Multi-Armed-Bandit-based Shilling Attack on Collaborative Filtering Recommender Systems

Agnideven Palanisamy Sundar[2†], Feng Li[1†], Xukai Zou[2†], Qin Hu[2†] and Tianchong Gao[3⋆]

[1]Department of Computer and Information Technology
[2]Department of Computer and Information Science
[3]Department of Cyberspace Security
[†]Indiana University-Purdue University Indianapolis, Indianapolis, IN, USA.
[⋆]Southeast University, Nanjing, Jiangsu, China.
{fengli,xzou}@iupui.edu, {agpalan,qinhu}@iu.edu, tgao@seu.edu.cn

***Abstract*—Collaborative Filtering (CF) is a popular recommendation system that makes recommendations based on similar users' preferences. Though it is widely used, CF is prone to Shilling/Profile Injection attacks, where fake profiles are injected into the CF system to alter its outcome. Most of the existing shilling attacks do not work on online systems and cannot be efficiently implemented in real-world applications. In this paper, we introduce an efficient Multi-Armed-Bandit-based reinforcement learning method to practically execute online shilling attacks. Our method works by reducing the uncertainty associated with the item selection process and finds the most optimal items to enhance attack reach. Such practical online attacks open new avenues for research in building more robust recommender systems. We treat the recommender system as a black box, making our method effective irrespective of the type of CF used. Finally, we also experimentally test our approach against popular state-of-the-art shilling attacks.**

I. Introduction

Since the beginning of the information age, there has been an exponential increase in data uploaded to the internet. This growth has also led to a drastic rise in e-commerce and content curation websites. Websites use Recommender Systems(RS) to ensure customer satisfaction. RS is an information filtering system that functions by using existing data to calculate and predict possible future outcomes. It can be broadly classified into two types: Content-based Filtering [1] and Collaborative Filtering [2], based on the features used to make predictions.

Content-based filtering, as the name suggests, makes recommendations by comparing the contents of the item to the users' profile. But, over-specialization causes all the recommendations to be similar to the items previously consumed by the user.

The core idea behind Collaborative Filtering (CF) is that similar users have similar preferences. Typically, the users and items are represented in the form of a User-Item matrix, also known as a Utility matrix. The utility matrix is a sparse matrix, with entries only where the users have rated the items. The remainder ratings of the sparse matrix are calculated based on the top N users who are similar to the concerned user. Recommendations are made from the calculated ratings. This simplicity and effectiveness have led to CF being prone to shilling attacks.

Shilling Attack/Profile Injection Attack introduced in [3] is a particular type of attack, where a malicious user (attacker) introduces a series of fake user profiles into the collaborative filtering system to push a target item. The fake profiles also rate other items with a motivation to be similar to many authentic user profiles, causing the target to be pushed further. There are various shilling attack methods like in [3]–[5] and detection techniques like in [6]–[8] that have come into existence in the past years.

The downside of most existing attacks is that their performance varies drastically with the type of CF algorithm used. Though these attacks work in offline evaluations, they cannot be used to execute an effective real-world attack. Moreover, these are single-time attacks; all the fake profiles and all their ratings are injected at once. Most attack schemes do not get feedback from the recommender system to assert the efficacy of the attack.

To overcome these drawbacks, we propose an online shilling attack scheme with high-efficiency under different CF algorithms. For an attack to work under multiple algorithms, the items selected by the attack should suit the particular algorithm under consideration, and cannot be the same for all systems. But, there is a high degree of uncertainty associated with choosing the most optimal items without knowing the algorithm. To tackle this problem, we develop a Multi-Armed-Bandit-based item selection process that uses the recommender system's feedback. We inject observer profiles, exclusively to understand and categorize the recommendations made by the system. We use these recommendations to reduce uncertainty categorically while simultaneously extending the attack reach.

This paper proposes an online attack method that aims to be efficient with different types of collaborative filtering methods used. The contributions of our work are the following:

- We design an online attack scheme that treats the recommender system as a black-box, knowing what the system is capable of doing but not the algorithm behind it.

This is the author's manuscript of the article published in final edited form as:

Sundar, A. P., Li, F., Zou, X., Hu, Q., & Gao, T. (2020). Multi-Armed-Bandit-based Shilling Attack on Collaborative Filtering Recommender Systems. 2020 IEEE 17th International Conference on Mobile Ad Hoc and Sensor Systems (MASS), 347–355. https://doi.org/10.1109/MASS50613.2020.00050

• We employ a multi-armed bandit-based approach for selecting the most optimal items to enhance the attack performance.
• We inject observer profiles to get the recommendations made by the CF and use these recommendations to extend attack reach.

II. Background

A. Understanding Collaborative Filtering

The function of a recommender system is to suggest items that may be of interest to a website's users. This suggestion is based on the other items that the users have rated or purchased on the website. The recommender system intends to make these recommendations to let the users explore items that may have been otherwise missed. For instance, users may be recommended with movies they have never heard of, based on the other movies they have rated. At the same time, a recommender system also suggests items that a user might need, reducing the effort needed to find it. For example, batteries are recommended to users with a flashlight in their online shopping cart, making the user experience more pleasant and easier.

Collaborative Filtering: It is the most commonly used system in practice. It can be broadly classified as User-User-based and Item-Item-based.

1) User-User-based CF [9]: This CF works by finding users who have purchased/rated a similar set of items and recommends the items purchased by one user to the other. To illustrate, if users u_1 and u_2 purchased items i_1, i_2 and i_3, then these two users are considered to be similar to each other. When user u_1 purchases another item i_4, then this item will be recommended to user u_2 based on user-user CF.

