Design Considerations for a Social Network-Based Recommendation System (SNRS)

Jianming He and Wesley W. Chu
Computer Science Department
University of California, Los Angeles, CA 90095
jmhek@cs.ucla.edu, wwc@cs.ucla.edu

Abstract. The effects of homophily among friends have demonstrated their importance to product marketing. However, it has rarely been considered in recommender systems. In this chapter, we propose a new paradigm of recommender systems which can significantly improve performance by utilizing information in social networks including user preference, item likability, and homophily. A probabilistic model, named SNRS, is developed to make personalized recommendations from such information. We extract data from a real online social network, and our analysis of this large dataset reveals that friends have a tendency to select the same items and give similar ratings. Experimental results from this dataset show that SNRS not only improves the prediction accuracy of recommender systems, but also remedies the data sparsity and cold-start issues inherent in collaborative filtering. Furthermore, we propose to improve the performance of SNRS by applying semantic filtering of social networks, and validate its improvement via a class project experiment. In this experiment we demonstrate how relevant friends can be selected for inference based on the semantics of friend relationships and finer-grained user ratings. Such technologies can be deployed by most content providers. Finally, we discuss two trust issues in recommender systems and show how SNRS can be extended to solve these problems.

1 Introduction

In order to overcome information overload, recommender systems have become a key tool for providing users with personalized recommendations on items such as movies, music, books and news. Intrigued by many practical applications, researchers have developed algorithms and systems over the last decade. Some of them have been commercialized by online venders such as Amazon.com, and Netflix.com. These systems predict user preferences (often represented as numeric ratings) for new items based on the user's past ratings of other items. The algorithms used in recommender systems are usually two types—*content-based filtering* and *collaborative filtering*. Let us define a *target item* as the item being considered for recommendation, and a *target user* as the user who is receiving recommendations. In content-based filtering, a target item is recommended to a target user if the item is similar to the ones that the user liked in the past in terms of explicit content attributes [18, 28], while in collaborative filtering, a target item is recommended to a target user if it is an item that has been liked in the past by people who are similar to this user. Collaborative filtering finds users who are similar to a target user based on their previous ratings of other items [4, 3, 25].

Despite all of the efforts above, recommender systems still face many challenges. First, there are continuous demands for further improvements on the prediction accuracy of recommender systems. Second, the algorithms for recommender systems suffer from many issues. For example, in order to measure item similarity, content-based methods rely on explicit item descriptions. However, such descriptions may be difficult to obtain for abstract items like ideas or opinions. On the other hand, collaborative filtering has a *data sparsity* problem [1]. In contrast to the huge number of items in recommender systems, each user normally rates only a few items. Therefore, the user/item rating matrix is typically very sparse. It is difficult for recommender systems to accurately measure user similarities from that limited number of reviews. A related problem is the *cold-start* problem [1]. Even for a system that is not particularly sparse, when a user initially joins, the system has no reviews from this user. Therefore, the system cannot accurately interpret this user's preference.

To tackle those problems, two approaches have been proposed [4, 27, 17, 18]. The first approach condenses the user/item rating matrix through dimensionality reduction techniques such as *Singular Value Decomposition (SVD)* [4, 22, 27]. By clustering users or items according to their latent structure, unrepresentative users or items can be discarded, and thus the user/item matrix becomes denser. However, these techniques do not significantly improve the performance of recommender systems, and sometimes even make the performance worse. The second approach "enriches" the user/item rating matrix by: 1) using a default rating; 2) incorporating implicit user ratings, e.g., the time spent on reading articles [19]; 3) filling in with half-baked rating predictions from content-based methods [17]; or 4) exploiting transitive associations among users through their past transactions and feedback [10]. These methods alleviate the data sparsity problem to some extent, but still cannot solve the cold-start issue. In this chapter, we plan to solve these problems from a different perspective. Specifically, we propose a *social network-based recommender system* (SNRS) [9] which predicts user interests by utilizing rich semantic information in social networks, especially *social relationships*.

In a social network, two persons connected via a social relationship tend to have similar attributes to each other. This is a fundamental property of social networks, and it is also known as the *homophily principle* [20]. In product marketing, the importance of social relationships has long been recognized [30, 32]. Intuitively, when we want to buy an unfamiliar product, we often consult with our friends who have already experienced the product, since they are those whom we can reach for immediate advice. When friends recommend a product to us, we also tend to accept the recommendation because we consider their inputs as trustworthy. Many marketing strategies, such as Hotmail, that leveraged social relationships have achieved great success [12]. Thus, social relationships play a key role when people make decisions about products, and it is the basis for constructing SNRS.

The recent emergence of online social networks (OSNs) gives us an opportunity to investigate the role of social relationships in recommender systems. With the increasing popularity of Web 2.0, many OSNs such as Myspace.com, and Facebook.com have emerged. Members in those networks have their own personalized space where they not only publish their biographies, hobbies, interests, blogs, *etc*., but also list their *friends*. Here, friends are defined in a general sense: any two users who are connected by an explicit social relationship are considered as friends. In reality, they can be family members, buddies, classmates and so on. In addition, we define *immediate friends* as friends who are just one hop away from each other in a social network graph, and *distant friends* as friends who are multiple hops away. OSNs provide platforms where people can place themselves on exhibit and maintain connections with friends. As OSNs continue to gain more popularity, the unprecedented amount of personal information and social relationships can promote social science research which was once limited by the lack of data. In this chapter, we design a new paradigm of recommender systems by utilizing such information in social networks.

While the benefits of utilizing social network information in recommender systems can be significant, how to materialize such an idea is especially challenging considering the complexity of social networks. Many challenging questions can be raised in this context. In particular, we investigate the following questions: 1) Does homophily really exist when friends rate items? 2) How to effectively use different types of social network information to make better predictions? 3) If predictions rely on the opinions of immediate friends, what if a target user has no immediate friend who has reviewed the same target item? 4) How does SNRS handle heterogeneities in social networks such as different types of friend relationships? 5) How does SNRS handle situations where the reviews from immediate friends are not trustworthy?

The remainder of the chapter is organized as follows. First, in Section 2 we give a background of collaborative filtering algorithms. Then, in Section 3 we introduce the dataset that we crawled from a real online social network, Yelp.com. We will study this dataset to determine whether homophily exists when friends rate items. In Section 4, we present our SNRS system. Following that, we evaluate the performance of SNRS on the Yelp dataset in Section 5, focusing on its prediction accuracy and coverage. In Section 6, we propose to further improve the prediction accuracy of SNRS by applying semantic filtering for social networks. We design a student experiment in a graduate class to validate its effectiveness. In Section 7, we propose extensions of SNRS to handle the trust issues caused by users with unreliable domain knowledge. Finally, we review related work in Section 8.

2 Background

After the pioneering work done in the Grouplens project in 1994 [25], collaborative filtering (CF) soon became one of the most popular algorithms in recommender systems. Many variations of this algorithm have also been proposed such as hybrid approaches of combining CF with content-based filtering [2, 17, 23, 31], or adopting different weighting schemes [1, 11]. In this chapter, we will use the traditional CF proposed in the Grouplens project as one of the comparison methods. Therefore, the remainder of this section will focus on this algorithm.

The assumption of CF is that people who agreed in the past tend to agree again in the future. Therefore, CF first finds users with tastes similar to those of the target users. CF will then make recommendations to the target user by predicting the target user's rating of the target item based on the ratings of his/her top-K similar users. User ratings are often represented by discrete values within a certain range, e.g., 1 to 5. A 1 indicates an extreme dislike of the target item, while a 5 shows high praise. Let R_{UI} be the rating of the target user U on the target item I. Thus, R_{UI} is estimated as the weighted sum of the votes of similar users as follows.

$$R_{UI} = \overline{R_U} + Z \sum_{V \in \Psi} w(U,V) \times (R_{VI} - \overline{R_V}) \,, \tag{1}$$

where $\overline{R_U}$ and $\overline{R_V}$ represent the average ratings of the target user U and every user V in U's neighborhood, Ψ, which consists of the top-K similar users of U. $w(U, V)$ is the weight between users U and V, and $Z = \dfrac{1}{\sum_V w(U,V)}$ is a normalizing constant to normalize total weight to one. Specifically, $w(U, V)$ can be defined using the *Pearson correlation coefficient* [25].

$$w(U,V) = \frac{\sum_I (R_{UI} - \overline{R_U})(R_{VI} - \overline{R_V})}{\sqrt{\sum_I (R_{UI} - \overline{R_U})^2 \sum_I (R_{VI} - \overline{R_V})^2}} \tag{2}$$

where the summations over I are over the common items for which both user U and V have voted.

As we can see, the traditional CF models user-to-user relations based purely on user rating similarities, and does not utilize at all the semantic friend relations among users. However, such semantics are essential to the buying decisions of users. In the following sections, we are going to present a new paradigm of recommender systems which improves the performance of recommender systems by using the semantic information in social networks.

3 Yelp.com

For this research, we collect a dataset from a real online social network Yelp.com. As one of the most popular Web 2.0 websites, Yelp provides users with local searches for restaurants, shopping, spas, and nightlife *etc*. Besides maintaining the traditional features of recommender systems, Yelp provides social network features so that it can attract more users. Specifically, Yelp allows users to invite their friends to join Yelp or make new friends with those who already exist at Yelp. The friendship at Yelp is a mutual relationship, which means that when a user adds another user as a friend, the first user will be automatically added as a friend of the second user. Yelp provides a homepage for each local commercial entity and each user. From the homepage of a local entity, we can find all the reviews of this entity. From the homepage of a user, we can find all the reviews written by this user as well as friends explicitly specified by this user.

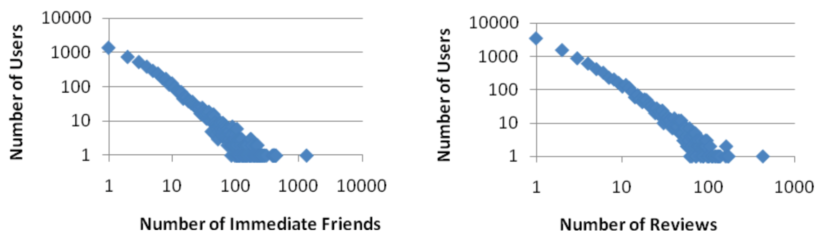
Specifically, we picked restaurants, the most popular category at Yelp, as the problem domain. We crawled the homepages of all the Yelp restaurants in the Los Angeles area that were registered before November 2007, which ended up being 4,152 restaurants. Then, by following the reviewers' links in the Yelp restaurant homepages, we further crawled the homepages of all these reviewers, which resulted in 9,414 users. Based on the friend links in users' homepages, we were able to identify friends from the crawled users, and thus reconstruct a social network of Yelp users. Note that the friends we collected for each user may only be a subset of the actual friends listed on the user's homepage. That is because we require every user in our dataset to have at least one review in the crawled restaurants. In other words, the social network that we crawled has a focus on dining.

A preliminary study of this dataset yields the following results. The dataset contains 4,152 restaurants 9,414 users, and 55,801 user reviews. Thus, each Yelp user, on average, writes 5.93 reviews and each restaurant, on average, has 13.44 reviews. If we take a closer look at the relations between the number of users and the number of their immediate friends (as shown in Figure 1(a)), we can see that it actually follows a *power-law distribution*; this means that most users have only a few immediate friends while a few users have a lot of immediate friends. A similar distribution also applies to the relations between the number of users and the number of reviews, as shown in Figure 1(b). Because most users on Yelp review only a few restaurants, it thus causes a data sparsity issue as in most recommender systems. In particular, the sparsity of this dataset, i.e., the percentage of user/item pairs whose ratings are unknown, is 99.86%.

