavior and item distribution has been found in [29]. A combination of profile-based and item-based algorithms has been used to improve detection rate of shilling attacks [30]. Time interval information is used in [31]; they propose a segmentation technique based on temporal item anomalies detection. Time series and target items analysis have been combined in [28].
- *Detectors based on Supervised learning*: detection is transformed to a binary classification problem (normal and fake profiles). NaÃŕve Bayesian [10], [11] has been extensively used in this chapter. Labeled data is required to perform supervised learning and it is not usual to have it, so current research aims towards unsupervised learning detectors. [27] first make use of generic statistical attributes (RDMA, DegSim and LengthVar) to identify fake profiles from genuine ones, then they use the

obtained values to train a supervised model for detection of malicious profiles.

- *Detectors based on Unsupervised learning*: they usually use the *Matrix Factorization* technique [7]. The most common unsupervised detectors are:
  - *Principal Component Analysis (PCA) method* [8], that computes the user-user covariance matrix. PCA is applied in [25] before and after inserting Gaussian noise in user profiles. PCA is combined with data complexity in [26]: PCA is used to choose malicious profiles, and data complexity is used to refine the chosen profiles, selecting the authentic ones. PCA has also been used to discover shilling groups in Amazon dataset [34].
  - *Multidimensional Scaling (MDS) algorithm* [9], is a two-phase detector. It computes the user-user dissimilarity matrix.
  - *Support Vector Machine (SVM)* [12] to get hidden factors from the rating matrix. Usually, time-based detection techniques assume that fake ratings are injected in short intervals of time; [36] use SVM with long duration and decentralized injection attacks information.
  - *Clustering* [33] grouping user classes, making it easy to detect malicious groups.

Robust algorithms [20] are an alternative approach to the shilling attack detection. Robust algorithms try to reduce the influence of shilling attacks. Most of the proposed designs are based on MF: [13] claims that the squared error function used in MF as loss function is too much sensitive to the large residuals; they incorporate the R1-norm to make a robust MF that improves robustness and recommendation accuracy. In the same way, M-estimators are used in [15] to create a robust MF that deals with presence of unmodeled noise. Analogously, the kernel mapping and kernel distance is used in [14] to design a robust MF model. A similar approach to the M-estimators has been proposed in [16]; in this case they use the least median squares estimator. Finally, the squared error function is improved by using least trimmed squares MF [17]. The Probabilistic Latent Semantic Analysis (PLSA) has also been used to obtain robust recommendations [18], as well as the Singular Value Decomposition (SVD) [19].

Trust information has also been extensively used to design robust algorithms [24]. This is a reasonable idea since trusted users should have a lower probability of being malicious ones. Robust algorithms usually make use of multidimensional trust models: topic and item levels trust [21], rating similarity, reliability of items and the user's trustworthiness [22], [23]. A combination of trust (credibility) and temporal information is used in [32]; they propose a detection structure based on time series analysis and abnormal group user rating. Our paper proposal can be seen as a robust algorithm based on model-based CF that exploits a fine-grained trust information: the reliability of each prediction. Unlike conventional trust approaches, where trust is a measure applied to users or items, our proposal lies on the reliability (confidence) of each individual prediction or recommendation.

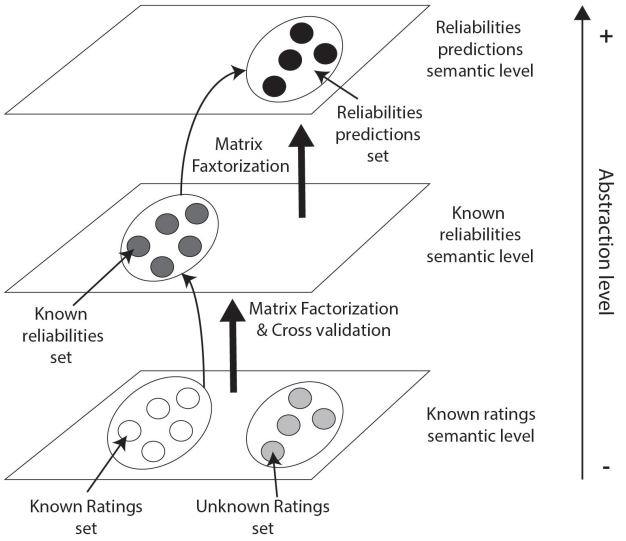

FIGURE 2. Architectural design of the robust machine learning used method [39].

III. ROBUST RECOMMENDATION METHOD

As stated in the introduction section, we work on the *hypothesis: the prediction reliability raising of an item, compared to the rest of items, can be used as a shilling attack estimator*. So, we can detect malicious recommendations by monitoring prediction reliability variations of items. As far as we know, this is an innovative approach: it is not based on making stronger the MF method, and it is not directly based on trust models. We propose a machine learning architectural method, where two semantic levels are combined to obtain reliability predictions. Our method [39] uses two sequential MF processes: the first one obtains real prediction errors, and the second one gets estimated prediction errors (prediction reliabilities).

Figure 2 (from [39]) shows these above concepts: from the whole set of feasible ratings (voted and not voted ones) RS make predictions and recommendations on the set of unknown ratings, obtaining recommendation of items that we have not consumed. Our method, instead, first gets predictions on the set of known ratings (bottom level in Figure 2); in this way we obtain prediction errors (known reliabilities), such as we do by using cross-validation in testing processes (middle level in Figure 2). Starting from the set of known reliabilities and by using a new MF process we obtain reliabilities predictions (upper level in Figure 2). This is the same process used in traditional RS: RS obtain predictions of unknown ratings by using known ratings; we obtain predictions of unknown reliabilities by using known reliabilities. The whole process is supported by a two levels architecture: reliabilities prediction provides a higher abstraction level than ratings prediction does.

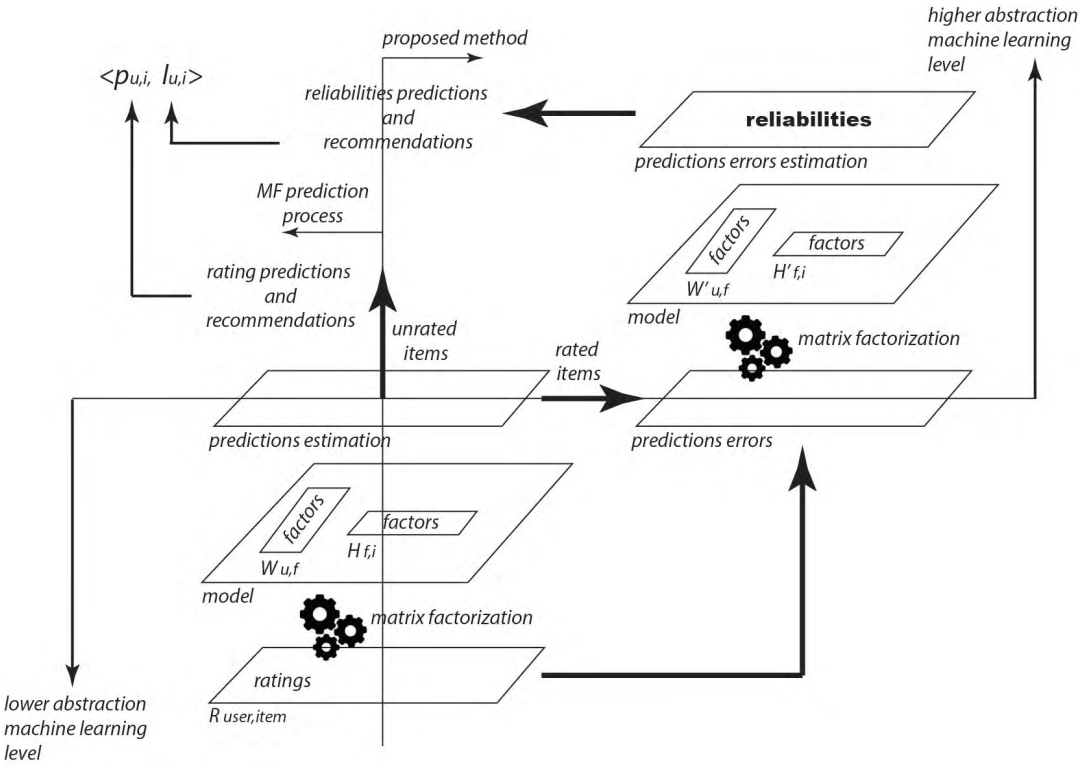

FIGURE 3. Design details of the robust machine learning proposed method [39].

Whereas Figure 2 shows the architectural big picture, Figure 3 contains the necessary details to understand the method. Figure 3 shows the same three levels from Figure 2, but now both MF processes are included; they are represented by means of black gears icons that support the generated MF models: the $W_{users,factors}$ matrix and $H_{factors,items}$ matrix in the lower level, and the $W'_{users,factors}$ matrix and $H'_{factors,items}$ matrix in the upper level. The vertical line that splits Figure 3 separates two main functionalities: a) prediction estimation (left side of the line) and b) reliability estimation (right side of the line). Prediction estimations are obtained by using the usual RS model-based MF method; each $p_{u,i}$ defines the prediction of the rating value of item i to the user u. By using the whole prediction estimations set of known ratings we can get the prediction errors set (labeled as ''prediction errors'' in the right side of Figure 3), since we know each real rating corresponding to each prediction estimation. This is our starting point to get the reliabilities estimation by means of a second MF (right side of Figure 3). The obtained result is the set of predictions errors estimation (the expected reliabilities). We define each obtained reliability of each existing prediction ($p_{u,i}$) as $l_{u,i}$. Finally, our method provides a set of pairs $< prediction, reliability >$ for each item i not voted for user u. This is shown on the upper-left side of Figure 3 as $< p_{u,i}, l_{u,i} >$.

IV. EXPERIMENTS DESIGN

To test the proposed method robustness, we will conduct a complete set of classic profile injection attacks. Traditional profile injection attacks try to promote items (push attacks) or they attempt to damage items (nuke attacks). In both cases, the set of promoted or damaged items is called *target*. When a malicious user profile is successfully inserted in the dataset, it needs to contain, in addition to the target set, a set of voted items to make it possible for this malicious profile to be used for the CF recommendation method. This set of voted items in the malicious profile can be divided in two sets: the *filler* set and the *selected* set. The filler set is usually randomly chosen. When attackers have some knowledge of the ratings distribution in the database, it is possible to choose a set of particularly relevant items to reinforce the filler set. This set is called *selected*, and its function is to get that the malicious profile has an important role in the recommendation process. Finally, the set containing the rest of items is called *unrated*, since they do not have ratings in the malicious profile.

We define:

- I as the set of items of the RS dataset.
- U as the set of users of the RS dataset.
- I_u as the set of items voted by user $u \epsilon U$
- $r_{u,i}$ as the rating of user $u \epsilon U$ on item $i \epsilon I$.
- I_T as the set of *target* items; R_T as the set of the *target* items ratings.
- I_F as the set of *filler* items; R_F as the set of the *filler* items ratings.
- I_S as the set of *selected* items; R_S as the set of the *selected* items ratings.
- $I_\emptyset$ as the set of *unrated* items.

TABLE 1. Attack types and their profile composition.

