Is Trust Robust? An Analysis of Trust-Based Recommendation

John O'Donovan & Barry Smyth
Adaptive Information Cluster
School of Computer Science and Informatics
University College Dublin
Belfield, Dublin 4
{john.odonovan, barry.smyth}@ucd.ie

1. ABSTRACT

Systems that adapt to input from users are susceptible to attacks from those same users. Recommender systems are common targets for such attacks since there are financial, political and many other motivations for influencing the promotion or demotion of recommendable items [2].

Recent research has shown that incorporating trust and reputation models into the recommendation process can have a positive impact on the accuracy and robustness of recommendations. In this paper we examine the effect of using five different trust models in the recommendation process on the robustness of collaborative filtering in an attack situation. In our analysis we also consider the quality and accuracy of recommendations. Our results caution that including trust models in recommendation can either reduce or increase prediction shift for an attacked item depending on the model-building process used, while highlighting approaches that appear to be more robust.

Categories and Subject Descriptors: H.3.3 Information Storage and Retrieval: Information Search and Retrieval.

General Terms: Algorithms, Human Factors, Reliability.

Keywords: Recommender systems, collaborative filtering, robustness, trust.

2. INTRODUCTION

The role of recommender systems in the online world has grown larger with the proliferation of the web. Recommender systems are a tool for helping users locate the right information when they need it [14]. These systems are becoming more noticeable in day to day e-commerce, helping users to locate products on web sites such as EBay and Amazon.com. In order to provide quality recommendations on items, these systems harness ideas from a broad range of AI research such as user-profiling, machine learning and information filtering. In recent years two distinct flavours of recommender systems have come to dominate- content-based and collaborative [13, 14]. There have been many different algorithmic variations and implementations of these techniques. Content-based recommenders operate by comparing descriptions of content for recommendable items, and hence are limited by their need for rich textual descriptions of content [14]. They can also suffer from a restriction which tends to limit recommendations in such a way that serendipitous recommendations are rare. This restriction occurs because of a narrowing of the recommendation space to items similar to those in the user's profile. Collaborative recommenders however are not subject to such problems since they operate by drawing on the rating histories of a set of user profiles which the system deems to have similar tastes to the target user who receives the recommendation. The collaborative filtering technique has been successful because it models the social process of looking to friends for recommendations on previously unseen items. It is not limited to making similar recommendations to those already liked and can offer surprising suggestions to the user.

This paper focuses on the collaborative approach to the recommendation task. In particular we looked at ways to improve the resistance of collaborative recommenders to malicious attacks. We propose several trust-based techniques for improving the robustness of collaborative recommenders in the face of these attacks. Following on from previous research in [10], which introduces trust models in collaborative filtering as a means to improve accuracy, we identify a security problem with such models in the face of attack. A problem, termed the *reinforcement problem* (outlined in Figure 3) manifests itself in terms of a greater prediction shift for an attacked item for the trust-based approach than with a benchmark collaborative filtering algorithm. The cause of this large prediction shift is that the attacking profiles can reinforce each other's trust values as the trust values are being built up during the early stages if those profiles have been present from the start. Lets say for instance that there were 10 attacking profiles in the system $p_1, ...p_{10}$. During the trust building process, the trust rating for user p_1 is calculated by allowing p_1 to generate predictions for the other users in the system. Included in this group are the remaining attacking profiles $p_2, ...p_{10}$. Assuming that the attacking profiles have similar rating trends, profiles $p_2, ...p_{10}$ will reinforce the predictions generated by p_1 yielding a higher trust value for that attacking profile. This means that in the final recommendation process, the opinions of the attacking profiles will actually carry *more* weight than the genuine profiles.

IUI'06, January 29–February 1, 2006, Sydney, Australia.

As a solution to this problem we propose to modify the manner in which we choose the profiles for which p_1 generates recommendations during trust building. By selecting the users which receive recommendations during the trust building process according to metrics such as the time a user profile has been in the system, the diversity and the reliability of the profiles for instance, we show that we can lower the prediction shift for an attacked item to less than that of the standard benchmark technique, with our best performing technique reducing the prediction shift for an attacked item by 75%.

From an end-user perspective, a change in their predicted rating due to a malicious attack does not necessarily imply that the user will be unhappy with the change. It is possible that a user may share the opinions of the attacking users. To analyse this possibility, we conduct a series of accuracy tests for the attacked item, and for the system as a whole. We find that in general, as the size of the attack approaches 15% of the profiles in the system the prediction shift caused by the attack profiles tends to make users *less* satisfied with the predicted rating.

3. RELATED WORK

We propose to use trust models to increase the robustness of collaborative filtering systems. There are two defined research areas especially relevant to this work. Firstly, research in the area of trust modelling in recommender systems, and secondly, research concerning attacks on recommender systems and methods of improving robustness of collaborative filtering algorithms in the face of such attacks. Now we present an overview of current, relevant research in these two categories.

