Statistical Attack Detection

Neil Hurley
School of Computer Science and Informatics,
University College Dublin, Ireland
neil.hurley@ucd.ie

Zunping Cheng
School of Computer Science and Informatics,
University College Dublin, Ireland
zunping.cheng@ucd.ie

Mi Zhang
School of Computer Science and Informatics,
University College Dublin, Ireland
mi.zhang@ucd.ie

ABSTRACT

It has been shown in recent years that effective profile injection or shilling attacks can be mounted on standard recommendation algorithms. These attacks consist of the insertion of bogus user profiles into the system database in order to manipulate the recommendation output, for example to promote or demote the predicted ratings for a particular product. A number of attack models have been proposed and some detection strategies to identify these attacks have been empirically evaluated. In this paper we show that the standard attack models can be readily detected using statistical detection techniques. We argue that insufficient consideration of the effectiveness of attacks under a constraint of statistical invariance has been taken in past research. In fact, it is possible to create effective attacks that are undetectable using the detection strategies proposed to date, including the PCA-based clustering strategy which has shown excellent performance against standard attacks. Nevertheless, these more advanced attacks can also be detected with careful design of a statistical detector. The question posed for future research is whether attack models that produce effective attack profiles that are statistically identical to genuine profiles are really possible.

Categories and Subject Descriptors

H.3.3 [Information Storage and Retrieval]: Information Search and Retrieval—collaborative filtering, robustness; G.3 [Probability and Statistics]: Robust regression

General Terms
Performance, Security

1. INTRODUCTION

The possibility of designing user rating profiles to deliberately and maliciously manipulate the recommendation output of a collaborative filtering system was first raised in [14]. One scenario proposed was that an author, motivated to increase recommendations of his book, might create a set of false profiles that rate the book highly, in an effort to artificially promote the ratings given by the system to genuine users. Since then, these attacks have been dubbed as shilling attacks [8] or profile injection attacks [1]. Several attack models have been proposed and the performance of these attacks in terms of influencing the system predictions has been evaluated for a number of memory-based and model-based collaborative filtering algorithms. It is generally accepted that the best ‘general purpose’ attack strategy is the so-called ‘Average attack’ proposed initially in [8], although it requires some knowledge of the rating statistics of genuine users. To counteract attacks, researchers have investigated the application of classification techniques to user profiles, in order to identify attack profiles and filter them from the dataset. Several attack detection strategies have been proposed. Of these, the principal components analysis (PCA) based detection strategy, recently proposed in [10], yields the best detection performance, obtaining over 90% precision in the detection of Average and other attack models when evaluated on the Movielens dataset.

In this paper, we review attack models in the context of their detectability. We propose to use Neyman-Pearson statistical detection to identify attack profiles and show how to statistically model the standard attacks that have been proposed to date. Our analysis shows that the success of the PCA-based detector is largely due to the unrealistic manner in which items are rated in attack profiles of standard attacks. With this realisation, we show how it is possible to create an effective attack which is undetectable by the PCA detector with a simple modification of the Average attack. To address the detection of such obfuscated attacks, we model an attacked dataset as a multivariate Gaussian mixture model and design supervised and unsupervised Neyman-Pearson detectors based on this model. These detectors significantly out-perform the PCA detector on obfuscated attacks.

The significance of this analysis is as follows. Firstly, it demonstrates that the attack models proposed to date are insufficient and have failed to properly address the question of designing undetectable attacks. Even simple modifications of the existing models are sufficient to fool the best detectors proposed to date. Secondly, we have shown that Neyman-Pearson statistical detection is a powerful tool that can be successfully applied to the detection of a wider range of obfuscated attacks. In the end, we argue that it is important to consider robustness of recommender systems, taking into account the level of knowledge available to the attacker, both
of genuine rating statistics and of the recommendation algorithm and how the attacker can exploit that knowledge in the development of effective attacks. While memory-based algorithms have shown weak robustness to standard attack models, we believe that carefully designed attacks can also exploit vulnerabilities of other recommendation algorithms. It is therefore important to design good detection algorithms for more sophisticated attacks than have been considered to date.

