Trust-Based Recommendation: an Empirical Analysis

Daire O'Doherty
Universite catholique de Louvain
1348 Louvain-La-Neuve, Belgium
daire.odoherty@student.uclouvain.be

Salim Jouili
EURA NOVA
Rue Emile Francqui, 4
1435 Mont-Saint-Guibert, Belgium
salim.jouili@euranova.eu

Peter Van Roy
Universite catholique de Louvain
1348 Louvain-La-Neuve, Belgium
peter.vanroy@uclouvain.be

ABSTRACT

The use of trust in recommender systems has been shown to improve the accuracy of rating predictions, especially in the case where a user's rating significantly differs from the average. Different techniques have been used to incorporate trust into recommender systems, each showing encouraging results. However, the lack of trust information available in public datasets has limited the empirical analysis of these techniques and trust-based recommendation in general, with most analysis limited a single dataset.

In this paper, we provide a more complete empirical analysis of trust-based recommendation. By making use of a method that infers trust between users in a social graph, we are able to apply trust-based recommendation techniques to three separate datasets. From this, we measure the overall accuracy of each technique in terms of the Mean Absolute Error (MAE), the Root Mean Square Error (RMSE) as well as measuring the prediction coverage of each technique. We thus provide a comparison and analysis of each technique on all three datasets.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval—*information filtering, Selection process, Retrieval Models*; I.5.1 [**Computing Methodologies**]: Pattern recognition—*Models*

General Terms

Algorithms

Keywords

Recommender Systems, Trust-Based Recommendation, Social networks

1. INTRODUCTION

In the context of recommender systems, the emergence of trust [23, 21, 5, 15, 22] as a key link between users in social networks is a growing area of research, and has given rise to a new form of recommender system, that which incorporates trust information existing between users into the recommendation process. The main concept behind this is the sociological idea that users are more inclined to have similar opinions and tastes to people they know and trust. Trust-enhanced recommender systems thus refine the classical recommendation techniques, by making use of trust relationships between users in a network. These connections generally take the form of weighted trust assertions between users, indicating how much one user trusts another. By using such a Web of Trust, a number of techniques have been proposed in the literature that have been shown to perform significantly better than traditional recommendation techniques, in terms of overall accuracy and coverage of rating predictions, particularly where a user's rating of an item differs significantly from the average rating for this item.

Previous studies have provided empirical comparisons of different techniques of incorporating trust into recommendation algorithms. However, due to the unavailability of suitable datasets, which provide trust information, such analysis has usually been restricted to just a single dataset.

In this paper, we provide a further empirical analysis and compare the main techniques of incorporating trust into recommendation algorithms to traditional collaborative filtering algorithms by performing these techniques on three separate datasets. To do this, we use a trust inference technique developed as part of our previous work [14] which uses only structural information. The advantage of this technique is that it is generic and suitable for application on any number of social networks based primarily on a bipartite graph that describes a set of users connected to a set of items, regardless of the content or purpose of the dataset. Using this method to automatically infer trust relations between actor pairs in a network allows us to apply trust-based recommendation techniques to a number of different datasets. This allows us to (1) show the applicability of trust to a wider range of domains than has previously been explored and (2) verify if the previous conclusions related to the improvement of recommendation by means of trust will hold or not.

2. STATE-OF-THE-ART

The area of trust-based recommender systems has been the object of extensive study for the past years. Indeed, trust has been shown to provide significant improvements to classical collaborative filtering techniques. The main difference between most of these trust-enhanced methods is the acquisition of trust values between actor pairs. Many algorithms have been developed to acquire trust in social graphs, of these we present a few.

In [6], Golbeck introduces the TidalTrust algorithm to estimate

trust values between actor pairs in a social network. This algorithm uses trust values that are explicitly provided by the users of the network, and trust estimates are thus propagated through the network using a modified breadth-first search. The trust estimate from user u to user v is thus the weighted average of the trust scores attributed to v by users which u has already trusted, $u's$ neighbours. u thus issues a request for a trust estimate to all its neighbours, and if the neighbour has a direct trust value for v, they return this value, if not they forward this trust request to all of their neighbours in turn. Each trust value that is returned to u by a neighbour n is then weighted according to the trust value that u has attributed to n, this same weighting process occurs when neighbours receive answers to forwards requests.

Massa et al [12, 13] developed a very similar approach to that of Golbeck called MoleTrust, which also incorporates explicit user trust assertions and propagate trust through the network in two phases. Firstly all cycles in the graph are removed, thus turning it to a directed acyclic graph and then similar to that used by TidalTrust, trust values are propagated to a source user u for another user v using a trust-based weighted mean of the trust values attributed to v by the trusted neighbours of u. As well as this, MoleTrust incorporates an extra parameter into the propagation phase called the *trust propagation horizon*. This parameter specifies the maximum hop distance of the graph from the source user u for which trust is propagated. This parameter is used for two reasons, firstly to reduce computational cost, but also, as they say, because the reliability of the propagated trust decreases with every new trust propagation hop.

O'Donovon and Smith [17] developed a method to automatically generate trust between users based on ratings history between the two users. The resulting trust metric can be seen as a reliability metric between two users. In this work, the authors distinguish two types of trust, item-level and profile-level. Trust is then built up between users u and v, by measuring the reliability of $v's$ past recommendations for u, which is seen as the percentage of predicted ratings v has made for u which have been within a certain threshold of $u's$ actual rating for the recommended item. This trust is then seen in two levels, as a general reliability score for v dubbed the *profile level* or at a finer grained *item level* which measures reliability of v to recommend item i.

Wang et al [20] develop a method to generate trust between users based on their common *tastes*. By first grouping items into different clusters, the authors use a frequency measure of the number of ratings each user asserts to different groups of items, and thus build up a personalized taste set for user and infer trust based on the common taste sets between users.

All of these methods generate trust estimates between users, and use these trust relationships in various collaborative filtering techniques to generate personalized predictions of items to users. Each study has shown that the incorporation of trust into recommendation techniques improves the accuracy of recommender systems in comparison to basic collaborative filtering, especially where the user's rating for an item differs from the average rating for this item.