3.1 Attacks & Robustness

Recommender systems are being deployed in increasing numbers of commercial applications. This has motivated recent research into the robustness and security of recommender systems. Work in [5] by Lam and Riedl examines four open questions concerning attacks on recommenders: what is the recommendation algorithm being used; how detectable is the attack and what are the properties of the attacked item. The study also looks at questions such as the attack intent, the targets, required knowledge and cost of attack on different collaborative filtering algorithms, for instance the user based algorithm [14] and the item-based algorithm [15]. They define a *shilling attack* as malicious user or set of users attempting to change the behaviour of the system to suit their own needs. Lam and Riedl conclude in [5] that current techniques for detecting and evaluating shilling attacks in recommender systems is in need of improvement, that shilling attacks are effective and easy to carry out, and that an attacker only needs a small amount of information about the system in order to perform a very successful attack. Lam and Riedl also point out in [5] that malicious attacks are non-trivial to detect with typical measures of recommender system performance.

Another comprehensive study on attacks in collaborative filtering was performed by Burke and Mobasher in [3] and [8]. This work is especially relevant to the consumer selection techniques introduced later in this paper. [8] outlines six attack types. For instance, a *sampling attack* in which the attack profiles are sampled from real user profiles in the database. A *bandwagon attack* which inserts extremely popular items into the attacking profiles in the hope of maximising overlap between the attacking and the genuine profiles. If the attacking profiles have rated the same items as other users then they will be used more frequently in the recommendation process, and potentially have their input weighted more heavily if the system uses a technique such as the standard Resnick prediction formula [14]. Another attack strategy outlined in [3] is termed the *favourite item attack*. This attack does not target the system as a whole, it is aimed only at one specific user. In this attack, it is assumed that the attacker has some prior knowledge of a user's tastes, and only puts one specific user's preferred items into the attacking profiles. In [8] and [3], Burke and Mobasher evaluate the effectiveness of each of the attack strategies on an item-based collaborative filtering algorithm according to two metrics. The difference in rating for an attacked item before and after an attack (called *prediction shift*) and the percentage of times a targeted item gets into the top-n recommendations for a target user. This is termed *hit ratio*. For the more common user-based collaborative filtering algorithm however, analysed in a similar manner by Burke in [2], there is only an evaluation of prediction shift for the targeted item. In our evaluation section we look at both prediction shift and accuracy for an attacked item as we feel that although an attacker may shift the predicted rating for an item, that shift may actually be in a direction which is favoured by the user receiving the recommendation.

Ongoing research by O'Mahony et al. [11] [12] also examines the robustness of collaborative filtering algorithms under attack. O'Mahony et al. [11] formalises two aspects to robustness in collaborative filters, recommendation *accuracy* which determines whether or not the products recommended after an attack are actually liked, and recommendation *stability* which examines whether the system recommends different items after an attack. Work in [11] also defines several types of attack. For instance, a *push* attack, where one particular item is targeted for promotion by the attacker and a *nuke* attack, where one item is targeted for demotion. In their evaluation [11] examine accuracy with respect to attack size using an absolute error metric. We adopt a similar metric for our evaluation of accuracy.

We propose a trust-based strategy for combatting the effects of attacks on collaborative filtering systems. We now look at some of the relevant research which deals with building trust and reputation models in recommenders, and with ways to use trust metrics to improve recommendations.

3.2 Trust Models

One approach that is potentially useful in dealing with attacks by malicious or untrustworthy users is to model user trust explicitly. Recent research has tackled a related issue by using more directly available trust relationships. Work by Massa et al. in [7] builds a trust model from explicit trust ratings given by users in the popular *Epinions.com* service. *Epinions.com* is a web site where users can review and evaluate a range of items. In this system users can also assign trust ratings to reviewers based on the degree to which they have been helpful and reliable in the past. Collaborative filtering systems rely heavily on rich rating data. They require good *overlap* between user profiles in order to provide good recommendations [13]. Massa et al. argue in [7] that trust data can be extracted and used as part of the recommendation process to relieve problems such as sparsity in the

ratings data, which reduces overlap between profiles. This is done by comparing users according to their degree of connectedness in a trust-graph encoded by *Epinions.com*. They show that many more users can be compared in this manner than with standard collaborative filtering techniques, say Pearson's similarity for example. The comparison between users is simply the distance between them in the trust graph in terms of the number of arcs connecting the users. This technique goes a long way towards tackling the sparsity problem, but it is not shown in [7] that recommendation accuracy is maintained.

Further research carried out by Massa et. al in [6] on the *Epinions.com* data introduces a trust-aware recommendation architecture which again relies on a web of trust for defining a value for how much a user can trust every other user in the system. In this system, predictive accuracy for users who have not rated many items is shown to be increased by using trust data. Again, the trust data is used to increase the amount profile overlap and therefore the number of comparable users. Massa's work shows that there is a trade-off situation between recommendation coverage (the ability of the system to make a recommendation) and recommendation accuracy. Their evaluation lacks a comparison with a standard collaborative filtering technique such as Resnick's algorithm [14]. Massa et al have also developed a trust aware skiing recommender system [1] which also compares users based on propagation of trust values.