The paper is organised as follows. In Section 2, the attack detection problem is defined and the state-of-the-art in attack detection is reviewed in Section 3. In Section 4, a simple statistical model of standard attacks is developed and compared with PCA-based detection. Obfuscation is considered in Section 5 and a Gaussian mixture statistical model is developed for detection. Final conclusions are drawn in Section 6.

Notation

Column vectors are represented by bold lowercase letters such as \( \mathbf{x} \). The transpose of a column vector to a row vector is written as \( \mathbf{x}^T \). Matrices are represented in roman text, capital letters such as \( \mathbf{A} \).

We consider a ratings database consisting of a set of user profiles, that is ratings that a user gave for items in the system. Write a user profile as the vector \( \mathbf{y}^u = (y_1, \ldots, y_m) \) where \( m \) is the total number of items in the system. The vector components \( y_i \in \phi \cup \psi \) where \( \psi = \{ \psi_{\min}, \ldots, \psi_{\max} \} \), is a set of discrete rating values, assumed to be integers and \( \phi \) is used to represent that item \( i \) has not been rated. Rather than use the discrete scale, in the Gaussian model developed in Section 5, we consider \( y_i \in [-\infty, \infty] \) i.e. that it takes any value on a continuous scale.

2. ATTACK DETECTION

Figure 1 summarises the steps involved in attack detection. This is a binary classification problem, with two possible outcomes for each profile, namely, Genuineness, meaning that the classifier has determined that the profile is that of a genuine system user or Attack, meaning that the classifier has determined that this is an instance of an attack profile. One approach to the detection problem, followed by work such as [5, 1], has been to view it as a problem of determining independently for each profile in the dataset, whether or not it is an attack profile. This is the ‘single profile’ input shown in Figure 1. The input is a single rating vector \( \mathbf{y}_u \), for some user \( u \) from the dataset. Before processing by the classifier, a feature extraction step may extract a set of features, \( \mathbf{f}_u = (f_1, \ldots, f_k) \) from the raw rating vector \( \mathbf{y}_u \). The classifier takes \( \mathbf{f}_u \) as input and outputs, “Attack” or “Genuine”. If the classifier is a supervised classifier, then a training phase makes use of an annotated dataset of profiles, i.e. a set of profiles labelled as Genuine or Attack, in order to learn the classifier parameters.

Because most attack scenarios consist of groups of profiles working in concert to promote or demote a particular item, work such as [9, 13] has suggested that there is benefit to considering groups of profiles together when making the classification. This is represented by the ‘Group of Profiles’ input, in which the classifier considers an entire group of profiles, possibly after some feature extraction, and outputs a label for each profile in the group. Note that not all steps may take place in any particular scenario. For instance, there may be no feature extraction, in which case, \( \mathbf{f} = \mathbf{y} \) and if unsupervised classifiers are used, then there is no need for a training phase.

2.1 Neyman-Pearson Statistical Detection Theory

In this paper, we will treat attack detection as an instance of a statistical detection problem, a problem of making a decision among a number of possible statistical models that describe some set of observations. In fact it is a binary hypothesis testing problem, in which the observations may have come from two possible choices or hypotheses, usually denoted by \( H_0 \) and \( H_1 \). In our context, the observations are the profiles contained in the system database. Given an observation (i.e. a profile), the task is to make a decision between \( H_0 \), that the profile is a genuine profile or \( H_1 \) that the profile is an attack profile. Many decision rules are possible, particularly if the hypotheses are not viewed as equal and different costs are assigned to different types of error. We focus on Neyman-Pearson detectors that do not require a priori probabilities of the hypotheses.

