ProCF: Probabilistic Collaborative Filtering for Reciprocal Recommendation

Xiongcai Cai, Michael Bain, Alfred Krzywicki, Wayne Wobcke, Yang Sok Kim, Paul Compton, and Ashesh Mahidadia

School of Computer Science and Engineering,
The University of New South Wales, Sydney NSW 2052, Australia
{xcai,mike,alfredk,wobcke,yskim,compton,ashesh}@cse.unsw.edu.au

Abstract. Similarity in people to people (P2P) recommendation in social networks is not symmetric, where both entities of a relationship are involved in the reciprocal process of determining the success of the relationship. The widely used memory-based collaborative filtering (CF) has advantages of effectiveness and efficiency in traditional item to people recommendation. However, the critical step of computation of similarity between the subjects or objects of recommendation in memory-based CF is typically based on a heuristically symmetric relationship, which may be flawed in P2P recommendation. In this paper, we show that memory-based CF can be significantly improved by using a novel asymmetric model of similarity that considers the probabilities of both positive and negative behaviours, for example, in accepting or rejecting a recommended relationship. We present also a unified model of the fundamental principles of collaborative recommender systems that subsumes both user-based and item-based CF. Our experiments evaluate the proposed approach in P2P recommendation in the real world online dating application, showing significantly improved performance over traditional memory-based methods.

Keywords: Social Network Mining, Recommender Systems.

1 Introduction

Memory-based collaborative filtering (CF) is the basis of many commercial recommender systems, of which Amazon's [13] item-based approach is probably the best known. In this work we present a unified framework incorporating both item-based and user-based CF and within it develop a novel probabilistic method of similarity that overcomes some of the limitations of previous approaches.

Conventional *recommender systems* attempt to discover user preferences over items by modelling the relation between users and items. The aim is to recommend items that match the *taste* (likes or dislikes) of users in order to assist the active user, i.e., the user who will receive recommendations, to select items from an overwhelming set of choices. It is used to 1) predict whether a particular user will like a particular item (a prediction problem), or 2) identify a set of N items that will be of interest to a certain user (a Top-N recommendation problem). Recently, recommender systems have also been extended to *people to people (P2P)* recommendation to model the relation between the active user and other users by finding user preferences over other users.

Assuming that users with similar tastes would rate items (or other users) similarly, memory-based collaborative filtering (CF) methods recommend items based on heuristic aggregated user preferences for items, independent of the availability of item descriptions. In this paper we formalise memory-based CFs in a uniform way that allows the derivation of a *probabilistic* method, ProCF, that is shown to improve performance in a P2P recommendation application.

Section 2 discusses related work. Section 3 defines the problems. Section 4 develops a probabilistic approach for both recommendation and ranking. Experimental evaluation is in Section 5 and we conclude in Section 6.

2 Related Work

CF algorithms fall into two categories: model-based and memory-based approaches. Model-based CF [1,2,10,16] uses the collection of ratings to learn a model, which is then used to make rating predictions. Although model-based methods have reported higher accuracy of recommendation than memory-based approaches, there are some limitations. These methods are computationally expensive since they usually require all users and items to be used in creating models, and the number of users and items is typically large. Memory-based CF is popular in many commercial recommender systems, being effective and easy to implement. Memory-based approaches [2,11,13,18] make rating predictions based on the entire set or a sample of items previously rated by users. The unknown rating value $r_{c,s}$ of the active user c for an item s is typically computed as an aggregate of the ratings of users similar to c for the same item s. This aggregate can be an average or a weighted sum, where the weight is a distance that measures the similarity between users c_1 and c_2. By using similarity as a weight, more similar users make a greater contribution to a predicted rating.

In memory-based CF, similarity computation between items or users is essential. The definition of similarity measure varies depending on the recommendation application. Often the similarity between two users is based on the ratings of items both users have rated. Two of the most popular approaches are correlation [11,18] and cosine-based [2,17]. Extensions to these include default voting, inverse user frequency, case amplification, and weighted-majority prediction [2,7]. Usually these use heuristics to model the weights and are not able to handle the different rating scales of different users. Solutions to this problem include the adjusted weighted sum and preference-based filtering [14], which focuses on predicting the relative preferences of users instead of absolute rating values.