An empirical comparison of some of the more used methods of incorporating trust into collaborative filtering techniques was made by Victor et al in [19]. This analysis compares a number of different trust-enhanced recommendation techniques in terms of prediction coverage, and average error rate. These techniques were applied to a set of controversial reviews from the Epinions.com dataset, for which they develop an algorithm to identify controversial reviews based on the level of disagreement between the ratings in this set. In similar work, Victor et al [18] also compared the different algorithms for generating and propagating trust values (TidalTrust, MoleTrust and O'Donovan et al). Again, this comparison was based on the application of these methods to controversial items in the network. However, this comparison was performed uniquely on the Epinions.com dataset.

In this paper, we provide a more complete empirical analysis of trust-based recommendation. By making use of a method to infer trust between users in a social graph, we succeed in applying trust-based recommendation to three different data sets instead of limiting the analysis to only the Epinions.com dataset. In fact, to the best of our knowledge, this is the first work that provides an empirical analysis of trust-based recommendation by means of a trust inference method.

3. RECOMMENDATION TECHNIQUES

In this section we present the standard and trust-enhanced recommendation techniques analysed in this paper.

3.1 Standard Techniques

In this section we present two standard collaborative filtering (CF) techniques, used in many recommender systems. To generate a recommendation for an item i to user u two of these algorithms use a similarity measure between u and a user v who has already rated i to weight the rating that v gave to i in the computation. This similarity measure is generally given by Pearson's Correlation Coefficient (PCC).

3.1.1 Simple mean

$$r(u,i) = \frac{1}{|R|} \sum_{v}^{v \in R} r_{v,i} \tag{1}$$

Equation 1 shows a simple average of all ratings for item i, where $|R|$ is the total number of raters for i. This method is still used in some recommender systems.

3.1.2 Pearson weighted mean

$$r(u,i) = \frac{\sum_{v}^{v \in R^+} w_{u,v} r_{v,i}}{\sum_{v}^{v \in R^+} w_{u,v}} \tag{2}$$

Equation 2 shows the standard *weighted sum* algorithm for rating prediction. Like the simple mean formula (Eq. 1), this formula uses the the average of the ratings of other users for item i, but personalizes the prediction for a target user u by weighting the ratings according to how similar the rating user v is to the target user u. This weighting, $w_{u,v}$, is usually calculated using PCC. As stated in [19], in practice only the ratings of users with a positive correlation to u are considered in the prediction, this is represented in Eq. 2 by the set R^+.

3.1.3 Pearson collaborative filtering

$$r(u,i) = \bar{r_u} + \frac{\sum_{v}^{v \in R^+} w_{u,v} (r_{v,i} - \bar{r_v})}{\sum_{v}^{v \in R^+} w_{u,v}} \tag{3}$$

Equation 3 shows Resnick's classic CF formula [16], where $w_{u,v}$ represents the weight of similarity between users u and v. This similarity weight is again usually calculated by using PCC. A prediction for an item i for user u is based on the mean rating $\bar{r_u}$ of user u and the sum of the ratings of other users for item i. As described in [1], this algorithm is a form of *adjusted* weighted sum, which takes into account the fact that different users may use the ratings scale in different ways, by using the deviations of a user's rating for item i from the user's average rating overall. These ratings are thus weighted using the results of the similarity measure $w_{u,v}$ calculated using PCC.

3.2 Trust-enhanced Techniques

In this section we present the trust-enhanced recommendation techniques used in this paper. Generally, these algorithms are enhancements of the standard techniques, and others have been developed to follow the idea that users are more inclined to appreciate recommendations from their trusted peers.

3.2.1 Trust-based weighted mean

$$r(u,i) = \frac{\sum_v^{v \in R^T} t_{u,v} r_{v,i}}{\sum_v^{v \in R^T} t_{u,v}} \tag{4}$$

Equation 4 shows an enhanced version of the Weighted Mean formula shown in Eq. 2. This formula uses the trust value present between users u and v as a weight for the ratings of other users in place of the similarity measure. This is the technique used by the TidalTrust algorithm [6] presented above to incorporate their trust metrics into the recommendation process.

3.2.2 Trust-based collaborative filtering

$$r(u,i) = \bar{r_u} + \frac{\sum_v^{v \in R^T} t_{u,v}(r_{v,i} - \bar{r_v})}{\sum_v^{v \in R^T} t_{u,v}} \tag{5}$$

Equation 5 shows a refined version of Resnick's CF formula which incorporates trust. In this method, the similarity weight attributed to ratings by user v for user u, calculated using PCC is replaced with the value of trust $t_{u,v}$ present between u and v. This is technique used by the MoleTrust algorithm [10] to incorporate their trust metrics into the recommendation process.

3.2.3 Trust-filtered mean

$$r(u,i) = \frac{1}{|R^T|} \sum_v^{v \in R^T} r_{v,i} \tag{6}$$

Equation 6 shows a trust filtering method, whereby the raters of item i are filtered according to their trust values, where only the raters who are trusted above a certain threshold are used in the computation of predicting a rating. Using these raters, we then take a simple average of their ratings for item i. This method provides results according to the idea that "users are more likely to accept recommendations from their most trusted friends". We use this method in our comparison to evaluate the applicability of these claims.

3.2.4 Trust-filtered collaborative filtering

$$r(u,i) = \bar{r_u} + \frac{\sum_v^{v \in R^{T+}} w_{u,v}(r_{v,i} - \bar{r_v})}{\sum_v^{v \in R^{T+}} w_{u,v}} \tag{7}$$

Equation 7 shows Resnick's CF technique refined to include the trust value $t_{u,v}$ as a further filtering mechanism to choose only the item raters who are trusted above a certain threshold. In this formula, the weight $w_{u,v}$ is still used in the weighting process of the rating values, and is calculated using PCC. This technique is that used by O'Donovan and Smith in [17].

