Zhang XL, Lee TMD, Pitsilis G. Securing recommender systems against shilling attacks using social-based clustering. JOURNAL OF COMPUTER SCIENCE AND TECHNOLOGY 28(4): 616–624 July 2013. DOI 10.1007/s11390-013-1362-0

Securing Recommender Systems Against Shilling Attacks Using Social-Based Clustering

Xiang-Liang Zhang[1] (张响亮), Tak Man Desmond Lee[1], and Georgios Pitsilis[2]

[1]*King Abdullah University of Science and Technology, Thuwal 23955-6900, Saudi Arabia*

[2]*Faculty of Science, Technology and Communication, University of Luxembourg, Luxembourg*

E-mail: xiangliang.zhang@kaust.edu.sa; desmondtm@gmail.com; georgios.pitsilis@googlemail.com

Received September 10, 2012; revised May 2, 2013.

Abstract Recommender systems (RS) have been found supportive and practical in e-commerce and been established as useful aiding services. Despite their great adoption in the user communities, RS are still vulnerable to unscrupulous producers who try to promote their products by shilling the systems. With the advent of social networks new sources of information have been made available which can potentially render RS more resistant to attacks. In this paper we explore the information provided in the form of social links with clustering for diminishing the impact of attacks. We propose two algorithms, CluTr and WCluTr, to combine clustering with "trust" among users. We demonstrate that CluTr and WCluTr enhance the robustness of RS by experimentally evaluating them on data from a public consumer recommender system *Epinions.com*.

Keywords shilling attack, recommender system, collaborative filtering, social trust, clustering

1 Introduction

Recommender systems (RS) have been proved effective tools of e-commerce for helping users to choose potentially appealing products. Collaborative filtering (CF), the approach that has been widely used in RS, makes recommendations to the users by aggregating the inputs from their most relevant users known as "neighbors".

"Neighbors" are useful only when they are reliable. Attackers could inject a large number of fake profiles, called *shilling profiles* or *bots*, in a way that they become similar and hence influential to victim users, with the purpose to promote (*push attack*) or to discredit (*nuke attack*) some products.

The free access provided by web-based RS can put them at risk and make them exploitable by attackers. The best known types of *shilling* attacks are *RandomBot* and *AverageBot*[1]. When injecting shilling profiles, *RandomBot* requires little information of each particular user as it creates fake profiles by randomly sampling from uniform distribution. The *AverageBot* is more powerful. It creates fake profiles sampled in a normal distribution with a mean averaged on all users' ratings. When promoting (or discrediting) a target product, attackers make all fake users give this product the highest rating (or the lowest rating). As the fake users are close neighbors to genuine users, their ratings on target products will affect the predicted ratings in RS for the genuine users. Defeating these attacks has been one of the challenges of RS research[2-5].

Since shilling profiles are generated from the same distribution and have the same ratings on target products, fake users are much more likely than genuine users to be highly similar to each other. Filtering out shilling users by clustering has been studied in [2, 4]. Clustering has also been employed in RS for improving prediction accuracy and scalability[6-9]. It partitions the users of an RS into clusters based on the rating data, and purifies the neighborhood by maximizing the similarities between users and their cluster centers. Besides user-based clustering, partitioning on items has also been explored[10], but was found less practical and useful for RS.

Similarities based on user rating behaviors express the correlation of users' preferences. This measurement has two key issues. First, it can be easily manipulated by attackers through creating fake profiles. Second, it causes false alarm on reporting a cluster of shilling users who can be a group of genuine users with very similar preferences.

Regular Paper

The preliminary version of the paper was published in the Proceedings of EDB2012.

In order to prevent recommendations from being mis-represented, we seek for other sources of information to guide the clustering of user profiles. The advent of social networks has made external sources of information available, like the "trust" that people place on one another, which has already been used in RS for improving the recommendations[6,8,11]. Social information is more difficult than user rating profiles to imitate[12]. Genuine users can receive incoming trust from anyone in the social graph, while fake users would be only trusted by other fake users. We are therefore motivated to explore how social-based clustering can secure RS.

In this work, we design two methods for building robust RS to prevent profile injection attacks through introducing the explicit trust from the social data in the process of clustering users. They are named as CluTr (clustering by using "trust" to filter out suspicious fake users) and WCluTr (clustering with weighed similarities derived from "trust"). We evaluate the effectiveness of our methods for preventing shilling attacks on data from a public consumer RS *Epinions.com*, and compare them with the ordinary CF approaches. The experimental results show that both proposed methods can highly improve the robustness of RS against shilling attacks. More specifically, our proposed algorithms are 20 times more robust than the conventional user-based CF algorithm in various attack models.

In the rest, we discuss related work in Section 2, describe designed methods to resist shilling attack in Section 3, report evaluation results in Section 4, and finally conclude the paper in Section 5.

2 Related Work

Shilling attacks potentially harm the recommender system and therefore they are receiving more and more attention along with the increasing importance of RS. Lam and Riedl explored several open questions regarding the effectiveness of shilling attacks (RandomBot and AverageBot)[1]. They found that user-based CF algorithms were less tolerant to shilling attacks than item-based CF algorithms. Watching for sharp changes in the value of traditional algorithm performance metrics such as mean absolute error (MAE) may be useful for detecting some attacks. However, many effective attacks will not be visible through simple aggregate metrics like MAE. In addition, new or obscure items, particularly in the user-based CF algorithm, are especially susceptible to attack.

