Detection of Shilling Attacks on Collaborative Filtering Recommender Systems by Combining Multiple Random Forest Models

Vjeran Grozdanić*, Klemo Vladimir*, Goran Delač*, Marin Šilić*
* University of Zagreb, Faculty of Electrical Engineering and Computing, Unska 3, 10 000 Zagreb, Croatia
vjeran.grozdanic@fer.hr, klemo.vladimir@fer.hr, goran.delac@fer.hr, marin.silic@fer.hr

***Abstract*—Collaborative filtering recommender systems are one of the essential recommender systems and are still widely used in combination with other algorithms to make predictions for users. However, they are vulnerable to shilling attacks, and if there isn't any detection system to prevent those attacks, original recommendations can be heavily influenced to benefit the attackers. Designing attack-resistant recommendation systems is not an easy task, and many researchers have tried to tackle that problem. In this paper, a new approach that combines multiple random forest models is proposed. Each of the random forest models is specialized in detecting one group of shilling attacks, and then all the models are combined into an ensemble model. Experimental results show that the proposed ensemble is capable of detecting attack profiles at high rate without causing significant bias in the original recommendation system.**

Keywords*—*collaborative filtering, random forest, attack detection, shilling attacks, classification

I. Introduction

It's hard to imagine today's online world without the recommender systems. They have become essential part of the web applications like Netflix [1], Amazon [2] and many others. Some of the most used algorithms in recommender systems come from collaborative filtering (CF) group of algorithms. CF algorithms have shown great prediction quality both in academic research and in industrial applications and up to this day they still give state-of-the-art results when it comes to recommendation prediction results [3].

Given the wide use of CF methods in the industry recommender systems, it is not surprising to see increase in malicious activities and attacks against CF algorithms [4]. With properly executed attack, one can significantly increase the ratings and the popularity of the chosen product, during a *push* attack, or if wanted, one can significantly decrease the popularity of the target product which is performed with a *nuke* attack.

Without the proper attack detection system incorporated into the recommender system, it is easy to manipulate the system through various attacks. For that reason, a lot of focus has been put on designing robust recommender systems capable of ignoring the impact of the attack. One of the first approaches in attack detection was ad-hoc algorithm [5] which was used to classify fake profiles. Later approaches started using various machine learning algorithms to tackle the attack detection problem.

Item 1	...	Item K	...	item N	...	Target item
r_1	...	r_k	...	r_n	...	r_t
Selected items		Filler items		Unrated items		Target item(s)

TABLE I: Visual representation of the attack profile

Recent work has been very successful in coping with the problems of attack detection and prevention. Many different approaches has been tried to create as robust systems as possible [6]. Clustering approach [7] [8] [9] is designed believing that fake profiles will eventually end up in the same cluster or a group after the clustering algorithm is applied. Probabilistic approach [10] [11] focuses on using the probabilistic methods to detect fake profiles. Classification approach [12] [13] uses extracted features to decide if the profile is fake or authentic.

II. Shilling Attacks

Attacks on the collaborative filtering methods are called *profile injection attacks*, because they are done by injecting fake profiles into the recommender system, thus causing the bias in the recommendation that the system produces. Also, a very popular name for this kind of attacks is shilling attacks [14], where the origin of the name addresses how cheap they are to perform, because the attacks can easily be automated and executed only from attacker's computer.

There are multiple ways in which the attacks can be categorized. One of the biggest factors of the attack is the intent, does the attacker want to increase the popularity of a target item or decrease it. When the goal of the attack is to increase the popularity of the item, and along with that the number of times the item will be recommended, it is called push attack, because the popularity is being pushed. On the other side, there are nuke attacks, where the goal of the attack is to decrease the popularity of an item.

Shilling attacks are categorized by the strategies the attacker would use while creating the fake profiles. Each attack profile (visible in a table I) is made out of 4 different sets of items and grades for each of the item and those sets are:

- **Target item(s)** - set which contains only the item(s) which is being attacked
- **Selected items** - set of carefully selected items to increase the reach of the attack
- **Filler items** - set of randomly selected items which are used to make attack profiles bigger and to add some noise into the attack
- **Unrated items** - set containing all the items available in the system for which the attacker hasn't provided the rating

Every attack can be described by the strategies used while generating the ratings for each of the sets of the attack profile. One of the most basic attacks proposed in [14] is random attack which is performed by generating random rating for every item in both selected items and filler items sets, and this attack is easy to perform since the attacker doesn't need deep knowledge about the system besides what items are in it.