3.2.5 Ensemble Trust

$$\begin{aligned} r(u,i) = \bar{r_u} &+ \frac{\sum_v^{v \in R^T} t_{u,v}(r_{v,i} - \bar{r_v})}{\sum_v^{v \in R^T} t_{u,v} + \sum_v^{v \in R^+ \setminus R^T} w_{u,v}} \\ &+ \frac{\sum_v^{v \in R+ \setminus R^T} w_{u,v}(r_{v,i} - \bar{r_v})}{\sum_v^{v \in R^T} t_{u,v} + \sum_v^{v \in R+ \setminus R^T} wu,v} \end{aligned} \tag{8}$$

Finally, Eq. 8 shows the *Ensemble Trust* CF technique, as proposed by Victor et al [19]. In this work, the authors explain that this technique aims to take into account all possible ways to obtain a positive weight for a rater of an item while favoring trust over similarity. This method thus aims to increase the percentage of predictions made by the recommender system, which they call the *coverage*, a term also used in this work.

4. INFERRING TRUST RELATIONSHIPS

In order to apply trust-enhanced recommendation techniques, we needed to obtain trust information between actor pairs in each dataset. As both the MovieLens and LibimSeTi datasets do not provide such information a prioi, we use a trust inference formula developed in our previous work [14] to automatically infer trust relationships between users in each of the datasets used.

4.1 A Method for Trust Inference

In previous work [14], we develop a method to infer trust between actor pairs in social networks based on the structural features of the bipartite graph typical of most social networks. Typically, social networks correspond to a bipartite graph representing a set of actors (e.g. users) connected to a set of items (e.g. books). Each connection corresponds to an act through which an actor performs an operation on an item (rating, buying, commenting...). Formally, let $G = (A \cup I, E)$ be a bipartite graph where A and I are two disjoint sets, the set of $actor$ and the set of $item$ vertices respectively, and $E \subseteq A \times I$ is the set of edges (i.e. interactions between actors and items). The difference with a classical graph lies in the fact that edges only exist between actor vertices and item vertices. Using only the structural information of the graph, we infer trust between users by taking into account two key structural factors:

1. Firstly, by using the *Jaccard Index* to compute a distance measure between vertices in set A. As we are dealing with a bipartite graph, each vertex does not have a direct connection to any vertex in the same set. We thus consider the set of vertices $S \in A$, through which vertex u has an indirect connection in the graph through the vertices in set I for which u has a directed edge. We define this as the *two-hop neighbourhood* of u represented by N_u below, connecting u to vertices in the same set, through $u's$ interaction with vertices in set I.

$$J(u,v) = \frac{|N_u \cap N_v|}{|N_u \cup N_v|} \tag{9}$$

2. Secondly, we compute a distance measure between vertices in set A in relation to the *popularity* of the vertices in set I for which they both have a directed edge (i.e. their *shared items*), represented by the indegree of the shared item i ($deg(i)$ in equation 10. This distance measure follows the intuition that the popularity of an item in a social graph is inversely related to the level of trust between actor pairs who have rated it.

$$D(i) = (\frac{2}{1 + e^{(-deg(i)^{\sigma} + 2^{\sigma})}} - 1) \quad (10)$$

By combining these two aspects, our trust inference formula is presented as a whole in Eq. 11.

$$Trust(u, v) = \alpha J(u, v) + \beta(1 - \frac{\sum_{i}^{i \in SI} D(i)}{|SI|}) \quad (11)$$

where SI is the set of shared items between the users, and $\alpha + \beta = 1$. The constants α and β are used to weight each structural aspect of the trust inference formula (Eq. 11), according to their contribution to the computation of trust between users. The parameter α defines the contribution of the distance measure shown in Eq. 9, and parameter β defines the contribution of the distance measure shown in Eq. 10.

As explained in our previous work, the parameter $0 < \sigma < 1$ is incorporated into the equation for the second structural factor in order to provide a way to modify the distribution of the values computed by $D(i)$ between $[0, 1]$. In other words, this parameter is used to define from which degree value an item is considered as popular. The value for this parameter depends entirely on the connectivity of the items in the dataset, as items become more and more connected. As the dataset evolves, the value of σ can be adjusted to reflect this.

4.1.1 Choosing Parameter Values

For the purposes of this empirical analysis, we wanted to be able to get a clear comparison of each recommendation technique, and how they perform on each dataset. For this, we decided to use the same values for the parameters: α, β and σ and not to tailor the values of the parameters specifically to any of the datasets to find values would be most applicable to each specific dataset, but instead to use the same set of values that would return comparable results for all.

Table 4.1.1 shows the values of α, β, and σ chosen for the remainder of this paper. We remark here that the structural aspect of Eq. 10 weighted by the parameter β, has the most important contribution to the computation of trust between actor pairs.

The choice of these values was based on the results of serious of tests further to the training phase presented in [14], whereby the trust and distrust prediction accuracy of the formula was measured for possible combinations of values for the parameters α and β. Considering the high connectivity properties of the datasets used, as will be discussed later, the value of 0.2 was chosen for parameter σ so as not to overly punish the items in the well connected MovieLens dataset, but at the same time taking into consideration the sparseness and much lower connectivity properties of the Epinions dataset.

5. EXPERIMENTS

5.1 Datasets

α	β	σ
0.4	0.6	0.2

Table 1: Selected parameters

The purpose of this work is to provide an analysis and comparison of the recommendation techniques presented in section 3, on a number of different datasets. To this end, we have selected 3 publicly available datasets previously used for the study of recommendation systems. Each dataset differs in their intended purpose and properties, such as the connectivity and size of the graph, as well as the sparsity and distribution of ratings. These datasets thus allow a good analysis of the performance and behavior of the different techniques in different circumstances.

5.1.1 MovieLens

The MovieLens dataset [16] is widely used for the study of recommender systems, and was publicly distributed by the GroupLens Research group (http://www.grouplens.org/). MovieLens is an online social network where users can rate movies. The dataset used is a commonly used version of the network and is available directly from the GroupLens website. It contains 6040 users, 3900 different movies, and $1,000,209$ ratings. Movie ratings are on a discrete scale from 1 to 5 stars, and this version of the dataset is well connected, with each user in the set having at least 20 ratings encoded.

