Effectiveness of Crawling Attacks Against Web-based Recommender Systems

Runa Bhaumik, Robin Burke, Bamshad Mobasher[*]
Center for Web Intelligence
School of Computer Science, Telecommunication and Information Systems
DePaul University, Chicago, Illinois
(rbhaumik,rburke,mobasher)@cs.depaul.edu

Abstract

The modeling of Web user navigational patterns is a critical component of many Web applications such as those involving Web personalization, recommender systems, and Web analytics. Because such open adaptive systems depend on users' input, malicious third parties may seek to distort the system's behavior by generating false clickstreams. Recent research in collaborative recommender systems has shown that personalization systems that use explicit user feedback in the form of ratings are vulnerable to such attacks. In this paper, we extend this work to the area of adaptive systems that use implicit measures of user behavior such as the navigational patterns employed in Web personalization. We find that, although such usage-based Web recommender systems use different recommendation algorithms, they are nevertheless subject to similar manipulation through appropriate attacks. In this paper, we introduce several examples of "crawling attacks" and demonstrate their effectiveness against some common Web personalization algorithms.

Introduction

Web-based recommender systems help users overcome information overload, assist them in finding items of interest more efficiently and support them navigating in large information spaces. The input to such systems can be explicit ratings provided by users indicating their likes and dislikes, or implicit indicators gathered through analysis of users' interactions. In general, implicit information is easier to gather in quantity than ratings, which require user initiative. For example, Amazon.com monitors each visitor's activity and uses this information to build a navigational profile. Such recommenders model and analyze users' navigational behavior as stored in access logs, a process that is commonly referred to as Web usage mining (Cooley, Mobasher, & Srivastava 1997; Srivastava *et al.* 2000).

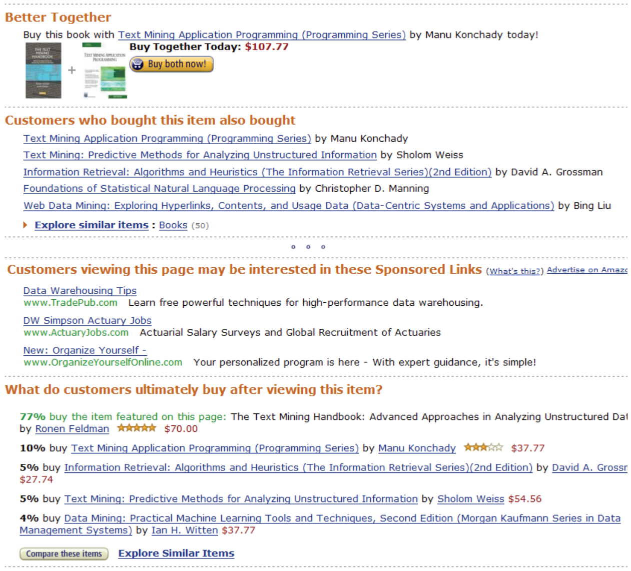
Web usage mining is the application of data mining techniques to discover usage patterns from Web log data, in order to understand and better serve the needs of Web-based applications. In Web personalization such mining techniques can be used to capture, model and analyze the behavioral patterns and profiles of users in order to recommend appropriate pages (Mobasher, Cooley, & Srivastava 2000). This task is accomplished by matching the active user session with the usage patterns discovered through Web usage mining. A number of recent surveys provide detailed discussions of a variety of data mining techniques that can be used for Web personalization (Mobasher 2005; 2007).

Previous research has considered "profile injection" attacks where an attacker inserts biased ratings into a recommender system in the form of explicit ratings (Burke *et al.* 2005). A profile injection attack consists of a set of *attack profiles* being inserted into the system with the aim of altering the system's recommendation behavior with respect to a single target item. The attacker (or an automated client) generates a large number of psuedonyms and masquerading as these users provides ratings for certain items, injecting biased profiles into the system's database with the aim of altering the system's recommendation behavior with respect to a single target item. These attacks do not require a great deal of knowledge about the details of the recommender system or its algorithms (O'Mahony *et al.* 2004; Lam & Riedl 2004; Burke, Mobasher, & Bhaumik 2005; Mobasher *et al.* 2005). It seems likely that such findings would extend to Web personalization systems.

Generating profile injection attacks is more straightforward for a usage-based recommender system than for one that is based on explicit ratings. If a system requires explicit ratings, it will require users to create some sort of account. While a determined attacker may be able to outwit schemes designed to prevent automated account registration, this adds to cost of generating a new profile. On the other hand, navigational based recommendation is typically performed on log data, which is not associated with user accounts, since the whole idea of such usage-based techniques is to avoid the overhead of requiring users to log in, create accounts, etc. Thus an attacker need only successfully disguise his automated site crawler as a large number of different legitimate Web clients, something easily achieved through anonymized browsing techniques.

With such a crawler, an attacker could inject biased clickstream data by visiting the desired combination of items, thereby producing patterns to be mined by the personalization system. Unless countermeasures are in place, such

[*]This work was supported in part by the National Science Foundation Cyber Trust program under Grant IIS-0430303.



Better Together

Buy this book with Text Mining Application Programming (Programming Series) by Manu Konchady today!

Buy Together Today: $107.77

Buy both now!

Customers who bought this item also bought

Text Mining Application Programming (Programming Series) by Manu Konchady
Text Mining: Predictive Methods for Analyzing Unstructured Information by Sholom Weiss
Information Retrieval: Algorithms and Heuristics (The Information Retrieval Series)(2nd Edition) by David A. Grossman
Foundations of Statistical Natural Language Processing by Christopher D. Manning
Web Data Mining: Exploring Hyperlinks, Contents, and Usage Data (Data-Centric Systems and Applications) by Bing Liu

Explore similar items : Books (50)

Customers viewing this page may be interested in these Sponsored Links (What's this?) Advertise on Amazc

Data Warehousing Tips
www.TradePub.com Learn free powerful techniques for high-performance data warehousing.

DW Simpson Actuary Jobs
www.ActuaryJobs.com Actuarial Salary Surveys and Global Recruitment of Actuaries

New: Organize Yourself -
www.OrganizeYourselfOnline.com Your personalized program is here - With expert guidance, it's simple!

