Unsupervised Shilling Detection for Collaborative Filtering

Bhaskar Mehta
L3S / University of Hannover, Appelstrasse 4, 30167 Hannover, Germany

Abstract

Collaborative Filtering systems are essentially social systems which base their recommendation on the judgment of a large number of people. However, like other social systems, they are also vulnerable to manipulation. *Lies and Propaganda* may be spread by malicious users who may have an interest in promoting an item, or downplaying the popularity of another one. By doing this systematically, with either multiple identities, or by involving more people, malicious *shilling* user profiles can be injected into a collaborative recommender system which can significantly affect the robustness of a recommender system. While current detection algorithms are able to use certain characteristics of shilling profiles to detect them, they suffer from low precision, and require a large amount of training data. The aim of this work is to explore simpler unsupervised alternatives which exploit the nature of shilling profiles, and can be easily plugged into collaborative filtering framework to add robustness. Two statistical methods are developed and experimentally shown to provide high accuracy in shilling attack detection.

Introduction

Collaborative filtering technology is being widely used on the web as an approach to information filtering and recommendation by commercial service providers like *Amazon* and *Yahoo!*. For filtering in multimedia data, where pure content based recommendations perform poorly, collaborative filtering is the most viable and effective solution, and is heavily used by providers like *YouTube* and *Yahoo! Launchcast*. For malicious attackers, or a group interested in popularizing their product, there is an incentive in biasing the collaborative filtering technology to their advantage. Such activity is similar in nature to spam observed widely on the web, e.g. link farms for search engine manipulation. Since collaborative filtering is based on social networking, it is also vulnerable to social attacks, i.e. a group of users working together to bias the system. A lot of electronic systems, especially web-enabled ones provide free access to users via simple registration processes. This can be exploited by attackers to create multiple identities for the *same* system and insert ratings in a manner that manipulates the system. *Profile injection attacks* add a few profiles (say 1-3% of the total profiles) which need to be identified and protected against. Such attacks have been refered to as *shilling* attacks(Lam & Riedl 2004), and the added profiles are called *shilling profiles*. Further, profile injection attacks can be classified in two basic categories: inserting malicious profiles which rate a particular item highly, are called *push* attacks, while inserting malicious profiles aimed at downgrading the popularity of an item are called *nuke* attacks(O'Mahony *et al.* 2004). Since shilling profiles looks very similar to an authetic user, it is a difficult task to correctly identify such profiles. In this work, we focus on detecting push attacks: nuke attacks can be detection using the same methodology.

The current techniques in detection are based on reverse engineered heuristics which perform sub-optimally. In particular, by looking only at individual users and not the combined effect of such malicious users, current detection algorithms have low accuracy in detecting shilling profiles. In this work, we provide an indepth analysis of shilling profiles and describe new approaches to detect malicious shilling profiles. In particular, we provide unsupervised algorithms which are highly accurate and fast. We also look in depth at properties of shilling profiles, and analyze optimal shilling strategies which use item means. Note that we concentrate on unsupervised methods since they involve much lesser computational effort as compared to supervised approaches, especially if training data has to be generated. Moreover, we concentrate on those methods which can be easily plugged into existing CF framework.

CHARACTERISTICS OF SHILLING PROFILES

The common property of all shilling detection algorithms (for collaborative filtering) is that they exploit specific properties of profiles injected in order to identify them. After reverse engineering the profile signatures, appropriate heuristics are used to capture information which characterizes shilling users. This is similar to the 2-player model where a move is made by maximizing a certain objective. In order to understand why shilling detection algorithms work, or don't work, one needs to understand the goals of shilling users and methods used to achieve them.

The primary objective of shilling is to maximize (or minimize in the case of nuke attacks) the predicted value for the chosen item for the largest possible number of users. This can be achieved by constructing profiles which are highly

Table 1: No. of top-20 neighborhoods that each user belongs to after adding 5% shillers with 3% filler using Avg. attack profiles

# Neigh	0-20	20-40	40-60	60-80	80-100	100-120
Normal	818	253	62	15	19	9
Shilling	0	1	10	13	17	9

correlated to a fraction of the users and affect them significantly.

