Improving Recommendation Quality by Merging Collaborative Filtering and Social Relationships

Pasquale De Meo*, Emilio Ferrara**, Giacomo Fiumara*, Alessandro Provetti*

*Dept. of Physics, Informatics Section. **Dept. of Mathematics. University of Messina, Italy

{*pdemeo, eferrara, gfiumara, ale*}*@unime.it*

Abstract—**Matrix Factorization techniques have been successfully applied to raise the quality of suggestions generated by Collaborative Filtering Systems (CFSs). Traditional CFSs based on Matrix Factorization operate on the ratings provided by users and have been recently extended to incorporate demographic aspects such as age and gender. In this paper we propose to merge CF techniques based on Matrix Factorization and information regarding social friendships in order to provide users with more accurate suggestions and rankings on items of their interest. The proposed approach has been evaluated on a real-life online social network; the experimental results show an improvement against existing CF approaches. A detailed comparison with related literature is also present.**

Keywords-**Collaborative Filtering, Recommender Systems, Social Networks, Matrix Factorization**

I. INTRODUCTION

The term Collaborative Filtering (CF) refers to a wide range of algorithmic techniques targeted at learning users' preferences to recommend them items (like commercial products or movies) that are potentially relevant to their tastes [1]. CF techniques are an effective tool to support Web users in finding contents of their interest and, at the same time, to limit the amount of (often useless) information they receive when looking for recommendations.

CF techniques assume that users are allowed to rate available items over some scale and that the ratings are stored in a *user-rating matrix* $\mathbf{R}$. Generally, CF techniques work by comparing the ratings provided by users. Users who in the past have given similar ratings to the same objects can be used to give reliable estimates of a user's experience. The rating a user u_x would assign to an item i_j is predicted by considering the ratings to i_j provided by the users who are most similar to u_x in terms of ratings.

In the latest years, many researchers have been sought to improve the accuracy of CFS. In this scenario, *Non-Negative Matrix Factorization* techniques (in short, MF) [2] have been largely and successfully applied to CFS [3], [4], [5].

MF techniques take the user-rating matrix $\mathbf{R}$ as input and factorize it into the product of two matrices $\mathbf{P}$ and $\mathbf{Q}$ in such a way as to any row of $\mathbf{P}$ (resp., $\mathbf{Q}$) is associated with a user (resp., an item). Such a mapping has a nice interpretation that can be clarified through an example: assume to consider the movie domain and assume that each user is allowed to rate movies. MF techniques map the space of users (resp., movies) onto a *low dimensionality space* in which any dimension can be conceptually interpreted as a movie genre (like "drama" or "comedy").

Once such a mapping has been performed, each user is identified by a vector whose components specify how much she is interested in a given movie genre.

This has beneficial effects to raise the quality of produced recommendations. For instance, let us consider two users u_1 and u_2 and assume that both of them are strongly interested *only* in adventure movies and dislike other genres; finally assume that *none* of the movies watched by u_1 have been watched by u_2 and vice versa (even though this setting could appear unrealistic, in real cases it is quite common).

In such a case, the comparison of the rating histories of u_1 and u_2 would be not effective: in fact, since the overlap of the movies highly rated by u_1 and u_2 is empty, the movies liked by u_1 are deemed as not relevant to u_2 and vice versa.

Such a conclusion is, of course, counterintuitive because both u_1 and u_2 are interested to adventure movies and, therefore, it is likely that some of the movies liked by u_1 could be of interest to u_2, and vice versa. By contrast, if we would use MF techniques, we would compare two users on the basis of the genres they like, rather than on their past rating histories and, therefore, we would not incur in the mistakes described above.

Original MF techniques consider only user ratings and, after this, they have been extended in a range of directions: for instance, the approach of [3] considers different rating styles (i.e., the fact that some users tend to be more generous than others in assigning ratings) whereas other approaches incorporate demographic information about users, like gender or age [6]. Experimental trials show that the usage of auxiliary information is useful to produce more and more accurate recommendations [3], [6].

