An Empirical Analysis of Collaborative Recommender Systems Robustness to Shilling Attacks

Anu Shrestha, Francesca Spezzano and Maria Soledad Pera

Boise State University, Boise, ID, USA

Abstract

Recommender systems play an essential role in our digital society as they suggest products to purchase, restaurants to visit, and even resources to support education. Recommender systems based on collaborative filtering are the most popular among the ones used in e-commerce platforms to improve user experience. Given the collaborative environment, these recommenders are more vulnerable to shilling attacks, i.e., malicious users creating fake profiles to provide fraudulent reviews, which are deliberately written to sound authentic and aim to manipulate the recommender system to promote or demote target products or simply to sabotage the system. Therefore, understanding the effects of shilling attacks and the robustness of recommender systems have gained massive attention. However, empirical analysis thus far has assessed the robustness of recommender systems via simulated attacks, and there is a lack of evidence on what is the impact of fraudulent reviews in a real-world setting. In this paper, we present the results of an extensive analysis conducted on multiple real-world datasets from different domains to quantify the effect of shilling attacks on recommender systems. We focus on the performance of various well-known collaborative filtering-based algorithms and their robustness to different types of users. Trends emerging from our analysis unveil that, in the presence of spammers, recommender systems are not uniformly robust for all types of benign users.

Keywords

Recommender system, shilling attack, robustness, fraudulent reviews, Collaborative filtering

1. Introduction

The effect of shilling attacks on recommender systems, where malicious users create fake profiles so that they can then manipulate algorithms by providing fake reviews or ratings, have been long studied [1, 2, 3]. So far, recommender system researchers have: (1) Characterized and modeled recommender system shilling attacks (where malicious users insert fake profiles to manipulate recommendations), (2) Defined new metrics to quantify the impacts of these attacks on commonly used recommender systems, and (3) Applied a *detect + filtering* approach to mitigate the effects of spammers on recommendations. Nevertheless, we observe from the literature that the analysis thus far has focused on assessing the robustness of recommender systems via *simulated attacks* [4, 5]. Unfortunately, there is a lack of evidence on what is the

OHARS'21: Second Workshop on Online Misinformation- and Harm-Aware Recommender Systems, October 2, 2021, Amsterdam, Netherlands

anushrestha@u.boisestate.edu (A. Shrestha); francescaspezzano@boisestate.edu (F. Spezzano); solepera@boisestate.edu (M. S. Pera)

impact of fake reviews or fake ratings in a *real-world* setting.

In this paper, we present an analysis conducted to understand the influence of fraudulent reviews on the recommendation process in real-world scenarios. We do this through a study of known datasets with gold standards in different domains and several commonly-used recommendation algorithms. Specifically, we utilize data from two widely-used e-commerce platforms, Yelp! and Amazon. Among various recommendation algorithms, we consider collaborative filtering-based approaches as they are the most efficient and popular recommenders in such platforms. Thus, we focused our exploration on the robustness of these algorithms to shilling attacks.

The main contribution of this paper is two-fold. First, we analyze the performance of widely used five collaborative filtering-based algorithms in the presence of spammers and compared them when spammers are removed. By doing so, we seek to answer whether shilling attacks affect the robustness of the considered recommender algorithms. Second, we investigate if there is a specific user group (non-mainstream users) that are affected more than others (mainstream users) by spammers.

Our results are validated by an empirical evaluation using classical measures for evaluating predictive and top-N recommendation strategies. We show that RMSE scores decrease and NDCG@5 scores increase when removing spammers in the majority of the considered algorithms and datasets. This serves as an indication that the performances of considered collaborative-filtering-based recommender algorithms are indeed affected by spam ratings/reviews. Further, a deep investigation to quantify the effects of spammers on recommendations received by certain groups of users lead us to conclude that, for the Yelp! datasets removing spammers improves the predictive ability (RMSE) of all the considered recommender algorithms regardless of the type of users, i.e., mainstream or non-mainstream. However, in the case of Amazon datasets, we observed a trend where removing spammers lessen the predictive ability for mainstream users based on RMSE whereas improves for non-mainstream users according to both RMSE and NDCG@5. Therefore, non-mainstream users whose rating behavior does not align with the majority of the users are the most affected ones by spam ratings for Amazon datasets. Overall, we observed that 25%-29% of benign users in Amazon datasets are users who would not be equally satisfied by recommenders affected by shilling attacks. Thus, recommender algorithms are not uniformly robust for all types of benign users in the presence of spammers ratings/reviews.

The rest of this paper is organized as follows. In Section 2, we summarize related work; we then outline the dataset, algorithms, and evaluation strategies used in our empirical analysis in Section 3. In Section 4 we report on our results and, finally, conclusions are drawn in Section 5.

2. Related work

Collaborative filtering-based recommender systems are widely used to provide recommendations to users in opinion-based systems, yet they are vulnerable to shilling attacks [6, 2]. These attacks consist of fake user profiles injected into the system with the goal of providing spam ratings or reviews to promote or demote specific products. While some Shilling attacks promote the recommendations of certain attacked items (referred to as the push attack), others might demote

the predictions that are made for attacked items (referred to as the nuke attack) [4, 7]. Previous work has defined several attack strategies, including random, average, bandwagon, love/hate, segmented, and probe attacks. These strategies differ in the way fake profiles choose filler items, i.e., other rated items chosen beyond attacked items to camouflage the fraudulent behavior. More sophisticated attacks have been recently proposed [6], for instance, the one by Fang et al. [8] looks at how to choose filler items to recommend an attacker-chosen targeted item to as many users as possible. In the field of machine learning, many efforts have been devoted over the years to develop techniques for automatic detection of such fraudulent profiles; the techniques presented in [9] and [10] are among the most recent ones. In the field of recommender systems, researchers have focused on studying the effects of such shilling attacks mainly on collaborative filtering-based recommenders since the early 2000s [11, 4] and developed strategies to make such algorithms more robust to shilling attacks [2]. We highlight, for example, outcomes of the research conducted by Seminario and Wilson [12, 13] who explicitly look at power user and power items attacks, i.e., attacks targeting influential users and items, respectively, within collaborative filtering-based recommender systems.