Mobasher *et al.* studied six different attack models and measured their effectiveness in both user-based and item-based CF[13]. The prediction shift and hit ratio were used to measure how attacks can affect the RS. Their results show that the different types of attacks can effectively and practically harm the standard CF algorithms. The same authors in [14] presented a formal framework for specifying attack models and attack profiles and introduced a classification approach for attack detection. Their work mainly focused on showing the effectiveness of an attack, rather than the detection of attacks. Particularly, a type of segmented attack was studied. Unlike RandomBot and AverageBot that have no special target users, this segmented attack pushes an item to a targeted group of users with known or easily predicted preferences. Profiles are inserted that maximize the similarity between the pushed item and items preferred by the group. The segmented attack is both effective and practical against standard item-based CF algorithms.

The detection of shilling attack has also been studied. Su *et al.* worked on finding the group shilling, which is a type of coalition shilling attack[4]. They constructed a bipartite graph for the users and items, and used similarity spreading algorithm to find user clusters. They then labeled one cluster to be an abnormal group of shilling users, if the size (number of members) and average similarity of this cluster with the other clusters were smaller than pre-defined thresholds.

With the assumption that shillers work together and they are highly correlated, Mehta used probabilistic latent semantics analysis (PLSA) and principal component analysis (PCA) to eliminate the clusters of shilling users[2]. PLSA, a soft clustering method, computes a probabilistic distribution over communities (clusters of users) based on latent factors. It has been employed to remove much of the influence of biased attack profiles for model-based systems in [3]. PCA, a linear dimensionality reduction model, was used to select dimensions which are very different, or as in this work, very similar to other dimensions. The intuition of identifying the community (cluster of users) to be removed is that the cluster containing shilling profiles will be tighter, leading to lower average distances from the centroid of the cluster. Therefore, average Mahanalobis distance of a community was used to examine how closely knit a community is.

Zhang *et al.* detected the attacks by treating the ratings for an item as a time series according to their given time[5]. The sample average and sample entropy within a disjoint window of k consecutive ratings are calculated to capture the change in an item's likability and the distributional change in an item's ratings.

To challenge the RS, Cheng and Hurley[15] created the effective random attacks and average attacks with very low pair-wise similarities, dropping the assumption of high similarities among malicious attack profiles in

[2]. The authors showed that k-means and k-NN cannot detect such low diversity attacks. The same authors have shown in [16] that it is wrong to conclude that model-based algorithms are more robust than memory-based ones.

Trust is the relationship of users and partially reflects the acceptance of preferences of others. It has always been a concern for computer scientists and much work has been done to formalize it in computing environments[12]. Trust can either be implicit, that is automatically inferred from other data, or it can be explicitly expressed directly by the users, a very common characteristic of social networks. Employing forms of explicit or implicit trust to boost recommendations performance has been a subject of research in RS. Ziegler and Lausen studied the correlation between explicit trust and user similarity in RS[17]. They argued that similarity is specific to the application domain that the trust network is formed to. TrustWalker, proposed by Jamali and Ester, is a random walk model for combining trust-based and item-based recommendation[18]. Their approach aims to improve the performance of recommendations on sparse data with the utilization of trust network.

Massa and Avesani[19] have incorporated the explicitly provided trust links of users into the mechanism of prediction of item ratings. Ma *et al.* further investigated the influence of trusted friends in the on-line behaviour of a user, making use of the trust from the social graph in [20]. Furthermore, in [21] they elaborated how both user trust and distrust information can benefit the RS.

Despite clustering and trust have been combined together to improve recommendations[6,8], to the best of our knowledge there is no related work using both of them for protecting the RS. Yet more important, in previous research in recommender systems security, attacks on social profiles have not been studied in conjucntion with shilling profiles.

3 Shilling Attack Prevention by Clustering

As discussed in [1], the user-based CF is less resistant to attacks. Our paper aims at building a robust user-based CF which is resistant to the AverageBot push attack. In this section, we first present the mechanism of the push attack model and then introduce our two proposed methods: CluTr and WCluTr.

3.1 Shilling Attacks Models

AverageBot Model. A push attack aims to push up the rating of a target item. Intuitively, the attacker will inject a number of fake user profiles. These N_{fake} users rate the target item by the highest rating, e.g., 5, while they rate the other randomly selected N_{filler} items by samples of a normal distribution whose mean is equal to the averaged rating of these items given by a set of normal users. The high similarity between normal users and the injected fake users on rating the filler items is what makes these normal users potential victims.

We show an example of average attacks in Table 1, where two fake users are injected and they rate 5 (the highest rating) to the target item $\text{Item}_{\text{target}}$ that will be pushed to normal users and rate a filler item i by a random value sampled from $\mathcal{N}(\mu_i, \sigma)$. μ_i is the average rating of item i. σ is a constant set to 1.1 as suggested by [1] considering that this information is not available to attackers which are forced to make an educated guess. The non-target and non-filler items keep the record unchanged. The setting of parameters N_{fake} and N_{filler} in attack model will be discussed in Subsection 4.1.

