Robustness and Accuracy Tradeoffs for Recommender Systems Under Attack

Carlos E. Seminario and **David C. Wilson**
Software and Information Systems Department
University of North Carolina Charlotte
cseminar@uncc.edu
davils@uncc.edu

Abstract

Recommender systems assist users in the daunting task of sifting through large amounts of data in order to select relevant information or items. Common examples include consumer products and services, such as for songs, books, articles, etc. Unfortunately, such systems may be subject to attack by malicious users who want to manipulate the system's recommendations to suit their needs: to promote their own (or demote a competitor's) product/service, or to cause disruption in the recommender system. Attacks can cause the recommender system to become unreliable and untrustworthy, resulting in user dissatisfaction. Developers already face tradeoffs in system efficiency and accuracy, and designing for robustness adds an additional dimension for consideration. In this paper, we show how the underlying implementation choices for item-based and user-based Collaborative Filtering recommender systems can affect the accuracy and robustness of recommender systems. We also show how accuracy and robustness can change over a system's lifetime by analyzing a set of temporal snapshots from system usage over time. Results provide insight into some of the tradeoffs between robustness and accuracy that operators may need to consider in development and evaluation.

Introduction

Recommender systems assist users in the daunting task of sifting through large amounts of data in order to determine the best action to take regarding the selection of a variety of consumer products and services including movies, songs, books, articles, and restaurants, among others. Recommender systems are implemented as content-based, collaborative filtering (user-based or item-based), or a hybrid of the two (Adomavicius and Tuzhilin 2005). Furthermore, these systems are subject to attack by malicious users who want to manipulate the system's recommendations to suit their needs: to promote their own product/service, to demote a competitor's product/service, or to cause disruption in the recommender system. Research in attacks on recommender systems started in 2002 (O'Mahony, Hurley, and Silvestre 2002) and has continued to be studied, especially in the areas of attack detection and improvements in algorithm robustness (Lam and Riedl 2004; O'Mahony et al. 2004; O'Mahony, Hurley, and Silvestre 2005; Chirita, Nejdl, and Zamfir 2005; Burke et al. 2006; Mobasher et al. 2007; Mehta and Nejdl 2008). These intentional attacks can cause the recommender system to become unreliable and untrustworthy and can result in user dissatisfaction. Previous research has shown that collaborative filtering (CF) recommender systems (RS) under attack behave differently; typically, item-based CF has been shown to be more resistant to attack than user-based CF (Lam and Riedl 2004). Attacks are typed to either promote ("push") a target item by setting the rating to the maximum value or demote ("nuke") a target item by setting the rating to the minimum value; furthermore, attackers will submit one or more user profiles containing item ratings (called attack profiles) that push or nuke a specific item. In order to correlate with other legitimate users in the system, the attack profiles will contain ratings for non-target items; these ratings can be selected randomly or more intelligently if the attacker has prior knowledge of the ratings in the CF system. Research results also indicate that the type and size of attack can affect the recommendations produced by both item-based and user-based CF systems (Mobasher et al. 2007; Burke, O'Mahony, and Hurley 2011). Recently, researchers have started to investigate the temporal aspects of recommender systems showing how rating data changes over time and how these systems are evaluated temporally (Koren 2009; Lathia, Hailes, and Capra 2009; Burke 2010).

Our research investigates the accuracy and robustness of collaborative filtering recommender systems, particularly in the context of how system characteristics may change over time and usage. This paper builds on the work of researchers cited above and has as its main objective to investigate the tradeoffs that recommender system operators may encounter when balancing a system's accuracy and its robustness. In particular, we show how CF system recommendations are impacted under normal and attack conditions using accuracy and robustness metrics, as follows:

- How user-based and item-based prediction algorithms differ under normal and attack conditions using accuracy and robustness metrics,
- How the accuracy and robustness results change as the dataset evolves over time, using item-based prediction algorithms under normal and attack conditions.

Background

A comprehensive set of guidelines for evaluating recommender systems was provided by (Herlocker et al. 2004) and more recently in (Shani and Gunawardana 2011); these guidelines include a description of evaluation metrics such as Mean Absolute Error that is used to measure the prediction accuracy of a recommender system. Mean Absolute Error is calculated as follows.

$$MAE = \frac{\sum_{i=1}^{n} \mid ActualRating_i - PredictedRating_i \mid}{n} \quad (1)$$

where n is the total number of ratings predicted in the test run.

