Trust Based Recommendation Systems

Makbule Gulcin Ozsoy and Faruk Polat
Department of Computer Engineering
Middle East Technical University
Ankara, Turkey
Email: {e1395383,polat}@ceng.metu.edu.tr

***Abstract*—It is difficult for the users to reach the most appropriate and reliable information/item for them among vast number of items and comments related to these items. Recommendation systems and trust/reputation systems are one of the solutions to deal with this problem with the help of personalized services. These systems suggest items to the user by estimating the ratings that user would give to them. Use of trust data for giving recommendation has emerged as a new way for giving better recommendations. In the literature, it is shown that trust based recommendation approaches perform better than the ones that are only based on user similarity, or item similarity. In this paper, a comparative review of recommendation systems, trust/reputation systems, and their combined usage is presented. Then, a sample trust based agent oriented recommendation system is proposed and its effectiveness is justified with the help of some experiments.**

I. Introduction

Advancements in web technologies make users to be able to provide/get wide-range of information on subjects/items and contribute to the society using their experiences. The information is provided by large number of users for large number of subjects/items on different platforms, such as social networks (Twitter, Facebook etc.), personal blog pages, on-line transaction web-sites (Amazon.com etc.), and expertise web-pages (Slashdot.com, Advogato.org etc.). Having this kind of wide-range information leads to difficulties for the user to reach the most appropriate and reliable information/item. Recommendation systems are one of the solutions to deal with information overload problem via providing personalized services. Similarly, trust and reputation systems give support to the users by providing information on how reliable the seller/buyer is in a transaction.

Recommendation systems suggest items to the user by estimating the ratings that the user would give to that item. Recommendation can be given on any subject such as books, movies, news, jokes or vacations. The process of estimating ratings can be performed by using heuristics and machine learning approaches. In the literature there are three base approaches to give recommendations, namely

- content based approaches,
- collaborative filtering approaches and
- and hybrid approaches.

Content based filtering approaches use item similarity to give recommendations. Collaborative filtering approaches use user similarity to decide which item to recommend. Hybrid methods basically combine these approaches to give recommendations.

Trust and reputation systems give support to the users by providing information on how reliable the seller/buyer is in a transaction. The basic idea is to make sellers to rate buyers, and vice versa; and then calculate trust and reputation of users by combining the ratings [46]. In these systems, trust/reputation scores are calculated using different approaches such as simple summation, averaging, Bayesian systems, belief models, flow methods [22]. These systems can be centralized or decentralized.

Trust based recommendation systems are based on giving recommendation using only trust scores or combination of trust and similarity scores while giving suggestions. [12] states that users prefer to receive recommendations from people they know or they are similar and trust based recommendation approaches perform better than approaches that are only based on user similarity.

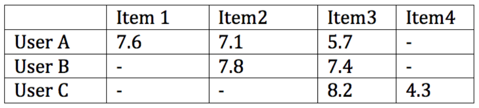
In this paper, a trust-based recommendation system where agents reach information and filter them using their trust relationships is proposed. The system uses the ideas presented in Social Network-based Item Recommendation (SNIR) [17]. The recommendation system implemented in this work is used for recommending movies. Besides SNIR, the movie recommendation system uses reliability measure and estimated scores for giving better recommendations.

The rest of this paper is organized as follows: information on recommendation systems and trust/reputation systems is given in sections II and III, respectively. In section IV, trust based recommendation systems are explored. In section V a trust based agent oriented recommender system is detailed. In section **??**, extension ideas which introduced in several different papers are summarized. The paper is concluded in section VI.

II. Recommendation Systems

The goal of recommendation systems is defined as suggesting a user the items that might be of interest for him/her [30]. Recommender systems are based on information retrieval, approximation theory, cognitive science and emerged as an independent area in mid 1990's with the focus on ratings structures [2]. The most common form of giving recommendation is estimating rating of un-seen/un-rated items by the user and choosing the items with the highest estimation values [46]. The process of estimating ratings can be performed by using domain knowledge (heuristics) and machine learning approaches.

In the following sections, fundamentals of recommendation systems (in section II-A), existing approaches (in section II-B)



	Item 1	Item2	Item3	Item4
User A	7.6	7.1	5.7	-
User B	-	7.8	7.4	-
User C	-	-	8.2	4.3

Fig. 1. *User x Item* matrix

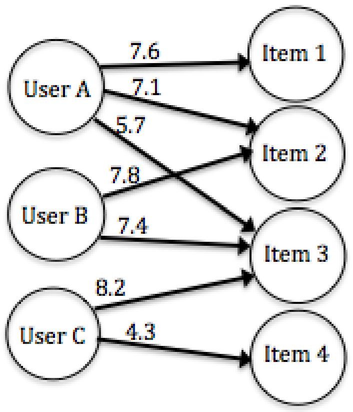


Fig. 2. *User x Item* graph

and the challenges and future research directions (in section II-C) are given.

A. Fundamentals

The data to be processed by a recommendation system has basically three elements, which are *user*, *item* and *rating*. In most of the algorithms, these elements are represented by a matrix, where rows indicate *users*, columns indicate *items* and matrix entries indicate the ratings. An example $user \times item$ matrix is given in Figure 1 with three users and four items and the ratings of them. The items may have various features in the implementation, such as gender and/or age for users, price and/or year for items, time and/or location for ratings.

Alternatively, graph representation of $user \times item$ matrix can also be used. In graph representation, users and items are represented with nodes and the weighted edges relate users with items, forming a bi-partite graph. An example graph is given in Figure 2, which shows the same system as Figure 1.

B. Recommendation Approaches

In the literature there are three basic approaches to give recommendations, namely content based, collaborative filtering and hybrid approaches. While content based approach uses item similarity to give recommendations, collaborative filtering approach uses user similarity to decide which item to recommend. Hybrid methods basically combines previous two approaches to give recommendations. The details of these approaches are explained in the following sub-sections, II-B1, II-B2, II-B3, respectively. Other approaches are introduced in the forthcoming sub-section II-B4.

1) Content Based Approach: Content based approach uses item similarity to give recommendations. Given the users' previous ratings to some of the items, the similarity between rated and candidate items is calculated. For example, if a user rated a comedy movie before, the system would recommend unwatched comedy films to the user. In order to calculate similarity, the system needs features defining items. For example, a movie recommendation system may need to collect information on genre, year, actors, director, etc.