Since homophily is the main assumption for building SNRS, we would like to see whether homophily appears in the Yelp dataset. In the following studies, we focus on two questions: 1) whether friends tend to review the same restaurant than non-friends; and 2) whether friends tend to give similar ratings to those of non-friends.

3.1 Review Correlations of Immediate Friends

In this study, we want to know if a user reviews a restaurant, what is the chance that at least one of the user's immediate friends has also reviewed the same restaurant? To answer this question, we count, for each user, the percentage of restaurants that have also been reviewed by at least one immediate friend. The average percentage over all users in the dataset is 18.6%. As a comparison, we calculate the same probability by assuming immediate friends review restaurants uniformly at random and independently. In a social network with n users, for a user with q immediate friends and a restaurant with m reviewers (including the current user), the chance that at least one of q immediate friends appears in m reviewers is $1-\binom{n-q-1}{m-1}\Big/\binom{n-1}{m-1}$. We calculate this value for every user and every restaurant the user reviewed. The average probability over all users is only 3.7%. Compared to the 18.6% observed in the dataset, it is clear that immediate friends do not review restaurants randomly.



(a) (b)

Figure 1: (a) The number of users versus the number of immediate friends in the Yelp network, and (b) the number of users versus the number of reviews both follow the power-law distribution.

We also compare the average number of co-reviewed restaurants between any two immediate friends and any two random users on Yelp. The results are 0.85 and 0.03 respectively, which again illustrates the tendency for immediate friends to co-review the same restaurants.

3.2 Rating Correlations of Immediate Friends

To validate whether immediate friends tend to give ratings that are more similar than those of non-friends, we compare the average rating differences (in absolute values) for the same restaurant between reviewers who are immediate friends, and non-friends. We find that for every restaurant in our dataset, if two reviewers are immediate friends, their ratings of this restaurant differ by 0.88 on average with a deviation of 0.89. If they are not, their rating difference is 1.05 and the standard deviation is 0.98. This result clearly demonstrates that immediate friends, on average, give more similar ratings than do non-friends.

From the studies above, we can see that immediate friends at Yelp have stronger correlations than non-friends when reviewing the same restaurants and give similar ratings. In other words, homophily indeed exists when friends rate items. This observation further leads us to the design of SNRS in Section 4.

4 A Social Network-Based Recommender System (SNRS)

Before we present SNRS, let us first use Angela's story to recall the critical factors in our buying decisions. Angela wants to watch a movie on a weekend. Her favorite movies are dramas. From the Internet, she finds two movies that are particularly interesting—"Revolutionary Road" and "The Curious Case of Benjamin Button". These two movies are all highly rated on the message board at Yahoo Movies. Because she cannot decide which movie to watch, she calls her best friend Linda with whom she often socializes. Linda has not viewed these two movies either, but she knows that one of her office mates had just watched "Revolutionary Road" and highly recommended it. So Linda suggests "Why don't we go to watch Revolutionary Road together?" Angela is certainly willing to take Linda's recommendation, and has a fun night at the movies with her friend.

If we review this scenario, we can see at least three factors that really contribute to Angela's final decision. The first factor is Angela's own preference for drama movies. If Angela did not like drama movies, she would be less likely to pick something like "Revolutionary Road" to begin with. The second factor is the global popularity of these two movies. If these movies had received unfavorable reviews, Angela would most likely lose interest and stop any further investigation. Finally, it is the recommendation from Angela's friend, Linda, which leads to Angela's finally choosing "Revolutionary Road." Interestingly, Linda's opinion is also influenced by her office mate. If we consider the decisions that we make in our daily lives, many of them are actually influenced by these three factors.

Figure 2 further illustrates how these three factors impact upon customers' final buying decisions. Intuitively, a customer's buying decision or rating is decided by both his/her own preference for similar items and his/her knowledge about the characteristics of the target item. A user's preference, such as Angela's interest in drama movies, is usually reflected in the user's past ratings of other similar items, e.g., the number of drama movies that Angela previously viewed and the average rating that Angela gave to those movies. Knowledge about the target item can be obtained from public media such as magazines, television, and the Internet. Meanwhile, the feedback from friends is another source of knowledge regarding the item, and they are often more trustworthy than advertisements. When a user starts considering the feedback from his/her friends, this user is then influenced by his/her friends. Note that such an influence is not limited to that from our immediate friends. Distant friends can also indirectly exert their influence on us; in the previous scenario, for example, Angela was influenced by Linda's office mate. Each one of these three factors has an impact on a user's final buying decision. If the impact from all of them is positive, it is very likely that the target user will select the item. On the contrary, if any has a negative influence, e.g., very low ratings in other user reviews, the chance that the target user will select the item will decrease. Bearing this in mind, we propose SNRS in the following sections.

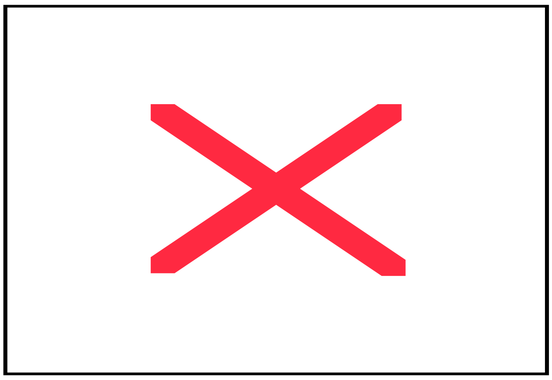


Figure 2: The three factors that influence a customer's buying decision: user preference for similar items, information regarding the target item from the public media, and feedback from friends.

4.1 SNRS Architecture

Let us now introduce the variables used in this chapter, and formalize the problem that we are dealing with. Specifically, we use capital letters to represent variables, and use capital and bold letters to represent their corresponding variable sets. The value for each variable or variable set is represented by the corresponding lower case letter.

Formally, we consider a social network as a graph $G = (\boldsymbol{U}, \boldsymbol{E})$ in which $\boldsymbol{U}$ represents nodes (users) and $\boldsymbol{E}$ represents links (social relationships). Each user U

in $\boldsymbol{U}$ has a set of attributes $\boldsymbol{A}_U$ as well as immediate neighbors (friends) $\boldsymbol{N}(U)$ such that if $V \in \boldsymbol{N}(U)$, then $(U, V) \in \boldsymbol{E}$. The values of attributes $\boldsymbol{A}_U$ are represented as $\boldsymbol{a}_U$. Moreover, a recommender system contains the records of users' previous ratings, which can be represented by a triple relation of $T = (\boldsymbol{U}, \boldsymbol{I}, \boldsymbol{R})$ in which $\boldsymbol{U}$ is the users in the social network G; $\boldsymbol{I}$ is the set of items (products or services), and each item I in $\boldsymbol{I}$ has a set of attributes $\boldsymbol{A}'_I$. $\boldsymbol{R}$ is a set of item ratings for item I; that is, R={R_{UI}.} where R_{UI} = user U's rating on item I and takes a numeric value k (e.g., k ={1, 2,..., 5}). Moreover, we define $\boldsymbol{I}(U)$ as the set of items that user U has reviewed, and refer to the set of reviewers of item I as $\boldsymbol{U}(I)$. The goal of this recommender system is to predict $Pr(R_{UI} = k \mid \boldsymbol{A}' = \boldsymbol{a}'_I, \boldsymbol{A} = \boldsymbol{a}_U, \{R_{VI} = r_{VI} : \forall V \in \boldsymbol{U}(I)\})$; i.e., the probability distribution of the target user U's rating of the target item I given the attribute values of item I, the attribute values of user U, and V's rating on item I for all reviewers V on item I. Once we obtain this distribution, R_{UI} is equal to the expected value of the distribution. Items with high estimated ratings will be recommended to the target user, and users with high estimated ratings on the target item are the potential buyers.

To achieve the goal, we propose SNRS as shown in Figure 3. SNRS consists of two major components: an *immediate friend inference engine* and a *distant friend inference engine*. As we pointed out in Angela's story, a user's buying decision is influenced not only by his/her immediate friends; his/her distant friends can also exert their influence indirectly through his/her immediate friends. Therefore, SNRS incorporates these two types of influences, but it deals with them differently. In particular, the immediate friend inference engine focuses on exploiting the homophily effects among immediate friends, and the distant friend inference engine leverages the immediate friend inference engine to bring homophily effects among distant friends into consideration.

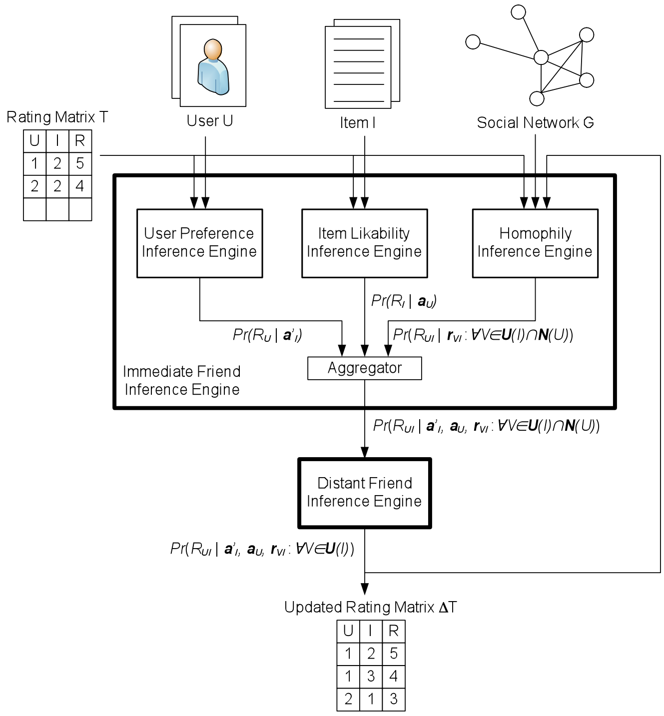


Figure 3: The architecture of a social network-based recommender system.

More specifically, the immediate friend inference engine contains four smaller components: 1) *User preference inference engine* computes the probability distribution of a target user U's rating based on U's preferences to the items similar to a target item I; 2) *Item likability inference engine* computes the probability distribution of the rating that item I receives based on the characteristics of the reviewers similar to user U; 3) *Homophily inference engine* utilizes homophily effects among immediate friends, and computes the probability distribution of user U's rating of item I based on U's immediate friends' ratings on item I; and finally, 4) an *aggregator* takes the results from the aforementioned three inference engines, combines them, and predicts user U's rating distribution for item I. We shall discuss these components of SNRS in the following sections.