In order to achieve these objectives, profiles have to be constructed in a special manner. Most attack strategies involve rating items around the mean vote, which minimizes the deviation from other existing votes, except the item under attack. Usually, only a subset of the item set is voted on by a shilling profile; this is called the *filler size* is reported as a percentage of the item space. Filler items are usually selected at random.

Various attack models have been studied in literature recently(average, random, bandwagon & segment attack[1]). The results of these studies show that the impact of well constructed profiles can be huge. Even a 1% attack (number of shilling profiles) can skew the system and push the attacked item to top of the ratings. Such attacks are especially severe for items without many votes where shilling users can easily become highly authoritative and force higher ratings. The most effective attack is the average attack where small attack sizes can cause large deviations in the targetted item; it is usually also the most difficult attack to detect. We focus on detecting average attacks in this paper. The specific construction of shilling profiles also have interesting properties some of which are used by detection algorithms:

1. *Low deviation from mean vote value, but high deviation from the mean for the attacked item*: RDMA (*Rating deviation from Mean Agreement*) and WDA (*Weighted Degree of Agreement*) are statistical measures which are based on this idea. The reason for this property is that by placing most votes close to the mean, similarity with other users (say Pearson's correlation) is increased significantly.
2. *High similarity with large number of users*: Shillers[2] have a high correlation with a significant number of users due to the mean-like votes for most items. A direct result of being highly correlated with a user is that a shilling user becomes an authoritative neighbor and figures prominently in the set of k-nearest neighbors. Fig 1. shows the high correlation pattern observed for 20 shillers, compared with 50 normal users. Notice that shilling users are neighbors for a much larger number of users, in comparison to authentic users.
3. *Shillers work together*: A large fraction of top-20 neighbors for all users are shillers for a well constructed attack. Shillers magnify each other's effect and together push the attacked item to a significantly higher rating. While this is an important characteristic, no algorithms have used this for detection so far. Experiments show that after a shilling attack, the top-20 neighbors of every user are full of shilling users. Table 1. demonstrates these properties for a bandwagon attack[3]. According to our estimates, with a small attack of 3% shilling profiles, approximately 15% of every user's closest neighbors are shilling users.
4. *Shillers are highly correlated*: Shillers tend to have very high correlation coefficients (> 0.9) due to the same underlying model which is used to generate them. Average attacks and random attacks have been observed to have this characteristic, and previous work has also used this characteristic to construct a metric which captures the average similarity for top-25 neighbors. Fig 1. also highlights this pattern.

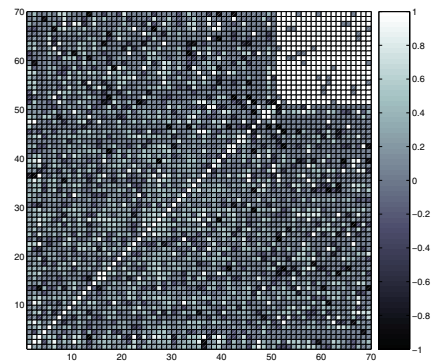


Figure 1: Shillers are highly correlated: 50 authentic profiles and 20 shilling profiles are used for calculating the Pearson's Correlation coefficient. Notice how shilling users exhibit a noticably higher degree of correlation

OPTIMAL SHILLING STRATEGY

In this section, we discuss what the optimal strategy for a shiller should be while constructing a shilling profile. Assume that the end system $\mathcal{S}$ has n users ($u_1 ... u_n$) and m items. We use the notation $v_{u_i,y}$ for the vote give to an item y by a user u_i, and $\hat{v_i}$ denotes the average vote of a user u_i. $C_{i,j}$ is the Pearson's correlation coefficient between user u_i and u_j.

We assume that the system provides recommendation using Pearson's correlation based collaborative filtering. In this scheme, a user's vote on an unknown/unvoted item is calculated based on the votes of other users who are similar to the current user. In a general scheme, it is also possible to use all users, and weight their opinion with their similarity to the current user. Formally, the predicted vote for user u_i for an item y can be expressed as

$$v_{u_i,y} = \hat{v_i} + \frac{\sum_j C_{i,j}(v_{u_j,y} - \hat{v_j})}{\sum_j |C_{i,j}|} \quad (1)$$

[1]The interested reader is referred to (Burke *et al.* 2006) for a detailed study on how shilling profiles are constructed.