What do customers ultimately buy after viewing this item?

77% buy the item featured on this page: The Text Mining Handbook: Advanced Approaches in Analyzing Unstructured Dat
by Ronen Feldman ★★★★★ $70.00

10% buy Text Mining Application Programming (Programming Series) by Manu Konchady ★★★☆☆ $37.77

5% buy Information Retrieval: Algorithms and Heuristics (The Information Retrieval Series)(2nd Edition) by David A. Grossr
$27.74

5% buy Text Mining: Predictive Methods for Analyzing Unstructured Information by Sholom Weiss $54.56

4% buy Data Mining: Practical Machine Learning Tools and Techniques, Second Edition (Morgan Kaufmann Series in Data Management Systems) by Ian H. Witten $37.77

Compare these items **Explore Similar Items**

Figure 1: The Amazon.com product page recommends a set of similar products to the customer based on implicit navigation.

an attack could completely undermine the functionality of the personalization system. For example, Amazon.com and many other Web sites generate a common form of navigation oriented recommendation: when a user is viewing a particular item A, the system may recommend items B, C, D that other users have viewed often in conjunction with A (see for example Figure 1). It is easy to imagine how such a recommender might be manipulated to frequently recommend a target item T, for example, through a specialized crawler that generates browsing session through the site always including T in conjunction with certain popular pages or items.

In this paper, we investigate the effectiveness of various attack models against navigation-oriented (also called "usage-based") personalization algorithms. In particular, we examine the k-Nearest-Neighbor (kNN) algorithm, commonly used in collaborative filtering, a stochastic recommender based on a Markov model, and a data mining approach using association rule mining. Our empirical results, based on a real usage data, show that personalization systems using Web navigation data and implicit preferences as their basis are also vulnerable to biases injected by users.

The rest of the paper is organized as follows. We first begin by modeling attacks that might be mounted against a Web personalization sytem. Next, we give detailed descriptions of our recommendation algorithms based on clickstream data. Finally, we present our experimental results and discussion.

Attack Types

Navigation based recommender systems use implicit feedback captured in the clickstream data and are mainly dependent on the navigation profiles of anonymous users who visit pages in a particular order or combination. In the attack scenario, an attacker's approach may be to insert bias into the recommender system by creating a navigational activity. Fake profiles can easily be generated using a crawling mechanism capable of visiting certain pages in a particular order or combination. We term these attacks as "crawling" attacks. Conceptually, the profile is partitioned in three parts as depicted in Figure 2. The target page (item) i_t will always be included because it is the page being promoted. As described below, some attacks require identifying a group of pages with desirable characteristics. This set is denoted by I_S. I_F is a set of pages chosen randomly to fill the rest of the profile. The strategy for selecting pages in I_S and I_F defines the characteristics of an attack model. In the work presented here, we employ binary ratings for pages – a page is either present or absent in the profile. Some Web personalization systems use numeric ratings derived from the user's dwell time on each page, but binary approaches are more common.

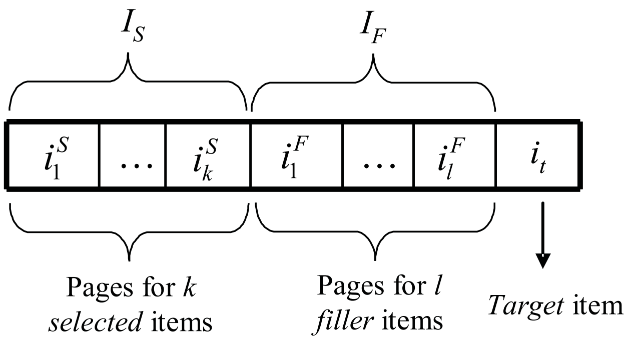


Figure 2: The general form of an attack profile.

The most thoroughly studied profile injection attacks are the *Random Attack* and *Average Attack* introduced in (Lam & Riedl 2004). To generate a random attack, random items are chosen by the attacker from the item database and assigned random ratings, taken from the overall distribution of user ratings in the database, except for the target item, given a high rating. An average attack is similar, except that each item is rated according to its individual distribution in the rating data. It is possible to construct analogous attacks for Web personalization systems, including pages randomly or including them according to their overall frequency of occurrence. However, these attacks have not been found to be effective, primarily due to the sparsity of pages in typical datasets. Web pages are visited usually by traversing links, so a set of randomly-chosen pages is very unlikely to have sufficient similarity to any real user session.

In this paper, we introduce two practical and effective attack types: the *Popular Page Attack* and *Localized Attack*, which are better suited for manipulating navigation oriented Web recommender systems.

The **popular page attack** requires that an attacker identify a few of the most popular pages on the target Web site. The attacker creates a custom crawler that generates repeated visits to those popular pages and the target page.

Over time, such visits will generate a large number of profiles that associate the target page with the popular pages. Such profiles will have a high probability of being similar to a large number of user profiles. For many sites, the most popular pages will be relatively obvious: For example, the site's home page. In this attack model, the set I_S contains these popular pages selected to be part of the attack profile. This attack is analogous to the previously studied bandwagon attack (Burke, Mobasher, & Bhaumik 2005). In our experiments, we use the top two most frequently accessed pages as the selected items of I_S. The set I_F is a set of pages chosen randomly from the other pages accessed less frequently, to fill the rest of the profile.

Our second attack model, **localized attack** is designed to promote a page to a targeted group of users with known preferences. For example, an author who has written a fairy tale for children might be interested in making sure that the book is recommended to the buyers who show an interest in children's books. The goal of this attack is to maximize the similarity between the attack profiles and user profiles containing a group of neighboring pages or items associated with the targeted page or item. This group of pages are localized in the sense that related items tend to be organized in close proximity such as in common categories or directories. For example, if the target page i_t is the page "/books/childrens-books/fantasy/book35.html" promoting a particular children's book, then the attack profiles may contain the pages associated with "/books/childrens-books/fantasy/" and "/books/children-books/", since these would be areas visited by users "in the neighborhood" of the target item. The attacker needs to use the site structure identify pages local to the target item, and associate the target with them. Users interested in these neighbor pages would then be highly likely to get the target page as a recommendation.