4.2 Immediate Friend Inference

Since the immediate friend inference engine considers homophily from immediate friends only, the probability distribution it estimates is actually $Pr(R_{UI} = k \mid \mathbf{A}'=\mathbf{a}'_I, \mathbf{A}=\mathbf{a}_U, \{R_{VI}= r_{VI} : \forall V \in \mathbf{U}(I) \cap \mathbf{N}(U)\})$. The set of user V is limited from all reviewers of item I to U's immediate friends who also rate item I. Note that information from other reviewers of item I will be used in the distant friend inference engine. Since direct computing $Pr(R_{UI} = k \mid \boldsymbol{A}'=\boldsymbol{a}'_I, \boldsymbol{A}=\boldsymbol{a}_U, \{R_{VI}= r_{VI} : \forall V \in \boldsymbol{U(I)} \cap \boldsymbol{N(U)}\})$ is difficult, we assume that the influence of three factors, i.e., item attributes, user attributes, and ratings of immediate friends, are independent of each other. Therefore, we factorize this probability as follows.

$$\begin{aligned} &\Pr(R_{UI} = k \mid \boldsymbol{A}' = \boldsymbol{a}'_I, \boldsymbol{A} = \boldsymbol{a}_U, \{R_{VI} = r_{VI} : \forall V \in \boldsymbol{U}(I) \cap \boldsymbol{N}(U)\}) \\ &= \frac{1}{Z}\Pr(R_{UI} = k \mid \boldsymbol{A}' = \boldsymbol{a}'_I) \times \Pr(R_{UI} = k \mid \boldsymbol{A} = \boldsymbol{a}_U) \times \\ &\Pr(R_{UI} = k \mid \{R_{VI} = r_{VI} : \forall V \in \boldsymbol{U}(I) \cap \boldsymbol{N}(U)\}) \end{aligned} \tag{3}$$

First, $Pr(R_U = k \mid \boldsymbol{A}'= \boldsymbol{a}'_I,)$ is the conditional probability that the target user U will give a rating k to an item with the same attribute values as item I. This probability represents U's preference for items similar to I. Because this value depends on the attribute values of items rather than an individual item, we drop the subscript I in R_{UI} for simplification. Second, $Pr(R_I = k \mid \boldsymbol{A} = \boldsymbol{a}_u)$ is the probability that the target item I will receive a rating value k from a reviewer whose attribute values are the same as U. This probability reflects the general likability of the target item I by users like U. For the same reason, because this value depends on the attribute values of users rather than a specific user, we drop the subscript U in R_{UI}. Finally, $Pr(R_{UI} = k \mid \{R_{VI} = r_{VI} : \forall V \in \boldsymbol{U(I)} \cap \boldsymbol{N(U)}\})$ is the probability that the target user U gives a rating value k to the target item I given the ratings of U's immediate friends for item I. This is where we actually take homophily effects into consideration in SNRS. We shall present the components for estimating each of the above probabilities in the following sections.

4.2.1 User Preference

$Pr(R_U = k \mid \boldsymbol{A}'= \boldsymbol{a}'_I)$ measures the target user U's preference for the items similar to item I. For example, if we want to predict Angela's rating to "Revolutionary Road," $Pr(R_U = k \mid \boldsymbol{A}'= \boldsymbol{a}'_I)$ gives us a hint of how likely it is that Angela will give a rating k to a drama movie which also has Kate Winslet in the cast. To estimate this probability, we adopt the naïve Bayes assumption. We assume that the item attributes in $\boldsymbol{A}'$, e.g., category and cast, are independent of each other. Therefore, we adopt this approach, and have

$$\begin{aligned} \Pr(R_U = k \mid \boldsymbol{A} = \boldsymbol{a}_I) &= \frac{\Pr(R_U = k) \times \Pr(A_1, A_2, \ldots, A_n \mid R_U = k)}{\Pr(A_1, A_2, \ldots, A_n)} \\ &= \frac{\Pr(R_U = k) \times \prod_{j=1}^{j=n} \Pr(A_j \mid R_U = k)}{\Pr(A_1, A_2, \ldots, A_n)}, \boldsymbol{A} = \{A_1, A_2, \ldots, A_n\} \end{aligned} \tag{4}$$

where $Pr(A'_1, A'_2, \ldots, A'_n)$ can be treated as a normalizing constant, $Pr(R_U = k)$ is the prior probability that U gives a rating k, and $Pr(A'_j \mid R_U = k)$ is the conditional probability that each item attribute A'_j in $\boldsymbol{A}'$ has a value a'_j given U rated k; e.g., *Pr(*movie type = drama / R_U = 4*)*. The last two probabilities can be estimated from counting the review ratings of the target user U. Specifically,

$$\Pr(R_U = k) = \frac{\left|\boldsymbol{I}(R_U = k)\right| + 1}{\left|\boldsymbol{I}(U)\right| + n}, \qquad \text{and} \tag{5}$$

$$\Pr(A'_j = a'_j \mid R_U = k) = \frac{\left|\boldsymbol{I}(A'_j = a'_j, R_U = k)\right| + 1}{\left|\boldsymbol{I}(R_U = k)\right| + m}, \tag{6}$$

where $|\boldsymbol{I}(U)|$ is the number of reviews of user U in the training set, $|\boldsymbol{I}(R_U = k)|$ is the number of reviews that user U gives a rating value k, and $|\boldsymbol{I}\ (A'_j = a'_j, R_U = k)|$ is the number of reviews to which U gives a rating value k while attribute A'_j of the corresponding target item has a value a'_j. Notice that we insert an extra value 1 to the numerators in both equations, and add n, the range of review ratings to the denominator in Equation 5, and m, the range of A'_j's values, to the denominator in Equation 6. This method is also known as the Laplace estimate, a well-known technique in estimating probabilities [7], especially on a small size of training samples. Because of the Laplace estimate, "strong" probabilities, like 0 or 1, from direct probability computation can be avoided.

Moreover, in some cases when item attributes are not available, we can approximate $Pr(R_U = k \mid \boldsymbol{A}' = \boldsymbol{a}'_I)$ by the prior probability $Pr(R_U = k)$. Even though $Pr(R_U = k)$ does not contain information specific to certain item attributes, it does take into account U's general rating preference; e.g., if U is a generous person, U gives high ratings regardless of the items.

4.2.2 Item Likability Inference Engine

$Pr(R_I = k \mid \boldsymbol{A} = \boldsymbol{a}_u)$ captures the general likability of item I from users like user U. For example, from a reviewer who is similar to Angela (e.g., the same gender and age), how likely is it that "Revolutionary Road" will receive a rating of 5? Similar to the estimation in user preference, we use the naïve Bayes assumption and assume that user attributes are independent. Thus, we have

$$\begin{aligned}\Pr(R_I = k \mid \boldsymbol{A} = \boldsymbol{a}_U) &= \frac{\Pr(R_I = k) \times \Pr(A_1, A_2, \ldots, A_m \mid R_I = k)}{\Pr(A_1, A_2, \ldots, A_m)} \\ &= \frac{\Pr(R_I = k) \times \prod_{j=1}^{j=m} \Pr(A_j \mid R_I = k)}{\Pr(A_1, A_2, \ldots, A_m)}, \boldsymbol{A} = \{A_1, A_2, \ldots, A_m\}\end{aligned} \tag{7}$$

where $Pr(R_I = k)$ is the prior probability that the target item I receives a rating value k, and $Pr(A_j \mid R_I = k)$ is the conditional probability that user attribute A_j of a reviewer has a value of a_j given item I receives a rating k from this reviewer. These two probabilities can be learned by counting the review ratings on the target item I in a manner similar to what we did in learning user preferences. When user attributes are not available, we use $Pr(R_I = k)$, i.e., item I's general likability, regardless of users, to approximate $Pr(R_I = k \mid A = a_u)$. In addition, $Pr(A_1, A_2, ..., A_m)$ is a normalizing constant.

4.2.3 Homophily Inference Engine

Finally, $Pr(R_{UI} = k \mid \{R_{VI} = r_{VI} : \forall V \in \boldsymbol{U}(I) \cap \boldsymbol{N}(U)\})$ is where SNRS utilizes homophily effects from immediate friends. To estimate this probability, SNRS needs to learn the correlations between the target user U and each of U's immediate friends V from the items that they both have rated previously, and then assume each pair of friends will behave consistently on reviewing the target item I also. Thus, U's rating can be predicted from r_{VI} according to the correlations. A common practice for learning such correlations is to estimate user similarities or coefficients, based on either user profiles or user ratings. However, user correlations are often so sensitive that they cannot be fully captured by a single similarity or coefficient value. Different measures return different results, and have different conclusions on whether or not a pair of users is really correlated [12]. At another extreme, user correlations can be also represented in a joint distribution table of U's and V's ratings on the same items that they have rated; i.e., $Pr(R_{UJ}, R_{VJ})$ $\forall J \in \boldsymbol{I}(U) \cap \boldsymbol{I}(V)$. This table fully preserves the correlations between U's and V's ratings. However, in order to build such a distribution with accurate statistics, it requires a large number of training samples. This is especially a problem for recommender systems, because in most of these systems, users review only a few items compared to the large amount of items available in the system, and the co-rated items between users are even fewer. Therefore, in this study, we use another approach to remedy the problems in both cases.

In Section 3, we showed that it is true that immediate friends tend to give similar ratings more than do non-friends. Therefore, for each pair of immediate friends U and V, we consider their ratings on the same item to be close with some error ε. That is,

$$R_{UI} = R_{VI} + \varepsilon, \quad I \in \boldsymbol{I}(U) \cap \boldsymbol{I}(V), V \in \boldsymbol{N}(U) \cap \boldsymbol{U}(I) \tag{8}$$

From this equation, we can see that error ε can be simulated from the histogram of U's and V's rating differences $Hist(R_{UJ} - R_{VJ})$ for $\forall J \in \boldsymbol{I}(U) \cap \boldsymbol{I}(V)$. Thus, $Hist(R_{UJ} - R_{VJ})$ serves as the correlation measure between U and V. For rating ranges from one to five, $Hist(R_{UJ} - R_{VJ})$ is a distribution of nine values, i.e. from -4 to 4. Compared to similarity measures, it preserves more details in friends' review ratings. Compared to a joint distribution approach, it has fewer degrees of freedom.

Assuming *U's* and *V's* rating difference on the target item I is consistent with $Hist(R_{UJ} - R_{VJ})$. Therefore, when R_{VI} has a rating r_{VI} on the target item, the probability that R_{UI} has a value k is proportional to $Hist(k - r_{VI})$.

$$\Pr(R_{UI} = k \mid R_{VI} = r_{VI}) \propto Hist(k - r_{VI}). \tag{9}$$

When the target user U has more than one immediate friend who co-rates the target item, the influences from all of those friends can be incorporated in a product of normalized histograms of individual friend pairs.

$$\Pr(R_{UI} = k \mid \{R_{VI} = r_{VI} : \forall V \in \boldsymbol{U}(I) \cap \boldsymbol{N}(U)\} = \frac{1}{Z}\prod_{V} \frac{1}{Z_V} Hist(k - r_{VI}) \tag{10}$$

where Z_V is the normalizing constant for the histogram of each immediate friend pair, and Z is the normalizing constant for the overall product.