[2]We use the term *shiller* to denote a user which a shilling profile points to.

[3]All data and plots in this paper are generated using the MovieLens dataset with 100,000 votes, with 944 users and 1682 movie-items.

The Pearson's correlation coefficient is calculated according to the following equation:

$$C_{i,j} = \frac{\sum_y (v_{u_i,y} - \hat{v_i})(v_{u_j,y} - \hat{v_j})}{\sqrt{\sum_y (v_{u_i,y} - \hat{v_i})^2}\sqrt{\sum_y (v_{u_j,y} - \hat{v_j})^2}} \quad (2)$$

Note that the correlation coefficient is measure only over items that two users have commonly voted on. Let us add a shilling user s to the user set. This shilling user wishes to cause item $\mathcal{I}$ to be recommended more often. The strategy to do this is to change the predicted value of the item $\mathcal{I}$ for as many users as possible. An effective attack would make this value as high as possible. *Prediction shift* is a measure used in literature to measure how effective an attack is. It is defined as the difference in the predicted value of an item before and after a shilling attack.

$$\mathcal{P} = \sum_u \grave{v}_{u_i,y} - v_{u_i,y} = \sum_u \mathcal{P}_u \,, \quad (3)$$

where $\grave{v}_{u_i,y}$ denotes the predicted value of item y for user u_i after an attack and $\mathcal{P}_u$ denotes the prediction shift in user u. Thus the aim of the shilling user s is to maximize the prediction shift $\mathcal{P}$. Clearly, the attacked item is rated as v_{max} (the maximum allowed rating) by the shilling user to have the maximum deviation. Also, the total shift is maximized when each of the respective prediction shifts $\mathcal{P}_u$ are maximized.

$$\mathcal{P}_u = \frac{\sum_j C_{i,j}(v_{u_j,y} - \hat{v_j}) + C_{i,s}(v_{max} - \hat{v}_s)}{\sum_j |C_{i,j}| + |C_{i,s}|} - const$$

$\mathcal{P}_u$ can be written as a function of the correlation coff $C_{i,s}$ (replacing $C_{i,s}$ by x) of the form.

$$\mathcal{P}_u = \frac{\kappa_1 + \kappa_2 x}{\kappa_3 + |x|} - const$$

Note that the correlation coefficient lies in [1-,1] and $\acute{P}_u$ is positive everywhere in [0,1] making P_u a strictly increasing function; the maximum value of P_u is reached at $x = 1$. Thus the overall prediction shift is maximized if the correlation coefficient of the shilling profile is maximized with all the user profiles. If the neighborhood of every user is also limited to a fixed size, then clearly, the impact of the shilling profile is maximum if the shilling user is a part of these neighborhood. Since the neighbors are formed based on the Pearson's correlation, maximizing the correlation with maximum users is the primary objective of shillers.

Maximizing correlation with maximum users

The above analysis shows that a shilling profile must be constructed to maximize correlation with the maximum number of users. Here, we try to motivate the use of mean item votes for maximizing the correlation coefficient. We use concepts used in Canonical Correlation Analysis to analyze the optimal strategy: Canonical correlation analysis seeks vectors a and b such that the random variables $a'X$ and $b'S$ maximize the correlation $\rho = cor(a'X, b'S)$.

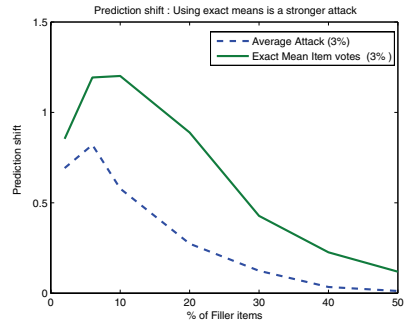


Figure 2: Prediction shift as a function of filler size measured over the complete user population(excluding shilling users). Notice how using exact means has a much stronger prediction shift as compared to an average attack.

