Attack Detection in Time Series for Recommendation Systems

Sheng Zhang, Amit Chakrabarti, James Ford, Fillia Makedon
Department of Computer Science, Dartmouth College
{clap, ac, jford, makedon}@cs.dartmouth.edu

ABSTRACT

Recent research has identified significant vulnerabilities in automated recommendation systems. Shilling attacks, in which attackers introduce biased ratings in order to influence future recommendations, have been shown to be effective against collaborative filtering algorithms. We postulate that the distribution of item ratings in time can reveal the presence of a wide range of shilling attacks. To construct a time series of ratings for an item, we use a window size of k to group consecutive ratings for the item into disjoint windows and compute the sample average and sample entropy in each window. We show that observing the time series of these two features can expose attack events given reasonable assumptions about their duration. If the number of attack profiles is known, an optimal window size can be derived theoretically in order to best detect the rating distribution changes caused by attacks. For practical applications where the number of attacks is unknown, we propose a heuristic algorithm that adaptively changes the window size. Our experimental results show that monitoring rating distributions in a time series is an effective approach for detecting shilling attacks.

1. INTRODUCTION

Recommendation systems have become popular in the past several years as an effective way to help people deal with information overload. However, since these systems are dependent on external sources of information, they are vulnerable to attacks. In the most common of these attacks (which have been termed "shilling" attacks [10]), attackers influence a recommendation system in a manner advantageous to themselves by introducing biased rating profiles. Shilling attacks can be classified as *push* and *nuke* attacks according to their intent—making a target item more likely or less likely to be recommended, respectively.

Because recommendation systems are widely used in the realm of e-commerce, there is a natural motivation for producers of items (manufacturers, authors, *etc.*) to use these shilling attacks so that their items are recommended to users more often. Therefore, an important research challenge in recommendation systems is to detect and defeat shilling attacks.

A considerable complication in detecting shilling attacks is that it is difficult to precisely and completely define the set of shilling attack patterns. New attacks will continue to arise over time, so an attack detection approach should avoid being restricted to any predefined set of attacks. Our goal in this paper is to seek methods that are able to detect a diverse and general set of recommendation attacks.

Our work begins with the following observation. If we assume that attack profiles are injected into the system within a relatively short period of time,[1] most shilling attack models (discussed in detail in Section 2) share a common characteristic despite their diversity: over their attack period they induce changes in the rating distributions of target items (and possibly other items). For example, a push attack, regardless of its attack model, will cause the rating distribution of a target item to be concentrated on high ratings (or the highest possible rating) during its duration. Similarly, a target item's rating distribution will be concentrated on low ratings (or the lowest rating) in a nuke attack. Our thesis is that examining the rating distribution for each item over a time series can yield a considerable diagnostic power in detecting a large set of attacks.

The idea of treating shilling attacks as events that disturb the rating distribution differs from previous methods that decide whether a user's rating profile is a biased (attack) profile or a normal profile based on how it compares with others overall. Detecting attacks in time series has two key benefits. First, it enables detection of attacks that are difficult to isolate in previous methods where each attack profile is considered separately. Attack profiles generated by some attacks (such as sampling attacks) looks very similar to normal profiles, and thus are almost indistinguishable when only considering individual rating pattern. They are perhaps better detected by systematically mining for rating distribution changes. Second, unusual distributions in time series can reveal previously undefined or unknown attacks. This is a significant advance over heuristic rule-based categorizations or supervised classifications. We note that the time series approach may also find valuable non-malicious anomalies. A simple case might be that a book quickly becomes popular due to a special event.

[1]One attack event may span from several minutes to several days in order to take effect, depending on the size of a recommendation system.

Two key questions that arise are how to effectively extract properties of rating distributions and how to construct a times series in a manner that is appropriate for attack detection. For the first question, we suggest that sample average and sample entropy are effective properties. Sample average captures the change in an item's popularity, while sample entropy captures the distributional change (the degree of dispersal or concentration) in an item's ratings. For the second question, we construct a time series for an item by taking every disjoint k consecutive ratings given to the item (according to their given time) as a window (the basic unit). Sample average and sample entropy are then computed for each window. We show that observing the time series of these two properties exposes attack events.

We give a theoretical analysis to quantify the changes in sample average and sample entropy in time series when attack profiles are injected. Assuming that the number of attack profiles is known, an optimal window size is derived to maximally amplify changes caused by attacks, which helps to enhance the performance of attack detection. For practical applications where this assumption does not hold, we propose a heuristic algorithm to estimate the number of attack profiles and adaptively adjust the window size.

The rest of the paper is organized as follows. Section 2 summarizes the related work on shilling attacks. Section 3 describes our approach of constructing the time series. In Section 4 and 5, we present a theoretical analysis and a heuristic algorithm to find the optimal window size. Finally, we give experimental results in Section 6 and conclude with a discussion in Section 7.

2. RELATED WORK

In this section, we describe popular recommendation attack models and summarize the related work on recommendation attacks.

Five popular attack models are briefly introduced here in the context of a push attack. In *Random attacks* (see Figure 1), a target item will be given the highest rating (r_{max}), but ratings to *filler items* (a proportion of the remaining items) in each rating profile are chosen randomly (usually from a normal distribution). *Average attacks* are a more sophisticated variation: the ratings for filler items in crafted attack profiles are distributed around the mean for each item. *Segmented attacks* target users in a particular segment (*e.g.*, readers expressing an interest in fantasy books). Therefore, attackers concentrate on a set of popular items of similar content to the target item, and give the highest rating to the target and this segment and the lowest rating to filler items. *Bandwagon attacks* can be viewed as an extension of random attacks. They take advantage of the Zipf's law distribution of popularity in consumer markets: a small number of items will receive the lion's share of attention and also ratings. Attackers in this model give the highest rating to selected frequently rated items and random ratings to filler items. Besides the above four models, there is a *Sampling attack* model introduced in [13], in which attack profiles are constructed from entire user profiles sampled from the actual rating database augmented by the highest rating for the pushed item.