Memory-based probabilistic CF is an alternative. Yu *et al.* [20] use a mixture model for user preferences. Deshpande and Karypis [8] proposed conditional probability based similarity in item-based CF. These models only consider common purchase information, which causes the problem that frequently purchased items tend to have high conditional probabilities, leading to reduced diversity in recommendation [9]. Adding a scaling parameter to control for the effect of popular items in the model may help, but finding a suitable parameter value becomes challenging. Also, these methods are uni-directional, relying only on users' *taste*, so they are not applicable to P2P recommendation, which is reciprocal.

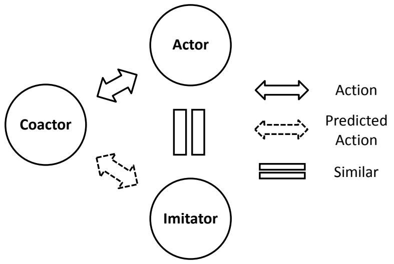
People recommenders deal with the problem of finding meaningful relationships among people or organisations. In online social networks, relationships can be friends [19] e.g., on Facebook, professional contacts [3] e.g., on LinkedIn, online dating [5,12], or jobs on employment websites [15]. The nature of these domains makes P2P recommender systems significantly different from traditional item to people (I2P) recommenders. The basic difference in the people recommender domain is the characteristic of *reciprocal* relationships.

3 Problem Statement

Recommender systems can be classified into two general classes: classical item to people recommender systems (I2PRec) and people to people recommender systems (P2PRec). In classical I2PRec, there are two types of entities, buyers (e.g., customers) and items (e.g., books, movies, songs). In recent P2PRec [5], there only exists a uniform entity type: users (e.g., online dating service subscribers, job seekers and employers). To distinguish the different roles in a recommendation, we use *subject*, $S = \{s_1, ..., s_{|S|}\}$, to refer to the recommendation recipient (e.g., customers in I2PRec and active partner seekers for P2PRec) and *object*, $O = \{o_1, ..., o_{|O|}\}$, to refer to the recommendation candidate (e.g., books in I2PRec and partner seekers in P2PRec). Recommender systems using CF methods rely on collaborative information. There are several types of collaborative information. One important distinction is between *explicit* (i.e., ratings, up and down votes) and *implicit* (i.e., clicks, purchases, contacts, replies) expressions of user preferences. Depending on the type of system, implicit information may be positive-only, i.e., no recorded negative preference observations, or positive-and-negative, i.e., both positive and negative preference observations are available.

In I2PRec, collaborative information used in traditional CF is merely based on the behaviours of subjects, i.e., the preference of buyers determines the transactions that represent the collaborative information. However, in P2PRec, collaborative information usually depends on behaviours of *both* subject and object, since the relationship between the subject and the object can only be established when both parties agree on it, denoted by a successful interaction (i.e., the subject makes contact to express interest and gets positive feedback from the object). We use so^+ to refer to this, representing the establishment of a successful interaction. Similarly, so^- refers to an unsuccessful interaction. This requires P2PRec to consider collaborative information based on behaviours of both subject and object rather than only those of subject, which consequently prevents traditional I2PRec from solving the P2P recommendation problem [4].

The task of P2P recommendation from implicit, positive and negative, preferences is to rank the objects from a candidate set O^c for a subject according to the probability of establishing a relationship between them based on the collaborative information. This task is related to, but distinct from, rating prediction,



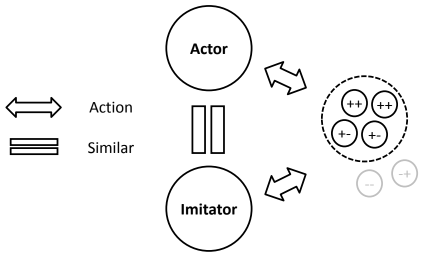


Fig. 1. A general framework for CF. In user-based CF (UBCF) (resp. item-based CF (IBCF)), Coactor is the candidate entity (active entity), Actor the entity similar to the active entity (candidate entity), Imitator the active entity (candidate entity). (Section 3)

Fig. 2. Construction of probabilistic similarity. Actor and Imitator both interact with Coactor entities, with four possible outcomes, e.g., the label (++) means that both interactions have a positive outcome.

where the task is to predict how much a subject will like an object. Therefore, a ranked list of candidate objects for each subject, rather than an explicit rating, is the output of a typical P2P recommender.