5.1.2 LibimSeTi

LibimSeTi (http://www.libimseti.cz/) is an on-line dating service, where users are able to store personal profiles, as well as rate the profiles of other users. The dataset [2] used contains $17,359,346$ anonymous ratings, on a discrete scale from 1 to 10 , with $168,791$ profiles made by $135,359$ LibimSeTi users. From this dataset, we used $20,000$ randomly chosen profiles, as will be explained.

5.1.3 Epinions

The epinions dataset used was that available from trustlet.org. Epinions.com is an on-line social network where users contribute reviews and share their opinions on any number of items or topics, from books and DVDs to holidays and restaurants. Users can also provide ratings on a scale of 1 to 5 for these items. In addition to this rating system, Epinions also provides a "*Web Of Trust*" facility, whereby users can explicitly provide "*trust*" assertions, indicating their individual trust for other users in the network, based on the quality of reviews provided. These ratings, as well as the web of trust service are used to provide recommendations for item reviews deemed to be most applicable to individual users. The availability of this explicit trust information has made this dataset very popular in the study of trust-enhanced recommendation systems. However, for the purposes of this comparison, we do not use this explicit trust information as we are interested in providing an equal comparison of trust in recommender algorithms across all datasets, and thus we only use the ratings information to automatically generate a web of trust, in the same manner as the other datasets.

5.2 Performance Measures

To measure the performance of each of the recommendation algorithms for the purpose of comparison, we follow the common technique of hiding a certain number of ratings from the graph, and applying each algorithm in turn to predict the value of this rating. From these predictions, we measure the accuracy of each algorithm according to two commonly used accuracy measures in the field of recommender systems [8].

Firstly, we use the Mean Absolute Error (MAE) as shown in Equation 12, where N is the total number of ratings to predict, $P_{u,i}$ is the predicted value of the rating of item i by user u and $R_{u,i}$ is the real value of the rating of item i from user u. The MAE measures the average absolute deviation between the predicted rating $P_{u,i}$ of the algorithm and the user's true rating $R_{u,i}$.

Secondly, we use the Root Mean Square Error (RMSE) shown in Equation 13. We use the RMSE as well as the MAE as it gives us a broader view of the performance of each algorithm. As discussed in [8, 13], while using only the MAE, each error in prediction is treated with equal value, whereas with the RMSE, larger errors in prediction are given relatively more weight, as the difference between the predicted rating $P_{u,i}$ from the real rating $R_{u,i}$ is squared. This means that the RMSE punishes predicted values that deviate further from the real rating value. As a result, lower RMSE values indicate that an algorithm is more accurate in the sense that its predicted values do not deviate very far from the true rating.

$$MAE = \frac{\sum_{i=1}^{N} |P_{u,i} - R_{u,i}|}{N} \tag{12}$$

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N} (R_{u,i} - P_{u,i})^2}{N}} \tag{13}$$

As well as these accuracy measures, we follow the analysis of performance as discussed in [8, 19, 20, 13], and measure the *coverage* of the algorithms. The coverage is simply the percentage of the hidden user to item ratings for which the algorithm was able to produce some sort of prediction. We compute this by taking the number of ratings for which the algorithm was able to make a prediction and divided it by the total number of hidden ratings that were to be predicted, as shown in Equation 14, where P is the total number of predictions that were made by the algorithm, and N is the total number of user to item ratings that were requested of the algorithm to predict. The coverage of predictions has been shown to be an important measure of performance for recommendation algorithms, where some techniques have been shown to be unable to produce predictions for a significant percentage of ratings. This can happen for example with Resnick's collaborative filtering formula (equation 3) when there are no users who have rated an item i who have a positive Pearson correlation with the target user u.

$$Coverage = \frac{P}{N} \tag{14}$$

5.3 Validation Process

Before we began, we extracted a sample subset of $20,000$ randomly chosen users from both the LibimSeTi and the Epinions dataset. For the MovieLens dataset, with just over $6,000$ users we used the entire dataset. In previous studies [6, 11, 19, 17], the performance of recommender systems is typically assessed using a *leave-one-out* method, which entails hiding each rating in the graph one by one, and thus predicting the value of this rating using the recommendation algorithm. This process is then repeated a certain number of times to build up a measure of accuracy of the algorithm. However, in this work we wish to measure the performance of each algorithm when there are different percentages of ratings missing and thus less information available. In order to do this, and to compare each of the different recommendation techniques on each of the data samples, we followed a standard *k-fold* cross validation process, which we will explain in this section. The process described in this section was repeated in an identical fashion for all three datasets.

To begin, we first split the data samples into randomly chosen subsets of $1,000$ users. For each subset, we then randomly removed a certain percentage of the ratings from the graph, which were then used as the ratings to be predicted by each recommendation algorithm. Once we had removed these ratings, we then applied our trust inference formula to the remaining subset, as detailed in section 4. Doing this thus generated a new trust graph computed by our formula, linking users of the subset together by means of weighted symmetric trust links. Following this, each recommendation algorithm described in section 3 was then applied to the subset to predict the ratings that had previously been removed. The trust-enhanced algorithms thus used the trust information contained in the computed trust graph in the process of computing their predictions. By comparing the resulting predicted ratings of each algorithm to the corresponding actual rating we were able to compute the accuracy and coverage measures as detailed in section 5.2, and measure the performance of each recommendation algorithm for each subset. The final performance results of each algorithm presented in Tables 3, 4, and 2 are thus the mean of the results computed for each subset. This process was then repeated 5 times for 10%, 20%, 30%, and 50% of the ratings randomly removed.

6. ANALYSIS & DISCUSSION

In this section we present the results of the experiments performed on the selected datasets, then give an overview of the results before going in depth for each dataset. Tables 2, 3, and 4 report the performance results for each algorithm performed on the MovieLens, Epinions, and LibimSeTi datasets respectively. Each table presents the results (MAE, RMSE and Coverage) achieved by the tested algorithms, performed with 10%, 20%, 30% and 50% of the ratings hidden.

6.1 Overview

Overall, we remark that the trust-based techniques, on average, outperform their standard counterparts for all three datasets. This follows the results of previous studies [4, 11, 19] and thus adds further weight to the idea that trust can enhance recommendation algorithms. In terms of the accuracy, we also remark that the performance of each algorithm differs depending on the dataset on which it is used.