Content based recommendation may use heuristics-based and machine learning approaches. Heuristic-based approaches may use weights of features, tf-idf (term frequency-inverse document frequency) values and etc. Machine learning based approaches may use Bayesian classifiers, decision trees, clustering [31], [37].

Content based approach suffers from some limitations. First, it needs feature of items, which is not available most of the time. Collecting the features and deciding which features are important involves a significant knowledge-engineering effort [34] and may not be practical all times [2]. Second problem is about distinguishing items. If two items are represented with the same features somehow, there is no way to distinguish these two items from each other. Third, it can become repetitive, such that since only similarity of items is considered, no item on different subject is recommended to the user. For example, if user continuously reads/rates news on health, that user is never recommended news in politics. So, there is a need for randomness [2], which can not be provided by content based approaches. Last, new users may suffer from not being recommended accurately, since they have not rated enough number of items. Note that this problem is known as *cold start* problem in the literature.

2) Collaborative Filtering Approach: Collaborative filtering approach uses user similarity to decide which item to recommend. In this approach, similar users to the user in taste are identified, and then the items rated by these identified users (neighbors) are recommended. This approach relies only on neighbors' opinions, and does not use any information on items' features. GroupLens [24] is one of the first collaborative filtering recommendation system, which recommends movies. Other examples are the book recommendation system from Amazon.com, and the Jester system that recommends jokes [15].

The most widely used used collaborative filtering approach is k-Nearest Neighbours(kNN) method [6], [41]. In this method, first, k most similar users are identified. Then, the rating for the item which is not rated by the user, but rated by neighbors is calculated. For this calculation weighted/not-weighted average of the neighbors rating are calculated. At the end, the items with the highest scores are chosen. Besides kNN, clustering methods [7], Bayesian networks [7], Artificial Neural Networks [4], SVD [4], Probabilistic Latent Semantic Analysis(PLSA) [44], Latent Drichlet Allocation(LDA) [29] are some of the methods used for collaborative filtering.

Collaborative filtering approaches suffer from some limitations. First, similar to content based approaches, new users may suffer from not being recommended accurately, since they have not rated enough number of items and similar neighbours can not be found. Second, similar to new user, new items become problematic. If a new item introduced to the system, it can not be recommended until it is rated by some user. The item may wait forever for being rated. Third, data sparsity causes problems related to finding similar users. Since number of items is very large against the number of users in a system, it is unlikely that there will be many common item ratings for two different users. Similarly, if a user has some uncommon tastes, it is harder for that user to have accurate recommendations [2].

3) Hybrid Approach: Hybrid methods basically combines content based and collaborative filtering approaches to give recommendations [38]. These systems can be designed in four ways [2]:

- Separate: Implement sub-parts(content-based and collaborative filtering) separately and combine their rating estimations. These systems can use linear combination [9] or voting schema [36].
- Content based in collaborative filtering: Include some characteristics of content based into collaborative filtering approach. These approaches maintain the content-based profiles for each user and used this information to find out neighbours [3], [36].
- Collaborative filtering in content based: Include some characteristics of collaborative filtering into content based approach. Most common approach for this is dimensionality reduction, for example using latent semantic indexing (LSI) [33].
- Unified: Construct a unified model using both content based and collaborative filtering approaches. Some example approaches uses probabilistic latent semantic analysis [42] and Bayesian mixed-effects regression models [10].

Hybrid approaches are usually based on probabilistic methods [5] such as genetic algorithms [18], neural networks [39], Bayesian networks [11], clustering [43] and latent features [28].

4) Other Approaches: Besides the three main approaches explained above, [8] extended the categorization by including demographic and knowledge-based approaches. Demographic recommendation systems uses demographic (e.g. age) information to categorize users and give recommendations. Knowledge-based recommendation systems infers users' need and preferences, then gives recommendation accordingly. The difference of this system from other recommendation systems is its knowledge of mapping between items and users' needs.

C. Challenges and Improvements

The expectations from a recommender system are *accuracy*, *coverage*, *novelty*, *diversity*, *stability*, *resistance to attacks* [5], [34]. These requirements can be explained as follows:

- Accuracy: Giving good recommendations and estimations.
- Coverage: Capacity of predicting rating of an item.
- Novelty: Degree of difference between recommended and known items.
- Diversity: Degree of differentiation among recommended items.
- Stability: Not strongly changing recommendations in a short period of time.
- Resistance to attacks: Not being affected by attacks (e.g. copy-profile attack [30])

The main challenges a recommender system usually face are:

- Data sparsity: It is unlikely that two users rated same items many times. This makes it harder to calculate similarity.
- Cold start for user: The user who has not rated sufficiently many items may suffer from not getting accurate results. This may occur since similar users to this user can not be found.
- Cold start for item: The newly introduced item may suffer from not getting enough ratings, which leads to not being recommended to anyone.
- Attacks: There can be attacks to the recommendation system, such as copying the users whole profile and make system to think that the attacker and the user are very similar. This way an attacker can fool the system and make it suggest any item rated by attacker to the user.

[5] has divided the evolution of recommendation systems into three phases. In the first phase, explicit ratings provided by users are used, and mostly content based approaches are used. In the second phase, with Web 2.0, additionally social information elated to the users are used. In the third phase, with Web 3.0, the writers conjecture that context-aware information from mobile devices and sensors will be used. They state that use of location-aware recommendation systems are emerging, and similar other systems will emerge in future. In [1], it is already shown that using time (when), location (where), accompaniments (with whom) information, the performance of a recommendation system can be improved.

[2] suggests many areas to be investigated. Some of the examples are representing user/item information better, improving more advanced recommendation models, using information from multiple resources, utilizing of multi-criteria ratings, developing more flexible methods, and measuring performance based on different criteria.

III. Trust and Reputation Systems

Trust and reputation systems give support to the users by providing information on how reliable the seller/buyer is in a transaction. The basic idea is to make sellers to rate buyers, and vice versa; and then calculate trust and reputation of users by combining the ratings [46]. The obtained trust and reputation values are used as assistant in future transactions among users.

