Trust-Based Local and Social Recommendation

Simon Meyffret
Université de Lyon, CNRS
INSA-Lyon, LIRIS, UMR5205,
F-69621, France
simon.meyffret@liris.cnrs.fr

Lionel Médini
Université de Lyon, CNRS
Université Lyon 1, LIRIS,
UMR5205,
F-69622, France
lionel.medini@liris.cnrs.fr

Frédérique Laforest
Université de Lyon, CNRS
Telecom Saint-Etienne, LT2C,
F-42000, France
frederique.laforest@univ-st-etienne.fr

ABSTRACT

In this article, we propose an evolution of trust-based recommender systems that only relies on local information and can be deployed on top of existing social networks. Our approach takes into account friends' similarity and confidence on ratings, but limits data exchange to direct friends, in order to prevent ratings from being globally known. Therefore, calculations are limited to locally processed algorithms, privacy concerns can be taken into account and algorithms are suitable for decentralized or peer-to-peer architectures.

We have implemented and evaluated our approach against five others, using the Epinions trust network. We show that local information with good default scoring strategies are sufficient to cover more users than classical collaborative filtering and trust-based recommender systems. Regarding accuracy, our approach performs better than most others, specially for cold start users, despite using less information.

Categories and Subject Descriptors

H.3.3 [**Information Search and Retrieval**]: Information filtering

General Terms

Algorithms, Design, Experimentation

Keywords

recommendation, social network, trust, local knowledge, score propagation

1. INTRODUCTION

The Internet gives users easy and immediate access to a lot of resources (documents, media, services, etc.). Among this abundance of items, information overload is an ever growing problem. Recommender systems are one classically proposed solution to cope with this problem [18]. Collaborative filtering recommender systems rely on a users graph to predict items that could fit users' interests based on related users ratings. The users graph can typically be a profiles similarity graph [1]. Trust-based recommender systems build the users graph with links that express the trust users have on the opinion of others [14, 9, 8]. A typical example is the Epinions[1] web site [16]. The main problems of these systems rely on the ratings sparsity and on the so-called cold start users who have no or very few connections in their graph. To remedy sparsity, trust-based recommender systems increase the number of users relations by propagating trust relations based on transitivity property.

But propagating relations cannot be reproduced in social networks where relations are explicitly decided by users (even when invited by the social network provider). Moreover, due to privacy concerns, the trend is to limit the transmission of information on a user only to authorized related users.

To go further in the future, efforts are conducted to decentralize social networks infrastructures (*e.g.* Diaspora), so as to impede their owners to gather all information on people. The FreedomBox[2] concept imagines that each user will have his/her own node hosted in his/her own web server. But many recommender systems rely on global knowledge to predict missing ratings: they use the whole users graph. For instance, collaborative filtering systems predict ratings *a priori* using a costly computational process that builds the whole users' similarity graph from all users' profiles. Such a mechanism requires ideal conditions where all ratings are fully known by the system and therefore it can hardly be used in decentralized architectures. Moreover users have to share personal data with the system in order to get recommendation. On the other side, local approaches use knowledge limited to a subpart of the users' graph and profiles. Each rating calculation is processed using the requesting user's profile and the set of directly linked users, it does not require the system to know personal data from users. This constraint enhances privacy, requires lighter processing and complexity is independent of the number of users. They are also feasible on decentralized architectures. Local approaches do not require *a priori* processing of predictions; they can predict ratings on demand. Local approaches are *in fine* compatible with P2P architectures and allow strong privacy strategies.

In this paper we propose a recommender system that relies on local knowledge in order to enhance privacy and in

[1]http://www.epinions.com
[2]http://wiki.debian.org/FreedomBox

order to be implementable in P2P architectures with decentralized data. We want to introduce recommendation in social networks, where classical e-commerce recommender systems are not adapted. Our system does not modify the users social network and shares information only between authorized related users, therefore it could be implemented as an additional service in existing social networks such as Facebook, Diaspora or LastFM.
We herein propose and evaluate a recommender system that propagates ratings locally in a users' network without propagating relations. We have used the Epinions dataset so as to compare our algorithms with five collaborative filtering and trust-based recommender systems. We show that a well-chosen limited vision of the ratings remains efficient in ratings prediction, even with a sparse dataset. Besides, our local approach provides a P2P compatible architecture taking into account users privacy concerns: users can explicitly define whom they want to communicate with.

The paper is organized as follows: we first present existing approaches used in the literature (section 2), we show a motivating example used to explain our algorithms in section 3 and define our local approach using a social network in section 4. In section 5, we compare and evaluate our approach regarding main systems described in the state of the art. We then conclude and discuss this approach.

2. STATE OF THE ART

Collaborative filtering systems try to predict the rating of items for a particular user based on the items previously rated by other users [1]. They rely on users graph and users profiles knowledge. Users profiles contain ratings from users to items, purchased history, interest, etc. The users graph contains users (the nodes) and links between them (relations between users). Typically, those links are Pearson's correlation coefficients computed between users [2]. To recommend items to a specific user, the system uses the profile of similar users, *i.e.* users directly linked to the user in the users graph. Although they are successfully used by most e-commerce websites, collaborative filtering systems are not well-protected against malicious [11] or peculiar [17] users and hardly cope with cold start users (who rated few or no items), or with the overall sparsity of existing ratings [7]. [2] propose to use default voting in order to cope with sparsity. When two users has not enough ratings in common to compute a similarity, then some default ratings are introduced. Those can be a neutral rating, the item mean rating or the user mean rating.