O'Mahoney *et al.* [13] first performed empirical studies of the resistance of a user-based CF algorithm [7] under shilling attacks. They further presented a theoretical analysis of the effect of noise—including that injected by attackers (shills)—on the performance of memory-based algorithms. Lam and Riedl [10] evaluated the impact of attacks on both user-based and item-based algorithms, and their study suggested that the item-based approach [14] is much less affected by these attacks. More recently, Burke *et al.* [5] and Mobasher *et al.* [12] showed that segmented attacks have a higher likelihood of success on attacking item-based algorithms than random attacks and average attacks.

In order to detect shilling attacks in recommendation systems, Chirita *et al.* [6] proposed several empirical metrics for analyzing rating patterns of attack profiles and evaluated their potential for detecting random attacks. Zhang *et al.* [16] introduced a probabilistic approach to effectively detect random attacks. It works by computing the log-likelihood of each rating profile given the low-dimensional linear model that best describes the original rating matrix. Unfortunately, both approaches are incapable of detecting average attacks. To the best of our knowledge, there is no approach in the literature that provides a systematic methodology to detect a large variety of shilling attacks.

Previously, time series have been exploited for detecting attacks in network traffic analysis [4, 8]. Entropy has also been proposed for anomaly detection in other contexts, *e.g.*, intrusion detection by [11] and network traffic anomaly detection by [9].

3. CONSTRUCTING A TIME SERIES

Our thesis is that the analysis of rating distributions in time is a powerful tool for the detection of recommendation attacks. The intuition behind this thesis is that all (known) attack models cause changes in the rating distributions of target items (and possibly other items). For example, Table 1 lists the effects of the five attack models surveyed in Section 2. Rating distributions of target items will always become concentrated on high ratings (in push attacks) or low ratings (in nuke attacks) whatever attack model is used. For filler items, distributions become concentrated on low ratings when segmented attacks are injected. When other attack models are used, the rating distributions of filler items may also be concentrated depending on the variance of the distribution that generates ratings.

Figure 2 illustrates how the rating distribution of a target item (from MovieLens, described in Section 6) changes as the result of a push attack. Each plot shows a distribution of 50 ratings. On the left is the distribution during a normal case, and on the right is the distribution during a period in which 40 attack profiles are injected to give the highest rating to this target item. The right plot shows that the distribution becomes more concentrated during attacks.

The rating distribution is a high-dimensional object and so can be difficult to work with directly. However, we can make the observation that in most cases, one can extract very useful information from the following two properties: *the degree of dispersal or concentration* of the distribution and *the degree of popularity*. In the above example, the fact that the distribution is concentrated is a strong indication that should be useful for attack detection, and the degree of popularity is useful for identifying that the item is being pushed.

The measure we use to capture the degree of dispersal or concentration of a rating distribution is the *sample entropy*. Assume that we have an empirical histogram $X = \{n_i, i = 1, \cdots, r_{max}\}$, meaning that the ith possible rating occurs n_i

Filler Items	Target
random ratings	r_{max}

random attack

Targeted Segment	Filler Items	Target
r_{max}	r_{min}	r_{max}

segmented attack

Filler Items	Target
item average ratings	r_{max}

average attack

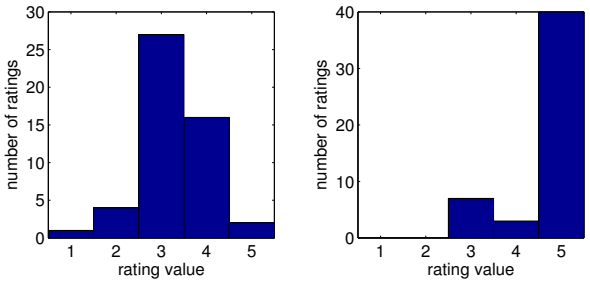
Frequently Rated	Filler Items	Target
r_{max}	*random ratings*	r_{max}

bandwagon attack

Figure 1: Four popular recommendation attack models (assuming that a target item is pushed).

Table 1: Effects on rating distributions by various attack models

Attack Model	Rating Distributions Affected
Random attack	Target items, filler items (possibly—depending on the variance of the used distribution)
Average attack	Target items, filler items (possibly)
Segmented attack	Target items, items in the targeted segment, filler items
Bandwagon attack	Target items, frequently rated items, filler items (possibly)
Sampling attack	Target items



(a) Normal (b) During push attacks

Figure 2: Distribution changes of a target item's ratings induced by push attacks. The distribution becomes concentrated to the highest rating during attacks.

times in the sample. Then the sample entropy is defined as:

$$H(X) = -\sum_{i=1}^{r_{max}} (\frac{n_i}{S}) \log_2 (\frac{n_i}{S}),$$

where $S = \sum_{i=1}^{r_{max}} n_i$ is the total number of ratings in the histogram. The value of the sample entropy lies in the range $[0, \log_2 r_{max}]$. The value 0 is taken when the distribution is maximally concentrated, *i.e.*, all ratings are the same. Sample entropy takes on the value $\log_2 r_{max}$ when the distribution is maximally dispersed, *i.e.*, $n_1 = n_2 = \cdots = n_{r_{max}}$.

The measure we use to capture the degree of popularity is the *sample average*. Using the above notation and assuming that the ith possible rating has the value i, the sample average is defined as

$$M(X) = \frac{\sum_{i=1}^{r_{max}} n_i * i}{S}$$

The value of the sample average lies in the range $[1, r_{max}]$.

To construct the time series of the above two measures for an item, we first sort all the ratings for the item by their time stamps, and then group every disjoint k consecutive ratings into a window. Here, the number of ratings (k) in a window is referred to as the *window size*. For each window, we compute its sample average and sample entropy. Therefore, we obtain two time series, each corresponding to one of the measures.

Grouping equal-size ratings together is not the only way to construct a time series. In the network literature, a basic unit of a time series is usually a unit of time, *e.g.*, 5 minutes. The reason that we choose our approach is that many items are rarely rated in a recommendation system because of the Zipf's law distribution. Even for frequently rated items, the number of ratings during some time slots might still be tiny. Therefore, grouping every k ratings together enforces a measure of fairness in each window.