Golbeck et al. [4] introduce an algorithm which generates locally calculated reputation ratings from a semantic web social network. This research is developed in an email application *TrustMail*. In this system, trust scores are calculated through inference and propagation of the form $(A \Rightarrow B \Rightarrow C) \Rightarrow (A \Rightarrow C)$, where A, B and C are users with interpersonal trust scores. When an email is received by the system, the sender is looked up in the trust network and an inline rating is provided for the email to denote its level of importance. In the *TrustMail* application, trust values can be defined at either a general level or a topic-specific level. This is similar to our definitions of *profile level* and *item level* trust later in this paper. One drawback with the system described in [4] is that users must explicitly enter trust information before the system can infer further ratings. There is also a security risk in that bad trust values malicious or otherwise will be propagated throughout the system and cause accuracy to drop. We identify a similar security risk in our *CItem* algorithm which we call the *reinforcement problem* later in this paper and we propose a solution.

4. A MODEL OF TRUST

In this work we present details of five different approaches to computing trust. This work follows from the initial research in [10] carried out by the authors on trust modelling in recommendation. In [10] we presented a detailed analysis of the trust modelling process, and provided a mathematical explanation of each model. Here we will present a short overview our model as an understanding of this is critical to the material we present thereafter.

Figure 1 is an overview of this trust building process. We define two types of profiles, those who receive recommendations (*consumer* profiles) and those who produce recommendations (*producer* profiles). Collaborative filtering usually relies on the inputs of many producer profiles to arrive

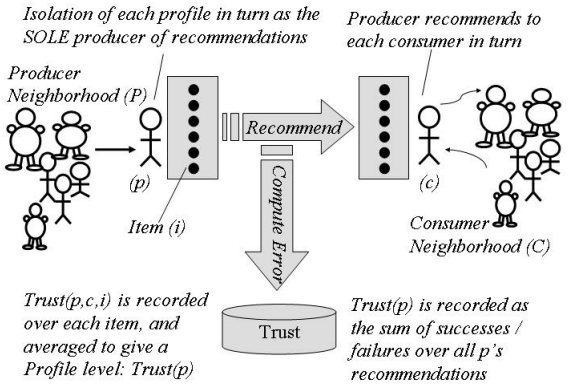

Figure 1: Calculation of Trust Scores from Rating Data

at a recommendation to be presented to a particular consumer, this is generally done by combining their individual contributions according to some suitable function such as Resnick's formula, shown in Equation 1. In this formula, the predicted rating $c(i)$ of some item i for a consumer c is calculated as the weighted average of a set $p(i)$ of ratings for i from a similar set of profiles P_i. This approach to collaborative filtering has been widely adopted as a benchmark. In Resnick's formula, the prediction for item i in consumer profile c is a combination of ratings from the set of producer profiles P, weighted according to a similarity function. In many cases, this function is Pearson's correlation coefficient, denoted by $sim(c, p)$ in Equation 1. In this function, the values $\overline{c}$ and $\overline{p}$ represent the mean ratings for consumer c and producer p respectively. [10] argues that profile similarity is but one of a number of possible metrics to be considered when choosing recommendation partners for collaborative filtering. Trust is forwarded as another metric by which to consider the contributions from potential recommendation partners.

Our goal is to model the trust of *individual* users in the system, and on a finer grain, to assess the trust or reputation that a the user has for the recommendation of a particular item; for example "Arnold's reputation for recommending the movie Terminator". To achieve this, we build our model of trust by analysing the rating histories of users in the system, not just from a similarity aspect, but by examining the predictive accuracy of each producer profile individually. This approach follows from the intuition that if a user has a history of providing good recommendations in the past, chances are that his recommendations will be good in the future.

$$c(i) = \overline{c} + \frac{\sum_{p \epsilon P(i)} (p(i) - \overline{p}) sim(c, p)}{\sum_{p \epsilon P_i} |sim(c, p)|} \quad (1)$$

4.1 Mining Trust Values

Figure 1 shows the pairwise alignment of each of each producer profile p and consumer profile. The producer p serves as the sole recommendation partner for a consumer profile c.

In our model-building process, each profile in the neighborhood temporarily serves as the producer profile, and during this time each remaining profile temporarily serves as the consumer profile. We accept a predicted rating by producer p on item i for some consumer c as correct if that rating falls within an minimum error bound ϵ of the actual rating that c has for item i. This correctness is given by Equation 2. Now we can define two recommendation sets, the full set of recommendations produced by profile p, which is shown in Equation 3, and the set of correct recommendations which p has generated as the sole producer profile for item i. This set is shown in Equation 4.

$$Correct(i, p, c) \Leftrightarrow |p(i) - c(i)| < \epsilon \tag{2}$$

$$RecSet(p) = \{(c_1, i_1), ..., (c_n, i_n)\} \tag{3}$$

$$CorrSet(p) = \{(c_k, i_k) \in RecSet(p) : Correct(i_k, p, c_k)\} \tag{4}$$

[10] presents two distinct trust metrics, *profile-level* trust, which is the average trust for a producer profile p across all of the items that p has generated recommendations for (given by Equation 5), and a more fine grained *item-level* trust, which is the mean trust score for over every recommendation p has generated for a given item i. It was shown in [10] that item-level trust produced better accuracy results when integrated into the recommendation process. For this reason, we only employ an item-level trust metric in our evaluation section. The formula for calculation of item-level trust is shown in 6. This formula computes the percentage success for recommendations generated by producer p on some item i.