Trust-based recommender systems invite users to state that they trust the ratings expressed by other users [14, 9, 8]. They address the above-described drawbacks by using trust instead of similarity. Cold start users do not need to rate items to start using the system, they need to trust other users [9, 15]. Security is improved since trust relations make users know their direct relations. They have to deal with the sparsity of trust networks (In Epinions [16], a user trusts in average 8 users). To do so, they propagate trust in the network creating new relations between users [3].
[10, 5] propose several definitions of trust. In this paper, trust is defined as the belief of a user in the usefulness of information provided by another user [7].

To predict a rating for a specific user, collaborative filtering and trust-based approaches usually aggregate other users' ratings with the following function [1]:

$$r_{a,i} = \frac{\sum_{a' \in A_i} \omega_{a,a'} \times r_{a',i}}{\sum_{a' \in A_i} \omega_{a,a'}} \tag{1}$$

Where $r_{a,i}$ is the rating given by user a to item i, A_i is the set of users having rated item i and $\omega_{a,a'}$ is a weight between users a and a', typically a similarity or a trust coefficient.

In this paper, we call *GlobalCF* (respectively *ItemBased*) the collaborative filtering algorithm defined in eq.1 using the Pearson's correlation coefficient ρ between users' ratings (resp. items' ratings) as ω similarity coefficient [2]:

$$\rho = \frac{\sum_{n=1}^{N} (r_n - \bar{r})(p_n - \bar{p})}{N \times \sqrt{\sum_{n=1}^{N} (r_n - \bar{r})^2} \times \sqrt{\sum_{n=1}^{N} (p_n - \bar{p})^2}} \tag{2}$$

GlobalCF (resp. ItemBased) relies on global knowledge on users (resp. items) profiles to compute similarities between users (resp. items). It constructs a similarity graph and selects similar users (resp. items) to predict missing ratings. Both approaches require ideal conditions where all ratings are known by the systems and all users and items can be easily identifiable.

Traditional trust-based recommender systems initially rely on user-defined trust values to provide the ω weight defined in eq.1. They use these values to construct the users graph. Trust is considered as a transitive property, so that the graph can be automatically updated by propagating inferred trust values from previous calculations [4, 9]: if a trusts b and b trusts c, a new trust value from a to c is defined and a new link between users a and c is added in the graph. Several iterations are performed in order to explore the graph up to a given depth.

MoleTrust [9] predicts the trust value of a source user to a target user by gradually propagating trust in the users graph, up to a given depth k. If more than one trust path links two users, the mean of all computable trusts is used. This approach thus requires an extended-local knowledge of the trust network, since all the paths surrounding those users must be followed to compute the final trust value. Moreover, the whole users graph is explored up to depth k in order to compute the final score.

TidalTrust [3] takes into account shortest paths with a modified breadth first search in the trust network. It also requires an extended-local knowledge of the users graph.

TrustWalker [6] is a hybrid approach between trust-based recommender systems and item-based collaborative filtering. It aggregates trusted users' ratings on the item or similar items, to depth k, without adding new relations in the graph. This significantly improves coverage but requires a global knowledge on items ratings. RandomWalk is the pure trust-based version of TrustWalker [6]. It only relies on local knowledge and propagates trust to depth k.

Existing approaches offer recommendation to the cost of new relations between users or global knowledge on users profiles. Trust propagation is not purely local, as it knows every path from a user to another. Therefore privacy can not be assured to the users of the system. That is why we introduce a recommender system propagating locally ratings through social relations. Our approach does not create new relations and shares data (*i.e.* ratings) only between direct friends. As it uses ratings aggregation, a final user does not know where ratings come from, following the privacy concern of our system.

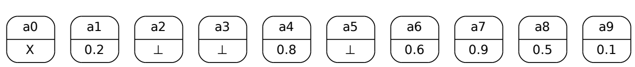
3. DEFINITIONS

In this paper, an actor refers to any social entity in a social network [19]. We use the term "friends" to refer to actors directly connected in the social network. A social relation is an undirected link between two friends in the social network. It means that both friends, and only friends, share data with each other.

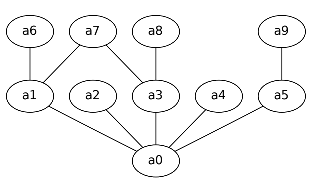
In order to be compatible with trust networks, trust relations can be added on existing social ones. They are weighted and oriented relationships between friends ranging from 0 (no trust) to 1 (full trust). The greater the trust value between actor a and actor f, the more a trusts f's scoring, and then the more f's preferred items are valuable for a. Trust can be explicitly set by actors in their social network or implicitly built from the social relations. By definition, we set trust to 1 between friends if no trust network is used.

Let A be the set of actors and I the set of items. Let F_a be the set of a's friends. The trust relation from actor a to a friend f is noted $t_{a,f}$. This relation is only defined between friends. For each $(a, f) \in A \times F_a$ can be associated zero or one trust $t_{a,f}$. If no trust is associated, a default one is given. Let $\mathcal{T}$ be the set of trust triples $(a, f, t_{a,f})$ with $(a, f) \in A \times F_a$ and $t_{a,f} \in [0, 1]$. This relation is neither symmetric nor transitive.