Random & Average
$I_S = \emptyset$
$R_F = \{r_{u,f} \vert u \epsilon U, f \epsilon I_F\}, r_{u,f} = random(N(\mu, \sigma^2))$ **Random** $\Rightarrow$ μ is the **dataset** rating average $mean, \sigma^2$ is the **dataset** rating variance **Average** $\Rightarrow$ μ is the **f** rating average $mean, \sigma^2$ is the **f** rating variance $\vert I_F \vert = \frac{\sum_{u \epsilon U} \vert I_u \vert}{\vert U \vert} - 1$, I_F is randomly chosen
$R_T = \{r_{u,t} \vert u \epsilon U, t \epsilon I_T\}$ $r_{u,t} = 5$ (max value) $\Leftrightarrow$ **push** attack, $r_{u,t} = 1$ (min value) $\Leftrightarrow$ **nuke** attack $\vert I_T \vert = 1$, I_T is randomly chosen
$I_\phi = I - (I_T \cup I_F)$ $\vert I_\phi \vert = \vert I \vert - \left(\frac{\sum_{u \epsilon U} \vert I_u \vert}{\vert U \vert} \right)$
Bandwagon
$R_S = \{r_{u,s} \vert u \epsilon U, s \epsilon I_s\}$ **Bandwagon** $\Rightarrow$ $r_{u,s} = 5$ (max value) **ReverseBandwagon** $\Rightarrow$ $r_{u,s} = 1$ (min value) $\vert I_S \vert = \frac{\left(\frac{\sum_{u \epsilon U} \vert I_u \vert}{\vert U \vert} \right)}{2} - 1$ **Bandwagon** $\Rightarrow$ I_S contains the $\vert I_S \vert$ **most** popular items of the dataset **ReverseBandwagon** $\Rightarrow$ I_S contains the $\vert I_S \vert$ **least** popular items of the dataset
$R_F = \{r_{u,f} \vert u \epsilon U, f \epsilon I_F\}, r_{u,f} = random(N(\mu, \sigma^2))$ **Bandwagon$_{\text{random}}$** and **ReverseBandwagon$_{\text{random}}$** $\Rightarrow$ μ is the **dataset** rating average $mean, \sigma^2$ is the **dataset** rating variance **Bandwagon$_{\text{average}}$** and **ReverseBandwagon$_{\text{average}}$** $\Rightarrow$ μ is the **f** rating average $mean, \sigma^2$ is the **f** rating variance $\vert I_F \vert = \frac{\left(\frac{\sum_{u \epsilon U} \vert I_u \vert}{\vert U \vert} \right)}{2}$, I_F is randomly chosen
$R_T = \{r_{u,t} \vert u \epsilon U, t \epsilon I_T\}$ **Bandwagon** $\Rightarrow$ $r_{u,t} = 5$ (max value) $\Leftrightarrow$ **push** attack $r_{u,t} = 1$ (min value) $\Leftrightarrow$ **nuke** attack **ReverseBandwagon** $\Rightarrow$ $r_{u,t} = 1$ (min value) $\Rightarrow$ **nuke** attack $\vert I_T \vert = 1$, I_T is randomly chosen
$I_\phi = I - (I_T \cup I_F \cup I_S)$ $\vert I_\phi \vert = \vert I \vert - \left(\frac{\sum_{u \epsilon U} \vert I_u \vert}{\vert U \vert} \right)$
Love-hate
$I_S = \emptyset$
$R_F = \{r_{u,f} \vert u \epsilon U, f \epsilon I_F\}$ $r_{u,t} = 1$ (min value) $\Leftrightarrow$ **push** attack, $r_{u,t} = 5$ (max value) $\Leftrightarrow$ **nuke** attack $\vert I_F \vert = \frac{\sum_{u \epsilon U} \vert I_U \vert}{\vert U \vert} - 1$, I_F is randomly chosen
$R_T = \{r_{u,t} \vert u \epsilon U, t \epsilon I_T\}$ $r_{u,t} = 5$ (max value) $\Leftrightarrow$ **push** attack, $r_{u,t} = 1$ (min value) $\Leftrightarrow$ **nuke** attack $\vert I_T \vert = 1$, I_T is randomly chosen
$I_\phi = I - (I_T \cup I_F)$ $\vert I_\phi \vert = \vert I \vert - \left(\frac{\sum_{u \epsilon U} \vert I_u \vert}{\vert U \vert} \right)$
Perfect Knowledge
$I_S = \emptyset$
$R_S = \{r_{u,s} \vert u \epsilon U, s \epsilon I_S\}$ $r_{u,s} = r_{u,i} \pm$ random([0..1]) $\vert I_S \vert = \frac{\sum_{u \epsilon U} \vert I_U \vert}{\vert U \vert} - 1$, I_S is randomly chosen
$R_T = \{r_{u,t} \vert u \epsilon U, t \epsilon I_T\}$ $r_{u,t} = 5$ (max value) $\Leftrightarrow$ **push** attack, $r_{u,t} = 1$ (min value) $\Leftrightarrow$ **nuke** attack $\vert I_T \vert = 1$, I_T is randomly chosen
$I_\phi = I - (I_T \cup I_F)$ $\vert I_\phi \vert = \vert I \vert - \left(\frac{\sum_{u \epsilon U} \vert I_u \vert}{\vert U \vert} \right)$

The set of attacks we include in our design can be classified as:

- attack={push,nuke}
- push={random, average, bandwagon$_{\text{random}}$, bandwagon$_{\text{average}}$, love-hate, perfectKnowledge}
- nuke={random, average, reverseBandwagon$_{\text{random}}$, reverseBandwagon$_{\text{average}}$, love-hate, perfectKnowledge}

All the attacks fulfill:

- $I = I_T \cup I_F \cup I_S \cup I_\phi$
- $I_T \cap I_F \cap I_S \cap I_\phi = \emptyset$

Table 1 summarizes the details of each of the attacks designed to test the proposed method. Bold texts show attacks variations: push or nuke, random or average, and reverse or direct. Table 1 defines each set of items (R_T, R_F, R_S) and each set of ratings (I_T, I_F, I_S, I_ϕ). We have decided to insert malicious profiles containing the same number of ratings that the dataset users' average. Random attacks use the existing knowledge of the mean and variance of the ratings from the whole dataset. Average attacks need a deeper knowledge of the dataset statistics: it is necessary to know the ratings mean and variance for each item.

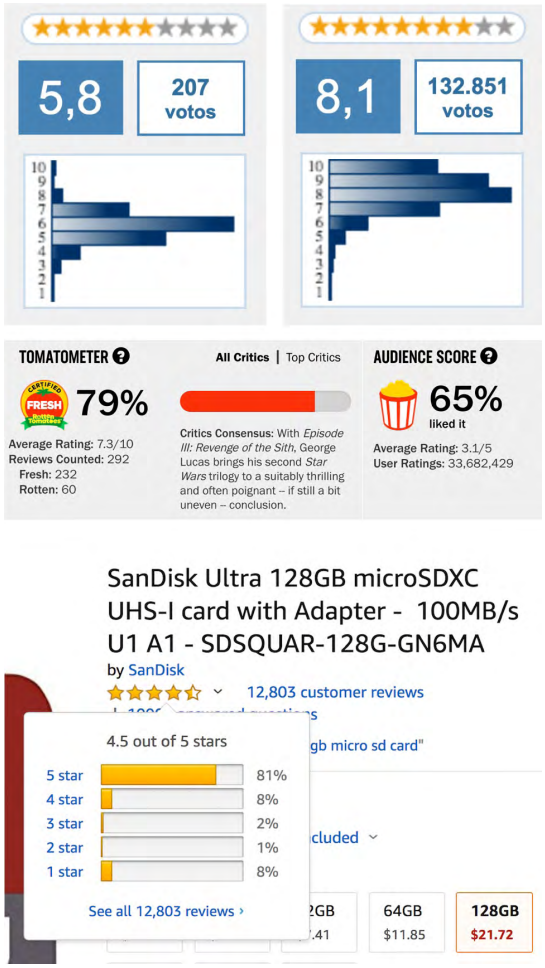

FIGURE 4. Distribution of ratings for different items belonging to current commercial RS. Source: Amazon (bottom), Rotten Tomatoes (middle) and FilmAffinity (upper).

Finally, bandwagon attacks require to know the popularity of each item. Commercial RS usually provide this information; Figure 4 shows some examples taken from *Amazon, Rotten Tomatoes* and *FilmAffinity*.

Our experiments design takes the open *Movielens 1M* dataset to create the shilling attacks. Table 2 shows the main values of this CF dataset. We have inserted malicious profiles to the dataset by conducting several profile injection attacks (Table 1). We have run more than thirteen thousand independent experiments. For each of these experiments we have created a separated *Movielens 1M* version containing the corresponding malicious injected profiles. To make it possible the reproducibility of the experiments we provide the whole set of *Movielens 1M* attacked versions (13,200 datasets), available through IEEE DataPort [51]. We have also used the CF4J framework [6] to implement the experiments. Source code of the experiments carried out to this contribution are available through CodeOcean platform.

The designed experiments provide separate profile injection attacks attending to the following five grouping types. Table 3 abstracts the designed 13,200 separated experiments.

TABLE 2. Movielens 1M dataset main facts.

# ratings	1,000,209
# items	3,706
# users	6,040
# voted items (users average)	166
ratings mean (whole dataset)	3.58
ratings deviation (whole dataset)	1.11

TABLE 3. Designed experiments.

Type	Options	Quantity
Target	{push, nuke}	2
Attack method	{random, average, bandwagon-random, bandwagon-average, love-hate, perfect-knowledge}	6
Item popularity	{Q1, Q2, Q3, Q4}	4
Target item	25 target items	25
Malicious users	Q4: {0, 100, 150, 200, ..., 500}	11
	Total of experiments	13,200

(a) *Target*: nuke experiments and push experiments (2 variations).

(b) *Attack method*: (6 variations).
push={random, average, $\text{bandwagon}_{\text{random}}$, $\text{bandwagon}_{\text{average}}$, love-hate, perfectKnowledge}
nuke={random, average, $\text{reverseBandwagon}_{\text{random}}$, $\text{reverseBandwagon}_{\text{average}}$, love-hate, perfectKnowledge}

(c) *Item popularity*: (4 variations)
Items have been classified in four groups (quartiles), attending to their popularity: Q1 (25% most popular items), Q4 (25% least popular items), and Q2 & Q3 in the same way.

(d) *Target items*: combining the three above variation types, we obtain a set of 48 total variations; e.g. <nuke, average, Q3>, <push, love-hate, Q1>, etc. We randomly choose 25 target items from each of the 48 variations; each target item will be separately tested. (25 variations).

(e) *Malicious users*: finally, the 48 x 25 = 1,200 randomly chosen target items have been attacked for eleven different number of malicious users, ranging: 0, 50, 100, ..., 500. The profile attack with no malicious users (0 users) has been included for control purposes.

To run the testing process, we have used PMF MF and we have chosen 15 hidden factors, 50 iterations and 0.055 as regularization parameter. These values have been used both for the predictions estimation MF and the reliabilities estimation MF; that is to say, for the two MF levels represented in Figure 3.