Table 1 shows examples of attack profiles of each type promoting a children's book. In the case of popular page attack, an attacker repeatedly visits the most popular pages (in this case, the index and book page along with popular subcategories of the site) together with the target page (book35). The localized attack profile contain the pages most closely associated with the children's book domain along with the target page.

Recommendation Algorithms

The overall process of personalization based on Web usage mining consists of three phases, namely *data preprocessing*, *pattern discovery*, and *recommendation*.

In the data preprocessing phase the raw clickstream data is transformed into a set of user profiles. Raw Web log data is cleaned and sessionized to generate user sessions. Each user session is a logical representation of a user's single visit to the Web site (usually within certain time interval). For details of steps in Web usage preprocessing, see (Cooley, Mobasher, & Srivastava 1999). The output of this phase is a set of n pages, $P = \{p_1, p_2, \cdots, p_n\}$, and a set of m user sessions $U = u_1, u_2, \cdots, u_m$, where each $u_i \in U$ consists of pages from a subset of P. The Web session data can be conceptually viewed as a $m \times n$ session-page matrix $UP = [w(u_i, p_j)]_{m \times n}$, where $w(u_i, p_j)$ represents the weight of page p_j in user session u_i. The weights can be binary, indicating the existence or non-existence of the page in the session, or they may be a function of the occurences or duration of the page in that session. For the purposes of this paper, we are using only binary data. Each user session u can be viewed as a l-length sequence of ordered pairs:

Popular Page Attack Profile	Localized Attack Profile
/index/	/index/
/books/	/books/
/books/horror/	/books/childrens/
/video/action/	books/childrens/fantacy/
/books/childrens/fantacy/ book35/	/books/childrens/fantacy/ book35/

Table 1: Example attack profiles for both attack types

$$u =< ({p_1}^u, w({p_1}^u)), ({p_2}^u, w({p_2}^u)), \cdots, ({p_l}^u, w({p_l}^u)) >$$

where each ${p_i}^u = p_j$ and $w({p_i}^u) = w(u, p_j)$, for some $j \in \{1, \cdots, n\}$.

Given a set of user profiles or sessions as described above, a variety of unsupervised knowledge discovery techniques can be applied to obtain patterns. Techniques such as clustering of users (or sessions) can lead to the discovery of important user or visitor segments. Other techniques such as item (e.g., page) clustering, association rule mining, or sequential pattern discovery, can be used to find important relationships among items based on the navigational patterns of users in the site. In the cases of clustering and association rule discovery, generally, the ordering relation among the pages is not taken into account, thus a user profile is viewed as a set (or, more generally, as a bag) of pages. In the case of sequential patterns, however, we need to preserve the ordering relationship among the pages within transactions, in order to effectively model users' navigational patterns, and thus the sequence representation is used as input to the pattern discovery phase.

The task of the recommendation algorithm is to match the active session of a current user with the patterns discovered through Web usage mining in the pattern discovery phase, and to recommend a set of objects to that user. The type of recommendation algorithm used depends to a large extent on the types of patterns discovered in the previous phase. A discussion of different recommendation algorithms based on Web usage mining can be found in (Mobasher 2005; 2007). In this paper, we examine three algorithms: kNN, Markov model and association rule. We describe these algorithms in more detail in the following sections.

kNN-Based Algorithm

Collaborative filtering based on k-Nearest-Neighbor(kNN) approach involves comparing the active record for a target user with historical records of other users in order to find the top k users who have similar tastes or interests. The mapping of a visitor record to its neighborhood could be based on similarity in ratings of items, access to similar contents or pages, or purchases of similar items. This neighborhood is then used to recommend items not already accessed or purchased by the active user. kNN is not widely-used in Web personalization for efficiency reasons – it does not scale well to extremely large numbers of profiles. However, it is the most thoroughly-studied algorithm for collaboraitve recommendation and we include it as a baseline.

In the context of personalization based on clickstream data, kNN involves measuring the similarity or correlation between the active session $\vec{s}$ and each session $\vec{t}$ in historical records. The top k most similar users to $\vec{s}$ are considered to be the neighborhood for the active session $\vec{s}$, denoted by $NB(s)$. A variety of similarity measures can be used to find the nearest neighbors. In traditional collaborative filtering domains (where feature weights are item ratings on a discrete scale), the Pearson r correlation coefficient is commonly used. This measure is based on the deviations of users' ratings on various items from their mean ratings on all rated items. However, this measure may not be appropriate when the primary data source is clickstream data (particularly in the case of binary weights). Instead we use the *cosine* coefficient, commonly used in information retrieval, which measures the cosine of the angle between two vectors. The cosine coefficient can be computed by normalizing the dot product of two vectors with respect to their vector norms. Given the active session $\vec{s}$ and a session $\vec{t}$ in historical records, the similarity between them is obtained by:

$$sim(\vec{t}, \vec{s}) = \frac{\vec{t} \cdot \vec{s}}{|\vec{t}| \times |\vec{s}|}.$$

In order to determine which items (not already visited by the user in the active session) are to be recommended, a recommendation score is computed for each item $p_i \in P$, based on the neighborhood $NB(s)$ for the active session, where P is the set of all pages. The recommendation score for a page p with respect to $NB(s)$ is computed by

$$Score(p, NB(s)) = \frac{\sum\limits_{t \in NB(S)} w(t,p)}{|NB(s)|}$$

where $w(t,p)$ is the weight for the page p in the active session t. In our experiments we use binary weights for pages, indicating whether a document is accessed or not in user's session. If a fixed number N of recommendations are considered, then the top N items with the highest recommendation scores are considered to be part of the recommendation set.

Markov Model Algorithm

Markov models have long been used for studying and understanding stochastic processes and have been shown to be well suited for predicting the navigational activity in the Web site (Deshpande & Karypis 2001; Pitkow & Pirolli 1999).