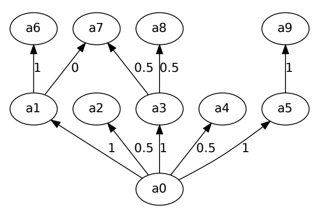
A rating is explicitly set by an actor on an item and a score is a value predicted by the system, or scorer. Scores and ratings are real values between 0 (the actor does not like the item) and 1 (the actor likes the item).
The rating given by actor a to item i is noted $r_{a,i}$. Let $\mathcal{R}$ be the set of ratings triples $(a, i, r_{a,i})$ with $a \in A$, $i \in I$ and $r_{a,i} \in [0, 1]$. For each $(a, i) \in A \times I$ can be associated zero or one rating $r_{a,i}$. Let $A_i = \{a \in A | \exists(a, i, r_{a,i}) \in \mathcal{R}\}$ be the set of all actors who have rated item i. Let $\mathcal{F}_{a,i} = F_a \cap A_i$ denote a's friends who have a rating for item i.



(a) Actors' ratings on item i_0



(b) Social network



(c) Trust relations

Figure 1: Social network, trust and ratings example centered around a_0

We illustrate this approach with a simple, yet adequate, example illustrated in figure 1. Let us consider ten actors $\{a_0, a_1, \cdots, a_9\}$; the objective is to predict a score for item i_0 to a_0, denoted X. For the sake of readability, we only show the social network and trust starting directly or indirectly from our main actor a_0. Figure 1a shows ratings by actors on item i_0. a_1's rating is 0.2, a_2 and a_3 have not rated item i_0 yet, a_4's rating is 0.8, etc. In our motivating example:

- $A = \{a_0, a_1, a_2, a_3, a_4, a_5, a_6, a_7, a_8, a_9\}$ and $I = \{i_0\}$
- $\mathcal{T} = \{(a_0, a_1, 1), (a_0, a_2, 0.5), \cdots, (a_5, a_9, 1)\}$
- $\mathcal{R} = \{(a_1, i_0, 0.2), (a_4, i_0, 0.8), \cdots, (a_9, i_0, 0.1)\}$
- $F_{a_1} = \{a_6, a_7\}$ and $A_{i_0} = \{a_1, a_4, a_6, a_7, a_8, a_9\}$

4. SOCIAL SCORING

We propose to use social scoring to predict missing ratings (in other words, to compute scores) that would have been given by actors to items. We use the actors' social network to propagate scores. Actors manage their own private profiles and share them on demand with their direct friends.

We first present the score propagation in the actors' social network. We then introduce several coefficients aiming at improving recommendation accuracy. Trust allows actors to explicitly weight friends' ratings. Correlation modulates friends' ratings by computing similarity between friends. A default score is introduced to remedy sparsity problems. Confidence on scores offers a way to take into account distance between actors. Confidence is also provided to the final user to indicate the system belief on the recommendation result accuracy. We finally present our scorer CoTCoDepth.

4.1 Score propagation

In this section, we introduce our generic formula called "$k-$depth social scoring" (s_k) that forms the basis of our social scoring. "Asking" a score denotes sending a request from a peer (an actor) to another peer (his/her friend) in order to get a score relative to an item from this friend. A requester is a peer asking for a score, at each propagation. The original requester is the first peer asking for a score, the one who propagates the request.

$k-$depth social scoring gets actor's rating if it exists. If not, it asks the actor's friends to provide their ratings (if any) or to predict their scores, using their friends' ratings and so on, to depth k. If no one returns a score, the final score is unpredicted ($\perp$).

The coefficient $\omega_{a,f}$ between actors a and f weights the mean of friend's scores. It will be specified in the following sections, depending on the scoring. Let $\mathcal{F}^k_{a,i,\omega}$ be the set of a's friends f where $s_k(f, i)$ is defined and $\omega_{a,f}$ is not null.

$$\mathcal{F}^k_{a,i,\omega} = \{f \in F_a | s_k(f, i) \neq \perp \wedge \omega_{a,f} \neq 0\} \quad (3)$$

s_0 is the rating set by the actor. The $k-$depth social scoring is then defined as in eq. 4[3]. This formula is an adaptation to our constraints of the one described in section 2 (eq. 1).

$$s_k(a, i) = \begin{cases} r_{a,i} & \text{if } \exists r_{a,i} \\ \dfrac{\sum_{f \in \mathcal{F}^{k-1}_{a,i,\omega}} \omega_{a,f} \times s_{k-1}(f, i)}{\sum_{f \in \mathcal{F}^{k-1}_{a,i,\omega}} \omega_{a,f}} & \text{if } \nexists r_{a,i} \wedge \mathcal{F}^{k-1}_{a,i,\omega} \neq \emptyset \\ \perp & \text{otherwise} \end{cases} \quad (4)$$

Figure 2 runs the motivating example with $\omega_{a,f} = 1$ and $k = 2$. a_0 asks to all his/her friends their scores for the item. a_1 has already a rating and returns immediately 0.2 without asking a_6's score. a_2, having no rating nor friend, returns

[3]For clarity, we use a simplified version of the formula. The complete version does not ask the requester to limit cycles.