Denote as w_j the jth window for an item. If ratings to the item are i.i.d. from a distribution $P = \{p(x), x \in [1, r_{max}]\}$ with mean μ and variance σ^2, we can use the following two Propositions to show $M(w_j)$ and $H(w_j)$ are asymptotically Normal. We note that nothing is assumed about the distribution P except the existence of a mean and variance.

PROPOSITION 1. *If ratings to an item are i.i.d. with mean μ and variance σ^2,*

$$\frac{M(w_j) - \mu}{\sigma/\sqrt{k}} \rightarrow N(0, 1).$$

In other words, the sample average for the item can be approximated using a normal distribution with mean μ and standard deviation $\sigma/\sqrt{k}$.

The above proposition follows from the Central Limit Theorem [15].

PROPOSITION 2. *If ratings to an item are i.i.d. from a distribution P,*

$$\frac{H(w_j) - H}{\sqrt{Var(-\log_2 p(x))}/\sqrt{k}} \rightarrow N(0, 1),$$

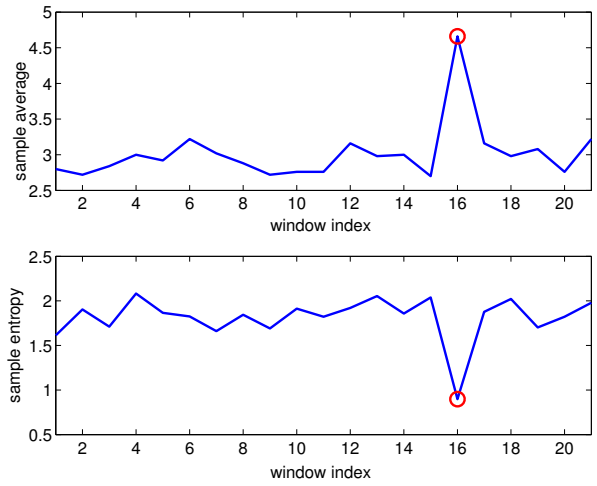


Figure 3: A push attack event stands out clearly when viewed through the lens of sample average and sample entropy. The window size is 50 and the window containing attacks is marked with a circle.

where H is the true entropy. In other words, the sample entropy for the item can be approximated using a normal distribution with mean H and standard deviation $\frac{\sqrt{Var(-\log_2 p(x))}}{\sqrt{k}}$.

This proposition follows from a result in [2] (see also [1]).

As both sample average and sample entropy are asymptotically Normal, we can decide whether a window is an anomaly by computing its z-score (the difference from the mean divided by the standard deviation) for each measure. We illustrate the effectiveness of this approach through the example in Figure 3. The figure plots sample average and sample entropy of the target item around the time of the push attack event whose histograms were previously shown in Figure 2. The window size is 50 and the window containing attacks is marked with a circle. Both plots show that the attack event stands out clearly, while sample average and sample entropy of those windows containing normal ratings vary within a small range. During the attack event, sample average of the window containing attacks inclines sharply and sample entropy declines sharply, consistent with a rating distribution concentration on the highest rating.

4. A THEORETICAL ANALYSIS

Having introduced an approach for constructing a time series for an item, we now quantify the changes of sample average and sample entropy caused by an attack event. Our discussion below is focused on a target item, and all notations used previously for Proposition 1 and 2 will still apply here. We note that the following analysis also holds for other items affected by attacks, *e.g.*, filler items in segmented attacks.

When attack ratings for the target item are injected, more than one window may contain attack ratings. We focus our analysis on the window that has the largest number of attack ratings, and denote it as *anomalous window*. In case there are two or more anomalous windows, one is chosen randomly.

In the following, we will first compute the expected fraction of attack ratings in the anomalous window (subsection 4.1); and then find the optimal window size k to maximize the absolute value of its z-scores for the sample average (subsection 4.2) and for the sample entropy (subsection 4.3). Maximizing the absolute value of these two z-scores helps to identify the anomalous window more accurately in our approach.

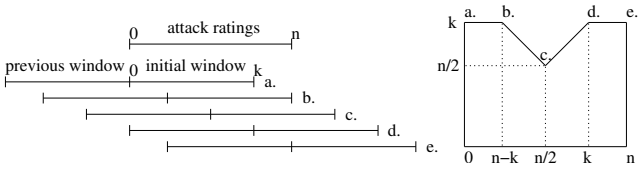


Figure 4: The number of attacks in the anomalous window when the start position of the initial window moves from the start of attacks to the end of attacks in case 2 ($\lceil n/2 \rceil < k < n$).

4.1 The expected fraction of attack ratings

Assuming that the number of attack profiles during an attack event is n, the number of attack ratings for the target item is also n. For ease of expression, we will initially assume that there are no normal (real) ratings given to the target item during the attack. In other words, the n attack ratings are consecutive. The more general situation in which real ratings and attack ratings are intermixed will be discussed in Subsection 4.4. Because the start position of attack ratings is random, we now compute the expected fraction (denoted as λ) of attack ratings in the anomalous window in each of the following three cases.

In case 1 where $n \geq 2k - 1$ ($k \leq \lceil n/2 \rceil$), there always exists a window that is filled with attack ratings. Therefore, we have $\lambda = 1$.

In case 2 where $\lceil n/2 \rceil < k < n$, we compute λ by moving a window from the start of attacks to the end of attacks. When the start position of the window is located in the start of attacks (see Figure 4), the number of attacks in the anomalous window (initial window) is k. This holds until the start position moves to the $(n-k)$th attack. After that, the number of attacks in the anomalous window (initial window) drops linearly until the start position moves to the middle point, $n/2$. Then, the previous window becomes the anomalous window and the number of attacks in that window increases linearly until the start position of the initial window moves to the kth attack. After that, the anomalous window (previous window) is always filled with attacks. Overall, the expected number of attacks in the anomalous window is the area of the right plot divided by n, which gives us $2k - k^2/n - n/4$. Thus, λ in this case is $2 - k/n - n/(4k)$.