$$Trust^P(p) = \frac{|CorrSet(p)|}{|RecSet(p)|} \tag{5}$$

$$Trust^I(p, i) = \frac{|\{(c_k, i_k) \in CorrSet(p) : i_k = i\}|}{|\{(c_k, i_k) \in RecSet(p) : i_k = i\}|} \tag{6}$$

4.2 The *CITEM* Algorithm

We have shown how trust scores are calculated for individual producers of recommendations. The next step is to put these values to work in the recommendation process. It has been shown in [4, 7, 10] that trust values can be utilised in a variety of ways during recommendation. For our evaluations, we will use the *CITEM* algorithm introduced in [10]. This approach is a combination of *trust-based weighting* of producer profile ratings and *trust-based filtering* of producer profile ratings using trust values at the item level. The weighting approach simply adds the extra metric of trust to the standard similarity weighting in Resnick's prediction formula. This modified version of Resnick's formula is shown in Equation 7 to include the trust weighting. Here, $w(c, p, i)$ is the simple harmonic mean between the item level trust for producer p and item i, and the similarity between producer p and the profile receiving the recommendation, ie: the consumer profile c. The formula for calculating the simple harmonic mean is shown in Equation 8. Figure 2 illustrates the weighted producer profiles as larger members of the producer neighborhood. In this figure, there are more

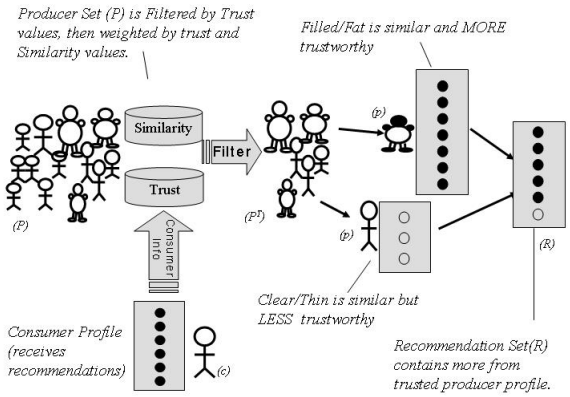

Figure 2: Integrating trust models into standard collaborative filtering.

shaded items in the final recommendation set since these have come from producer profiles with high trust values.

$$c(i) = \overline{c} + \frac{\sum_{p \epsilon P(i)} (p(i) - \overline{p}) w(c, p, i)}{\sum_{p \epsilon P(i)} |w(c, p, i)|} \tag{7}$$

$$w(c, p, i) = \frac{2(sim(c, p))(trust^I(p, i))}{sim(c, p) + trust^I(p, i)} \tag{8}$$

Trust-based filtering simply filters out the producer profiles that form a consumer's neighborhood according to the trust scores of the producer profiles. If a producer profile has a trust value which is less than the threshold T, then it's inputs are not considered in the recommendation process for a particular item. The filtering process is illustrated in Figure 2, where the original producer neighborhood P gets reduced to neighborhood P^T containing only trusted profiles.

Combining these two approaches of filtering and weighting gives us the *CItem* algorithm, which is given in Equation 9. In this formula, P^T is the set of trusted producer profiles.

$$c(i) = \overline{c} + \frac{\sum_{p \epsilon P^T(i)} (p(i) - \overline{p}) w(c, p, i)}{\sum_{p \epsilon P^T(i)} |w(c, p, i)|} \tag{9}$$

4.3 The Reinforcement Problem

Including trust metrics in the recommendation process can have a positive effect on predictive accuracy in collaborative filtering [10]. The issue of security against attack in recommenders is becoming increasingly relevant, giving cause for an assessment of the performance of our trust-based recommendation techniques in the face of malicious attack. An obvious flaw was noted in the basic technique, which is illustrated in Figure 3. This is a graph of the prediction shift for a "pushed" item with varying attack sizes. It is clear that the *CItem* approach has been skewed much more than the benchmark Resnick algorithm by the false

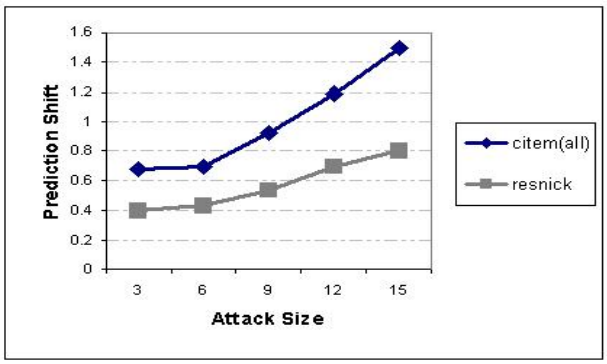

Figure 3: The reinforcement problem with trust-based recommendation.

attack profiles, all of which give approximately average ratings for random items, and then a maximum rating for the pushed item "Toy Story".