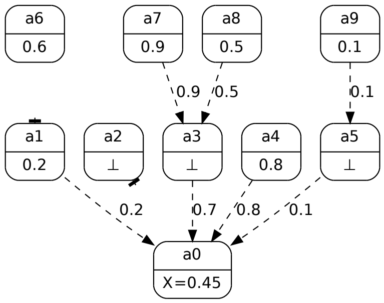
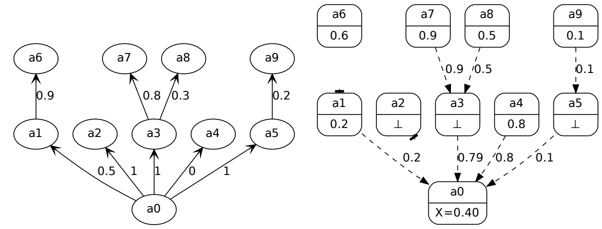


Figure 2: $k-$Depth Social Scoring Example



(a) Similarity network (b) Score propagation

Figure 3: Correlation Example

$\perp$. a_3 has no rating but two friends, a_7 and a_8. The former computes a mean using the two latter ratings and returns it (0.7). a_4's rating is transmitted as previously. a_5 returns a_9's rating. With this $k-$depth social scoring, a_3 and a_5 help a_0 to compute the score through their friends and then participate in the score computation.

The $k-$depth social scoring propagates scores only between immediate friends. Scores are aggregated before being transmitted to the requester, then scores are propagated in the network without interfering with existing relations.

The following sections describe different coefficients ω used in our approach.

4.2 Trust

Trust values are explicitly defined between friends. This allows our approach to be compatible with both trust networks and social networks. If scores are propagated in a trust network or in a social network containing trust values, trust values are considered to weight relations: $\omega_{a,f} = t_{a,f}$ in eq.4. If scores are propagated in a social network without trust value, we set trust to 1 between all friends, cancelling trust coefficient: $\omega_{a,f} = 1$ in eq.4.

Score propagation is the same as in figure 2, except that scores are weighted by trust. In our example, the predicted score for a_0 is then 0.40 since a_0 trusts less a_4 who has a high rating for i_0. Since actors define trust explicitly, this allows them to refine the importance of some friends.

4.3 Correlation

Trust relations are by definition subjective. We use correlation (similarity) between actors to refine this coefficient. Unlike global approaches, the correlation is not computed between all actors to build a global graph, but only between direct friends. The correlation modulates the existing social graph without adding new links. This coefficient is a classic Pearson's correlation coefficient, as described in section 2 (eq. 2), denoted by ρ. To compute $\omega_{a,f} = \rho_{a,f}$ between two friends a and f, only items rated by both friends are used. If the two friends have no item in common, a default value is returned: 0.5[4], the average between no correlation (0) and correlation (1). This allow the system to get recommendation for those friends.

As shown in figure 3, the correlation changes the way actors deal with scores. The similarity varies how one friend infers in the score computation among other friends. a_7 and a_8 share their ratings as before, but since a_3 is more similar to a_7, a_7's score has a higher weight. a_0 does not take into account a_4's score since their share no similarity.

Thanks to the correlation, the scorer takes into account similarity between friends to compute scores, friends with similar tastes are "promoted" during the recommendation.

4.4 Default Score

Due to the usual sparsity of existing dataset, many item scores cannot be computed since no friend has rated these items up to the given depth, which leads to a limited coverage. In order to counter this drawback, one can propagate deeper in the network, implying heavier traffic and lower privacy. [6] returns a similar item's rating, requiring a global knowledge, *c.f.* section 2. In this paper we provide two solutions based on default scores depending on how data can be used by the system. In case of missing rating, a friend may return a default score which is his/her mean rating or the item's mean rating. This approach grandly increases the coverage, providing a score for almost every requester.

The calculation of an item mean rating requires to know all ratings on the item but does not require to know who gave them. It does not respect our local assumption, however it respects our privacy assumptions. It requires an anonymous global knowledge on ratings.

We define $P_{default}$ the probability to return a default score if an actor cannot compute a score. This adds randomness in the recommendation, which is usually considered as a good thing in order to recommend new items to a specific actor [1]. It reduces the computational burden by not always returning a score. Finally, it also reduces noisy recommendations, *i.e.* recommendation not based on a specific item. For that purpose, it must remains low.[5] Therefore, equation 4 becomes:

$$s_k(a,i) = \begin{cases} r_{a,i} & \text{if } \exists r_{a,i} \\ \dfrac{\sum_{f \in \mathcal{F}^{k-1}_{a,i,\omega}} \omega_{a,f} \times s_{k-1}(f,i)}{\sum_{f \in \mathcal{F}^{k-1}_{a,i,\omega}} \omega_{a,f}} & \text{if } \nexists r_{a,i} \wedge \mathcal{F}^{k-1}_{a,i,\omega} \neq \emptyset \\ \mathit{default}(a,i) & \text{otherwise} \end{cases} \tag{5}$$

With $\mathit{default}(a,i)$ defined with the following probabilities:

$$P(\mathit{default}(a,i) = \left\{ \begin{array}{l} \overline{r_a} \\ \overline{r_i} \end{array} \right.) = P_{default} \tag{6}$$
$$P(\mathit{default}(a,i) = \perp) = (1 - P_{default})$$

[4]We want to take into account all actor's friends, with similar ones having more impact than others. This default similarity has been validated empirically.

[5]In our evaluation, we have empirically set $P_{default} = 0.02$.

$P_{default} = 0.02$ means that 2 % of friends who cannot compute a score will return the default score (either the actor mean $\overline{r_a}$ or the item mean $\overline{r_i}$) instead of $\perp$.

4.5 Confidence

In the previous sections, actors are weighted by the trust and/or the similarity but each rating has the same weight. For example, any actor asking only one friend's rating will return this rating with the actor's weight and not the friend's weight. In the previous example, a_5 has only one friend a_9 and since the mean is normalized by the sum of all similarities, the similarity between a_5 and a_9 does not change anything. Then, a_9's rating is as important as a_7's and a_8's gathered through a_3, despite a smaller similarity.