Let \( \mathbf{y} \in \psi \) be an observation i.e. a user profile, over the space \( \psi \) of all possible profiles. A test is formed based on \( f_{\psi \mid H_0}(\mathbf{y}) \) and \( f_{\psi \mid H_1}(\mathbf{y}) \), the probability distribution functions (pdf) of \( \mathbf{y} \) under the assumption that \( H_1 \) or \( H_0 \) is true, respectively. We can partition \( \psi \) into two regions \( \mathcal{R}_0 \) and \( \mathcal{R}_1 \) such that \( H_0 \) is accepted if \( \mathbf{y} \in \mathcal{R}_0 \) and \( H_0 \) is rejected if \( \mathbf{y} \in \mathcal{R}_1 \), and therefore write the decision function as:

\[
\psi(\mathbf{y}) = \begin{cases} 1 & \text{if } \mathbf{y} \in \mathcal{R}_0 \\ 0 & \text{if } \mathbf{y} \in \mathcal{R}_1 \end{cases}
\]

The performance of a test can be determined by considering the different types of error that can occur. A false alarm refers to \( H_0 \) being rejected when true and a miss refers to \( H_1 \) being rejected when true. Write \( p_f \) as the probability of false alarm, \( p_m \) as the probability of a miss, \( p_d = 1 - p_m \) as the probability of good detection. The Neyman-Pearson (N-P) criterion sets a bound \( \alpha \) on \( p_m \) and finds the decision function that maximises \( p_d \) under the constraint that
\( p_f \leq \alpha \). The optimal test in this sense, is the N-P test which has the general form

\[
\psi^*(y) = \begin{cases} 
1 & \text{if } l(y) > \eta \\
q & \text{if } l(y) = \eta \\
0 & \text{if } l(y) < \eta
\end{cases}
\]

(2)

\( \eta \geq 0 \) is a threshold and \( q \) is the probability with which \( H_1 \) is chosen when \( l(y) = \eta \) and \( l(y) \) is the likelihood ratio defined as

\[
l(y) = \frac{f_{Y|H_1}(y)}{f_{Y|H_0}(y)}.
\]

(3)

Thus, if we have a good model of genuine profiles, represented as the pdf \( f_{Y|H_0}(y) \) and a good model of attack profiles, represented as the pdf \( f_{Y|H_0}(y) \), then (3) provides the optimal test to distinguish between these alternatives. It is straightforward to compute the pdf of the precisely defined attack models that have been proposed in the literature and we will do this in Section 4. A good model of genuine profiles is more difficult and will be discussed afterwards.

### 2.2 ROC Analysis

The threshold \( \eta \) of the N-P test can be computed analytically for simple distributions and may be selected through an empirical evaluation in practice. The receiver operating characteristic (ROC) curve plots \( p_d \) against \( p_f \) for different threshold values and allows an appropriate trade-off between the detection performance and false alarm error to be chosen. In previous studies such as [10], precision and recall have been used to evaluate attack detection performance, with just a single figure reported for each attack configuration. However, this does not consider the detection performance over different detection sensitivities (which, in the case of the PCA clustering algorithm in [10] corresponds to different attack cluster sizes). Hence, we prefer to present ROC results. To make the correspondence to previous studies, note that \( p_d \) is equivalent to precision.

### 3. RELATED WORK

#### 3.1 Single Profile Detection

A profile injection attack whose goal is to promote the ratings given to a target item is often referred to as a push attack. As a targeted pushed item in an attack profile will have unusually high ratings and other ratings may be chosen to support the influence of the profile towards high predictions for the target, distinctive characteristics are likely to exist in attack profiles and may be manifested in many ways [1]. An unsupervised individual profile detection algorithm is described in [5]. Detection is based on certain common attributes of attack profiles, for example that there is a higher than usual rating deviation from mean in such profiles and that such profiles are likely to have a higher than usual similarity to their closest neighbours. In [3] profile attributes based on those proposed in [5] and others along similar lines were developed into features for inclusion in a feature vector input to a supervised classifier. The authors evaluated three supervised classifiers: kNN, C4.5, and SVM. SVM and C4.5 were found to have near perfect performance on identifying attack profiles correctly, but on the other hand, they also misclassify more genuine profiles than kNN. SVM had the best performance considering all error types.

