Elevating Prediction Accuracy in Trust-aware Collaborative Filtering Recommenders through T-index Metric and TopTrustee lists

Alireza Zarghami
Information Systems Group, EEMCS
University of Twente (UT)
Enschede, Netherlands
a.zarghami@utwente.nl

Soude Fazeli, Nima Dokoohaki
ECS, ICT,
Royal Institute of Technology (KTH)
Stockholm,Sweden
{soude,nimad}@kth.se

Mihhail Matskin
Norwegian University of Science
and Technology (NTNU)
Trondheim, Norway
misha@imit.kth.se

Abstract

The growing popularity of Social Networks raises the important issue of trust. Among many systems which have realized the impact of trust, Recommender Systems have been the most influential ones. Collaborative Filtering Recommenders take advantage of trust relations between users for generating more accurate predictions. In this paper, we propose a semantic recommendation framework for creating trust relationships among all types of users with respect to different types of items, which are accessed by unique URI across heterogeneous networks and environments. We gradually build up the trust relationships between users based on the rating information from user profiles and item profiles to generate trust networks of users. For analyzing the formation of trust networks, we employ T-index as an estimate of a user's trustworthiness to identify and select neighbors in an effective manner. In this work, we utilize T-index to form the list of an item's raters, called TopTrustee list for keeping the most reliable users who have already shown interest in the respective item. Thus, when a user rates an item, he/she is able to find users who can be trustworthy neighbors even though they might not be accessible within an upper bound of traversal path length. An empirical evaluation demonstrates how T-index improves the Trust Network structure by generating connections to more trustworthy users. We also show that exploiting T-index results in better prediction accuracy and coverage of recommendations collected along few edges that connect users on a Social Network.

Keywords: *Recommendation, Collaborative Filtering, Trust Networks, Social Networks, Social Trust, Ontological modeling, Performance*

1. Introduction

Recommender systems(RS) have emerged as a significant response to the information overload problem where it is challenging for users to find desired information. They follow the goal of the classical information retrieval field which is providing users with interesting content according to their historical behavior. There are two types of techniques used in Recommendation Systems: Content-based and Collaborative Filtering(CF). Content-based methods compare representations of an item's content with representations of content of user's interests [1]. As it requires the user to express an explicit model of its preferences, the cognitive load on the user is high. On the other hand, since it only considers user preferences, the user misses the opportunity to find new and potentially interesting items. Collaborative Filtering is the other technique for Recommender Systems which successfully overcomes these problems, since CF only depends on users' opinions and ratings of items instead of the explicit content description required by Content-based Recommender Systems. CF algorithms search for like-minded users and introduce them as neighbors to predict a new item's rating for a user based on its neighbors' ratings. Traditional CF algorithms are based on the similarity of user profiles as a weight for making recommendations. However, it is difficult to compute similarity measure between users when ratings are sparse. Therefore, the profile similarity on its own is not effective and we need to consider additional factors which can measure and interpret the similarity between user profiles.

Among many systems which have exploited the notion of trust, Recommender Systems are considered as the dominant ones. Collaborative Filtering (also known as Social) Recommenders can be extended to take into account the trust relation in-between users, in order to give better suggestions to users. As a matter of fact, users would prefer to receive recommendations from those they trust the most.

In this paper, we propose a mechanism for augmenting a Recommender System. We refer to this approach as T-index which digests trust values between users on the Trust Networks in order to provide more improved recommendations to users. To do so, we introduce a T-index measure inspired by H-index [2] to discover the users within our trust network who provide trust values higher than or equal to *T*, for number of users larger than *T*. Therefore, more users from divergent areas of users' preferences might be accessible within a few edges of the path that connects users on the network. We demonstrate the applicability of this approach in the context of a movie recommender. Initial results with a large extent of users have proved our hypothesis.

The rest of the work is structured as follows: Section 2 provides the background and related works. Next, Section 3 describes our approach, and then Section 4.2 presents our experimental results and discussions. Finally, we conclude and we present an overview of the future work in Section 5.

2. Background

In this section, we present a definition of trust and the existing mechanisms for trust computation and propagation relevant to the context of our work.

2.1. Trust Ontology

Social Networks have emerged to connect lots of users on the Web. Users are able to express information about their relationships such as how much they can rely on people in the online community [3]. This phenomena leads us to the social notion of trust which helps users to find their trustworthy friends and share their preferences for an item like a movie or music. This could also help recommendations to be generated from trustworthy partners [4]. FOAF (Friend-of-a-Friend) vocabulary [5] describes users' information and their social connections through concepts and properties in the form of an ontology using Semantic Web technologies [6], [3]. Golbeck[3] introduces an ontology that extends FOAF vocabulary for modeling trust relationship between users. TidalTrust algorithm[3] is proposed by Golbeck to infer trust values on Social Networks. Her inference algorithm is exploited in two applications: FilmTrust[7] uses movie ratings and reviews to personalize the website for each user based on its trust relationships with others and TrustMail[8] is an email client that shows the trust rating of the sender and uses it as the score for the message. Although Golbeck's ontology provides an efficient structure, every relationship describes only one subject. Dokoohaki et al. [9] present an ontology for modeling structure of trust relations between users that is more efficient in terms of the size of the generated networks using ontology. We extend this ontology to model trust between users with an extra element for measuring T-index-based trustworthiness of a user.