In case 3 where $k \geq n$, λ can be computed in a similar way to the above. When the start of the initial window moves from the start of attacks to the middle point (see Figure 5), the number of attacks in the anomalous window (initial window) decreases linearly from n to $n/2$. When the start of the initial window moves from the middle point to the end, the number of attacks in the anomalous window (previous window) increases from $n/2$ to n. Therefore, the expected number of attacks is the area of the right plot divided by n, that is $3n/4$. Thus, λ is $3n/(4k)$ in this case.

Combining all three cases, we conclude that the expected

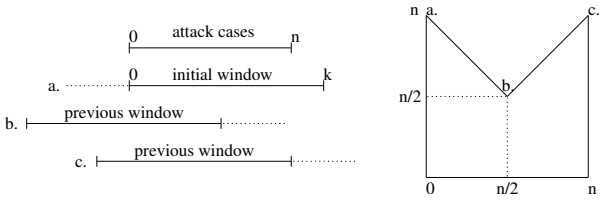


Figure 5: The number of attacks in the anomalous window when the start position of the initial window moves from the start of attacks to the end of attacks in case 3 ($k \geq n$).

fraction of attack ratings in the anomalous window is

$$\lambda = \begin{cases} 1 & k \leq \lceil n/2 \rceil \\ 2 - \frac{k}{n} - \frac{n}{4k} & \lceil n/2 \rceil < k < n \\ \frac{3n}{4k} & k \geq n \end{cases} \quad (1)$$

4.2 Sample average

Denote the anomalous window as $\widetilde{w}$, and denote its z-score for sample average as $Z_M(\widetilde{w})$. We now quantify the expectation of $Z_M(\widetilde{w})$ and compute the optimal k that maximizes its absolute value. Recall that by Proposition 1, the mean of sample average for the item is μ and the standard deviation is $\sigma/\sqrt{k}$. Assuming that the item is being pushed, it follows that

$$\begin{aligned} |E(Z_M(\widetilde{w}))| &= \frac{E(M(\widetilde{w})) - \mu}{\sigma/\sqrt{k}} \\ &= \frac{(1-\lambda)\mu + \lambda r_{max} - \mu}{\sigma/\sqrt{k}} \\ &= \frac{\sqrt{k}\lambda(r_{max} - \mu)}{\sigma}. \end{aligned}$$

The above equation shows that $|E(Z_M(\widetilde{w}))|$ relates to the window size k, the mean value of the item's rating distribution μ, and the standard deviation σ. When μ and σ are smaller, $|E(Z_M(\widetilde{w}))|$ becomes larger, which implies that it is easier to identify the anomalous window $\widetilde{w}$ at that time.

Since μ and σ are fixed, maximizing $|E(Z_M(\widetilde{w}))|$ is reduced to maximizing the term $\sqrt{k}\lambda$. From the previous subsection, λ has three different representations when k is changed. We thus do the optimization in each case.

In case 1 where $k \leq \lceil n/2 \rceil$, $\lambda = 1$. Thus, we have $\sqrt{k}\lambda = \sqrt{k}$. This is maximized when $k = \lceil n/2 \rceil$. The maximum value is $\sqrt{\lceil n/2 \rceil} \approx \frac{\sqrt{2}}{2}\sqrt{n}$.

In case 2 where $\lceil n/2 \rceil < k < n$, $\lambda = 2 - \frac{k}{n} - \frac{n}{4k}$. It follows that

$$\sqrt{k}\lambda = 2\sqrt{k} - \frac{k\sqrt{k}}{n} - \frac{n}{4\sqrt{k}}.$$

With basic calculus, we can determine that the above term is maximized when $k = \frac{2+\sqrt{7}}{6}n \approx 0.7743n$. The corresponding maximum value is about $0.7944\sqrt{n}$.

In case 3 where $k \geq n$, $\lambda = \frac{3n}{4k}$. Thus, we have $\sqrt{k}\lambda = \frac{3n}{4\sqrt{k}}$. This case is maximized when $k = n$, with a maximum value of $0.75\sqrt{n}$.

Combining all the above cases, we get the following Theorem.

THEOREM 1. *The absolute value of the expected z-score for sample average in the anomalous window, $|E(Z_M(\widetilde{w}))|$, is maximized when the window size is equal to $\frac{2+\sqrt{7}}{6}$ times the number of attacks, i.e., $k = \frac{2+\sqrt{7}}{6}n$.*

Theorem 1 shows that if the number of attacks is known, the optimal window size exists and can be computed to best detect the change of sample average caused by attacks. Note that this Theorem also holds if the item is given other ratings, *e.g.*, the lowest rating in nuke attacks.

4.3 Sample entropy

Denote the z-score of $\widetilde{w}$ for sample entropy as $Z_H(\widetilde{w})$. We now quantify $|E(Z_H(\widetilde{w}))|$ and compute the optimal k to maximize this value. Recall that from Proposition 2, the mean of sample entropy for the item is the true entropy H corresponding to its original distribution P and the standard deviation is $\sqrt{\texttt{Var}(-\log_2 p(x))}/\sqrt{k}$.

If the fraction of attack ratings in the anomalous window is λ', the entropy of $\widetilde{w}$ can be bounded using the following idea. To generate a rating in $\widetilde{w}$, we first toss a coin. With probability $1 - \lambda'$, a normal rating is generated from the original distribution P; and with probability λ', the highest rating r_{max} is generated. If the random variable that describes the outcome of the coin toss is denoted as C, we have,

$$\begin{aligned} H(\widetilde{w}) \leq H(\widetilde{w}, C) &= H(C) + H(\widetilde{w}|C) \\ &= H_2(\lambda') + (1-\lambda')H \\ &\leq 1 + (1-\lambda')H, \end{aligned}$$

where $H_2(\lambda') = -\lambda' \log_2 \lambda' - (1-\lambda')\log_2(1-\lambda')$. On the other hand,

$$H(\widetilde{w}) \geq H(\widetilde{w}|C) = (1-\lambda')H.$$

As $\lambda = E(\lambda')$, the absolute value of the expected z-score for sample entropy in the anomalous window is as follows:

$$|E(Z_H(\widetilde{w}))| = \frac{H - E(H(\widetilde{w}))}{\sqrt{\texttt{Var}(-\log_2 p(x))}/\sqrt{k}},$$

$$\frac{\sqrt{k}(\lambda H - 1)}{\sqrt{\texttt{Var}(-\log_2 p(x))}} \leq |E(Z_H(\widetilde{w}))| \leq \frac{\sqrt{k}\lambda H}{\sqrt{\texttt{Var}(-\log_2 p(x))}}.$$