Robustness metrics such as Hit Ratio and Prediction Shift have been discussed in detail in (Mobasher et al. 2007; Burke, O'Mahony, and Hurley 2011). These were used to measure the success of the attack (from the attacker's standpoint) such that a high Hit Ratio or a high Prediction Shift meant that the attack succeeded in changing the recommendations produced by the CF system. The Prediction Shift metric is defined as follows: Let U_T and I_T be the sets of users and items, respectively, in the test data. For each user-item pair (u, i), the Prediction Shift denoted by $\Delta_{u,i}$ can be measured as $\Delta_{u,i} = p\prime_{u,i} - p_{u,i}$ where p and $p\prime$ are the pre- and post-attack predictions, respectively. A positive value means that the attack has succeeded in making the pushed target item more positively rated. The Average Prediction Shift for a target item i over all users can be computed as $\Delta_i = \frac{\sum_{u \epsilon U_T} \Delta_{u,i}}{|U_T|}$ and the Average Prediction Shift for all items tested can be computed as $\overline{\Delta} = \frac{\sum_{i \epsilon I_T} \Delta_i}{|I_T|}$.

Although prediction shift is a good indicator that an attack has successfully (from the attacker's standpoint) made a pushed item more desirable, or a nuked item less desirable, the item may still not make it into the top-N list of recommendations presented to the user and where the top-N list of recommendations are ranked by predicted ratings. So, another metric, Hit Ratio, was developed to indicate the percentage of users that have the target item in their top-N list of recommendations.

Let R_u be the set of top-N recommendations for user u. If the target item appears in R_u for user u, the scoring function H_{ui} has value 1; otherwise it is zero. Hit Ratio for a target item i is given by $HitRatio_i = \frac{\sum_{u \epsilon U_T} H_{u,i}}{|U_T|}$. The Average Hit Ratio can be calculated as $\overline{HitRatio} = \frac{\sum_{i \epsilon I_T} HitRatio_i}{|I_T|}$.

Temporal aspects of recommender systems showing how rating data changes over time and how these systems are evaluated have been covered in (Koren 2009; Lathia, Hailes, and Capra 2009; Burke 2010). Evaluation of attacks on temporal datasets is relatively new, so this study is extending the use of static robustness metrics, such as Hit Ratio and Prediction Shift, for dynamic analysis.

Recommender System Algorithms

In order to generate predictions, user-based and item-based CF recommender systems follow a consistent process: first, establish similarity between users (for user-based CF systems) or items (for item-based CF systems), then weight the similarities to emphasize users (or items) that are most influential in establishing similarity, and, finally, compute a prediction that takes into account the users' (or items') ratings as well as their similarities.

User-Based Algorithms

For user-based CF systems, similarities between users are typically determined using the Pearson Correlation technique as described in (Resnick et al. 1994; Herlocker et al. 1999). Used in conjunction with Pearson Correlation, similarity weighting is used to rank similarities according to the number of co-rated items between two users; similarities calculated from user pairs with a large number of co-rated items will be ranked higher (i.e., given a higher weight) than similarities calculated from user pairs with a smaller number of co-rated items.

Two popular methods are used for prediction calculation: weighted prediction and mean-centered prediction.The weighted prediction method used is described in (Desrosiers and Karypis 2011) and ensures that the predicted ratings are within the allowable range, e.g., between 1.0 and 5.0. After similarities are calculated, the k most similar users that have rated the target item are selected as the neighborhood. After identifying a neighborhood, a prediction is computed for a target item i and target user u as follows:

$$p_{u,i} = \frac{\sum_{v \epsilon V} sim_{u,v} * r_{v,i}}{\sum_{v \epsilon V} \mid sim_{u,v} \mid} \quad (2)$$

where V is the set of k similar users and $r_{v,i}$ is the rating of those users who have rated item i, and $sim_{u,v}$ is the mean-adjusted Pearson correlation coefficient described above. Rating predictions calculated based on *zero or one* co-rated items are discarded as one co-rated item is insufficient to provide a reliable prediction.