2.2. Trust-aware Recommender Systems

Social-aware Recommenders which are extended with trust phenomena have proven to provide users with more reliable recommendations. Massa and Avesani[10] introduce an architecture for a trust-aware recommender capable of trust aggregation for all of the users in a Social Network. As a result, the "importance"of a certain user is predicted by using a graph walking algorithm through the "Web of Trust" which is explicitly expressed by users [10]. Andersen et al.[11] propose an axiomatic approach in which the users' opinions are aggregated in trust networks to generate personalized recommendations. Their method makes a recommendation for each node through a voting network based on positive and negative votes already assigned to a subset of the nodes. The methods mentioned above are all limited to some explicit trust rating to infer other trust relations. Some efforts have been made to formalize the trust where it can not be explicitly expressed by users. O'Donovan and Smyth [12] represent computational models of trust as profile-level and profile-item-level based on the past behavior of user profiles. According to their model, a recommender's rating is correct if the difference between its rating and the target user's rating is less than a predefined value. Lathia et al.,[13] propose a trust-learning method that is similar to the models presented by O'Donovan and Smyth in [12]. The main idea is that the recommenders, who provide useful information, should be rewarded and those who have no information available, should be downgraded. The trust-based collaborative filtering algorithm used in their method requires a centralized user-item matrix which might lead to scalability problem as the number of users increases. Weng et al.[14] assume each user as a peer connected to other users in a decentralized trust network of users. The trust between two users is computed based on the "Goodman-Kruskal measures of association of cross classifications"[15]. In this paper, we adapt the formalization proposed by Lathia et al.[13] to derive the trust value between users. We introduce an agent-setting in which every user is considered to be an agent connected to other users to form a trust network. Such a setting should provide better scalability since the distributed allocation of trust-related data is supported.

3. A Semantic Trust-ware Recommendation Framework

Our goal is to create trust relationships among all types of users with respect to different types of items, accessible through unique URI across heterogeneous networks and environments. To achieve this, we have developed an ontological framework, shown in Fig. 1, composed of three main modules: Semantic Profile Manager, Trust Engine and Recommendation System.

Upon rating an item by a user, the Semantic Profile Manager module either creates or updates an ontology-based profile for both user and item. To keep track of items rated by the user, the user profile extends the *RankRelation* concept.

The Trust Engine module generates a so-called *trust network* of users based on the profile information of users and items in a distributed manner. To do so, a user profile extends the trust ontology to keep top-n neighbors and its mutual trust values with them. Note that there is no global view of a trust network for users and they are only provided with information regarding their neighbors and rating history. Therefore, it is possible to maintain users in different groups on several servers to achieve better scalability. To cope with privacy requirements, these servers can be located in different organizations while profiles of users and items are accessible only through their URI.

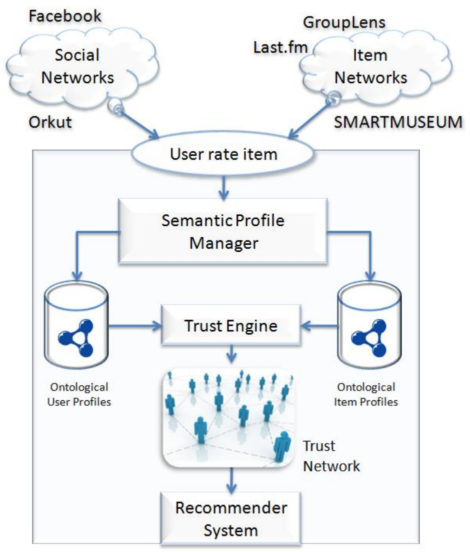


Figure 1. Ontological Framework

The Recommendation System module enables traversals through the trust network to collect recommendations for a target user and finally makes a predicted rating for the user.

The whole model is built on top of a knowledge acquisition system to improve manipulation of ontological data. The presented ontological framework provides us with high interoperability and openness to deal with heterogeneous networks.

3.1. TopTrustee and T-index

To build trust relationships among users, we enhance Collaborative Filtering with two novel concepts: T-index and TopTrustee.

3.1.1. T-index. H-index [2] was defined by Jorge E. Hirsch, a physicist, "'as the number of papers with citation number higher or equal to *H*, as a useful index to characterize the scientific output of a researcher"'. Extending this idea, we propose an estimate of a user's trustworthiness called T-index, similar to the H-index in showing the number of trust relationships between a user and its trusters with a trust value higher than or equal to *T*. T-index can be introduced as the Indegree of nodes in a trust network which provides not only the number of incoming edges as a regular Indegree, but it also considers the weight of incoming trust relationships. For a node on a network, Indegree represents the number of head endpoints adjacent to a node while Outdegree is the the number of tail endpoints.

Algorithm 1 Computing T-index

```
 1: procedure ComputeT-index ⟨$user, TrusterList$⟩
 2:     $TrusterValueList \leftarrow TrusterList$.sort($trustValue, desc$)
 3:     for all $trustValue$ in $TrusterValueList$ do
 4:         $trustValue \leftarrow$ multiply($trustValue, Max_{T-index}$)
 5:     end for
 6:     $Counter \leftarrow 1$
 7:     for all $trustValue$ in $TrusterValueList$ do
 8:         if $Counter < trustValue$ then
 9:             $Counter \leftarrow Counter + 1$
10:         else
11:             break
12:         end if
13:     end for
14:     $T\text{-}index \leftarrow Counter$
15:     return $T\text{-}index$
16: end procedure
```

The algorithm 1 describes how T-index is computed for a user. First, we introduce the maximum value of T-index as a global variable which defines the precision of T-index computation. Thus, we multiply all trust values (shown as label of arrows in Fig. 3) by this maximum value. Then, we start to count the number of trusters while their trust values are greater than the counter.

The procedure for T-index computation is invoked whenever a new incoming trust relationship is either created or updated. In other words, once a neighbor is either added or updated in a user's list of neighbors, the user sends a message to its new or updated neighbor in order to call the procedure for the respective user. As a result, the user's T-index is updated in its neighbors' *TrusterList*s and also in the *TopTrustee* lists of the items rated by the user.