Table 1. Example of Average Attacks

	Item_1	Item_2	Item_3	Item_4	$\cdots$	$\text{Item}_{\text{target}}$	$\cdots$
User_1	4		2			2	
User_2		5	3		$\cdots$		$\cdots$
$\text{User}_{\text{fake}}$	$\mathcal{N}(\mu_1, \sigma)$		$\mathcal{N}(\mu_3, \sigma)$			5	
$\text{User}_{\text{fake}}$	$\mathcal{N}(\mu_1, \sigma)$	$\mathcal{N}(\mu_2, \sigma)$				5	

Shilling Social Trust Model. We intend to use social "trust" information in our proposed RS. Apparently, attacks can inject shilling social trust too. We assumed that a malicious user would behave accordingly and pretend to trust normal users and his/her accomplices, following the intuitive scenario of attempting to establish social connections with victims. The different scenarios will be discussed in Subsection 4.1. This shilling trust model, along with the average attack model, assumes generation of random connectivity in the social network, originated from the fake users.

3.2 Clustering Using Explicit Trust

The two proposed attack-resistant schemes are based on clustering of users. The intuition behind the idea of applying clustering is that malicious users would be allocated into different clusters from the legitimate users, and thus their influence to targets will be minimized. We use a widely adopted clustering method, called k-means, to group the similar users. Theoretically, this method partitions a set of N points into a predefined number K of clusters $\{C_k, k = 1, \ldots, K\}$. Every data point x_i is associated with its nearest cluster center, one of $\{m_k, k = 1, \ldots, K\}$, where the cluster center m_k is the average of all data points that belong to this cluster. k-means clustering minimizes the sum of squared distance between each point and its cluster center:

$$\sum_{k=1}^{K}\sum_{x_i\in C_k} d(x_i-m_k)^2,$$

where

$$m_k=\frac{1}{|C_k|}\sum_{x_i\in C_k} x_i, \qquad (1)$$

and $|C_k|$ is the number of users belonging to cluster C_k.

In user-based CF when clustering users, the distance $d(x_i-m_k)^2$ is computed as the average of Pearson similarity between user x_i and all other users in the same cluster. In this case, the clustering aims to maximize the sum of inner-cluster averaged Pearson similarities:

$$\sum_{k=1}^{K}\frac{1}{|C_k|}\sum_{x_i\in C_k}\sum_{x_j\in C_k} S(x_i,x_j),$$

where $S(x_i,x_j)$ is the Pearson similarity between x_i and x_j. The k-means algorithm begins by choosing a number of centers and recursively adjusting the centers' locations. However this heuristic approach cannot guarantee a global minimum for the sum in (1) as it may converge close to a local minimum due to randomly selected initial cluster centers. In order to achieve more reliable results, a variation of k-means called k-means++ proposed by Arthur *et al.*[22] is employed in this paper. By incorporating carefully selected initial cluster centers, k-means++ increases the chances of reaching the global minimum. It was also shown to be able to reduce the computational time[22].

The algorithm comprises four steps as follows:

1) Randomly choose the first center x_{c1} among the data points $(x_1,x_2,\dots,x_n)$;

2) Calculate the distance $(d_1,d_2,\dots,d_n)$ of each data point to its closest center point (x_{c1} when only the first center is available);

3) Choose a new center from the remaining non-center data points. The probability p_i of a particular data point x_i being selected is proportional to the square of d_i calculated in step 2. p_i can be calculated using basic principle in probability;

4) Repeat steps 2 and 3 until k centers have been chosen. Then proceed with the standard k-means.

Pearson's correlation coefficient[23] in a recommender system is used to express the similarity in taste of two users as far as their rating profiles. The Pearson's correlation coefficient of users u and v is calculated as:

$$w_{u,v}=\frac{\sum(r_{u,k}-\bar{r}_u)(r_{v,k}-\bar{r}_v)}{\sqrt{\sum(r_{u,k}-\bar{r}_u)^2\sum(r_{v,k}-\bar{r}_v)^2}}, \qquad (2)$$

where $\bar{r}_u$ and $\bar{r}_v$ denote as the average of all ratings of user u and v respectively, while $r_{u,k}$ and $r_{v,k}$ are the ratings for item k given by users u and v.

To build robust RS by combining clustering and social trust, we design two different methods to enhance the genuine neighborhood: CluTr and WCluTr introduced in the following.

CluTr*: Clustering with Trust-Based Filtering.* In this method, users are firstly clustered by k-means++ based on their profile similarity, $w_{u,v}$ calculated by (2). To exclude the potentially fake users from the recommendation process, we apply filtering on the formed clusters. Following the well-founded conjecture in real social networks, suspicious and unreliable users are considered as those whose incoming trust from any other members of the same cluster is null. Moreover, to make the computation of predictions meaningful we exclude clusters whose population is smaller than a threshold t.