For example, if we consider the figures for 10% of ratings hidden in the results for the Epinions dataset shown in Table 3, we can see that the algorithms using a weighted mean strategy, Trust-based WM (Eq. 4) and Pearson WM (Eq. 2) with MAE of 0.919, and 0.951 respectively slightly outperform the techniques based on Resnick's formula: Trust CF (Eq. 5), and Pearson CF (Eq. 3) with MAE of 0.927 and 0.966 respectively. However, the opposite is true for the LibimSeTi dataset (Table 4) and the MovieLens dataset (Table 2) where the Resnick-based techniques significantly outperform those using a weighted mean, demonstrated by a superiority of 0.56 of the Trust CF algorithm over the Trust WM algorithm in the MovieLens dataset with 10% of the ratings hidden. One possible cause of this swing in fortunes could be the different connectivity properties and the distribution of the item ratings of each of the datasets. As discussed by Massa in [11] using the same Epinions dataset, the vast majority of ratings (74%) in the Epinions dataset are given a weight of either 4 or 5 stars. Of these, 45% are given the highest rating of 5, meaning that there is very little variance in rating values. As the Resnick-based algorithms use the variance of a user's rating for an item to the user's average rating as part of their computations, such lack in variance may indeed be hindering these algorithms. On the other hand, as discussed earlier, the MovieLens

dataset is highly connected with users having rated at least 20 items each, and as shown in [3] these ratings are follow a normal distribution with a significant degree of variance. This variance thus allows the Resnick-based algorithms to use more information and outperform those using only a weighted mean.

6.1.1 Trust Filtered Mean

Considering the Trust Filtered Mean algorithm (Eq. 6), which takes a simple average of all of the ratings given trusted users, and thus filtering out all other ratings, the results are not intuitively straightforward. Indeed, for the results performed on the MovieLens dataset, the Trust Filtering Mean algorithm performs identically to the Simple Mean algorithm (Eq. 1) in terms of accuracy, with a slightly worse RMSE performance for some of the tests.

A possible explanation for such similarity in scores could be the high connectivity of the items in the MovieLens dataset, meaning that even when the algorithm filters out all untrusted users who have rated an item there is still a relatively high number of raters left, and an average of their rating values might still give a result very close to the average rating as computed by the simple mean algorithm. This doesn't exactly follow the original motivation laid out for this technique, that users would be more likely to appreciate recommendations from those whom they trust. Furthermore, for the Epinions dataset, the Trust Filtered Mean actually performs worse than the Simple Mean, performing quite similar accuracy results with only 10% of ratings hidden but falling to an inferiority of 0.1 for 50% of the ratings hidden. This may be attributed to both the sparsity of the Epinions dataset, meaning that the algorithm has less information to be able to infer trust between users, as well as this, due to the lack of variance in the ratings, it seems that filtering only the trusted information can still not outperform a simple mean. Again, even though it performs the second best out of all of the other algorithms it still doesn't quite fit our methodology.

However, for the LibimSeTi dataset, we can see that the trust filtered mean algorithm consistently outperforms the simple mean in terms of both the MAE and the RMSE performance measures, and increases this superiority as the number of ratings that are hidden increases, with a superiority of up to 0.37 for the MAE with 50% of the ratings hidden. This increased performance may be attributed to the distribution of the ratings within the LibimSeTi dataset. As discussed in [3], this LibimSeTi dataset contains a large proportion of extreme ratings, with a large percentage of ratings in the dataset having either the lowest possible value of 1, or the highest possible value of 10. As will be discussed, previous studies [4, 12] have shown that with such difference in opinion between the ratings of items (i.e controversial items), trust can be used to provide a more personalized recommendation to the individual users. From these results, we can see that this seems to hold true for this dataset, and indeed seems to follow the motivation of the trust filtered mean algorithm, whereby only using the ratings information provided by trusted users provides better accuracy results when a user's rating for an item differs significantly from the average rating of this item.

6.1.2 Trust Filtered Collaborative Filtering

One surprise result in terms of accuracy was that of the Trust-Filtered collaborative filtering technique (Eq. 7) proposed in [17]. In comparison to the other algorithms, this technique performed quite poorly in the experiments, and in fact, was consistently the worst performer in terms of MAE. This result goes against previous intuition that users are more inclined to appreciate more recommendations from their trusted neighbours.

One explanation for this performance could be related to the way in which the technique filters users. From the set of users with a positive Pearson correlation (PCC), this technique also filters non-trusted users. Such extreme filtering constrains the amount of users that can be used in the recommendation process and may indeed eliminate genuinely trusted users from the recommendation process who do not have a positive correlation to the target. This consequently constrains the amount of information that the recommendation algorithm can use to make a prediction, thus limiting predication accuracy.

However, if we take a closer look at the RMSE results for this technique, we can see that it actually outperforms a number of the other techniques. From this, we may deduce that although the technique gives the correct prediction less often than the other techniques, the error in prediction is never too high, as the RMSE punishes large errors in prediction.

6.1.3 Coverage

In terms of coverage, we can also see that the trust-enhanced techniques vastly improve the coverage of each of the recommendation algorithms, particularly when the dataset itself is very sparse, as is the case with the Epinions dataset. But as well as this, in the case of the more connected MovieLens and LibimSeTi datasets, we notice that when the percentage of hidden ratings is increased the coverage of the standard collaborative filtering techniques using PCC drops considerably, whereas the rate of coverage of the trust-enhanced techniques stays reasonably stable, and drops at a much slower rate. Taking an example from the LibimSeTi dataset, when 50% of the ratings are removed, the trust-enhanced techniques outperform the standard techniques in terms of coverage by up to 20%, with the trust-based algorithms having a coverage rate of 96.69% while the standard techniques having a coverage rate of only 76.32%. A possible explanation for this increase in accuracy in this case could be the potential of our trust inference formula to find trusted neighbours beyond those whom a user shares an item. As discussed in section 4, our formula uses two structural features to infer trust information, and does not only concentrate on shared items. The use of the *two-hop neighbourhood* in this formula allows our formula to infer trust between users who have not yet rated one of the same items. Bearing in mind that PCC is primarily based on shared items, this means that the set of trusted users for a user is more likely to be larger than that of the set of users for which PCC returns a positive result, especially when 50% of the ratings have been hidden.