In our opinion, the approaches mentioned above could be extended in such a way as to consider the *social nature of the Web* and the *thick fabric of social relationships among Web users*. In fact, the emergence of collaborative platforms like Facebook or Flickr encouraged users to socialize in multiple ways: for instance, users join popular social networks like Facebook and spend the most of their time by interacting with their friends, sharing contents like photos or videos, and posting comments/reviews.

We believe that *social relationships* can provide an effective tool to raise the accuracy of the recommendation process. This corresponds to an intuitive idea: in real life, people often ask advices to their friends to take a decision and these advices play a crucial role in their final decisions.

The process of requiring the help of other people to produce recommendations is not new in Computer Science. In fact, several authors introduced the concept of *trust* between users and suggested to use trust values in conjunction with CF techniques to generate suggestions [7], [8], [9].

In this paper we propose to consider social relationships between users in conjunction with user ratings to generate recommendations. Our approach relies on MF techniques, but introduces some novel contributions.

First of all, differently from traditional approaches, our approach merges both information about user ratings and information on social relationships. In detail, our approach to learning the latent space relies on the idea that if two users are tied by a social relations like friendship, then they should be mapped onto "close" vectors in the latent space. This has a relevant consequence: opinions/ratings of friends of a user u_x will be more influential than opinions/ratings provided by users who are unknown to u_x.

As a further contribution, our approach suggests to use social relationships instead of trust ones. This provides three main novelties: the first is that trust relationships are generally *asymmetric*, in the sense that if a user trusts another one, the opposite may not hold true. By contrast, social relationships can be both *symmetric* (e.g., friendship relations) and *asymmetric* (e.g., a user u_x who posts a comment on the blog of a user u_y but u_y never replied u_x).

As a second novelty, users are required to explicitly declare what users they trust: for instance, in a system like Slashdot[1] users are allowed to declare their friends and foes. As for social relationships, we can use *explicit* information provided by the users but we can also *unobtrusively* monitor user behaviors to learn her preferences and her relationships with others. For instance, we can analyze the comments posted by multiple users on a given subject (like the review of a commercial product) to learn the personal opinion of a particular user as well as to identify pairs of users showing divergent (resp., convergent) opinions.

Finally, as for the third novelty, we observe that the definition of trust relationships relies on the fact that a user explicitly assumes that her interactions with another one are beneficial to her. By contrast, in social relationships, there are many reasons driving two users to interact and some of them *do not necessarily imply* that a user get some benefit from another one: for instance, a user could get in touch with another only to expand her knowledge on a topic.

The plan of the paper is as follows: in Section II we

review MF techniques for Collaborative Filtering. In Section III we describe our approach. In Section IV we illustrate the experiments we carried out to validate our technique. Related literature is discussed in Section V. Finally, in Section VI we draw our conclusions.

II. MATRIX FACTORIZATION FOR COLLABORATIVE FILTERING

In the latest years, many researchers proposed to apply *dimensionality reduction* (DR) techniques to increase the accuracy of Collaborative Filtering (CF) techniques.

In detail, a CF algorithm operates on a matrix $\mathbf{R}$ called *user-rating* matrix. The $\mathbf{R}$ matrix has $n_u \times n_i$ entries, being n_u (resp., n_i) the number of users (resp., items); its generic entry $\mathbf{R}_{xj}$ is the rating the user u_x applied on the item i_j. We set $\mathbf{R}_{xj} = $ "$\star$" if the user u_x has not rated i_j. The matrix $\mathbf{R}$ is *quite sparse* because in real-life domains, the number of available items is usually very large in comparison with the number of items typically evaluated by a user. Data sparsity is one of the major drawbacks plaguing CF algorithms; in fact, it negatively affects the computation of similarities between users and items and this, ultimately, yields poor results in predicting unknown ratings [10].