2) Item-Item-based CF [10]: This type of CF forms relationships between items based on how often those items are purchased together. Consider that the item sets $\{i_1, i_2, i_3\}$, $\{i_1, i_2, i_4\}$, and $\{i_1, i_2, i_5\}$ are purchased by users u_1, u_2, and u_3 respectively. When another user u_4 purchases item i_1, then item i_2 will be recommended to the user. If two items are found together frequently in the purchase history of multiple users, then those items will be strongly related in the item-item based CF.

Collaborative filtering can be interpreted as a way to extract relationships and similarities based on how users interact with items in an online platform. Fig. 1 illustrates the difference in the outcome of the two types of CF.

B. Developments in Collaborative Filtering

In recent years, with the need to improve the recommenders' performance, online platforms have been including newer features into the CF process [11]. These features are modified according to the platform's needs. Some of those features are:
• The similarity of items depends on the probabilistic association between items. In other words, the number of items purchased by a customer is also taken into account while calculating the similarity.

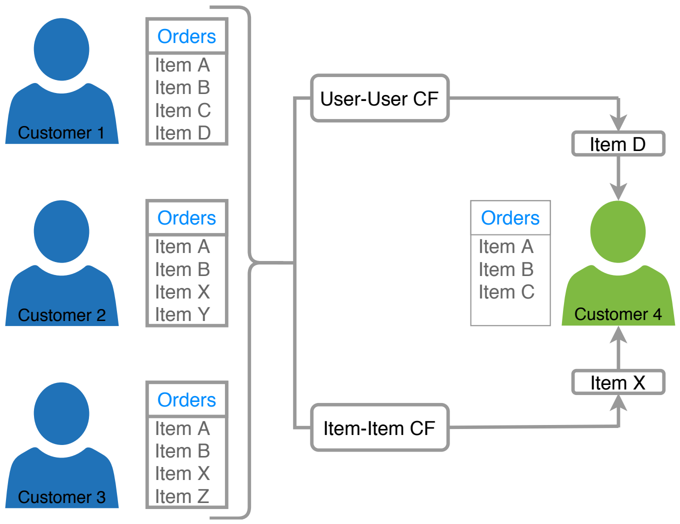

Fig. 1: A toy example of user-user and item-item collaborative filtering methods.

• The similarity between items also depends on the period of the purchase/rating of the items. The items bought months apart from each other will have far less similarity score than items which are purchased on the same day.
• Sometimes, the recommender system recommends a variety of moderately related items than a narrowly targeted list. For instance, if a user only purchases books, sometimes non-books are also recommended to the user.

Intuitively, not all online platforms need these added features to be better, but there is no guarantee of whether such elements are present in a CF system. But these changes do affect the way items are recommended; each recommendation includes a fraction of the CF's hallmark.

C. Objective of a Shilling Attack

A shilling attack is a method of introducing many fake profiles into a recommender system to alter the outcome of the recommendation process. These fake profiles generate unauthentic ratings to several items. They are causing the recommender system to work in the attacker's favor. Shilling attacks are executed either to promote (push) a target item that the attacker is favoring or to demote (nuke) a rival item.

Instinctively, we know that introducing many fake profiles would be beneficial in expanding the reach of an attack. But, it also increases the cost of the attack, as well as the risk of being detected. Here, the cost would include resources like time, computing power, and the money involved in executing the attack. So, an attacker aims to achieve the maximum reach possible for a given cost.

D. Factors Influencing the Effectiveness of a Shilling Attack

The attackers have two significant factors that can be modified: the number of profiles injected and items rated by these profiles.
Attack Size: Attack size is the number of fake profiles injected by the attacker. The larger the attack size, the better the reach

of the attack but the cost involved will also exponentially increase.

Choice of Rated Items (Selected Items): Items to be rated by the injected profiles are selected in such a way that-

• The attack profiles have maximum similarity with other authentic users in a User-User based CF system.

• The target items have a high degree of co-occurrence with many of the other items in an Item-Item based CF system.

The attacker does not know the type of CF used in most of the cases. The existing attack schemes use offline strategies to choose these selected items. Moreover, other factors, like the developments we discussed earlier, also impact the attack but are unknown to the attacker.

III. Methodology

In our method, we focus on creating an online attack, utilizing the CF's recommendation feedback while concurrently attaining maximum efficiency for a given attack size. We show that our method is system-agnostic by treating the CF as a black box; the internal parameters and the CF algorithm used are unknown to the attacker. The attacker can only rate/review the items and view the recommendations made to him by the system. To make the attack online, we deploy a continuous attack strategy, where the selected items are added over a more extended period.

Owing to the different CF algorithms and the newer developments in CF, it is evident that each recommendation made by the system has imbibed the essence of the entire CF system. The best way to subsume the recommendations into the attack is by adding these recommended items to the fake profile's selected items list. To get the recommendations, we exclusively create multiple fake profiles (observer profiles) and fill them with some items appropriate for the target item.

Not all recommendations made by the system adds value to the attack, but only the ones relevant to the target items. This criterion leaves an *uncertainty* associated with choosing the most optimal recommended items from all the profiles created. Only a limited number of items can be added to the selected items for a given cost. Such a limitation leads to making two modifications to reduce this uncertainty.

A. First, instead of treating each profile individually, grouping them into categories, makes it easier to reduce the uncertainty as a group: Categorizing Injected Profiles.

In most of the existing works, only the attack profiles are injected into the recommender system, but in our work, we inject two types of profiles into the system.

1) Attacker Profile: The attacker profile is the major part of the injected profiles used for promoting the target item to as many authentic users as possible. The profile has three types of rated items.

- Target Item: The item which needs to be promoted or demoted.
- Selected Items: The set of items that are rated to increase the reach of an attack.
- Filler Items: The items which are rated to camouflage the presence of an attack profile among authentic profiles to avoid detection.