Once we obtain $Pr(R_U = k \mid \boldsymbol{A}' = \boldsymbol{a}'_I)$, $Pr(R_I = k \mid \boldsymbol{A} = \boldsymbol{a}_u)$, and $Pr(R_{UI} = k \mid \{R_{VI} = r_{VI} : \forall V \in \boldsymbol{U}(I) \cap \boldsymbol{N}(U))\})$, these probabilities are fed into an aggregator where the ultimate rating distribution of R_{UI} is shown in Equation 3. R'_{UI}, the predicted value of R_{UI}, is the expected value of the distribution.

$$R'_{UI} = \sum_k k \times \Pr(R_{UI} = k \mid \boldsymbol{A}' = \boldsymbol{a}'_I, \boldsymbol{A} = \boldsymbol{a}_U, \{R_{VI} = r_{VI} : \forall V \in \boldsymbol{U}(I) \cap \boldsymbol{N}(U)\} \tag{11}$$

4.3 Distant Friend Inference

We have just introduced the approach for predicting a target user's rating of a target item from those of the user's immediate friends for the same item. However, in reality, there are many cases where no immediate friends of a target user have reviewed the same target item; thus, the rating of the target user cannot be predicted from immediate friend inference. To solve this problem, we propose distant friend inference.

The idea of distant friend inference is intuitive. Even though V, an immediate friend of a target user U, has no rating for the target item, if V has his/her own immediate friends who rated the target item, we should be able to predict V's rating of the target item via the immediate friend inference, and then to predict U's rating based on the predicted rating from V. This process conforms to real scenarios as in our previous example, where Linda's office mate influences Linda who further influences Angela. Followed by this intuition, we decide to apply an *iterative classification* method [13, 21, 26] for distant friend inference.

Iterative classification is an approximation technique for classifying relational entities. This method is based on the fact that relational entities are correlated with each other. Estimating the classification of an entity often depends on the classification estimations of its neighbors. The improved classification of one entity will help to infer the related neighbors and vice versa. Unlike traditional data mining which assumes that data instances are independent and identically distributed (i.i.d.) samples, and classifies them one by one, iterative classification iteratively classifies all the entities in the testing set simultaneously because the classifications of those entities are correlated. Note that iterative classification is an approximation technique, because exact inference is computationally intractable unless the network structures have certain graph topologies such as sequences, trees or networks with low tree width. In previous research, iterative classification has been used successfully to classify company profiles [21], hypertext documents [13], and emails [5].

The pseudo-code for distant friend inference is shown in Table 1. This pseudo-code predicts the users' ratings for each target item at a time. The original iterative classification method classifies the whole network of users. However, since the number of users in social networks is usually large, we reduce the computation cost by limiting the inference to a user set *N* which includes the target users of the target item *I*, and their corresponding immediate friends. In each iteration, we generate a random ordering *O* of the users in *N*. For each user *U* in *O*, if *U* has no immediate friend who belongs to *U(I)*, which is the set of users whose rating (either ground truth or predicted value) is observable, the estimation of R_{UI} will be skipped in this iteration. Otherwise, $\Pr(R_{UI} = k \mid A'=a'_I, A=a_U, \{R_{VI} = r_{VI} : \forall V \in U(I) \cap N(U)\})$ will be estimated by immediate friend inference, and R_{UI} is then obtained from Equation 11. Because user rating is an integer value, in order to continue the iterative process, we round R_{UI} to a close integer value, and insert into or update *U(I)* with R_{UI} if different. This entire process iterates *M* times or until no update occurs in the current iteration. In our experiment, the process usually converges within ten iterations.

```
For each item I in the testing set do
    Select a set of users N for inference. N includes the target users of item I and their corresponding immediate friends.
    For iteration from 1 to M do
        Generate a random ordering, O, of users in N
        For each user U in O do
            If U has no immediate friend who exists in U(I)
                Continue
            Else
                Apply immediate friend inference
                R_UI = Σ_k k *Pr(R_UI = k | A=a_U, A'=a'_I, {R_VI =r_VI : ∀V∈U(I)∩N(U)})
                Insert into or Update U(I) with R_UI if different
            End If
        End For
        If no updates in the current iteration
            Break
        End If
    End For
    Output the final predictions for the target users
End For
```

Table 1: Pseudo-code for distant friend inference

It is worth pointing out that after we compute $Pr(R_{UI} = k \mid \boldsymbol{A'}=\boldsymbol{a'}_I, \boldsymbol{A}=\boldsymbol{a}_U, \{\{R_{VI} = r_{VI} : \forall V \in \boldsymbol{U(I)} \cap \boldsymbol{N}(U))\})$, there are two other options for updating R_{UI} besides rounding the expectation in distant friend inference. The first option is to select R_{UI} with the value k such that it maximizes $Pr(R_{UI} = k \mid \boldsymbol{A'}=\boldsymbol{a'}_I, \boldsymbol{A}=\boldsymbol{a}_U, \{R_{VI} = r_{VI} : \forall V \in \boldsymbol{U(I)} \cap \boldsymbol{N}(U))\}\}$. However, by doing so, we are actually discarding clues of small probabilities at the same time. After several iterations, the errors caused by the greedy selection will be exacerbated. The target users are likely to be classified with the majority class. The other option is to directly use $Pr(R_{UI} = k \mid \boldsymbol{A'}=\boldsymbol{a'}_I, \boldsymbol{A}=\boldsymbol{a}_U, \{R_{VI} = r_{VI} : \forall V \in \boldsymbol{U(I)} \cap \boldsymbol{N}(U))\})$ as soft evidence to classify other users. However, in our experiments, this approach does not return results as good as those obtained by rounding the expectation.

5 Experiments

We evaluate the performance of SNRS on the Yelp dataset, mainly focusing on the issues of the prediction accuracy, data sparsity, and cold-start. We used a restaurant's price range as the item attribute. Since there is no useful user attribute, we substituted $Pr(R_I = k \mid \boldsymbol{A} = \boldsymbol{a}_u)$ with $Pr(R_I = k)$ when estimating item likability. As a comparison, we implemented CF and t*rust-based collaborative filtering (TCF)* [8]. The basic idea of TCF is to combine trust-based weighting with filtering. It first estimates two types of implicit trust: profile-level and item-level trust among users based on their ratings. Then it filters out users with low trust values. To make predictions, it uses the CF. Instead of using user similarity as in Equation 1, TCF uses a harmonic mean of user trust and user similarity. Compared with their use of implicit user trust, SNRS in fact utilizes interpersonal trust underlying friend relationships. For this reason, we are interested in comparing the performance of SNRS with that of TCF.

5.1 Cross-Validation

We carried out this experiment in a 10-fold cross-validation. The prediction accuracy was measured by the mean absolute error (MAE), which is defined as the average absolute deviation of predictions about the ground truth data over all the instances, i.e., target user/item pairs, in the testing set. The smaller the MAE, the better the inference. The second metric is the coverage, which is defined as the percentage of the testing instances for which the method can make predictions.

The experimental results are listed in Table 2. From this table, we note that SNRS achieves the best performance in terms of MAE (0.727). For example, it is lower than that of CF by 14.3% and that of TCF by 6.2%. Thus, the use of social network information in SNRS improves the prediction accuracy. In terms of the coverage, SNRS reaches the highest coverage (0.807). The reason for this high coverage is because SNRS is able to make use of estimated user ratings for predictions. Considering the low MAE and high coverage of SNRS, it demonstrates that SNRS is promising. In addition, TCF improves the MAE of CF at a cost of reduction in the coverage. This is because, in some cases, even though the similarity for a pair of users can be estimated, if the trust between them cannot be obtained, TCF still cannot make predictions.

	MAE	**COVERAGE**
SNRS	***0.727***	*0.807*
TCF	*0775*	*0.454*
CF	*0.848*	***0.616***

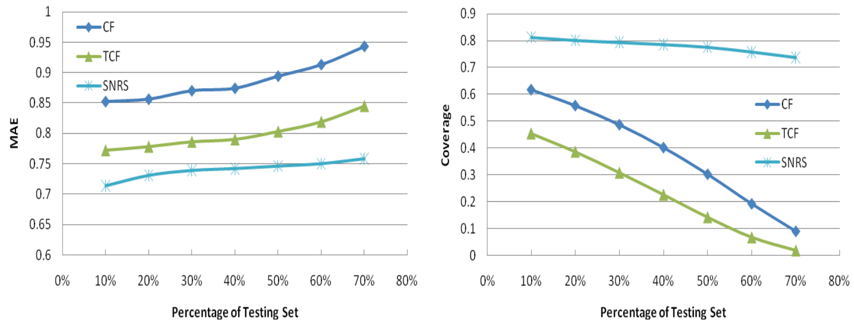
Table 2: Comparison of the MAEs of selected methods in a 10-fold cross validation on the Yelp dataset. The methods used are: collaborative filtering (CF), trust-based collaborative filtering (TCF), and social network-based recommender system (SNRS).

5.2 Data Sparsity

CF suffers from problems with sparse data. In this study, we want to evaluate the performance of SNRS at various levels of data sparsity. To do so, we randomly divide the entire user/item pairs of our dataset into ten groups, and then randomly select n groups as the testing set, and the rest as the training set. The value of n controls the sparsity of the dataset. At each value of n, we repeat the experiment 100 times. The performance is measured by the average MAEs and the coverage.

Figure 4(a) compares the MAEs of the above methods when the percentage of testing set varies from 10% to 70%. There are two observations. First, the MAEs of SNRS are consistently lower than those of CF and TCF. Second, although the MAEs of all the methods increase as the training set becomes sparser, the MAEs of SNRS grow at a much slower pace. For example, the MAEs of SNRS increase 6.2% from 0.714 to 0.758 when the testing set is increased from 10% to 70% of the entire dataset, while the MAEs of CF and TCF grow to 10.7% and 9.5% respectively under the same conditions.

Figure 4(b) compares the coverage of these methods. We noted the coverage of SNRS is the highest for all test conditions. For example, when the size of testing set is 40% of the whole dataset, the coverage of SNRS is 0.786; while that of CF and TCF is 0.713 and 0.401 respectively. The decrease in the coverage of SNRS is also the slowest as the training set becomes sparser. In particular, the ratio of the decrease in the coverage of SNRS is 9% when the size of the testing set changes from 10% to 70% of the entire dataset, while the same ratio of CF is 85.4%.



(a) (b)

Figure 4: Comparison of the (a) MAEs and the (b) coverage of CF, TCF, and SNRS for different testing set sizes.

5.3 Cold-Start

Cold-start is an extreme case of data sparsity where a new user has no reviews, in which CF cannot make recommendations to the new user. Neither can SNRS do so if this new user has no friends. However, in some cases of cold-start, when a new user is invited by some existing users in the system, the preference of this new user can be estimated by those of the user's friends. In this study, we simulated the latter case of cold-start by creating the following experimental settings: 1) Since there is no prior ratings of the target user, we simply set the output from $Pr(R_U = k \mid \boldsymbol{A}' = \boldsymbol{a}'_I,)$ as a uniform distribution. 2) Because we cannot learn the rating correlation between this new user and the user's friends, we directly used the friends' rating distributions on the target item, $Pr(R_{UI} \mid \{R_{VI} :$

$\forall V \in \boldsymbol{N}(U) \cap \boldsymbol{U}(I)\}$), as the result from friend inference. 3) Except for the target user, the ratings of all other users were known.

We simulated cold-start for every user in the dataset. The resulting MAE is 0.753 and the coverage is 1. This result demonstrates that even in cold-start, SNRS can still perform decently. The coverage of SNRS is high compared to that in the 10-fold cross-validation (0.807) because the ratings of every target user's friends are all observable in the setting of this experiment.