DR techniques have proven to be effective in fighting against data sparsity [11]. The key idea of DR techniques is to *map the user-rating* matrix onto a *latent space* of dimension k, being k a fixed integer. Due to this mapping, an item i_j is represented by an *item vector* $\mathbf{q}_j \in \mathbb{R}^k$ and a user u_x is associated with a *user vector* $\mathbf{p}_x \in \mathbb{R}^k$.

A dimension in the latent space identifies a feature describing an item or a user; for instance, if we consider the movie domain, the dimensions may be interpreted as the genres of a movie. For a given item i_j, the generic entry $\mathbf{q}_j[\ell]$, $0 \leq \ell \leq k$ of the item vector specifies whether i possesses the ℓ-th factor and the strength of this possession. An analogous interpretation holds for user vectors.

To better clarify these concepts, let us consider again the movie domain; a possible latent space associated with it would consist of movie genres like "comedy," "adventure," "romance" and "horror." An item i_j could be represented as $\mathbf{q}_i = [0.5, 0.7, 0, 0]$ specifying that the genre of i_j is a blend of "comedy" and "adventure."

In the latent space, the approval rating of a user u_x to an item i_j is computed as the inner product $\mathbf{p}_x \cdot \mathbf{q}_j$.

A possible approach to learning the latent space is given by *Singular Value Decomposition (SVD)*. The SVD approximates $\mathbf{R}$ by means of its eigenvalues and it has been widely and successfully applied in the Information Retrieval context [12], [13]. Unfortunately, in the context of CF, the matrix $\mathbf{R}$ is *not only sparse* but many of its entries are also *unspecified*, i.e., there are a lot of entries labeled with the symbol "$\star$" because many users do not have rated a large number of items. In such a case, a user may not be aware about the existence of an item and, therefore, we can not conclude

that she likes/dislikes the item itself. In such a configuration, SVD shows poor performance [4].

Recently, many researchers suggested to apply *matrix factorization techniques* to CF systems [4], [3], [5], [6]. In this case, the computation of the latent space requires to solve a suitable *optimization problem.*

More formally, let $\mathbf{P}$ (resp., $\mathbf{Q}$) be a $n_u \times k$ (resp., $n_i \times k$) matrix such that the x-th row $\mathbf{p}_x$ of $\mathbf{P}$ represents the vector associated with the user u_x; in an analogous fashion, the j-th row $\mathbf{q}_j$ of $\mathbf{Q}$ represents the vector associated with the item i_j . The optimization problem to solve is

$$\min_{\langle \mathbf{P}, \mathbf{Q} \rangle} \mathcal{L} = \min \frac{1}{2} ||\mathbf{R} - \mathbf{P}\mathbf{Q}^T||_F^2 + \frac{\lambda}{2} \left(||\mathbf{P}||_F^2 + ||\mathbf{Q}||_F^2 \right) \quad (1)$$

Here $\mathbf{Q}^T$ is the transpose of the matrix $\mathbf{Q}$ whereas the symbol $|| \cdot ||_F$ denotes the *Frobenius* norm of a matrix[2]. Finally, the term $\frac{\lambda}{2} \left(||\mathbf{P}||_F^2 + ||\mathbf{Q}||_F^2 \right)$ is known as *Tikhonov regularization* and it is used to avoid *overfitting*. The parameter λ is usually computed by applying *cross-validation.*

A popular strategy to optimize the function $\mathcal{L}$ is based on the so called *gradient descent method* [2]. To implement such a strategy, we must consider *only* the ratings actually provided by the users and ignore missing entries in $\mathbf{R}$. To this purpose, if we set $\delta_{xj} = 1$ if u_x has rated i_j and 0 otherwise, the optimization problem to solve is

$$\min \frac{1}{2} \sum_{x=1}^{n_u} \sum_{j=1}^{n_i} \delta_{ij} \left(\mathbf{R}_{ij} - \mathbf{p}_x \cdot \mathbf{q}_j \right)^2 + \frac{\lambda}{2} \left(||\mathbf{P}||_F^2 + ||\mathbf{Q}||_F^2 \right) \tag{2}$$