From the whole set of shilling attack datasets, we do not consider attacks when the number of malicious profiles is too small compared to the quantity of the dataset information; e.g.: when we run a Q1-based experiment by using a small number of malicious users. Q1 experiments involve the subset of the database containing the largest number of items; in this case it is not reasonable to deploy an attack containing a reduced number of malicious profiles, because its limited number of ratings will not significantly affect

the model-based CF method operation. On the other hand, attacks against Q4 subsets can have an effect even using a small number of malicious profiles. In order to formalize this concept, we will consider results as representative when experiments follow this condition:

- $\#maliciousProfiles \geq \left(\overline{ratings_{items}} - \overline{ratings_{user}}\right)(Q - q + 1)$, where: $Q4$ quartiles, q is the quartile involved in the experiment.

In our dataset: the items average of ratings is 270, and the users average of ratings is 166 (Table 2). Following the equation:

Minimum number of malicious profiles for $Q1$: $(270 - 166) * 4 = 416$; for $Q2 : (270 - 166) * 3 = 312$, for $Q3 : (270 - 166) * 2 = 208$; for $Q4 : (270 - 166) * 1 = 104$. Since multiples of 50 malicious profiles are used in our experiments, we round to these values and we obtain: $Q1$ representative results are set for $\{400, 450, 500\}$ malicious profiles, $Q2$ representative results are set for $\{300, 350, \ldots, 500\}$, $Q3$ for $\{200, 250, \ldots, 500\}$ and $Q4$ for $\{100, 150, \ldots, 500\}$. Overall, we provide 8, 400 representative results.

Finally, the 13,200 tests are used to feed four different experiment types:

(a) *Evolution of the prediction values:* we use these experiments to test the validity of each shilling attack file. We expect a raising of the prediction values in the *push* experiments, and a decrease of the prediction values when the nuke attacks are tested. These experiments do not run the proposed algorithm.

(b) *Evolution of the prediction reliability values in the set of target items:* these are the key experiments of the paper. We measure the raising of the reliabilities associated to each type of shilling attack performed on the different scenarios of data sparsity: $Q1$ to $Q4$. We expect a usual behavior where reliability values raise. Changing scenarios, such as a reduced number of malicious profiles trying to attack 'dense' datasets ($Q4$ & $Q3$), can exhibit temporal decreasing reliability evolutions, since they change the existing patterns of ratings. According as the number of malicious profiles increases, rating patterns stabilize and reliabilities start to raise.

(c) *Evolution of the prediction reliability values in the set of non-target items:* we can consider these experiments as control tests: whereas in the experiments type b) we monitor testing items, in these experiments we monitor not testing items. We expect that the prediction reliabilities evolution will be not significant in the not testing items, making it possible to differentiate these cases from the relevant ones: the significant reliabilities raising in the testing set of items.

(d) *Increase of prediction reliability values:* we test here our shilling attack estimator. We simply subtract each testing item average reliability from the non-testing items average reliability. For computational purposes we just use ten non-testing items. Since c) prediction reliabilities will not be significant, we expect a results behavior similar to the explained in paragraph b).

TABLE 4. Distribution of the figures showed in the results section.

Subsection	Experiment	push	nuke
Performance of the shilling attacks	Evolution of the prediction values	Figure 5	Figure 6
	Probability that a RS user will be recommended a target item	Figure 7	Figure 8
Performance of the proposed method	Evolution of the prediction reliability values in the set of target items	Figure 9	Figure 10
	Evolution of the prediction reliability values in the set of non-target items	Figure 11	Figure 12
	Increase of prediction reliability values	Figure 13	Figure 14

V. RESULTS

This section contains two main subsections: the first one shows the performance of all the designed shilling attacks experiments. Mainly it shows: a) the evolution of recommendation values in the shilling attacks scenarios; we expect a raising in the push attacks and a decrease in the nuke attacks, and b) the probability that a RS user will be recommended a target item when the dataset is attacked: in push attacks we expect raising probabilities as the size of the attack increases; conversely, we expect a total decrease when nuke attacks are tested. Subsection two shows the proposed method performance; it contains the three types of results explained at the end of section four: 1) evolution of the prediction reliability values in the set of target items, 2) evolution of the prediction reliability values in the set of non-target items and 3) increase of prediction reliability values. Table 4 organizes the distribution of figures in this section.

Each of the involved figures contains four graphs, labeled a) to d). Graphs labeled as "a" correspond to the experiments results where the most voted items are taken: these are the sets we named as $Q1$, containing the 25% most voted items. Analogously, graphs labeled as "d" correspond to the $Q4$ set of items (the 25% less voted ones). $Q2$ experiments are shown in graphs labeled "b", and $Q3$ ones in the "c" graphs. We have designed experiments in this way due to the importance of the dataset sparsity: the less sparse the dataset is, the more malicious profiles will be necessary for the shilling attack to succeed. For this reason, robust algorithms can fail neutralizing weak shilling attacks on not very sparse datasets: it is difficult to neutralize the really small effect of a reduced number of malicious profiles inserted in a heavy dataset. Nevertheless, the opposite situation is important: to neutralize effective shilling attacks; that is to say, to neutralize situations in which a large number of malicious profiles have been injected in a particularly sparse dataset. These concepts links with the representative number of malicious profiles that we have stated in the previous section: "$Q1$ representative results are set for $\{400, 450, 500\}$ malicious profiles, $Q2$ representative results are set for $\{300, 350, \ldots, 500\}$, $Q3$ for $\{200, 250, \ldots, 500\}$ and $Q4$ for $\{100, 150, \ldots, 500\}$".

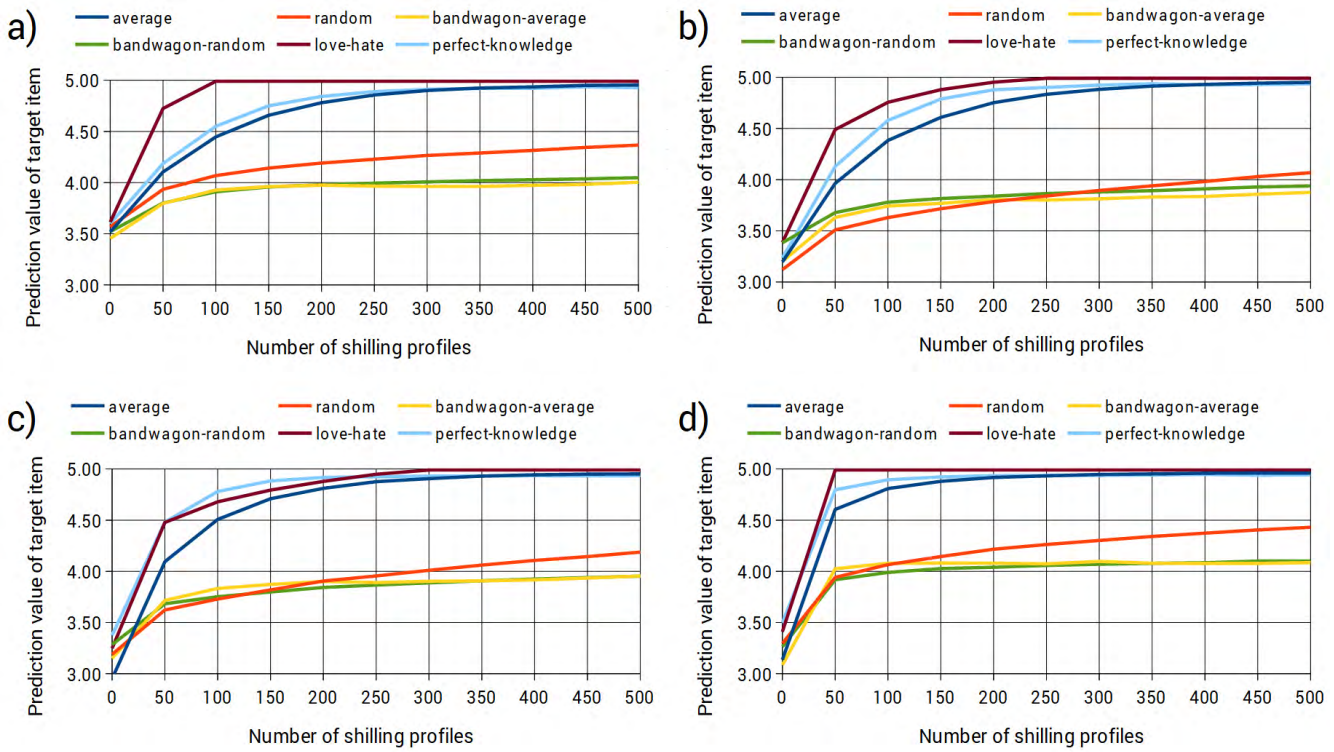

FIGURE 5. Evolution of the prediction values to the target items. Push attacks. Attack types: random, love-hate, bandwagon random & average, perfect knowledge. Graphs: a) *Q*1 (25% most voted items), b) *Q*2 (from 25% to 50% most voted items), c) *Q*3 (from 50% to 75% most voted items), d) *Q*4 (25% less voted items). X-axis: number of injected malicious profiles.

A. PERFORMANCE OF THE DESIGNED SHILLING ATTACKS

1) EVOLUTION OF THE PREDICTION VALUES

Figure 5 shows the evolution of the prediction values to the target items as the number of injected malicious profiles grows. As expected, in the push attacks, trend is towards the maximum value (rating five, in Movielens dataset). We can observe that shilling attacks reach their maximum prediction values, by injecting a reduced number of malicious profiles, when $Q4$ datasets are tested. As expected, as the $Q3$, $Q2$ & $Q1$ datasets are tested, a larger number of malicious profiles are necessary to reach high prediction values. Additionally, we can observe that attacks on $Q4$ get higher prediction values than attacks on $Q3$, and so on. All these observations support the validity of the designed push shilling attacks. Finally, we detect that the performance of bandwagon and random attacks is lower than love-hate, average and perfect knowledge attacks.

Figure 6 is equivalent to Figure 5 but now we show results when nuke attacks are tested. The attack types are the same that we used for push experiments, exception of the bandwagon ones: Figure 5 shows results for both reverse bandwagon random and reverse bandwagon average. As expected, predictions drop to the minimum rating value (one in the Movielens dataset). Perfect knowledge and average perform better than reverse bandwagons, love-hate and random attacks. As expected, the more malicious profiles are injected in the dataset, the more predictions evolve towards the minimum rating value. Likewise, attacks are more effective when applied to the very sparse $Q4$ dataset, and effectiveness progressively diminishes for $Q3$, $Q2$ and $Q1$ experiments.

We can conclude affirming that experiments in this subsection show that the designed shilling attacks work right, and they show the expected behaviors and evolutions.

2) PROBABILITY THAT A RS USER WILL BE RECOMMENDED A TARGET ITEM

The main push shilling attacks objective is to get positive recommendations for the target items. Figure 5 shows us that this numeric tendency is fulfilled: recommendations shift to the maximum rating value. But this is not enough to perform an effective shilling attack: it is also necessary to get a relevant probability that target items are recommended; e.g.: to recommend the pushed target items to only two users it is not an effective attack, whereas to recommend the pushed target items to the 40% of the dataset users is a complete success. To measure this probability, we have designed the appropriate experiments whose results are shown in Figure 7. The executed experiments obtain the number of users that receive recommendations of target items, then Figure 7 shows the resulting probabilities.