The mean-centered prediction method, as documented in (Resnick et al. 1994; Herlocker et al. 1999; Desrosiers and Karypis 2011), is computed for a target item i and target user u as follows:

$$p_{u,i} = \overline{r_u} + \frac{\sum_{v \epsilon V} sim_{u,v}(r_{v,i} - \overline{r_v})}{\sum_{v \epsilon V} \mid sim_{u,v} \mid} \quad (3)$$

where V is the set of k similar users who have rated item i, $r_{v,i}$ is the rating of those users who have rated item i, $\overline{r_u}$ is the average rating for the target user u over all rated items, $\overline{r_v}$ is the average rating for user v over all co-rated items, and $sim_{u,v}$ is the mean-adjusted Pearson correlation coefficient described above. This technique is used to compensate for the fact that different users may use different rating values to quantify the same level of satisfaction for an item. Similarity threshold and kNN neighborhoods functionality were used in conjunction with this prediction method.

Item-Based Algorithms

For item-based CF systems, similarities between items are typically determined using the Adjusted Cosine Similarity technique as described in (Sarwar et al. 2001). Similar to Pearson Correlation, this method subtracts the corresponding user average from each co-rated pair to take into account the differences in rating scale between different users. Similarity weighting is also used to rank similarities.

Two popular methods are used for prediction calculation: weighted prediction and mean-centered prediction. The weighted prediction method used implements the classic approach described in (Sarwar et al. 2001) and ensures that the predicted ratings are within the allowable range, e.g., between 1.0 and 5.0. The prediction of item i for user u is made by computing the sum of the ratings given by user u on the items similar to item i. Each rating is then weighted by the corresponding similarity $s(i, j)$ between items i and j.

$$p_{u,i} = \frac{\sum_{j \epsilon allsimilaritems}(s_{i,j} * r_{u,j})}{\sum_{j \epsilon allsimilaritems}(\mid sim_{i,j} \mid)} \quad (4)$$

This method computes the prediction on an item i for a user u by computing the sum of the ratings given by the user on the items similar to i. Each rating is weighted by the corresponding similarity $s_{i,j}$ between items i and j. This approach captures how the active user rates the similar items. Also, rating predictions calculated based on *zero or one* co-rated items are discarded as one co-rated item is insufficient to provide a reliable prediction.

The mean-centered prediction method , as documented in (Desrosiers and Karypis 2011), is computed for a target item i and target user u as follows:

$$p_{u,i} = \overline{r_i} + \frac{\sum_{j \epsilon N_u(i)} sim_{i,j}(r_{u,j} - \overline{r_j})}{\sum_{j \epsilon N_u(i)} \mid sim_{i,j} \mid} \quad (5)$$

where $N_u(i)$ is the set of items rated by user u most similar to item i, $r_{u,j}$ is u's rating of item j, $\overline{r_j}$ is the average rating for item j over all users who rated item j, $\overline{r_i}$ is the average rating for target item i, and $sim_{i,j}$ is the mean-adjusted Pearson correlation coefficient described above.

Experimental Setup

We conducted two experiments to investigate tradeoffs in robustness and accuracy. The first, called the Static Analysis, considers the context of the entire dataset without regard for how the dataset of user profiles evolves over time; user-based and item-based CF algorithms are compared. The second, called the Dynamic Analysis, considers how the dataset evolves over time, i.e., we split the entire dataset temporally and analyze results at each of three time steps using an item-based CF algorithm. Both experiments employ the following common experimental setup.

Algorithms

The following recommender algorithms were used:

- User-based: Pearson Correlation similarity with similarity weighting, neighborhood formation with number of nearest neighbors ((kNN) set to 50 and similarity thresholding set to 0.0, significance weighting set to 50, mean-centered prediction, and weighted prediction.
- Item-based: Adjusted Cosine similarity with similarity weighting, neighborhood formation using similarity thresholding set to 0.0, significance weighting set to 50, mean-centered prediction, and weighted prediction.

Prediction Accuracy Metric For purposes of this study, the Mean Absolute Error (MAE) metric was used to measure the accuracy of the rating predictions. We employ the MAE accuracy evaluation mechanism provided by the underlying test harness, Mahout[1]. The training set was 70% of the data, the test set was 30% of the data, and 100% of the users were used.

Robustness Metrics For purposes of this study, Hit Ratio and Prediction Shift metrics (Burke, O'Mahony, and Hurley 2011) were used to measure the success of the attack (from the attacker's standpoint) such that a high Hit Ratio or a high Prediction Shift meant that the attack succeeded in changing the recommendations produced by the CF system.