In section III-A, definitions of trust and reputation are given. In section III-B, the network architectures for reputation and trust systems are identified. These architectures determine how trust and reputation scores are shred and calculated in the system [22]. In section III-C, different methods used for trust and reputation score calculations are detailed. Similarities and differences of trust and reputation systems with collaborative filtering systems are identified in section III-D. Finally, the challenges for trust/reputation systems are explained in section III-E.

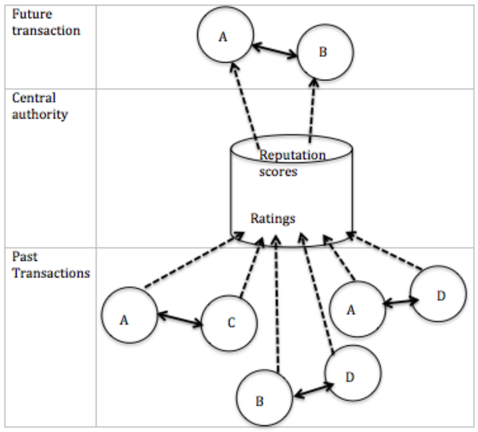


Fig. 3. Centralized architecture

A. Definitions

Giving definition of trust is hard, since it is used with different meanings in daily life [22]. In [22], trust is defined in two categories *reliability trust* and *decision trust*.

- Reliability trust: ”The subjective probability by which an individual, A, expects that another individual, B, performs a given action on which its welfare depends”. This definition highlights *dependency* and *reliability (probability)*.
- Decision trust: ”The extent to which one party is willing to depend on something or somebody in a given situation with a feeling of relative security, even though negative consequences are possible”. This definition highlights *dependency*, *reliability*, *utility* and *risk*.

Reputation is defined as ”what is generally said or believed about a person’s or thing’s character or standing” [22]. In the same paper, it is indicated that reputation is derived by collecting trustworthiness measure provided by referrals/ratings given in a community, and subjective trust is derived by combining received referrals/rating and personal experience.

B. Trust/Reputation Networks

The network architectures for reputation and trust systems determine how trust and reputation scores are shared and calculated in the system [22]. The main types are centralized and distributed architectures, which are explained in sections III-B1 and III-B2 respectively.

1) Centralised Architecture: Each individual who got involved in the transaction sends rating of each other to a central authority. The overall reputation score for each individual is calculated by this central authority and provided to the other individuals publicly. The Figure 3 shows an example centralized architecture. In this figure, past transactions; namely transactions between A and C, B and D, A and D; sends ratings to the central authority. In central authority the reputations of the individuals are calculated, and given to the other individuals in future transactions.

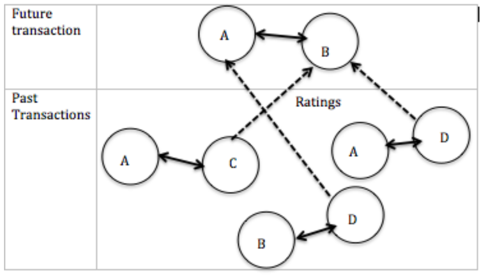


Fig. 4. Distributed Architecture

2) Distributed Architecture: Each individual records the ratings of other individuals, and calculates reputation scores of others locally. This information is provided to other individuals only on request. The individual (A) who wants to get involved with another individual (B) finds the other individuals who already have an experience with B, and requests its reputation scores. Then the requesting individual (A) combines all collected reputation scores to decide if B is reliable or not. The Figure 4 shows an example of distributed architecture. In the distributed architecture, each individual collects ratings of others, and provides these ratings whenever another individual requests.

C. Trust/Reputation Computation Methods

The individual can decide the trust/reputation score of other individuals by direct experience (first-hand experience) and/or by getting information of others’ experience (second-hand experience). There are various different score calculation methods in the literature, such as simple summation, averaging, Bayesian systems, belief models, flow methods [22].

In simple summation method, the difference of total positive scores and negative scores are calculated. This is what eBay uses for reputation calculations [40]. In averaging method, the average of the ratings are calculated, which Amazon (mainly an on-line book-store) and Epinions (a product and shop review site) use [22].

In Bayesian systems, the estimation of reputation score is statistically calculated. For the calculations the combination of the new score with the previous rating scores are used [32], [50].

Belief models are related to probability theory, where summation of the probabilities can be less than 1 and the remaining can be considered as uncertainty [22]. The reputation score based on belief model can be calculated by using a combination of beliefs and uncertainty of the individuals [20], [51].

In flow methods, the trust/reputation scores are calculated using some iteration through long chains, in which incoming flow increases reputation and outgoing flow decreases it. The Google PageRank [35], or EigenTrust model [25] are the examples for the flow method. Figure 5 contains an example for trust/reputation score calculation using a flow method. In the figure, the individual A trusts B, and B trusts C. Using the flow method, the system can infer that A can also trust C.

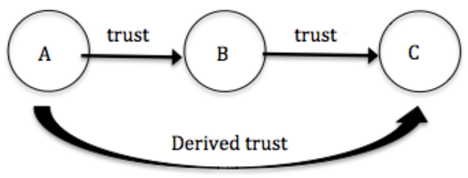


Fig. 5. Derived trust using flow methhod

D. Collaborative Filtering and Trust/Reputation Systems

Collaborative Filtering systems are similar to trust/reputation system since both use information provided by the members of the community. However, in collaborative systems the base assumption is that users' rating of items are subjective, so it is assumed that if two user are similar then their subjective tastes are similar.

Unlike collaborative filtering systems, the basic assumption of the trust/reputation system is based on assignment of consistent (objective) rating scores as a result of a transaction [22]. In [22], an example to this situation is given: People judge a music file with viruses as bad consistently, but some people may find that music good, or bad. But it also indicated in the same paper that some collaborative filtering techniques can be consistent (objective) or some of the trust/reputation systems can be subjective.

E. Challenges

Researchers have worked on the trust and reputation systems widely, but there is limited work focusing on robustness analysis of these systems [21]. [19] and [23] are the recent papers on this subject.