A Markov model is represented by the 3-tuple $\langle A, S, T \rangle$ where A is a set of possible actions, S is the set of n states for which the model is built and T is the Transition Probability Matrix (TPM) that stores the probability of performing an action $a \in A$ when the process is in a state $s \in S$. Specifically, $T = [p_{i,j}]_{n \times n}$, where $p_{i,j}$ represents the probability of a transition from state s_i to state s_j. The *order* of the Markov model corresponds to the number of prior events used in predicting a future event. So, a kth-order Markov model predicts the probability of the next event by looking at the past k events. The simplest Markov model predicts the next action by only looking at the last action performed by the user. Given a set of all paths R, the probability of reaching a state s_j from a state s_i via a (non-cyclic) path $r \in R$ is given by: $p(r) = \prod p_{k,k+1}$, where k ranges from i to $j-1$. The probability of reaching s_j from s_i is the sum over all paths: $p(j|i) = \sum\limits_{r \in R} p(r)$.

Markov models can also be used to discover high-probability user navigational paths in a Web site. For example, Borges and Levene (Borges & Levene 1999) modeled user sessions as a hypertext probabilistic grammar (or alternatively, an absorbing Markov chain) whose higher probability paths correspond to the user's preferred trails. An algorithm is provided to efficiently mine such trails from the model.

In the context of recommender systems, A is the set of items and S is the visitor's navigation history, defined as a k-tuple of items visited, where k is the order of the Markov model. Consider the problem of predicting the next page accessed by a user on a Web site. The input data for building the Markov model consists of Web-sessions, where each session represents the sequence of the pages accessed by the user during his/her visit to the site. In this problem, the actions for the Markov model correspond to all sessions of length k that were observed in different sessions, where k is the order of Markov model. In the case of first order models, the states will correspond to single pages and in the case of second order models the states will correspond all pairs of consecutive pages and so on.

Once the states of Markov model are built, TPM can then be computed. The most commonly used approach for estimating the probabilities in the TPM is to use a training set of action-sequences, and each $P_{i,j}$ is estimated based on the frequency of the event that action a_i follows the state s_j. In our experiment we use first order Markov model. Making a prediction for Web session is straightforward. For example, consider a user who has accessed pages P_1, P_5 and P_4. If we want to predict the next page that will be accessed by the user using First order Markov model, we will first identify the state s_4, that is associated with page P_4 and then look up the TPM to find the page P_i that has the highest associated probability. In our algorithm, we consider each page view in the active session window as a separate state and generate recommendations for each of these states.

Association Rule Algorithm

Association rule discovery techniques, such as the Aprioi algorithm were initially developed as techniques for mining market basket data. It also has been used in various domains including Web mining. These algorithms capture the relationships among items based on their patterns of co-occurences across transactions (without considering the ordering of items). In the case of Web transactions, association rules capture relationships among pages based on the navigational patterns of users. The Apriori algorithm (Agrawal & Srikant 1994) finds groups of items occuring frequently together in many transactions (i.e., satisfying a user specified minimum support threshold). In this case, it finds the pages appearing in the preprocessed log. Such groups of items are referred to as frequent item sets. Association rules which satisfy a minimum confidence threshold are then generated from the frequent itemsets.

The recommendation engine based on association rules matches the current user session window with frequent itemsets to find candidate pages for giving recommendations. Given an active session window *w* and a group of frequent itemsets, we consider all the frequent itemsets of size $|w|+1$ containing the current session window. The recommendation value of each candidate pageview is based on the confidence of the corresponding association rule whose consequent is the singleton containing the pageview to be recommended. Our algorithm uses a sliding window over the target user's active profile or session. The size of this window starts at w and is iteratively decreased until an exact match with the antecedent of a rule is found. The details of this algorithm are given in (Mobasher *et al.* 2001).

Experimental Evaluation

Evaluation Metrics

A number of metrics have been proposed in Web usage mining literature for evaluating recommender systems. We use a measure called *hit ratio* in the context of top-N recommendation which has commonly been used to evaluate predictions based on Web usage mining models (Zhou, Jin, & Mobasher 2004). For each user session in the evaluation set, we input the first j pages, as a surrogate for a user's active session, into the recommender system and generate a recommendation set. The value j is the *window size* for the experiments. We then compare the recommendation set with the $(j+1)$ page – the next page that the user actually visited. A "hit" is noted if the page $(j+1)$ appears in the recommendation set. We define the *hit ratio* as the total number of hits divided by the total number of sessions in the evaluation set.

In measuring the impact of an attack, we are interested not in raw performance – how well the recommender does its job – but rather in the change in performance induced by an attack. In the attacks discussed here, the attacker will desire that the target item will be more likely to be recommended after the attack than before. We measure this benefit to the attack with the metric *target hit ratio*. The same operation is performed as for the hit ratio metric described above, but instead of measuring the frequency of selection of the next item the user actually visited, we measure how frequently the target item is recommended. Let R_u be the set of top N recommendations for user u. For each push attack on a target item i, the value of a hit for user u denoted by H_{ui}, can be evaluated as 1 if $i \in R_u$; otherwise, it is zero. We define target hit ratio as the number of hits across all user sessions divided by the total number of user sessions in the evaluation set, computed as:

$$TargetHitRatio_i = \sum_{u \in U} H_{ui} / |U|$$

The average target hit ratio can then be calculated as the sum of the hit ratios for attacks on each item i across all items divided by the total number of target items in the test date. In our experiments we report the average of this value across all test user sessions. Of course, a measure of the impact of the attack is also dependent on what the Target Hit Ratio would have been in the absence of the attack profiles. In most cases, the pre-attack value is vanishingly small as the results below show.

In both cases, the recommendation process uses only a portion of current user's activity, we term this as **active session window**. This is consistent with the needs of on-line Web personalization systems that seek to provide recommendations as quickly as possible, after the user has visited only a few pages. In kNN based method, we consider this active session window as a whole session and generate recommendation. For the first-order Markov model, we consider each pageview in this active session window as a separate state and generate recommendations for each of these states. We then average over all hit ratios computed for the user and finally over all users to produce an overall average.

Experimental Methodology

In this section we describe the characteristics of our data sets and the experimental methodologies used in this paper.