Let us construct a random variable $X = (X_1, \ldots, X_n)'$ where X_i represents the ith data (user profile i). Let S represent the shiller's profile. We would like to maximize the correlation between Y and X with the additional constraints that all users are given equal weight. This leads us to use $a = (\frac{1}{n}, \ldots, \frac{1}{n})$. Trivially, $b = 1$. Note that $a'X$ leads to the average of the user profiles (we represent the mean vote of an item y_k by $\hat{y_k}$).

$$a'X = \sum_i \frac{1}{n} X_i = \hat{X} \sim (\hat{y_1}, \ldots, \hat{y_m}) \quad (4)$$

The expression to maximize now is

$$\rho = \frac{cov(\hat{X}, S)}{\sqrt{var(\hat{X})var(S)}} = \frac{\sum_i (\hat{y_i} - \hat{u})(s_i - \hat{s})}{\sqrt{\sum_i (\hat{y_i} - \hat{u})^2}\sqrt{\sum_i (s_i - \hat{s})^2}}$$

where $\hat{y_i}$ represents the average vote for an item i. and $\hat{u}$ denotes the overall average. It is easy to see that placing $s_i = \hat{y_i}$ maximizes the above expression to make $\rho = 1$ (vector differentiation w.r.t s produces the same result). This implies that the optimal strategy for maximizing correlation with all users is to use mean votes for individual items. Attack generation models discussed in (Burke *et al.* 2006) also use this idea for filler votes with the addition of gaussian noise to make the profiles more varied. Note that attacking an item y_i requires placing the maximum vote for this item; however this does not significantly effect the correlation with other users, since the other votes are still based around the item mean.

DETECTION ALGORITHMS

Current feature based algorithms tend to pick users with the maximum impact in terms of the measures/features used. However authentic users who are authoritative and different from many other users can also show significant impact and be falsely classified. Importantly, by working in groups, the effect of shilling users is large only in groups, and individual shilling profiles can be undetectable especially when in small numbers. Thus it makes sense to eliminate clusters of shilling users, rather than individual shilling users. Below, we outline two algorithms based on this intuition: PLSA

is a mixture model which computes a probabilistic distribution over communities (clusters of users) based on latent factors and has been reported to be robust to shilling(Mobasher, Burke, & Sandvig 2006); PCA is a linear dimensionality reduction model which can be used to select dimensions which are very different, or as in this work, very similar to other dimensions.

Soft clustering using PLSA

Probabilistic Latent Semantics Analysis(PLSA)[4] is a well known approach for text analysis and indexing used to discover hidden relationships between data. It is a highly successful approach for indexing documents and has been well research. Extensions to handle Collaborative filtering are also extremely popular; PLSA enables the learning of a compact probabilistic model which captures the hidden dependencies amongst users and items. It is a graphical model where latent variables are used to render users and items conditionally independent. The hidden variables can be interpreted as a probability distribution over communities of users or clusters; each user is allowed to be a part of multiple clusters, with a certain probability. The patterns in data along with the model fitting algorithm ensure that the learnt distribution minimizes the log-likelihood of the data.

While accuracy has been a well known advantage of PLSA, recent studies have also concluded that PLSA is a very robust CF algorithm, and is highly stable in the face of shilling attacks. (Mobasher, Burke, & Sandvig 2006) indicates that the prediction shift for PLSA is much lower than similarity based approaches. However, a clear explanation for this has not been provided so far. We investigated the reasons for PLSA's robustness over many experiments and observed the model to understand the mechanisms. The intuition is that PLSA leads to clusters of users(and items) which are used to compute predictions, rather than directly computing neighbors. However this intuition is challenged by experimental results using a k-means clustering algorithm in the same work. Clearly, shilling profiles deceive clustering algorithms due to their high similarity with normal users.

PLSA is a mixture model where each data point has its own distribution. Membership to a distribution is however not constrained; a data point can belong (probabilistically) to many distributions, with combination weights chosen so that the observed ratings are explained best. This results in *soft clustering* where a data point can lie in multiple clusters. We posit that this is also the reason why shilling is less effective against PLSA: shilling users are close to many users, but often dominant in one cluster due to their extraordinary similarity. Since user ratings are more noisy than shilling profiles, the likelihood of user ratings being explained by shilling profiles is limited, though not minuscule. This explanation has also been verified experimentally: We learn an model of an EachMovie data-subset with 5000 users to which 200 shilling profiles are added. On learning a PLSA model with *40* communities, we select the dominant community for each user. On analysis, we notice that all the shillers are in either 1 or 2 communities. By correctly identifying this community, we can isolate the shillers and remove them.