The reason for this large prediction shift is intuitive. Using the trust approach we have described, say for example that 10 malicious profiles ($p_{a1}, ..., p_{a10}$) find their way into the system, all with a maximum rating for the pushed item and possibly with other similar ratings too. In this situation, when we are building our trust scores, each attack profile will reinforce the ratings of each other profile. For example, p_{a1} will generate recommendations (as the sole producer profile) for his nine counterpart attack profiles $p_{a2}, ..., p_{a10}$, and these recommendations will have zero error since all the attack profiles have given the same rating for the pushed item. This means that the attacking profiles actually get *higher* trust scores than regular profiles.

Without an explicit knowledge of which profiles are malicious, it is impossible to completely block the effects of attackers in collaborative filtering [11]. We can however increase the system's resistance to attack. In order to achieve this, and to tackle the reinforcement problem in our *CItem* algorithm, we propose that modifying the selection of the set of consumer profiles C during the trust-building process can reduce the effect of push attacks in the recommender system. This is achieved by minimising the chances of an attack profile serving as a consumer profile.

4.4 Consumer Selection Strategies

To overcome the problem of reinforcement of attack profiles in the trust modelling process, we filter the profiles to be included in the trust building process. Intuitively if we do not allow the attack profiles to serve as consumers in the trust building process depicted in Figure 1, then they cannot reinforce the ratings of the remaining attacking profiles. In a real world system however, this is not a trivial task. The performance of any recommendation algorithm will be effected by the nature of the attacking profiles. For example, a push attack where the attacking profiles are all recently entered in the system could be completely stopped by simply selecting older profiles as consumers in the trust-building process. Unfortunately we do not always have the luxury of this information however, so here we explain five different consumer selection strategies and analyse in our evaluation the effects that attacks of varying size have on the predicted rating for a pushed item, and on the predictive accuracy of the recommender using each of the selection strategies. In the list below each strategy consists of a set of 100 consumer profiles, apart from *CItem(all)* which contains all 754 profiles involved in the trust building process.

1. *CItem(all)* - This is the basic *CItem* approach described in [10] which is susceptible to attack in terms of a prediction shift for a pushed item. In this selection strategy, all of the profiles in the training data are allowed to temporarily serve as consumer profiles.

2. *CItem(diverse)* - This approach selects a diverse set of consumer profiles from the training data during the model building stage. In this approach we follow the diversity formula for selection of profiles based on work in [16]

3. *CItem(time)* - In this strategy, we simply select older profiles to form the consumer set C

4. *CItem(random)* - Selection of the consumer set C randomly across all of the training profiles.

5. *CItem(genuine)* - This is the "ideal" protection against malicious profiles. In an ideal situation we could simply remove the malicious profiles from the data altogether. In our experiments however, we are interested in examining the robustness of the trust based approach to recommendation, so we leave the malicious profiles in the data but do not allow *any* of them serve as consumer profiles during the trust modelling stage.

5. EVALUATION

Work in [9] argues that trust models can be mined from ratings data in collaborative filtering systems and that these models can be used to improve the predictive accuracy of collaborative filtering. In this paper we have highlighted an important robustness issue with the trust based models presented in [10] in which the system can be highly susceptible to attack. We have proposed a solution to this problem by modifying the way we select our consumer profiles during the trust building process. In our evaluation we use the same dataset and experimental setup as in [10]. We describe three experiments to assess the prediction shift for an attacked item, the change in recommendation accuracy for an attacked item and to evaluate the mean predictive accuracy of each of our consumer selection strategies under normal (non-attack) operation of the *CItem* algorithm. In each of our experiments, we also make comparisons to a standard Resnick collaborative filtering algorithm.

5.1 Experimental Setup

For this evaluation we use the MovieLens dataset [14] which contains 943 profiles of movie ratings. The rating scale for this dataset ranges from 1 to 5, with a rating of 5 indicating a "liked" item. Profile sizes vary from 18 to 706 with an average size of 105. We divide these profiles into two groups: 80% are used as the producer profiles for trust building and the remaining 20% are used as the consumer (test) profiles.

5.2 Building the Trust Models

In a deployed system, the trust models which we have described can be built on the fly. Whenever a user rates

an item, the system can allow producer profiles to generate predictions for that real rating, and build trust scores based on the accuracy of those predictions. For our experiments however we implement a "snapshot" of such a system in which the trust models are computed offline.

We build five different models of trust, each varying based on the consumer selection strategies described in the previous section. For each of the selection strategies we run a standard *leave-one-out* training session over all of the producer profiles. Accordingly, each profile temporarily becomes the consumer, and the remaining producer profiles generate predictions for that consumer. These predictions are generated by using Resnick's prediction formula, but with each producer profile serving as the sole recommendation partner in the prediction process. This recommendation partner is shown as p in Figure 1. Once a prediction is made by a producer profile, we then analyse the distance of the predicted rating from the known rating of the consumer. If this distance is within a threshold of the actual rating, we increment the trust score for that producer profile on the recommended item. In this manner we model trust for each producer of recommendations on an *item-level*. To find the *profile-level* trust for a producer we would simply average the item level trust across all of the items in the profile. For these experiments however, we are only interested the *CItem* algorithm which uses trust scores at the item level.

In the *CItem(all)* model, we allow every profile to serve as a consumer during the training session. For each of the other strategies, *CItem(time)*, *CItem(diverse)*, *CItem(genuine)* and *CItem(random)* we select a subset of 100 profiles to serve as consumers, which is just over 13% of the training data.