We can see that this difference in coverage remains the same for the Epinions dataset with an improvement of almost 40% when 50% of the ratings are removed, but it does not hold for the MovieLens dataset, where the trust-based techniques only deliver around 0.8% improvement. This suggests that as both the LibimSeTi dataset and more particularly the Epinions dataset are more sparse than the highly connected MovieLens dataset, the likelihood is higher that the increased removal of ratings will remove shared items between users. When considering the fact, as discussed in [11] that more than 50% of the users in the Epinions dataset have provided less than 5 ratings this become more evident. However, with each user having a minimum of 20 rated items in the MovieLens dataset this is less likely to be the case, as users have rated much more items and thus are more likely to have more shared items.

Apart from the Simple Mean algorithm (Eq. 1), we remark that Ensemble Trust algorithm (Eq. 8) proposed by Victor et al in [19]

	10% hidden			20% hidden			30% hidden			50% hidden		
Algorithm	MAE	RMSE	Cov %	MAE	RMSE	Cov %	MAE	RMSE	Cov %	MAE	RMSE	Cov %
Pearson CF	0.707	0.99	98.74	0.714	0.9988	98.08	0.721	1.008	97.50	0.741	1.030	96.08
Pearson WM	0.764	1.041	98.74	0.769	1.049	98.08	0.775	1.055	97.50	0.792	1.078	96.08
Simple Mean	0.755	1.024	100	0.756	1.027	100	0.756	1.028	100	0.762	1.034	100
Trust CF	0.700	0.980	99.96	0.702	0.982	99.96	0.704	0.985	99.95	0.712	0.993	99.93
Trust Filtered Mean	0.755	1.025	99.96	0.756	1.027	99.96	0.756	1.029	99.95	0.762	1.035	99.93
Trust Filtered CF	0.78	0.993	98.10	0.790	1.004	97.25	0.795	1.011	96.48	0.813	1.033	94.62
Trust WM	0.756	1.026	99.96	0.757	1.029	99.96	0.757	1.030	99.95	0.764	1.037	99.93
Ensemble Trust	0.700	0.979	99.99	0.702	0.982	99.99	0.704	0.984	99.99	0.712	0.992	99.97

Table 2: Results for MovieLens

	10% hidden			20% hidden			30% hidden			50% hidden		
Algorithm	MAE	RMSE	Cov %	MAE	RMSE	Cov %	MAE	RMSE	Cov %	MAE	RMSE	Cov %
Pearson CF	0.966	1.356	49.17	0.991	1.391	41.74	1.012	1.413	34.69	1.048	1.462	20.94
Pearson WM	0.951	1.367	49.17	0.971	1.399	41.74	0.988	1.420	34.69	1.015	1.465	20.94
Simple Mean	0.898	1.29	96.99	0.904	1.303	93.88	0.923	1.328	89.73	0.935	1.349	79.87
Trust CF	0.927	1.308	83.37	0.946	1.329	79.28	0.965	1.356	73.76	1.004	1.403	59.86
Trust Filtered Mean	0.898	1.297	83.37	0.911	1.308	79.28	0.924	1.331	73.76	0.945	1.361	59.86
Trust Filtered CF	1.029	1.360	48.47	1.0503	1.394	40.80	1.074	1.4215	32.9	1.110	1.470	18.8
Trust WM	0.919	1.323	83.37	0.941	1.345	79.28	0.952	1.368	73.76	0.981	1.41	59.86
Ensemble Trust	0.927	1.308	83.37	0.946	1.33	79.28	0.965	1.356	0.737	1.004	1.403	59.86

Table 3: Results for Epinions

leads to the best coverage performance providing almost optimal coverage in every dataset regardless of the number of ratings hidden, this again follows the observations made by the authors themselves. As stated in their work [19], the motivation for this algorithm was to account all possible ways to obtain a positive weight for a user to include them in the recommendation process, while favoring a trust weight over a similarity weight computed by PCC. It is worth noting also here, that again this benefit in coverage does not come at the expense of accuracy, with the Ensemble Trust algorithm performing equally well in terms of accuracy as the Trust-enhanced collaborative filtering (Eq. 5, indeed in the results for the MovieLens dataset in table 2 we see that Ensemble Trust returns a slightly better MAE.

6.2 MovieLens Results

Table 2 presents the results of each algorithm performed on the MovieLens dataset. Further to that discussed in section 6.1, on close inspection of the results, we see that the standard formulae perform quite well, with the Pearson CF (Eq. 3) returning a MAE of 0.707 with 10% of the ratings hidden. On the other hand, we also notice that the Trust CF algorithm (Eq. 5) performs slightly but not significantly better with a MAE of 0.70. This is more than likely due to the high connectivity of the dataset, with many ratings available for PCC to use. However, we notice that as the percent of ratings that are removed increases the accuracy of the trust-based formulae do not drop as quickly those of the standard formulae. Again this may be attributed to the fact that our trust inference formula is able to connect users to a wider range of users, and even though it does not use the ratings information like PCC does it seems that it is still able to gather enough information to perform well, and indeed slightly better. We also remark here that the Ensemble Trust algorithm (Eq. 8) actually performs the best for this dataset, slightly outperforming the TrustCF technique for both MAE and RMSE measures. The performance of the RMSE measure shows, as stated in section 5.2, that even though Ensemble Trust maximizes the coverage of rating predictions as much as possible it doesn't compromise the accuracy of the predictions and doesn't suffer from the coverage-accuracy trade-off. As discussed in section 6.1, this can be attributed to the fact out trust formula does not use any form of propagation and infers direct trust between users. These results show that by using both the PCC information and the trust information Ensemble Trust is able to maximize coverage as well as accuracy.