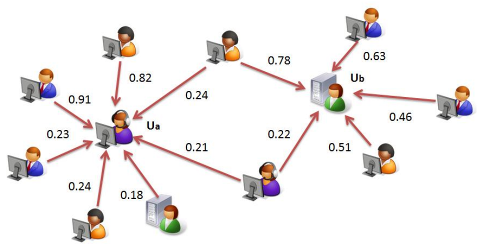


Figure 2. An Example of Trust Network for calculating T-index

In this work, we define a cluster as a group of users who all trust a common user, called the *Centric User* as the most trustworthy one within the cluster. Fig. 2 shows u_a and u_b as the centric users of two clusters. In the example presented in Fig. 3, we assume the maximum value of T-index as 10. According to Algorithm 1, we must multiply all the trust values by 10. We eventually achieve two result sets of $\{9,8,2,2,2,2,2\}$ and $\{8,6,5,5,2\}$ for u_a and u_b, respectively. Therefore, we start counting members of each set from the beginning of the list until the counter becomes higher than the trust value of a member. As a result, we obtain the value of u_a's T-index as 2 while the T-index value for u_b would be 4. Although u_a's Indegree is larger than u_b's Indegree, its T-index value is less than the one for u_b. It shows that u_a is less trustworthy within its trusters in comparison with u_b.

3.1.2. Item's TopTrustee. Suppose a subset of users have shown interest in a particular item. If a new user rates the item, it can share the similar interest with those users. This idea leads us to define *TopTrustee* as a concept associated with each item for keeping a few users who are interested in the corresponding item.

An item's *TopTrustee* is a user who has already rated the item and can join item's *TopTrustee* list if its T-index value is higher than a certain threshold. In fact, *TopTrustee* list introduces trustworthy users to the user who has just rated the item. The users in *TopTrustee* list may have no trust relationship with the user yet since they can not be reached through the maximum path length of *L*. However, they might be a source of useful information for the item's rater. To form *TopTrustee* lists of items we exploit T-index.

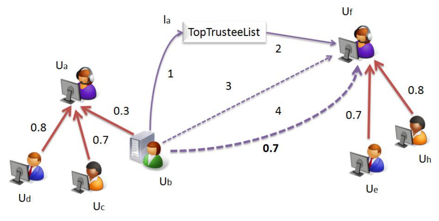


Figure 3. A scenario of utilizing TopTrustee List

As shown in Fig. 3, when u_b rates item i_a, its mutual trust values with all users in two sets are computed and updated. The first set is its *top-n neighbors* as the first *n* users who are not only directly connected to the user but also provide the highest mutual trust values with the user. The other set is the item's *TopTrustee* list. u_f has rated i_a and is already located in i_a's *TopTrustee* list. After computing the trust value between u_b and u_f based on the trust formula presented by [13], u_b finds u_f more trustworthy than u_a as one of its current top-n neighbors even though u_f is not accessible to u_b within path length of *L*. Eventually, u_b adds u_f to its top-n neighbors. As a result, u_b can be provided by u_f with more reliable recommendations in comparison with u_a's recommendations.

3.2. The Semantic Profiling Manager

The Semantic Profile Manager module is responsible for creating and updating ontology-based profiles for both user and item.

3.2.1. Ontological User Profile. We take advantage of the trust model presented by Dokoohaki et al. [9] to define the trust between users who are expressed using the *FOAF Agent* concept. Dokoohaki's trust ontology has three concepts. *Relationship* is the main element which expresses the trust relations on top of the Social Network of FOAF user profiles. *MainProperties* and *AuxiliaryProperties* are the other main components of aforementioned ontology, which respectively define essential and optional attributes for relations which exist in between users on the network. Two associations connect both *MainProperties* and *AuxiliaryProperties* to the *Relationship* concept. *Relationship* always has a sink and a source, which is described by a *Truster* and a *Trustee*. Reader is refered to [9] for more information about the complete structure of trust ontology. In our model, a trust value is computed based on users' ratings to different items, possibly in different contexts. To compute the trust value between users, we follow the approach proposed in [13] based on the difference of a user's rating and its recommender's rating to their common item(s). As a result, as the distance between their rating values increases, trust decreases linearly.

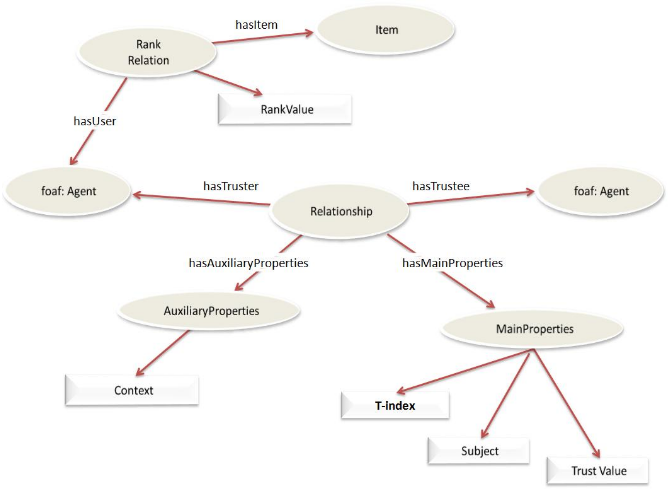


Figure 4. User Ontology Model

As shown in Fig. 4, we create an instance of *Relationship* concept between two users for whom a trust value is computed. The users are specified as *Truster* and *Trustee* and their trust value and subject is assigned as *MainProperties* [9] to the instance defined earlier. In addition, we assign T-index as a *MainProperty* of the *Relationship* instance. We also define the *RankRelation* concept for associating a user to an item by a rank value. This concept is used to keep track of rated items by a user that we refer to as *user profile*.