Moreover, there is no knowledge of distance between actors in the social graph. A rating provided by a friend of friend of friend has the same weight of a direct friend's rating if those two friends have the same weight.

Finally, there is no way for a user to distinguish between a rating given by a friend and a score computed by a friend, especially a default score. Or the former should be promoted over the latter since it should be more accurate.

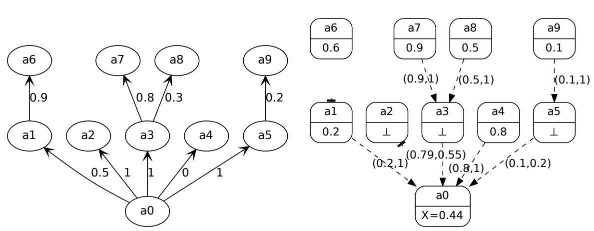
To cope with this issue, we have introduced confidence from actors on scores: $c \in [0, 1]$. This coefficient is associated and transmitted with each score. If actor a rated item i, by definition $c_{a,i} = 1$. If the score is a default score (*c.f.* section 4.4), by definition $c_{a,i} = C_{default}$, which must be low to minimize noisy recommendations[6]. Otherwise, with a given depth k and a given weight ω, the confidence from actor a to i's score is :

$$c_{a,i} = \frac{\sum_{f \in \mathcal{F}^k_{a,i,\omega}} \omega_{a,f} \times c_{f,i}}{|\mathcal{F}^k_{a,i,\omega}|}$$

Confidence is not normalized, except for the original requester, then the further the responder, the less the requester is confident in the response. Without knowing the actual distance of the score source, the confidence decreases at each hop.

Score propagation is the same as previously but confidence is transmitted with scores. Score computation uses only ω, except for the original requester. The score computed by the original requester takes into account this confidence coefficient in the mean computation and normalization. The final coefficient (noted $\omega^{(c)}$), for a given item i, is then:

$$\omega^{(c)}_{a,f} = \omega_{a,f} \times c_{f,i}$$



(a) Similarity network (b) Score propagation

Figure 4: Confidence Example

[6]In our evaluation, we have empirically set $C_{default} = 0.01$.

In figure 4, the coefficient ω is based on similarity. a_5 is not confident on a_9's score since they share little similarity (0.2). This confidence is propagated to a_0. Then a_0's confidence on a_5's score remains low, which means that this score has less weight in the final mean. The confidence returned to a_0 is then $\frac{0.5\times1+1\times0.55+0\times1+1\times0.2}{0.5+1+0+1} = 0.5$.

The confidence reduces the weight of scores propagated from friends of friends. The more friends are related (regarding trust or similarity), the higher their confidence. In addition, confidence is provided to the final user as an accuracy indicator on the recommendation.

4.6 CoTCoDepth Social Scoring

Our final scorer for social recommendation is the aggregation of all previous definitions: the Confident Trust Correlative $k-$Depth Social Scorer, CoTCoDepth Scorer for short (or $s^*_k(a, i)$ in eq.7).

CoTCoD1 propagates to $k = 1$ (immediate friends), CoTCoD2 to $k = 2$ (friends of friends)... CoTCoDk denotes CoTCoDepth without default score ($P_{default} = 0$); CoTCoDk_a denotes CoTCoDepth with actor mean rating as default score; CoTCoDk_i denotes CoTCoDepth with item mean rating as default score; CoTCoDk_{ia} denotes CoTCoDepth with item mean rating as default score or actor mean rating if the item mean is not computable ($P_{default} = 0.02$, *c.f.* eq.6).

$$s^*_k(a, i) = \begin{cases} r_{a,i} & \text{if } \exists r_{a,i} \\ \dfrac{\sum_{f \in \mathcal{F}^{k-1}_{a,i,\omega}} \omega_{a,f} \times s^*_{k-1}(f, i)}{\sum_{f \in \mathcal{F}^{k-1}_{a,i,\omega}} \omega_{a,f}} & \text{if } \nexists r_{a,i} \wedge \mathcal{F}^{k-1}_{a,i,\omega} \neq \emptyset \\ \mathit{default}(a, i) & \text{otherwise} \end{cases} \tag{7}$$

For a given item i, the weight ω for score propagation is defined in eq.8 and the weight $\omega^{(c)}$ for the original requester is defined in eq.9.

$$\omega_{a,f} = t_{a,f} \times \rho_{a,f} \tag{8}$$

$$\omega^{(c)}_{a,f} = t_{a,f} \times \rho_{a,f} \times c_{f,i} \tag{9}$$

In our motivating example, a_0 is then recommended item i_0 with a score of 0.36 due to a lower a_3's confidence.

Our scorer uses a social network to recommend items. It is based on local knowledge such as the immediate social relations and local trust. We have then introduced firstly a local similarity based on friendship relations in order to promote friends with the same tastes and secondly a confidence on the scoring itself. In order to prevent the usually sparsity problem in trust-based recommendation, we propose a default score returned by actors. Our scorer presented here propagates scores (and confidence) locally in the social network without creating new relations.

In the next section, we evaluate our system and compare it with existing trust-based or collaborative filtering recommender systems. We show that our local approach provides recommendations that are as accurate as the ones given by existing systems and provides more accurate recommendation for cold start users.