6.3 Epinions Results

Table 3 shows the results performed on the subset of the Epinions dataset. First of all, from these results we remark that, counter-intuitively, the naive Simple Mean algorithm (Eq 1) actually outperforms all of the other algorithms. Of course, this is not the case for both MovieLens and LibimSeTi datasets and certainly goes against intuition. However, these results are not completely a surprise and are similar to those observed in [11] for the same dataset. The main explanation for this result, is the fact introduced in section 6.1 that the vast majority of ratings (74%) in this dataset have

	10% hidden			20% hidden			30% hidden			50% hidden		
Algorithm	MAE	RMSE	Cov %	MAE	RMSE	Cov %	MAE	RMSE	Cov %	MAE	RMSE	Cov %
Pearson CF	1.444	2.148	89.43	1.453	2.167	85.85	1.461	2.176	83.05	1.472	2.196	76.32
Pearson WM	1.452	2.275	89.43	1.454	2.298	85.85	1.455	2.304	83.05	1.453	2.327	76.32
Simple Mean	1.459	2.223	100	1.464	2.247	100	1.475	2.266	100	1.491	2.308	100
Trust CF	1.410	2.077	99.02	1.418	2.094	98.73	1.433	2.116	98.33	1.456	2.153	96.69
Trust Filtered Mean	1.437	2.206	99.02	1.438	2.224	98.73	1.447	2.242	98.33	1.454	2.275	96.69
Trust Filtered CF	1.488	2.149	89.21	1.498	2.170	84.9	1.504	2.178	81.85	1.516	2.199	72.72
Trust WM	1.43	2.202	99.02	1.432	2.220	98.73	1.441	2.239	98.33	1.454	2.279	96.69
Ensemble Trust	1.410	2.077	99.02	1.418	2.095	98.73	1.433	2.116	98.33	1.456	2.153	96.69

Table 4: Results for LibimSeTi

a value of either 4 or 5, with (45%) of these having the highest rating of 5. With such little variance in the rating values the simple unweighted mean of all the ratings of users will usually return values close to the real rating value. This is also reflected in the performance of the Trust Filtering Algorithm (equation 6), which performs equally well as the Simple Mean. This algorithm likewise uses simply a non-weighted average of the ratings for an item, but this time only from trusted users. But with such little variance between ratings we noticed that filtering through trusted users does not improve on the simple mean, and in fact in terms of coverage performs worse. However, concentrating only of the performance of the trust-based algorithms against the standard algorithms, we can see that the trust enhanced algorithms again outperform their standard similarity based counterparts in terms of accuracy.

In terms of coverage, the percentage of rating predictions the trust-based algorithms were able to make was almost twice as much as those based on PCC. Again, this may be attributed to the sparsity of this Epinions dataset. As discussed, the calculation of PCC uses the deviation of ratings for shared items from the average rating of a user, and with the sparsity and low level of rating distribution of this dataset as discussed in section 6.1, it appears that the algorithm finds it difficult to compute positive similarity measures between a sufficient number of users. On the other hand, as discussed in section 6.1, the use of the *two-hop neighbourhood* structural aspect of our trust inference formula (Eq. 11) allows users to connect to a larger range of other users, resulting in the set of trusted users who have rated an item for which a rating value is to be predicted to larger than the set of users with a positive similarity measure from the PCC, thus giving much greater coverage to the trust enhanced techniques. The results from this Epinions dataset show the advantage of this feature of our formula in very sparse datasets.

As well as this, we remark that trust-enhanced techniques do not seem to suffer from the so-called "accuracy-coverage trade-off" described in [19], where a rise in coverage is usually at the expense of accuracy as previously experienced by some trust-enhanced measures. One possible explanation for this may be that previous comparisons have used trust propagation techniques [7] to increase the coverage of their trust enhanced formulas, and have experienced the issue whereby shorter propagation paths deliver more accurate results, and thus increasing the number of hops of propagation delivers less accurate trust estimations, and thus negatively impacting the accuracy of subsequent rating predictions. Our method however, does not seem to suffer from this drop in accuracy as it does not use any sort of trust propagation, trust is inferred directly between users and consequently our resulting trust metrics seem to avoid the accuracy trade off experienced through trust propagation.

6.4 LibimSeTi Results

Table 4 presents the results for each of the recommendation techniques performed on the LibimSeTi dataset. In comparison to the improvement in the recommendation accuracy shown in the MovieLens dataset, on a close inspection of the results for the LibimSeTi dataset, we can a see particularly larger scale of improvement in the accuracy of predictions when trust is incorporated into the recommendation process. This is demonstrated by a superiority of 0.34 for performance of the the Trust CF algorithm in relation to the MAE performance for 10% of the ratings hidden over its standard counterpart Pearson CF, with both having a MAE of 1.410 and 1.444 respectively. Whereas the superiority between these two algorithms for the same test on the MovieLens dataset was only 0.07.

One possible explanation for this increased superiority over different datasets may be the increased number of extreme ratings present in the LibimSeTi dataset, As discussed in [3], this LibimSeTi dataset contains a large proportion of extreme ratings, with a large percentage of ratings in the dataset having either the lowest possible value of 1, or the highest possible value of 10. In many of the previous studies that have incorporated trust into the recommendation process [4, 12], it has been shown that difference in opinion between the users in a dataset is where the trust-enhanced techniques perform best in relation to the standard techniques, where a user's rating for a particular item differs significantly from the average rating for this item. Thus, considering these extreme opinions present in the LibimSeTi, the use of trust in the recommendation techniques, as previous studies have shown, seems to provide more personalization to the predicted recommendations reflecting these differences in opinions.

Further to that discussed in 6.1, we notice that the Ensemble Trust (Eq. 8) and the Trust CF formulae are almost identical throughout the results. This may be because the set of users who have a positive pearson correlation (PCC) to the target user is similar to the set of users who are trusted by the user.

Overall, we see common behavior to that of the datasets above, as the percentage of ratings that are hidden increases, the coverage of the algorithms drop, but the trust formulas do not drop as quickly

as the standard techniques, remaining reasonably optimal.