Some of the defined threats to the trust/reputation systems are playbook, unfair ratings, discrimination, proliferation, re-entry, and the sybil attack [21]. In playbook threat, after performing actions that increase the reputation, the user can use his/her reputation to profit from it by providing low quality services to others. In unfair rating attacks, the user can give ratings to the services/others that do not represent the real opinion. In discrimination attacks, the user can favour one group of ratings, and give lower ratings to the others. In proliferation attacks, a service can be offered from multiple channels to the user. When a user is offered multiple similar services, he/she chooses service randomly. With proliferation attacks, the seller increases the probability of being chosen. In re-entry attacks, the user with low reputation can leave the community and re-enter, so that he/she can avoid the consequences of the low score. In sybil attack, a user can create multiple copies of his/her identity, and give multiple ratings to the services. This increases the effect of a single user in the system unfairly.

IV. Trust Based Recommendation Systems

Recommendation systems aim to estimate the ratings of an item that a user would give to that item and suggest the items with the highest rating scores. The most commonly used approach is collaborative filtering, which finds similar users(neighbours) to the user and consider items that are used by the neighbours for recommendation. However, this approach may give poor results, because of the data sparsity and cold start problems. In order to solve these problems and give more reliable recommendations trust based recommendation systems are introduced. Lately, researchers tend to incorporate distrust information to these systems too.

In the following two sections, information on the literature related to trust based recommender systems and and the use of distrust is given, in section IV-A and section **??** respectively.

A. Using Trust in Recommendation Systems

It is empirically shown that people rely recommendation from trusted fellows; such as friends, family members; more than recommendation provided by an automatic recommendation system [45]. In [52], it is shown that there is positive correlation between trust and interest similarity. Similarly, [12] states that users prefer to receive recommendations from people they know or they are similar. This indicates that the use of trust/reputation systems together with other recommendation systems; e.g. collaborative filtering; may may increase the performance.

One of the first studies on trust based recommendation systems belong to Raph Levin, Advogato project [26]. In this project, the central authority decides the reputation of the individuals.The individuals are labelled as trusted and untrusted in this system.

In [34], the trust value is calculated implicitly, by comparing the previous recommendation ratings and actual ratings. They have considered past ratings recommender and recommended users. They have compared the previous ratings of the users and decide if one user can predict the correct ratings of items that will be recommended to another user. They have named the percentage of correct predictions over all items as profile-level trust value. Besides profile-level trust, the system also calculated the item-level trust, which measures the percentage of correct recommendations for an item.

In [13], social networks and trust are combined to give recommendations using a website, FilmTrust, which is created by the researchers for this purpose. The FilmTrust system asks its users to explicitly assign trust values to other users in order to indicate following/not following information. The writers of the paper indicate that trust usage adds quality to the recommendations. In another work [12], The same researchers stated that there is a relationship between trust and user similarity, and trust based recommendation approaches performs better than approaches that are only based on user similarity.

In [30], a concept of "web of trust" is used. In this system, the users give explicit trust scores to each other. Also, scores to the items given by users, as in the collaborative filtering, is collected. Then, item ratings are predicted, using mainly trusted friends recommendations. At the end, the items with highest prediction are recommended to the users. Using this approach, the coverage of the recommender system is expanded, the quality of recommended items is increased, the cold-start problem is solved for new-users and reliability of recommendations is increased.

[48] combines social networks and trust calculations to give recommendation by using multi-agent systems. They have used a trust propagation mechanism along paths in the given social network. They have analysed the performance of their framework on different networks with different density, homogeneity and data sparseness. Also, unlike previous works, they have defined trust as utility of agent, not only similarity of preferences. They also stated in their work that evolution and robustness are two directions that need further research.

Lately, [46] also proposed to use a combination of recommendation systems, social networks and trust/reputation systems to create better recommendations and to deal with problems observed in pure recommendation systems.

B. *Using Distrust in Recommendation Systems*

Most of the works related to trust based recommender systems are interested in trust information but discard distrust information. [16] states that distrust is at least important as trust in real-world recommendation systems.

[14] is one of the first works in the literature that uses distrust in recommendation. In this work the trust values are given in range of 1-9, where 9 indicates absolute trust and 1 indicates absolute distrust. They have assumed that trust and distrust are symmetrical opposite concepts.

Unlike [14], [27] indicates that distrust is totally different than trust and known trust based methods may not be simply converted into distrust based systems. In their paper they state that most common method in trust based systems is to propagate trust, but it is unreliable in distrust based systems. For example, an enemy of an individual's enemy is not necessarily another enemy of that person. In [27], trust and distrust information given by users explicitly is used together to give recommendations. They have modelled trust and distrust separately and then combined the results. They have shown that usage of distrust in recommender systems increases performance. They indicate that modelling trust and distrust together, and using social networks to get the information can be an interesting idea.

In [47], distrust is used in three different ways. First one is to use distrust for filtering out the unwanted users from recommendation process. Second one is to use it for debugging false positives. For example, if *a* trusts *b* and *b* trusts *c*, a trust propagation model would find out that *a* trusts *c*. But considering the fact that *a* does not trust *c*, this result would be a false positive. The last, third, way of distrust usage modelled in [47] is to use distrust as a negative indication of trust. As a result of their work, they have indicated that the first two methods are promising but the last one is not a good choice to use.

V. A Trust-Based Agent Oriented Recommender System

In this section, a trust based agent oriented recommendation system is proposed. In this system agents reach information and filter them using their trust relationships to recommend movies. The system uses the ideas presented in SNIR [17]. The movie recommendation system uses reliability measure and estimated scores for giving better recommendations.

SNIR is detailed in section V-A. In section V-B, the proposed movie recommendation system is explained. Lastly, in section V-C, the evaluation results for the proposed method is given.

A. *Social Network-based Item Recommendation (SNIR)*

In [17], it is stated that many recommendation systems consider only one similarity/trust value for each neighbor user. They stated that in reality a user may consider more than one similarity/trust value for each neighbor. In order to detail their idea, they proposed Social Network-based Item Recommendation (SNIR).

Social Network-based Item Recommendation (SNIR) system is one of the collaborative filtering systems. In this system, they aimed to create an agent-based recommendation system that uses past preferences of the user and the other users who are in the social network of the related user. Also they have introduced a topic-based trust value between users to enhance the performance of the recommendation system. They indicate that, similar users may share different opinions/preferences on different topics. For example, user A may give higher trust value to user B for topic T1, and lower trust value for topic T2. In order to evaluate their system, they used Flickr, a photo sharing social web-site.