Results from Figure 7 show the expected behavior: a) the probability that target items are recommended raises when the number of injected profiles increases, and b) as much sparser the dataset is, the tested probability raises higher.

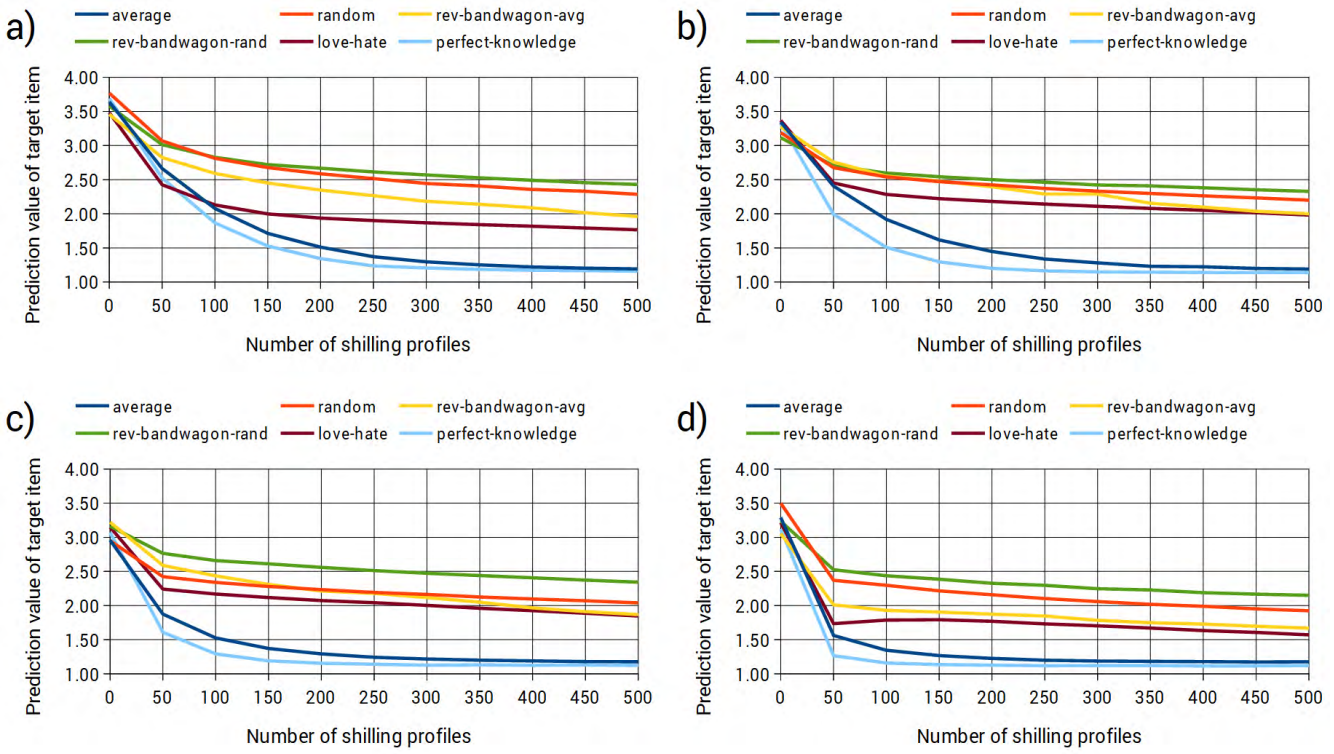

FIGURE 6. Evolution of the prediction values to the target items. Nuke attacks. Attack types: random, love-hate, reverse bandwagon random & average, perfect knowledge. Graphs: a) *Q*1 (25% most voted items), b) *Q*2 (from 25% to 50% most voted items), c) *Q*3 (from 50% to 75% most voted items), d) *Q*4 (25% less voted items). X-axis: number of injected malicious profiles.

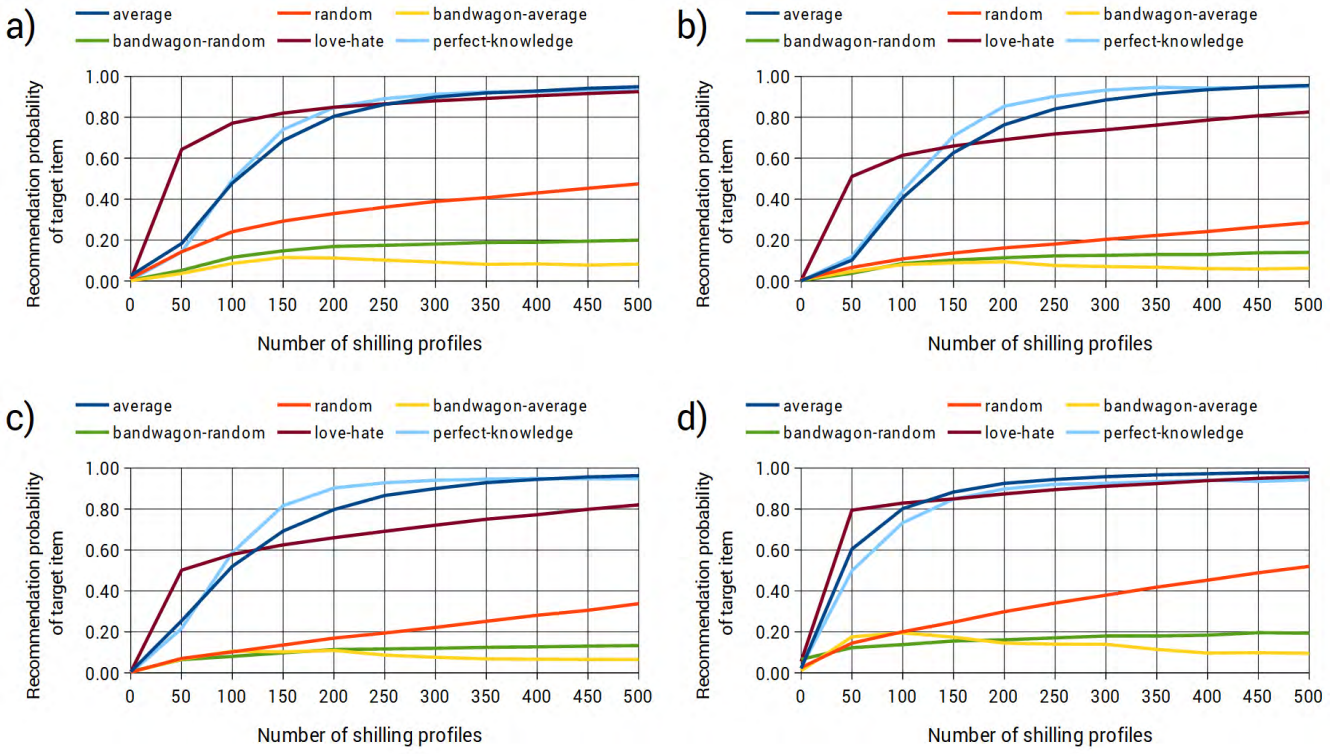

FIGURE 7. Probability that a RS user will be recommended a target item. Push attacks. Attack types: random, love-hate, bandwagon random & average, perfect knowledge. Graphs: a) *Q*1 (25% most voted items), b) *Q*2 (from 25% to 50% most voted items), c) *Q*3 (from 50% to 75% most voted items), d) *Q*4 (25% less voted items). X-axis: number of injected malicious profiles.

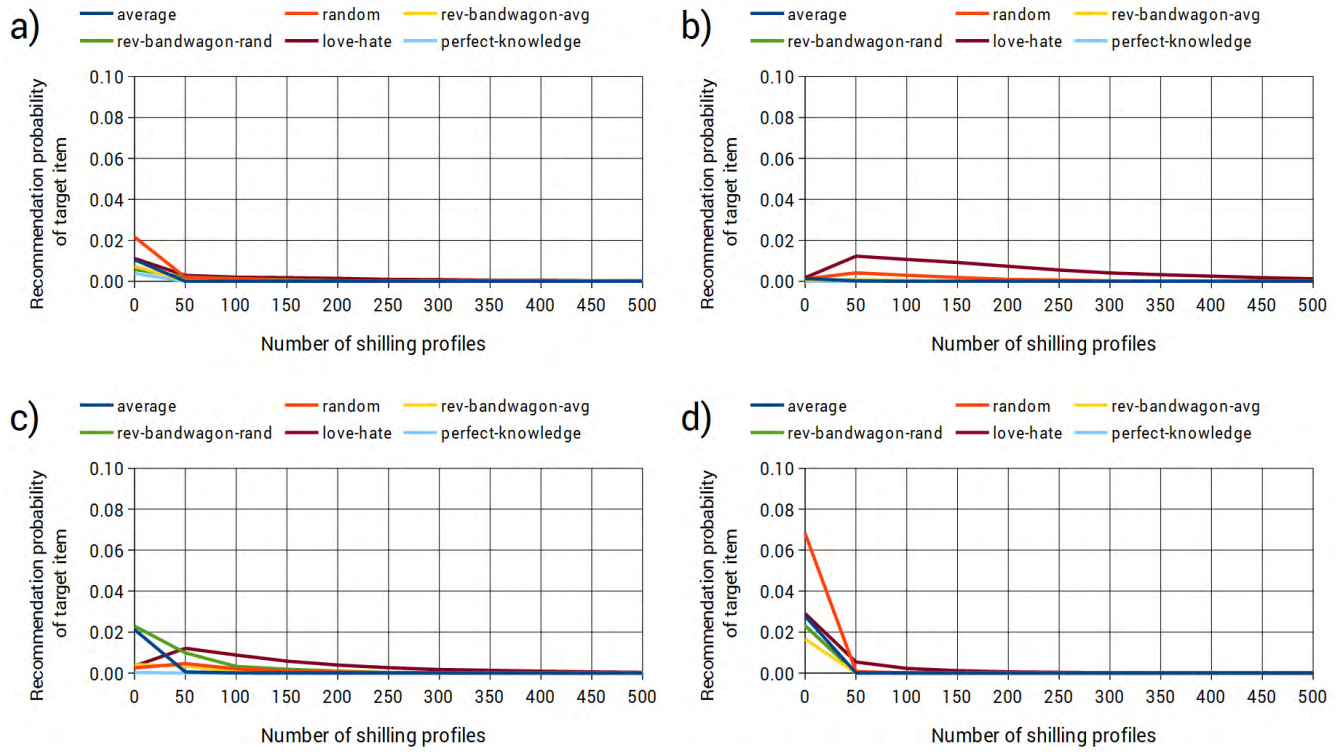

FIGURE 8. Probability that a RS user will be recommended a target item. Nuke attacks. Attack types: random, love-hate, reverse bandwagon random & average, perfect knowledge. Graphs: a) *Q*1 (25% most voted items), b) *Q*2 (from 25% to 50% most voted items), c) *Q*3 (from 50% to 75% most voted items), d) *Q*4 (25% less voted items). X-axis: number of injected malicious profiles.