5.3 Generating Attack Profiles

In the following two experiments, we attack the prediction algorithms by inserting false profiles to attempt to shill the predictions in a specific direction. The attack strategy we implement is a *push attack* as described in [11]. The goal of the attacking profiles is to create a favourable prediction shift for the target item "Toy Story", which has a mean rating of 3.87 and a set of 452 raters. We distributed ratings in the attack profiles according to the *average attack* strategy outlined by Burke in [2]. Attack profiles are assigned average ratings for a random set of items and then a maximum rating for the "pushed" item.

Attack size has important implications for e-commerce applications. When it comes to understanding the cost of an attack, for example, if an author must purchase many copies of his book, and possibly some other books from Amazon.com in order to build a valid attack profile, it will be very expensive to have any great effect on the products which get recommended. If the *cost of attack* is greater than the *rewards*, then there will be little point in launching an attack in the first place. Bearing this in mind, we analyse prediction shifts for each of the models with a varying attack size.

5.4 Prediction Shift

In this experiment we analyse prediction shift for the *CItem* algorithm while varying the size of the attacks from 0 to 15% of the data in intervals of 3%. The results of this experiment are presented in Figure 4. In this graph, the prediction shift (ΔR_i) is measured from zero. For instance, if an item rating was 2.5 before the attack, and 3 afterwards, then the shift would be 0.5. We can clearly see the susceptibility of the *CItem(all)* approach to the reinforcement problem described earlier. This manifests itself in terms of a prediction shift that is actually twice that of the Resnick algorithm for an attack size of 15%. It may seem counter-intuitive that there can be a prediction shift greater than 1.23 (5 - 3.87). This occurs because the graph indicates an average shift.

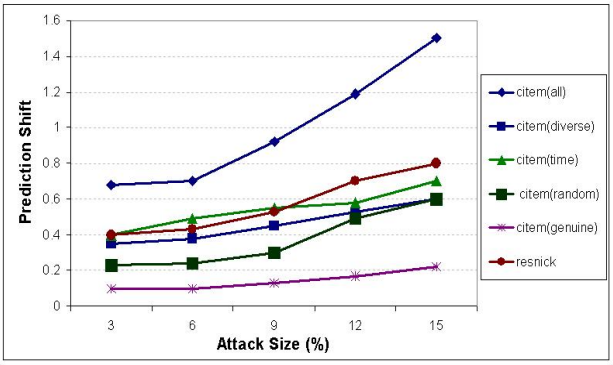

Figure 4: The average prediction shift for pushed item "toy story" (compared to Resnick) of each of the trust-based recommendation strategies, with varying attack sizes.

However, Figure 4 clearly shows that the other consumer selection strategies for the *CItem* algorithm dramatically lower the prediction shift for the attacked item, in most cases to a lower amount than for the Resnick algorithm. Our *CItem(genuine)* model gives us the lowest prediction shift. For an attack size of 15%, this technique boasts a prediction shift of 0.2 in comparison with a 0.8 shift for the Resnick algorithm, giving a 75% improvement. We define this algorithm to be our ideal since in a large scale deployed system, identifying a suitable set of truly genuine profiles is not a trivial task. The results for the random, diverse and time models are within a distance of 0.2 from each other and the Resnick algorithm across all of the attack sizes. However, as the attack size tends towards 15%, the time, diverse and random models tend to shift much less than the Resnick technique, both *CItem(random)* and *CItem(diverse)* have a prediction shift of 0.2 lower than the Resnick technique for an attack size of 15%. The random technique actually does well in this experiment, considering the benefit of ease of implementation.

Although there is a notable reduction in the prediction shifts for the attacked item using different consumer selection strategies, we must consider what the implications of these techniques are for the user of the recommender system. To assess this we perform two experiments to evaluate the predictive accuracy of the *CItem* algorithm using each consumer selection strategy. Firstly, we examine the predictive accuracy on a single attacked item, and then we measure the mean predictive accuracy across all of the items in the test set without the attacking profiles. The latter experiment is similar to the accuracy test in [10], but using the new selection strategies.

5.5 Recommendation Accuracy

To analyse the predictive accuracy of our recommendation techniques for the attacked item "Toy Story", we evaluate the mean recommendation error generated by each tech-

nique for the items in our consumer profiles. For each consumer profile we temporarily remove each rated item and allow the producer profiles to generate recommendations for the removed item. We record the error as the absolute distance between the predicted rating and the consumer's actual rating for that item. We record this information for each of the *CItem* models, using three attack sizes: 3, 9 and 15%. The results of this accuracy experiment are presented in Figure 5. It is immediately clear from these results that all of the recommendation techniques take a big predictive accuracy hit as the attack size increases, The most effected being the *CItem(all)* technique. This result is to be expected since if we look back to the prediction shift results we can see that the *CItem(all)* technique has a shift of around 1.5 for an attack size of 15%. If we consider that the mean rating for the attacked item is 3.87, and 3.52 for the whole dataset then for many profiles we can speculate that while the size of the push attack approaches 0.35 in the prediction shift graph we should be seeing a reduction in mean error compared with the Resnick algorithm. The accuracy should then drop as the predicted ratings are skewed towards the max rating of 5. By this intuition we should be seeing a greater drop in the prediction errors for the *CItem(genuine)* and *CItem(diverse)* with attack sizes of 3 and 6, since the average prediction shift for these in Figure 4 is less than 0.35. We do not see this clear drop in absolute error from the graph in Figure 5 which leads us to consider one possibility that the attack for this experiment may have caused a larger prediction shift than that in Figure 4, possibly due to the fact that the distribution of ratings for "Toy Story" has large deviations lower than the mean rating, and that these profiles were used in the evaluation. These accuracy results do tell us that when attack profiles get into the recommendation process, they cause serious problems. We need to develop more intelligent ways for the system to select profiles with which to build trust models. In our discussion section we introduce some avenues of research to assess this problem further.