WCluTr*: Clustering on Weighted Social Similarity.* On top of rigidly filtering out the suspicious fake users in CluTr, we incorporate the trust information into the computation of the similarity between users and develop the method called WCluTr. We define a new function for measuring the social similarity of users, derived from their social connectivity with others. This idea is inspired from [8], where the prediction accuracy had been improved through weighting the similarities by trust. We call S_{average} the social similarity between two users:

$$S_{\text{average}}(u,v)=\frac{1}{2}\Big(\frac{|T_u\cap T_v|}{|T_u|}+\frac{|T_u\cap T_v|}{|T_v|}\Big),$$

where T_u and T_v are the sets of trusted users by user u and v, respectively. $|\cdot|$ measures the number of members in a user set.

We then combine it with the profile similarity $w_{u,v}$ computed with Pearson's formula to receive the new weighted similarity $w_{u,v}^{\text{weighted}}$.

$$w_{u,v}^{\text{weighted}}=w_{u,v}(1+\alpha S_{\text{average}}(u,v)). \qquad (3)$$

The parameter α is used to adjust the contribution of trust similarity. The setting of this parameter will be discussed in Subsection 4.1. $w_{u,v}^{\text{weighted}}$ is used for clustering in WCluTr.

Classic user-based CF systems employ Resnick's formula for the prediction of recommendation of user u on item i:

$$p_{u,i}=\bar{r}_u+\frac{\sum_{v\in C_u}[w_{u,v}(r_{v,i}-\bar{r}_v)]}{\sum_{v\in C_u}|w_{u,v}|}, \qquad (4)$$

where $\bar{r}_u$ is the average rating value of user u, $\bar{r}_v$ is the average rating value of user v, and $r_{v,i}$ is the rating of user v on item i. In our clustering-based methods, the neighborhood of user u is only considered inside his/her cluster C_u, meaning that a prediction in (4) would involve neighbors from a single cluster C_u instead of the whole population. In WCluTr algorithm, $w_{u,v}$ in (4) is replaced by $w_{u,v}^{\text{weighted}}$ similarity when computing predictions.

In general, CluTr and WCluTr algorithm proceed with the following steps:

1) obtain user cluster $(C_1, C_2, \ldots, C_K)$ by k-means++, using profile similarity $w_{u,v}$ for CluTr and $w_{u,v}^{\text{weighted}}$ similarity for WCluTr;

2) dismiss small clusters that have no sufficient number of members;

3) apply trust filtering described in Algorithm 1.

Algorithm 1. Filtering with Trust

```
for each cluster $C_k$ do
    for user $u$ in the cluster $C_k$ do
        if user $u$ has no incoming trust within the cluster
          then label it as belonging to NO cluster
        end if
    end for
end for
```

The whole process of CluTr and WCluTr algorithm is described in Fig.1.

4 Experimental Evaluation

In this section, we will report the data and evaluation results. CluTr and WCluTr are tested under various attack models. They are expected to make accurate recommendations for both the non-target items and more importantly the target items.

4.1 Data and Metrics

The dataset we use is taken from a commercial RS *Epinions.com*, collected by Paolo Massa in 2003[①]. The reason we choose this particular dataset is that it provides directional trust information along with user ratings for items. To limit the computational complexity of the testing set up, we use a subset of the data that contains randomly selected 5 000 users along with their 206 000 trust expressions and 1.1 million ratings for 103 000 products.

The push attacks are injected according to the AverageBot model as introduced in Subsection 3.1. The number of injected fake users N_{fake} is set to 25, 50 and 100 for evaluating the robustness of our designed methods w.r.t. various intensity of attacks. The number of filler items N_{filler} is set to $N_{\text{allitem}} \times 1\% = 1\,000$, where N_{allitem} is the total number of item (products) in the dataset. When clustering users into groups, the number of clusters produced by k-means is set to $K = 10$ so that the clusters would have sufficient number of users for deriving safe conclusions. The untrustable clusters are dismissed if their size is smaller than $t = 30$. The adjusting parameter α for weighting similarities in (3) is set to 3.5 as a result of optimization in repeating trials.

To examine whether attackers can take advantage of

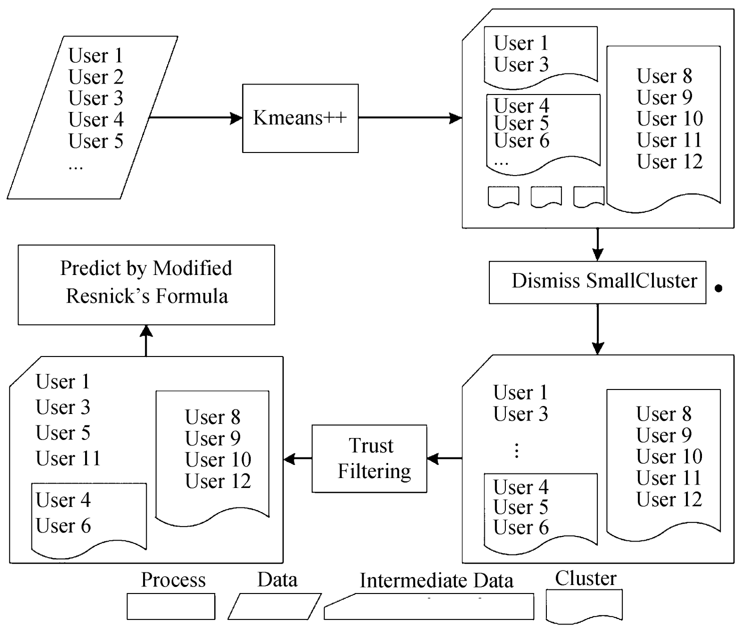

Fig.1. Flowchart of CluTr and WCluTr.