2) Observer Profile: These profiles are injected to learn from the recommender system. The observer profiles use the recommendations made by the system to populate the attacker profiles.

- Selected Items Subset: These are the items that are a randomly chosen subset of the attacker profile's initial selected items.
- Random Items: These items are randomly chosen from the entire website's list of items.
- Recommended Items: These are the items that are recommended to the observer profiles after the subset of selected items and random items have been rated.

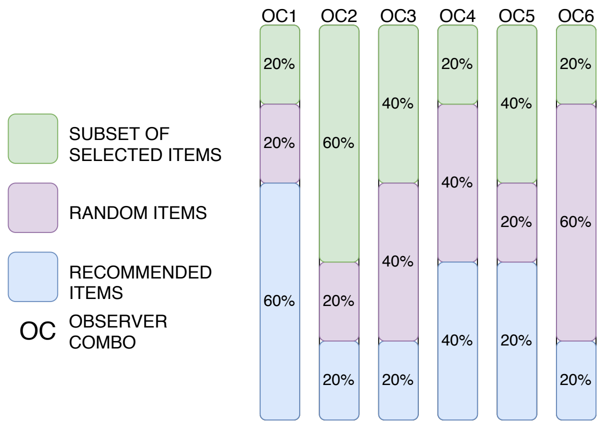

Fig. 2: A toy example of possible observer profile combos.

The observer profiles are divided into multiple combinations, with each combination having an equal number of profiles. The combinations differ from each other based on the ratio of selected subset, random, and recommended items. Fig. 2 illustrates six of the possible observer profile combos. This figure is only an example of the possible combinations.

B. Second, to ensure high efficiency, the uncertainty reduction process needs to happen simultaneously with the item selection proces: Multi-Armed-Bandit-based Uncertainty Reduction.

In MAB, a fixed number of resources need to be apportioned among multiple opposing choices in a way such that the overall gain is maximized [12]. The individual reward associated with each of the options is not known at the beginning. This problem models the exploration vs. exploitation dilemma. The name comes from a gambler having to choose the right one-armed bandit, or slot machine, to play from a row of bandits with varying rewards. The gambler has to select the number of plays in each bandit as well as the sequence of play. In scientific research, MABs are used in pharmaceutical trials, information retrieval, and even recommender systems. There are many optimal solutions for the MAB problem, but we are only using *Thompson Sampling* [13] approach in our method. Epsilon Greedy bandit algorithm undertakes a random exploration strategy which is not suitable for our work. Upper Confidence Bound acts under the optimistic assumption that the selection made has the highest possible reward and expends on exploring other options to decrease uncertainty. Thompson Sampling, on the other hand, is solely bayesian and is more suitable for our work.

Thompson Sampling uses the concept of probability and depends on the Beta distribution to make each selection.

$$\beta(x) = \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} x^{\alpha-1}(1-x)^{\beta-1} \tag{1}$$

Here, α value is increased by 1 for a win, and β value is increased by 1 for a loss after each iteration.

In the shilling attack process, MAB can be used to balance the exploration vs. exploitation with uncertainty in the item selection process. Each of the observer profile combos act as individual arms in the MAB. With each trial of the MAB, the uncertainty associated will the selected observer combo gets reduced. The probability of success replaces this uncertainty in each combo. If the recommended item is relevant to the target item, it is considered a win. If the recommended item is either irrelevant or is already part of the attacker profile, then it is a loss. Through multiple iterations, the MAB eventually starts exploiting the observer combo that makes the most useful recommendations.

C. Rating Scheme

For each new item added to the attacker profile or the observer profile, a rating needs to be included. For the simplicity of discussion, we pick the two most common rating scales as examples. If the system uses the 0 - 1 rating scale, then the ratings given to all the items should be 1. If the 1 - 5 rating scale is used in the system, then the rating provided by the attacker profile for an item should be equivalent to the average rating of the item in that platform. The rating given by the observer profile should be similar to the system mean rating. These details are readily available in most of the online platforms.

IV. MAB-based Shilling Attack Scheme

In this section, we explain the scheme in which the online attack is executed. Our attack scheme is categorized into three phases for ease of understanding.

A. Setup Phase

The selection of the items for the initial attacker profile and observer profiles constitute the setup phase. The attacker profile mostly consists of the essential selected items, which are chosen as per the target item. The most popular items which are similar to the target item are chosen to be the selected items. For example, if the target item is a sci-fi novel, then the selected items should be set of the most famous novels.

We want the initial items in the attack profile to be 70% filled with the selected items at the setup phase. The rest of the items are filler items chosen at random from the entire item set. This mix of items ensures that the attacker profile is very similar to the target item while also evading detection because of filler items. The target item is not introduced during the setup phase.

Algorithm 1: Iterative and Termination Phase.

```
Input  : Attacker and Observer profiles formed from
         Setup Phase.
         Assume $\alpha$ = 1 and $\beta$ = 1 for all
combos(bandits)
Output: Efficient Attacker and Observer profiles
Iterative Phase:
    MAB Selection Process:
            •Use eqn.1 to get beta distribution.
            •Randomly sample a value from the
             probability density function beta
             distribution of all the combos;
            •Select the combo with maximum sampled
             value;
    Check recommendations for first observer profile in
      selected observer combo;
    if Recommended item is related to Target item then
        MAB Combo Reward = 1;
        $\alpha$ = $\alpha$ + 1
        Add item to all Attacker Profiles;
    else
        MAB Combo Reward = 0;
        $\beta$ = $\beta$ + 1
    end
    Add item to current observer profile;
    Move current observer profile to the end of queue
      in observer combo;
Termination Phase
    if New Items Count <= Batch Threshold then
        goto 1;
    else
            if Attacker Profile Length < Filler Size then
                Terminate one batch of attack profiles;
                Reset New Items Count;
                Add Target item to terminated batch;
                goto 1;
            else
                Attack Size Reached;
                Terminate Attack;
            end
    end
return Fully-populated Attacker and Observer profiles
```

The observer profile has fewer items than the attacker profile. The number of combinations that can be used in an attack is dependent on the size of the attack. If many profiles are injected, and the number of items that can be rated by each profile is higher, then more observer combinations can be explored.