Additionally, love-hate, average and perfect knowledge are confirmed as the more dangerous attacks: they get to surpass the 80% of recommended users in all the tested scenarios. Figure 8 shows the equivalent results when experiments test nuke attacks. The most relevant in Figure 8 is the effectivity of the nuke attacks: all of them get quickly that target items stop being recommended. This is the expected behavior, since whereas to promote an item involves competition with many others, to damage recommendations it is enough to decrease a little this item's prediction value (only a little set of the top predictions are promoted to recommendations).

B. PERFORMANCE OF THE PROPOSED METHOD

1) EVOLUTION OF THE PREDICTION RELIABILITY VALUES IN THE SET OF TARGET ITEMS

Experiments in this subsection are the key of the paper: they test the most important aspect of our hypothesis: the prediction reliability raising of items can be used as a shilling attack estimator. We have averaged prediction reliabilities on the designed scenarios: several injection profiles attacks and several sizes of injected profiles. Results are shown in Figure 9 for the four different sparsity levels we used in previous experiments: graph '*d*' shows the sparser dataset (*Q*4), graph '*a*' shows the less sparse dataset (*Q*1), graph '*b*' shows *Q*2 results and graph '*c*' the *Q*3 ones.

Graph '*d*' in Figure 9 clearly shows the expected behavior: reliability increases as the shilling attack size (number of injected profiles) does. Furthermore, this behavior is fulfilled for all the tested attacks. Graph '*c*' shows the same general behavior, but now we can see an initial decreasing before the generalized increase. The initial decreasing corresponds to the temporary situation in which the shilling attack has not yet been able to vary the CF model; this is the reason why reliability falls: there is not 'consensus' to make accurate recommendations. According as the attack size increases, the model learns the malicious patterns and it stabilizes the evolution of the prediction reliabilities: the shilling attack is doing its work, but it is leaving a trail of detectable reliability raising. As the dataset sparsity decreases (graphs '*b*' and '*a*') it is necessary to inject more profiles to get the model learn the malicious patterns; this is the reason why the initial decrease of reliability is wider in graphs '*a*' and '*b*'. Although this circumstance can be seen as a drawback, it is important to remember that these situations correspond with failed shilling attacks, since they have not yet been able to significantly modify the CF model. In summary: the proposed method behaves more efficiently in cases in which shilling attacks acquire relevance.

As expected, love-hate and average attacks are more difficult to counteract by using the proposed method, since they perform a better behavior (Figure 5 & 7). It is interesting to realize that our method works fine with the particularly difficult perfect knowledge attack. Figure 10 shows the nuke version of Figure 9; it is remarkable the existing similarity

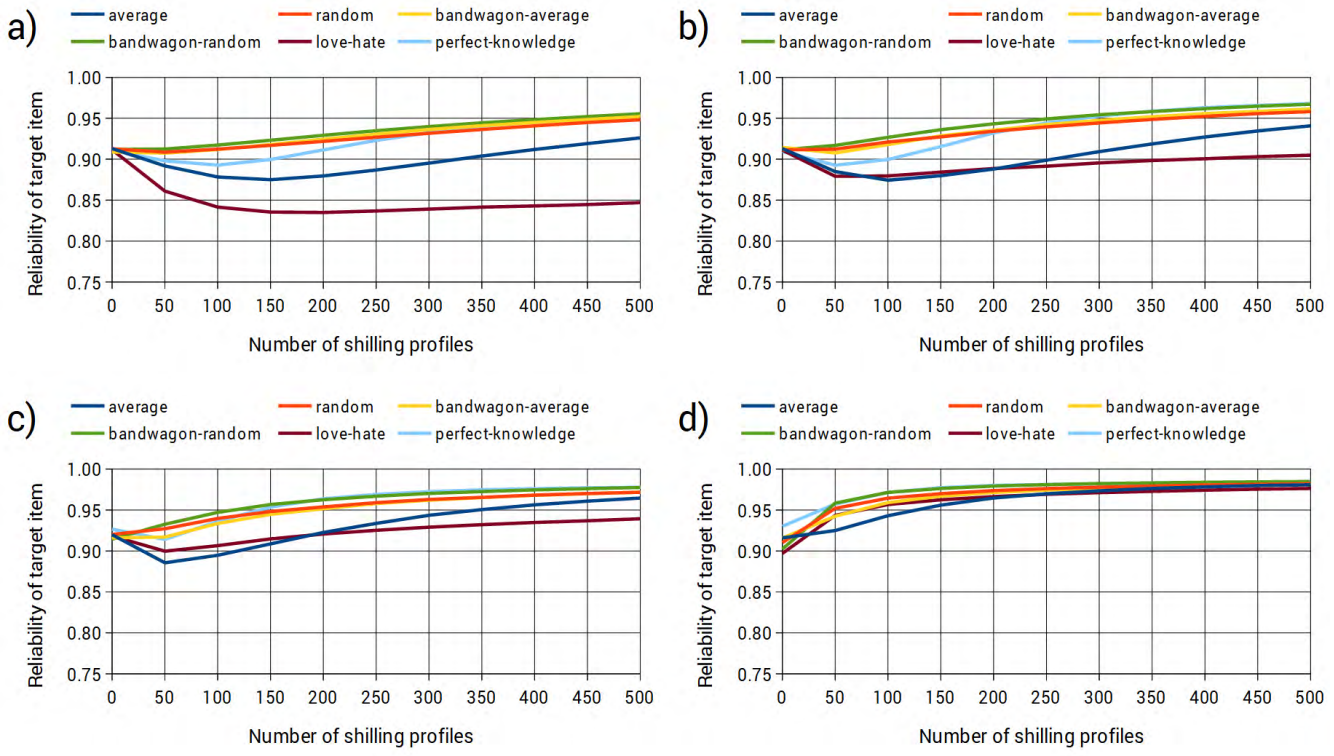

FIGURE 9. Reliability of target items. Push attacks. Attack types: random, love-hate, bandwagon random & average, perfect knowledge. Graphs: a) *Q*1 (25% most voted items), b) *Q*2 (from 25% to 50% most voted items), c) *Q*3 (from 50% to 75% most voted items), d) *Q*4 (25% less voted items). X-axis: number of injected malicious profiles.

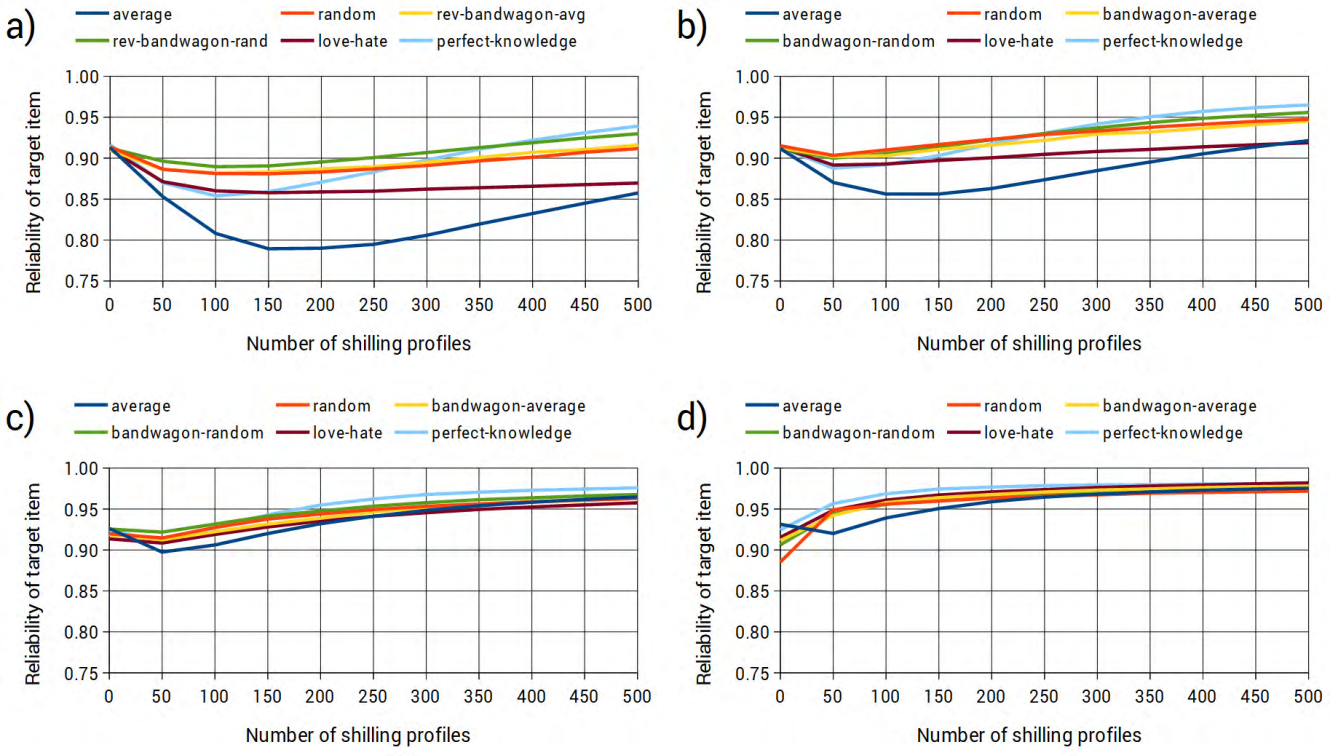

FIGURE 10. Reliability of target items. Nuke attacks. Attack types: random, love-hate, reverse bandwagon random & average, perfect knowledge. Graphs: a) *Q*1 (25% most voted items), b) *Q*2 (from 25% to 50% most voted items), c) *Q*3 (from 50% to 75% most voted items), d) *Q*4 (25% less voted items). X-axis: number of injected malicious profiles.