7. CONCLUSION

In this paper we present an empirical analysis of a number of techniques used to incorporate trust into recommender systems in comparison to standard recommendation algorithms. Using a trust inference formula developed in our previous work, we apply these techniques to three different datasets without the need for explicit trust assertions, by automatically inferring trust links between users. We apply each technique to each dataset to predict rating values with 10%, 20%, 30% and 50% of the ratings hidden from the graph. From these prediction of each algorithm we measure the Mean Absolute Error, Root Mean Square Error and the prediction coverage and thus analyze the effects of these techniques on three different datasets with increasing numbers of ratings removed. From these results, we remark that the more ratings that are hidden in the graph, the accuracy of the standard algorithms drop at a quicker rate than those of the trust-based algorithms. We also remark that the prediction coverage of trust-based techniques remains relatively high even with 50% of the original ratings from the graph.

It is clear from these results that the incorporation of trust into recommendation algorithms can increase both the accuracy and the coverage of personalized recommendations. However, it is important to note the affect on each algorithm of the connectivity of the network in question. Particularly in the case of a sparse dataset, the use of trust can significantly improve both the coverage and the accuracy of recommendations. On the other hand, when used in a highly connected network where the ratings of items are reasonably distributed, such as MovieLens the improvement is not distinct, with trust showing signs of improvement but not very significant improvement. However, with a well connected network but where the ratings are more extreme and show disagreement between users such as the LibimSeTi dataset, we can see that trust can distinctly improve accuracy and personalization of recommendations.

8. REFERENCES

[1] G. Adomavicius and A. Tuzhilin. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Trans. on Knowl. and Data Eng.*, 17(6):734–749, June 2005.

[2] L. Brozovsky and V. Petricek. Recommender system for online dating service. In *Proceedings of Znalosti 2007 Conference*, Ostrava, 2007. VSB.

[3] N. Delannay and M. Verleysen. Collaborative filtering with interlaced generalized linear models. *Neurocomputing*, 71(7-9):1300–1310, Mar. 2008.

[4] J. Golbeck. Generating predictive movie recommendations from trust in social networks. In *Proceedings of the 4th international conference on Trust Management*, iTrust'06, pages 93–104, Berlin, Heidelberg, 2006. Springer-Verlag.

[5] J. Golbeck. Trust and nuanced profile similarity in online social networks. *ACM Trans. Web*, 3(4):12:1–12:33, Sept. 2009.

[6] J. A. Golbeck. *Computing and applying trust in web-based social networks*. PhD thesis, University of Maryland, 2005.

[7] R. Guha, R. Kumar, P. Raghavan, and A. Tomkins. Progation of trust and distrust. In *13th international conference on World Wide Web*, WWW'04, pages 403–412, 2004.

[8] J. L. Herlocker, J. A. Konstan, L. G. Terveen, and J. T. Riedl. Evaluating collaborative filtering recommender systems. *ACM Trans. Inf. Syst.*, 22(1):5–53, Jan. 2004.

[9] C. D. Jensen, S. Poslad, and T. Dimitrakos, editors. *Trust Management, Second International Conference, iTrust 2004, Oxford, UK, March 29 - April 1, 2004, Proceedings*, volume 2995 of *Lecture Notes in Computer Science*. Springer, 2004.

[10] P. Massa and P. Avesani. Trust-aware collaborative filtering for recommender systems. In R. Meersman and Z. Tari, editors, *On the Move to Meaningful Internet Systems 2004: CoopIS, DOA, and ODBASE*, volume 3290 of *Lecture Notes in Computer Science*, pages 492–508. Springer Berlin Heidelberg, 2004.

[11] P. Massa and P. Avesani. Trust-aware recommender systems. In *Proceedings of the 2007 ACM conference on Recommender systems*, RecSys '07, pages 17–24, New York, NY, USA, 2007. ACM.

[12] P. Massa and P. Avesani. Trust metrics on controversial users: balancing between tyranny of the majority and echo chambers. *International Journal on Semantic Web and Information Systems*, 3(1):39–64, 2007.

[13] P. Massa and B. Bhattacharjee. Using trust in recommender systems: An experimental analysis. In Jensen et al. [9], pages 221–235.

[14] D. O'Doherty, S. Jouili, and P. Van Roy. Towards trust inference in bipartite social networks. In *Proceedings of the Second ACM SIGMOD Workshop on Databases and Social Networks, DBSocial 2012, Scottsdale, USA*, ACM, 2012.

[15] J. O'Donovan. Capturing trust in social web applications. In G. Jennifer, editor, *Computing with Social Trust*, volume 3, pages 213–257, 2009.

[16] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl. Grouplens: an open architecture for collaborative filtering of netnews. In *Proceedings of the 1994 ACM conference on Computer supported cooperative work*, CSCW '94, pages 175–186, New York, NY, USA, 1994. ACM.

[17] B. Smyth and J. O'Donovan. Trust in recommender systems. In *10th international conference on Intelligent user interfaces*, pages 167–174, 2005.

[18] P. Victor, M. D. Cock, and C. Cornelis. Trust and recommendations. In F. Ricci, L. Rokach, B. Shapira, and P. B. Kantor, editors, *Recommender Systems Handbook*, pages 645–675. Springer, 2011.

[19] P. Victor, C. Cornelis, M. D. Cock, and A. M. Teredesai. Trust- and distrust-based recommendations for controversial reviews. *IEEE Intelligent Systems*, 26:48–55, 2011.

[20] J. Wang, J. Yin, Y. Liu, and C. Huang. Trust-based collaborative filtering. In *FSKD*, pages 2650–2654. IEEE, 2011.

[21] J. Weng, C. Miao, and A. Goh. Improving collaborative filtering with trust-based metrics. In H. Haddad and H. Haddad, editors, *SAC*, pages 1860–1864, New York, NY, USA, 2006. ACM.

[22] E. Zheleva, L. Getoor, J. Golbeck, and U. Kuter. Using friendship ties and family circles for link prediction. In *Proceedings of the Second international conference on Advances in social network mining and analysis*, SNAKDD'08, pages 97–113, Berlin, Heidelberg, 2010. Springer-Verlag.

[23] C.-N. Ziegler and G. Lausen. Analyzing correlation between trust and user similarity in online communities. In Jensen et al. [9], pages 251–265.