Attack Profiles

Attack profiles were created and added to the non-attack datasets to simulate attacks on the recommender system. The attack profiles are similar to the non-attack user profiles. In this paper, the static and dynamic analyses determine whether a given CF algorithm is robust to attack given a best-case scenario from the attacker's standpoint, i.e., the emphasis was on creating an attack with target items that are easy to manipulate and an attack type that requires very little information about the underlying dataset. The characteristics of the attack profiles used in this study are as follows:

- Attack intent: Push, i.e., a single target item is selected and set to the maximum rating of 5.
- Attack type: Random, i.e, the non-target items in the attacker's profile are rated randomly from a normal distribution with mean 3.6 and standard deviation 1.1; these values correspond to the mean and standard deviation of the MovieLens 100K dataset used in this study. Although previous studies have shown that random attacks are not very effective compared to other attack types such as Average and Bandwagon (Lam and Riedl 2004; Mobasher et al. 2007), these attack types require near perfect information about the dataset, are not considered as realistic, and are difficult, if not impossible to obtain in the real world (Chirita, Nejdl, and Zamfir 2005). On the other hand, random attacks require very little information about the dataset and they can be made more effective with large filler sizes (number of randomly rated items in the attack profile) and/or large attack size (number of attack profiles).
- Attack Size: 5%, i.e., the number of attack profiles (attackers) added to the non-attack dataset and is a percentage of the total number of non-attack users. The 6-week dataset had 10 attack profiles added, the 18-week dataset

[1] http://www.mahout.apache.org

had 30 attack profiles added, and the 30-week dataset (ML100K) had 50 attack profiles added.

- Filler Size: 100%, i.e., the percentage of all non-target items that were rated according to the attack type. The 6-week dataset had 1,230 non-target items rated, the 18-week dataset had 1,548 non-target items rated, and the 30-week dataset (ML100K) had 1,663 non-target items rated.
- Target items: Four target items of differing genres were selected randomly from the 6-week dataset created in the dynamic analysis. These target items had low number of ratings and low ratings; these properties made the target items particularly susceptible to attack (Lam and Riedl 2004). Each set of attack profiles had one target item set to the maximum rating of 5 for a Push attack. The Average Hit Ratio and Average Prediction Shift metrics were computed over all target items. The same set of target items were used for all the attacks in this study.

Static Analysis of Robustness Tradeoffs

Our first experiment examined tradeoffs in accuracy and robustness in the context of the entire dataset, without regard for system usage over time, for both user-based and item-based CF algorithms.

Datasets

The data used in this study consisted of the MovieLens dataset downloaded from GroupLens[2] Research. Specifically, the 100K dataset with 99,693 unique ratings for 1,664 movies and 943 users (referred to as ML100K in this study). Ratings consist of integer values between 1 (did not like) to 5 (liked very much). User profiles consist of userid, itemid, rating, and timestamp. Four attack datasets were built using ML100K, one for each target item. Each attack dataset included non-attack user-profiles as well as attack profiles.

Testing

Accuracy (MAE) evaluation testing was done for each variation of prediction method (x2) and CF algorithm, user-based and item-based. Robustness calculations were executed once per target item (x4) per prediction method (x2) and CF algorithm, user-based and item-based.

Results and Discussion

Figure 1 shows the results of varying the prediction algorithm for user-based and item-based CF recommendations prior to any attack activity. User-based and item-based mean-centered predictions are both significantly better than weighted predictions, with item-based showing the more dramatic improvement.

Figures 2 and 3, however, show a different story. Robustness metrics for user-based CF recommendations are insensitive to prediction algorithm used while item-based CF recommendations using weighted prediction are significantly different than the recommendations provided by the mean-centered algorithm.