Other, way more effective attack also proposed in [14] is average attack which fills the selected items set and filler items set with a system-wide mean rating of the specific item. Performing the average attack requires more knowledge about the system because the attacker needs to know the item's mean rating in the system, but often that information is publicly available. Both random and average attacks can be used as either push or nuke attacks, the only difference is if the target item is going to get maximum rating (push attack) or minimum rating (nuke attack).

There are many more variants of shilling attacks, but only one more is important for this paper, and that is a bandwagon attack. It is an attack designed to capitalize on popular items in the systems to expand the impact of the attack. During the attack, the attacker carefully selects k most popular items in the system and assigns maximum rating to them, those items are part of the selected items set, and for the filler items set, it randomly selects the items and the ratings for them. Random attack could be considered a special case of the bandwagon attack when the k is set to 0.

Attack on the CF recommender system is also determined by the attack *profile size*, the number of the items attacker has rated, and the *attack size*, usually shown as a percentage of the fake profiles in the system, which were injected by the attacker in the system.

III. CLASSIFICATION MODEL

For a good classification results, the good classification attributes are needed. In [15] were proposed both generic and model-specific attributes which could be used for the classification of fake profiles. Generic attributes are the ones which are used for every different variant of attack, while model-specific attributes were proposed as a result of careful inspection of how each of the attacks is performed and those attributes have significant importance when detecting specific attacks for which they were designed.

A. Classification Attributes

Classification attributes used in this paper have all been taken from [15]. When there are a lot of items rated from a single user, it is not very probable that an authentic user is behind that profile. Driven by that idea, the *length variance*(L_u) was proposed:

$$L_u = \frac{|n_u - \overline{n_u}|}{\sum_{u \in U}(n_u - \overline{n_u}^2)} \tag{1}$$

where n_u is the length of the profile of the user u, $\overline{n_u}$ is the average length of the user profile in the system and U is a set of all users.

Another proposed attribute was *weighted deviation from mean agreement* (WDMA) which is based on *rating deviation from mean agreement* (RDMA), but it puts higher emphasis on the rating deviations for sparse items.

$$WDMA_u = \frac{\sum_{i=0}^{n_u} \frac{|r_{u,i} - \overline{r_i}|}{l_i^2}}{n_u} \tag{2}$$

In (2) n_u is the length of the profile of the user u, $r_{u,i}$ is the rating user u has given to the item i, $\overline{r_i}$ is the mean rating for item i, and l_i is a number of users which have rated item i.

Similar to WDMA, *weighted degree of agreement* (WDA) is defined with the difference that here the size of the user profile isn't important. WDA can be calculated with the following formula:

$$WDA_u = \sum_{i=0}^{n_u} \frac{|r_{u,i} - \overline{r_i}|}{l_i} \tag{3}$$

where n_u is the length of the profile of the user u, $r_{u,i}$ is the rating user u has given to the item i, $\overline{r_i}$ is the mean rating for item i, and l_i is a number of users which have rated item i.

Standard deviation of the user's ratings can also help distinguish fake and authentic profiles:

$$\sigma_u = \sqrt{\frac{\sum_{i=0}^{n_u}(r_{u,i} - \mu_u)^2}{n_u}} \tag{4}$$

In (4) n_u is the number of ratings user u has given, $r_{u,i}$ is the rating user u has given to the item i, while μ_u is the mean rating of user u.

The idea that fake profiles are closer to their most similar profiles than the authentic ones is captured in the degree of similarity attribute:

$$DegSim_u = \frac{\sum_{v \in neighbors(u)} W_{u,v}}{k} \tag{5}$$

where k is the number of the closest neighbors for user u, and $W_{u,v}$ is the similarity function between user u, and it's neighbor v.