5. EVALUATION

In order to evaluate the social approach, we have implemented our CoTCoDepth Scorer and run it with $k \in$

$\{1, 2, 3\}$, without default score ($P_{default} = 0$) and with actor mean or item mean as default score ($P_{default} = 0.02$). We have then observed the influence of k and of the number of friends (*i.e.* connectivity degree) on coverage and precision. We have also compared our scorers with the approaches described in section 2: GlobalCF, ItemBased, MoleTrust, RandomWalk, and TrustWalker. In order to provide a fair comparison, we have used $max - depth = 3$ for MoleTrust, RandomWalk and TrustWalker. The evaluation dataset is presented in section 5.1. We explain our evaluation campaign in section 5.2. We present and discuss our results in section 5.3.

5.1 Epinions dataset

[16] proposes a dataset derived from Epinions[7]. This dataset contains items rated by actors and oriented links between actors with trust values. The dataset is very sparse (only 0.01 % ratings): actors rated 12 items and have 11 trustees in average.

The ratings distribution in this dataset follows the power law: few actors rated a lot of items but most of actors rated only a few items. We call "cold start users" actors with less than 5 ratings [9], they are the most difficult to recommend and are majority: more than 50 % in the dataset. More information on the dataset can be read in [16].

We have have also split the dataset into views to observe the influence of the connectivity degree in the social network. Actors with at least one friend but with less than five friends are "weakly connected" (47 % of actors with 22 % of ratings), actors with five to nine friends are "fairly connected" (11 % of actors with 12 % of ratings) and actors with ten friends or more are "highly connected" (18 % of actors with 57 % of ratings). 23 % of actors with only 8 % of ratings do not have any relation.

5.2 Evaluation campaign

In order to evaluate our algorithms, we have used the classical "leave one out" method. This method consists in taking the whole dataset, removing only one rating and then trying to predict it using all others ratings. There is no random selection so this campaign is reproducible. This is done for all ratings in the dataset and results are aggregated to return statistical metrics on the algorithms. Since similarity is computed using ratings, it must be recomputed for each removed rating in the dataset. Therefore, once a rating is removed, similarity coefficients associated with the actor and with the item are recomputed.

The dataset contains 104k items including 54k items with only one rating, which represents 10 % of all ratings. Classical collaborative filtering approaches, including trust-based ones, cannot predict any score for those items since no other actor has rated the item and once the rating has been removed, no similarity is computable. Those approaches can reach about 90 % coverage at maximum.

The statistical metrics are the Coverage, the Root Mean Square Error (RMSE) and the F-Measure. The coverage is the proportion of predicted ratings regarding all ratings to predict. It does not indicate if the ratings were correctly predicted but shows how many predictions an algorithm can fulfill. The RMSE represents the average error of the prediction. It is basically the error standard deviation without mean. The lower the RMSE, the more accurate the prediction. However it is only computable with predicted scores. Therefore we use the *F-Measure* F_1 which combines the two metrics above [6][8]:

$$Precision = 1 - \frac{RMSE}{range} \tag{10}$$

$$F_1 = \frac{2 \times Precision \times Coverage}{Precision + Coverage} \tag{11}$$

5.3 Results

5.3.1 Influence of k and connectivity degree

We have run CoTCoD1, CoTCoD2 and CoTCoD3 with $P_{default} = 0$; CoTCoD3$_a$, CoTCoD3$_i$ and CoTCoD3$_{ia}$ with $P_{default} = 0.02$ on Epinions.

Coverage.

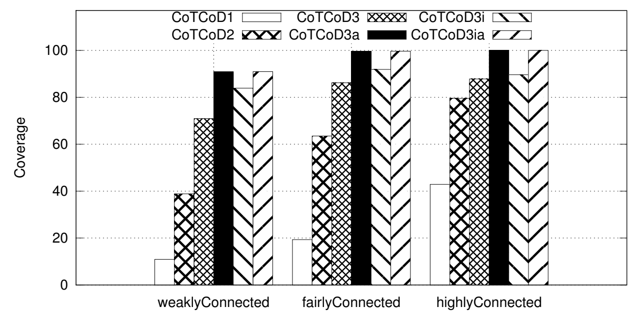


Figure 5: Influence of connectivity degree on coverage

Figure 5 shows that increasing either k or the number of friends improves coverage, as one would expect. Obviously, immediate vicinity is not sufficient to predict scores. Propagating scores in the social network offer an efficient solution to improve coverage.

The *default score* enhances the coverage as expected. Returning an actor mean ratings is even more covering than returning an item mean ratings. This is due to the 10 % unpredictable ratings (*c.f.* section 5.2): since the only available rating for the item is removed, no mean is computable. But most actors still have other ratings therefore then can return their mean, enhancing the coverage. With this approach, nearly 100 % ratings are predicted for fairly and highly connected actors.

Using a sparse trust network implies that we need to propagate further in the graph in order to cover enough ratings. We can see that increasing k improves coverage, since it increases the number of actors involved in the recommendation. To avoid the small world effect [12], we did not propagate scores further than $k = 3$. However, propagating to $k = 3$ with $P_{default} = 0$ covers more than 86 % ratings made by actors trusting more than four actors, despite using a sparse trust network. Moreover, introducing a low probability to return a default score ($P_{default} = 0.02$) increases the coverage to reach more than 90 % ratings, all actors included, and almost 100 % ratings from actors trusting more than four actors.

[7] www.epinions.com

[8] *range* is the rating interval, *i.e.* the largest possible error. In Epinions, *range* $= 4$

Precision.