Most recently, the concept of differential privacy has been explored to make matrix factorization-based collaborative filtering recommender algorithms more robust [14]. The vulnerability of deep-learning-based recommender systems to shilling attacks has been studied in [15]. In particular, Lin et al. introduce a framework that considers complex attacks aimed towards specific user groups. On a different perspective, Deldjoo et al. [16] explore dataset characteristics to explain an observed change in the performance of recommendation under shilling attacks.

Our work add to this body of knowledge by exploring the robustness of collaborative recommender systems to shilling attacks by using real-world data with spam reviews ground truth, as opposed to attack simulation and investigating if some users are more vulnerable than others.

3. Experimental Settings

As previously stated, our goal is to analyze commonly-used memory-based and model-based collaborative filtering-based recommendation algorithms' robustness to shilling attacks using a number of datasets with ground truth on spam reviews. In the rest of this section, we describe the experimental protocol for our analysis.

3.1. Datasets

For analysis purposes, we rely on four datasets (described below) produced based upon data from two well-known e-commerce platforms: Yelp! and Amazon.

Yelp! We consider Yelp! reviews from two domains: hotels (**YH**) and restaurants (**YR**) [17]. Yelp filters fake/suspicious reviews and puts them on a spam list. A study found the Yelp filter to be highly accurate [18], and many researchers have used filtered spam reviews as ground truth for spammer detection [19, 20]. Spammers, in our case, are users who wrote at least one filtered review. We removed users who rated the same products multiple times and reviews with a rating of zero.

Dataset	Users	Items	Ratings	Spammers
YH	5,027	72	5,857	14.92%
YR	34,523	129	66,060	20.25%
AB	167,725	29,004	252,056	3.57%
AH	311,636	39,539	428,781	4.12%

Table 1
Details on the datasets considered for our analysis.

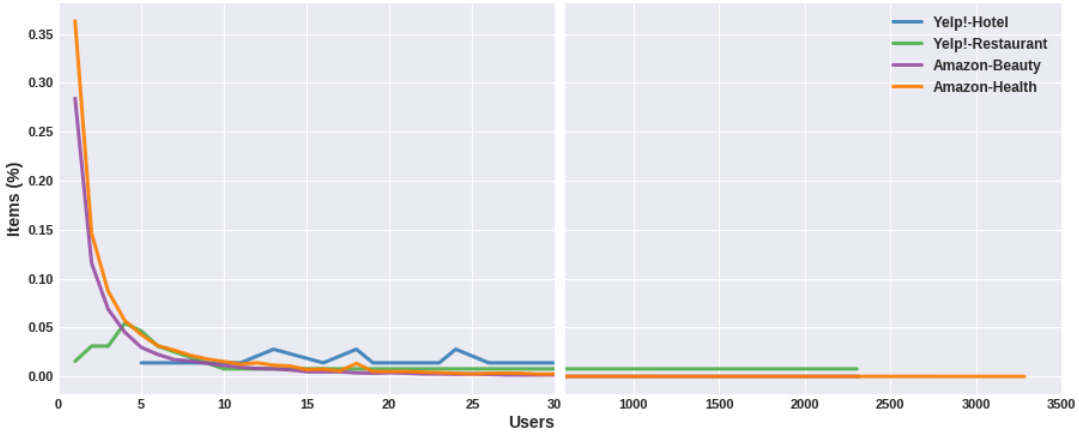


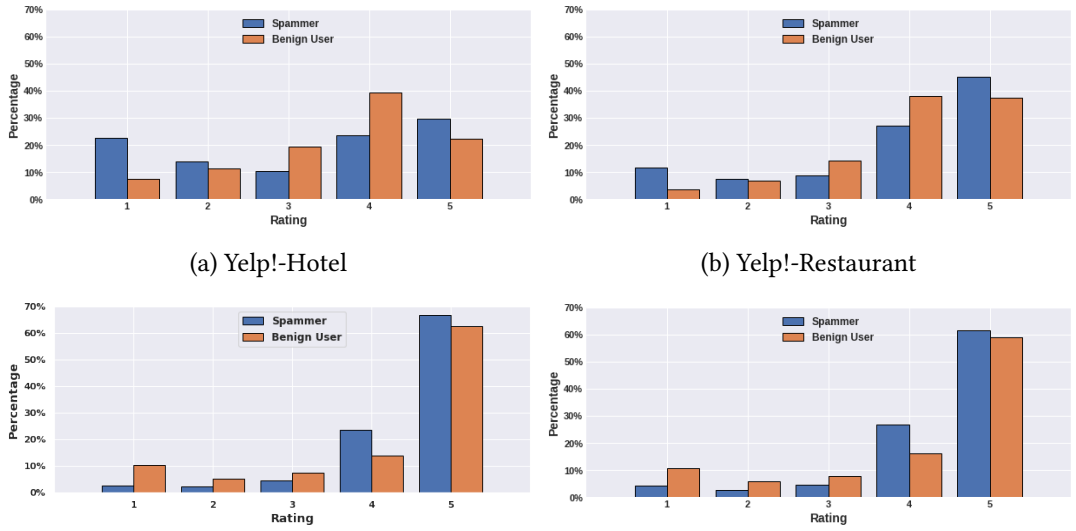
Figure 1: Rating distribution across the four datasets considered in our analysis.