CTI Data This data is based on the server logs of the host Computer Science department spanning a one-month period. We refer to this data set as the "CTI" data. Standard preprocessing techniques (Cooley, Mobasher, & Srivastava 1999) were applied to clean the data, remove spider references, and sessionize the data. Furthermore, sessions smaller than 6 page references were removed from the data set and the average session size is 9.8 pages. The initial preprocessed data set contained more than 100,000 sessions and over 4,000 pages. Further aggregation was performed to "roll up" low-support (infrequently accessed) pages to their common root node in the site hierarchy. The final data set spanned approximately 700 aggregated page views representing access to items (or item categories in the case of aggregated references). The site is highly dynamic, involving numerous online applications, including online admissions application, online advising, online registration, and faculty-specific Intranet applications.

This dataset was randomly divided into a training and evaluation sets. The training set consisting of 17,040 user sessions was used to build the models while the test set consisting of more than 4,000 user sessions was used to evaluate the recommendations generated by the models.

For this site, we constructed a popular page attack profiles by choosing top two most frequently visited pages in the set

Course	Faculty
/courses/default.asp	/people/
/courses/syllabisearch.asp	/people/facultyinfo.asp
/courses/syllabilist.asp	/people/search.asp

Table 2: Examples of faculty and course related pages

I_S and 20 randomly chosen pages accessed less frequently from CTI site in the set I_F.

To simulate localized attacks in which an item is pushed by creating strong associations with a segment of users that show interest in a particular groups of items, we considered two scenarios: one in which a particular faculty page would be pushed to all users showing an interest in any of the faculty-related pages, and another in which a particular course would be pushed to all of the visitors looking at any of the course-related pages. See Table 2 for example of typical page views. In this example, the set I_S contains 6 faculty and 6 course related pages for faculty and course segments respectively; and the set I_F is empty. Our evaluation set consisted of those users whose session included any one of these segment pages within the first six pageviews. There were 525 and 993 user sessions in the evaluation sets for the faculty-segment and course-segment respectively. The experimental results shown here averages these two segments.

As attack targets, we used a set of pages, consisting of 25 pages each in the faculty and course areas. These were chosen to be specific courses or faculty home pages, not top-level navigation pages. Each of these target pages was attacked individually and the results reported below represent averages over targets and all sessions in test data set.

For all the attacks, we generated a number of attack profiles and inserted them into the system database and then generated recommendations. We measure "size of attack" as a percentage of the pre-attack user count. For example, if the training set contains 100,000 user sessions, an attack size of 1% corresponds to 1,000 attack profiles added to the system. Since attacks can be generated automatically through an appropriately designed crawler, there would be little difficulty in producing an attack of this or larger scale.

For these experiments, we implemented the popular page attack by choosing the two top most frequently accessed pages. Filler items for this attack were chosen randomly. The localized attack was implemented by choosing two areas of the CTI site: those associated with courses and those associated with faculty members. To generate nearby pages, we looked at pages with associated URLs as shown in Table 2. There are no filler items in this attack.

NC Data The second data set is based on the server logs for *Network Chicago* which combines the programs and activities of the Chicago Public Television and Radio (*www.networkchicago.com*). The preprocessed data set contains over 51,000 user sessions and 295 Web pages. This data set is filtered to include only sessions of size 6 or more (8.8 pages in average), resulting in about 5000 sessions. In contrast to the CTI data, this site is comprised primarily of

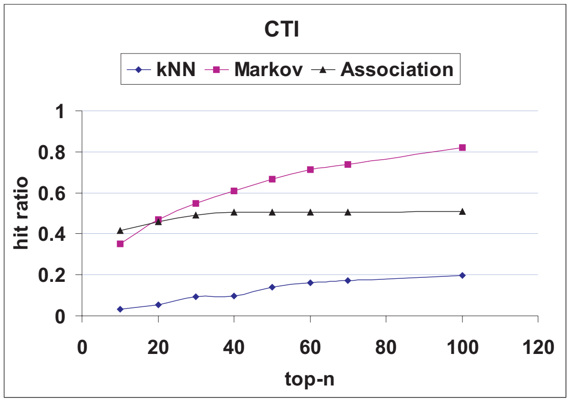


Figure 3: The hit ratio results for CTI Site. Session Window Size =3 support = .003

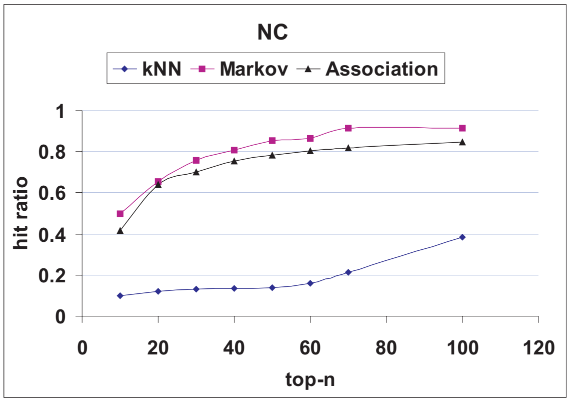


Figure 4: The hit ratio results for NC Site. Session Window Size =3 support = .003

static pages grouped together based on their association with specific content areas. This data set is used as the "NC data". The NC site also has a broad audience and is characterized by long navigational paths. There are two main content areas in this site: " WTTW11" and "98.7 WFMT". The first one is Chicago's premier public television station and the second one is public radio station. This dataset was randomly divided into a training and evaluation sets. The training set of 4489 user sessions was used to build the models while the test set consisting of 498 user sessions was used to evaluate the recommendations generated by the models.