①TrustLet. http://www.trustlet.org/, Sept. 2012.

the knowledge in the trust system, we model four different scenarios of injecting shilling social trust information as discussed in Subsection 3.1. The intuition behind these scenarios is that a potential attacker would pretend to be as much as socially connected as a normal user:

1) none: no trust is injected;

2) trust1: attackers randomly trust 10% of users;

3) trust2: attackers trust all users;

4) trust3: attackers trust all fake users plus 10% of users so as to look more truthful to normal users.

Three evaluation metrics are employed to validate the two proposed algorithms. First, predictive accuracy should be measured for ensuring that the produced recommendations are least affected by the deployed defense mechanisms, along with the protection against attacks. We use the most popular accuracy metric mean absolute error (MAE) to the ratings masked out in this paper:

$$\mathrm{MAE} = \mathrm{mean}(|r_{u,j} - p_{u,j}|), \quad \forall u, j \text{ s.t. } r_{u,j} \text{ exists}.$$

We call *Error* the difference between an existing rating $r_{u,j}$ that a user u has given to item j and the value derived from the prediction algorithms $p_{u,j}$. The mean in function MAE is the arithmetic mean over all u, j pairs that $r_{u,j}$ exists. When item j is attacked, MAE will measure the error caused by both the system itself and the attack.

Second, prediction shift is the metric we use to measure the effectiveness of attacks in RS. As used in [1, 13], *prediction shift* is defined as the difference between the predicted rating before and after attacks. A positive value means that attacks have raised the rating. In our experiments, the effectiveness of a particular attack towards a particular item i is measured by

$$\Delta_i = 1/|U| \sum_{u \in U} (p'_{u,i} - p_{u,i}),$$

which is the average of prediction difference after $(p'_{u,i})$ and before attack $(p_{u,i})$ on all users U. To generate a reliable result, we launch a number of independent attacks on different target items i, and report the mean and probability distribution of prediction shift Δ_i on a set of samples.

Third, hit ratio measures the likelihood that a user changes his/her opinion on a product from *dislike* to *like*. A slight prediction shift may not affect the quality of recommendation, especially when a user has strong preference, i.e., give the highest or lowest rating on an item. Such cases cannot be captured by the prediction shift alone. We are interested in investigating significant impacts of attacks. As the ratings in our data range from 1 to 5, we set the rating of 3 as a neutral rating value. The hit ratio is computed by

$$H = \frac{1}{|U| \times |I|} \sum_{u \in U} \sum_{i \in I} h(u, i), \tag{5}$$

where

$$h(u, i) = \begin{cases} 1, & \text{if } p'_{u,i} > 3 \text{ and } p_{u,i} \leqslant 3, \\ 0, & \text{otherwise.} \end{cases}$$

U and I are sets of users and items for which predictions are availibale.

4.2 Experimental Results

This subsection reports the comparison results of the baseline method and proposed algorithms measured by three metrics.

4.2.1 Mean Absolute Error

To ensure that the protection mechanism of proposed algorithms does not affect the recommendation in regular cases, we first measure the performance of algorithms without shilling attacks. We employ the MAE metric mentioned in Subsection 4.1. Smaller MAE means better results.

Table 2 demonstrates performance comparison of our algorithms against the baseline approach, which is the conventional user-based CF algorithm without clustering and trust information. CluTr and WCluTr have better performance on predicting users rating than the baseline collaborative filtering algorithm without the purpose on shilling attack prevention. We also compare against using weighted similarity alone in the baseline CF as part of the defense scheme. The worse performance of weighted similarity alone implies that clustering is the key factor for the better performance of CluTr and WCluTr.

Table 2. MAE of Weighted Similarity, CluTr and WCluTr Algorithms Compared with That of the Baseline Algorithm Without Clustering and Trust when no Attacks

Algorithms	MAE
Baseline CF	0.803 5
Weighted similarity in baseline CF	0.801 9
CluTr	0.741 3
WCluTr	0.738 0

With the confidence on CluTr and WCluTr from these positive evaluation results, we next evaluate how successfully CluTr and WCluTr can protect RS systems from shilling attacks.

In Table 3, we show the accuracy measure MAE of various algorithms when different numbers of fake users are injected in different trust attack scenarios, as discussed in Subsection 4.1. In each case, the smallest MAE is highlighted in bold. It can be seen that CLUTR and WCLUTR outperform the other alternative schemes in all cases, while WCLUTR is even better than CLUTR. When larger number of fake users are injected, the baseline method performs poorly with higher MAE. However, CLUTR and WCLUTR consistently have lower MAE than 1 regardless of all types of attacks.