Having multiple observer combos means getting a better

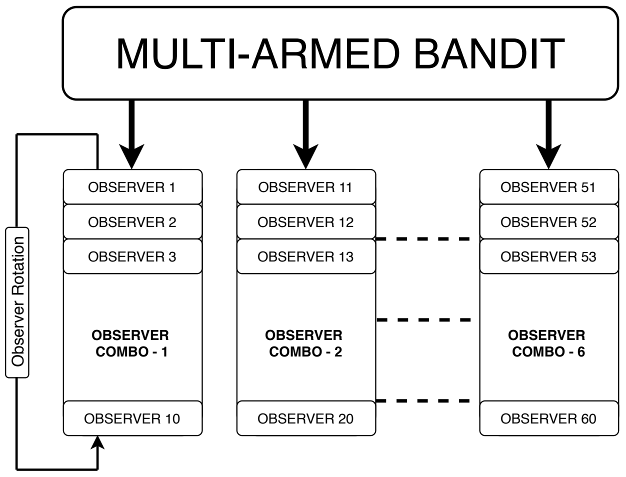

Fig. 3: Using MAB for observer combo selection.

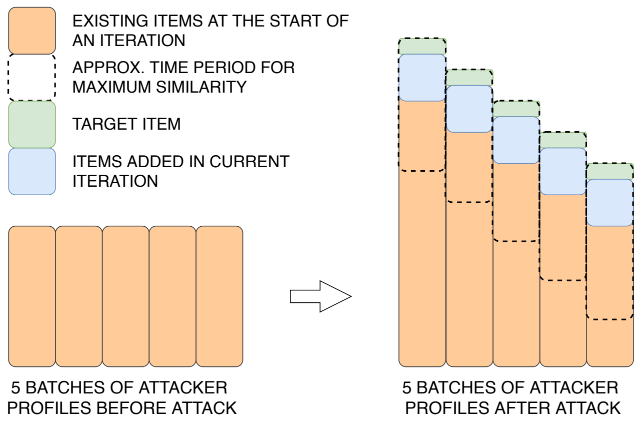
EXISTING ITEMS AT THE START OF AN ITERATION
APPROX. TIME PERIOD FOR MAXIMUM SIMILARITY
TARGET ITEM
ITEMS ADDED IN CURRENT ITERATION
5 BATCHES OF ATTACKER PROFILES BEFORE ATTACK
5 BATCHES OF ATTACKER PROFILES AFTER ATTACK

Fig. 4: Attacker profiles at the end of the attack.

understanding of how the recommender system treats the different types of users. The system treats the user who has only purchased books different from the user who has consumed an array of items. So, the recommendations made to these users differ drastically. Having different observer combos helps in mimicking various types of authentic users.

B. Iterative Phase

In the iterative phase, the recommendations made to the observer profiles are added to the attacker profile. In this phase, the multi-armed bandit algorithm is used to select the observer combo. Each observer combo has a specific number of profiles arranged in a queue; the recommendation is chosen from the first profile in the line. The observer profiles are presented as given in Fig. 3. If the recommended item is related to the target item, then it is added to the attacker profile as well as the observer profile. If not, then it is only added to the observer profile. Adding the recommended item to the observer profile ensures that the item is not recommended again. After each iteration, the profile which made the recommendation is moved to the back of the queue.

Alg.1 explains the algorithm involved in the iterative and termination phases. Initially, all the observer combos are likely to be selected equally. But, as the number of iterations increases, the observer combos, which have a higher chance of giving a reward, are selected with a higher probability by the MAB.

C. Termination Phase

As the name suggests, the termination phase terminates the addition of new items to the attacker profile in batches and the termination of the attack process. Two main activities take place in the termination phase. These two actions help in tackling some of the newer developments in the CF.

First, after a threshold number of new items are added to all the attacker profiles, one batch of the attacker profile is retired after adding the target item as the final item to the batch. The size of the batch depends on the number of iteration cycles we want to continue and the attack size. Such batched termination creates attack profiles of different lengths.

As mentioned earlier, some of the more recent CF models take into consideration the number of items rated by an account for similarity calculation. By creating attacker profiles of different lengths, we are ensuring that there is some attack profile batch to match each kind of authentic users: from people who have rated only a few items to people who have rated many items.

At the termination of each batch, adding the target item creates a higher similarity between the newly added selected items and the target item. If the period in which the items were rated is taken into consideration in the CF, then the target would still have a high similarity with all the selected items.

Once the final batch reaches the predetermined maximum number of items allowed per attack profile, the attack process gets terminated. This termination keeps the estimated cost of the attack in check. After the final batch termination, the target item is added to all the observer profiles.

The attacker profiles at the end of the attack are illustrated in fig. 4. Initially, all the attacker profiles are of the same length. After the attack, the orange portion shows the items that were part of the batch during the previous iterative cycle. The blue-colored portion shows the newly added items after the termination of the previous batch. The green-colored part is the target item, the last item added before batch termination. The dotted region is a representation of the items which will have high similarity with the target if the time-period of rating is taken into consideration by the CF algorithm. By adding the target item as the last item of a batch, we are ensuring that all the newly added items get closely related to the target. It is important to note here that the different profile length or adding the target as the last item does not affect the attack even if the system doesn't use any additional features.