5.4 Role of Distant Friends

In Section 5.1, we noticed SNRS achieved the highest coverage because it is able to make use of estimated ratings of immediate friends which are inferred from distant friends. This observation leads us to further study the role of distant friends in SNRS. Specifically, we compared the performance of SNRS with and without distant friend inference in a 10-fold cross-validation. The experimental results are shown in Table 3. From these results, we can see that by considering the influences from distant friends, the coverage of SNRS is increased from 0.364 to 0.807, which is equivalent to a 122% improvement. However, the improvement is achieved at the cost of a slight reduction in the prediction accuracy. In our experiments, the MAE increases from 0.683 to 0.727, which is only a 6.4% difference. This is consistent with our intuition that the impact from distant friends is not as direct as that from immediate friends, and certain errors will be inevitably introduced when considering distant friends, but compensated for by the enormous gain in the coverage.

	MAE	COVERAGE
With Distant Friend Inference	0.727	**0.807**
Without Distant Friend Inference	**0.683**	0.364

Table 3: Comparison of the performance of SNRS with and without distant friend inference.

Our experimental results revealed that social network information can be used to improve the performance of recommender systems. In the next section, we shall discuss how to remedy some issues in SNRS that are caused by heterogeneities in social network information.

6 Semantic Filtering of Social Networks

Social networks contain rich semantics that are valuable to SNRS. However, this information can also interfere with the predictions of SNRS if not carefully applied. In this section, we discuss the issues of SNRS caused by the heterogeneities in social relationships and items.

Friends exhibit similar behaviors when selecting items; however, the favorite items that friends have in common depend on their social relationships. For example, two friends who have common interests in music CDs may not necessarily agree about their favorite restaurants. Therefore, to find the favorite restaurants, we should not consider friends that share only a common preference in music. Instead, an appropriate set of friends needs to be selected according to the target items. In fact, we considered this issue when performing experiments on the Yelp dataset. Rather than considering all friends listed in users' profiles, we keep only those friends who also have an interest in food. For example, even though two real friends may have reviewed many common hotels on Yelp, they are not necessarily considered as friends in SNRS unless they both have reviewed restaurants. However, this solution is still a gross approximation, because even within the domain of restaurants, friends can be further grouped based on their opinions on different food categories, price range, restaurant environment, *etc*.

Item clustering can theoretically be applied to SNRS to select relevant friends for inference. That is, by clustering similar items into different groups, homophily effects among friends can be estimated based on the ratings of items within the same group. Thus, it is possible for SNRS to identify friends who have a high correlation in music CDs but a low correlation in restaurants. However, because the number of items used to measure user similarity becomes less due to item clustering, the estimated similarity values may not be as accurate as those without clustering.

A better way to select relevant friends is to utilize the semantics in social relationships. Unfortunately, such semantics are not readily available in most current OSNs. When a user indicates someone as a friend, it is not clear how and why they became friends, and more importantly, we do not know in which aspects they have homophily effects. Some OSNs ask how friends know each other, e.g., whether they were/are classmates or colleagues. Information like this definitely helps us understand friend relationships. However, it is still too general to have practical application in recommender systems. Instead, the semantics that we really want to know from friend relationships should be more specific to the domain of interest, in particular, the factors that influence users' buying decisions. For example, in terms of dining, it would be ideal for SNRS to know whether two individuals are friends because they have similar taste in food and/or similar preference regarding the price of the meals. Although items have many characteristics, the factors that matter in most users' buying decisions in choosing restaurants may be limited to only a few common ones, such as food taste, nutrition value, price, service and environment. By carefully designed questionnaires or other means of marketing analysis, such factors can be obtained.

Thus, more semantics regarding users' rating intentions and social relationships can be collected.

By providing users with the mechanism to rate items for each factor in their buying decisions, e.g., asking them to rate a restaurant based on food taste and price, *etc*, recommender systems can improve the understanding of users' rating intentions. Currently, most recommender systems ask users to input only overall ratings which, however, consist of too many factors and are difficult to understand. For example, when a user gives an overall rating of 4 to a restaurant, it is not clear whether it is because of the food taste or the price of the meal. On the other hand, if a user can provide ratings for those factors, the rationale behind the overall rating can be well explained. Besides understanding users' rating intentions, SNRS can also obtain the semantics in social relationships by asking users to rate their friends on those factors. A user's high rating of a friend on a specific factor means this user tends to agree with the friend's opinion, and together they have a stronger homophily effect.

To predict a user's rating of a factor, SNRS needs to select those friends on whom this user has a strong homophily effect regarding the same factor. The selection of friends is thus dynamic according to the semantics in the factors of user ratings. We call this process semantic filtering, and denote SNRS with semantic filtering as SNRS-SF. The framework of SNRS-SF is almost the same as that of SNRS, except that immediate friend inference and distant friend inference are now based on semantically filtered social networks.

Since overall rating is not determined by a single factor, relevant friends for predicting a user's overall rating cannot be selected in the same way as we did for predicting fine-grained rating. To do so, we consider friends as those who are selected as relevant friends for two or more of the most important factors. For example, if a system considers that price and taste are the most important decision factors in terms of dining, then a user's relevant friends for predicting overall ratings are those users who are considered relevant friends for predicting the user's ratings on price and taste. In the following section, we shall use this approach.

6.1 Semantic Filtering Experiments

Since the Yelp dataset does not have fine-grained user ratings, we cannot use the Yelp dataset for semantic filtering experiments. Therefore, we designed an experiment for a graduate student class and collected a social network and fine-grained user ratings from students.

The goal of this experiment is to predict students' ratings for reading online articles. It was conducted in a graduate student class, "Intelligent Information Systems," with 22 students. We first selected 21 articles which focus mainly on four topics: local news, U.S. news, technologies, and culture. These articles all contain strong opinions expressed by the authors. The article information and the corresponding categories of these 21 articles are listed in Table 4.

Article ID	Article Information	Category
1	Adenhart's death is a tragic loss for baseball By Kendall Salter, Daily Bruin, April 10, 2009.	Local
2	Aggressive biking, skateboarding poorly fit our walking campus By Karen Louth, Daily Bruin, March 6, 2009.	Local
3	Backers of stem cell research are on guard By Robert T. Garrett, The Dallas Morning News, April 10, 2009.	Technology
4	Budget cuts should not degrade education By Daily Bruin, March 12, 2009.	Local
5	File-Sharing Site Admin Sentenced to 6 Months Jail By Enigmax, TorrentFreak, April 11, 2009.	Technology
6	Google Earth accused of aiding terrorists By Rhys Blakely, Times Online, December 9, 2008.	Technology
7	Hot Topic: A Gay Marriage Tipping Point? By Brian Montopoli, CBS News, April 6, 2009.	Culture
8	How Environmentalists Plan to Control Your Life By Fox News, April 6, 2009.	Culture
9	Identity theft hits close to home By Patt Morrison, Los Angeles Times, March 12, 2009.	Culture
10	Is an Italian rail company taking L.A. for a ride? By Tim Rutten, Los Angeles Times, March 25, 2009.	Local
11	Israel boycott shows ignorance and limits ideas By DailyBruin, March 5, 2009.	Local
12	L.A.'s animal terrorists By Tim Rutten, Los Angeles Times, March 11, 2009.	Local
13	Learning to Love the Bailout By The New York Times, April 11, 2009.	U.S.
14	Obama Flinches on Immigration By Editorials, The New York Times, March 23, 2009.	U.S.
15	The age of Friendaholism By Meghan Daum, Los Angeles Times, March 7, 2009.	Technology
16	The First Showdown on Health Care By Editorial, The New York Times, April 11, 2009.	U.S.
17	The recession heats up romance novels By Meghan Daum, Los Angeles Times, April 4, 2009.	Culture
18	Unemployment, and CEO pay, on the rise By Tim Rutten, Los Angeles Times, April 4, 2009.	U.S.
19	We need a bailout too By Rosa Brooks, Los Angeles Times, February 19, 2009.	U.S.
20	Why not gay marriage? By Raymond Lesniak, NJ.com, August 16, 2007.	Culture
21	Wild wild Web By Patt Morrison, Los Angeles Times, February 26, 2009.	Technology

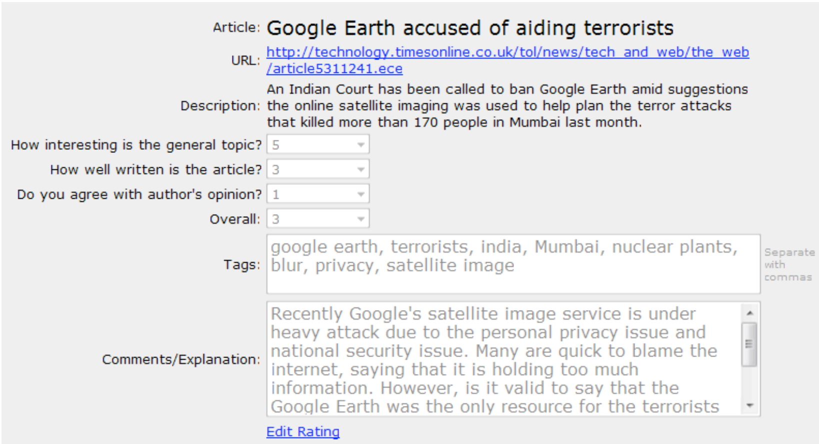
Table 4: The article information

Before asking these students to review the online articles, we first collected their demographic information, including gender, age, student type, employment, and religion. We then asked the students to answer a set of survey questions related to the articles as shown in Table 5. These survey responses will provide prior information about the students.

Q1	Has the rise in unemployment affected you or someone in your family?
Q2	Given the current state of the economy, are you concerned about getting a job after you graduate?
Q3	Are you concerned about increased government spending? What if increased government spending leads to higher tuition cost?
Q4	Are you affiliated with a political party?
Q5	Do you consider yourself conservative, liberal or moderate?
Q6	Does the government do enough to regulate immigration?
Q7	Do you support gay marriage?
Q8	Do you think there is a need for health care reform?
Q9	Should every American have health insurance?
Q10	Do you agree with the use of stem cells for medical research?
Q11	Do you know anyone with an incurable illness who may benefit from stem cell research?
Q12	Should websites and tools that could be used improperly be outlawed? (Google Earth, Bit Torrent, P2P, *etc.*)
Q13	South Korea has a three-strikes law where repeated copyright offenders can be banned from the Internet? Do you think this is fair?

Table 5: Survey questions.