For the popular page attack, we selected top two most frequently visited pages in the set I_S and the set I_F has been filled up by 20 randomly chosen pages accessed less frequently from NC site. To construct a localized attack, we focused on the program "Check Please" in the category WTTW11. This program mainly guides customers selecting restaurants and foods. It also displays the reviews by cuisine, food recipes etc. The set I_S contains pages associated with recipes. This attack targets users who regularly visit this specific area and tries to push target pages. We have selected 10 randomly chosen target pages to promote

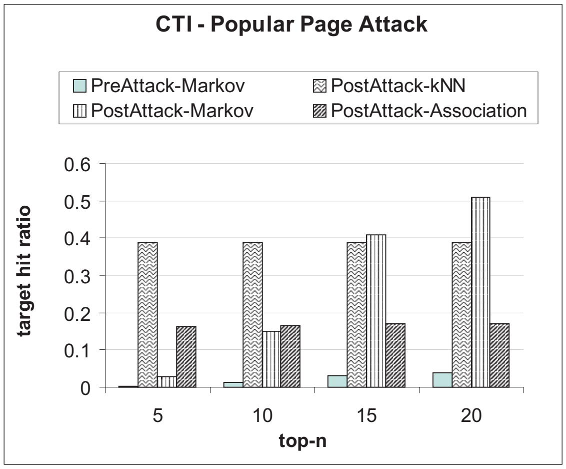


Figure 5: The target hit ratio results for CTI Site for popular page attack. Session Window Size =3 , support = .003, Attack Size = 10%

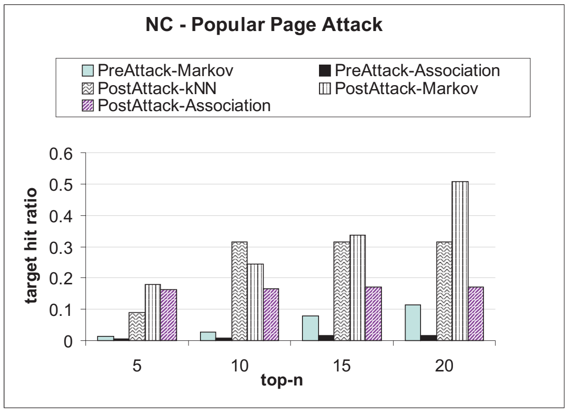


Figure 6: The target hit ratio results for NC Site for popular page attack. Session Window Size =3 ,support = .003, Attack Size = 10%

along with the group of pages shown in "Check Please" program. Our evaluation set consisted of those users, whose session included any one of these recipe-related (segment) pages within the first six pageviews. There are 89 user sessions in the evaluation set for the recipe segment.

Experimental Results

We first compare the accuracy of all algorithms by measuring hit ratio. The window size is set to 3, small enough so the user can get recommendations after visiting just a few pages. As Figure 3 and Figure 4 show, the Markov model algorithm is significantly more accurate than kNN algorithms in both sites, with the association rule algorithm not far behind. Even if at lower number of recommendations, Markov model performs better than kNN. The results shown here are not surprising, because the Markov model is well suited for predicting the sequential process of Web navigation.

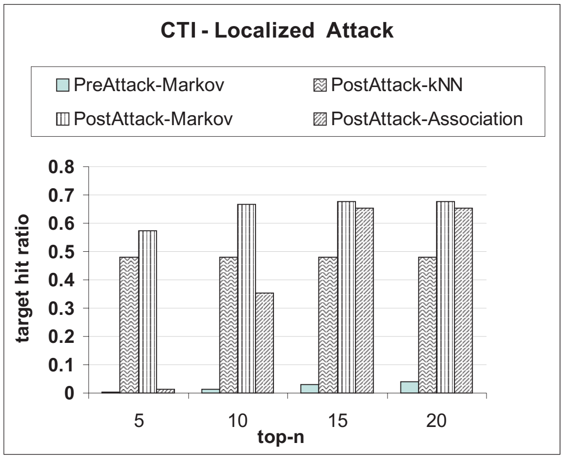


Figure 7: The target hit ratio results for CTI Site for localized attack. Session Window Size =3 ,support = .003 , Attack Size = 10%

The effectiveness of popular page attack and localized attack against all the algorithms are shown in Figure 5 - Figure 8, for two different Web sites. Here the results are shown varying top-n pages where window size is fixed to 3 and attack size is set to 10%. In the CTI site, the target hit ratio for kNN algorithm is relatively flat across different recommendation set sizes against both the attacks. This was not the case for the Markov algorithm and association rule. This would seem to indicate that the attack has a relatively scale-free impact on the kNN algorithm: as the retrieval set gets larger the attacked item is represented in the same proportion. Consider the situation where at 10% attack size, neighbors for a test user are all from attack profiles. In this situation, in kNN algorithm, the page score for the attack page is always high for these attack neighbors and always at the top of the recommendation pages. Similar behavior is observed for NC cite at top-10 pages and higher in kNN algorithm.

These results also show the attack effectiveness prior to an attack. The target hit ratio results for Markov model aprior to an attack are higher than all other algorithms in both sites. We only present results for pre-attack models, where the value of the Target Hit Ratio is not zero prior to the attack.

Consequently, the attack effectiveness are different in two sites against all attacks. In the case of CTI site, the popular page attack has relatively low impact at all retrieval set sizes when compared to the localized attack. In contrast, localized attacks are less effective in NC site than popular page attack. The site characteristics may have an inmpact on these results as NC site is comprised of static pages grouped together based on their association with specific content area. It is also interesting to see that the target hit ratio for the association rule algorithm against both attacks is very low at the smallest retrieval set size (which is 5 in our experiments), but at a higher size, it is as high as Markov algorithm. In our experiments, we have set filler items to 20 against popular page attack. Other experiments(not shown here) show that

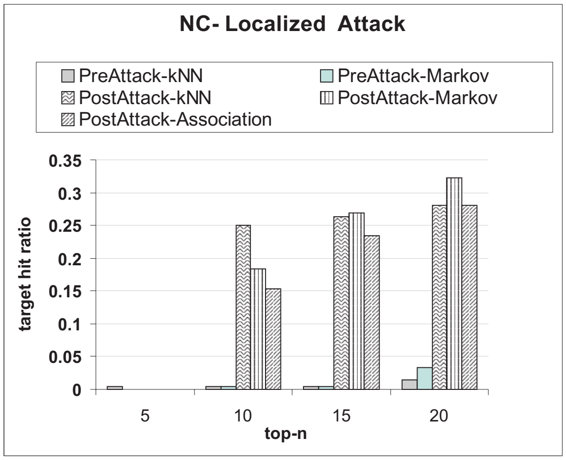


Figure 8: The target hit ratio results for NC Site for localized attack. Session Window Size =3 ,support = .003, Attack Size = 10%