#### 3.2 Group Profile Detection

##### 3.2.1 Neighbourhood Filtering

In [13] an unsupervised group detection and filtering scheme is presented. Rather than filtering profiles from the dataset in a preprocessing step, in this method, filtering is applied to the profiles in the active user’s neighbourhood during prediction for a particular item. This approach has the advantage of identifying just those attack profiles that are targeting the active item. The strategy is based on an algorithm proposed in [6] in the context of reputation reporting systems that aims to provide a reputation estimate for buyers and sellers engaged in on-line marketplaces that is robust to malicious agents who attempt to fraudulently enhance their own reputations. The approach involves the clustering of neighbourhoods into two clusters based on the cluster means and standard deviations.

##### 3.2.2 Detecting attacks using Profile Clustering

In [10] the observation that attacks consist of multiple profiles which are highly correlated with each other, as well as having high similarity with a large number of genuine profiles motivates the development of a clustering approach to attack detection, using Probabilistic Latent Semantic Analysis (PLSA) and Principal Component Analysis (PCA). The second of these strategies provides the better performance and is based on a PCA of the covariance matrix of the user profiles. Essentially this strategy attempts to identify a cluster where the sum of the pair-wise covariances between profiles in the cluster is minimised (or maximised\(^3\)). PCA has been widely used as a dimension reduction strategy for high-dimensional data. Identifying profiles with dimensions, the method is explained intuitively in [10] as a method of identifying those highly-correlated dimensions (i.e. profiles) that would safely be removed by PCA. Alternatively, a cluster \( C \) can be defined by an indicator vector \( x \) such that \( x_i = 1 \) if user \( u_i \in C \) and \( x_i = 0 \) otherwise. With \( S \) defined as the covariance matrix, the sum of the pair-wise covariances of all profiles in \( C \), may be written as the quadratic form

\[
x^T S x = \sum_{i \in C, j \in C} S(i, j).
\]

Moreover, for the normalised eigenvectors \( e_i \) of \( S \), associated with eigenvector \( \lambda_i \) such that \( \lambda_1 \leq \cdots \leq \lambda_m \), the quadratic form evaluates as

\[
x^T S x = \sum_{i=1}^{m} (x^T e_i)^2 (e_i^T S e_i) = \sum_{i=1}^{m} (x^T e_i)^2 \lambda_i.
\]

With this observation, the method described in [10] may be understood as a method that seeks the binary vector \( y \) that minimises the quadratic form by choosing \( y \) so that it has strong correlation with those 3 – 5 eigenvectors corresponding to the smallest eigenvalues.

The UnRAP algorithm [2] also uses clustering to distinguish attack profiles. It bases clustering on a measure called the \( H_e \) score which has proved successful in identifying highly correlated biclusters in gene expression data. In the context of attack detection, the \( H_e \) score measures for each user, a sum of the squared deviations of its ratings from the user mean, item mean and overall mean ratings. In general, the authors report that this method performs well particularly

\(^3\)We will discuss this point more carefully in a later section.
4. STANDARD ATTACK MODEL DETECTION

We adopt the terminology of [11], in discussing standard attack models. In characterising attacks, the ratings in an attack profile are partitioned into three sets. The item $i_t$ represents a target item that is the focus of the attack. The purpose of the attack profile is to manipulate the predicted ratings for the target item. We will focus on push attacks, where the goal is to promote the predicted ratings to the maximum value. A set $S$ of selected items may also be rated and given specific ratings in order to support the attack. Finally, a set $F$ of filler items is chosen and given random ratings. In [11], filler items are not described as being specifically constructed to support the attack. However, in practice, the selection of filler items and the ratings assigned to them has a major impact both on the success of the attack, from the attackers point of view, and on the detectability of the attack, from the system manager’s point of view. Hence, the distinction between $S$ and $F$ is not particularly useful.

4.1 The Random and Average Attacks

Two general attack models were originally proposed in [8] and named the Random and Average attacks. The target $i_t$ is assigned the maximum rating (assuming a push attack). There are no selected items, $S$. The ratings of filler items are chosen independently from a Gaussian distribution. For the Random attack, it is assumed that the ratings are drawn from the Gaussian distribution $N(\mu, \sigma^2)$ where $\mu$ and $\sigma$ represent the overall mean and standard deviation of all ratings across the entire database. For the Average attack, a rating $r_i$ for item $i$, is drawn from $N(\mu_i, \sigma_i^2)$ where $\mu_i$ and $\sigma_i$ represent the mean and standard deviation of all ratings given for item $i$.