All the attributes mentioned above are generic attributes, but in order to tackle with average attack, model specific attribute was also designed in [15]. First, the set of ratings that are potential targets is defined as $P_{u,T} = \{i \in P, where\ r_{u,i} = r_{max}\}$ (or $where\ r_{u,i} = r_{min}$ for nuke attacks). P_u represents profile of user u, $P_{u,F}$ represents rest of the user profile and it is calculated as $P_{u,F} = P_u - Pu, T$. Then a *mean variance* can be calculated using the following formula:

$$MeanVar_u = \frac{\sum_{i \in P_{u,F}} (r_{u,i} - \overline{r_i})^2}{|P_{u,F}|} \tag{6}$$

where $r_{u,i}$ is the rating user u has given to the item i, and $\overline{r_i}$ is the mean rating for item i. For the attack profiles during average attack, this value is expected to be very low comparing to the authentic profiles.

B. Influence of the attack

In order to tell how effective an attack is, we need to be able to measure the influence of it. There are two well known methods for measuring that influence.

First method, a prediction shift, which tells how much has the mean predicted rating of the item shifted after the attack. Let $p(u, i)$ be a predicted rating which the user u would give to the item i before the attack, and let the $p'(u, i)$ be a predicted rating which the user u would give to the item i after the attack has happened. Then, we can calculate prediction shift using the following formula:

$$\Delta p = \frac{\sum_u^U |p(u,i) - p'(u,i)|}{n} \tag{7}$$

where n is the number of the users in set U.

Using just one method for measuring the effect of the attack could be misleading, that's the reason why often, along with the prediction shift, another method is used, a target *hit ratio*. The second method is calculated by creating top-n predictions for every profile in the system before and after the attack, and then count for how many users has the target item occurred in the top-n recommendations. The influence of the attack can then be seen by comparing values of the target hit ratio before and after the attack.

C. Ensemble model

For the approach proposed in this model, a basic idea was to build independent classificator for each type of the shilling attacks and then create an ensemble model from all of those clasificators and implement single-vote voting classification, where it is enough if one of the classificators would mark the profile as fake, for the ensemble model to classify the profile as fake. This idea has some shortcomings, major one is that it is impossible to cover all the attacks with different classifiers and for each attack out there train a specific classifier, so after closer examination and experimenting, we can notice that it is possible to group some attacks together. For example, random attack is just a special case of the bandwagon attack when k is set to 0, and both variants of push and nuke random attacks can be put in that same group.

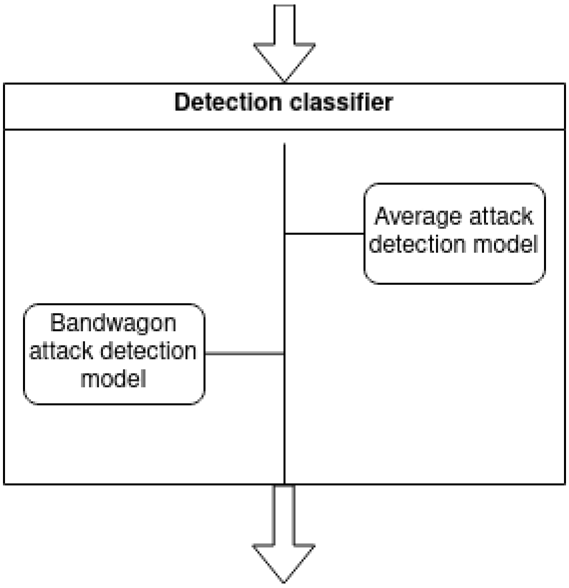
Detection classifier
Average attack detection model
Bandwagon attack detection model

Fig. 1: Architecture of proposed classifier model with single-vote voting classification

Also, using the same logic, both push and nuke average attacks can be put together in the same group and only one classifier will be enough to detect fake profiles. For each group of the attacks, the random forest [16] classifier is used to detect fake profiles, and after grouping the attacks mentioned in this paper, only two detection classifiers are needed. Each random forest classifier is trained independently, without knowing about the existence of the other classifier(s).

Features mentioned above were used to train random forest clasifiers, and classifiers have been trained using the cross-validation [17] to pick the best model for that group of the attacks and avoid overfitting.

Architecture of the proposed model can be seen in the picture 1.

IV. RESULTS

To measure the efficiency of the attack detector, we have picked a standard measure, the F_1 score, which is the harmonic mean of the precision and recall.

As a base testing and training dataset, user ratings from MovieLens 100k [18] were used. The dataset was split in 70-30 train-test ratio, and then two different types of training datasets were created. The first type of datasets contained random push and nuke attacks, and bandwagon attacks (push attack). The second type of datasets contained push and nuke average attacks.