However, the most accurate algorithm, although marginally, is still the *CItem(genuine)* which beats all of the others including the Resnick algorithm across all of the attack sizes. For an attack size of 6, our consumer selection techniques seem to prevent an increase in error when compared to the benchmark Resnick technique. At this attack level, our *CItem(genuine)* algorithm has a mean absolute error of 0.7, which is a 33% improvement on the benchmark. This result shows us that trust models can in fact preserve recommendation accuracy. It is clear that more research is needed to discover how to build these models for optimal performance.

5.6 Predictive Accuracy for a Non-Attack Situation

To make a comparison between the recommendation strategies we have described in [10] and the consumer selection strategies in this paper we perform a mean absolute error analysis of the predictive accuracy of each *CItem* technique in a non-attack situation. In this experiment accuracy is recorded in the same manner as for the previous experiment, but in this case we record the difference between predicted ratings and actual ratings across all of the items in the test data. Once again, for each of the *CItem* techniques we choose a set of 100 consumer profiles for the trust building session. The error metric in Equation 2 is set at 1.8 to give a good distribution of trust values. The results are

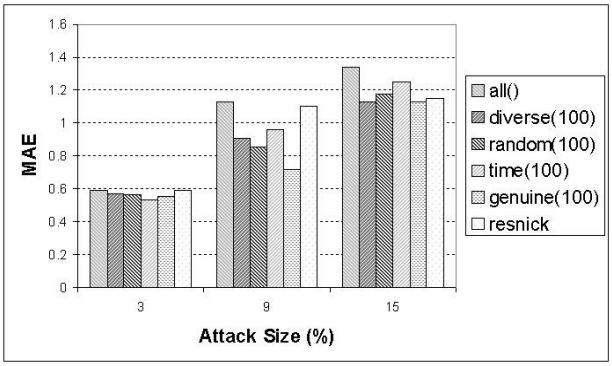

Figure 5: The average prediction error (compared to Resnick) for attacked item "Toy Story" *CItem* algorithm using different consumer selection strategies and varying attack sizes.

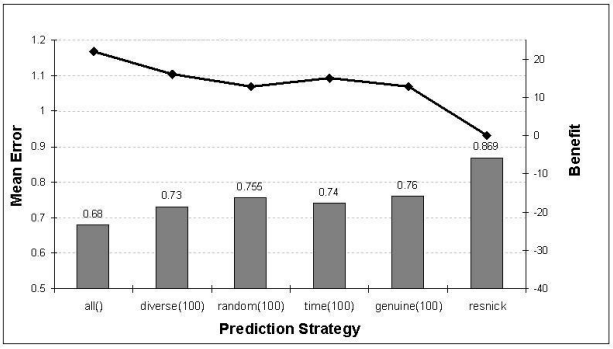

Figure 6: The average prediction error and relative benefit (compared to Resnick) of each of the *CItem* algorithm using different consumer selection strategies.

presented in Figure 6. The black line indicates the relative benefit each prediction strategy shows over the benchmark Resnick algorithm. We can clearly see that every flavour of the *CItem* algorithm has a lower mean absolute error than the benchmark. As with the findings in [10], the *CItem(all)* technique improves on the benchmark by 22%. Interestingly, the benefit in accuracy is reduced across each of the new techniques by around 0.1, compared with a 7-fold reduction in the amount of consumer profiles used in the trust building session. The trust building technique has a computational complexity of $O(n^2)$ on the number of users in the system, so this reduction means that the model building process is 7 times faster for a reduction of 0.1 in accuracy.

6. DISCUSSION

In this paper we examined the robustness of trust based recommendation using different trust models, and an average attack strategy. It would be interesting to carry out a further examination of our trust based algorithms under different attack scenarios. For instance in the simple case of a sudden attack on a collaborative recommender in which all of the attacking profiles are recent, our *CItem(time)* strategy should perform as well as the *CItem(genuine)* technique since none of the attack profiles would be selected as consumers in the model building process. Another interesting

question arises from our analysis of prediction shift: would the prediction shifts be similar for a *nuke* attack in which an item is targeted for *demotion* by the attacking profiles?