Identifying the community to be removed is vital: noticing how the profiles are close to each other, we have to identify a measure which examines how closely knit a community is: one possibility is to use Mahanalobis distance, which is traditionally used to identify outliers in multivariate data. We suggest using the average Mahanalobis distance of a community as follows: for each community $\mathcal{C}$ which is set of users, we find the Mahanalobis distance d_u of each user u as

$$d_u = \sqrt{(u - \overline{u})\mathbf{C}_0^{-1}(u - \overline{u})^T} \ , \tag{5}$$

where the matrix $\mathbf{C}_0$ is the covariance matrix of the community $\mathcal{C}$, and $\overline{u}$ is the mean profile over the same. Notice how $d_u > 0$ since $\mathbf{C}$ is positive semi-definite(PSD). We measure the 'closeness' of the community $\mathcal{C}$ by the average Mahanalobis distance over the user-set of $\mathcal{C}$. The intuition is that the cluster containing shilling profiles will be tighter, leading to lower average distances from the centroid of the cluster.

Initial implementation showed that computing Mahanalobis distances is very time consuming due to the inversion of large covariance matrices. To get around this, we observe that a fixed Mahanalobis distance defines a hyper-ellipsoid which is scaled by the variance in observed data in a direction. If variances are assumed to be one, *Mahanalobis* distance reduces to *Euclidean* distance. Based in this observation, we use z-scores[5] instead of actual votes to find the closeness of a cluster, and thus use the simpler Euclidean distance measure:

$$d_u = \sqrt{(u - \overline{u})(u - \overline{u})^T} \ , \tag{7}$$

Experimental results have showed that these two measures correlate very well if using z-scores.

Algorithm 1 PLSASelectUsers ($\mathbf{D}$)

1: $\mathbf{D} \leftarrow$ z-scores($\mathbf{D}$)
2: Train a PLSA model($P(z|u), P(y|z)$) for $\mathbf{D}$.
3: **for all** Users $u \in \mathbf{D}$ **do**
4: $\quad Comm_u = k$ where $P(z_k|u)$ is maximum
5: **end for**
6: **for all** Community k **do**
7: $\quad U_k \leftarrow$ The set of users u with $Comm_u = k$
8: $\quad Distance(k) \leftarrow \frac{1}{n}\sum_{u \in U_k}(u - \overline{U_k})^2$
9: **end for**

Output: Return U_k with smallest $Distance$ value

[4]Due to lack of space, PLSA is not explained here. Please see (Hofmann 2004) for details.

[5]*z-scores* can be computed for a user u for a item y, where the user has voted $v_{u,y}$, by using the following equation:

$$z_{u,y} = (v_{u,y} - \hat{v}_u)/\sigma_u, \tag{6}$$

Variable Selection using PCA

Experimental results show that shilling users have very high similarity to one another, and thus might form a cluster. However shilling users are designed to be very similar to all other users as well, which can make clustering problematic. We can at best hope to discover one cluster of highly correlated shilling users. Conventional techniques which assume multiple distributions in underlying data cannot successfully deal with such data.

What we want to do is to exploit the highly inter-correlation structure to find a group of shilling users which vary least w.r.t. each other. *Principal Component Analysis*(PCA) is a multivariate analysis technique used to find the intrinsic dimensionality of high dimensional data by projecting it to a low dimensional space where the axes selected capture the maximum variation in data[6]. If a dataset has variables which are very similar and highly correlated, then these variables would uninteresting for a PCA since very low information is added by these highly correlated dimensions. A dimensionality reduction method which identifies the most important and representative dimensions would thus discard these dimensions. Variables which are highly correlated to most other variables would be one of the first to be discarded.

If we interpret users as variables (i.e. the dimensions of the data are the users, and the observations are the items), we have data where a number of dimensions are very similar. Thus dimensionality reduction would discard these dimensions since their covariance will be low. A closer look at our data shows us that the covariance between shilling profiles is much lower than normal users. This low covariance is observed not only in between shilling users, but also between shilling users and normal users. Covariance between normal users is observed to be much higher. PCA of this dataset will compute principal components which are oriented more towards real users who exhibit the maximum variance of the data. We therefore need to select those users (which are viewed as dimensions, by transposing the data) which show the least covariance with all other users. This amounts to selecting some variables from the original data using PCA, which is known in literature as *Variable selection using PCA*.