3.2.2. Ontological Item Profile. We have developed an ontology for item's knowledge domain which can be extended by all other ontologies in the same domain. We introduce a new concept called *TopTrustee*, which is derived from the notion of item's *TopTrustee* described in section 3.1.2, and we assign it to an individual item to create a list of users who rate the item. The list of raters is ordered by their T-index. In a real world scenario, these *TopTrustee* lists can be implemented by Distributed Hash Tables (DHT) [16] with unique URI as their keys.

3.3. Trust Engine

We adapt the formalization of trust presented by Lathia et al.[13] based on the difference between a user's rating and its recommender's rating to their common item(s). As the difference between their rating values decreases, the trust value between them increases linearly. Suppose we have two users u_a and u_b. Trust between them is formalized as follows [13]:

$$T(u_a, u_b) = 1 - \frac{\sum_{i=1}^{n}(r_{u_a,i_i} - r_{u_b,i_i})}{r_{max} * n} \tag{1}$$

This formula computes the total differences between a user's rating values and its recommender's rating values over *n* historical ratings of u_a multiplied by the maximum value in each rating scale (i.e., 5). To improve the prediction accuracy of the generated recommendations, [13] suggest transposing the rating values of the truster based on its past experience with a trustee by considering number of user's ratings which are the same, lower or higher than its recommender's ratings [13]:

$$tr(r) = \frac{(r - 1 * lower_r) + (r * same_r) + (r + 1 * higher_r)}{lower_r + same_r + higher_r} \tag{2}$$

The transposition of the rating values, $tr(r_{u_b,i_i})$ can be used instead of r_{u_b,i_i} to guarantee that a user and its recommenders follow the same scale for rating.

3.4. Trust Network

We gradually build up the trust relationships between users based on the rating information of user profile and item profile to generate a so-called *trust network* of users.

As mentioned, we keep top-n neighbors of a user in an ontological structure based on their mutual trust values. The list is updated on "'rating a new item"' event. If the event leads to some modifications in top-n neighbors of a user, then T-index value is recalculated and updated in all *TopTrustee* lists which include the user. The scenario is described as follows: when a user rates a new item, we compute its trust with all item's *TopTrustee*s who do not exist in its current top-n neighbors but might potentially be trustworthy users. We also update trust values between the user and its top-n neighbors. Eventually, we form a new top-n neighbors by selecting the most trustworthy users from the union of its preceding neighbors and the potential trustees.

3.5. Recommendation System

There is no central view of similar users' ratings in distributed recommender systems. Thus, in order to generate a recommendation, we need to find a solution for aggregating neighbors' opinions. A random walk though neighbors for collecting an item's ratings would be an appropriate solution [14]. In fact, the length of edges that connect the users by traversal through the formed trust network should be limited to an upper bound (*L*). However, defining a suitable value for *L* is challenging as it leads to a trade-off between accuracy and performance. Therefore, as the number of parallel traversals and *L* increases, although we can achieve better prediction accuracy and coverage for recommendations, we will require more bandwidth and computations resources.

On the other hand, a user is allowed to traverse through its direct or indirect neighbors as long as its mutual trust value does not fall below a predefined minimum threshold (*v*). For obtaining the trust value between two indirect users, we follow the regular multiplication of trust value assigned to edges of their connecting path:

$$T_{u_a,u_c} = T_{u_a,u_b} * T_{u_b,u_c} \tag{3}$$

This trust propagation formula facilitates u$_a$ as a source to find its mutual trust values with all of its indirect neighbors. Then, u$_a$ determines to keep those with trust value less or equal to *v*. In the following, we show how users perform traversals based on limiting trust value to *v* by an example.

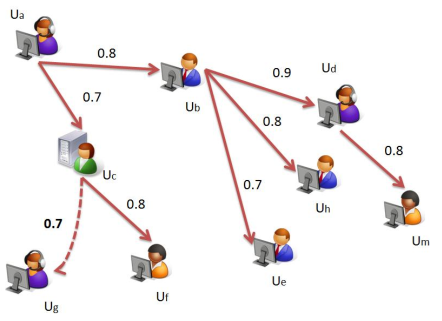


Figure 5. An example of traversals with respect to minimum threshold of trust value

As shown in Fig. 5, u_a finds u_g with the mutual trust value of $T(u_a, u_g) = 0.49$ which is computed by Formula (3). If *v* is defined as 0.5, then traversals that originated from u_a are not allowed to reach u_g. However, u_m can be reached by u_a with $T(u_a, u_m) = 0.58$, which is acceptable as it is a value greater than *v*. Now, if we define *L* as 2, u_a loses u_m which is more trustworthy but is located further away than u_g. Hence, considering a minimum threshold of trust value (v) appears to be more effective than limiting the length to L. Nevertheless, we aim to evaluate our proposed method within close neighbors. Thus, we need to take length of traversal path(L) into account, along with v for trust value of the traversal path to address this situation. In this work, we define the maximum length of traversal path (L) as 3. Collecting neighbors' opinions in short traversals improves not only the performance of recommendation systems, but also their reliability. As a result, the system becomes more resistant to failures whether in network connections or users since fewer users are involved.

After collecting all the information from a user's neighborhood by traversals, we aim to minimize the risk of recommending irrelevant items to a user[13]. Therefore, predicted rating provides us with the fact whether or not the user is interested in an item. The prediction value is taken as a weighted average of user a's neighbors ratings[17]:

$$p(a,i) = \frac{\sum_{b \in N_{(a,i)}} (tr(r_{b,i}) * T(a,b))}{\sum_{b \in N_{(a,i)}} (T(a,b))} \qquad (4)$$

This equation shows the predicted rating generated for user a, by aggregating the information from either its direct or indirect neighbors who can be good sources of information regarding item i. Thus, $N(a,i)$ is set of top-n neighbors who rated the item i, and we transpose their ratings as $tr(r_{b,i})$ based on Formula (2) before combining the ratings with trust values.