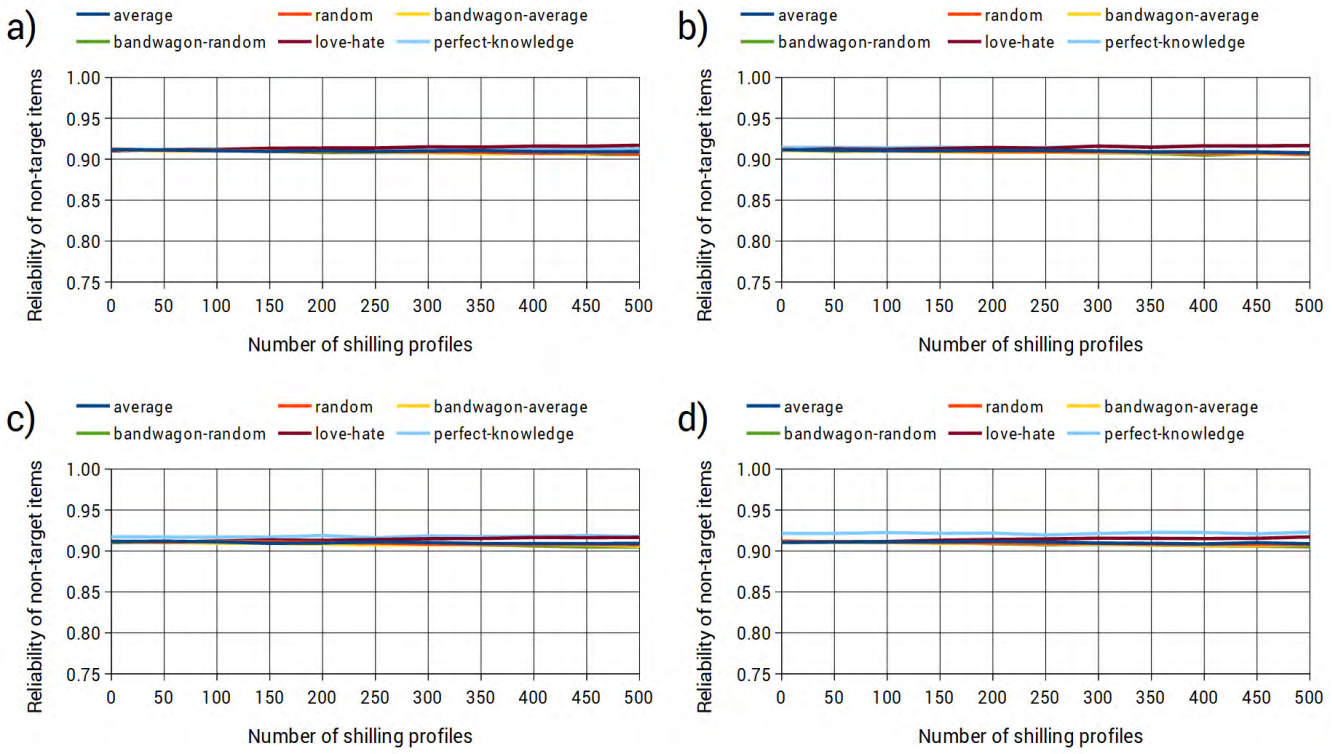

FIGURE 11. Reliability of non-target items. Push attacks. Attack types: random, love-hate, bandwagon random & average, perfect knowledge. Graphs: a) *Q*1 (25% most voted items), b) *Q*2 (from 25% to 50% most voted items), c) *Q*3 (from 50% to 75% most voted items), d) *Q*4 (25% less voted items). X-axis: number of injected malicious profiles.

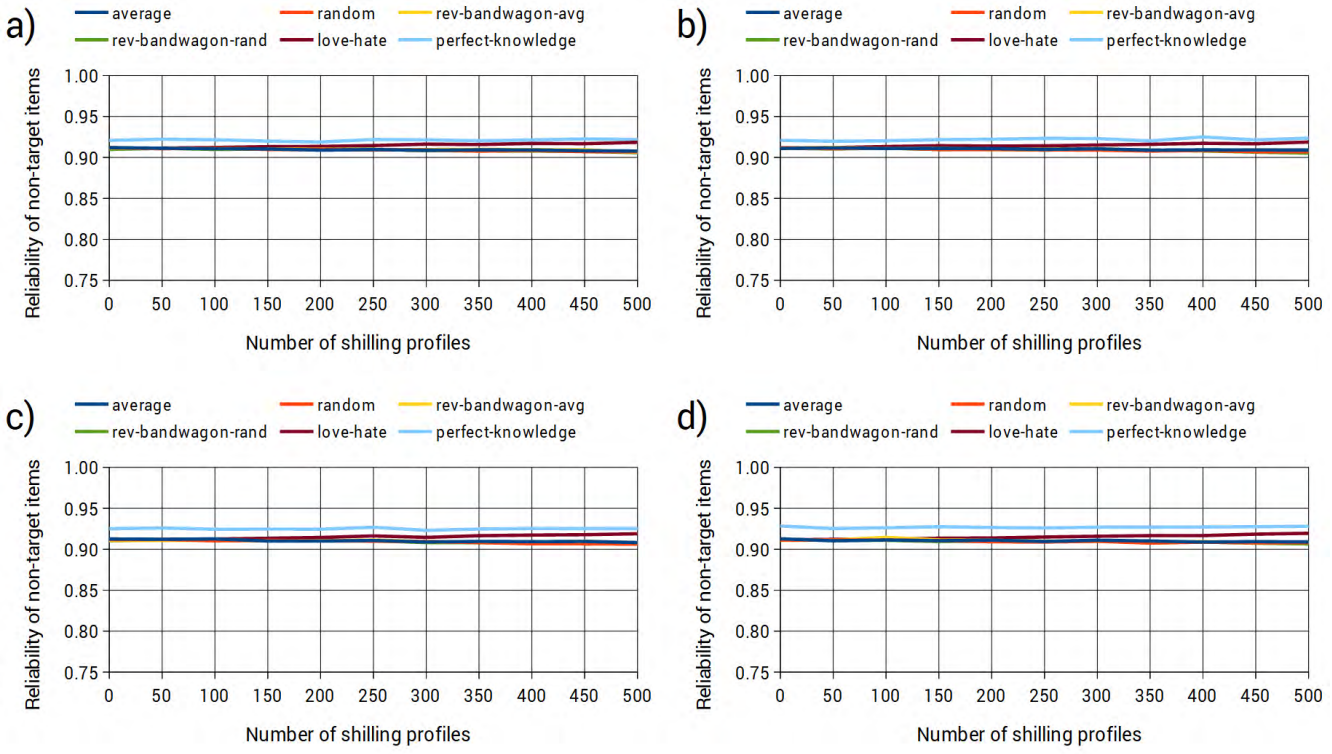

FIGURE 12. Reliability of non-target items. Nuke attacks. Attack types: random, love-hate, reverse bandwagon random & average, perfect knowledge. Graphs: a) *Q*1 (25% most voted items), b) *Q*2 (from 25% to 50% most voted items), c) *Q*3 (from 50% to 75% most voted items), d) *Q*4 (25% less voted items). X-axis: number of injected malicious profiles.

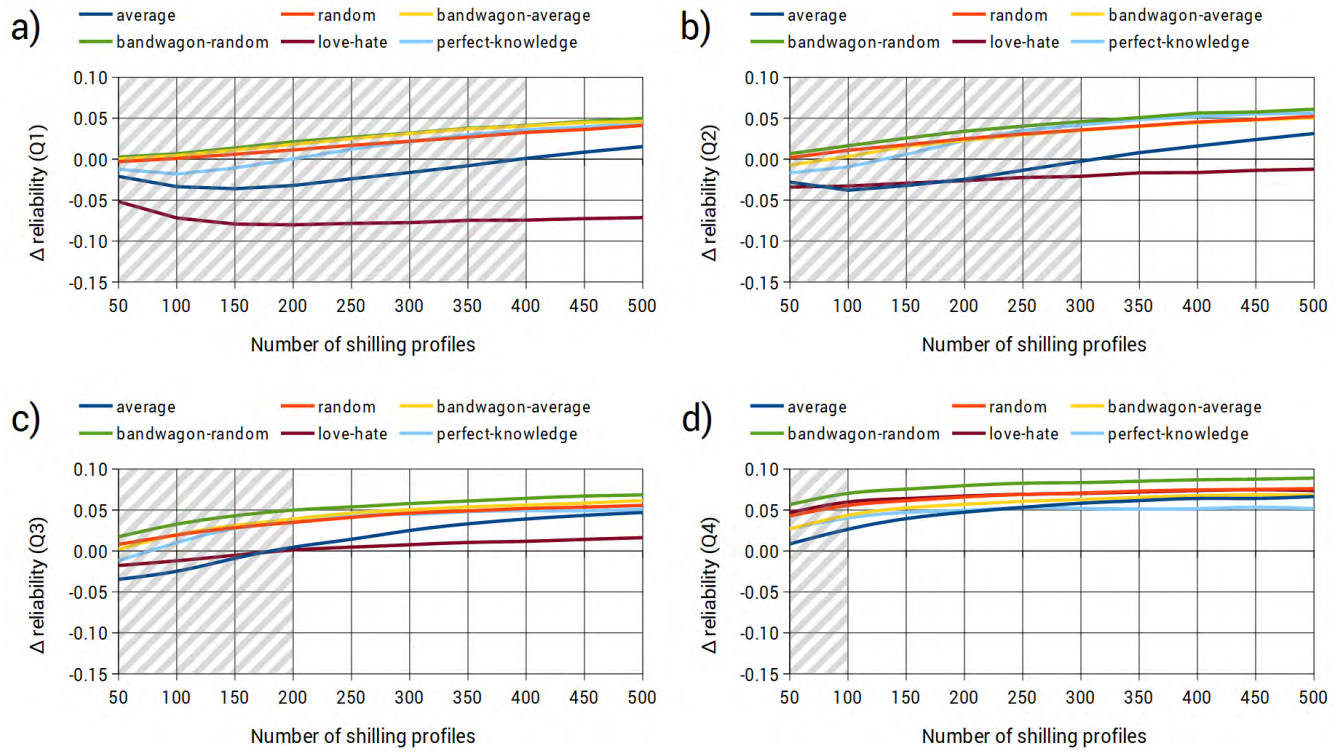

FIGURE 13. Reliability increase of target items. Push attacks. Attack types: random, love-hate, bandwagon random & average, perfect knowledge. Graphs: a) *Q*1 (25% most voted items), b) *Q*2 (from 25% to 50% most voted items), c) *Q*3 (from 50% to 75% most voted items), d) *Q*4 (25% less voted items). X-axis: number of injected malicious profiles.

between both figure graphs, so explanations made for Figure 9 are valid to Figure 10. The interesting issue is that the proposed model catches both the push and the nuke attacks in the same way: by analyzing high semantic information based on the matrix factorization hidden factors. This level of abstraction, based on two consecutive machine learning processes, makes it possible to tackle shilling attacks by analyzing the high-level reliability information.

2) EVOLUTION OF THE PREDICTION RELIABILITY VALUES IN THE SET OF NON-TARGET ITEMS

Once we have tested that there exist reliability raising in the shilling attacked target items, we just have to check that we will be able to detect the reliability raising in order to use it as shilling attack estimator. Formally, we must discard the possibility of reliability raising in non-target items to be able to perceive genuine attacks. That is to say: we cannot detect target items reliability raising in a context where all the dataset items raise their reliabilities in shilling attacks scenarios. This is the reason why our complete hypothesis has been stated as: "The prediction reliability raising of an item, compared to the rest of items, can be used as a shilling attack estimator". This subsection checks the "compared to the rest of items" portion of the hypothesis.

We do not expect reliability raising in the non-target items, since the injected profiles are only directed towards target items; nevertheless, we have designed a complete set of experiments to check it. Figure 11 & 12 show the obtained results; their y-axis graphs have the same scale as Figure 9 & 10. We can see, both for the push and the nuke experiments, that there are not significative variations in the reliability evolution of the non-target items.

3) INCREASE OF PREDICTION RELIABILITY VALUES

Once we have tested that there exists a reliability raise in the target items of a shilling attack (section V-B.1) and that the non-target items do not experiment any relevant raise (section V-B.2), we use both information to obtain the shilling attack estimator. We just subtract the reliability value from each analyzed target item minus the reliability average of non-target items. Specifically, we use the average of the N most similar items to the target one, and we assign $N = 10$. Positive values of the estimator denote shilling attack detection, whereas negative values denote no attack or the possibility of an attack start. In this paper we focus in the positive values, that will indicate us shilling attack scenarios.