Amazon We also consider Amazon reviews from two domains: beauty (**AB**) and health (**AH**) [21]. In this case, we define ground truth based on helpfulness votes following the approach suggested by [9] and based on the findings provided by Fayazi et al. [22]. Thus, we treat as a spammer every user who wrote at least one spam review. We define a review as spam if the rating is 4 or 5 and the helpfulness ratio is ≤ 0.4.

We provide descriptive statistics for the four datasets in Table 1. It is important to note that rating distribution is not similar across the datasets. As illustrated in Figure 1, rating trends from AH are dissimilar to the other counterparts, with a vast number of users rating only 1 item. Moreover, the rating distribution of benign users vs. spammers on attacked products (i.e., products receiving at least one spam review) is captured in Figure 2. From this figure, it emerges that benign users and spammer counterparts exhibit similar rating behavior in YR, AB, and AH, whereas in the case of YH, spammers noticeably assign ratings of '1' more often than benign counterparts; the opposite is true for ratings of '4'.

3.2. Algorithms

We focus our analysis on well-known and widely-used collaborative filtering-based recommendation algorithms implemented using Lenskit for python [23], except for Probabilistic Matrix Factorization, for which we relied on the implementation provided by Mnih et al. [24].



(a) Yelp!-Hotel

(b) Yelp!-Restaurant

(c) Amazon-Beauty

(d) Amazon-Health

Figure 2: Rating distribution for attacked products by spammers and benign users.

Item-item [25] is the popular item-based collaborative filter algorithm. It utilizes an item-item matrix to determine the similarity between the target item and other items (neighbors). We used this algorithm with 20 neighbors and cosine similarity as similarity measure.

Probabilistic Matrix Factorization (PMF) [24] is a commonly-used latent factor-based recommendation algorithm. Specifically, probabilistic matrix factorization decomposes the sparse user-item matrix into low-dimensional matrices with latent factors to generate recommendations. We used this algorithm with 40 latent factors and 150 iterations. This algorithm is known for its accuracy, scalability, and dealing with sparsity.

Alternating Least Squares (ALS) [26] is a matrix factorization-based algorithm designed to improve recommendation algorithms performance in large-scale collaborative filtering problems. This algorithm gain recognition following its success on the Netflix Challenge [27, 28]. In our case, we consider 40 latent factors, 5 damping factors, and 150 iterations for our experiment.

Bayesian Personalized Ranking (BPR) [29] is a rank-based matrix factorization algorithm, with 40 latent factors, 5 damping factors, and 150 iterations for our experiment. Note that as top-N recommendation algorithm, i.e., based on rating information is in the form of implicit feedback [29, 30], BPR scores items, but does not produce rating predictions. Thus, we are forced to exclude BPR from the RMSE-based analysis discussed in Section 4.

FunkSVD [31] is the well-known gradient descent matrix factorization technique with 40 latent features and 150 training iterations per feature.

3.3. Evaluation Framework

By following the classical evaluation framework for shilling attacks on recommender systems [4], we measured the performances on the original dataset (with spammers) and when we remove all the spammers (shilling attack), using well-known performance metrics. In all cases, we performed 5-fold cross-validation. We tested whether differences in the metric values with and without spammers were statistically significant using a paired t-test.

Metrics. For assessment, we turn to *Root Mean Square Error* (RMSE) and *Normalized Discounted Cumulative Gain* (NDCG), which are classical measures for evaluating predictive and top-N recommendations. We also consider measures explicitly defined to quantify the impact of spammer attacks on recommenders: *Prediction Shift* (PS), which captures the average absolute changes in predicted ratings for attacked items and *Hit Ratio* (HR), which considers if attacked items are promoted to user top-n recommendations (cf. Burke et al. [4] for formal definitions of these metrics).

We first examined recommender performance by considering all users in the respective datasets. We then segmented users into *fairness* categories as computed by the Fairness and Goodness algorithm described in the next paragraph in order to allow for more in-depth explorations. By segmenting users based on fairness scores, it is possible to identify mainstream and non-mainstream users. The latter are those whose rating patterns do not align with the majority, i.e., liking what most people dislike and vice versa [32].

Fairness and Goodness The Fairness and Goodness algorithm (F&G) [33] provides a measure for capturing user rating behavior. While many measures exist for this task [9], we chose to use F&G as Serra et al. [10] recently show it to be the best measure to identify trustworthy users in opinion-based systems. F&G computes a fairness score for each user and a goodness score for each item. Specifically, the *fairness* $f(u)$ of a user u is a measure of how fair or trustworthy the user is in rating items. Intuitively, a 'fair' or 'trustworthy' rater should give an item the rating that it deserves, while an 'unfair' one would deviate from that value. In the case of benign users, the latter could be the case of an uninformative or non-mainstream user. The *goodness* $g(i)$ of an item i specifies how much users in the system like the item and what its true quality is. Fairness and goodness are mutually recursively computed as:

$$f(u) = 1 - \frac{1}{|out(u)|} \sum_{i \epsilon out(u)} \frac{|W(u,i) - g(i)|}{R} \tag{1}$$

$$g(i) = \frac{1}{|in(i)|} \sum_{u \epsilon in(i)} f(u) \times W(u,i) \tag{2}$$

where $W(u,i)$ is the rating given by the user u to the item i, $out(u)$ is the set of ratings given by user u, $in(i)$ is the set of ratings received by item i, and $R = 4$ in this case which corresponds to the maximum error in a five-star rating system. Thus, the goodness of an item is given by the average of its rating, where each rating is weighted by the fairness of the rater, while the fairness of a user considers how much the ratings a user gives are far from the goodness of the

items. The higher the fairness, the more trustworthy the user is. Fairness scores of the user lie in the [0, 1] interval, and goodness scores lie in the [1, 5] interval.