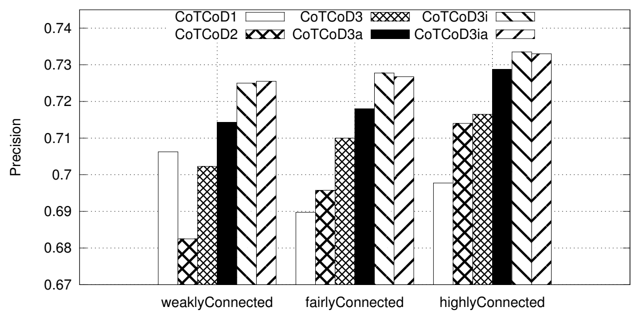


Figure 6: Influence of connectivity degree on precision

Figure 6 shows that increasing the number of friends improves the precision (*c.f.* eq.10), the more an actor is connected the more accurate the recommendations, as one would expect. Being highly connected implies having more recommendations, therefore the aggregation returns a more reliable answer. The CoTCoD1 precision is not significant since the coverage is too low, specially for weakly connected users.

However propagating further in the network also improves the accuracy, which was less expected. Immediate friends should be more likely to provide relevant recommendations. Here again, a deeper propagation leads to more scores, which explains a better accuracy.

We saw in the previous section that the item mean ratings as default score provide a lower coverage than the actor mean ratings. However the precision is higher, which is legitimate since it uses ratings on the requested item. Therefore using anonymous aggregated data enhances significantly the precision of our approach. Moreover, returning the item mean ratings if available or the actor mean ratings otherwise (CoTCoD3$_{ia}$) gets the coverage of CoTCoD$_a$ with a precision as high than CoTCoD$_i$: it is clearly the best approach.

5.3.2 *Comparison with existing approaches*

In order to compare our approach with others, we have implemented GlobalCF, ItemBased, MoleTrust3, RandomWalk3 and TrustWalker3, using a trust propagation of 3. We want to minimize the trust propagation since we are in decentralized architecture and this propagation is costly, 3 is the best trade-off between accuracy and coverage. We have implemented our approach both without (CoTCoD3) and with (CoTCoD3$_a$ and CoTCoD3$_{ia}$) default scoring in order to provide honest comparison. Epinions is the most used dataset for trust-based recommendation evaluation, it was used with all approaches. Results on Epinions dataset are summarized in table 1 for all actors and in table 2 for cold start users.

Method	RMSE	Cov.	F_1	Knowledge
GlobalCF	1.138	69.09	0.703	global
ItemBased	1.364	65.64	0.658	global
MoleTrust3	1.100	77.25	0.748	extended-local
RandomWalk3	1.271	53.44	0.599	local
TrustWalker3	1.092	85.99	0.788	local + global
CoTCoD3	1.150	77.25	0.741	local
CoTCoD3$_a$	1.105	90.50	0.804	local
CoTCoD3$_{ia}$	1.078	90.56	0.809	local + anonymous

Table 1: Results for all actors on Epinions

Table 1 indicates the RMSE, the coverage and the f-measure of all actors ratings prediction. Regarding only pure local approaches, CoTCoD3$_a$ provides the best results both in term of precision and coverage. RandomWalk is not effective enough with a low propagation depth of 3. That kind of approach usually propagates up to depth 6. MoleTrust3 offers the same coverage than CoTCoD3 with a lower error but it relies on extended-local knowledge. Moreover CoTCoD3$_a$ still has a higher f-measure than MoleTrust3. CoTCoD3$_a$ helps improve coverage but also accuracy. As stated by [9], the Epinions dataset contains mostly 5 as rating value which explains why returning an average rating improve accuracy. Regarding classical collaborative filtering approaches, both GlobalCF and ItemBased are outperformed by the trust-based approaches, since similarity is seldom computable with sparse datasets such as Epinions [6, 9]. TrustWalker[9], based on local and global knowledge, has one of the lowest error with 1.092. However it has a lower coverage than CoTCoD3$_a$, based on local knowledge, and CoTCoD3$_{ia}$, based on local or anonymous knowledge. CoTCoD3$_{ia}$ is the best system in our evaluation with the lower error and the highest coverage, therefore the highest f-measure. The default score strategy alleviate coverage limitation and is compatible with our architecture and privacy assumptions. It provides scores to items that cannot be recommended with classical collaborative filtering and trust-based approaches. Our approach is thus adapted for sparse networks such as Epinions, where actors need to explicitly express that they trust other people in order to connect with them, without propagating deeply in the network and only based on local or local and anonymous information.

Method	RMSE	Cov.	F_1	Knowledge
GlobalCF	1.248	15.13	0.248	global
ItemBased	1.639	21.49	0.315	global
MoleTrust3	1.167	50.51	0.590	extended-local
RandomWalk3	1.288	37.74	0.485	local
TrustWalker3	1.287	67.50	0.677	local + global
CoTCoD3	1.196	50.51	0.587	local
CoTCoD3$_a$	1.145	65.05	0.681	local
CoTCoD3$_{ia}$	1.103	65.48	0.688	local + anonymous

Table 2: Results for cold start users on Epinions

Table 2 compares those approaches regarding only cold start users. With those particular actors, who don't have enough ratings to perform well with classical collaborative filtering, CoTCoDepth outperforms all other local and global algorithms in terms of precision and coverage, except for TrustWalker3 which covers 2 % more ratings. RandomWalk3 and the global approaches do not cover enough ratings to be significant. TrustWalker3 suffers a high error. Here again CoTCoD3$_{ia}$ provides the best precision and coverage for cold start users. The pure local algorithm CoTCoD3$_a$ provides the second highest f-measure, just below CoTCoD3$_{ia}$.