For all the created attacks, filler items size and attack size were varied for each attack. This was done to improve generalization of the detector and to make it as robust as possible when detecting attacks. For filler set and attack sizes, sizes of 1%, 3%, 5%, and 10% were selected in order to create attack datasets, and then all of those attack

	F1	Precision	Recall
Test set	0.91	0.83	1.0

TABLE II: Results F_1, recision, and recall scores for the proposed detection model

datasets were combined with base dataset to create final training set.

The training set for each model contained 25% of authentic profiles, and 75% of fake profiles coming from different variations of attacks. That decision made the training set unbalanced, but it was chosen to put more emphasis on recall since we wanted to create a classifier which would detect as much as possible fake profiles to not let the attack influence the recommender system at all. If the classifier trained on the training set like this one is used in the production, it should not delete the profiles marked as fake ones because it might result in authentic profiles being deleted from the system.

In table II F1, precision and recall scores are shown for the proposed model, it is possible to notice a very high recall, equal to 1, but this is the consequence of the detection system design. Recall of 1 is not unusual to see when measuring the efficiency of the recommender systems and in many papers [10] [19] [20] it is possible to find the recall values close or equal to 1.

Table III shows how each attack influences the same recommender system with and without attack detection model. It is possible to see how big of a difference relatively simple detection system can make and why it is needed to have a proper attack detection systems.

In table IV the comparison between proposed and two other models is shown. In [21] Bayes classifier is used to detect shilling attacks, while in [22] authors proposed unsupervised PCA model to detect shilling attacks. The [21] was selected for comparison because in 2018. it has outperformed state-of-the-art models, and [22] was selected because it shares multiple detection attributes with the model proposed in this paper.

Comparison between models is done on the test set in which attack profiles have filler set size set to 5%, and attack size is also set to 5%, and comparisons were made for the three different attacks: random attack, average attack and bandwagon attack. When comparing the models, proposed method has higher F_1 score for two out of three chosen attacks. It is worth mentioning that for some other attack parameters (different filler set size or attack size), Bayes classifier has greater F1 score than the proposed model, and that unsupervised PCA model has much higher F_1 scores when the attack size is greater (10% or 20%).

		Mean predicted rating	Hit ratio
Clear test set		3.16	6.9
Random attack (push)	Without detector	4.47	207.2
	With detector	3.18	6.7
Average attack (push)	Without detector	4.55	218.9
	With detector	3.17	6.8
Bandwagon attack	Without detector	4.49	206.4
	With detector	3.16	6.7
Random attack (nuke)	Without detector	1.74	0.0
	With detector	3.18	6.8
Average attack (nuke)	Without detector	1.62	0.1
	With detector	3.18	6.8

TABLE III: Display of the influence of the attacks on the recommender system with and without attack detector

	Proposed model	Bayes classifier	Unsupervised PCA
Random attack	**0.91**	0.89	0.76
Average attack	0.91	**0.95**	0.66
Bandwagon attack	**0.94**	0.86	0.75

TABLE IV: Comparison of F1 score between proposed model, Bayes classifier [21] and Unsuprivised PCA classifier [22] when both filler set and attack size are 5%

V. Conclusion and future work

Without the proper attack detection system, even the simplest attacks can have very high influence on the recommendations of the system. In this paper, novel approach has been proposed with the idea of grouping similar attacks together and training specific classifiers for that group of the attacks. The proposed method uses random forest algorithm to detect fake profiles for each group of the attacks, and has shown promising results.

For the future work, more groups of attacks should be added and some other classifier algorithms could be explored and tested.

References

[1] J. Bennett and S. Lanning, "The netflix prize," in *Proceedings of the KDD Cup Workshop 2007*. New York: ACM, Aug. 2007, pp. 3–6. [Online]. Available: http://www.cs.uic.edu/~liub/KDD-cup-2007/NetflixPrize-description.pdf

[2] B. Smith and G. Linden, "Two decades of recommender systems at amazon.com." *IEEE Internet Comput.*, vol. 21, no. 3, pp. 12–18, 2017. [Online]. Available: http://dblp.uni-trier.de/db/journals/internet/internet21.html#SmithL17

[3] Y. Koren and R. M. Bell, "Advances in collaborative filtering." in *Recommender Systems Handbook*, F. Ricci, L. Rokach, B. Shapira, and P. B. Kantor, Eds. Springer, 2011, pp. 145–186. [Online]. Available: http://dblp.uni-trier.de/db/reference/rsh/rsh2011.html#KorenB11

[4] I. Gunes, C. Kaleli, A. Bilge, and H. Polat, "Shilling attacks against recommender systems: A comprehensive survey." *Artificial Intelligence Review*, vol. 42, no. 4, 2014.