[2]http://www.grouplens.org

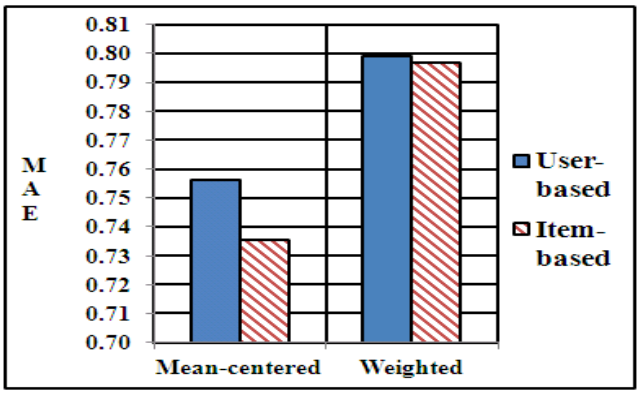

Figure 1: Mean Absolute Error for User-based and Item-based Recommendations – Before Attack

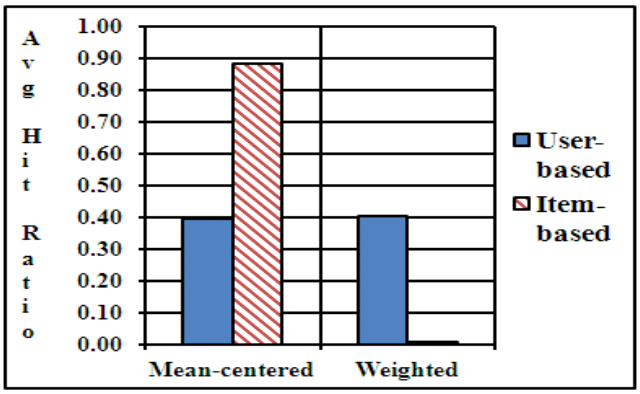

Figure 2: Average Hit Ratio for User-based and Item-based Recommendations – After Attack

The following observations can be made from these charts:

1. Based on just the MAE results, mean-centered prediction yields better prediction accuracy than weighted prediction for user-based and item-based CF recommendations.
2. Based on just the Hit Ratio and Prediction Shift results, weighted prediction yields the best protection, as compared to mean-centered predictions, against random attacks on item-based CF recommendation systems. Furthermore, using mean-centered prediction in an item-based CF recommendation system results in a recommender system that is susceptible to random attacks, even more so than user-based systems according to the Hit Ratio metric.
3. It appears that system operators implementing user-based CF recommenders should consider using mean-centered prediction in order to optimize accuracy (MAE) and robustness (Hit Ratio and Prediction Shift) rather than weighted prediction. Conversely, system operators implementing item-based CF recommenders need to consider the trade-off between a more accurate system that uses mean-centered prediction and a more robust system that

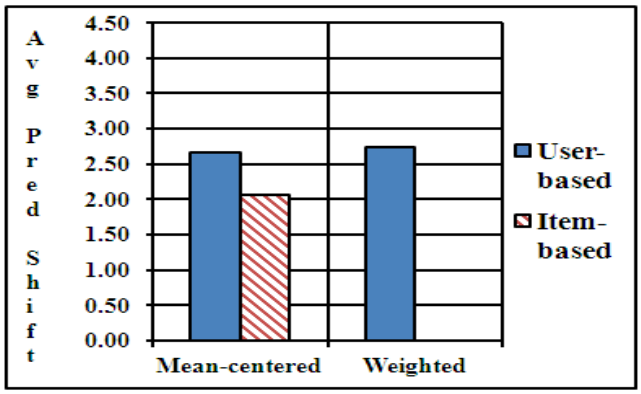

Figure 3: Average Prediction Shift for User-based and Item-based Recommendations – After Attack

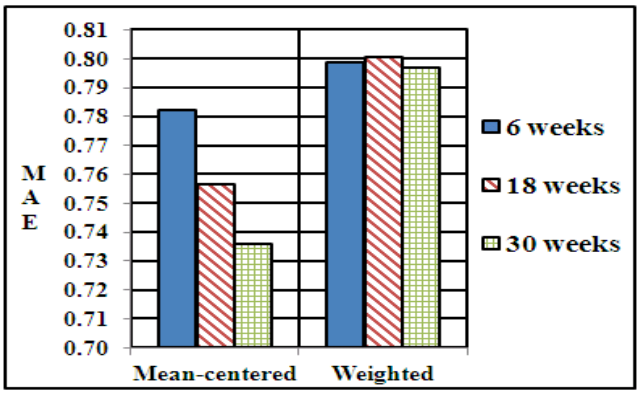

Figure 4: Mean Absolute Error for Temporal Item-based Recommendations – Before Attack