The gradient descent procedure consists of four steps:

1) The vectors $\mathbf{p}_x$ and $\mathbf{q}_j$ are initialized at random.
2) The partial derivatives of $\mathcal{L}$ are computed.
3) The vectors $\mathbf{p}_x$ and $\mathbf{q}_j$ are updated in the direction opposite to the partial derivatives

$$\mathbf{p}_x^{'} = \mathbf{p}_x - \beta \frac{\partial \mathcal{L}}{\mathbf{p}_x} \qquad \mathbf{q}_j^{'} = \mathbf{q}_j - \beta \frac{\partial \mathcal{L}}{\mathbf{q}_j}$$

Here β is a threshold to be determined.
4) Steps 2-3 are iterated until a particular number of iterations has been carried out or the improvement of the function $\mathcal{L}$ is less than a given threshold ε.

III. APPROACH DESCRIPTION

In this section we describe our approach to merge social relationships with user ratings. We assume that users can create various type of social relationships (e.g., getting friends or affiliating to the same groups). We say that a *social tie* exists between two users if they created a social relationship. A social tie can be *symmetric* or *asymmetric*. To introduce our approach, we need the following definitions:

---

[2]The *Frobenius norm* of a matrix $\mathbf{A}$ is defined as $||\mathbf{A}||_F = TR\left(\mathbf{A}\mathbf{A}^T\right)$ being TR the trace of the matrix $\mathbf{A}$, i.e., the sum of the elements located on its main diagonal.

Definition 1: Let $U = \{u_1, u_2, \ldots, u_{n_u}\}$ be a set of users and $I = \{i_1, i_2, \ldots, i_{n_i}\}$ be a set of items. A *Social Rating Network (SRN)* is a 4-tuple $SRN = \langle U, I, \phi, \psi \rangle$ where:

- $\phi : U \times I \rightarrow \mathbb{R}^+ \cup \{``\star''\}$ is a function (called *rating function*) which associates a user $u_x \in U$ and an item $i_l \in I$ with a real number r_{xl} if u_x rated i_l with r_{xl}, and with the symbol "$\star$" if u_x has not rated i_l.
- $\psi : U \times U \rightarrow \{0, 1\}$ is a function (called *social function*) which takes a pair of users $u_x, u_y \in U$ as input and returns 1 if and only if a *social tie* exists between them and 0 otherwise.

Due to the definition of social tie, the function ψ is generally *asymmetric*, i.e., $\psi(u_x, u_y) \neq \psi(u_y, u_x)$, for some pairs of users u_x and u_y. The function ψ is also useful to define the concept of *neighborhood* of a user u_x in an SRN:

Definition 2: Let $SRN = \langle U, I, \phi, \psi \rangle$ be a Social Rating Network and $u_x \in U$ be a user. The *neighborhood* N_x of u_x is defined as the set of users having a social tie with u_x

$$N_x = \{u_y \in \mathcal{U} : \psi(u_x, u_y) = 1\}$$

The concept of SRN specializes the concept of social network; in fact, unlike traditional social networks, users are not only allowed to *interact* among each others but also to *rate* objects. To describe our approach, we start observing that Equation 1 provides an effective tool for tackling data sparsity but it is *agnostic* about social relationships because no terms related to social ties involving existing users appear in it.

However, in reality, social relationships are a powerful tool for producing suggestions: for instance, if a person is uncertain about the purchase of a good, she often asks her friends/acquaintances opinions or advices that often play a crucial role in her final decision.

Our goal is, therefore, to *extend* Equation 1 to the case of SRNs by adding terms capable of taking into account social ties among users. To simplify the discussion, henceforth we shall focus on friendship relations as models of social relationships; however, without any loss of generality, the conclusions we will draw are still valid for other type of social relationships.