The above inequality shows that both the lower bound and the upper bound on $|E(Z_H(\widetilde{w}))|$ become larger when H is larger and $\sqrt{\texttt{Var}(-\log_2 p(x))}$ is smaller. Using a similar reasoning to that in the previous Subsection, we have that the upper bound is maximized when $k = \frac{2+\sqrt{7}}{6}n$, and the lower bound is maximized when $k = \frac{2H-1+\sqrt{7H^2-4H+1}}{6H}n$. Because when H is large, $\frac{2H-1+\sqrt{7H^2-4H+1}}{6H}$ converges to $\frac{2+\sqrt{7}}{6}$, we obtain the following Theorem.

THEOREM 2. *The absolute value of the expected z-score for sample entropy in the anomalous window, $|E(Z_H(\widetilde{w}))|$, is maximized when $k \approx \frac{2+\sqrt{7}}{6}n$ (for large H).*

Taken in conjunction with Theorem 1, this shows that an optimal window size can be found to simultaneously maximize both the absolute value of the expected z-score for sample average and for sample entropy in the anomalous window.

4.4 An Extension

The above analysis can be easily extended to the case in which real ratings and attack ratings are intermixed in time.

Define ω $(0 < \omega \leq 1)$ as the ratio of attack ratings to the total number of ratings (including both attack and real ratings) given to the item during an attack event. Because the number of attack ratings is n, the total number of ratings is n/ω and is denoted as the *length of an attack event*. Assuming that the ratio of attack ratings to real ratings is fixed within any sub-series of consecutive ratings, the following Corollary can be obtained using a similar reasoning to those in the previous two Subsections.

COROLLARY 1. *If the ratio of attack ratings to the total number of ratings (given to the item) during an attack event is ω, the absolute value of the expected z-scores for sample average and sample entropy in the anomalous window, $|E(Z_M(\widetilde{w}))|$ and $|E(Z_H(\widetilde{w}))|$, are maximized when $k \approx \frac{2+\sqrt{7}}{6}\frac{n}{\omega}$.*

5. A HEURISTIC APPROACH

The previous section shows that when the length of an attack event is known, an optimal window size can be found to best detect rating distribution changes caused by attacks. However, this assumption does not hold for practical applications. In this section, we propose a two-step heuristic approach to first estimate the length of an attack event and then adaptively adjust the window size.

At first, a default window size is chosen and two time series corresponding to sample average and sample entropy are constructed for the item. The default window size can be set as the largest number of attack profiles that is considered negligible. Since the default window size is relatively small, rating distributions of several windows will be affected by a typical attack event. Meanwhile, there might be some detected anomalies in normal windows. However, we can argue that it is unlikely we will find a series of consecutive spikes (with the same direction) in normal windows. Therefore, the length of an attack event can be estimated as the largest number of consecutive anomalies (with the same direction) times the default window size.

After estimating the length of an attack event, we can set the optimal window size as $\frac{2+\sqrt{7}}{6}$ times that length. With a more appropriate window size, attack events will stand out more clearly while fewer false alarm cases are returned.

We observe that in practice when the length of an attack event is much larger than the default window size, some windows containing attacks might not be reported as anomalies. This implies that the length of an attack event will be underestimated at that time. To solve this problem, we can always re-estimate the length of the event using the current window size and then adjust the window size accordingly. This can be repeated iteratively until there is no series of two or more consecutive anomalies.

We illustrate the effectiveness of this heuristic approach in Figure 6. It plots the z-score (for sample entropy) of the same target item previously used in Figures 2 and 3. The number of attacks to push the item is 100, and the fraction of attack ratings (ω) is 2/3. Those windows containing attacks are marked with a circle. The first plot shows the time series with a default window size of 20. If a threshold requiring that the absolute value of z-score be larger than 2 is used, three windows containing attacks (marked with filled circles) will be considered as anomalies. This gives us an estimate that the length of the attack event is $3 \times 20 = 60$. Therefore, we adjust the window size to 46

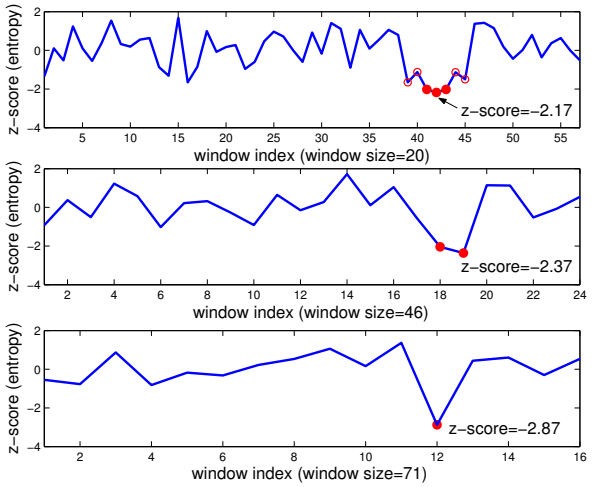


Figure 6: A push attack event stands out more clearly after a heuristic approach is used to repeatedly estimate the length of an attack event and adjust the window size appropriately.

(approximately 0.7743×60) and obtain the second time series. This time, two consecutive windows containing attacks are marked as anomalies. A re-estimation of the length of the attack event is thus $2 \times 46 = 92$. Finally, we obtain the third time series with a window size 71 (approximately 0.7743×92). Compared with the first time series, this one reveals a much clearer presence of the attack event. The z-score of the anomalous window is -2.87 while the lowest z-score of the windows containing attacks in the first time series is -2.17.