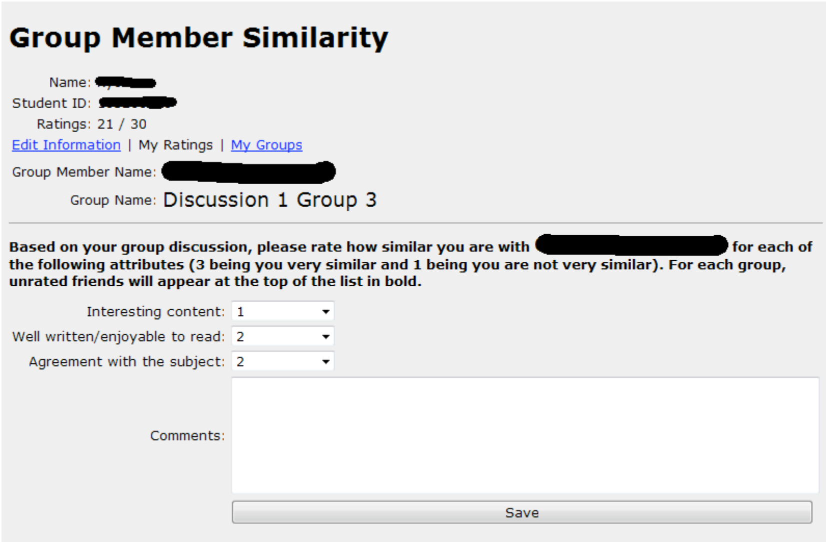
We then asked the students to review, as shown in Figure 5, every article and give ratings (from 1 to 5, with 5 being the best) on the following four factors: 1) Interestingness: Is the article interesting? 2) Agreement: How much do you agree with the author? 3) Writing: Is the article well written? and 4) Overall: Overall evaluation. The reason we include the first three ratings is because they usually play the most important roles when we give an overall score to an article. Since most students did not know each other before the experiment, it would be difficult to form a social network from their original relationships. We therefore divided the students into groups and let them get to know each other through discussions of the articles. Specifically, we divided the students into three groups twice. The first grouping was based on students' ethnicities, and the second grouping was based on students' responses to the survey questions. The goal of these groupings was to organize the students in such a way that the students in a group were more likely to be friends after the group discussions. Each group then had a meeting to discuss the articles. During the discussions, every student needed to explain the reasons why s/he liked or disliked each article. Thus, the other members in the group were able to know more about the student. After the discussions, the students evaluated other group members, as shown in Figure 6, (using ratings from 1 to 3) according to the following three aspects: 1) Do you have common interests on the articles? 2) Do you agree with his/her opinions on the articles? 3) Do you have common judgments about the author's writing skill? Since the students may rate each other differently, i.e., one considers the other as a friend, but not vice versa, the social relationship in this dataset is directional. The students were allowed to revise their previous ratings of the articles if they had a new understanding of the articles after the discussion.



Article: Google Earth accused of aiding terrorists
URL: http://technology.timesonline.co.uk/tol/news/tech_and_web/the_web/article5311241.ece
Description: An Indian Court has been called to ban Google Earth amid suggestions the online satellite imaging was used to help plan the terror attacks that killed more than 170 people in Mumbai last month.
How interesting is the general topic? 5
How well written is the article? 3
Do you agree with author's opinion? 1
Overall: 3
Tags: google earth, terrorists, india, Mumbai, nuclear plants, blur, privacy, satellite image
Separate with commas
Comments/Explanation: Recently Google's satellite image service is under heavy attack due to the personal privacy issue and national security issue. Many are quick to blame the internet, saying that it is holding too much information. However, is it valid to say that the Google Earth was the only resource for the terrorists
Edit Rating

Figure 5: Form for reviewing an article.

Compared to the Yelp dataset, there are three differences in this dataset. First, instead of having an overall rating, each article now has three fine-grained ratings (interestingness, agreement, and writing) which more clearly reflect students' opinions on the articles. Second, friend relationships are based on buying decision factors rather just friendship. We are now able to know whether the friendship is based on their similar interests or similar opinions, *etc*. Third, since every student reviewed every article in this experiment, the student/article rating matrix is completely filled in. Thus, the data sparsity of the dataset is 0. Compared to the extremely sparse data in the Yelp dataset, the fully observed students' ratings in this experiment allow us to measure the performance of SNRS-SF under the sparseness test in a full range.



Group Member Similarity

Name: [redacted]
Student ID: [redacted]
Ratings: 21 / 30
Edit Information | My Ratings | My Groups
Group Member Name: [redacted]
Group Name: Discussion 1 Group 3

Based on your group discussion, please rate how similar you are with [redacted] for each of the following attributes (3 being you very similar and 1 being you are not very similar). For each group, unrated friends will appear at the top of the list in bold.

Interesting content: 1
Well written/enjoyable to read: 2
Agreement with the subject: 2
Comments:
Save

Figure 6: Form for reviewing a group member.

6.2 Experiment Setup

We implement the following methods for performance comparison.

Collaborative filtering (CF). When predicting fine-grained student ratings, we select similar users based on their fine-grained ratings on all articles.

Collaborative filtering with item clustering based on item category (CF-C). In this method, 21 articles are clustered into four groups according to their categories. To predict a student's rating of an article, we measure his/her Pearson coefficient with other students based on their ratings of the articles in the corresponding group.

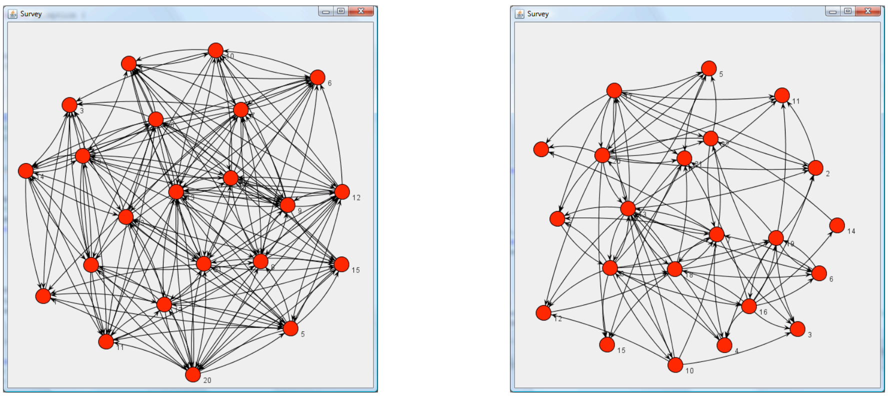
Collaborative filtering with item clustering by running K-means on students' ratings (CF-K). In this method we use *K*-means to cluster 21 articles based on their rating similarities. Since there are four types of ratings (three fine-grained ratings and an overall rating), we have four sets of clusters. In each set, the articles are clustered into three groups. Similar to CF-C, to predict each student's rating of an article, we measure Pearson coefficient of student pairs based on their ratings of the articles in the same cluster.

SNRS. In this method, we consider student V as student U's friend if U rates V with a value 3 on at least one of the three factors. The social network of these students is shown in Figure 7(a). Each node in the figure represents a student. If student U considers student V as a friend, then there is a corresponding directed edge from U to V.

SNRS with semantic filtering (SNRS-SF). In this method, when predicting fine-grained user ratings, we consider student V as student U's friend if U rates V

with a value of 3 on the given factor. When predicting overall ratings, we select *V* as *U*'s friend if *U* rates *V* with a value of 3 on at least two of the three factors. Figure 7(b) shows the social network of the students after we apply semantic filtering to overall ratings. When compared to Figure 7(a), we can see that many social relationships have been pruned. For example, before we apply semantic filtering, there are 179 friend links in Figure 7(a), and on average each student has 8.14 friends. In Figure 7(b), there are 94 friend links, and each student on average has 4.27 friends after semantic filtering.

Similar to the previous section, we control the sparseness of the dataset by randomly selecting a different percentage of the dataset as the testing set. For each size of the testing set, we repeated the experiment 100 times. For each pair of student/article in the testing set, we predicted the target student's fine-grained ratings and overall ratings of the target article by applying the above methods. MAEs and coverage are used as the performance metrics.



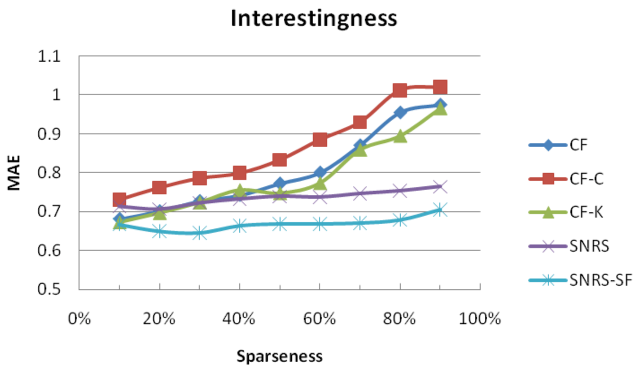
(a) (b)

Figure 7: The student social network (a) before and (b) after semantic filtering. Each node represents each student. For a pair of students *U* and *V*, node *U* has a directed edge to node *V* if *U* rates *V* with a value of 3 on at least one of the three factors and *U* rates *V* with a value of 3 for at least two of the three factors.

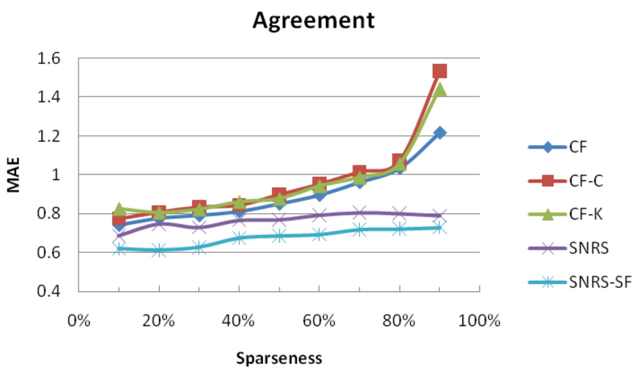
6.3 Experimental Results

Figures 8(a) through (d) show the MAEs for predicting student ratings of the interestingness, agreement, writing, and overall aspects of the articles. We notice that the trends of SNRS and SNRS-SF are very different from those of CF, CF-C, and CF-K. The MAEs of SNRS and SNRS-SF remain almost constant at all levels of data sparseness, while CF, CF-C, and CF-K all significantly increase as the sparseness increases. The improvements of MAE for SNRS and SNRS-SF over the CF group increase as data sparseness increases. For example, in Figure 8(a), when the sparseness is 90%, the MAEs of CF, SNRS, and SNRS-SF are 0.974, 0.764, and 0.705 respectively. This implies 21.6% and 27.6% prediction accuracy improvements over CF. These results reveal that the performance of recommender systems can be significantly improved by effectively using the semantic information in social networks, as consistent with our findings on the Yelp dataset. Further, the MAEs of SNRS-SF are lower than those of SNRS. Specifically, SNRS-SF yields average MAE reduction over SNRS of 9.8%, 11.6%, 7.4%, and 6.2% for predicting student ratings on interestingness, agreement, writing and overall aspects respectively. These results illustrate that applying semantic filtering can further improve the SNRS prediction accuracy. We note that the MAEs of CF, CF-C, and CF-K have similar results. CF-C perform worst among these three methods which implies that item clustering does not improve the prediction accuracy of CF as also represented in [19].

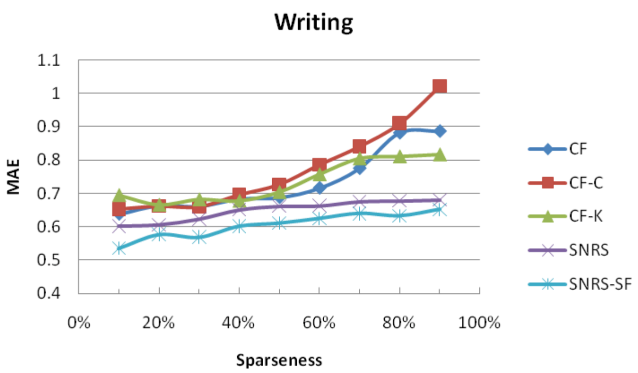
The trends of the coverage of all the above methods at different sparseness are shown in Figure 9. We notice that initially the coverage of CF, CF-C and CF-K is higher than that of SNRS and SNRS-SF. For example, in Figure 9(a), when the sparseness is 10%, the coverage of CF, CF-C and CF-K is 1, 1, and 0.897 respectively, while that of SNRS and SNRS-SF is 0.865 and 0.744. However, as the sparseness increases, the coverage of all the CF group decreases drastically. In particular, the coverage of CF starts to decrease significantly after the sparseness exceeds 0.7. It reaches almost 0 at the sparseness 0.9. The coverage of CF-C and CF-K start to decrease significantly even earlier when the sparseness exceeds 0.4. This is because item clustering makes the number of items in each cluster smaller; thus, the coverage has a greater impact due to the sparseness. On the other hand, we notice that the coverage of SNRS and SNRS-SF are rather insensitive to the sparseness. It starts to drop after the sparseness exceeds 0.8 and with a slower rate than CF, CF-C and CF-K. For example, even when the sparseness is 0.9, the coverage of SNRS and SNRS-SF is still 0.789 and 0.645, as shown in Figure 9(a). The coverage of SNRS-SF is slightly lower than that of SNRS because of fewer friends for each student.