We start observing that, if we would solve the optimization problem in Equation 1, *we would map any user u_x onto a point $\mathbf{u}_x$ in the latent space*. We guess that, if two users are friends, then they should be mapped onto *close points*, in the latent space; in other words, given three users u_x, u_y and u_z, such that only u_x and u_y are friends, the distance between the points $\mathbf{p}_x$ and $\mathbf{p}_y$ must be less than the distance between $\mathbf{p}_x$ and $\mathbf{p}_z$.

Such a requirement has an easy explanation: in fact, if a user u_x is close to her friends in the latent space, the opinions of the friends of u_x will be more relevant to recommend items to u_x than the opinions of other users who are unknown to u_x. From a mathematical standpoint, the

distance between two users u_x and u_y is given by $||\mathbf{p}_x - \mathbf{p}_y||$, being $||\cdot||$ the euclidian norm in the latent space. If we denote as N_x the neighborhood of u_x, our aim is that the term $\sum_{u_y \in N_x} ||\mathbf{p}_x - \mathbf{p}_y||$ is as small as possible.

These considerations suggest to add a *penalty term* to Equation 1. In detail, among the all possible mappings to the latent space, we decide to penalize those mappings in which users tied by a friendship relation are mapped onto "far" points. In the light of these considerations, the optimization problem to solve is as follows

$$\min_{\langle \mathbf{P}, \mathbf{Q} \rangle} \mathcal{L}' = \min \frac{1}{2} ||\mathbf{R} - \mathbf{P}\mathbf{Q}^T||_F^2 +$$
$$+ \lambda \left(||\mathbf{P}||_F^2 + ||\mathbf{Q}||_F^2 \right) + \mu \sum_{x=1}^{n_u} \sum_{u_y \in N_x} ||\mathbf{p}_x - \mathbf{p}_y|| \tag{3}$$

As in the previous case, λ and μ are two constants to be tuned to avoid overfitting. We applied the gradient descent algorithm to solve our optimization problem. In such a case, the partial derivatives of the objective function $\mathcal{L}'$ are

$$\frac{\partial \mathcal{L}'}{\partial \mathbf{p}_x} = \sum_{j=1}^{n_i} \delta_{ij} \left(\mathbf{p}_x \cdot \mathbf{q}_j - \mathbf{R}_{xj} \right) \mathbf{q}_j + \lambda \mathbf{p}_x + \mu \sum_{u_y \in N_x} \left(\mathbf{p}_x - \mathbf{p}_y \right)$$

$$\frac{\partial \mathcal{L}'}{\partial \mathbf{q}_j} = \sum_{x=1}^{n_u} \delta_{ij} \left(\mathbf{p}_x \cdot \mathbf{q}_j - \mathbf{R}_{xj} \right) \mathbf{p}_x + \lambda \mathbf{q}_j$$

IV. EXPERIMENTS

In this section we present the experiments we carried out to evaluate our approach. We built a social networks (called Cofe) in which users were allowed to rate movies. The early users of Cofe were students enrolled in a BsC degree in Computer Science at our University; after this, students were allowed to invite their friends to join Cofe, create friendship relationships with other members, insert movie titles and rate them.

We gathered data on 37 students and 297 movie ratings. We used as metric to assess the quality of recommendations the *Root Mean Square Error* (RMSE); to define it, assume to *randomly* split the dataset in two parts called *training* and *test set*. The training set is used to perform matrix factorization whereas the test set is used to assess the quality of recommendations. If we define *(i)* r_{xj} the rating u_x assigned to i_j, *(ii)* $\hat{r}_{xj}$ the rating predicted by a given method for u_x and i_j and *(iii)* N_t the size of training set, the RMSE is defined as $RMSE = \sqrt{\frac{1}{N_t} \left(r_{xj} - \hat{r}_{xj} \right)^2}$.

We compare our approach with the following methods:

- *Naive*. In this method, the rating of an item is predicted as the average of the ratings provided by all users.
- *NMF*. In this method, we solve the optimization problem in Equation 1 to predict missing ratings.