D. Toy Example of MAB Shilling Attack

To better understand the attack scheme, let us consider the toy example in fig. 5. In the example, we are injecting 200

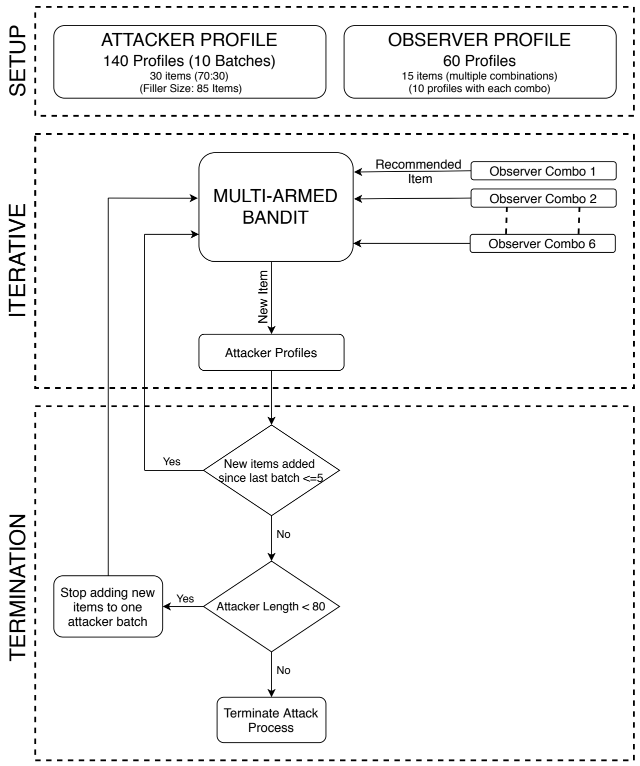

Fig. 5: Toy example of a shilling attack with 200 injected profiles.

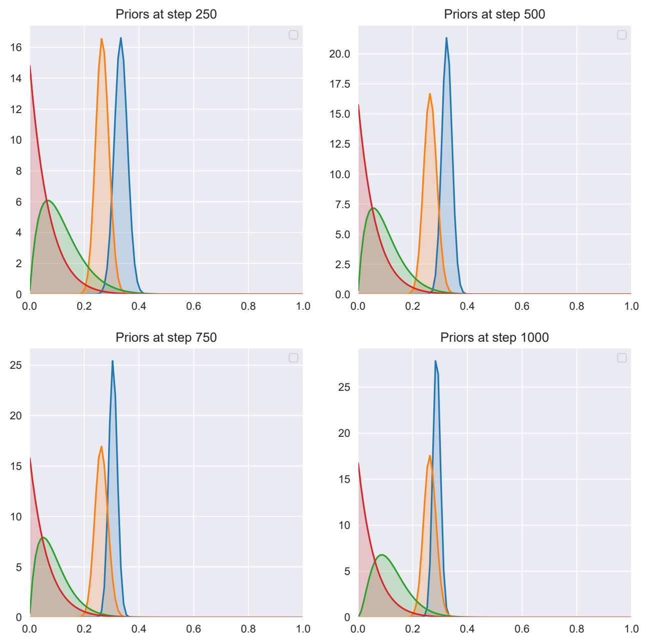

Fig. 6: Probability density of the beta distribution for four observer combos at different steps.

profiles into the system, out of which 140 are attacker profiles, and 60 are observer profiles. The attacker profiles are divided into ten batches, and the observer profiles into six observer combos, with ten profiles each. The initial number of rated items in the attacker profile is 30, and that of the observer profile is 15. For this example, we are going to consider a filler size of 80 items. This size implies that no injected profile should have more than 80 rated items. Given that the filler size, initial rated items, and the number of batches in the attacker profile, we can estimate that one batch should be terminated after every 5 new items added. This way, the first terminated batch will have 35 rated items, and the last terminated batch will have 80 rated items. The batch threshold value for this attack is 5.

In the iterative phase, the MAB algorithm is used to select one of the observer combos. The recommendation made to the first profile of the chosen observer combo is examined. If the recommended item is related to the target item, then it is a win for the MAB, and the item is added to both the chosen observer and all the attacker profiles. If the recommended item is not related, then it is a loss, and the item is only added to the chosen observer. After each new item added, the control is passed to the termination phase. This process is repeated until the attacker profile length is the same as the filler size.

The first step in the termination phase is to check if the new items added after the previous batch termination are less than the batch threshold, which is 5 in the example. If so, then the control is passed back to the iterative phase. If not, the length of the current attacker profiles is inspected. If the profile length is the same or greater than filler size, then the attack is terminated. If the profile length is less than filler size, then one batch of attacker profile is terminated after adding the target item as the last new item to the batch. The control is then passed back to the iterative phase, and the process continues until the last batch has a length of 80.

V. Experiments

In this section, we discuss the experimental evaluation of our method.

A. Dataset

We used the MovieLens 1M dataset for the evaluation of our method. This dataset was collected as part of the GroupLens Research Project for their work in [14]. It consists of 1,000,209 ratings from 6040 users on 3,900 items (including movies, series, and documentaries), with the genre included. Each user has rated a minimum of 20 items. The users rate the items on a scale of 1 to 5, with 1 being the least and 5 being the maximum possible rating.