To build a statistical model of these attacks, we assume all ratings in the profile have been generated using the filler item rule and ignore the fact that one target item has been specially rated. Note that, in this attack model, whether or not an item is chosen to be a filler is independent of the rating chosen for the item. Define $\theta_i$ as an indicator variable such that $\theta_i = 1$ if $y_i = \phi$ and $\theta_i = 0$ otherwise. The pdf, $f_{\theta_i, \mu_i}(y)$, of $y$ under hypothesis $H_1$ that it is an attack profile is given by

$$\prod_{i=1}^{m} \Pr[Y_i = y_i] = \prod_{i=1}^{m} \Pr[Y_i = \phi]^{\theta_i} (\Pr[Y_i = y_i|Y_i \neq \phi|\Pr[Y_i \neq \phi])^{1-\theta_i}$$

Furthermore, assuming that the continuous values generated by the Gaussian distribution are rounded to the nearest integer, so that the probability of obtaining the rating value $y_i$ is the probability that a Gaussian random number lies in the range $[y_i - \frac{1}{2}, y_i + \frac{1}{2}]$, we can write

$$\Pr[Y_i = y_i|Y_i \neq \phi] = \mathcal{Q}\left(\frac{y_i - \frac{1}{2} - \mu_i}{\sigma_i}\right) - \mathcal{Q}\left(\frac{y_i + \frac{1}{2} - \mu_i}{\sigma_i}\right)$$

where $\mathcal{Q}(.)$ is the Gaussian $Q$-function.

It remains to determine $\Pr[Y_i \neq \phi]$. Random and Average attacks are usually evaluated as a function of $|F|$, the filler size. Filler items are chosen with equal probability from the item set. As we will see later, this filler selection strategy is key to successful detection of the attacks, since it does not reflect how genuine users choose items to rate. If the filler size is $t$, we take $\Pr[Y_i \neq \phi] = 1/t$. M.

4.2 Genuine Profile Model

Examining the Random and Average attacks, it is clear that, while an effort has been made to ensure that the rating values of the selected filler items are similar to genuine rating values, no effort has been made to ensure that these items selected as fillers are similar to the selections made by genuine users. In reality, every item does not have an equal chance of being rated; some items are much more popular than others. A model that is sufficient to distinguish genuine profiles from attack profiles can use (4) for the pdf $f_{\phi, \mu_i}(y)$, of genuine profiles, but with $\Pr[Y_i \neq \phi]$ set to model the rating behaviour of genuine users. Note that this model of genuine profiles suggests that the rating values given by a user for one item are independent of the values given for another item. If this were an accurate model of real rating behaviour, then collaborative filtering algorithms could not be successful. However, all we require for detection is a model that distinguishes genuine from attack profiles. For each item $i$, we set $\Pr[Y_i \neq \phi] = p_i$, where $p_i$, the probability that an item is rated, can be estimated from a training set of genuine profiles, as the fraction of profiles that rated the item.

4.3 Bandwagon Attacks

In [11] the unrealistic filler selection strategies of the Random and Average attacks is recognised. The Bandwagon attack is constructed on the basis that some items are more popular than others. The purpose is to make more effective attacks rather than to make attack profiles more like genuine profiles. In the Bandwagon attack, $S$ is non-empty and contains some number $s$ of popular items. Items in $S$ are given the same rating value as the target item, in order to support the attack. The rest of the profile is filled with filler items in the same manner as an Average attack. In evaluations, $s$ has been chosen as small relative to the overall filler size. In fact, in [11], $s = 1$ is chosen. With such small $s$, the statistical differences been Bandwagon and Average attacks are insignificant and the Average attack model can be used. It is possible to model the Bandwagon attack more precisely, in a straight-forward manner along the lines of the Average attack model above. We will not describe the model in detail here, but note that, Bandwagon attack profiles are in general more detectable than Average attacks when accurately modelled.