To run both *NMF* and our approach (hereafter, *OA*) we fixed, after a pre-tuning activity, $\lambda = 0.001$. In addition, we

k	RMSE		
	Naive	*NMF*	*OA*
2	0.897	0.825	0.816
4	0.897	0.751	0.742
6	0.897	0.739	0.731
8	0.897	0.727	0.719
10	0.897	0.71	0.705
12	0.897	0.715	0.707
14	0.897	0.717	0.713
16	0.897	0.717	0.714

Table I
RMSE OF *Naive*, *NMF* AND *OA* RESULTS FOR DIFFERENT VALUES OF k

μ	*RMSE*
10^{-1}	0.695
10^{-2}	0.68
10^{-3}	0.675
10^{-4}	0.687
10^{-5}	0.707
10^{-6}	0.71

Table II
IMPACT OF μ ON RMSE

fixed the number of iterations of both *NMF* and *OA* equal to 5000.

In a first experiment we studied how the value of k (i.e., the number of dimensions in the latent space) impacted on the accuracy. We fixed $k = \{2, \ldots, 16\}$ and ran both *NMF* and *OA*; of course, the *Naive* algorithm is not influenced by k. The obtained results are reported in Table I.

From the analysis of this table we can conclude that:
- *OA* significantly outperforms the *Naive* method and is better than *NMF*.
- *OA* and *NMF* achieve their best performance when k is around 10. In fact, if the number k of dimensions exploited to represent user preferences and movie features is too low, it is not possible to accurately capture the differences between two movies or two users and, therefore, it is hard to correctly recommend movies to users.

By contrast, if k exceeds 10, the RMSE of both *OA* and *NMF* does not significantly decrease. This means that a number of latent factors equal to 10 is enough to correctly capture movie genres and a further increase of k is useless.

However, the higher k, the larger the size of $\mathbf{P}$ and $\mathbf{Q}$ and, therefore, the larger the space of memory required to store them. These considerations indicate us that a good trade-off between space requirements and recommendation accuracy is achieved when $k = 10$.

As a further experiment, we tuned the value of μ, i.e., we studied how the weight associated with social relationships reflect on the quality of suggestions. The RMSE value (when $k = 10$) and μ, ranged from 10^{-6} to 10^{-1}, are reported in Table II.

From Table II we observe that if $\mu \to 0$, the term representing social relationship tends to vanish and, therefore, *OA* degenerates into *NMF* (i.e., only user ratings are

considered). By contrast, if $\mu \to 10^{-1}$, an opposite effect arises: information on social relationships dominates over user ratings. The best value of RMSE is achieved if μ is around 10^{-3} because, in such a configuration, our approach takes advantage of both user ratings and social relationships.

V. RELATED WORKS

In this section we describe some approaches related to our research. In detail, we first consider Matrix Factorization (MF) techniques and describe how they have been applied in the context of Collaborative Filtering (CF); after this, we highlight the main novelties introduced by our approach. Then, we focus on *Trust-Based Collaborative Filtering Systems*, i.e., on CF systems in which users are allowed to explicitly declare if they trust (and sometimes distrust) other users. We explain the differences between trust-based CF systems and our research efforts.

A. Matrix Factorization Techniques for CF

The notion of Non-Negative Matrix Factorization (in short, MF) was introduced for the first time in the seminal paper of Lee and Seung in 1999 [2]. MF techniques have been widely applied in multiple domains like data clustering [14] and bioinformatics [15].

The algorithm proposed in [2] to perform matrix factorization is *iterative* and it strongly resembles the gradient descent method discussed in this paper.

One of the first approaches that exploited MF in the context of Collaborative Filtering is [16]. In that paper, the authors observe that the task of factorizing two matrices is computationally challenging and, therefore, they propose a strategy called *Alternating Least Square (ALS)*.

ALS-based techniques proceed iteratively and each iteration consists of two stages: in the first stage the matrix $\mathbf{P}$ is fixed and the problem described in Equation 1 is solved with respect to the matrix $\mathbf{Q}$. In the second stage, the vice versa. In both the two stages, the optimization problem can be reformulated (and solved) as a *least square* problem [16], [4].