B. Our Approach

We incorporate our MAB-based attack scheme by utilizing the genre of the items. 70% of the injected profiles are attacker profiles, and the remaining 30% are observer profiles. We choose the selected items related to the target by comparing their genre. We use four different observer combos for this attack, with 0%, 33%, 66%, and 100% selected items each, and the rest are filler items. We do not use recommended items in the observer combo formations, owing to the small number of total items. While using the MAB for item selection, if the

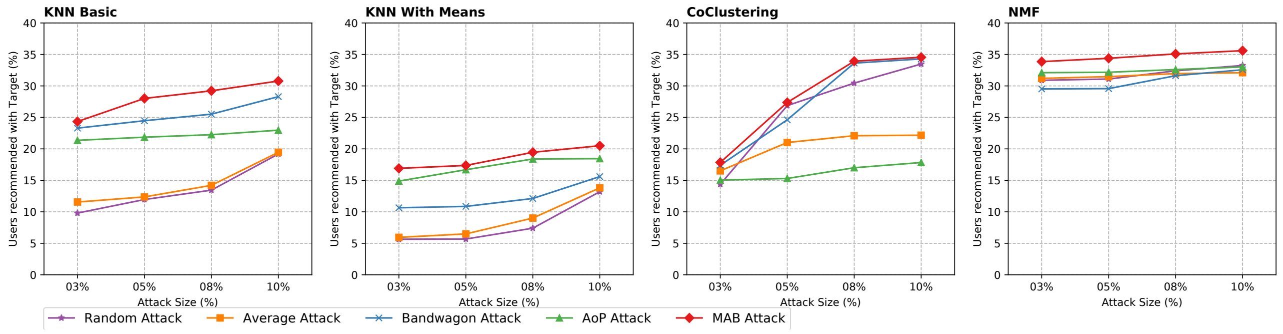

Fig. 7: Percentage of users with target item in their top-20 recommendation list when filler size is 3%.

genre of the item matches the target, then it is a win; otherwise, it is a loss.

Fig. 6 shows the probability density function of the beta distribution for the four observer combos at different steps while using kNN basic algorithm at 3% filler size, and 5% attack size. As the number of trials increases, the uncertainty associated with each combo decreases. Here, the color blue represents the 66% observer combo, which is the most explored arm and gives the most wins. One possible reason for this combo to be selected more often than the 100% combo could be that the 100% combo only recommends the items which are very similar to the target item. Most of the profiles in the 100% combo might have the same list of recommendations. Once an item is selected, the same item cannot be selected again, causing the trial to fail. On the other hand, the 66% combo would have a more diverse list of recommended items, leading to a higher win rate. It is important to note that the number of trials shown in the MAB, 1000, is depicted to show how the attack works. Generally, the attack terminates once the filler size is reached.

C. Baseline Attack Algorithms

We discuss the baseline algorithms against which we compare our method.

1) Random Attack [3]: It is the primary form of shilling attack, where the items rated by the attack profile are chosen at random. The system overall mean is used for rating each of these items, with a standard deviation of 1.1. The target item is given the maximum rating.

2) Average Attack [3]: This attack is a slightly more sophisticated form of the random attack. Here too, the rated items are chosen at random, but the rating is the average rating of the item. The target item is given the maximum rating.

3) Bandwagon Attack [4]: The Bandwagon attack operates by choosing all the rated items for the attack profiles from the most popular items in the dataset and give them high ratings. These items have a high rating from a large number of users.

4) Average Over Popular Attack [15]: The AoP attack is a variation of the average attack, used to obfuscate the attack signature. In AoP, the rating scheme is similar to the average attack, but the items are not chosen at random. The most popular items in the dataset are selected to be the filler items for this attack.

D. Baseline CF Algorithms

Different websites use different algorithms to fulfill their recommendation system needs. The attacker does not know of the algorithm used, making it necessary for the attack to be successful under different algorithms.

1) kNN Basic [16]: k-Nearest Neighbor algorithm works by finding the closest neighbors of a user and estimating the ratings for this user from the rating behavior of the neighboring users. The similarity measure between two users needs to be positive for them to be considered a neighbor. This algorithm is the vanilla implementation of the kNN approach.

2) kNN with Means [16]: This algorithm additionally takes into account the mean ratings of users in the similarity calculation process of the kNN algorithm. By including the mean rating values, the overly positive and overly negative users will not be part of an average user's neighborhood.

3) CoClustering Algorithm [17]: CoClustering algorithm simultaneously clusters users and items. This algorithm is designed to be a practical real-time CF approach owing to its low computational requirement. It works by generating user and item co-clusters and obtaining the average rating predictions based on these co-clusters.

4) Non-negative Matrix Factorization (NMF) [18]: NMF-based CF model is a single-element-based approach, such that none of the matrices involved in the factorization have negative elements. This algorithm works by examining the non-negative update process depending on each involved feature rather than on the entire feature matrices.

E. Evaluation Metrics

We conduct experiments to test the ability of the attack to push a target product. Initially, we select a target product that is not a part of the top-N recommended items of any of the 6,040 authentic users under various CF algorithms. Then we inject the same number of profiles into the system for each of the attacks. At the end of the attack, we evaluate the success of the attack by checking the percentage of authentic users with the target item as part of their top-N recommended items list. We repeat the evaluations by varying the attack size, filler size, and CF algorithms for all the attacks.