4.4 Evaluation

The ROC curve for statistical detection using the N-P test and the pdf’s developed in the previous sections is shown in

$$Q(x) = 1/(2\sqrt{\pi}) \int_x^\infty e^{-t^2} dt.$$
4.5 Profile correlation and PCA detection

The PCA detection strategy is based on calculating the principal components of the covariance matrix of the ratings dataset, converted to z-scores. In [10], it is striking that, while an argument is made to suggest that Average attack profiles are highly similar to each other and to other genuine profiles, the method actually identifies an attack cluster as a set of profiles with low pairwise covariance. If the similarity metric is based on covariance, then the method is actually identifying a set of maximally dissimilar profiles. It transpires that this discrepancy and the explanation for the success of the method, may be understood by considering how missing values are handled. The z-score for a profile $y$ is obtained by subtracting the profile mean $\mu_y$ and dividing by the standard deviation $\sigma_y$:

$$z_i = \frac{y_i - \mu_y}{\sigma_y}.$$

Given that a typical profile contains many unrated items, the z-score depends strongly on how we deal with missing values. One strategy is to ignore them and calculate $\mu_y$ and $\sigma_y$ based on the rated items only. Call the resulting matrix of z-score ratings $Z_0$. A second strategy is to treat missing values as zeros and compute $\mu_y$ and $\sigma_y$ across the entire profile. In the case of the 5-point rating scale of the Movielens dataset, with minimum rating 1, this is equivalent to considering a missing rating to be an extremely low evaluation of the corresponding item. Call the resulting matrix $Z_1$. In fact, as observed in Figure 2, using $Z_0$ is extremely unsuccessful. In fact, the success of the PCA Z1 strategy is largely explained by the fact that the covariance is dominated by the missing value distribution. It is also noteworthy that the standard Pearson correlation formula that is generally applied in the user-based kNN algorithm, computes the correlation on the overlap ratings only. Profiles can therefore be highly correlated according to Pearson similarity, while having low $Z_1$ covariance. In fact, small overlap increases the probability of obtaining extreme values of the Pearson correlation. We exploited this fact in the original shilling attack, reported in [14].

Average attack profiles, whose filler items are selected with equal probability, are likely to have smaller pair-wise overlaps than genuine users, who are generally more likely to rate certain items over others. In fact, for the Movielens dataset, the mean overlap size for Average attack profiles is 9.8, when the sparsity is 5%. The mean overlap size for genuine profiles is 23.7, although the sparsity is the same. It follows that the success of the standard Average attack can be largely explained by its unrealistic filler selection strategy. However, as we have seen, this is also its weakness from the point-of-view of detection.

5. OBFUSCATED ATTACKS

The possibility of obfuscating attacks has been considered previously in [16] where three different obfuscation strategies were proposed. These strategies represent ad hoc modifications of the standard attack models that an attacker might employ in order to avoid detection. No guiding principle is employed in the design of these obfuscations. An effective obfuscation strategy must try to minimise the statistical differences between genuine and attack profiles. We have seen that an attack profile that is undetectable by the PCA detector must reduce the differences in filler selection that are key to the success of this detector.

5.1 The AoP Attack

In fact, our analysis has shown that an attacker should choose filler items according to their overall popularity among the genuine user base. A simple and effective strategy to obfuscate the Average attack is to choose filler items with equal
probability from the top $x\%$ of most popular items, rather than from the entire catalogue of items, where $x$ is chosen to ensure that the profiles are undetectable by the PCA detector. However, choosing items from a subset of the catalogue increases the pair-wise overlap of attack profiles and, by our comments at the end of the previous section, renders the attacks less effective. Hence, the attacker must compromise between detectability and attack performance. We dub this attack strategy the AoP (Average over Popular items) attack and will refer to an “AoP $x\%$ attack” when the items are drawn from the top $x\%$ most popular items.