4. Evaluation

4.1. Setup

We use the *MovieLens*[1] dataset for evaluating our method. This dataset consists of 943 user profiles. Ratings are based on a five point scale. The profiles are divided into training and test sets that include 80% and 20% of the ratings, respectively. This work is fully implemented in an ontological environment. We use Protégé[18] to design ontology structures for user and item. Moreover, we work with Protégé API in Java to implement the recommendation system. At first, we apply the training data to build the trust-based social network of users in ontological representation. Then, we make recommendations using a traversal mechanism through the trust network. We visualize the trust-based social network by Welkin[19] to study effect of T-index on structure of the network.

Coverage has been used to evaluate recommender systems that measures the percentage of items that a recommender system can provide predictions for [20]. Mean Absolute Error (MAE) is the other metric for evaluating recommender systems. It measures the average absolute difference between a predicted rating made for a specific user and the user's actual rating [20].

In this work, we aim to show how the performance of our recommendation system can be improved in terms of coverage and accuracy by exploiting T-index. We also demonstrate how coverage and MAE are influenced by T-index variation.

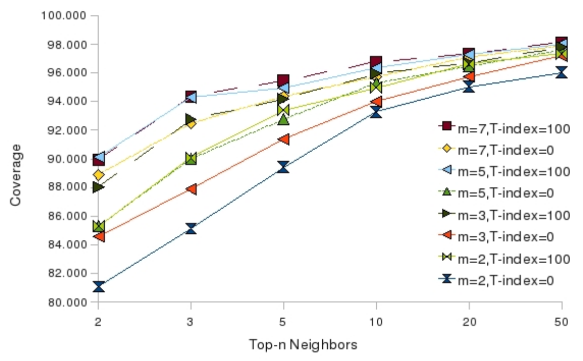
Furthermore, we study the effect of T-index variation on the network structure based on trust relationships in a social setting. As mentioned earlier, Indegree describes the number of head endpoints adjacent to a node. We demonstrate that users are able to find a greater number of centric users, when T-index is used which results in a more balanced Indegree distribution among top trustworthy users. We compare the Indegree distribution of the top-10 trustworthy users at different values of T-index. Then, we build trust networks both with and without T-index to observe the difference. The differences includes both inferred and trimmed edges made when T-index is employed.

In general, all distributed recommender systems achieve better coverage as the path length of traversal increases. In this experiment, traversals through the trust graph is bounded to contribute fewer users in making recommendations for evaluating coverage under stricter circumstances. Therefore, we attempt to generate recommendations for a user, based on the rating values of neighbors who provide mutual trust values higher than the minimum threshold(v) of 0.1 and can be reached within the upper bound for path length of traversals (L) as 3. According to the proposed trust mechanism and with $v = 0.1$, traversals rarely reach the path length of L.

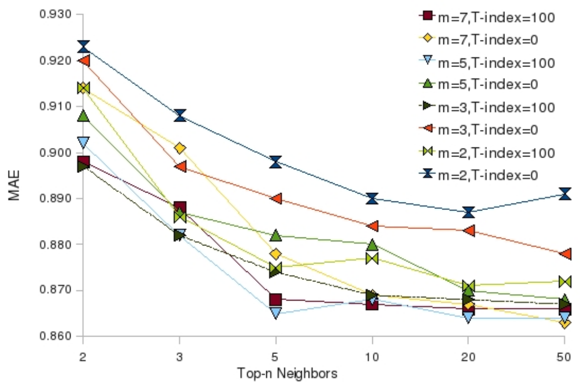
To study the effect of T-index, we evaluate our method using the following two sets of configuration.

- With and without T-index:
  We set the values of T-index to 0 and 100 for evaluating the result without and with T-index, respectively. We run our experiment at different settings for various sizes of top-n neighbors for each user(n) and different sizes of *TopTrustee* list for each item(m). The values of n and m are tuned to be: $n \in \{2, 3, 5, 10, 20, 50\}$ and $m \in \{2, 3, 5, 7\}$.
- T-index variation:
  In this configuration, we apply various values of T-index to find the most effective value for T-index. We will demonstrate that although using T-index (T-index=100) improves the results across a range of different values of m and n, we achieve the most significant improvement when $m = 5$ and $n = 5$ in the first configuration setting. Therefore, we choose the values of both n and m to be 5 for studying the Indegree distribution and trust networks structure most effectively. We also consider different values for T-index which range from 0 (meaning no

1. http://www.cs.umn.edu/research/GroupLens/data/



(a) Prediction Coverage



(b) MAE

Figure 6. Comparing the results with and without using T-index

T-index is used) to 1000 (the maximum value): T-index $\in \{0, 25, 50, 100, 200, 500, 1000\}$. To study the prediction coverage and accuracy of the generated recommendations, the values of n are tuned to be $\in \{2, 3, 5, 10, 20, 50\}$ while m stays the same as 5.

4.2. Evaluating The Recommendation System

As mentioned above, we first show how the performance of our recommendation system can be improved by studying the effect of T-index on prediction coverage and accuracy.