Algorithm 2 outlines our proposed approach for variable selection: after substituting votes with *z-scores*(see Eq. (6)), we transpose the data to cast users as variables and calculate the covariance matrix. Note that covariance of *z*-score leads to correlation coefficients; however these are different than the values typically in collaborative filtering, since these correlation values are measured over all the items. Eigendecomposition of the covariance matrix yields the eigenvectors which corresponds to the principal components (PCs). The eigenvector corresponding to the largest eigenvalue is known as the first principal component, and so on. For every variable, each PC contains a coefficient. We choose those users which have the smallest coefficient in the first m PCs.

[6]Due to lack of space, PCA is not explained in detail. The interested reader can see (Jolliffe 2002) for more details

Algorithm 2 PCASelectUsers $(\mathbf{D}, \text{Cutoff parameter} r)$

$\mathbf{D} \leftarrow$ z-scores$(\mathbf{D})$
$COV \leftarrow \mathbf{D}\mathbf{D}^T$ {Covariance of $\mathbf{D}^T$}
$\mathbf{U}\lambda\mathbf{U}^T =$ Eigenvalue-Decomposition(COV)
$PCA_1 \leftarrow \mathbf{U}(:,1)$ {}1st Eigenvector of COV
$PCA_2 \leftarrow \mathbf{U}(:,2)$ {}2nd Eigenvector of COV
for all columnid $user$ in $\mathbf{D}$ **do**
  $Distance(user) \leftarrow PCA_1(user)^2 + PCA_2(user)^2$
end for
Sort $Distance$

Output: Return r users with smallest $Distance$ values

EXPERIMENTAL SETUP AND RESULTS

To evaluate the performance of our proposed algorithms algorithm, we use the MovieLens dataset which consists of 100,034 votes by 944 users over 1682 movies and has been used previously for evaluating shilling detection. To this data, shilling profiles are added which all target the same item which is selected at random. shilling profiles are generated using the well studied models of Average, Random and Bandwagon attacks. We use the generative models explained in (Burke *et al.* 2006) which add gaussian noise to item or overall averages. Experimental results have been found to hold on larger datasets like EachMovie: we present results on the 100k ML dataset to be directly comparable with other reported results.

Experimental results[7] for PLSA based detection show that shilling profiles are indeed clustered together: in most experiments, all shillers end up in one cluster. Moreover, using the *closeness* measure also works well in most cases. For medium and large sized attacks, (see Fig. 4) more than 70% attackers are correctly identified. However the precision is low as many normal users are also misclassified. We find 20 communities to be ideal, which makes each cluster between 2-10%. For small attacks, PLSA based detection is ineffective. For very small filler sizes (% of rated items) and attack sizes (no. of profiles inserted), low recall and precision are observed. Also in 20% of the cases (2 out of 10 trials), the wrong cluster is selected, leading to maximum 80% recall and precision on an average. This experiment also explains the robustness of PLSA against shilling: the effect of shilling is large only in the cluster where most shilling users are. For all other clusters, the prediction shift is much lesser as the effect is weighted by the cluster weight of shilling users, which is usually a small fraction. However, for large attack sizes, we note that large clusters are formed with a majority of the users in the same cluster as shilling users, hence explaining the large prediction shift reported in (Mobasher, Burke, & Sandvig 2006).

Variable selection based on PCA does much better(see Figure 4): more than 90% precision and almost 100% recall is consistantly observed for the same shilling profiles used for PLSA based detection. PCA does very well in identifiying profiles which are very similar to most other profiles, since they add very little information. With 10% attack pro-

[7]For lack of space, we report a subset of our experiments.

Figure 3: Detection Recall and Precision for Average Attacks of varying filler sizes using PLSA based detection (*Algorithm 1*). 10% shilling profiles were added to 944 normal users.