In SNIR system, each agent has responsibility of deciding its own preferences, processing search query, identifying category of the items and ranking/recommending the products/items. The preference learner module mines the past activities/preferences of the users. The query processor module collects related items to the given query. The category identifier module decides the categories of the collected items. Ranking/recommending module decides the likelihood that the user will like the items and ranks them.

For the calculations related to category identification and ranking/recommendation modules, the following entities are introduced:

- A set of users: U = $\{u_1,...,u_m\}$
- A set of items: I = $\{i_1,...,i_n\}$
- A set of tags: T = $\{t_1,...,t_o\}$
- A set of categories: C = $\{c_1,...,c_r\}$

Each user likes one or more items. The items are described by one or more tags. The system considers a set of categories.

The category of an item is decided using a probabilistic approach. For each category a dictionary is defined, which contains related tags. Each item is attached to a list of tags. The order of the tag in the list indicates the importance of that tag for the given item. So, for each tag a weight is calculated according to its position. In Eq 1, t^i_j indicates the j^{th} tag for item *i*. l_i is the length of the tag list of the item *i*. α and γ are constants in range of (0,1).

$$w(t^i_j) = \frac{1 - \alpha * j^\gamma}{\sum_k^{l_i} 1 - \alpha * k^\gamma} \tag{1}$$

The probability of an item i belonging to category c_x is calculated as in Eq 2. *lookup(d, t)* value is equal to 1 if tag t is listed in dictionary d, and its value is 0 otherwise.

$$P(cat_i = c_x) = \frac{\sum_{k=i}^{l_i} w(t_k^i) * lookup(d_{c_x}, t_k^i)}{\sum_{c_y \in C} \sum_{k=i}^{l_i} w(t_k^i) * lookup(d_{c_y}, t_k^i)} \quad (2)$$

Most of the recommender systems in the literature aim to find out items to recommend. In SNIR, the aim is different. In this method, the candidate items are known and the method ranks those items to recommend. Also, in SNIR the network of users is known beforehand, such that the user has already indicated who his/her friends are in the social network.

Having a list of candidate items, the goal of SNIR is to calculate probability that user *a* likes an item *i* which is posted by user *b*, who is a neighbour in the social network. The equation can be re-written as in Eq 3 using Bayesian Theorem.

$$P(likes(u_a, i) \mid i \in postby(u_b)) = \frac{P(i \in postby(u_b) \mid likes(u_a, i))}{P(i \in postby(u_b))} \quad (3)$$

In SNIR, it is stated that content information can be introduced to the equation, since both recommender and recommended users consider the content while choosing items to recommend/use. A user prefers an item according to its recommender and its content (Eq 4). Second part of Eq 4, can be re-written using Bayesian Theorem (Eq 5). Also, a recommender prefers to recommend same kind of items (Eq 6).

$$P(i \in postby(u_b) \mid likes(u_a, i)) = \sum_{c_x \in C} P(i \in postby(u_b) \mid cat_i = c_x, likes(u_a, i)) \, P(cat_i = c_x \mid likes(u_a, i)) \quad (4)$$

$$P(cat_i = c_x \mid likes(u_a, i)) = \frac{P(likes(u_a, i) \mid cat_i = c_x) P(cat_i = c_x)}{P(likes(u_a, i)} \quad (5)$$

$$P(i \in postby(u_b)) = \sum_{c_x \in C} P(i \in postby(u_b) \mid cat_i = c_x) P(cat_i = c_x) \quad (6)$$

Combination of equations- Eq 3, Eq 4, Eq 5, Eq 6- the probability that user a likes an item i which is posted by user b can be calculated as in Eq 7.

$$P(likes(u_a, i) \mid i \in postby(u_b)) = \frac{\sum_{c_x \in C} P(i \in postby(u_b) \mid cat_i = c_x, likes(u_a, i)) \, P(likes(u_a, i) \mid cat_i = c_x) P(cat_i = c_x)}{\sum_{c_x \in C} P(i \in postby(u_b) \mid cat_i = c_x) P(cat_i = c_x)} \quad (7)$$

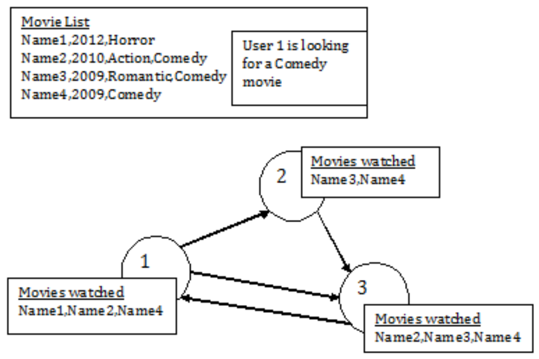


Fig. 6. An example social network with 3 users

[17] have used the SNIR for searching and recommending photos on Flickr, a photo sharing social web-site. They have assumed that if a comment written for a photo, it means that the user has liked that photo. The probabilities in Eq 7 are calculated as follows:

- $P(i \in postby(u_b) \mid cat_i = c_x, likes(u_a, i))$ = (number of photos posted by u_b that belong to c_x and are commented on by u_a) / (number of all photos that are in c_x and are commented on by u_a)
- $P(likes(u_a, i) \mid cat_i = c_x)$ = (number of all photos that are in c_x and are commented on by u_a) / (number of all photos that are in c_x)
- $P(i \in postby(u_b) \mid cat_i = c_x)$ = (number of photos posted by u_b that are in c_x) / (number of all photos that are in c_x)

The evaluation results of SNIR method show that it performs better than content based approaches and other recommendation schemas.

B. SNIR For Movie Recommendation

A new SNIR based movie recommendation is proposed in this paper. For this purpose an artificial social-network is created, in which users choose their friends and gives scores to the movies they have watched. The list of movies can be reached by all users while giving decision on a movie to watch. The list of movies contains the name of the movie, the year and the genre information. Genre information contains a list of genres. An example social network with three users can be seen in Figure 6. In this network the direction of arcs indicate the friendship information decided by the user at the beginning.