Dataset	Algorithm	RMSE		NDCG@5		HR@5		PS
		W Spammers	W/o Spammers	W Spammers	W/o Spammers	W Spammer	W/o Spammer	Attacked Items
YH	Item-Item	1.33	1.32	0.104	0.105	0.031	0.032	0.087
	PMF	1.125	1.124	0.57	0.57	0.0217	0.0216	0.119
	BPR			0.023	0.030	0.022	0.022	0.159
	ALS	1.028	1.020	0.041	0.043	0.0218	0.0217	0.143
	FunkSVD	1.023	1.019	0.034	0.032	0.022	0.022	0.129
YR	Item-Item	1.181	1.179	0.0073	0.0075	0.013	0.013	0.119
	PMF	1.040	1.037	0.56	0.56	0.014	0.014	0.122
	BPR			0.088	0.087	0.013	0.013	0.160
	ALS	0.971	0.970	0.049	0.051	0.013	0.013	0.148
	FunkSVD	0.993	0.981	0.012	0.013	0.0136	0.0137	0.138
AB	Item-Item	0.95	0.95	0.295	0.299	0.000140	0.000149	0.130
	PMF	0.912	0.905	0.552	0.553	0.000051	0.000051	0.121
	BPR			0.801	0.828	0.00023	0.00024	0.124
	ALS	0.802	0.802	0.265	0.264	0.0003	0.0003	0.116
	FunkSVD	0.637	0.644	0.028	0.032	0.0064	0.0061	0.11
AH	Item-Item	1.151	1.154	0.290	0.294	0.00013	0.00012	0.105
	PMF	1.053	1.051	0.518	0.519	0.000036	0.000036	0.101
	BPR			0.794	0.827	0.00023	0.00024	0.104
	ALS	0.952	0.952	0.198	0.204	0.00033	0.00032	0.10
	FunkSVD	0.994	0.933	0.070	0.067	0.00298	0.00296	0.10

Table 2
Performance analysis using different metrics on datasets with and without spam. Statistically significant differences are shaded in gray, $pvalue \leq 0.001$.

4. Results and Discussion

In this section, we present our experimental evaluation of five recommendation algorithms on four datasets of different domains. We discuss the effect of shilling attacks on recommendations offered to users in *real-world* scenarios, as opposed to *simulated* attacks. Specifically, we aim to answer the following research questions:

RQ1 Do spammer's ratings impact recommendations?
RQ2 Who is really affected by spammers?

By investigating recommender algorithm performance in the presence of spammers as well as when spammers are removed, the first question enables us to gauge the shilling attacks' effect on the robustness of the considered recommendation algorithms. For the second question, we used the fairness metric to determine mainstream and non-mainstream users and quantify the effect of spammers on recommendations received by non-mainstream users.

4.1. Do spammers ratings impact recommendations?

To answer RQ1, we consider the performance of the recommender algorithms yielded on four different datasets, as reported in Table 2. It comes across from the reported scores that removing spammers indeed leads to lower RMSE scores, i.e., better predictions. Previous works have shown PS values ranging from 0.5 to 1.5 when shilling attacks are simulated [34]. However,

we observe very low values in real-world scenarios: in our case, considered, PS ranges from 0.087 to 0.160, which we argue might not be enough to promote or demote products attacked by the spammers. When looking at algorithm performance from a top-N recommendation standpoint, from reported NDCG@5 we see that, often, NDCG@5 scores tend to increase when removing spammers. This means that users' preferred items are more likely to appear within the top-5 recommendations when spammers are excluded. Unfortunately, improvement is not always meaningful, i.e., from Table 2 we see that improvements are not always significant, especially on YH. We anticipated lower HR@5 scores when excluding spammers–we assumed fewer attacked items would be promoted among the top-5 recommendations. Instead, we see similar trends among HR@5 results as those observed for NDCG@5. In other words, for YH and YR, performance is comparable regardless of the presence of spammers (i.e., differences in performance are not significant); for AB and AH we see significant differences in performance.

Overall, we can say that, in theory, the performance of collaborative filtering-based recommender algorithms is affected by spammers' ratings/reviews. This is particularly noticeable for predictive recommenders (i.e., all algorithms yielded significant differences across the datasets). In practice, however, performance improvements are in their majority barely perceptible. This leads us to question whether algorithm robustness is reflected by average metrics like RMSE or NDCG. In the end, looking at recommender performance as a whole may not clearly quantify how much spammers are able to deceit recommenders and, more importantly, if there are specific user groups that are affected more than others. With this in mind, we conduct a more thorough analysis with the aim of understanding if the aforementioned differences in performance are more pronounced among certain types of users (i.e., non-mainstream ones).

4.2. Who is really affected by spammers?

To better understand which users are really affected by spammers, we analyzed users based on their fairness: the ability of a user to rate a product according to what it deserves. It is worth noting, however, that the rating a product deserves often aligns with what the majority of benign users (mainstream users) think about that product, as mainstream users often outnumber non-mainstream and spam users. We investigate trends according to RMSE, NDCG@5, and hit ratio. As noted in the prior subsection, prediction shift values were small, so we excluded this metric from our analysis).