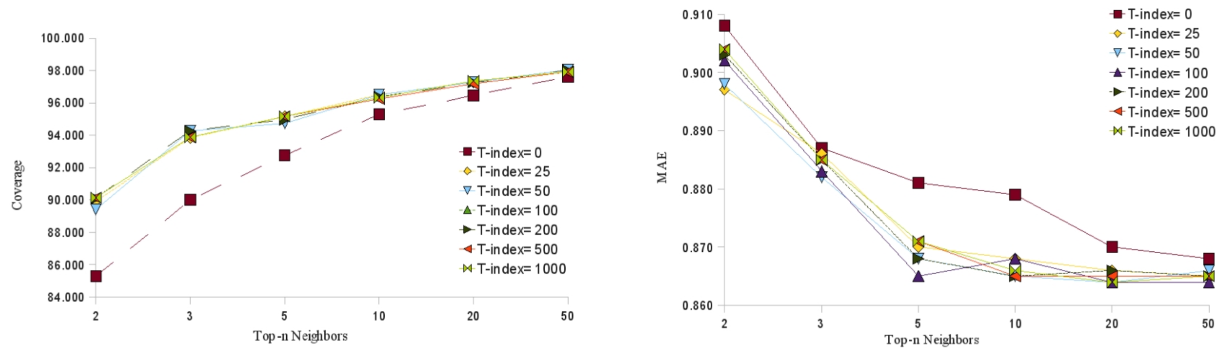
4.2.1. Prediction Coverage and Accuracy with and without T-index. We first study coverage for different predefined settings. As shown in Fig. 6(a), the minimum coverage for n= 2, m= 2 without T-index is more than 81% which is improved in camparison with the result of similar work[14] at the same path length (L= 3) and even for larger sizes of n. As mentioned, a user's neighborhood is updated upon rating a new item with *TopTrustee* list of size m. Thus, inputs of trust update function for a user, are top-n of its original neighbors and m item's *TopTrustee*s as potential neighbors. In other words, "'top-n neighbors"' generally used in all similar CF algorithms, is interepreted in this work as the union of top-n neighbors and m users of items' *TopTrusteeList*.

We observe that T-index has a better effect on coverage as size of neighborhood and items' *TopTrustee* list decreases. For instance, coverage is significantly improved when T-index is employed for n< 10 and m< 5. As shown in Fig. 6(a), with n=5, m=2 with T-index, achievable coverage is 93.39% which is better than the result with the same values for n and m=5 but without T-index (92.75%). Therefore, T-index improves coverage more effectively than the size of items' *TopTrustee* list, in some cases.

In the next step, we study MAE for the similar configurations to coverage. As shown in Fig. 6(b), MAE has the highest value (0.923) when n= 2, m= 2 without T-index. In some cases, we achieve more effective results for MAE with respect to T-index rather than size of *TopTrustee* list. For instance, MAE is 0.875 with n= 5, m=2 with T-index, which is better than result achieved with the same value for n= 5, m= 7 but without T-index (0.878). We observe that T-index significantly improves MAE for all values of n when m= $\{2, 3\}$, contrary to the coverage results. However, it makes MAE slightly better for n> 20 when m> 5. It outperforms similar works[14] considering the same threshold for path length of traversals (L= 3). It shows that including items' *TopTrusteeList* in "top-n neighbors" can improve the results. On the other hand, it reveals that utilizing T-index achieves better results.

So far, the results demonstrate that prediction coverage and accuracy are improved by employing T-index and item's *TopTrustee* even though path length of traversals for gathering recommendations is limited to 3. It is desirable due to the fact that as path length becomes smaller, recommendations are collected from more reliable neighbors and bandwidth consumption decreases which results in higher performance altogether.

4.2.2. Prediction Coverage and Accuracy with T-index variation. We study the effect of different T-index values on the performance of our recommendation system. First, we study coverage for several n and different T-index values while the value for m is the same and equal to 5. As shown in Fig. 7(a), coverage has improved at all values of n when T-index is employed. We also show that the coverage improvement is almost the same for all non-zero values of T-index. Nevertheless, we achieve improving results for coverage as the size of neighbors list (n) decreases.



(a) Coverage (b) MAE

Figure 7. Comparing the results with T-index variation

Finally, we study the effect of T-index variation on MAE. As with coverage, we observe in Fig. 7(b) that T-index improves MAE for all values of n. However, the extent of improvement of MAE changes with a constant value of T-index and different values of n. For instance, although MAE has the most effective result with T-index$=100$ and $n=5$, it has its worst value with the same T-index when $n=10$.

Despite coverage, T-index does not always make MAE better as the size of neighborhood list decreases. Fig. 7(b) shows that MAE is improved significantly with T-index when $n=5$ and 10 whereas MAE result is trivial when $n=3$ and 50. In conclusion, while using T-index results in better prediction accuracy and coverage of recommendations, accuracy is more affected by different values of T-index and the size of neighborhood list (n).

4.3. Trust Network Analysis

For analyzing the evolution of trust networks by exploiting T-index, we study the effect of T-index on the structure of generated trust networks. To do this, we first present the Indegree distribution of the Top-10 most trustworthy users both with and without T-index. Secondly, we compare the Indegree distribution results with different values of T-index. Then, we visualize trust networks with and without T-index to analyze the effect of T-index on network structure. Moreover, we show how trust networks' structure can be improved in terms of inferred and trimmed edges.