F. Result Analysis

We will discuss how the different attacks perform in the various algorithms and the impact of filler size. Fig. 7 and

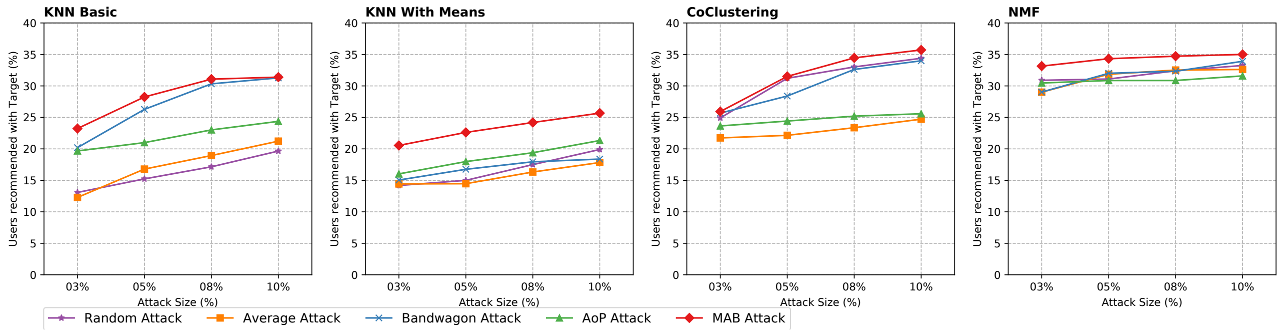

Fig. 8: Percentage of users with target item in their top-20 recommendation list when filler size is 5%.

fig. 8 show the extent of attack in the different algorithms. The x-axis shows the attack size, and the y-axis shows the percentage of authentic users who have the target item in their top-20 recommendations after the attack.

1) Changes with CF Algorithms: One of the key advantages of using our MAB-based approach is the ability to adapt to the CF system used. By using the recommendations made by the system, MAB manages to choose the optimal items to attack the system. The MAB performance with different CFs alters with how the CF handles the recommendation process and user information.

kNN Basic: In both the 3% and 5% filler sizes, the MAB based attack seems to be outperforming the other attacks. We can notice that the average and random attacks, being the simplest, seem to be performing significantly lower than others.

kNN with Means: By including the mean rating of a user, the CF alters the outcome of the attack. In the kNN basic method, the bandwagon attack performed better than the average attack. But, giving high ratings to most of its items reduces the similarity between bandwagon's attack profiles and the authentic users in kNN with means. This aspect affects its performance.

CoClustering: When it comes to the CoClusteing approach, both average and AoP attacks have low efficiency. This outcome could be because of the rating schemes used in these approaches. Using the system mean for rating the observer items and varying filler length of attack profiles, gives MAB method an edge over the other attacks.

NMF: By observing the attack reach in this method, we can notice that there is not a significant difference between the baseline attacks. The NMF algorithm does not seem to take the attacks' rating scheme into account during the recommendation process. The item selection method used by MAB appears to be giving it an edge over the other methods.

2) Changes with filler size: Fig. 7 shows the impact of the different attacks in different algorithms when the filler size is 3%. Similarly, fig. 8 shows filler size of 5%. Increasing the filler size does not hugely enhance the attack efficiency in either of the algorithms. In some instances, the similarity between attack and authentic profiles is lost because of the attack profile length. Moreover, profiles with too many rated items can easily be detected using simple detection techniques. By using attack profiles of different lengths, our approach has better reach and lesser detectible features.

VI. Related Work

A. Shilling Attacks

Shilling attack techniques came into existence from the early 2000s. The initial attack models focused more on disrupting the RS's performance rather than pushing or nuking a target product. In [3], Random attack and Average attack models were used to check the effectiveness of the attacks on User-User and Item-Item based CF. A more sophisticated model with better results in promoting the product was used in [4]. This attack, known as the bandwagon attack, chose the popular items to be the selected items. Reverse-bandwagon and love/hate attacks were two effective nuke attacks in [5]. Attacking only a segment of the RS instead of the entire items list was executed in [19]. Some recommenders gave possible ratings for its items, which was used in probe attack, [20]. As soon as shilling attacks were discovered to be possible, simultaneously, there were many research works to detect such attacks. A series of obfuscated attack strategies were created to avoid detection. In [21], user shifting and target shifting techniques were used to hide some of the attack profiles and target items. Bhaumik et al. in [22] used a combination of the random, average, bandwagon, and segmented attacks to avoid detection. Both [15] and [23] designed the attack profiles to be similar to the most popular users and most popular items, respectively. Compared to other methods, our method works online and considers other possible features present in the RS.

B. Multi-Armed Bandits

Multi-Armed bandits have recently been used in many applications. We discuss some of those applications here. In the healthcare segment, [24] employs an adaptive model and allocate more samples to provide better treatment options. In the finance segment, [25] use MAB for making online portfolio choices. In [26], the authors make an algorithm to choose between earning an immediate profit and learning for future profit when the demand information is incomplete. In RS, [27] apply MAB on large-scale RS even when no prior information about the user is available. In [28], the authors use MAB to maximize the influence of a product by selecting the optimal seed profiles for promotion. The authors of [29] utilize

MAB for selecting the proper response for dialogue in online learning systems. In [30], MAB is used for anomaly detection by interacting with human subjects to learn ground truth. In the telecommunication segment, [31] use MAB for the best wireless network selection by multiRadio Access Technology. In our work, we use MAB to reduce the uncertainty related to optimal item selection.

VII. Conclusion and Future Work

In this paper, we have explored the possibility of applying an online shilling attack, which utilizes the feedback from the recommendation system to increase the reach of the attack. The framework treats the Collaborative Filtering RS as a BlackBox and functions well with both user-user-based and item-item-based methods. We use a Multi-Armed Bandit based approach to reduce the uncertainty associated with the item selection process. Our results are encouraging and show that our online shilling attack approach has a better reach than the existing baseline methods. More research on shilling attacks is imperative as more and more businesses are using Collaborative Filtering systems. We are currently working on obfuscating our attack without affecting efficiency. In the future, we are planning to extend our approach to work on a Graph-based Recommender System as well.

Acknowledgment

This work was supported in part by the U.S. National Science Foundation grant CNS-1852105, CNS-1560020, and OAC-1839746.