In many cases there are similarities and patterns in attacking profiles, for example in the push attack described in this paper each attacking profile contained a rating of 5 for the pushed item "Toy Story". With the possible exception of the *diverse* and the *time* approaches to consumer selection which we have proposed, there is not much "intelligence" involved in the selection process as it stands. In all of the attacks described by O'Mahony in [11] and Burke in [3, 2] there are clear patterns in the attacking profiles, such as consistent average ratings in the case of the *average attack* and similarity to one profile in the case of a *favourite item* attack. We propose an examination of each common attack strategy to highlight their salient features so we can minimise the chances that an attack profile will be used in the consumer selection process. For this examination we will use pattern matching algorithms (possibly the apriori rule generating algorithm described in [13]) and other machine learning techniques to attempt to identify the attacking profiles.

7. CONCLUSIONS

Research into trust modelling in collaborative filtering is no doubt still in its infancy, we believe however that as recommender systems proliferate on the web, the issue of who to trust and who to avoid will become vitally important to receiving good quality, reliable recommendations. In this paper we have argued that conventional collaborative filtering techniques are susceptible to attack from malicious users. We have shown also that the trust based collaborative filtering algorithms presented in [10] can be even more susceptible since attacking profiles can actually reinforce each other's opinions during the trust building process. We have proposed a solution to this problem by modifying the manner in which we choose consumer profiles during the trust building process. We have shown four new approaches to the trust building process and shown in our evaluation that the prediction shift for an attacked item can be reduced by modifying the selection of consumer profiles for trust building. Our best technique provides a 75% reduction in prediction shift when compared with the benchmark Resnick algorithm and 87.5% when compared with the *CItem(all)* algorithm presented in [10]. We have also presented results of two experiments to evaluate the predictive accuracy of the new techniques in both attack and non-attack situations and shown that our best strategy consistently beats the benchmark Resnick algorithm in MAE tests.

8. ACKNOWLEDGMENTS

This material is based on works supported by Science Foundation Ireland under Grant No. 03/IN.3/I361

9. REFERENCES

[1] P. Avesani, P. Massa, and R. Tiella. Moleskiing: a trust-aware decentralized recommender system. *1st Workshop on Friend of a Friend, Social Networking and the Semantic Web. Galway, Ireland*, 2004.

[2] R. Burke, B. Mobasher, and R. Bhaumik. Limited knowledge shilling attacks in collaborative filtering systems. In *Nineteenth International Joint Conference on Artificial Intelligence, IJCAI-05*, Edinburgh, Scotland., 2005. IJCAI Inc.

[3] R. Burke, B. Mobasher, R. Zabicki, and R. Bhaumik. Identifying attack models for secure recommendation. In *Beyond Personalisation Workshop at the International Conferece on Intelligent User Interfaces*, pages 347–361, San Deigo, USA., 2005. ACM Press.

[4] J. Golbeck and J. Hendler. Accuracy of metrics for inferring trust and reputation in semantic web-based social networks. In *Proceedings of EKAW'04*, pages LNAI 2416, p. 278 ff., 2004.

[5] S. K. Lam and J. Riedl. Shilling recommender systems for fun and profit. In *Proceedings of the 13th international conference on World Wide Web*, pages 393–402. ACM Press, 2004.

[6] P. Massa and P. Avesani. Trust-aware collaborative filtering for recommender systems. *Proceedings of International Conference on Cooperative Information Systems, Agia Napa, Cyprus*, 2004.

[7] P. Massa and B. Bhattacharjee. Using trust in recommender systems: an experimental analysis. *Proceedings of 2nd International Conference on Trust Managment, Oxford, England*, pages 221–235, 2004.

[8] B. Mobasher, R. Burke, R. Bhaumik, and C. Williams. Effictive attack models for shilling item-based collaborative filtering systems. In *WebKDD,*, Chicago, Illinois, USA., 2005. ACM Press.

[9] J. O'Donovan and B. Smyth. Eliciting trust values from recommendation errors. In *Proceedings of the 18th International FLAIRS Conference*, pages 289–294. AAAI Press, 2005.

[10] J. O'Donovan and B. Smyth. Trust in recommender systems. In *Proceedings of the 10th International Conference on Intelligent User Interfaces*, pages 167–174. ACM Press, 2005.

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

[12] M. P. O'Mahony, N. Hurley, and G. C. M. Silvestre. An attack on collaborative filtering. In *Proceedings of the 13th International Conference on Database and Expert Systems Applications*, pages 494–503. Springer-Verlag, 2002.

[13] D. O'Sullivan, D. C. Wilson, and B. Smyth. Improving case-based recommendation: A collaborative filtering approach. In *Proceedings of the Sixth European Conference on Case Based Reasoning.*, pages LNAI 2416, p. 278 ff., 2002.

[14] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl. Grouplens: An open architecture for collaborative filtering of netnews. In *Proceedings of ACM CSCW'94 Conference on Computer-Supported Cooperative Work*, Sharing Information and Creating Meaning, pages 175–186, 1994.

[15] B. M. Sarwar, G. Karypis, J. A. Konstan, and J. Reidl. Item-based collaborative filtering recommendation algorithms. In *World Wide Web*, pages 285–295, 2001.

[16] B. Smyth and P. McClave. Similarity vs. diversity. In *ICCBR '01: Proceedings of the 4th International Conference on Case-Based Reasoning*, pages 347–361, London, UK, 2001. Springer-Verlag.