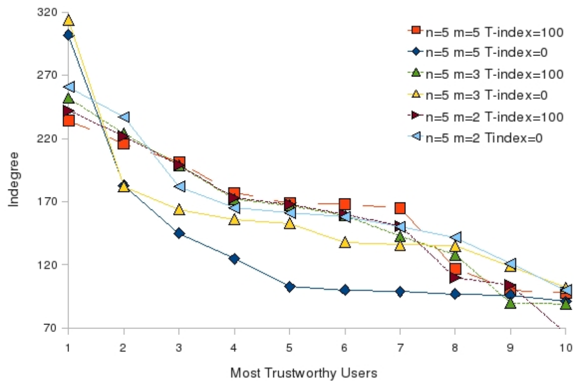
4.3.1. Indegree Distribution with and without T-index. We aim to identify the effect of T-index on trust networks in terms of Indegree distribution. Fig. 8(a) shows the distribution of Indegree for the first ten most trustworthy users with and without T-index. It can be seen that by employing T-index, more centric users can be found which results in more clusters. Therefore, a node's weights, in terms of incoming trust relationships, are effectively balanced in the trust graph of users, by utilizing T-index. In other words, as reliability of users declines, their Indegree gradually decreases with different sizes of items' *TopTrustee* list.

Indegree distribution results helps us to identify that how the trust networks are improved when T-index is employed. The results indicate that the average number of users who are able to find centric nodes increases. Therefore, users are provided with more trustworthy users for collecting the recommendations.

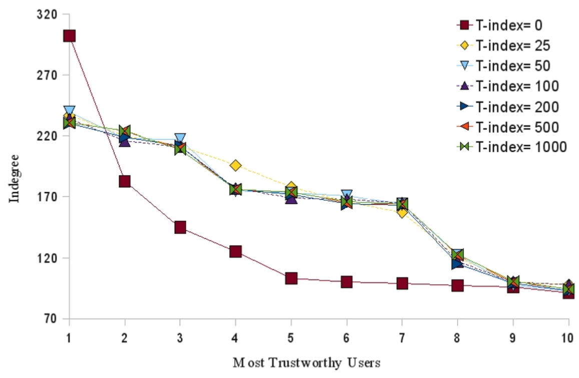
4.3.2. Indegree Distribution with T-index variation. In this step, we study the Indegree distribution of the top-10 trustworthy users for various values of T-index while n and m are both equal to 5. As mentioned earlier, Indegree represents incoming edges to a node as a user who is trusted by others. As shown in Fig. 8(b), when T-index is employed (T-index$<>0$), the top-10 trustworthy users' weights in terms of incoming trust relationships are more balanced. This means that users have on average more opportunities to find the most similar centric nodes as their main clusters. As a result, the load of incoming trust relationships imposed on the most trustworthy user, is distributed among other trustworthy users which makes our recommendation system more resistant against node failures or bottlenecks on the trust networks. Thus, the results significantly change when T-index is used, regardless of its non-zero values $(25, 50, 100, 200, 500, 1000)$.

4.3.3. Trust Network Structure Evolution. In this step, we show how the generated trust networks can evolve when T-index is applied. Similarly to Indegree distribution, we build up trust networks with and without T-index (T-index=100 and T-index=0, respectively), while n and m are both equal to 5.

Figs. 9(a) and 9(b) show the trust networks' structure

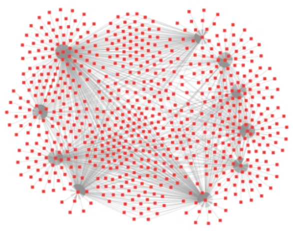


(a) With and without using T-index

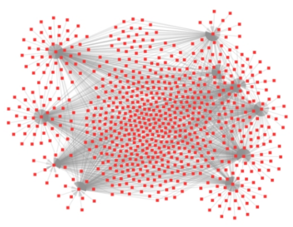


(b) With T-index variation

Figure 8. Comparing the Top-10 trustworthy users Indegree Distribution



(a) Generated through baseline algorithm



(b) Generated through T-index utilizing mechanism

Figure 9. Generated Trust Networks for Top-10 Trustworthy Users (n= 5, m= 5)

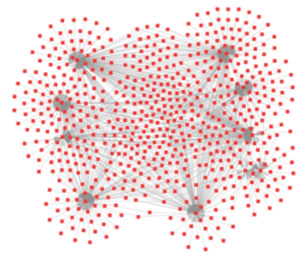
with and without T-index, for T-index=100 and T-index=0, respectively. For the sake of simplicity, we display only users(displayed as nodes) and their connections (trust relationships) to top-10 trustworthy users. As shown in the figures, the number of common users between clusters increases, which enables users from different clusters to find each other more easily. In our case, more users from divergent areas of users' interests as clusters can be accessible. We achieve the same result in generating the trust network with various values of n and m. In general, we show that trust relationships are more balanced in trust networks of users when T-index is used.

To justify the results, we compare the formed trust networks with and without T-index to show the inferred and trimmed edges individually. Fig. 10(a) indicates that inferred edges are mostly located between centric nodes. Therefore, the number of users which belong to different clusters, grows in the centric area of the figure. In contrast, 10(b) reveals that most of the trimmed edges are located in just one cluster.

5. Conclusion and Further Work

In this work, we have developed an ontological framework to create trust relationships among all types of users with respect to different types of items, accessed by unique URI across heterogeneous networks. We have built up a trust network of users to collect recommendations for a target user by using a walking algorithm. We have introduced an item's *TopTrustee* list which includes users who might not otherwise be reachable through a predefined maximum path length of traversals. Thus, when a user rates a new item, its neighborhood potentially consists of its own top-n neighbors plus m users of the item's *TopTrustee* list. We have also proposed a measure called T-index to prioritize the users of *TopTrustee* lists, based on their trustworthiness. Therefore, an item's *TopTrustee* list keeps the top-m trustworthy users who rate the item. We have depicted that by utilizing items' *TopTrustee* list, traversals length for finding users who rate a desired item, decreases which results in higher performance. We have shown that using T-index leads us to achieve better prediction coverage and accuracy of recommendations gathered in short length of traversal path. To justify the results, we have demonstrated how the structure of trust networks evolve when T-index is employed. Our empirical evaluation indicates that T-index increases the number of common users between different clusters and enables their users to access each other more conveniently. This leads to improved prediction coverage and accuracy of recommendations collected within the few edges that connect users. We show that the extent of improvement for accuracy, despite coverage, depends heavily on the T-index value as well as on the size of neighborhood list (n).