In order to explain how the method works, the example in Figure 6 will be used. In this system, there are three different users, who have indicated their relationship to each other. There are four different movies whose genres are combinations of Horror, Action, Comedy and Romantic. The users have already watched some of the movies and have scored those movies. Also, a like/dislike threshold value is introduced to the system. If the given score of the movie is less than the like/dislike threshold, it is assumed that the user has disliked that movie.

TABLE I. THE RESULT FOR USER1S QUERY: RECOMMEND ME A COMEDY MOVIE.

Movie	User2	User3
Name3	0.20	0.33

As an example, User1 wants to watch a movie which is Comedy. Using the Eq 7, the preference measure for each movie-user pair is calculated. This calculation is performed on the movies that user has not seen, that the neighbor user has seen and that are in the same genre as asked in the query. The result is given in Table I.

Note that for the calculations $P(cat_i = c_x)$ is not calculated as given in Eq 2. The probability of an item i is in category x is assumed to be equal, and is set to $(1/numberOf(genresOfMovie))$. For example, for movie Name3, the $P(cat_{Name3} = c_{romantic}) = 0.50$ and $P(cat_{Name3} = c_{comedy}) = 0.50$.

After the calculations, the movies that have the highest preference measure values and have not been seen by the user, User1, are recommended. In the example, movie Name3 is suggested to the user.

It can be observed from the Table I that the preference measure calculated by each neighbor users may differ. If the number of movies was higher and the difference between preference measures was larger, the decision of suggesting the movie, Name3, may become harder. For example, in the case where User2 recommends a movie with 0.98 and User3 recommends the same movie with 0.12, and there are many more choices to be suggested- which has larger preference measure than 0.12, what should the system do? In order to solve this problem, a reliability measure is introduced to the system. The reliability of users can be given beforehand, or can be updated according to past recommendations. In this project, second choice is preferred, and the reliability is updated according to the given recommendations. $P(r(u_b))$ represents the reliability of the neighbor user, u_b. Following the ideas in the [49], this probability can be calculated as in Eq 8, where *m* indicates the number of times that recommendation given by u_b is satisfied the user, and *n* indicates the total number of recommendations given by u_b.

$$P(r(u_b)) = m/n \tag{8}$$

Using the equations, Eq 7 and Eq 8, overall probability that user *a*, u_a, likes item *i* can be calculated as in Eq 9, where u represents the neighbours of the u_a.

$$P(likes(u_a, i) \mid i \in postby(u_b)) = \frac{\sum_u P(r(u))P(likes(u_a, i) \mid i \in postby(u))}{\sum_u P(r(u))} \tag{9}$$

For the example given in Figure 6, assume User1 finds User2 as 0.70 reliable and User3 as 0.30 reliable. Having the values given in Table I, movie Name3 will be suggested to the User1 with 0.24.

Another idea that can be applied to the recommendation system is to give the probable score that the user will give to the recommended movie, as provided in [13]. Using the scores given to the watched movies, the estimated score that the user will give to the recommended movie can be calculated. For this purpose the given movie scores are multiplied by reliability scores of the neighbours, Eq 10.

$$EstimatedScore(u_a, i) = \frac{\sum_u P(r(u))score(u, i)}{\sum_u P(r(u))} \tag{10}$$

For the example, assume User2 gives 7/10 for movie Name3, and User3 gives 9/10. The estimated score that user a will give to Name3 will be 7.6/10; assuming increment of score values is 0.1.

C. Evaluation Results

The evaluation of the proposed method is performed using the HetRec 2011 Dataset. It is an extension of MovieLens10M dataset. In this dataset, only those users with both rating and tagging information have been maintained from the original dataset. In the dataset there are 2113 users, 10197 movies and 20 different genres. All movies are assigned to one or more genres. All users have watched one or more movies and assigned rating scores to those films. The ratings of the movies are given in range [0, 5] with 0.5 increments. The total number of ratings is 855598.

In the dataset, there is no information on relationship among users (such as being friend), which is a prerequisite for this work. In order to have this kind of relationship each user is assigned to a randomly chosen number of friends (other users) which are also chosen randomly. The maximum number of friends that a user may have is given as 5, for the evaluation.

After having all information related to the users, relationships, and movies; each user is designed to ask for 5 different movie recommendations from each genre. This means that the recommendation system is run for $(numberOf(Genres) * 5 * numberOf(Users))$. To be able to compare the recommendation results, the list of already watched films is divided into two parts. The first part is used as a kind of training set, which is used for calculations. The second part is used for test purposes, and used for comparing recommended movies to the actually watched movies.

For each recommendation request, a list containing 10 different movies is introduced to the user. This list is assumed to be a single suggestion (= one group of suggestions). If there is no match between the list of recommended movies and second part of movies already watched by the user (test set), then number of negative recommendations is increased by one. If there is a match, for each match a delta score is calculated between the recommended score and the actual rating score. For each match, the number of true recommendations is increased by one. Using these values, for each user, *accuracy* and *mean absolute error* values are calculated.

$$Accuracy = 1 - \frac{numberOf(NegativeRecommendations)}{numberOf(TotalRecommendations)} \tag{11}$$

TABLE II. THE OVERALL RESULT

Avg. Accuracy	Avg. Mean Error Rate
0.073	0.814

$$MeanAbsoluteError = \frac{\sum_{i \in RecommendedMovies} (actualScore_i - recommendedScore_i)}{numberOf(TrueRecommendations)} \quad (12)$$

After the calculations of these scores for all users, the average scores are calculated. The overall result is given in Table II .

The accuracy value gives information on the performance on the process of choosing the recommended movies. Since half of the movies are used for test purposes, it can be said that the negative matches can still be good recommendation; even they are not in the original list. In order to have better information on accuracy, an evaluation system based on user survey/an on-line evaluation may be used.

The mean absolute rate gives information on how well the recommendation scoring works. If the actual score and the calculated are similar, then the system performs well.