Table 3. MAE of CLUTR and WCLUTR Algorithms Compared with That of the Baseline Algorithm Without Clustering and Trust, when RS are Threatened with Various Attacks

Number of Fake Users	Trust Attack Scenarios	MAE		
		No Clustering	CLUTR	WCLUTR
25	None	1.019 6	0.876 9	**0.875 3**
	Trust1		0.880 4	**0.871 3**
	Trust2		0.839 0	**0.828 6**
	Trust3		0.863 8	**0.852 0**
50	None	1.189 1	0.888 4	**0.886 2**
	Trust1		0.922 6	**0.916 2**
	Trust2		0.818 4	**0.817 4**
	Trust3		0.811 7	**0.808 2**
100	None	1.323 5	0.830 5	**0.822 5**
	Trust1		0.861 5	**0.810 0**
	Trust2		0.836 7	**0.830 4**
	Trust3		0.901 5	**0.892 0**

4.2.2 Prediction Shift

In Table 4, we show the prediction shift averaged on an item set I, $\bar{\Delta} = 1/|I| \sum_{i\in I} \Delta_i$, for all algorithms. For some rare situations when no user in a cluster rated an item, probably for the reason that the rating users are filtered out due to trust issues, we make up the prediction of rating by using the conventional user-based CF. That is in favor of the attackers as it introduces undependable prediction.

As seen in Table 4, the conventional user-based CF algorithm without clustering and trust is badly affected by average attacks with mean prediction shift valued from 0.85 to 1.5. Such high rating shift is potentially enough to change the victim's opinion of a target item. Especially when more fake users are injected, the predicted ratings are pushed to a higher value.

Compared with the baseline CF, our proposed method CLUTR and WCLUTR have much lower prediction shift. When attackers inject fake users, the pushed ratings are shifted by a value smaller than 0.85, which is less likely to change a user's opinion on an item from "dislike" to "like". Moreover, CLUTR and WCLUTR have good performance no matter how the attackers inject shilling trust information in the four different trust attack scenarios.

Table 4. Prediction Shift of CLUTR and WCLUTR Algorithms Comparing to That of the Baseline Algorithm without Clustering and Trust, when RS are Threatened with Various Attacks

Number of Fake Users	Trust Attack Scenarios	Prediction Shift		
		No Clustering	CLUTR	WCLUTR
25	None	0.850 6	0.407 5	**0.403 9**
	Trust1		0.399 5	**0.397 6**
	Trust2		**0.380 1**	0.384 1
	Trust3		0.349 1	**0.346 7**
50	None	1.2290	0.666 5	**0.659 6**
	Trust1		0.622 4	**0.600 2**
	Trust2		0.591 6	**0.580 8**
	Trust3		0.517 9	**0.507 4**
100	None	1.579 1	0.803 3	**0.791 4**
	Trust1		0.831 1	**0.810 0**
	Trust2		0.819 4	**0.804 4**
	Trust3		0.734 2	**0.730 8**

To investigate the distribution of prediction shift Δ_i in I, we show the cumulative distribution function (CDF) of Δ_i in Figs. 2 and 3. CDF describes the probability that the prediction shift Δ_i will be found at a value less than or equal to x, i.e., $\Pr(\Delta_i \leqslant x)$. A robust RS should have Δ_i as small as possible, i.e., the CDF curve should go up quickly to 1 with Δ_i increasing from 0 to a small value. A CDF curve close to the left-top corner of the diagram indicates a good model.

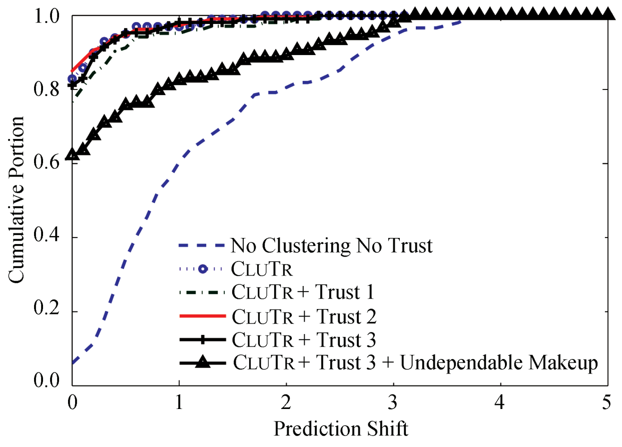

Fig.2. CDF of prediction shift Δ_i in CLUTR and baseline CF against different scenarios of trust attacks.

In Fig.2, we show the CDF of the prediction shift of the baseline CF without clustering and trust (No clustering No Trust), and our method CLUTR evaluated on four different scenarios of injecting fake trust information. We include one evaluation result of CLUTR with undependable makeup introduced by conventional CF, and four results without it. Obviously, CluTr with undependable makeup is more affected by the attacks.

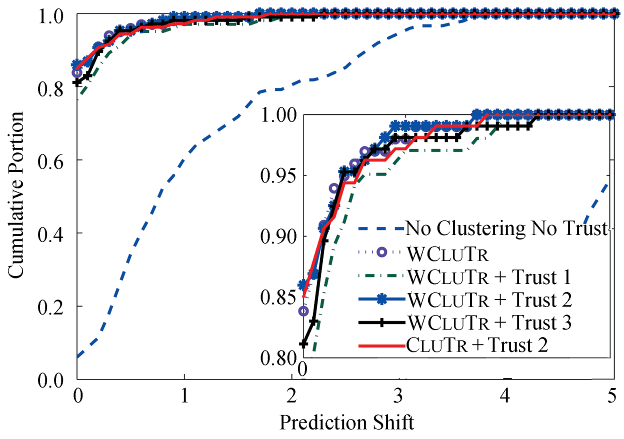

Fig.3. CDF of prediction shift Δ_i in WCluTr, baseline CF and CluTr against trust2 scenario.