We intend to employ T-index as a coefficient to trust formalization in order to contribute trustworthy users more effectively. Furthermore, to alleviate the problem of mali-


(a) Inferred Edges

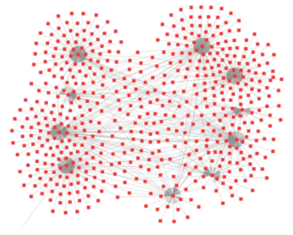

(b) Trimmed Edges

Figure 10. Alignment of Trust Networks for Top-10 Trustworthy Users (n= 5, m= 5)

cious nodes in a trust network, T-index can be obtained in a distributed manner like gossip-based aggregation[21].

Acknowledgment

This work has been done within the FP7-216923 EU IST funded SMARTMUSEUM project and is part of the IOP GenCom U-Care project which is sponsored by the Dutch government under contract IGC0816.

References

[1] J. L. Herlocker, J. A. Konstan, A. Borchers, and J. Riedl, "An algorithmic framework for performing collaborative filtering," *In Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, p. 230237, 1999.

[2] J. E. Hirsch, "An index to quantify an individual's scientific research output," *PNAS*, vol. 102, no. 46, pp. 16 569–16 572, November 2005. [Online]. Available: http://dx.doi.org/10.1073/pnas.0507655102

[3] J. A. Golbeck, "Computing and applying trust in web-based social networks," Ph.D. dissertation, University of Maryland at College Park, College Park, MD, USA, 2005, chair-Hendler, James.

[4] U. Shardanand and P. Maes, "Social information filtering: Algorithms for automating "'word of mouth"'," in *ACM Conference on Human Factors in Computing Systems*. ACM Press, 1995, pp. 210–217.

[5] FOAF, last modified Feb-2009, Available: http://www.foaf-project.org/.

[6] N. Dokoohaki and M. Matskin, "Structural determination of ontology-driven trust networks in semantic social institutions and ecosystems," in *International Conference on Mobile Ubiquitous Computing, Systems, Services and Technologies (UBICOMM '07) and the International Conference on Advances in Semantic Processing (SEMAPRO 2007)*. IEEE Computer Society, 2007, pp. 263–268.

[7] J. A. Golbeck, "Filmtrust: movie recommendations from semantic web-based social networks," in *Consumer Communications and Networking Conference, 2006. CCNC 2006. 3rd IEEE*, Department of Computer Science, University of Maryland, 2006, pp. 1314– 1315.

[8] J. Golbeck and J. Hendler, "Inferring binary trust relationships in web-based social networks," *ACM Trans. Interet Technol.*, vol. 6, no. 4, pp. 497–529, 2006.

[9] N. Dokoohaki and M. Matskin, "Effective design of trust ontologies for improvement in the structure of socio-semantic trust networks," *International Journal On Advances in Intelligent Systems*, vol. 1st, no. 1942-2679, pp. 23–42, 2008.

[10] P. Massa and P. Avesani, "Trust-aware collaborative filtering for recommender systems," *Lecture Notes in Computer Science*, vol. 3290, pp. 492–508, 2004.

[11] R. Andersen, C. Borgs, J. Chayes, U. Feige, A. Flaxman, A. Kalai, V. Mirrokni, and M. Tennenholtz, "Trust-based recommendation systems: an axiomatic approach," in *WWW '08: Proceeding of the 17th international conference on World Wide Web*. New York, NY, USA: ACM, 2008, pp. 199–208.

[12] J. O'Donovan and B. Smyth, "Trust in recommender systems," in *IUI '05: Proceedings of the 10th international conference on Intelligent user interfaces*. New York, NY, USA: ACM, 2005, pp. 167–174.

[13] N. Lathia, S. Hailes, and L. Capra, "Trust-based collaborative filtering," in *IFIPTM 2008: Joint iTrust and PST Conferences on Privacy, Trust management and Security*, Department of Computer Science, University College London, London, UK, 2008, p. 14.

[14] J. Weng, C. Miao, and A. Goh, "Improving collaborative filtering with trust-based metrics," in *SAC '06: Proceedings of the 2006 ACM symposium on Applied computing*. New York, NY, USA: ACM, 2006, pp. 1860–1864.

[15] L. A. Goodman and W. H. Kruskal, "Measures of association for cross classifications," *Journal of the American Statistical Association*, vol. 49(268), pp. 732–764, 1954.

[16] DHT, last modified March-2009, Available: http://en.wikipedia.org/wiki/Distributed_hash_table.

[17] R. Bell and Y. Koren, "Scalable collaborative filtering with jointly derived neighborhood interpolation weights," in *In IEEE International Conference on Data Mining (ICDM'07)*. IEEE, 2007, pp. 175–186.

[18] Protégé, copyright 2009 Stanford Center for Biomedical Informatics Research, Available: http://protege.stanford.edu/.

[19] Welkin, copyright 2004-2008 Massachusetts Institute of Technology, Available: http://simile.mit.edu/welkin/.

[20] J. L. Herlocker, J. A. Konstan, L. G. Terveen, and J. T. Riedl, "Evaluating collaborative filtering recommender systems," *ACM Trans. Inf. Syst.*, vol. 22, no. 1, pp. 5–53, 2004.

[21] M. Jelasity, A. Montresor, and O. Babaoglu, "Gossip-based aggregation in large dynamic networks," *ACM Trans. Comput. Syst.*, vol. 23, no. 3, pp. 219–252, 2005.