Figure 13 shows our estimator results when applied to the push shilling attack scenarios designed for this paper. Grey shadow areas delimitate situations where the shilling attacks are not particularly relevant: when the number of injected profiles is not enough to effectively alter the RS recommendations. Graph '*d*' in Figure 13 shows a convincing behavior of the proposed estimator: it returns positive values

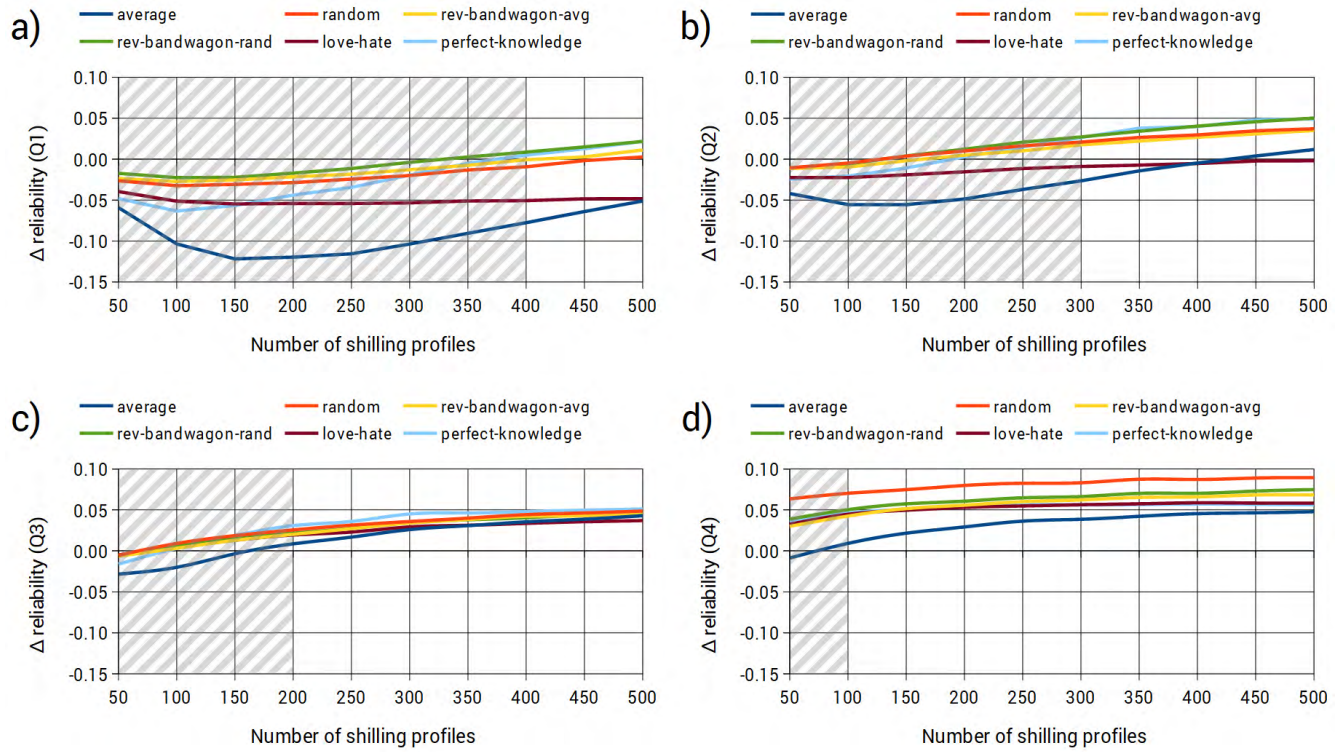

FIGURE 14. Reliability increase of target items. Nuke attacks. Attack types: random, love-hate, reverse bandwagon random & average, perfect knowledge. Graphs: a) *Q1* (25% most voted items), b) *Q2* (from 25% to 50% most voted items), c) *Q3* (from 50% to 75% most voted items), d) *Q4* (25% less voted items). X-axis: number of injected malicious profiles.

for all the attack types and all the sizes of malicious profiles. This means that it is working fine in sparse dataset scenarios. When the scenario is something less sparse (graph '*c*') the estimator only fails for the most demanding attack types and when the shilling attack is still not relevant (less than 150 malicious profiles). Graphs '*b*' and '*a*' show a continuity in the explained behavior: they need more injected profiles to neutralize the attack, but it should be noted that attacks based in few malicious profiles do not significantly alter the CF model and its recommendations. From Figure 13 we can determine that the love-hate attack is particularly difficult to neutralize by using the proposed method, and it opens a potential future work.

Figure 14 shows results of the nuke experiments. Basically, the observed behavior is similar to the push attacks one. As explained, the reason is the underlying design of the proposed method, based on a model-based machine learning approach that provides the abstract reliability values. We can to point out some minor particularities: 1) when sparsity is not high, the method performance is slightly lower in nuke scenarios compared to the push ones, and 2) whereas the love-hate attack is more difficult to neutralize in the push modality, in nuke scenarios the average attack is more effective.

C. DISCUSSION

We provide an open database containing more than three thousand individual datasets designed to test many different types of shilling attacks, diverse sparsity conditions and both push and nuke variations. First, we have tested the correct behavior of the shilling attacks by a) determining the evolution of prediction values, and b) obtaining the probability of RS users of being recommended a target item. Results show the correct operation of the attacked datasets. Secondly, we have checked the paper hypothesis by a) testing the reliability of predictions to target items, b) testing the reliability of predictions to non-target items, and c) combining both reliabilities. Results confirm our hypothesis validity.

The proposed model-based estimator to neutralize shilling attacks works particularly fine on sparse datasets. When the datasets are less sparse, the estimator needs a larger number of malicious profiles in order to learn the model variations and to provide the adequate reliability values. This drawback is not serious because these situations correspond with failed shilling attacks, since they have not yet been able to significantly modify the CF model.

The proposed method works fine in both push and nuke attacks: this is due to its design, based on a two machine learning stages that confers it a high abstraction level. This design makes it possible to provide prediction reliabilities that properly catch the evolution of the CF model in both push and nuke scenarios. All the complete set of tested attack types has been adequately managed by the proposed method, exception in some scenarios, of the love-hate and average attacks which will need of additional research in future works to fine tune their shilling attack estimator.

VI. CONCLUSIONS

Recommender Systems play an increasingly important role in people's decisions and opinions, for this reason there is a great interest in influencing their recommendations. Collaborative filtering is the most used machine learning approach to implement recommender systems. Recommender systems are subject to shilling attacks where malicious profiles are injected in order to affect the operation of the collaborative filtering engines.

Shilling attacks detection has been the traditional research field to tackle this issue, by filtering or removing malicious profiles. Nowadays research is heading towards the design of robust machine learning methods able to neutralize the malicious profiles impact. Our proposed approach provides a robust solution where instead of detecting shilling profiles it finds suspicious predictions, making it possible to avoid their promotion to final recommendations.

Whereas traditional detection of malicious profiles is based on statistical methods to find unusual rating patterns, our approach is based on a machine learning architectural design to find unusual variations of prediction reliabilities. This approach offers the advantage of dealing with shilling profiles that have passed the detection filters. It is also benefited from the semantic top-level abstraction that matrix factorization methods provide through their hidden factors. Particularly, our method is based on a two-level architecture based on matrix factorization.

A complete set of injection profiles experiments has been designed, covering the main shilling attacks approaches both in their push and nuke variations. Results from more than thirteen thousand individual experiments show: a) the designed shilling attacks correctly perform their function, b) the proposed method efficiently neutralize the great majority of the attacks, c) those attacks not entirely neutralized correspond to the most irrelevant ones: shilling attacks unable to significantly affect to the recommendation results, and d) the paper's hypothesis is fulfilled: by monitoring the unusual raising of prediction reliabilities we can avoid to promote suspicious predictions to inadequate recommendations.

Finally, the obtained results encourage to carry out a set of future works among which are: a) study of the obfuscate shilling attacks impact on the proposed method, b) to extend experiments to diverse collaborative filtering open datasets, c) to test vandalism attacks impact, and d) study of the balance between detection and robust counteraction, assigning the adequate weight to both approaches.

REFERENCES

[1] P. A. Chirita, W. Nejdl, and C. Zamfir, "Preventing shilling attacks in online recommender systems," in *Proc. 7th Annu. ACM Int. Workshop Web Inf. Data Manage.*, Bremen, Germany, Nov. 2005, pp. 67–74.

[2] R. Burke, B. Mobasher, C. Williams, and R. Bhaumik, "Classification features for attack detection in collaborative recommender systems," in *Proc. 12th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Philadelphia, PA, USA, Aug. 2006, pp. 542–547.

[3] S. Zhang, Y. Ouyang, J. Ford, and F. Makedon, "Analysis of a low-dimensional linear model under recommendation attacks," in *Proc. 29th Annu. Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, Seattle, WA, USA, Aug. 2006, pp. 517–524.

[4] Y. Wang, L. Quian, F. Li, and L. Zhang, "A comparative study on shilling detection methods for trustworthy recommendations," *J. Syst. Sci. Syst. Eng.*, vol. 27, no. 4, pp. 458–478, 2018.

[5] T. Kumari and P. Bedi, "A comprehensive study of shilling attacks in recommender systems," *Int. J. Comput. Sci. Issues*, vol. 14, no. 4, pp. 44–50, 2017.

[6] F. Ortega, B. Zhu, J. Bobadilla, and A. Hernando, "CF4J: Collaborative filtering for Java," *Knowl.-Based Syst.*, vol. 152, pp. 94–99, Jul. 2018.

[7] A. Hernando, J. Bobadilla, and F. Ortega, "A non negative matrix factorization for collaborative filtering recommender systems based on a Bayesian probabilistic model," *Knowl.-Based Syst.*, vol. 97, pp. 188–202, Apr. 2016.

[8] B. Mehta and W. Nejdl, "Unsupervised strategies for shilling detection and robust collaborative filtering," *User Model. User-Adapted Interact.*, vol. 19, nos. 1–2, pp. 65–97, 2009.

[9] J.-S. Lee and D. Zhu, "Shilling attack detection—A new approach for a trustworthy recommender system," *INFORMS J. Comput.*, vol. 21, no. 1, pp. 117–131, 2011.

[10] J. Cao, Z. Wu, B. Mao, and Y. Zhang, "Shilling attack detection utilizing semi-supervised learning method for collaborative recommender system," *World Wide Web J.*, vol. 16, nos. 5–6, pp. 729–748, 2013.

[11] Z. Wu, Y. Wang, Y. Wang, J. Wu, J. Cao, and L. Zhang, "Spammers detection from product reviews: A hybrid model," in *Proc. IEEE Int. Conf. Data Mining*, Atlantic City, NJ, USA, Nov. 2015, pp. 1039–1044.

[12] F. Zhang and Q. Zhou, "HHT–SVM: An online method for detecting profile injection attacks in collaborative recommender systems," *Knowl.-Based Syst.*, vol. 65, pp. 96–105, Jul. 2014.

[13] F. Zhang, Y. Lu, J. Chen, S. Liu, and Z. Ling, "Robust collaborative filtering based on non-negative matrix factorization and R1-norm," *Knowl.-Based Syst.*, vol. 118, pp. 177–190, Feb. 2017.