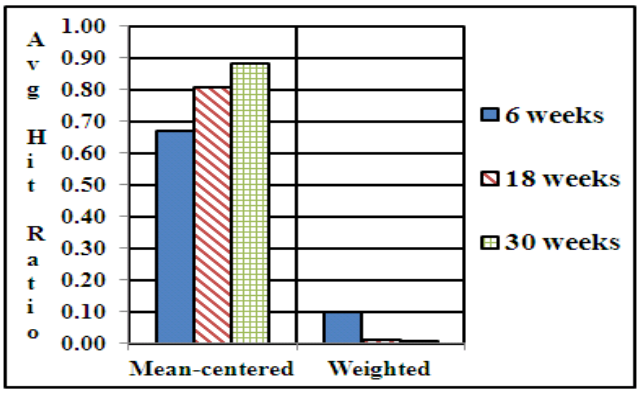

Figure 5: Average Hit Ratio for Temporal Item-based Recommendations – After Attack

uses weighted prediction.

Dynamic Analysis of Robustness Tradeoffs

Our second experiment examined tradeoffs in accuracy and robustness taking into account system usage over time, for the item-based CF algorithm.

Datasets

Three temporal non-attack datasets were derived from the full ML100K dataset and indicate that data were collected over a period of 30 weeks. The 6-week dataset has 17,081 ratings for 1,231 movies and 190 users, the 18-week dataset has 63,534 ratings for 1,549 movies and 621 users, and the 30-week dataset is the full ML100K dataset. Twelve attack datasets were built, one for each combination of target item (x4) and temporal dataset (x3). Each attack dataset included non-attack user-profiles as well as attack profiles. The 6-week attack datasets had 29,391 ratings for 1,231 movies and 200 users, the 18-week dataset had 110,004 ratings for 1,549 movies and 651 users, and the 30-week dataset had 182, 893 ratings for 1,664 movies and 943 users.

Testing

Test iterations (3 temporal non-attack datasets and 12 attack datasets): Accuracy (MAE) evaluation testing was done once per prediction method (x2) per dataset (x3) for the item-based CF algorithm. Robustness calculations were executed once per target item (x4) per prediction method (x2) per dataset (x3) for the item-based CF algorithm.

Results and Discussion

Figure 4 shows that mean-centered prediction outperforms weighted prediction in each time period. Also, mean-centered prediction improved as the dataset grew while the opposite was true for weighted prediction. Figures 5 and 6 show that Hit Ratio and Prediction Shift results using weighted prediction were dramatically different than the results produced using mean-centered prediction.

The following observations can be made from these results:

1. Based on just the MAE results, mean-centered prediction yields better prediction accuracy over time than weighted prediction for item-based CF recommendations.
2. Based on just the Hit Ratio and Prediction Shift results, weighted prediction yields the best protection, as compared to mean-centered predictions, against random attacks on item-based CF recommendation systems. Furthermore, using mean-centered prediction in an item-based CF recommendation system results in a recommender system that is susceptible to random attacks as the dataset grows.
3. System operators implementing item-based CF recommenders need to consider the trade-off between a more accurate system that uses mean-centered prediction and a more robust system that uses weighted prediction.

Conclusion and Future Work

Recommender system operators need to continue to weigh the costs and benefits of the underlying recommender system algorithms. In this study, we have shown that using mean-centered prediction, and holding other parts of the recommendation process constant, user-based CF recommen-

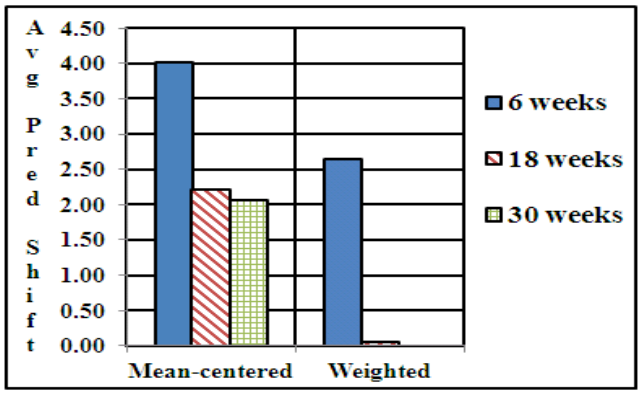

Figure 6: Average Prediction Shift for Temporal Item-based Recommendations – After Attack

dations achieves relatively good marks for both accuracy and robustness. We also show that item-based CF recommendation systems are (1) more robust to attack using weighted prediction albeit less accurate, and (2) less robust to attack using mean-centered prediction albeit more accurate. Furthermore, it was shown that, under certain conditions, item-based CF recommendations were more accurate and more robust to attack than user-based recommendations.