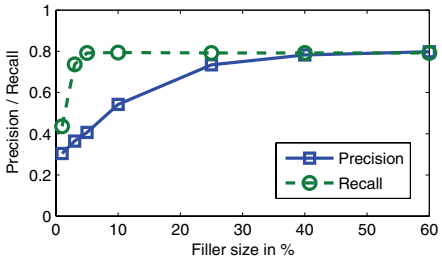
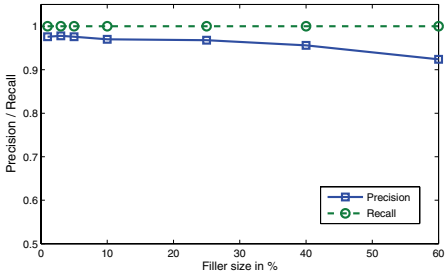


Figure 4: Detection Recall and precision for Average Attacks of varying filler sizes using PCA-Variable Selection based detection (*Algorithm 2*). 10% shilling profiles were added to 944 normal users.



files, the top 10% eliminated users are shillers with more than 90% accuracy. Similar numbers are observed for a variety of attacks (*random, average, bandwagon*) in variety of attack sizes, and filler sizes. Further accuracy is gained by considering the first 3 Principal components, and sorting the variables *together*.

PCA based user selection performs better than PLSA based detection in the experiments conducted. A comparison with other reported algorithms shows a clear advantage for PCA-based selection. The (Burke *et al.* 2006) approach is based on a large set of features which exploit the characteristic properties of shiller. However, the detection procedure results in a large number of false positives. Table 2. compares the reported performance of PCA vs the *Burke et al.* approach. However, drawbacks of both approaches do exist: our PLSA based approach identifies the correct cluster only 4 out of 5 times, and has low recall and precision against smaller attacks. When 50 shilling profiles were injected, the recall and precision were both around 20% lower than the reported numbers for detecting 100 profiles. Adding 1% profiles only results in zero recall. Clearly, smaller attacks are harder to detect. PCA based detection is more stable against attack size, but does not perform as well when attack profiles are not highly correlated. In this case, the attacks also have limited effect since the impact of a shilling profile is high only when it is similar to a number of users. Therefore, low-quality shilling data may not be very well detected by this method.

Table 2: Detection precision for Push Attacks of size 1% at different filler sizes compared with other algorithms. Numbers for the *Burke et al* algorithm have been reported from (Burke *et al.* 2006)

Filler	*Average Attack*		*Random Attack*	
	Burke et al	**PCA**	Burke et al	**PCA**
1%	22	90	26	96
5%	23	92	28	100
10%	29	96	29	94
20%	32	90	33	96
40%	39	80	40	98
60%	42	68	47	92

CONCLUSION

This paper discusses novel interpretations of known unsupervised algorithms for detecting highly correlated groups of shilling users which are hard to detect using other statistical measures. The algorithms presented here are fast and highly accurate and have not been discussed in literature to the best of our knowledge. Due to their unsupervised nature, these algorithms can scale very well, and can be used as a preprocessing step for recommendation algorithms. This paper also provides an explanation for the phenomenal robustness of PLSA under shilling attacks reported recently at AAAI-06(Mobasher, Burke, & Sandvig 2006). In particular, the PCA based variable selection stands out as an extremely effective detection method. More experiments, not mentioned here due to space, show excellent performance under other attack models as well (bandwagon, random attacks, obfuscated attacks). Future works includes developing principled collaborative filtering algorithms which are robust and modify their behavior in face of shilling attacks.

References

Burke, R.; Mobasher, B.; Williams, C.; and Bhaumik, R. 2006. Analysis and detection of segment-focused attacks against collaborative recommendation. In *Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD'06)*.

Hofmann, T. 2004. Latent semantic models for collaborative filtering. *ACM Trans. Inf. Syst.* 22(1):89–115.

Jolliffe, I. T. 2002. *Principal Component Analysis (2nd Edition)*. Springer.

Lam, S. K., and Riedl, J. 2004. Shilling recommender systems for fun and profit. In *WWW '04: Proceedings of the 13th international conference on World Wide Web*, 393–402. New York, NY, USA: ACM Press.

Mobasher, B.; Burke, R. D.; and Sandvig, J. J. 2006. Model-based collaborative filtering as a defense against profile injection attacks. In *AAAI*.

O'Mahony, M.; Hurley, N.; Kushmerick, N.; and Silvestre, G. 2004. Collaborative recommendation: A robustness analysis. *ACM Trans. Inter. Tech.* 4(4):344–377.