VI. CONCLUSION

Advancements in the technology made people to move their social activities into the web. Today, people share information/comments, buy/sell items electronically, find new movies, books etc. that are interesting for them on web pages. This lead to information overload, where users can not always decide which information, or the item is the most appropriate for them. Recommendation systems are one of the solutions to deal with this problem via providing personalized services. Similarly, trust and reputation systems give support to the users by providing information on how reliable the seller/buyer is in a transaction. Combination of these techniques, namely trust based recommendation systems, performs well in giving suggestions to the users.

Recommendation systems suggest items (e.g. books, movies, news, jokes, or vacations) to the user by estimating the ratings that user would give to them. In the literature, there are three main approaches to give recommendations, namely content based, collaborative filtering and hybrid approaches. Content based filtering approach uses item similarity to give recommendations. Collaborative filtering approach uses user similarity to decide which item to recommend. Hybrid methods attempt to combine these approaches to give better recommendations.

Trust/reputation systems provides reliability scores of sellers/buyers, which is calculated by scores given by users of the transactions. These systems can be constructed in centralized and distributed architectures, in which trust/reputation scores are calculated using different approaches such as simple summation, averaging, Bayesian systems, belief models, flow methods [22].

Trust based recommendation systems are based on giving recommendation using only trust scores or combination of trust and similarity scores while giving suggestions. [12] states that these systems perform better than approaches that are only based on user similarity. In the literature not only trust but also distrust information is used to give better recommendations.

Motivated by the challenges of combining recommendation systems and trust/reputation systems, a trust based agent oriented recommender system is proposed in this paper. The proposed method is based in SNIR [17] and used for giving movie recommendations. During the implementation, it is observed that a movie may be recommended by more than one neighbor with different preference measures. In order to combine the recommendation of the different neighbors, reliability score for each neighbor is introduced. Also, it is observed that score that the recommended user will give to the movie can be estimated.

REFERENCES

[1] G. Adomavicius, R. Sankaranarayanan, S. Sen, and A. Tuzhilin, "Incorporating contextual information in recommender systems using a multidimensional approach," *ACM Trans. Inf. Syst.*, vol. 23, no. 1, pp. 103–145, 2005.

[2] G. Adomavicius and A. Tuzhilin, "Towards the next generation of recommender systems: A survey of the state-of-the-art and possible extensions," *IEEE Trans. Knowl. Data Eng.*, vol. 17, no. 6, pp. 734–749, 2005.

[3] M. Balabanovic and Y. Shoham, "Content-based, collaborative recommendation," *Commun. ACM*, vol. 40, no. 3, pp. 66–72, 1997.

[4] D. Billsus and M. J. Pazzani, "Learning collaborative information filters," in *ICML*, 1998, pp. 46–54.

[5] J. Bobadilla, F. Ortega, A. Hernando, and A. Gutirrez, "Recommender systems survey," *Knowledge-Based Systems*, no. 0, pp. –, 2013. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0950705113001044

[6] J. Bobadilla, A. Hernando, F. Ortega, and J. Bernal, "A framework for collaborative filtering recommender systems," *Expert Syst. Appl.*, vol. 38, no. 12, pp. 14 609–14 623, 2011.

[7] J. S. Breese, D. Heckerman, and C. M. Kadie, "Empirical analysis of predictive algorithms for collaborative filtering," in *UAI*, 1998, pp. 43–52.

[8] R. D. Burke, "Hybrid recommender systems: Survey and experiments," *User Model. User-Adapt. Interact.*, vol. 12, no. 4, pp. 331–370, 2002.

[9] M. Claypool, A. Gokhale, T. Miranda, P. Murnikov, D. Netes, and M. Sartin, "Combining content-based and collaborative filters in an online newspaper," in *Proceedings of the ACM SIGIR '99 Workshop on Recommender Systems: Algorithms and Evaluation.* Berkeley, California: ACM, 1999.

[10] M. K. Condliff, D. D. Lewis, and D. Madigan, "Bayesian mixed-effects models for recommender systems," in *In ACM SIGIR 99 Workshop on Recommender Systems: Algorithms and Evaluation*, 1999.

[11] L. M. de Campos, J. M. Fernández-Luna, J. F. Huete, and M. A. Rueda-Morales, "Combining content-based and collaborative recommendations: A hybrid approach based on bayesian networks," *Int. J. Approx. Reasoning*, vol. 51, no. 7, pp. 785–799, 2010.

[12] J. Golbeck, "Trust and nuanced profile similarity in online social networks," *TWEB*, vol. 3, no. 4, 2009.

[13] J. Golbeck and J. Hendler, "Filmtrust: Movie recommendations using trust in web-based social networks," *Proceedings of the IEEE Consumer communications and networking conference*, vol. 96, 2006.

[14] J. Golbeck, B. Parsia, and J. Hendler, "Trust networks on the semantic web," in *In Proceedings of Cooperative Intelligent Agents*, 2003, pp. 238–249.

[15] K. Y. Goldberg, T. Roeder, D. Gupta, and C. Perkins, "Eigentaste: A constant time collaborative filtering algorithm," *Inf. Retr.*, vol. 4, no. 2, pp. 133–151, 2001.

[16] R. V. Guha, R. Kumar, P. Raghavan, and A. Tomkins, "Propagation of trust and distrust." in *WWW*, S. I. Feldman, M. Uretsky, M. Najork, and C. E. Wills, Eds. ACM, 2004, pp. 403–412.

[17] A. Gürsel and S. Sen, "Producing timely recommendations from social networks through targeted search," in *AAMAS '09: Proceedings of The 8th International Conference on Autonomous Agents and Multiagent Systems*. International Foundation for Autonomous Agents and Multiagent Systems, 2009, pp. 805–812. [Online]. Available: http://portal.acm.org/citation.cfm?id=1558109.1558123

[18] Y. Ho, S. Fong, and Z. Yan, "A hybrid ga-based collaborative filtering model for online recommenders," in *ICE-B*, 2007, pp. 200–203.

[19] K. J. Hoffman, D. Zage, and C. Nita-Rotaru, "A survey of attack and defense techniques for reputation systems," *ACM Comput. Surv.*, vol. 42, no. 1, 2009.

[20] A. Jøsang, "A logic for uncertain probabilities," *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, vol. 9, no. 3, pp. 279–212, 2001.

[21] A. Jøsang and J. Golbeck, "Challenges for Robust of Trust and Reputation Systems," Sep. 2009.