[5] P.-A. Chirita, W. Nejdl, and C. Zamfir, “Preventing shilling attacks in online recommender systems,” in *Proceedings of the 7th annual ACM international workshop on Web information and data management*, 2005, pp. 67–74.

[6] F. Rezaimehr and C. Dadkhah, “A survey of attack detection approaches in collaborative filtering recommender systems,” *Artificial Intelligence Review*, vol. 54, pp. 2011–2066, 2021.

[7] R. Bhaumik, B. Mobasher, and R. Burke, “A clustering approach to unsupervised attack detection in collaborative recommender systems,” in *Proceedings of the International Conference on Data Science (ICDATA)*. Citeseer, 2011, p. 1.

[8] W. Zhou, J. Wen, M. Gao, L. Liu, H. Cai, and X. Wang, “A shilling attack detection method based on svm and target item analysis in collaborative filtering recommender systems,” in *Knowledge Science, Engineering and Management: 8th International Conference, KSEM 2015, Chongqing, China, October 28-30, 2015, Proceedings 8*. Springer, 2015, pp. 751–763.

[9] L. Yang, W. Huang, and X. Niu, “Defending shilling attacks in recommender systems using soft co-clustering,” *IET Information Security*, vol. 11, no. 6, pp. 319–325, 2017.

[10] C. Li and Z. Luo, “Detection of shilling attacks in collaborative filtering recommender systems,” in *2011 International Conference of Soft Computing and Pattern Recognition (SoCPaR)*. IEEE, 2011, pp. 190–193.

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

[12] C.-Y. Chung, P.-Y. Hsu, and S.-H. Huang, “βp: A novel approach to filter out malicious rating profiles from recommender systems,” *Decision Support Systems*, vol. 55, no. 1, pp. 314–325, 2013.

[13] Z. Yang, Z. Cai, and X. Guan, “Estimating user behavior toward detecting anomalous ratings in rating systems,” *Knowledge-Based Systems*, vol. 111, pp. 144–158, 2016.

[14] S. K. Lam and J. Riedl, “Shilling recommender systems for fun and profit,” in *Proceedings of the 13th international conference on World Wide Web*, 2004, pp. 393–402.

[15] R. Burke, B. Mobasher, C. Williams, and R. Bhaumik, “Classification features for attack detection in collaborative recommender systems,” in *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2006, pp. 542–547.

[16] T. K. Ho, “Random decision forests,” vol. 1, pp. 278–282, 1995.

[17] A. Ng, “Preventing "overfitting" of cross-validation data,” *Proceedings of the Fourteenth International Conference on Machine Learning*, 01 1998.

[18] F. M. Harper and J. A. Konstan, “The movielens datasets: History and context,” *Acm transactions on interactive intelligent systems (tiis)*, vol. 5, no. 4, pp. 1–19, 2015.

[19] F. Zhang and Q. Zhou, “A meta-learning-based approach for detecting profile injection attacks in collaborative recommender systems.” *J. Comput.*, vol. 7, no. 1, pp. 226–234, 2012.

[20] Q. Zhou and F. Zhang, “A hybrid unsupervised approach for detecting profile injection attacks in collaborative recommender systems,” *Journal of Information & Computational Science*, vol. 9, no. 3, pp. 687–694, 2012.

[21] F. Yang, M. Gao, J. Yu, Y. Song, and X. Wang, “Detection of shilling attack based on bayesian model and user embedding,” in *2018 IEEE 30th International Conference on Tools with Artificial Intelligence (ICTAI)*. IEEE, 2018, pp. 639–646.

[22] 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 *2018 International Conference on Machine Learning and Cybernetics (ICMLC)*, vol. 2. IEEE, 2018, pp. 673–678.