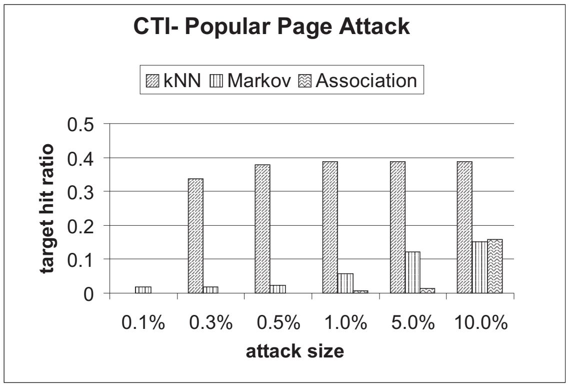


Figure 9: The target hit ratio results for CTI Site for popular page attack.Session Window Size =3, support = .003, Recommendation Size = 10

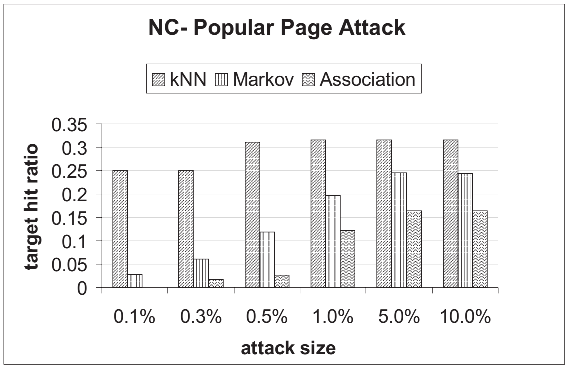


Figure 10: The target hit ratio results for NC Site for popular page attack. Session Window Size =3, support = .003, Recommendation Size = 10

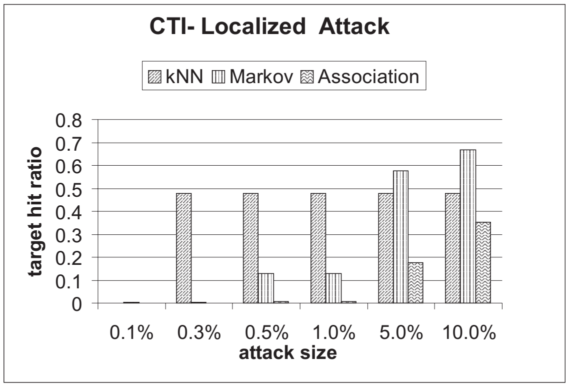


Figure 11: The target hit ratio results for CTI Site for localized attack. Session Window Size =3 , support = .003, Recommendation Size = 10

filling I_F with a large number of random pages decreases the attack effectiveness against kNN algorithm. The reason behind this is, a set of randomly-chosen pages is very unlikely to have sufficient similarity to any real user session.

The Markov and association rule algorithms do not appear to offer any improved robustness over the kNN recommender at higher recommendation pages. In previous research involving explicit rating data, model-based approaches have shown increase robustness against attacks by building models that to some extent segregated the attack profiles (Mobasher, Burke, & Sandvig 2006). This does not appear be the case when using first-order Markov model or association rule algorithm with Web navigation data. Indeed the impact of the segment attack was even more pronounced that in the case of the kNN-based algorithm for both sites.

To examine the impact of varying the attack sizes, we fixed the size of the recommendation set to 10, and varied the number of attack profiles. Interestingly, kNN again shows little change across attack sizes, not differing appreciably between 10% and 0.5% attack sizes. Even small attacks appear to be successful in producing enough neighbors for the test sessions to have an impact for both sites.

The results are quite different for the Markov and association rule algorithms. The localized attack is highly successful even at low 5% attack size against Markov algorithm in both sites, whereas impact of this attack is less against association rule than Markov in CTI. But attacks have little impact at lower attack sizes in both sites. Popular page attack is less successful than the localized attack in both sites. Clearly, the Markov and association rule algorithms are more robust at small attack sizes with the association rule algorithm performing slightly better than Markov.

Figure 13 and Figure 14 show the impact of increasing session window size for both sites against popular page attack. The session window size is the number of interactions that are used as the basis for recommendation generation. Here, all the algorithms increase their robustness by taking more data into account. However, the tradeoff is that recommendations become more difficult to generate. In the case of the kNN, there will be fewer nearby neighbors while in Markov there will be more dispersed set of link probabilities

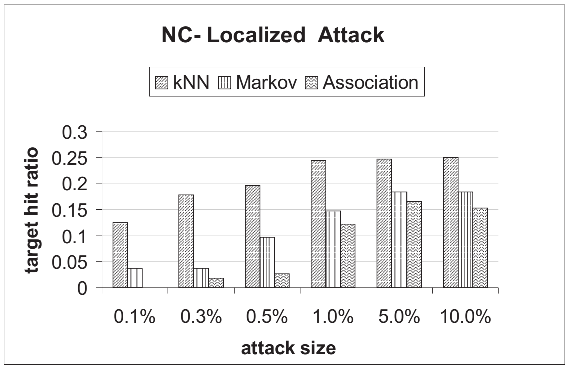


Figure 12: The target hit ratio results for NC Site for localized attack. Session Window Size =3 , support = .003, Recommendation Size = 10

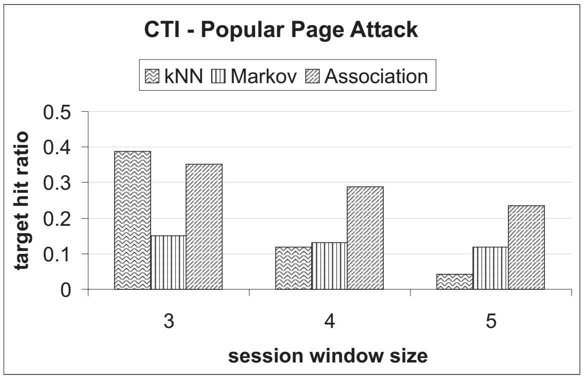


Figure 13: The target hit ratio results for CTI Site for Popular Page Attack , support = .003, Recommendation Size = 10, attack size 10%

in the Markov model. In association rule algorithm, the recommendation engine matches the current user session window with frequent itemsets to find candidate pages for giving recommendations. Given an active session window *w* and a group of frequent itemsets, we only consider all the frequent itemsets of size $|w| + 1$ containing the current session window. In our experiment, we set maximum frequent itemset to 4, so target hit ratio didn't increase as window size increases rather it decreased by accumulating more sessions to generate recommendation. Similar behavior is observed against localized attack not shown here.