[22] A. Jøsang, R. Ismail, and C. Boyd, "A survey of trust and reputation systems for online service provision," *Decision Support Systems*, vol. 43, no. 2, pp. 618–644, 2007.

[23] R. Kerr and R. Cohen, "Smart cheaters do prosper: defeating trust and reputation systems," in *AAMAS (2)*, 2009, pp. 993–1000.

[24] J. A. Konstan, B. N. Miller, D. Maltz, J. L. Herlocker, L. R. Gordon, and J. Riedl, "Grouplens: Applying collaborative filtering to usenet news," *Commun. ACM*, vol. 40, no. 3, pp. 77–87, 1997.

[25] K. Kurbel and I. Loutchko, "Towards multi-agent electronic marketplaces: what is there and what is missing?" *Knowledge Eng. Review*, vol. 18, no. 1, pp. 33–46, 2003.

[26] R. Levien, A. Aiken, R. Levien, and A. Aiken, "Attack resistant trust metrics for public key certification," in *In 7th USENIX Security Symposium*, 1998.

[27] H. Ma, M. R. Lyu, and I. King, "Learning to recommend with trust and distrust relationships." in *RecSys*, L. D. Bergman, A. Tuzhilin, R. D. Burke, A. Felfernig, and L. Schmidt-Thieme, Eds. ACM, 2009, pp. 189–196.

[28] S. Maneeroj and A. Takasu, "Hybrid recommender system using latent features," in *AINA Workshops*, 2009, pp. 661–666.

[29] B. M. Marlin, "Modeling user rating profiles for collaborative filtering," in *NIPS*, 2003.

[30] P. Massa and P. Avesani, "Trust-aware recommender systems," in *RecSys*, 2007, pp. 17–24.

[31] R. J. Mooney and L. Roy, "Content-based book recommending using learning for text categorization," in *ACM DL*, 2000, pp. 195–204.

[32] L. Mui, M. Mohtashemi, and A. Halberstadt, "A computational model of trust and reputation," 2002. [Online]. Available: http://csdl.computer.org/comp/proceedings/hicss/2002/1435/07/14350188.pdf

[33] I. S. C. Nicholas and C. K. Nicholas, "Combining content and collaboration in text filtering," pp. 86–91, 1999.

[34] J. O'Donovan and B. Smyth, "Trust in recommender systems," in *IUI*, 2005, pp. 167–174.

[35] L. Page, S. Brin, R. Motwani, and T. Winograd, "The pagerank citation ranking: Bringing order to the web," in *Proceedings of the 7th International World Wide Web Conference*, Brisbane, Australia, 1998, pp. 161–172. [Online]. Available: citeseer.nj.nec.com/page98pagerank.html

[36] M. J. Pazzani, "A framework for collaborative, content-based and demographic filtering," *Artif. Intell. Rev.*, vol. 13, no. 5-6, pp. 393–408, 1999.

[37] M. J. Pazzani and D. Billsus, "Learning and revising user profiles: The identification of interesting web sites," *Machine Learning*, vol. 27, no. 3, pp. 313–331, 1997.

[38] C. Porcel and E. Herrera-Viedma, "Dealing with incomplete information in a fuzzy linguistic recommender system to disseminate information in university digital libraries," *Knowl.-Based Syst.*, vol. 23, no. 1, pp. 32–39, 2010.

[39] L. Ren, L. He, J. Gu, W. Xia, and F. Wu, "A hybrid recommender approach based on widrow-hoff learning," in *FGCN (1)*, 2008, pp. 40–45.

[40] P. Resnick and R. Zeckhauser, "Trust among strangers in Internet transactions: Empirical analysis of eBay's reputation system," in *The Economics of the Internet and E-Commerce*, ser. Advances in Applied Microeconomics, M. R. Baye, Ed. Elsevier Science, 2002, vol. 11, pp. 127–157. [Online]. Available: http://www.si.umich.edu/ presnick/papers/ebayNBER/index.html

[41] J. B. Schafer, D. Frankowski, J. L. Herlocker, and S. Sen, "Collaborative filtering recommender systems," in *The Adaptive Web*, 2007, pp. 291–324.

[42] A. I. Schein, A. Popescul, L. H. Ungar, and D. M. Pennock, "Methods and metrics for cold-start recommendations," in *SIGIR*, 2002, pp. 253–260.

[43] S. K. Shinde and U. V. Kulkarni, "Hybrid personalized recommender system using centering-bunching based clustering algorithm," *Expert Syst. Appl.*, vol. 39, no. 1, pp. 1381–1387, 2012.

[44] L. Si and R. Jin, "Flexible mixture model for collaborative filtering," in *ICML*, 2003, pp. 704–711.

[45] R. R. Sinha and K. Swearingen, "Comparing recommendations made by online systems and friends," in *DELOS Workshop: Personalisation and Recommender Systems in Digital Libraries*, 2001.

[46] M. Tavakolifard and K. C. Almeroth, "Social computing: an intersection of recommender systems, trust/reputation systems, and social networks," *IEEE Network*, vol. 26, no. 4, pp. 53–58, 2012.

[47] P. Victor, C. Cornelis, M. D. Cock, and A. Teredesai, "Trust- and distrust-based recommendations for controversial reviews." *IEEE Intelligent Systems*, vol. 26, no. 1, pp. 48–55, 2011.

[48] F. E. Walter, S. Battiston, and F. Schweitzer, "A model of a trust-based recommendation system on a social network," *Autonomous Agents and Multi-Agent Systems*, vol. 16, no. 1, pp. 57–74, 2008.

[49] Y. Wang and J. Vassileva, "Bayesian network trust model in peer-to-peer networks," in *In Proceedings of Second International Workshop Peers and Peer-to-Peer Computing*. Springer-Verlag, 2003, pp. 23–34.

[50] A. Whitby, A. Jsang, and J. Indulska, "Filtering out unfair ratings in bayesian reputation systems," 2004.

[51] B. Yu and M. P. Singh, "An evidential model of distributed reputation management," in *AAMAS*, 2002, pp. 294–301.

[52] C.-N. Ziegler and G. Lausen, "Analyzing correlation between trust and user similarity in online communities," in *iTrust*, 2004, pp. 251–265.