6. CONCLUSION

This paper presents CoTCoDepth, a recommender system using limited knowledge and based on explicitly user-

[9]The accuracy of our TrustWalker implementation is coherent with [6], while coverage is 10 % lower. We do not understand the coverage indicated in [6] since, as stated in section 5.2, 90 % should be the maximum coverage with classical collaborative filtering approaches using a leave one out evaluation campaign and the Epinions dataset.

defined social relations. It is deployed on the users' devices, do not need heavy off-line preprocessing, and its complexity depends on the number of friends of each user. Scores are propagated in a P2P manner, without generating new links in the social graph. Each peer has knowledge of the local user's friends' scores, but cannot access friends-of-friends' data.

CoTCoDepth aggregates data from users' friends, computes local similarity between them and returns confidence on scores. It uses three default scoring strategies to enhance coverage. The first returns actors mean rating, it is purely local and offers the best coverage. The second returns items mean rating, it is based on anonymous ratings aggregation and offers the best precision. The third returns items mean rating if available or actors mean ratings otherwise, offering both the best coverage and precision.

We have evaluated our approach with the Epinions dataset and compared it with classical collaborative filtering (GlobalCF and ItemBased) and trust-based approaches (MoleTrust, RandomWalk and TrustWalker) regarding three metrics: coverage, predictive accuracy and f-measure. The results show that our approach has the highest f-measure for all users and cold start users. This is mainly due to a high coverage, thanks to our default score strategies. Our CoTCoDepth scorers have the best accuracy for cold start users (i.e. users that have rated few item) and a high coverage even for users that have few (less than five) connections with others.

The pure local versions of CoTCoDepth are deployable on peer to peer architectures. We have already implemented them in a P2P simulator: PeerSim [13]. Our future work will focus on the evaluation of a decentralized recommender system based on CoTCoDepth. We will observe the network load implied by our approach and simulate some P2P and decentralized constraints existing in such architectures (disconnection, dynamism, timeout, etc.). In order to avoid network flooding and overload, we will experiment heuristics to select a part of all available friends and measure consequences on coverage and accuracy.

To go further, our approach could be improved by using content-based recommendation algorithms, *e.g.* through locally computed items similarities. Recommendation offers interesting research problems as well for pervasive environments, such as recommending services or applications to users depending on the context.

7. 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*, 17(6):734–749, 2005.

[2] J. Breese, D. Heckerman, C. Kadie, and Others. Empirical analysis of predictive algorithms for collaborative filtering. In *Proceedings of the 14th conference on Uncertainty in Artificial Intelligence*, pages 43–52. Madison: Morgan Kaufmann, 1998.

[3] J. Golbeck. *Computing and applying trust in web-based social networks.* PhD thesis, University of Maryland at College Park, 2005.

[4] C. Hang, Y. Wang, and M. Singh. Operators for propagating trust and their evaluation in social networks. In *Proceedings of The 8th International Conference on Autonomous Agents and Multiagent Systems-Volume 2*, pages 1025–1032. International Foundation for Autonomous Agents and Multiagent Systems, 2009.

[5] O. Hasan. *Privacy Preserving Reputation Systems for Decentralized Environments.* PhD thesis, Institut National des Sciences Appliquées de Lyon, 2010.

[6] M. Jamali and M. Ester. TrustWalker: a random walk model for combining trust-based and item-based recommendation. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 397–406. ACM, 2009.

[7] D. Lee and P. Brusilovsky. Does Trust Influence Information Similarity? In *Proceedings of Workshop on Recommender Systems & the Social Web, the 3rd ACM International Conference on Recommender Systems*, pages 3–6. Citeseer, 2009.

[8] H. Ma, I. King, and M. Lyu. Learning to recommend with social trust ensemble. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, pages 203–210, New York, New York, USA, 2009. ACM.

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

[10] D. McKnight and N. Chervany. The meanings of trust. Technical Report 612, University of Minnesota, Carlson School of Management, 1996.

[11] B. Mehta, T. Hofmann, and W. Nejdl. Robust collaborative filtering. In *Proceedings of the 2007 ACM conference on Recommender systems - RecSys '07*, page 49, New York, USA, 2007. ACM Press.

[12] S. Milgram. The small world problem. *Psychology today*, 2(1):60–67, 1967.

[13] A. Montresor and M. Jelasity. PeerSim: A scalable P2P simulator. In *Proc. of the 9th Int. Conference on Peer-to-Peer (P2P'09)*, pages 99–100. IEEE, Sept. 2009.

[14] J. O'Donovan and B. Smyth. Trust in recommender systems. In *Proceedings of the 10th international conference on Intelligent user interfaces*, pages 167–174, New York, USA, 2005. ACM.

[15] G. Pitsilis and S. Knapskog. Social Trust as a solution to address sparsity-inherent problems of Recommender systems. *Recommender Systems & the Social Web*, 826(October):33–40, 2009.

[16] M. Richardson and P. Domingos. Mining knowledge-sharing sites for viral marketing. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 61–70, New York, New York, USA, 2002. ACM.

[17] J. Schafer, D. Frankowski, J. Herlocker, and S. Sen. Collaborative filtering recommender systems. In *The adaptive web*, pages 291–324. Springer-Verlag, 2007.

[18] J. B. Schafer, J. Konstan, and J. Riedi. Recommender systems in e-commerce. *Proceedings of the 1st ACM conference on Electronic commerce - EC '99*, pages 158–166, 1999.

[19] S. Wasserman and K. Faust. *Social network analysis: Methods and applications.* Cambridge Univ Pr, 1994.