Overall, our experimental results show that crawling attacks are effective against the most common personalization algorithms based on Web usage mining.

Conclusions and Future Work

Researchers have established that collaborative filtering systems using explicit ratings input are extremely vulnerable to "profile injection" attacks. In this paper, we extend this work to recommender systems that use Web usage data, showing that these algorithms are also vulnerable to biases injected by attackers. In particular, we have shown that two simple techniques based on minimal site data can form the basis for effective attacks. We have shown that the most common types of personalization algorithms based on Web navigation data, such as stochastic and association rule techniques, can be manipulated to push individual target Web pages, with association rules showing the greatest robustness. We plan to examine other Web usage mining algorithms such as sequential pattern mining, and unsupervised clustering approaches in our future research. In addition to looking for vulnerabilities, we also plan to develop effective approaches to the detection and prevention of such attacks.

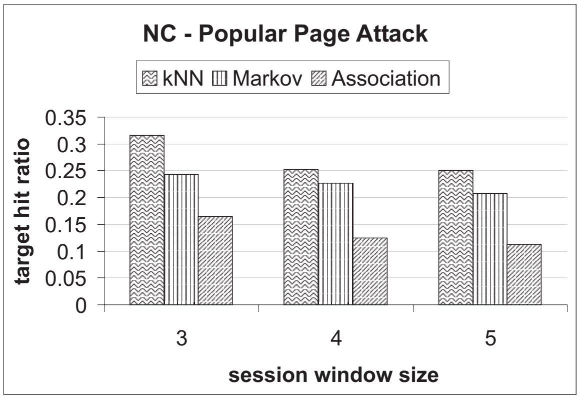


Figure 14: The target hit ratio results for NC Site for Popular Page Attack varying session window size =3, support = .003, Recommendation Size = 10, attack size 10%

References

Agrawal, R., and Srikant, R. 1994. Fast algorithms for mining association rules. In *Proceedings of the 20th International Conference on Very Large Data Bases (VLDB'94)*.

Borges, J., and Levene, M. 1999. Data mining of user navigation patterns. In Masand, B., and Spiliopoulou, M., eds., *Web Usage Analysis and User Profiling: Proceedings of the WEBKDD'99 Workshop*, LNAI 1836. Springer-Verlag. 92–111.

Burke, R.; Mobasher, B.; Zabicki, R.; and Bhaumik, R. 2005. Identifying attack models for secure recommendation. In *Beyond Personalization: A Workshop on the Next Generation of Recommender Systems*.

Burke, R.; Mobasher, B.; and Bhaumik, R. 2005. Limited knowledge shilling attacks in collaborative filtering systems. In *Proceedings of the 3rd IJCAI Workshop in Intelligent Techniques for Personalization*.

Cooley, R.; Mobasher, B.; and Srivastava, J. 1997. Web mining: Information and pattern discovery on the world wide web. In *Proceedings of the 9th IEEE International Conference on Tools with Artificial Intelligence (ICTAI'97)*, 558–567.

Cooley, R.; Mobasher, B.; and Srivastava, J. 1999. Data preparation for mining world wide web browsing patterns. *Journal of Knowledge and Information Systems* 1(1):5–32.

Deshpande, M., and Karypis, G. 2001. Selective markov models for predicting web-page accesses. In *Proceedings of the First International SIAM Conference on Data Mining*.

Lam, S., and Riedl, J. 2004. Shilling recommender systems for fun and profit. In *Proceedings of the 13th International WWW Conference*.

Mobasher, B.; Dai, H.; Luo, T.; and Nakagawa, M. 2001. Improving the effectiveness of collaborative filtering on anonymous web usage data. In *Proceedings of the IJCAI 2001 Workshop on Intelligent Techniques for Web Personalization (ITWP01)*.

Mobasher, B.; Burke, R.; Bhaumik, R.; and Williams, C. 2005. Effective attack models for shilling item-based collaborative filtering systems. In *Proceedings of the 2005 WebKDD Workshop, held in conjuction with ACM SIGKDD'2005*.

Mobasher, B.; Burke, R.; and Sandvig, J. 2006. Model-based collaborative filtering as a defense against profile injection attacks. In *Proceedings of the 21st National Conference on Artificial Intelligence*, to appear. AAAI.

Mobasher, B.; Cooley, R.; and Srivastava, J. 2000. Automatic personalization based on web usage mining. *Communications of the ACM* 43(8):142–151.

Mobasher, B. 2005. Web usage mining and personalization. In Singh, M. P., ed., *Practical Handbook of Internet Computing*. CRC Press.

Mobasher, B. 2007. Data mining for personalization. In Brusilovsky, P.; Kobsa, A.; and Nejdl, W., eds., *The Adaptive Web: Methods and Strategies of Web Personalization* , volume 4321 of *Lecture Notes in Computer Science*. Berlin Heidelberg New York: Springer-Verlag.

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

Pitkow, J., and Pirolli, P. 1999. Mining longest repeating subsequences to predict www surfing. In *In Proceedings of the 2nd USENIX Symposium on Internet Technologies and Systems*.

Srivastava, J.; Cooley, R.; Deshpande, M.; and Tan, P. 2000. Web usage mining: Discovery and applications of usage patterns from web data. *SIGKDD Explorations* 1(2):12–23.

Zhou, Y.; Jin, X.; and Mobasher, B. 2004. A recommendation model based on latent principal factors in web navigation data. In *In Proceedings of the 3rd International Workshop on Web Dynamics, Held at the WWW 2004 Conference, New York,*.