References

[1] R. Van Meteren and M. Van Someren, "Using content-based filtering for recommendation," in *Proceedings of the Machine Learning in the New Information Age: MLnet/ECML2000 Workshop*, vol. 30, 2000, pp. 47–56.

[2] D. Goldberg, D. Nichols, B. M. Oki, and D. Terry, "Using collaborative filtering to weave an information tapestry," *Communications of the ACM*, vol. 35, no. 12, pp. 61–70, 1992.

[3] 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.

[4] M. P. O'Mahony, N. J. Hurley, and G. C. Silvestre, "Recommender systems: Attack types and strategies," in *AAAI*, 2005, pp. 334–339.

[5] B. Mobasher, R. Burke, R. Bhaumik, and J. J. Sandvig, "Attacks and remedies in collaborative recommendation," *IEEE Intelligent Systems*, vol. 22, no. 3, pp. 56–63, 2007.

[6] 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.

[7] 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.

[8] R. Burke, B. Mobasher, C. William, and R. Bhaumik, "Detecting profile injection attacks in collaborative recommender systems," in *The 8th IEEE International Conference on E-Commerce Technology and The 3rd IEEE International Conference on Enterprise Computing, E-Commerce, and E-Services (CEC/EEE'06)*. IEEE, 2006, pp. 23–23.

[9] M. Balabanović and Y. Shoham, "Fab: content-based, collaborative recommendation," *Communications of the ACM*, vol. 40, no. 3, pp. 66–72, 1997.

[10] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, "Item-based collaborative filtering recommendation algorithms," in *Proceedings of the 10th international conference on World Wide Web*, 2001, pp. 285–295.

[11] B. Smith and G. Linden, "Two decades of recommender systems at amazon. com," *Ieee internet computing*, vol. 21, no. 3, pp. 12–18, 2017.

[12] H. Robbins, "Some aspects of the sequential design of experiments," *Bulletin of the American Mathematical Society*, vol. 58, no. 5, pp. 527–535, 1952.

[13] W. R. Thompson, "On the likelihood that one unknown probability exceeds another in view of the evidence of two samples," *Biometrika*, vol. 25, no. 3/4, pp. 285–294, 1933.

[14] 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.

[15] N. Hurley, Z. Cheng, and M. Zhang, "Statistical attack detection," in *Proceedings of the third ACM conference on Recommender systems*, 2009, pp. 149–156.

[16] N. S. Altman, "An introduction to kernel and nearest-neighbor nonparametric regression," *The American Statistician*, vol. 46, no. 3, pp. 175–185, 1992.

[17] T. George and S. Merugu, "A scalable collaborative filtering framework based on co-clustering," in *Fifth IEEE International Conference on Data Mining (ICDM'05)*. IEEE, 2005, pp. 4–pp.

[18] X. Luo, M. Zhou, Y. Xia, and Q. Zhu, "An efficient non-negative matrix-factorization-based approach to collaborative filtering for recommender systems," *IEEE Transactions on Industrial Informatics*, vol. 10, no. 2, pp. 1273–1284, 2014.

[19] R. Burke, B. Mobasher, and R. Bhaumik, "Limited knowledge shilling attacks in collaborative filtering systems," in *Proceedings of 3rd International Workshop on Intelligent Techniques for Web Personalization (ITWP 2005), 19th International Joint Conference on Artificial Intelligence (IJCAI 2005)*, 2005, pp. 17–24.

[20] R. Burke, B. Mobasher, R. Zabicki, and R. Bhaumik, "Identifying attack models for secure recommendation," *Beyond Personalization*, vol. 2005, 2005.

[21] C. Williams, B. Mobasher, R. Burke, J. Sandvig, and R. Bhaumik, "Detection of obfuscated attacks in collaborative recommender systems," in *Proceedings of the ECAI'06 Workshop on Recommender Systems*, vol. 94, 2006.

[22] 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 Mining (DMIN)*. Citeseer, 2011, p. 1.

[23] P. Adamopoulos and A. Tuzhilin, "On unexpectedness in recommender systems: Or how to better expect the unexpected," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 5, no. 4, pp. 1–32, 2014.

[24] A. Durand, C. Achilleos, D. Iacovides, K. Strati, G. D. Mitsis, and J. Pineau, "Contextual bandits for adapting treatment in a mouse model of de novo carcinogenesis," in *Machine Learning for Healthcare Conference*, 2018, pp. 67–82.

[25] W. Shen, J. Wang, Y.-G. Jiang, and H. Zha, "Portfolio choices with orthogonal bandit learning," in *Twenty-Fourth International Joint Conference on Artificial Intelligence*, 2015.

[26] K. Misra, E. M. Schwartz, and J. Abernethy, "Dynamic online pricing with incomplete information using multiarmed bandit experiments," *Marketing Science*, vol. 38, no. 2, pp. 226–252, 2019.

[27] Q. Zhou, X. Zhang, J. Xu, and B. Liang, "Large-scale bandit approaches for recommender systems," in *International Conference on Neural Information Processing*. Springer, 2017, pp. 811–821.

[28] S. Vaswani, B. Kveton, Z. Wen, M. Ghavamzadeh, L. V. Lakshmanan, and M. Schmidt, "Model-independent online learning for influence maximization," in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, 2017, pp. 3530–3539.

[29] B. Liu, T. Yu, I. Lane, and O. J. Mengshoel, "Customized nonlinear bandits for online response selection in neural conversation models," in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[30] K. Ding, J. Li, and H. Liu, "Interactive anomaly detection on attributed networks," in *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, 2019, pp. 357–365.

[31] S. Boldrini, L. De Nardis, G. Caso, M. T. Le, J. Fiorina, and M.-G. Di Benedetto, "mumab: A multi-armed bandit model for wireless network selection," *Algorithms*, vol. 11, no. 2, p. 13, 2018.