Figure 3 illustrates how the RMSE varies according to the fairness of benign users; for ease of readability, we highlight statistically significant differences in performance when spammers are excluded in Table 3. We start by observing that, regardless of the algorithm for both Yelp! datasets and with just one exception (YR, ALS, and FunkSVD, $(0.4 - 0.5]$), removing spammers reduces the RMSE for all users. For the Amazon datasets, instead, when the user fairness is greater than 0.4, removing spammers increases the RMSE for all users for each algorithm. We posit these results could be due to the rating distributions of spammers vs. benign users across these two platforms. As previously shown in Figure 2, spammer and benign users are more similar in Amazon than Yelp!, with the majority of ratings being 4 and 5. Therefore, removing spam could cause the recommender to lose information from mainstream users. On the other end, when fairness is less than or equal to 0.4 among Amazon users, in most cases where the difference is statistically significant, i.e., 14 out of 19 cases, removing spammers

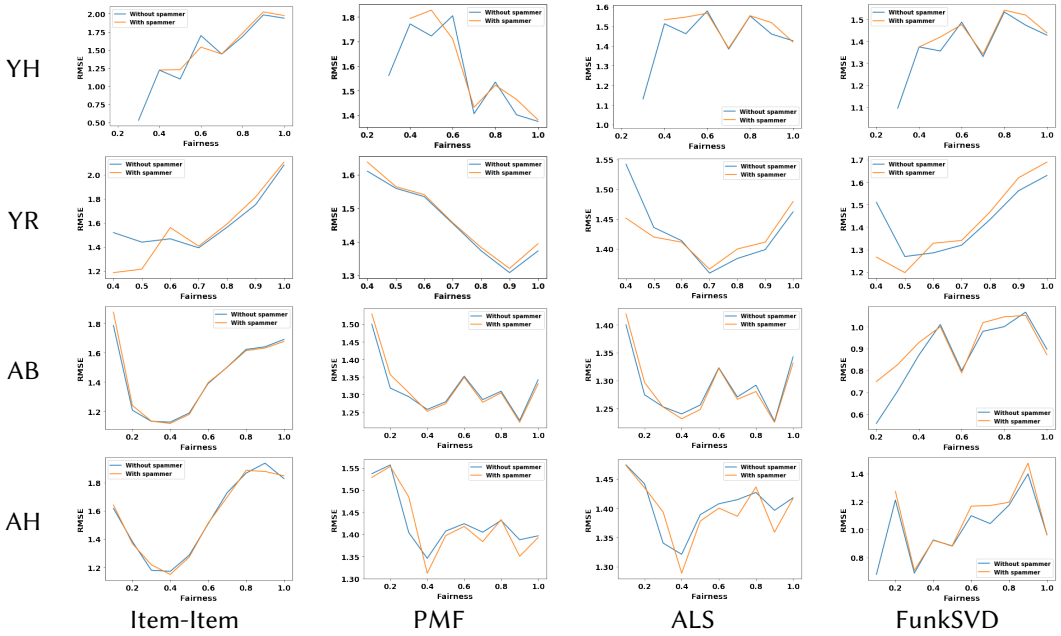


Figure 3: RMSE differences across fairness range.

Dataset	Algorithm	[0-0.1]	(0.1-0.2]	(0.2-0.3]	(0.3-0.4]	(0.4-0.5]	(0.5-0.6]	(0.6-0.7]	(0.7-0.8]	(0.8-0.9]	(0.9-1]
YH	#Benign Users	1	6	10	347	443	368	1281	550	882	389
	Item-Item									$W > W/o**$	
	PMF							$W > W/o**$		$W > W/o**$	
	ALS					$W > W/o**$		$W > W/o**$		$W > W/o**$	$W > W/o**$
	FunkSVD						$W > W/o*$		$W > W/o**$	$W > W/o**$	$W > W/o**$
YR	#Benign Users	0	0	2	269	6502	4898	4580	7450	1889	2071
	Item-Item									$W > W/o**$	
	PMF					$W > W/o**$	$W > W/o**$	$W > W/o**$	$W > W/o**$	$W > W/o**$	$W > W/o**$
	ALS					$W/o > W**$	$W > W/o**$	$W > W/o**$	$W > W/o**$	$W > W/o**$	$W > W/o**$
	FunkSVD					$W/o > W**$	$W > W/o**$	$W > W/o**$	$W > W/o**$	$W > W/o**$	$W > W/o**$
AB	#Benign Users	1716	5352	13914	27947	25755	18632	16354	15353	11860	24861
	Item-Item		$W > W/o**$	$W > W/o**$		$W/o > W**$			$W/o > W**$		
	PMF		$W > W/o*$		$W > W/o*$					$W/o > W**$	$W/o > W**$
	ALS	$W > W/o*$	$W > W/o**$	$W > W/o**$	$W/o > W**$	$W/o > W**$			$W/o > W**$	$W/o > W**$	$W/o > W**$
	FunkSVD									$W > W/o**$	
AH	#Benign Users	2574	6718	23273	46289	56630	41021	32512	30435	22363	36978
	Item-Item	$W > W/o**$	$W/o > W**$	$W > W/o**$	$W > W/o**$	$W/o > W**$	$W/o > W**$				
	PMF		$W/o > W**$	$W > W/o**$	$W > W/o**$	$W/o > W**$	$W/o > W**$	$W/o > W**$	$W/o > W**$	$W/o > W**$	$W/o > W**$
	ALS	$W/o > W*$	$W/o > W*$	$W > W/o**$	$W > W/o**$	$W/o > W**$	$W/o > W**$	$W/o > W**$	$W/o > W**$	$W/o > W**$	
	FunkSVD						$W > W/o*$				

Table 3
Statistically significant RMSE differences between recommendations without spammers (W/o) and with spammers (W) across different user fairness ranges (* means $pvalue < 0.03$ and ** means $pvalue \leq 0.01$). Cases where removing spammers reduces the RMSE are shaded.

enables algorithms to avoid noise signals and thus perform better for these users (lower RMSE). Note that there are more cases in AH than AB (4 out of 11 vs. 1 out of 8) where removing the spammers is not beneficial for non-mainstream users. This could be due to the fact that, as shown in Figure 1, AH data is more sparse than other datasets, making the process of generating recommendations more difficult for most of the users in such a setting, independently of the presence of spam.