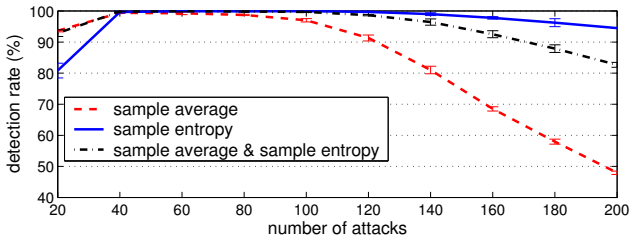
In each iteration of the above heuristic approach, the window size increases by a factor at least $2 \times \frac{2+\sqrt{7}}{6}$. Therefore, the total number of iterations (time series) needed is $O(\log \frac{n}{w})$.

6. EXPERIMENTS

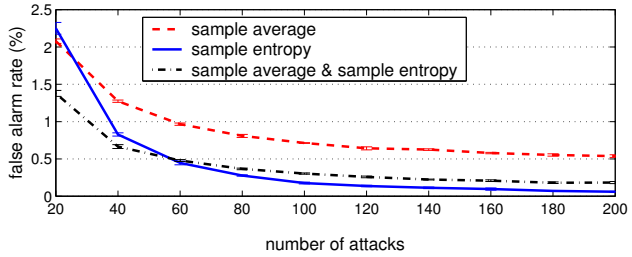
Our experiments are focused on answering three questions: (1) how well can our method detect attack events? (2) how does the context of attacks affect the performance of the method? and (3) Does our heuristic approach improve the detection performance compared with using a fixed window size?

We selected all the items (618 in total) with at least 500 ratings from a MovieLens data set consisting of about 1 million ratings. Ratings are discrete-valued between 1 and 5. We sort ratings for each item by their time stamp. To simulate an attack event, we insert a number of attack ratings for an item into its normal ratings. The start position of an attack event is random, and the ratio of attack ratings to the total number of ratings (given to the item) during the attack event is set to ω.

For each time series of sample average and sample entropy, we mark a window as an anomaly if the absolute value of its z-score is larger than 2. Because the distribution of sample average and sample entropy are approximately Normal, this threshold corresponds to the 95.5% confidence level. A naive combination of these two features is also considered by computing the square root of the sum of the squared values,

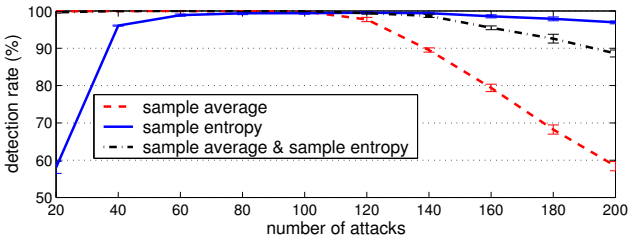


(a) detection rate

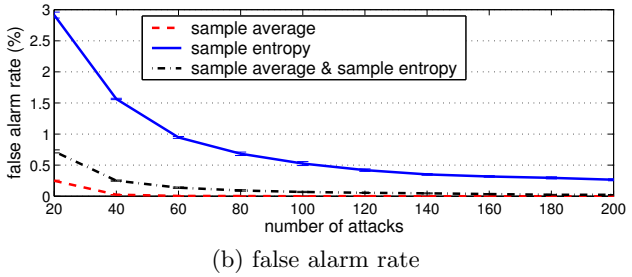


(b) false alarm rate

Figure 7: Detection results of push attacks when the number of attack ratings in an attack event varies. Sample entropy yields the best performance overall.



(a) detection rate



(b) false alarm rate

Figure 8: Detection results of nuke attacks when the number of attack ratings in an attack event varies. The combination of sample average and sample entropy achieves a good balance on the detection rate and false alarm rate.

that is $\sqrt{Z_M(w)^2 + Z_H(w)^2}$. A threshold of $2\sqrt{2}$ is used for this combined variable.

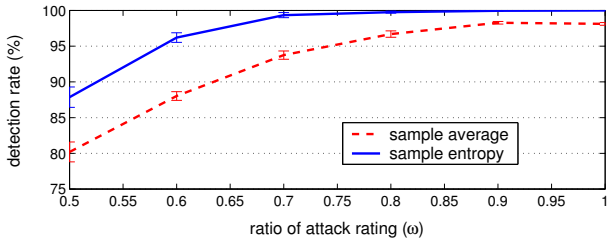
Two metrics, *detection rate* and *false alarm rate*, are used to evaluate the detection performance. The detection rate is defined as the number of detected attack events divided by the number of attack events. An attack event is considered to be detected if a window containing attacks is marked as an anomaly. The false alarm rate is defined as the number of normal windows that are predicted as anomalies divided by the number of normal windows.

We first measure how well our method detects attacks by using it to observe the time series of target items. We execute a push attack for each of the 618 items and attempt to detect the attack by examining its time series. The total number of detected attack events and the total number of false alarm cases are computed over all the items. The window size is set to 20 because this is about the minimum number of attacks that will have a considerable effect in MovieLens according to our experiments and results in [10]. The ratio of attack ratings (ω) is set to 0.8. The number of attack ratings to push the item varies from 20 to 200. Five trials were performed in total, and detection results are plotted in Figure 7. It shows that both time series (for sample average and sample entropy) and their combination yield high detection rates ($> 95\%$) and low false alarm rates ($< 1\%$) when the number of attacks is between 40 and 100. When this number is higher, the detection performance using the sample average drops quickly, while the performance using sample entropy is still stable. The figure implies that sample entropy is the better measure in this case. When the number of attacks is larger than or equal to 60, the sample entropy detection rate is always larger than 95% and the false alarm rate is consistently lower than 0.5%.

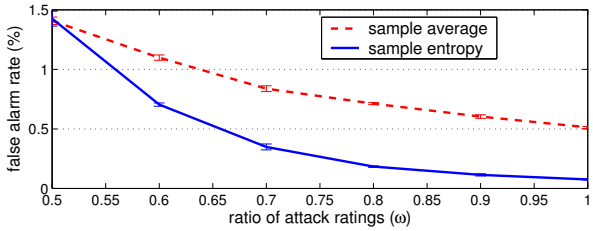
A similar experiment is conducted for nuke attacks, and results are plotted in Figure 8. This figure shows that sample average yields a consistently low false alarm rate; however, its detection rate drops when the number of attacks becomes larger. In contrast, sample entropy always obtains a high detection rate, but its false alarm rate is higher. The combination of these two measures achieves a good balance on both metrics in this case. In most case, its detection rate is larger than 90% with a false alarm rate smaller than 0.5%.