Given the scenario that prediction shift should not exceed 1 (or 0.5), we find that $\Pr(\Delta_i \leqslant 1) = 0.604$ (or $\Pr(\Delta_i \leqslant 0.5) = 0.34$) in the baseline CF, and $\Pr(\Delta_i \leqslant 1) \approx 0.981$ (or $\Pr(\Delta_i \leqslant 0.5) = 0.95$) in CluTr. That is to say, the prediction shift Δ_i in the baseline CF has 40% (66%) chance to exceed 1 (0.5), while it has only 2% (5%) chance to exceed 1 (0.5) in CluTr.

We can interpret the promising results of CluTr method as: when average attacks happen, only 2% (5%) of the target items will be affected by raising the predicted rating by 1 (0.5). In overall, CluTr is 20 times more robust than the conventional user-based CF algorithm. No matter how the attackers inject fake trust information, CluTr has good performance in all cases.

Fig.3 shows that WCluTr is as good as CluTr in terms of robustness. The embedded plots compare the performance of WCluTr with that of CluTr bothered by trust2, in which scenario CluTr performs better than the other scenarios as shown in Fig.2. Observing the CDF curves, we see that WCluTr performs slightly better than CluTr.

4.2.3 *Hit Ratio*

In Table 5, we show the hit ratio calculated by (5) for all algorithms. As we can see, CluTr and WCluTr algorithms have smaller hit ratio than the baseline method. With the protection of CluTr and WCluTr, the likelihood that attackers successfully affect the preference of a user on an item is around 0.02. However, the likelihood that users change their opinion from "dislike" to "like" is larger than 0.045 when RS employing the baseline method are attacked. This observation is consistent with the distribution of prediction shift given in Figs. 2 and 3, where we found that the prediction shift have only 2% chance to exceed 1 in CluTr and WCluTr.

Table 5. Hit Ratio of CluTr and WCluTr Algorithms Compared with That of Baseline Algorithm Without Clustering and Trust, when RS are Threatened with Various Attacks

Number of Fake Users	Trust Attack Scenarios	Hit Ratio		
		No Clustering	CluTr	WCluTr
25	None	0.045 8	0.025 4	**0.025 1**
	Trust1		0.019 9	**0.019 6**
	Trust2		**0.022 6**	**0.022 6**
	Trust3		0.019 2	**0.018 8**
50	None	0.060 7	0.026 6	**0.025 4**
	Trust1		0.027 2	**0.026 5**
	Trust2		**0.025 2**	0.025 5
	Trust3		0.027 6	**0.026 6**
100	None	0.076 8	0.028 9	**0.028 7**
	Trust1		0.029 9	**0.028 8**
	Trust2		0.032 1	**0.031 8**
	Trust3		0.034 9	**0.033 9**

All the above findings in measures of three criteria show that CluTr and WCluTr can effectively protect the RS from shilling attacks.

5 Discussion and Conclusions

In this paper, we proposed two attack-resistant algorithms for RS, CluTr and WCluTr, which combine the trust information with clustering for building robust RS against shillers. CluTr uses trust to filter out the suspicious fake users in the formed clusters. WCluTr additionally uses trust information to strengthen the similarities among genuine users and to weaken the similarities between fake users and others. To present the shilling attack scenarios, we studied one popular shilling attack, AverageBot which affects strongly the user-based CF algorithm. In addition, we introduced and evaluated four scenarios that can be implemented by attackers to inject shilling social trust. In these scenarios, attackers obfuscate the mutual trust among genuine users.

We evaluated our proposed method on the data with various attack scenarios. The experimental results show that our proposed methods CluTr and WCluTr are much more robust than conventional user-based CF algorithm against average attacks. When average attacks happened, as interestingly shown in our results, only 2% (5%) of the target items are affected by raising the predicted rating by 1 (0.5). The likelihood that users' opinion changes from "dislike" to "like" on a pushed product is about 0.02 to 0.03, which is much smaller than that in the baseline method.

The popularity of social network brings us the chance to derive more social link information. Studying the usage of such information for enhancing RS is in our future plans.

References

[1] Lam S K, Riedl J. Shilling recommender systems for fun and profit. In *Proc. the 13th International Conference on World Wide Web* (*WWW*), May 2004, pp.393-402.

[2] Mehta B. Unsupervised shilling detection for collaborative filtering. In *Proc. the 22nd National Conference on Artificial Intelligence*, Jul. 2007, pp.1402-1407.

[3] Mobasher B, Burke R D, Sandvig J J. Model-based collaborative filtering as a defense against profile injection attacks. In *Proc. the 21st National Conference on Artificial Intelligence*, Jul. 2006, pp.1388-1393.