ALS techniques lend themselves to a massive amount of parallelization. Therefore, the growing success of distributed computing platform like HADOOP, envisages a big success of these techniques.

As further extensions, the approach of [3] suggests to add *bias terms* in Equation 1; these terms model the fact that some users tend to provide more generous ratings than others or that some items tend to receive ratings generally higher than other items. In the same paper, the authors study how temporal changes in user preferences and/or in item properties (e.g., the fact that the popularity of an item may decay over time) impact on the process of generating recommendations.

Some approaches target at learning the latent space in presence of *implicit user feedbacks* [5]. In this scenario, users do not directly provide ratings on items but the analysis of user behaviours is useful to understand user preferences. For instance, in an e-commerce Web site, we can assume that a user likes an item if she bought it. Finally, some authors [6] suggest to incorporate in Equation 1 also *demographic variables* like the gender or the age of the users.

Differently from the works above, in our approach, the task of learning the latent space relies on information tied to both *user ratings* and *their social relationships*. In other words, we merge information about social relationships and user behaviours (encoded as the ratings they provided) in a *unique framework* to produce recommendations.

B. Trust-based CF systems

The usage of trust information to generate recommendations has been largely explored in the literature.

Some approaches suggest to use trust values between pairs of users in conjunction with Collaborative Filtering techniques to produce recommendations [17], [7].

The key challenge in trust-based approaches is represented by the computation of trust values. In fact, in most cases, users provide few explicit declarations of trust; this implies that trust values among many pairs of users are unknown and, therefore, a mechanism for inferring new trust values from existing ones must be designed.

Some approaches assume that existing users form a community which is modeled through a graph G whose nodes represent users and edges indicate relationships between them [8], [18]. In [8] the authors apply a modified version of the Breadth First Search algorithm on G to infer multiple values of trust for each user; the average of these values is computed to produce a final trust value.

The approach of [18] considers paths up to a fixed length in G and propagates trust values explicitly declared by users on them to infer new ones.

Recently, some approaches studied the problem of propagating trust in *signed social networks*, i.e., social networks in which ties between users may be either *positive* (indicating, for instance, that two users are friends) or *negative* (indicating a relationship like antagonism) [19], [20]. These approaches are grounded in *balance theory* introduced in social psychology. Roughly speaking, balance theory is based on principles like *the enemy of my friend is my enemy* and *the friend of my enemy is my enemy*.

Differently from the approaches described above, our approach is based on *social relationships*. This has two major effects: *(i)* Social relationships are *weaker* than trust ones: in fact, in a social network two users may interact for different reasons (e.g., to be informed on a new topic). By contrast, if a user u_x trusts a user u_y, this means that u_x has a reasonable expectation that interactions with u_y will be beneficial for her. *(ii)* Trust Relationships are *asymmetric*. On the contrary, social relationship can be asymmetric as well as symmetric.

During the latest years, the possibility of exploiting the social relationships together with the rating behaviors has been advanced [21], [22]. In [21] the authors suggest that drawing on similarity and familiarity between the users in their rating activities could support the decision making and increase recommendation quality. In [22] authors propose a model that combines social ties and ratings to improve the movie recommendation quality in the context of a real-world social rating network, providing encouraging results. With this paper we additionally substantiate the hypothesis that combining Collaborative Filtering and social relationships is helpful in order to build better Recommender Systems.

VI. Conclusions

In this work we presented a novel strategy to provide heightened quality recommendations to users, in the context of Social Rating Networks, those Social Networks in which usersare allowed to socially interact and to rate items. Our approach relies both on user ratings and on social relationships among users and merges this information by means of Matrix Factorization techniques. Experimental results show the effectiveness of our approach.

We plan to extend our approach to Trusted Social Networks, i.e., social networks in which users are allowed to define positive or negative trust in other users. Another issue to explore is related to the scalability of our approach. In detail, we plan to design and implement efficient (and possibly distributed) algorithms to perform Matrix Factorization.