In the future, the temporal analysis will be extended to encompass user-based CF systems. Furthermore, in order to investigate the scalability of the results obtained in this study, we intend to perform similar analyses with larger datasets, such as MovieLens 1M, MovieLens 10M, or Netflix, comparing user-based and item-based CF systems as well as other algorithms including SVD and SlopeOne.

References

Adomavicius, G., and Tuzhilin, A. 2005. Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions. *Knowledge and Data Engineering, IEEE Transactions on* 17(6):734 – 749.

Burke, R.; Mobasher, B.; Williams, C.; and Bhaumik, R. 2006. Detecting profile injection attacks in collaborative recommender systems. In *E-Commerce Technology, 2006. The 8th IEEE International Conference on and Enterprise Computing, E-Commerce, and E-Services, The 3rd IEEE International Conference on*, 23 –23.

Burke, R.; O'Mahony, M. P.; and Hurley, N. J. 2011. Robust collaborative recommendation. In Ricci, F.; Rokach, L.; Shapira, B.; and Kantor, P. B., eds., *Recommender Systems Handbook*. Springer.

Burke, R. 2010. Evaluating the dynamic properties of recommendation algorithms. In *In Proceedings of the 4th ACM Conference on Recommender Systems (RecSys 2010)*.

Chirita, P.-A.; Nejdl, W.; and Zamfir, C. 2005. Preventing shilling attacks in online recommender systems. In *WIDM '05: Proceedings of the 7th annual ACM international workshop on Web information and data management*, 67–74. New York, NY, USA: ACM.

Desrosiers, C., and Karypis, G. 2011. A comprehensive survey of neighborhood-based recommendations methods. In Ricci, F.; Rokach, L.; Shapira, B.; and Kantor, P. B., eds., *Recommender Systems Handbook*. Springer.

Herlocker, J. L.; Konstan, J. A.; Borchers, A.; and Riedl, J. 1999. An algorithmic framework for performing collaborative filtering. In *Proceedings of the ACM SIGIR Conference*.

Herlocker, J. L.; Konstan, J. A.; Terveen, L. G.; and Riedl, J. 2004. Evaluating collaborative filtering recommender systems. *ACM Transactions on Information Systems* 22(1):5–53.

Koren, Y. 2009. Collaborative filtering with temporal dynamics. In *In Proceedings of the 15th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD-2009)*.

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

Lathia, N.; Hailes, S.; and Capra, L. 2009. Evaluating collaborative filtering over time. In *In Proceedings of the 32nd Annual ACM SIGIR Conference on Information Retrieval, SIGIR Workshop on the Future of IR Evaluation (SIGIR-2009)*.

Mehta, B., and Nejdl, W. 2008. Attack resistant collaborative filtering. In *SIGIR '08: Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, 75–82. New York, NY, USA: ACM.

Mobasher, B.; Burke, R.; Bhaumik, R.; and Williams, C. 2007. Toward trustworthy recommender systems: An analysis of attack models and algorithm robustness. *ACM Trans. Internet Technol.* 7(4):23.

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

O'Mahony, M. P.; Hurley, N.; and Silvestre, G. C. M. 2002. Promoting recommendations: An attack on collaborative filtering. In *DEXA '02: Proceedings of the 13th International Conference on Database and Expert Systems Applications*, 494–503. London, UK: Springer-Verlag.

O'Mahony, M. P.; Hurley, N.; and Silvestre, G. C. M. 2005. Recommender systems: Attack types and strategies. In *Proceedings of the 20st National Conference on Artificial Intelligence (AAAI-05)*, 334–339.

Resnick, P.; Iacovou, N.; Suchak, M.; Bergstrom, P.; and Riedl, J. 1994. GroupLens: an open architecture for collaborative filtering of netnews. In *Proceedings of the ACM CSCW Conference*.

Sarwar, B.; Karypis, G.; Konstan, J.; and Reidl, J. 2001. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the World Wide Web Conference*.

Shani, G., and Gunawardana, A. 2011. Evaluating recommendation systems. In Ricci, F.; Rokach, L.; Shapira, B.; and Kantor, P. B., eds., *Recommender Systems Handbook*. Springer.