When we look at trends for NDCG@5, Figure 4 and Table 4 show that the quality of the generated recommendations seldom improves on the Yelp! datasets for users having fairness

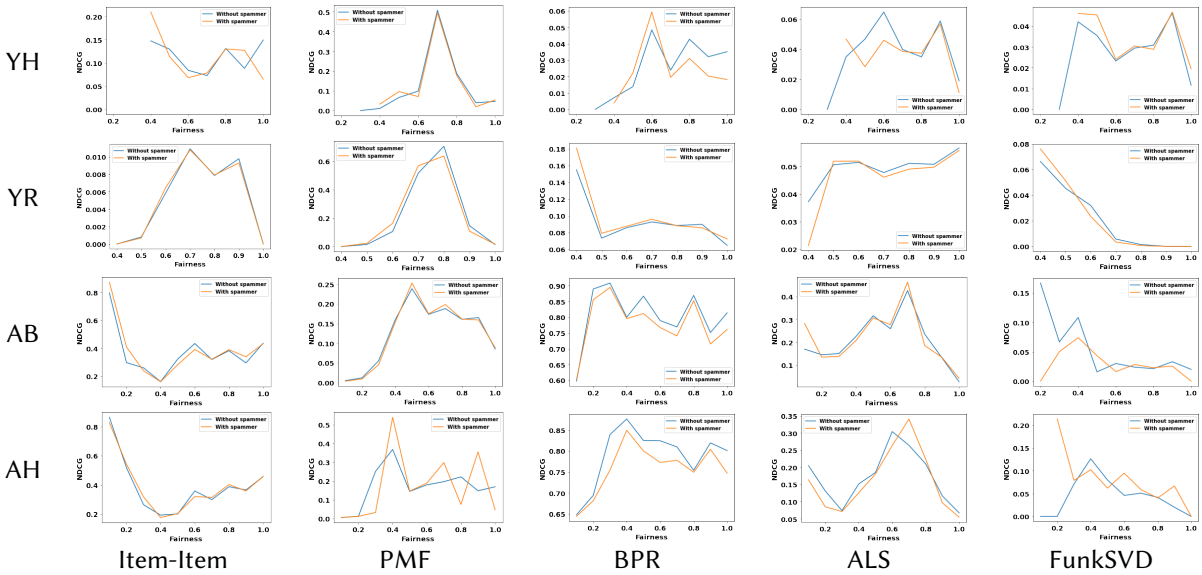


Figure 4: NDCG differences across fairness range.

Dataset	Algorithm	[0-0.1]	(0.1-0.2]	(0.2-0.3]	(0.3-0.4]	(0.4-0.5]	(0.5-0.6]	(0.6-0.7]	(0.7-0.8]	(0.8-0.9]	(0.9-1]
YH	Item-Item										
	PMF									$W/o > W**$	
	BPR					$W > W/o**$					
	ALS										
	FunkSVD					$W > W/o**$					
YR	Item-Item										
	PMF					$W > W/o**$	$W > W/o**$	$W > W/o**$	$W/o > W**$	$W/o > W**$	
	BPR							$W > W/o**$			
	ALS						$W > W/o**$				
	FunkSVD						$W/o > W**$	$W/o > W**$			
AB	Item-Item		$W > W/o**$	$W > W/o**$	$W/o > W**$		$W/o > W**$	$W/o > W**$	$W > W/o**$		$W > W/o**$
	PMF		$W/o > W**$	$W/o > W**$		$W > W/o**$		$W > W/o**$		$W/o > W**$	
	BPR				$W/o > W**$	$W/o > W**$	$W/o > W**$	$W/o > W**$		$W/o > W**$	
	ALS					$W/o > W**$	$W > W/o**$	$W > W/o**$			
	FunkSVD										
AH	Item-Item			$W > W/o**$	$W > W/o**$	$W/o > W**$	$W/o > W**$	$W/o > W**$	$W > W/o**$	$W > W/o**$	$W > W/o**$
	PMF			$W/o > W**$	$W > W/o**$	$W > W/o**$	$W > W/o**$		$W/o > W**$	$W > W/o**$	$W > W/o**$
	BPR		$W/o > W**$	$W/o > W**$	$W/o > W**$	$W/o > W**$	$W/o > W**$	$W/o > W**$	$W/o > W**$	$W/o > W**$	$W/o > W**$
	ALS		$W/o > W**$		$W/o > W**$			$W > W/o**$			
	FunkSVD										

Table 4
Statistically significant NDCG@5 differences between recommendations without spammers (W/o) and with spammers (W) across different user fairness ranges (* means $pvalue < 0.03$ and ** means $pvalue \leq 0.01$). Cases where removing spammers increases the NDCG@5 are shaded.

greater than 0.5, whereas for the Amazon datasets, the value of NDCG@5 is higher when spammers are removed in the majority of the cases (28 out of 47) and independently of the user type.