To answer the second question about how our method works when the attack setting changes, we first vary the ratio of the number of attack ratings to the total number of ratings (given to the target item) during an attack event. The target item is pushed and the number of attack ratings in an attack event is set to 100. Results are presented in Figure 9, which shows that both measures obtain good detection results when $\omega \geq 0.7$. When ω is smaller, rating distributions of affected windows become less concentrated. This results in a lower detection rate and a higher false alarm rate. However, even when $\omega = 0.5$, which means that the number of attack ratings equals the number of real ratings given to the item during an attack event, our approach using sample entropy can still get a 87.9% detection rate with a 1.4% false alarm rate.

We next consider a change in the ratings given to a target item. Instead of always giving the highest rating (5) to push an item, we now use ratings of both 4 and 5. We vary the fraction assigned the highest rating (β) to evaluate how our approach responds to this parameter. Results are plotted in Figure 10, which shows that both the detection rate and the false alarm rate are almost linear in their relation to β. When β is equal to 0.5, which means that half the attack ratings have the highest rating, using the time series of sample entropy can yield a 83.9% detection rate with a 1.7% false alarm rate. It is worth noting that, as one would expect, the effect of a push attack event becomes smaller when fewer injected ratings are given the highest rating.

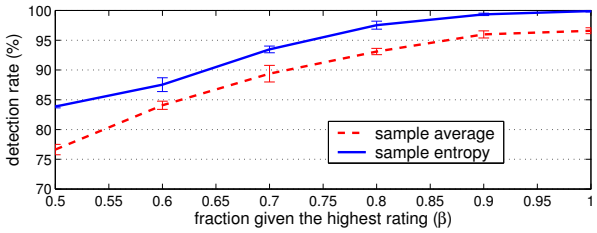


(a) detection rate

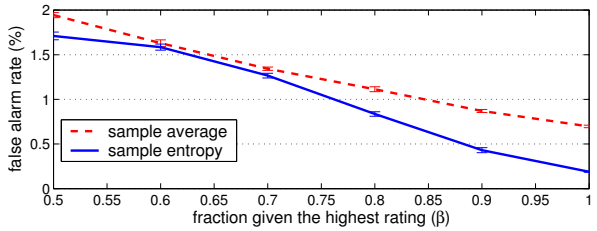


(b) false alarm rate

Figure 9: Result of push attacks (with 100 attack ratings) when ω, the ratio of attack ratings to the total number of ratings (given to a target item) varies. Detection is essentially perfect using sample entropy when this ratio is larger than or equal to 0.7.



(a) detection rate



(b) false alarm rate

Figure 10: Result of push attacks (with 100 attack ratings) when β, the fraction given the highest rating in an attack event varies. Both the detection rate and false alarm rate are almost linear in their relation to β.

In all the experiments above, attack events are detected by observing the time series of target items.[2] We now show that attacks may also be detected by observing the rating distribution changes of filler items when the variance of the attack ratings to filler items is sufficiently small. In average attacks, ratings given to a filler item are usually Normal with a mean equal to the item average. We generate 100 attack ratings for each item in this way (assuming that the item is a filler item) and try to detect attacks by analyzing its times series using sample entropy. Figure 11 presents the results when the standard deviation of the Normal distribution used varies. It shows that when the standard deviation is smaller than or equal to 0.3, sample entropy has a considerable power in detecting attacks—with a detection rate higher than 94% and a false alarm rate lower than 1%.

Finally, we answer the third question about how our heuristic approach (estimating the length of an attack event and adjusting the window size) works compared with using a fixed window size. Push attacks are used and the number of attack ratings in an attack event is randomly set between 50 and 200. The default window size is set to 20. Five trials were performed on all the items, and results from them are listed in Table 2. The detection rates of these two approaches are almost the same, while the heuristic approach yields a much lower false alarm rate.

We also compare these two approaches on a $24,983$ users-by-100 items Jester data set. Ratings in Jester are originally continuous-valued between -10 and 10. We rounded ratings to integers and randomly permuted the ratings given to each item (because time stamps are not available). We use the same experimental setting as that of the previous experiment, and list results in Table 3. The results show that for both sample average and sample entropy, the false alarm rate decreases by at least half when the heuristic approach is used. This verifies that our heuristic approach is effective in improving attack detection performance.

To summarize, our experimental results above demonstrate that for a broad range of settings in attacks, our approach has high detection rates and low false alarm rates. The heuristic approach of adjusting the window size appropriately is shown to be more effective than using a fixed window size.

[2]These experiments also apply to detecting segmented attacks by observing the time series of other items, and detecting bandwagon attacks by observing the time series of frequently rated items. All these items are either given the highest rating or the lowest rating during attacks (see Figure 1).

7. DISCUSSION

Our approach of attack detection is based on the assumption that the duration of an attack event is relatively short so that rating distributions of some items will be changed during that duration. We argue that this is a reasonable assumption because of the following two points. First, shilling attackers (*e.g.*, producers of items) usually hope their attack ratings can take effect as soon as possible, since normal user rating patterns may also change during attacks. Second, if an attack event cannot induce considerable changes in rating distributions of target items within a period of time, then the effect of this attack event will probably be negligible overall.

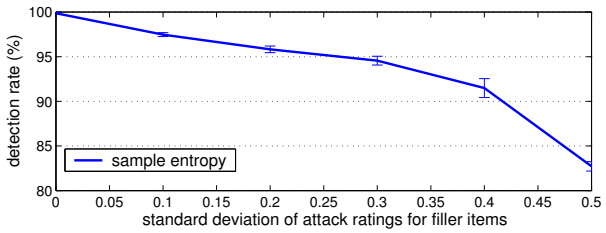
Another assumption we make in this time series-based approach is that ratings given to a certain item satisfy a distribution (whatever it is). This assumption applies well to almost all the items we have tested in MovieLens and Jester. However, in a large-scale recommendation system, ratings to a given item may have a trend over time (*e.g.*, a gradual increase) and/or a periodic trend (*e.g.*, ratings in the weekend are generally higher than ratings in the weekdays).