The AoP attack is evaluated on the MovieLens dataset by applying a push attack in turn to 100 randomly chosen target items. The attack size (number of attack profiles relative to the overall database size) is set to $3\%$. The standard user-based kNN algorithm, with $k = 20$ and using the Pearson similarity metric is used to make predictions for the target item. The prediction shift between predictions made pre- and post-attack for all users who rated the target item is evaluated. In Figure 3(a) the average prediction shift that results is plotted against filler size, for different values of $x$. In Figure 3(b), the performance is given as the percentage of hits obtained on the attack item, where a hit is defined as a rating of 4 or more. We see that as attacks are constructed from smaller subsets of popular items, the attacks become less effective. However, even when choosing from the top 20% of most popular items, a significant push is obtained, which can be best seen from the uplift in hits obtained over the pre-attack value.

A ROC curve of detection of the AoP attack using the PCA detector is shown in Figure 4. A sharp disimprovement in detection quality is observed over the standard Average attack. For example, when filler items are confined to the top 60% of most popular items, to obtain over 90% precision comes at a cost of nearly 20% false alarm probability and detection is hopeless when 20% of the most popular items are used.

5.2 AoP Attack Detection

It is possible to design an N-P detector that successfully detects the AoP attack. However, a more model of genuine ratings, $f_{Y|H_0}(y)$, is required, that properly captures the correlations in rating values, as well as the filler selection distribution. In fact, if all ratings are available, a good model for the distribution of $y$ is a multivariate Gaussian distribution $\mathcal{N}(\mu_0, \Sigma_0)$, where $\mu_0$ represents the vector of mean ratings for each item in the genuine profile and $\Sigma_0$ is the covariance matrix of genuine profiles. This model has been previously successfully applied to model rating behaviour in collaborative filtering. Similarly, we make the assumption that $f_{Y|H_1}(y)$, the distribution of attack profiles is also multivariate Gaussian, with mean $\mu_1$ and correlation $\Sigma_1$. The entire dataset, of genuine and attack profiles, can therefore be modelled as a Gaussian mixture model. As the rating values in an AoP attack are chosen independently, $\Sigma_1$ is diagonal, which should not be the case in general for $\Sigma_0$.

To deal properly with missing values, we train a factor model using the expectation maximisation (EM) algorithm. Thus, following [4], we assume that the rating matrix can be represented by the linear model

$$Y^T = AX^T + N,$$

where $X$ is a $n \times k$ matrix with $x_{u,j}$ representing the extent to which user $u$ likes latent category $j$, $A$ is a $m \times k$ matrix such that $a_{i,j}$ represents the extent to which item $i$ belongs in latent category $j$, $N$ represents independent noise in the ratings which is assumed to have variance $\psi$ and $k$ is the number of latent categories. We apply the EM algorithm presented in [4], to learn $A, X$ and $\psi$ from a training set of profiles (assumed to be genuine).

Given $A$ and a user profile $y$, it follows that $w = A^Ty$ is a $k$-dimensional multivariate Gaussian distribution. The N-P test for distinguishing between $H_0$ and $H_1$ based on $w$ reduces to

$$(w - \mu_{w_0})^T \Sigma_{w_0}^{-1} (w - \mu_{w_0}) - (w - \mu_{w_1})^T \Sigma_{w_1}^{-1} (w - \mu_{w_1}) \leq \eta$$

where $\mu_{w_0}$, the mean of $w$ under $H_0$, is $\mu_{w_0}$, is $A^T\mu_0$ and the covariance is $A^T\Sigma_0A$ and similarly for $H_1$.

An alternative, less computationally intensive strategy, since it does not require factor model training, is to choose the projection matrix $A$ based on item clustering. Given the pair-wise item-item similarities, it is possible to cluster the items into clusters of similar items. The components of $w$ are then chosen as the sum of the ratings for all items in each cluster. Many clustering algorithms are possible. We compute a partitioning of the similarity matrix using the *metis* graph partitioning software [7].

![Figure 3: Effectiveness of Popular Average Attack](image-url)
5.3 Supervised Detection

To apply the N-P detector (6), we need to estimate the mean and covariance of \( w \) under the two hypotheses. For a given attack model, it is possible to generate attack clusters and estimate the required parameters for a training set of genuine clusters and a training set of attack clusters. Of course one disadvantage of such supervised detection is that the actual attack may not be identical to the attack used for parameter estimation. We evaluate supervised detection on a dataset containing genuine profiles from Movielens and AoP 20% attack profiles, with 5% filler size. To compute the factor matrix, \( k = 25 \) is chosen. The resulting ROC curves are shown in Figure 5(a). Be careful to note that the different curves correspond to simulations in which different attack parameter estimations were derived by training on the corresponding attack model. In all cases, the actual attack was an AoP 20% attack.