Overall, our analysis reveals that removing spammers helps in reducing the number of attacked items that hit the top-5 recommendations for all the users in all the datasets (see HR@5 analysis in Table 5).[1] Moreover, removing spammers in Yelp! is beneficial for all the users when considering the predictive performance of algorithms (based on RMSE); for Amazon, top-N algorithms are better (according to NDCG@5) among mainstream users, who see more tailored

[1]For brevity, we exclude a figure akin to those complementing Tables 3 and 4.

Dataset	Algorithm	[0-0.1]	(0.1-0.2]	(0.2-0.3]	(0.3-0.4]	(0.4-0.5]	(0.5-0.6]	(0.6-0.7]	(0.7-0.8]	(0.8-0.9]	(0.9-1]
YH	Item-Item										
	PMF					$W/o > W**$				$W > W/o**$	
	BPR					$W/o > W**$				$W > W/o**$	
	ALS					$W/o > W**$				$W > W/o**$	
	FunkSVD					$W/o > W**$				$W > W/o**$	
YR	Item-Item						$W > W/o**$	$W > W/o*$			
	PMF						$W > W/o**$	$W > W/o**$		$W > W/o**$	
	BPR						$W > W/o**$	$W > W/o*$			
	ALS						$W > W/o**$	$W > W/o**$		$W > W/o**$	
	FunkSVD						$W > W/o**$	$W > W/o*$			
AB	Item-Item	$W > W/o**$	$W > W/o**$	$W > W/o**$	$W > W/o**$	$W > W/o**$	$W > W/o**$	$W > W/o**$	$W > W/o**$	$W > W/o**$	$W > W/o**$
	PMF	$W > W/o**$		$W > W/o**$		$W/o > W**$	$W/o > W**$	$W/o > W**$			
	BPR	$W > W/o**$	$W > W/o**$	$W > W/o**$		$W > W/o**$	$W > W/o**$	$W > W/o**$			
	ALS		$W > W/o**$	$W > W/o**$	$W > W/o**$	$W > W/o**$	$W > W/o**$			$W > W/o**$	$W > W/o**$
	FunkSVD	$W > W/o**$	$W > W/o**$	$W > W/o**$	$W > W/o**$	$W > W/o**$	$W > W/o**$		$W > W/o**$	$W > W/o**$	$W > W/o**$
AH	Item-Item	$W > W/o**$	$W > W/o**$	$W > W/o**$	$W > W/o**$	$W > W/o**$	$W > W/o**$		$W > W/o**$	$W > W/o**$	$W > W/o**$
	PMF	$W > W/o**$	$W > W/o**$	$W > W/o**$	$W > W/o**$	$W > W/o**$	$W > W/o**$	$W > W/o**$			$W > W/o**$
	BPR		$W > W/o**$	$W > W/o**$		$W/o > W**$		$W > W/o**$			
	ALS		$W > W/o**$	$W > W/o**$	$W > W/o**$	$W > W/o**$	$W > W/o**$	$W/o > W**$		$W > W/o**$	$W > W/o**$
	FunkSVD	$W > W/o**$	$W > W/o**$		$W > W/o**$	$W > W/o**$	$W > W/o**$	$W > W/o**$			

Table 5
Statistically significant HR@5 differences between recommendations without spammers (W/o) and with spammers (W) across different user fairness ranges (* means $pvalue < 0.03$ and ** means $pvalue \leq 0.01$). Cases where removing spammers reduces the HR@5 are shaded.

recommended items in their top-5 item list when spammers are removed from the system. Also, we see performance improvement in terms of both RMSE and NDCG@5 scores for Amazon non-mainstream users, i.e., the ones with low fairness, hence the ones whose ratings are very far from the ones of the majority of the users. Non-mainstream users affected by spammers represent 29% (resp. 25%) of benign users in AB (resp. AH). In a real-world scenario, these percentages would translate into hundreds of thousands of users who would not be equally satisfied by recommenders that are not robust to shilling attacks.

5. Conclusions

In this paper, we have taken a deeper look into how shilling attacks affect recommender systems in a real-world scenario. For this, we conducted an in-depth exploration of the performance of five well-known collaborative filtering algorithms on four different datasets.

We saw similar trends among the performance of predictive and top-N recommenders: users are exposed to better recommendations when spammers are excluded (RQ1). This highlights the importance of further research on spammer detection and robust recommender systems. At the same time, we question if the small differences in performance (albeit statistically significant) would be evident to recommender systems' users and whether metrics considered for assessment which aggregate performance for all users could obfuscate users who are more deeply affected by spammers. This leads us to explore differences in performance between mainstream vs. non-mainstream users (RQ2). We saw that Amazon non-mainstream users are the ones most affected by spam ratings according to both RMSE and NDCG@5.

Based on the findings emerging from the analysis presented in this paper, it follows that future work will be devoted to looking at other types of recommender algorithms, beyond collaborative filtering-based, to see if the trends we have observed in our analysis remain. Moreover, we plan also to test the effectiveness of adversarial training for recommender systems [3] under the real-world attacks considered in this paper.

Acknowledgements

This work has been supported by the National Science Foundation under Awards no. 1943370 and 1820685.

References

[1] P.-A. Chirita, W. Nejdl, 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.

[2] M. Si, Q. Li, Shilling attacks against collaborative recommender systems: a review, Artificial Intelligence Review 53 (2020) 291–319.

[3] Y. Deldjoo, T. D. Noia, F. A. Merra, A survey on adversarial recommender systems: from attack/defense strategies to generative adversarial networks, ACM Computing Surveys (CSUR) 54 (2021) 1–38.