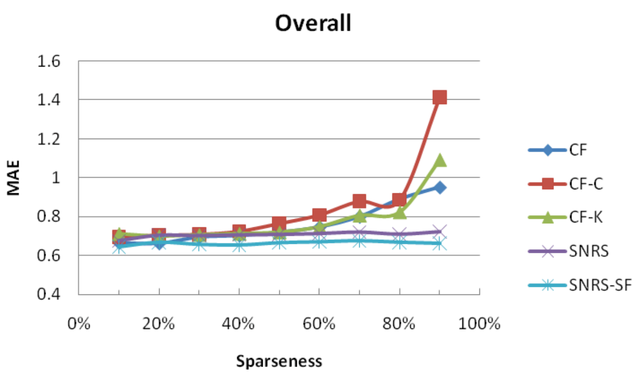
(a)



(b)

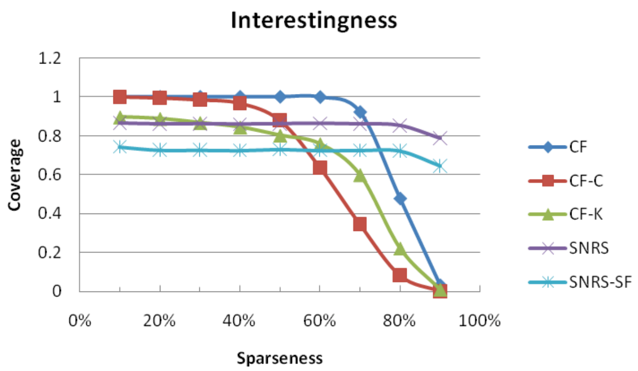


(c)

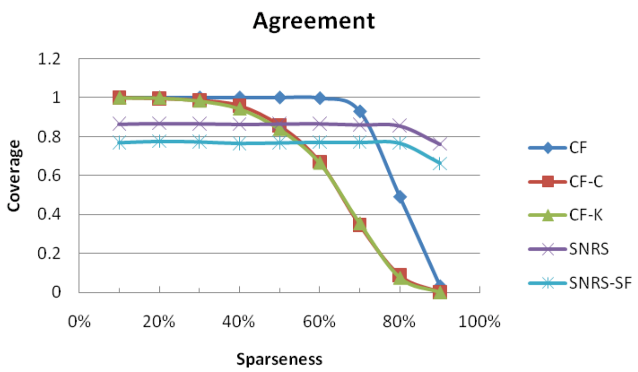


(d)

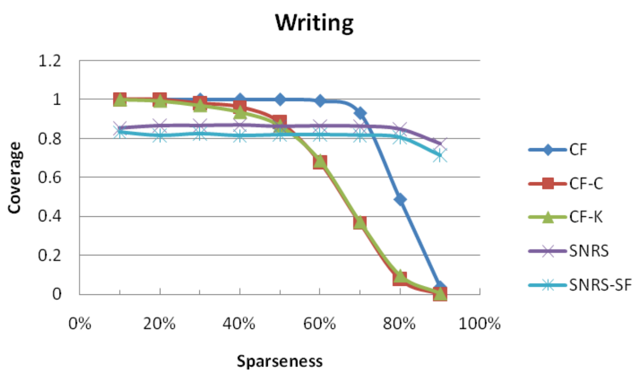
Figure 8: The comparisons of the MAEs of CF, CF-C, CF-K, SNRS, and SNRS-SF for predicting fine-grained ratings on (a) interestingness (b) agreement, (c) writing, and (d) overall aspects of the articles.


Interestingness
Coverage
1.2
1
0.8
0.6
0.4
0.2
0
0%
20%
40%
60%
80%
100%
Sparseness
CF
CF-C
CF-K
SNRS
SNRS-SF

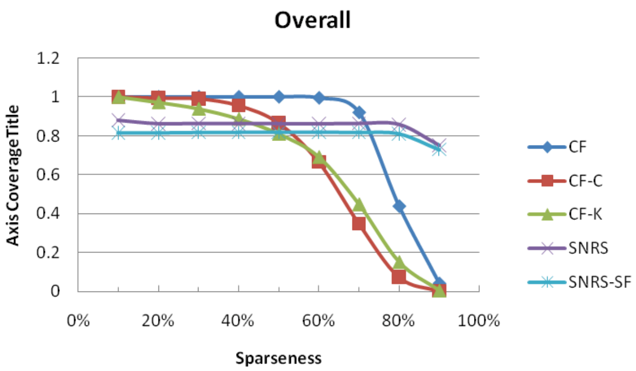
(a)


Agreement
Coverage
1.2
1
0.8
0.6
0.4
0.2
0
0%
20%
40%
60%
80%
100%
Sparseness
CF
CF-C
CF-K
SNRS
SNRS-SF

(b)


Writing
Coverage
1.2
1
0.8
0.6
0.4
0.2
0
0%
20%
40%
60%
80%
100%
Sparseness
CF
CF-C
CF-K
SNRS
SNRS-SF

(c)



(d)

Figure 9: The comparisons of the Coverage of CF, CF-C, CF-K, SNRS, and SNRS-SF for predicting fine-grained ratings on (a) interestingness (b) agreement, (c) writing, and (d) overall aspects of the articles.

7 Trust in SNRS

SNRS implicitly assumes that all users in the social network are trustworthy. However, in most recommender systems, this assumption is not necessarily valid. In this section, we shall discuss two trust issues, and propose how SNRS can be extended to handle them.

7.1 Shilling Attacks from Malicious Users

Intrigued by incentives, malicious users in recommender systems can purposely provide false reviews to promote their own products or attack similar products of competitors. For example, in a user-based collaborative filtering system, a malicious user can simply fake a set of reviews with the exact same ratings as those of a target user. Then this malicious user will be considered as the most similar user of the target user. If malicious users want to promote their own products, they can simply give the products high ratings, and these products will have a high chance of being recommended to the target user. This problem is known as shilling attacks.

The main reason that shilling attacks can become threats is that recommender systems rely too much on user rating similarity but overlook another important aspect, i.e., trust among users. Some studies have introduced explicit trust defined by users [16] and implicit trust inferred from user ratings [8], and have shown some improvements. However, unlike these approaches, SNRS is in essence built on trust, and thus it is able to handle shilling attack problems. Instead of using rating similarity, SNRS makes predictions by exploiting homophily among friends. Since users know their friends themselves, it is less likely for them to add malicious users as friends. If a user suspects that some friends may be potential malicious users, the user can remove those friends from the friend lists. Thus, in SNRS, the fact that two users are friends indicates the trust between them. In addition, with the capability of rating friends on each factor of a user's buying decisions (as discussed in Section 6), SNRS not only knows who are friends, but also on which aspect of buying decisions two friends trust each other. Therefore, the risk of shilling attacks can be further reduced.

7.2 Misleading by Friends with Unreliable Knowledge

It is worth pointing out that malicious users are not the only cause of the trust problems in recommender systems. Due to limited knowledge of target items, users who are trustworthy may still provide inaccurate reviews that do not truly reflect the truth of the items. Since SNRS relies on friends' opinions to make predictions, those inaccurate reviews will produce misleading recommendations of SNRS. For example, Alice has a taste similar to her friend Bob for Italian food, but Bob seldom goes to Thai restaurants. In this case, even though Bob is trustworthy to Alice, his opinion on Thai restaurants may not be so useful. To the best of our knowledge, little research if any has been devoted to solving problems caused by users with unreliable knowledge.

The key problem in this example is that the quantification of user correlations is based on all of the common items that every pair of users has reviewed, and it does not consider differences in item categories, like the difference between Thai food and Italian food. Conceptually, SNRS can solve this problem by introducing item clustering, e.g. the clustering of items based on their contents or rating similarities (as shown in Section 6). Therefore, SNRS can quantify two friends' homophily effects based only on the items within the same cluster as a target item. However, in practice, this solution may not work well because item clustering will make the data sparser.

To solve this problem, we propose to relax item categories when quantifying homophily effects. Instead of treating different categories as totally isolated, we consider some of them as still related based upon domain knowledge such as item taxonomies. For example, assuming we know from item taxonomies that Chinese food and Thai food are all Asian food, thus Chinese food is more similar to Thai food comparing to Italian food. Therefore, even though we cannot use Bob's preference for Italian food, we can still leverage his preference for Chinese food, if any, to guide the recommendation to Alice about Thai food.

In particular, we model item taxonomies into a type abstraction hierarchy (TAH) [6]. A TAH is often used to facilitate approximate query answering. It has a tree structure representing objects at different levels of abstraction. The leaf nodes in a TAH are usually the most specific objects. As the level goes up, the nodes in the TAH become more general. In Figure 10, we show a sample TAH generated from food taxonomy. Let us refer to the leaf nodes in a TAH generated from item taxonomy as item categories, such as Thai food and Chinese food. Thus, every item in the system can be mapped into a corresponding leaf node according to its category.

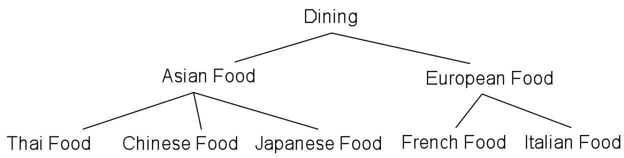


Figure 10: A TAH for relaxation of food styles

Let us assume that a target item belongs to category *T*; *C* refers to each category in item taxonomy; $\boldsymbol{I}_C$ is the set of items of category *C*. We define W_{CT} as the similarity between category *C* and category *T*. Thus, homophily effects among friends *U* and *V* can be estimated as,

$$Hist(W_{CT}(R_{UI} - R_{VI})) \qquad I \in \boldsymbol{I}(U) \cap \boldsymbol{I}(V) \tag{12}$$

Equation 12 counts *U*'s and *V*'s rating differences on all of their commonly reviewed items. But for each item *I* that they both reviewed, the contribution of the rating difference on *I* to the final histogram is multiplied by a factor of W_{CT} which is the similarity between the categories to which the target item and item *I* belong.