When the detector is trained on AoP 20% attacks, it gives close-to perfect performance. Performance reduces if the training set does not reflect the real attack as may be observed from ROC curves that result from training on AoP 30%, 40% and 60% attacks.

The results of using the partitioning strategy with \( k = 25 \) are shown in Figure 5(b). While they are not as good as the factor model when the training set consists of the profiles from the actual attack, this strategy performs a little better when the actual attack does not match the attack on which the model was trained.

5.4 Unsupervised Detection

Unsupervised Gaussian mixture model clustering can be applied to partition the dataset into two clusters and learn the required parameters of the mixture model: \( \mu_0, \mu_1, \Sigma_0, \Sigma_1 \), and \( \alpha_0, \alpha_1 \), where \( \alpha_i \) is the probability that a profile belongs in cluster \( i \). The EM algorithm is used to train the model and the N-P test is applied using the resulting parameter estimates. The results for AoP 20%, 40% and 60% attacks are shown in Figure 6. As expected, the performance is significantly worse than supervised detection, but, comparing with Figure 4, significantly better than PCA detection. Nevertheless, a cost of misclassifying 10% of genuine profiles is required to obtain 80% or more precision on the AoP 20% attack, for instance, or only 50% precision on the AoP 60% attack.

5.5 Other Obfuscation Strategies

The N-P detectors show that an attacker must consider rating values, as well as filler distribution in designing an obfuscation. The AoP strategy is just one simple means of obfuscating a profile and is presented to illustrate that the standard attack models that do not consider detectability, along with attack effectiveness, are not particularly useful. In practice, it is necessary to consider the knowledge available to an attacker and how much an attacker can learn about the system. The Average attack assumes that an attacker can discover or learn the mean and standard deviation of genuine ratings for each item in the dataset. Given a sufficient number of genuine profiles, it is also possible to envisage that an attacker could estimate the covariance of genuine profiles and to use this estimate to generate multivariate Gaussian attack profiles, to reduce the detection performance of the N-P detector presented above.

To illustrate that such obfuscated attacks can still be effective, we consider an idealised 'perfectly obfuscated' attack on the user-based kNN algorithm, in which attacks consist of genuine user profiles, whose rating for the target item has been set to maximum. A 3% attack is performed on 100 randomly selected items from the Movielens dataset, as before. The resulting mean prediction shift is 0.42 and the uplift in hits is 9.5%. This confirms that the user-based algorithm has an inherent vulnerability to profile-injection attack. Contrary to work such as [12], it is not safe to assume that other algorithms do not exhibit such vulnerability. The attacker...
can tailor attacks to the recommendation algorithm of the system under attack. We have shown in other work how it is possible to design attack profiles tailored to model-based recommendation algorithms. A stealthy attacker will consider attack effectiveness together with attack detectability, considering all available knowledge of the recommendation algorithm and the database profile statistics. The system designer, therefore must design detection strategies that are effective against such sophisticated attacks. In the end, attack detection must be based on the statistical differences between genuine and attack profiles and N-P detectors provide an optimal way to detect such differences, once good statistical models are obtained.

6. CONCLUSIONS

In this paper, we have discussed profile injection attacks and developed statistical models for their detection. It is clear that attack and detection can result in an arms race scenario between the system manager and attacker. It is important to know the limits of this arms race and work such as [15] has provided an interesting theoretical basis. Nevertheless, our work shows that the optimal attacks are not random, but exploit explicit vulnerabilities in the recommendation algorithms. Further work to understand the limits of such informed attack scenarios is required and will be the focus of our future research.

Acknowledgements

This work is supported by Science Foundation Ireland, grant number 07/RFP/CMSF219.

7. REFERENCES