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*, pp. 734–749, 2005.

[2] D. Lee and H. Seung, "Learning the parts of objects by non-negative matrix factorization," *Nature*, vol. 401, no. 6755, pp. 788–791, 1999.

[3] Y. Koren and R. Bell, "Advances in collaborative filtering," *Recommender Systems Handbook*, pp. 145–186, 2011.

[4] Y. Koren, R. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," *Computer*, vol. 42, no. 8, pp. 30–37, 2009.

[5] I. PilÁSzy, D. Zibriczky, and D. Tikk, "Fast als-based matrix factorization for explicit and implicit feedback datasets," in *Proceedings Of The 4th ACM Conference On Recommender Systems*. ACM, 2010, pp. 71–78.

[6] Y. Seroussi, F. Bohnert, and I. Zukerman, "Personalised rating prediction for new users using latent factor models," in *Proceedings Of The 22nd ACM Conference On Hypertext And Hypermedia*. ACM, 2011, pp. 47–56.

[7] P. Massa and P. Avesani, "Trust-aware collaborative filtering for recommender systems," *On The Move To Meaningful Internet Systems 2004: Coopis, Doa, And Odbase*, pp. 492–508, 2004.

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

[9] J. Odonovan, "Capturing trust in social web applications," *Computing With Social Trust*, pp. 213–257, 2009.

[10] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, "Analysis of recommendation algorithms for e-commerce," in *Proceedings Of The 2nd ACM Conference On Electronic Commerce*, 2000, pp. 158–167.

[11] ——, "Application of dimensionality reduction in recommender system-a case study," in *ACM Webkdd Workshop*, 2000, pp. 285–295.

[12] C. Manning, P. Raghavan, H. SchÜTze, and E. Corporation, *Introduction To Information Retrieval*. Cambridge University Press, 2008, vol. 1.

[13] C. Martin and M. Porter, "The extraordinary svd," *Arxiv:1103.2338*, 2011.

[14] C. Ding, X. He, and H. Simon, "On the equivalence of nonnegative matrix factorization and spectral clustering," in *Proceedings Of The Siam Data Mining Conference*, 2005, pp. 606–610.

[15] I. Jung, J. Lee, S. Lee, and D. Kim, "Application of nonnegative matrix factorization to improve profile-profile alignment features for fold recognition and remote homolog detection," *Bmc Bioinformatics*, vol. 9, no. 1, p. 298, 2008.

[16] R. Bell and Y. Koren, "Scalable collaborative filtering with jointly derived neighborhood interpolation weights," in *Proceedings Of The 7th IEEE International Conference On Data Mining*. IEEE, 2007, pp. 43–52.

[17] J. Golbeck and A. Mannes, "Using trust and provenance for content filtering on the semantic web," in *Proceedings Of The Models Of Trust For The Web Workshop*, 2006.

[18] R. Guha, R. Kumar, P. Raghavan, and A. Tomkins, "Propagation of trust and distrust," in *Proceedings Of The 13th International Conference On World Wide Web*. ACM, 2004, pp. 403–412.

[19] J. Kunegis, A. Lommatzsch, and C. Bauckhage, "The slashdot zoo: Mining a social network with negative edges," in *Proceedings Of The 18th International Conference On World Wide Web*. ACM, 2009, pp. 741–750.

[20] J. Leskovec, D. Huttenlocher, and J. Kleinberg, "Predicting positive and negative links in online social networks," in *Proceedings Of The 19th International Conference On World Wide Web*. ACM, 2010, pp. 641–650.

[21] P. Bonhard and M. Sasse, "knowing me, knowing youusing profiles and social networking to improve recommender systems," *Bt Technology Journal*, vol. 24, no. 3, pp. 84–98, 2006.

[22] A. Said, E. De Luca, and S. Albayrak, "How social relationships affect user similarities," in *Proceedings Of The 2010 Workshop On Social Recommender Systems*, 2010, pp. 1–4.