Table 2: Results of the heuristic approach (estimating the length of an attack event and adjusting the window size) versus results of using a fixed window size in MovieLens. The heuristic approach achieves a much lower false alarm rate, and the detection rates of both approaches are almost the same.

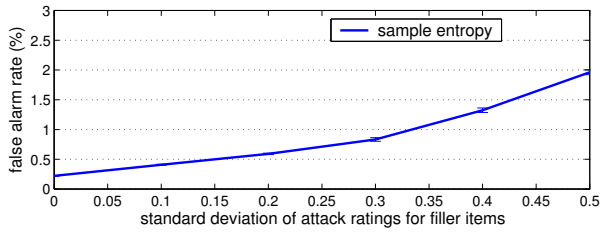
	sample average		sample entropy	
	detection rate	false alarm rate	detection rate	false alarm rate
fixed window size	$83\% \pm 1.2\%$	$0.66\% \pm 0.03\%$	$98\% \pm 0.53\%$	$0.18\% \pm 0.03\%$
heuristic approach	$82\% \pm 1.1\%$	$0.56\% \pm 0.02\%$	$98\% \pm 0.44\%$	$0.09\% \pm 0.04\%$
improvement	-1.2%	15%	0%	50%

Table 3: Results of the heuristic approach versus results of using a fixed window size in Jester. The heuristic approach achieves a much lower false alarm rate.

	sample average		sample entropy	
	detection rate	false alarm rate	detection rate	false alarm rate
fixed window size	$100\% \pm 0\%$	$2.21\% \pm 0.08\%$	$100\% \pm 0\%$	$0.73\% \pm 0.08\%$
heuristic approach	$100\% \pm 0\%$	$0.97\% \pm 0.06\%$	$100\% \pm 0\%$	$0.23\% \pm 0.09\%$
improvement	0%	56%	0%	68%



(a) detection rate



(b) false alarm rate

Figure 11: Attack events can also be detected by examining the time series of filler items when the standard deviation of attack ratings given to filler items is small.

More complex models (*e.g.*, ARIMA, the Auto-Regressive Integrated Moving Average [3]) are needed to incorporate such trends.

Acknowledgment

This material is based in part upon work supported by the National Science Foundation under award number IDM 0308229 and CCF 0448277, and startup funds from Dartmouth College.

8. REFERENCES

[1] A. Antos and I. Kontoyiannis. Convergence properties of functional estimates for discrete distributions. *Random Structures and Algorithms*, 19(3–4):163–193, 2001.

[2] G. Basarin. On a statistical estimate for the entropy of a sequence of independent random variables. *Theory of Probability and Its Applications*, 4(3):333–336, 1959.

[3] P. J. Brockwell and R. A. Davis. *Introduction to time series and forecasting*. Springer, 2nd edition, 2002.

[4] J. D. Brutlag. Aberrant behavior detection in time series for network monitoring. In *Proc. of the 14th USENIX Systems Administration Conference*, pages 139–146, 2000.

[5] R. Burke, B. Mobasher, R. Bhaumik, and C. Williams. Segment-based injection attacks against collaborative filtering recommender systems. In *Proc. of the IEEE Int. Conf. on Data Mining*, pages 577–580, 2005.

[6] P.-A. Chirita, W. Nejdl, and C. Zamfir. Preventing shilling attacks in online recommender systems. In *Proc. of ACM Int. Workshop on Web Information and Data Management*, pages 67–74, 2005.

[7] J. L. Herlocker, J. A. Konstan, A. Borchers, and J. Riedl. An algorithmic framework for performing collaborative filtering. In *Proc. of the 22nd ACM SIGIR*, pages 230–237, 1999.

[8] A. Lakhina, M. Crovella, and C. Diot. Diagnosing network-wide traffic anomalies. In *Proc. of SIGCOMM*, pages 219–230, 2004.

[9] A. Lakhina, M. Crovella, and C. Diot. Mining anomalies using traffic feature distributions. In *Proc. of SIGCOMM*, pages 217–228, 2005.

[10] S. K. Lam and J. Riedl. Shilling recommender systems for fun and profit. In *Proc. of the 13th WWW*, pages 393–402, 2004.

[11] W. Lee and D. Xiang. Information-theoretic measures for anomaly detection. In *Proc. of the IEEE Symp. on Security and Privacy*, pages 130–143, 2001.

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

[13] M. O'Mahony, N. Hurley, N. Kushmerick, and G. Silvestre. Collaborative recommendation: A robust analysis. *ACM Transactions on Internet Technology*, 4(4):344–377, 2004.

[14] B. M. Sarwar, G. Karypis, J. A. Konstan, and

J. Reidl. Item-based collaborative filtering recommendation algorithms. In *Proc. of the 10th WWW*, pages 285–295, 2001.

[15] L. Wasserman. *All of statistics: a concise course in statistical inference*. Springer, 2004.

[16] S. Zhang, J. Ford, and F. Makedon. Analysis of a low-dimensional linear model under recommendation attacks. Submitted. Available at www.cs.dartmouth.edu/ clap/paper/sigir06.pdf.

Appendix

A. Notation

Table 4: Notation

k	window size (the number of ratings in a window)
$H(\cdot)$	sample entropy
$M(\cdot)$	sample average
μ	mean of the ratings to the item
σ	standard deviation of the ratings to the item
P	distribution of the ratings to the item
H	true entropy corresponding to the distribution P
$\widetilde{w}$	anomalous window (the window that has the largest number of attack ratings)
λ	expected fraction of attack ratings in the anomalous window
$Z_M(\cdot)$	z-score for sample average
$Z_H(\cdot)$	z-score for sample entropy
ω	ratio of attack ratings to the total number of ratings (given to the item) during an attack event