[14] H. Yu, R. Gao, K. Wang, and F. Zhang, "A novel robust recommendation method based on kernel matrix factorization," *J. Intell. Fuzzy Syst.*, vol. 32, no. 3, pp. 2101–2109, 2017.

[15] B. Mehta, T. Hofmann, and W. Nejdl, "Robust collaborative filtering," in *Proc. 1st ACM Int. Conf. Recommender Syst.*, Minneapolis, MN, USA, 2007, pp. 49–56.

[16] F. Zhang and S. Sun, "A robust collaborative recommendation algorithm based on least median squares estimator," *J. Comput.*, vol. 9, no. 2, pp. 308–314, 2014.

[17] Z. Cheng and N. Hurley, "Robust collaborative recommendation by least trimmed squares matrix factorization," in *Proc. 22nd IEEE Int. Conf. Tools Artif. Intell.*, Arras, France, 2010, pp. 105–112.

[18] B. Mobasher, R. Burke, and J. J. Sandvig, "Model-based collaborative filtering as a defense against profile injection attacks," in *Proc. 21st Nat. Conf. Artif. Intell.*, Boston, MA, USA, 2006, pp. 1388–1393.

[19] H. Polat and W. Du, "Privacy-preserving collaborative filtering," *Int. J. Electron. Commerce*, vol. 9, no. 4, pp. 9–35, 2005.

[20] M. Si and Q. Li, "Shilling attacks against collaborative recommender systems: A review," *Artif. Intell. Rev.*, vol. 42, no. 4, pp. 767–799, 2018.

[21] F. G. Zhang and S. H. Xu, "Analysis of trust-based e-commerce recommender systems under recommendation attacks," in *Proc. 1st Int. Symp. Data, Privacy, e-Commerce*, Chengdu, China, 2007, pp. 385–390.

[22] D. Jia, F. Zhang, and S. Liu, "A robust collaborative filtering recommendation algorithm based on multidimensional trust model," *J. Softw.*, vol. 8, no. 1, pp. 11–19, 2013.

[23] D. Jia and F. Zhang, "A robust collaborative recommendation algorithm incorporating trustworthy neighborhood model," *J. Comput.*, vol. 9, no. 10, pp. 2328–2335, 2014.

[24] S. Ray and A. Mahanti, "Improving prediction accuracy in trust-aware recommender systems," in *Proc. 43rd Hawaii Int. Conf. Syst. Sci.*, Kauai, HI, USA, 2010, pp. 1–9.

[25] Z.-J. Deng, F. Zhang, and S. P. S. Wang, "Shilling attack detection in collaborative filtering recommender system by PCA detection and perturbation," in *Proc. Int. Conf. Wavelet Anal. Pattern Recognit.*, Jul. 2016, pp. 213–218.

[26] F. Zhang, Z.-J. Deng, Z. M. He, X.-C. Lin, and L.-L. Sun, "Detection of shilling attack in collaborative filtering recommender system by PCA and data complexity," in *Proc. Int. Conf. Mach. Learn.*, Jul. 2018, pp. 673–678.

[27] K. Patel, A. Thakkar, C. Shah, and K. Makvana, "A novel supervised approach to detection of shilling attack in collaborative filtering based recommendation system," *Int. J. Comput. Sci. Inf. Secur.*, vol. 14, no. 4, pp. 208–211, 2016.

[28] W. Zhou *et al.*, "Shilling attacks detection in recommender systems based on target item analysis," *PLoS ONE*, vol. 10, no. 7, Jul. 2015, Art. no. e0130968.

[29] Z. Yang and Z. Cai, "Detecting abnormal profiles in collaborative filtering recommender systems," *J. Intell. Inf. Syst.*, vol. 48, no. 3, pp. 499–518, 2017.
[30] J. Nehriri and S. Hosseinalizadeh, "A new mechanism to improve the detection rate of shilling attacks in the recommender systems," *J. Inf. Technol. Manage.*, vol. 9, no. 4, pp. 871–892, 2017.
[31] H. Xia, B. Fang, M. Gao, H. Ma, Y. Tang, and J. Wen, "A novel item anomaly detection approach against shilling attacks in collaborative recommendation systems using the dynamic time interval segmentation technique," *Inf. Sci.*, vol. 306, pp. 150–165, Jun. 2015.
[32] W. Zhou, J. Wen, Q. Qu, J. Zeng, and T. Cheng, "Shilling attack detection for recommender systems based on credibility of group users and rating time series," *PLoS ONE*, vol. 13, no. 5, May 2018, Art. no. e0196533.
[33] J. H. Dhimmar and R. Chauhan, "An accuracy improvement of detection of profile-injection attacks in recommender systems using outlier analysis," *Int. J. Comput. Appl.*, vol. 122, no. 10, pp. 22–27, 2015.
[34] Y. Wang, Z. Wu, Z. Bu, J. Cao, and D. Yang, "Discovering shilling groups in a real e-commerce platform," *Online Inf. Rev.*, vol. 40, no. 1, pp. 62–78, 2016.
[35] W. Zhou, J. Wen, M. Gao, H. Ren, and P. Li, "Abnormal profiles detection based on time series and target item analysis for recommender systems," *Math. Problems Eng.*, vol. 2015, May 2015, Art. no. 490261.
[36] Y. Hao and F. Zhang, "Detecting shilling profiles in collaborative recommender systems via multidimensional profile temporal features," *IET Inf. Secur.*, vol. 12, no. 4, pp. 362–374, 2018.
[37] A. M. Turk and A. Bilge, "Robustness analysis of multi-criteria collaborative filtering algorithms against shilling attacks," *Expert Syst. Appl.*, vol. 115, pp. 386–402, Jan. 2019.
[38] X. Li, M. Gao, W. Rong, Q. Xiong, and J. Wen, "Shilling attacks analysis in collaborative filtering based Web service recommendation systems," in *Proc. IEEE Int. Conf. Web Services*, Jun. 2016, pp. 538–545.
[39] B. Zhu, F. Ortega, J. Bobadilla, and A. Gutiérrez, "Assigning reliability values to recommendations using matrix factorization," *J. Comput. Sci.*, vol. 26, pp. 165–177, May 2018.
[40] J. Bobadilla, A. Gutiérrez, F. Ortega, and B. Zhu, "Reliability quality measures for recommender systems," *Inf. Sci.*, vol. 442, pp. 145–157, May 2018.
[41] A. Hernando, J. Bobadilla, F. Ortega, and J. Tejedor, "Incorporating reliability measurements into the predictions of a recommender system," *Inf. Sci.*, vol. 218, no. 1, pp. 1–16, 2013.
[42] M. A. Mazurowski, "Estimating confidence of individual rating predictions in collaborative filtering recommender systems," *Expert Syst. Appl.*, vol. 40, no. 10, pp. 3847–3857, 2013.
[43] P. Moradi and S. Ahmadian, "A reliability-based recommendation method to improve trust-aware recommender systems," *Expert Syst. Appl.*, vol. 42, no. 21, pp. 7386–7389, 2015.
[44] J. Bobadilla, F. Ortega, A. Hernando, and A. Gutiérrez, "Recommender systems survey," *Knowl. Syst.*, vol. 46, pp. 109–132, Jul. 2013.
[45] J. Bobadilla, A. Hernando, F. Ortega, and A. Gutiérrez, "Collaborative filtering based on significances," *Inf. Sci.*, vol. 185, no. 1, pp. 1–17, 2012.
[46] M. Y. H. Al-Shamri, "User profiling approaches for demographic recommender systems," *Knowl.-Based Syst.*, vol. 100, pp. 175–187, 2016.
[47] J. Yu, M. Gao, W. Rong, Y. Song, and Q. Xiong, "A social recommender based on factorization and distance metric learning," *IEEE Access*, vol. 5, pp. 21557–21566, 2017.
[48] M.-L. Wu, C.-H. Chang, and R.-Z. Liu, "Integrating content-based filtering with collaborative filtering using co-clustering with augmented matrices," *Expert Syst. Appl.*, vol. 41, no. 6, pp. 2754–2761, 2014.
[49] Z. Yang, B. Wu, K. Zheng, X. Wang, and L. Lei, "A survey of collaborative filtering-based recommender systems for mobile Internet applications," *IEEE Access*, vol. 4, pp. 3273–3287, 2016.
[50] T. K. Paradarami, N. D. Bastian, and J. L. Wightman, "A hybrid recommender system using artificial neural networks," *Expert Syst. Appl.*, vol. 83, pp. 300–313, Oct. 2017.
[51] *MaliciousML: MovieLens Dataset Attacked With Diverse Profile Injection Attacks*. Accessed: Mar. 7, 2019. [Online]. Available: https://doi.org/10.21227/rcsd-h160

SANTIAGO ALONSO received the B.S. degree in software engineering from the Universidad Autónoma de Madrid and the Ph.D. degree in computer science and artificial intelligence from the Universidad Politécnica de Madrid, in 2015, where he is currently an Associate Professor, participating in master and degree subjects and doing work related with advanced databases.

His main research interests include natural computing (P-systems) and did some work on genetic algorithms. His current interests include machine learning, data analysis, and artificial intelligence.

JESÚS BOBADILLA received the B.S. degree in computer science from the Universidad Politécnica de Madrid and the Ph.D. degree in computer science from the Universidad Carlos III de Madrid. He was in charge of the FilmAffinity.com Research Team, involved in the collaborative filtering kernel of the website. He is currently a Professor with the Department of Applied Intelligent Systems, Universidad Politécnica de Madrid. He has been a Researcher with the International Computer Science Institute, University of California at Berkeley, Berkeley, CA, USA. He is a habitual author of programming languages books working with McGraw-Hill, Ra-Ma, and Alfa Omega publishers. His research interests include information retrieval, recommender systems, and speech processing.

FERNANDO ORTEGA was born in Madrid, Spain, in 1988. He received the B.S. degree in software engineering, the M.S. degree in artificial intelligence, and the Ph.D. degree in computer sciences from the Universidad Politécnica de Madrid, in 2010, 2011, and 2015, respectively.

From 2008 to 2015, he was a Research Assistant with the Intelligent Systems for Social learning and Virtual Environments Research Group. From 2015 to 2017, he was working in BigTrueData leading machine learning projects. From 2017 to 2018, he was an Assistant Professor with the U-tad: Centro universitario de tecnología y arte digital. Since 2018, he has been an Assistant Professor with the Universidad Politécnica de Madrid. He is the author of 30 research papers in most prestigious international journals. His research interests include machine learning, data analysis, and artificial intelligence. He leads several national projects to include machine learning algorithms into the society.

RICARDO MOYA was born in Madrid, Spain, in 1985. He received the B.S. degree in informatics engineering, the M.S. degree in science and technology of computing, and the Ph.D. degree in computer sciences from the Universidad Politécnica de Madrid, in 2011, 2013, and 2015, respectively. He works in Telefónica Investigación y Desarrollo in the area of the Chief Data Officer as the Data Scientist and the Technical Leader in the development and productization IA Solutions. He is also a Lecturer with the U-tad: Centro universitario de tecnología y arte digital, teaching subjects and directing projects related to the fields of data science and big data. He is the Founder and an Executive Member of the AI-Network Association.