[4] Su X F, Zeng H J, Chen Z. Finding group shilling in recommendation system. In *Special Interest Tracks and Posters of the 14th WWW*, May 2005, pp.960-961.

[5] Zhang S, Chakrabarti A, Ford J, Makedon F. Attack detection in time series for recommender systems. In *Proc. the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Aug. 2006, pp.809-814.

[6] DuBois T, Golbeck J, Kleint J, Srinivasan A. Improving recommendation accuracy by clustering social networks with trust. In *Proc. ACM Workshop on Recommender Systems & the Social Web*, Oct. 2009.

[7] Sarwar B M, Karypis G, Konstan J, Riedl J. Recommender systems for large-scale e-commerce: Scalable neighborhood formation using clustering. In *Proc. the 5th Int. Conf. Computer and Information Technology*, Dec. 2002.

[8] Pitsilis G, Zhang X L, Wang W. Clustering recommenders in collaborative filtering using explicit trust information. In *Proc. the 5th IFIP WG 11.11 International Conference on Trust Management* (*IFIPTM*), Jun. 2011, pp.82-97.

[9] Truong K Q, Ishikawa F, Honiden S. Improving accuracy of recommender system by item clustering. *Transactions on Information and Systems*, 2007, E90-D(9): 1363-1373.

[10] O'Connor M, Herlocker J. Clustering items for collaborative filtering. In *Proc. SIGIR 2001 Workshop on Recommender Systems*, Sept. 2001.

[11] O'Donovan J, Smyth B. Trust in recommender systems. In *Proc. the 10th International Conference on Intelligent User Interfaces*, Jan. 2005, pp.167-174.

[12] Marsh S P. Formalising trust as a computational concept [Ph.D. Thesis]. University of Stirling, Apr. 1994.

[13] Mobasher B, Burke R, Bhaumik R, Williams C. Effective attack models for shilling item-based collaborative filtering systems. In *Proc. WebKDD Workshop*, Aug. 2005.

[14] Mobasher B, Burke R, Bhaumik R, Williams C. Toward trustworthy recommender systems: An analysis of attack models and algorithm robustness. *ACM Trans. Internet Technol.*, Oct. 2007, 7(4): Article No.23.

[15] Cheng Z P, Hurley N. Effective diverse and obfuscated attacks on model-based recommender systems. In *Proc. the 3rd ACM Conf. Recommender Systems*, Oct. 2009, pp.141-148.

[16] Cheng Z P, Hurley N. Robustness analysis of model-based collaborative filtering systems. In *Proc. the 20th Irish Conf. Artificial Intelligence and Cognitive Science*, Aug. 2009, pp.3-15.

[17] Ziegler C, Lausen G. Analyzing correlation between trust and user similarity in online communities. In *Proc. the 2nd Int. Conf. Trust Management*, May 2004, pp.251-265.

[18] Jamali M, Ester M. TrustWalker: A random walk model for combining trust-based and item-based recommendation. In *Proc. the 15th ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining*, Jun. 2009, pp.397-406.

[19] Massa P, Avesani P. Trust-aware recommender systems. In *Proc. the 1st ACM Conference on Recommender Systems*, Oct. 2007, pp.17-24.

[20] Ma H, King I, Lyu M R. Learning to recommend with social trust ensemble. In *Proc. the 32nd Int. ACM SIGIR Conf. Research and Develop. in Inform. Retrieval*, Jul. 2009, pp.203-210.

[21] Ma H, Lyu M R, King I. Learning to recommend with trust and distrust relationships. In *Proc. the 3rd ACM Conference on Recommender Systems*, Oct. 2009, pp.189-196.

[22] Arthur D, Vassilvitskii S. k-means++: The advantages of careful seeding. In *Proc. the 18th Annual ACM-SIAM Symposium on Discrete Algorithms*, Jan. 2007, pp.1027-1035.

[23] Rodgers J, Nicewander A. Thirteen ways to look at the correlation coefficient. *The American Statistician*, 1988, 42: 59-66.

Xiang-Liang Zhang is currently an assistant professor and directs the Machine Intelligence and knowledge Engineering Laboratory in King Abdullah University of Science and Technology (KAUST), Saudi Arabia. She was an European ERCIM research fellow in the Department of Computer and Information Science, Norwegian University of Science and Technology (NTNU), Norway, in 2010. She earned her Ph.D. degree in computer science from INRIA-University Paris-Sud 11, France, in July 2010. She received M.S. and B.S. degrees from Xi'an Jiaotong University, China, in 2006 and 2003, respectively. Her main research interests and experiences are in machine learning, data mining, and cloud computing.

Tak Man Desmond Lee was a master student of computer science in King Abdullah University of Science and Technology (KAUST), Saudi Arabia. He is currently working in Saudi Aramco.

Georgios Pitsilis received his B.Eng. degree in engineering and B.Eng. degree in informatics from the Technological Institution of Athens, Greece, in 1991 and 1999 respectively, MSc. degree from Oxford Brookes University, UK in 2000, and Ph.D. degree in computer science from Newcastle University, UK in 2007. He has worked as a postdoc researcher at the Universities of Luxembourg, NTNU, and Newcastle University for 5 years. His research interest are in the areas of trust management, recommender systems, data mining and distributed computing.