Given two categories *C* and *T*, the value of W_{CT} can be decided based on the following two observations. First, let us define *D(C, T)* as the distance from categories *C* and *T* to their lowest common ancestor *LCA(C, T)* in the TAH. (Note that *C* and *T* are the leaf nodes in the same depth.) The smaller the distance, the closer *C* and *T* are in the domain space; thus, they are more closely related. Second, categories in a specific domain are more strongly related to one another than in general domains. We use |*LCA(C, T)*| as the number of all the leaf nodes under *LCA(C, T)* to measure its generalities. The larger |*LCA(C, T)*|, the more general the domain space that both *C* and *T* belong to. Following these observations, we propose to measure WCT as in Equation 13.

$$W_{CT} = \begin{cases} 1 & \textit{if } C = T, \\ \dfrac{1}{D(C,T)\log_2(|LCA(C,T)|+1)} & \text{otherwise.} \end{cases} \tag{13}$$

Therefore, the similarity between Thai food and Chinese food is $\frac{1}{\log_2 4} = 0.5$; while the similarity between Thai food and Italian food is $\frac{1}{2\log_2 6} = 0.19$. Since 0.19 is less than 0.5, it is consistent with our intuition. Note that similar intuitions have been used to estimate the similarity between two concepts in a TAH [15]. The difference in our work is that we estimate the similarity between leaf nodes in a TAH, while [15] has no such a restriction. In addition, Equation 13 assumes a linear decay model of W_{CT} in $D(C,T)$, which is arguable. Future work can be made on selecting a better model to fit a specific domain.

Once we obtain W_{CT}, the homophily of a pair of users can be quantified (as shown in Equation 12). By doing so, even though these two users may not have enough commonly reviewed items in the same category as a target item, their rating correlations in other categories can remedy the data sparsity if used properly.

8 Related Work

Studies show that recommendations from friends are far more useful than those from recommender systems [28]. However, the systems that really utilize interpersonal relationships in social networks are few, if in fact there are any. Most recommender systems use information in social networks, especially user profiles, as an extra information resource to remedy the data sparsity issue. For example, in [23] the author uses contents in user profiles to find similar users. In [17] the authors use approximated predictions from contents in user profiles to "enrich" the original user/item rating matrix. However, none of them uses homophily among friends for inference.

Most directly related work is found in [33]. The authors proposed to combine social networks with recommender systems. They estimated the weights in collaborative filtering with an exponential function of the minimal distance of two users in a social network. This is an over-simplified correlation between users. Distance has no semantic meaning of similarity. Two distant friends may still share common opinions. As noted by the authors, this approach does not work well. [33] proposed another approach to reduce the computational cost in recommender systems by limiting the candidate similar users within a user's social network neighbors. This approach will actually exacerbate the data sparsity problem of a recommender system, because there are far fewer candidates of similar users than before. In contrast, we use a histogram of friends' rating differences to quantify homophily effects among friends rather than using their minimal distances. In addition, we consider the impact not only from immediate friends, but also from distant friends in an iterative classification.

9 Conclusions

Social networks provide an important source of semantic information regarding user behaviors and friend interactions. This information, especially homophily effects among friends, is valuable to recommender systems. Through statistical analyses of the dataset crawled from Yelp.com, we show that friends undoubtedly tend to review the same restaurants and give more similar ratings than non-friends. Based on these observations, we designed a social network-based recommender system—SNRS. To the best of our knowledge, this is the first attempt to incorporate the semantics of social networks into recommender systems.

SNRS predicts user ratings by exploiting information in social networks, including the user's own preferences, item's likability, and homophily effects among friends. It incorporates impacts from distant friends via an iterative classification. We evaluated the performance of SNRS with several other methods on the Yelp dataset through a 10 fold cross-validation, and SNRS achieves the best result. In terms of prediction accuracy, it yields a 14.3% improvement compared to that of CF; while in terms of coverage, it yields a 31% improvement compared to CF. In the sparsity test, SNRS returns consistently accurate predictions and high coverage over a wide range of data sparsity. Even in a cold-start test, SNRS still performs reasonably well. We also studied the role of distant friends in SNRS, and found that when the influences from distant friends are considered, the coverage of SNRS can be significantly improved with only a slight reduction in the prediction accuracy.

To deal with heterogeneities in social networks, we further proposed an approach for filtering social networks based on the semantics in fine-grained user ratings and ratings of friends. Using this approach, relevant friends can be selected for inference according to the type of target items. A specific class experiment was designed to evaluate the effectiveness of semantic filtering in the social network that was formed by a large group of graduate students. The experimental results reveal that SNRS with semantic filtering can further improve the prediction accuracy by 11.6%.

Finally, we investigated two trust issues in SNRS. We showed that SNRS has the capability of handling shilling attacks as well as the problems caused by friends with unreliable knowledge. Further research in these areas is desirable.

In our future work, we propose to study the performance of SNRS in other datasets, such as categories other than restaurants on Yelp. We also want to investigate how to apply SNRS to other Web 2.0 domains such as Facebook. For example, Facebook recently started personalizing user contents such as news feeds. Intuitively, our framework may also be applicable to the recommendations of news feeds, since the recommendation has to consider users' own preferences, the global popularity of news itself (i.e., item likability), and users' social networks.

References

1. G. Adomavicius and A. Tuzhilin. "Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions." IEEE Transactions on Knowledge and Data Engineering, 2005.
2. C. Basu, H. Hirsh, and W. Cohen. "Recommendation as Classification: Using Social and Content-Based Information in Recommendation." In Recommender System Workshop '98, pp. 11-15, 1998.
3. J. S. Breese, D. Heckerman, and C. Kadie. "Empirical Analysis of Predictive Algorithms for Collaborative Filtering." In Proceedings of the 14th Conference on Uncertainty in Artificial Intelligence, pp. 43-52, 1998.
4. D. Billsus and M. Pazzani. "Learning Collaborative Information Filters." In Proceedings of International Conference on Machine Learning, 1998.
5. V. Carvalho, and W. W. Cohen. "On the Collective Classification of Email Speech Acts." Special Interest Group on Information Retrieval, 2005.
6. W. W. Chu, K. Chiang, C. C. Hsu, and H. Yau, "An Error-Based Conceptual Clustering Method for Providing Approximate Query Answers." In Communications of the ACM, 39(13), December 1996.
7. B. Chandra, M. Gupta, and M. P. Gupta. "Robust Approach for Estimating Probabilities in Naïve-Bayes Classifier." In Pattern Recognition and Machine Intelligence, vol. 4815/2007, pp. 11-16, 2007.
8. J. O'Donovan and B. Smyth. "Trust in Recommender Systems." In Proceedings of IUI'05, January 9–12, 2005.
9. J. He and W. W. Chu. "A Social Network-Based Recommender System (SNRS)." Annals of Information Systems: Special Issue on Data Mining for Social Network Data (AIS-DMSND), Vol. 12, 2010.
10. Z. Huang,, H. Chen, D. Zeng (2004). Applying Associative Retrieval Techniques to Alleviate the Sparsity Problem in Collaborative Filtering. ACM Trans. on Information Systems, 22(1), 116–142.
11. J. L. Herlocker, J.A. Konstan, A. Borchers, and J. Riedl. "An Algorithmic Framework for Performing Collaborative Filtering." In Proceedings of ACM SIGIR, pp. 230-237. 1999.
12. S. Jurvetson. "What Exactly is Viral Marketing?" Red Herring, 78:110–112, 2000.
13. Q. Lu and L. Getoor. "Link-Based Classification." In Proceedings of the Twentieth International Conference on Machine Learning (ICML), pp. 496-503, 2003.
14. N. Lathia, S. Hailes, and L. Capra. "The Effect of Correlation Coefficients on Communities of Recommenders." In Proceedings of SAC'08, March 16-20, 2008.
15. W, Mao and W. W. Chu. "Free-text Medical Document Retrieval via Phase-based Vector Space Model " in Proceedings of American Medical Informatics Association (AMIA), Annual Symposium, 2002.

16. P. Massa and P. Avesani. "Trust-aware Collaborative Filtering for Recommender Systems." In Proceedings of Federated Int. Conference On The Move to Meaningful Internet: CoopIS, DOA, ODBASE, pp. 492-508, 2004.
17. P. Melville, R. J. Mooney, and R. Nagarajan. "Content-Boosted Collaborative Filtering for Improved Recommendations." In Proceedings of the Eighteenth National Conference on Artificial Intelligence (AAAI-2002), pp. 187-192, Edmonton, Canada, July 2002.
18. R. J. Mooney and L. Roy. "Content-Based Book Recommending Using Learning for Text Categorization." In Proceeding of ACM SIGIR'99 Workshop Recommender Systems: Algorithms and Evaluation, 1999.
19. M. Morita and Y. Shinoda." Information Filtering based on User Behavior Analysis and Best Match Text Retrieval." In Proceedings of the Seventh Annual Information ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 272-281, 1994.
20. M. McPherson, L. Smith-Lovin, and J. M. Cook. "Birds of a Feather: Homophily in Social Networks." Annu. Rev. Sociol. 2001.27:415-44.
21. J. Neville and D. Jensen. "Iterative Classification in Relational Data." In Proceedings of the Workshop on Learning Statistical Models from Relational Data at the Seventeenth National Conference on Artificial Intelligence (AAAI), pp. 13-20, 2000.
22. M. O'Connor and J. Herlocker. "Clustering Items for Collaborative Filtering." In Proceedings of the ACM SIGIR Workshop on Recommender Systems, Berkeley, CA, 1999.
23. M. Pazzani. "A Framework for Collaborative, Content-Based, and Demographic Filtering." Artificial Intelligence Rev., pp. 393-408, Dec. 1999.
24. M. Pazzani and D. Billsus. "Learning and Revising User Profiles: The Identification of Interesting Web Sites." Machine Learning, vol. 27, pp. 313-331, 1997.
25. P. Resnick, N. Iakovou, M. Sushak, P. Bergstrom, and J. Riedl. "GroupLens: An Open Architecture for Collaborative Filtering of Netnews." In Proceedings of Computer Supported Cooperative Work Conference, 1994.
26. P. Sen and L. Getoor. "Empirical Comparison of Approximate Inference Algorithms for Networked Data." ICML Workshop on Open Problems in Statistical Relational Learning, Pittsburgh, PA, 2006.
27. B. M. Sarwar, G. Karypis, J. Konstain, and J. Riedl. "Application of Dimensionality Reduction in Recommender Systems – A Case Study." In Proceedings of ACM WebKDD Workshop, 2000.
28. B. M. Sarwar, G. Karypis, J. Konstan, and J. Riedl. "Item-Based Collaborative Filtering Recommendation Algorithms." In Proceedings of 10th International Conference on World Wide Web (WWW'01), 2001.
29. R. Sinha, and K. Swearingen. "Comparing Recommendations made by Online Systems and Friends." In Proceedings of the DELOS-NSF Workshop on Personalization and Recommender Systems in Digital Libraries, 2001.
30. M. R. Subramani and B. Rajagopalan. "Knowledge-Sharing and Influence in Online Social Networks via Viral Marketing." Communications of the ACM, 46(12):300–307, 2003.

31. J. Wang, A. P. Vires, M. J.T. Reinders. “Unifying User-Based and Item-Based Collaborative Filtering Approaches by Similarity Fusion.” In Proceedings of the 29th Annual International ACM SIGIR Conference on Research & Development on Information Retrieval (SIGIR’06), August 6-11, 2006.
32. S. Yang and G. M. Allenby. “Modeling interdependent Consumer Preferences.” Journal of Marketing Research 40 282-294, 2003.
33. R. Zheng, F. Provost and A. Ghose. “Social Network Collaborative Filtering: Preliminary Results.” In Proceedings of the Sixth Workshop on eBusiness (WEB2007), December 2007.