[4] R. Burke, M. O'Mahony, N. Hurley, Robust collaborative recommendation, in: Recommender systems handbook, Springer, 2015, pp. 961–995.

[5] C. E. Seminario, Accuracy and robustness impacts of power user attacks on collaborative recommender systems, in: RecSys, ACM, 2013, pp. 447–450.

[6] A. P. Sundar, F. Li, X. Zou, T. Gao, E. D. Russomanno, Understanding shilling attacks and their detection traits: a comprehensive survey, IEEE Access 8 (2020) 171703–171715.

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

[8] M. Fang, N. Z. Gong, J. Liu, Influence function based data poisoning attacks to top-n recommender systems, in: Proceedings of The Web Conference 2020, 2020, pp. 3019–3025.

[9] S. Kumar, B. Hooi, D. Makhija, M. Kumar, C. Faloutsos, V. S. Subrahmanian, REV2: fraudulent user prediction in rating platforms, in: WSDM, ACM, 2018, pp. 333–341.

[10] E. Serra, A. Shrestha, F. Spezzano, A. C. Squicciarini, Deeptrust: An automatic framework to detect trustworthy users in opinion-based systems, in: CODASPY, ACM, 2020, pp. 29–38.

[11] S. K. Lam, J. Riedl, Shilling recommender systems for fun and profit, in: WWW, ACM, 2004, pp. 393–402.

[12] C. E. Seminario, D. C. Wilson, Attacking item-based recommender systems with power items, in: Proceedings of the 8th ACM Conference on Recommender systems, 2014, pp. 57–64.

[13] C. E. Seminario, D. C. Wilson, Nuke'em till they go: Investigating power user attacks to disparage items in collaborative recommenders, in: Proceedings of the 9th ACM Conference on Recommender Systems, 2015, pp. 293–296.

[14] S. Wadhwa, S. Agrawal, H. Chaudhari, D. Sharma, K. Achan, Data poisoning attacks against differentially private recommender systems, in: Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, 2020, pp. 1617–1620.

[15] C. Lin, S. Chen, H. Li, Y. Xiao, L. Li, Q. Yang, Attacking recommender systems with

augmented user profiles, in: Proceedings of the 29th ACM International Conference on Information & Knowledge Management, 2020, pp. 855–864.
[16] Y. Deldjoo, T. Di Noia, E. Di Sciascio, F. A. Merra, How dataset characteristics affect the robustness of collaborative recommendation models, in: Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, 2020, pp. 951–960.
[17] A. Mukherjee, V. Venkataraman, B. Liu, N. S. Glance, What yelp fake review filter might be doing?, in: ICWSM, 2013, pp. 409–418.
[18] K. Weise, A lie detector test for online reviewers, in: Bloomberg BusinessWeek, 2011.
[19] S. Rayana, L. Akoglu, Collective opinion spam detection: Bridging review networks and metadata, in: SIGKDD, 2015, pp. 985–994.
[20] S. K. C., A. Mukherjee, On the temporal dynamics of opinion spamming: Case studies on yelp, in: WWW, 2016, pp. 369–379.
[21] J. McAuley, J. Leskovec, From amateurs to connoisseurs: modeling the evolution of user expertise through online reviews, in: WWW, 2013, pp. 897–908.
[22] A. Fayazi, K. Lee, J. Caverlee, A. Squicciarini, Uncovering crowdsourced manipulation of online reviews, in: SIGIR, 2015, pp. 233–242.
[23] M. D. Ekstrand, Lenskit for python: Next-generation software for recommender systems experiments, in: CIKM, 2020, pp. 2999–3006.
[24] A. Mnih, R. Salakhutdinov, Probabilistic matrix factorization, in: NIPS, 2008, pp. 1257–1264.
[25] B. Sarwar, G. Karypis, J. Konstan, J. Riedl, Item-based collaborative filtering recommendation algorithms, in: WWW, 2001, pp. 285–295.
[26] Y. Hu, Y. Koren, C. Volinsky, Collaborative filtering for implicit feedback datasets, in: IEEE ICDM, 2008, pp. 263–272.
[27] Y. Koren, R. Bell, C. Volinsky, Matrix factorization techniques for recommender systems, Computer 42 (2009) 30–37.
[28] Y. Zhou, D. Wilkinson, R. Schreiber, R. Pan, Large-scale parallel collaborative filtering for the netflix prize, in: AAIM, Springer, 2008, pp. 337–348.
[29] S. Rendle, C. Freudenthaler, Z. Gantner, L. Schmidt-Thieme, Bpr: Bayesian personalized ranking from implicit feedback, arXiv preprint arXiv:1205.2618 (2012).
[30] H. Wang, N. Wang, D.-Y. Yeung, Collaborative deep learning for recommender systems, in: Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining, 2015, pp. 1235–1244.
[31] A. Paterek, Improving regularized singular value decomposition for collaborative filtering, in: Proceedings of KDD cup and workshop, volume 2007, 2007, pp. 5–8.
[32] R. Z. Li, J. Urbano, A. Hanjalic, Leave no user behind: Towards improving the utility of recommender systems for non-mainstream users, in: WSDM, ACM, 2021, pp. 103–111.
[33] S. Kumar, F. Spezzano, V. S. Subrahmanian, C. Faloutsos, Edge weight prediction in weighted signed networks, in: IEEE ICDM, 2016, pp. 221–230.
[34] S. K. Lam, J. Riedl, Shilling recommender systems for fun and profit, in: WWW, 2004, pp. 393–402.