Similarity. Memory-based CF approaches to recommender systems differ according to the method used to compute the similarity between the various subjects/objects, which consequently determines the overall performance of the recommender. In general, the similarity between two entities should be high if they have interacted with a large set of others in common and low if few in common. In this case, the similarity between two entities is *symmetric* since the common set is shared with both entities and no other information is considered. But this may not be sufficient, since other subjects/objects that were not interacted with in common by the similar pairs, but only by one in the pair, can have impact on the similarity and thus the recommendation. When using information from such entities not interacted with in common by the similar pairs, similarity becomes naturally *asymmetric*, since each entity in the pair can have a different set contacted by it alone. For example, in P2P recommendation, Bob may prefer Alice since she is younger than 30 years old and likes music, while Charlie could also prefer Alice, but only because she has age less than 30. Thus, Bob is similar to Charlie since they have a common preference and vice versa. In symmetric similarity, similarity to each other is the same since they prefer the same woman. However, Diana, who is 20 years old but hates music could be preferred by Charlie but not by Bob, since Bob prefers women who like music. This shows that Bob and Charlie's preferences are not the same, and moreover, that Bob's similarity to Charlie is different from Charlie's to Bob, and thus *asymmetric*.

Similarity can be represented by similar pairs. There are two roles within a similar pair: *actor* and *imitator*, as illustrated in Fig. 1. To define those two roles, we first define the *coactor*:

Definition 1. *A* coactor *is the entity who interacts with both entities in a similar pair, i.e., either the candidate of recommendation in user-based CF or the active entity in item-based CF.*

In the example, Bob is similar to Charlie. If Emma is preferred by Bob and likes music, we can recommend Emma to Charlie by user-based CF. Or if Emma prefers Bob, we can recommend Charlie to Emma by item-based CF. In both cases, we call Emma a *coactor*, the person who either receives recommendations or is recommended. The two roles within a similar pair are defined as:

Definition 2. *An* actor *is the entity within a similar pair that provides collaborative information, i.e., the entity who actually interacted with the coactor.*

Bob is a *actor* in our example since he interacted with the coactor Emma.

Definition 3. *An* imitator *is the other entity in the similar pair with similar behaviour to the actor, i.e., either the active entity in user-based CF or candidate in item-based CF.*

Charlie is an *imitator* in our example for his similarity to the actor Bob. Given the above definitions, we formally define symmetric and asymmetric similarity:

Definition 4. *A* symmetric similarity *is a measurement of the amount of commonality in the behaviour of an actor and its imitator.*

Definition 5. *An* asymmetric similarity *is a measurement of the* combined *(e.g., summed) amount of commonality and difference in the behaviour of an actor and its imitator.*

These definitions subsume both user-based and item-based CF into a novel unified model, which reveals the fundamental principles that drive the formation of collaborative recommender systems by discovering the relation among the entities involved in the collaborative recommendation (Fig. 1).

4 Methods

In this section, we describe the probabilistic CF framework and apply it to the task of P2P recommendation from implicit, positive and negative, feedback.

4.1 Conditional Probabilistic Similarity

We define probabilistic similarity in the context of item-based CF as the probability that an action succeeds given that the subject of recommendation had a positive interaction with one of the similar objects and the subject attempted the same action with another similar object. This definition is sufficiently general to cover both item-based and user-based CF (Fig. 2). It also covers people to people recommendation, in which there are two types of actions: acceptance, i.e., positive feedback, and rejection, i.e., negative feedback. Depending on the type of action we can then define the probability of acceptance or rejection:

Definition 6. *The acceptance probability $P(i|j)$ is the probability a positive action will occur with entity pair i given a positive action occurred with pair j:*

$$P(ic^+|ac^+) = P(ic^+|ac^+, ic) = \frac{P(ac^+, ic^+)}{P(ac^+, ic)} = \frac{|ae^+ \cap ie^+|_{e\in E}}{|ae^+ \cap ie|_{e\in E}} \tag{1}$$

where P is the probability, E the set of entities in the training set, i the imitator in Definition 3, c the coactor in Definition 1 and a the actor in Definition 2.

Definition 7. *The rejection probability $P(x|y)$ is the probability a negative action will occur with entity pair x given a positive action occurred with pair y:*

$$P(ic^-|ac^+) = \frac{|ae^+ \cap ie^-|_{e\in E}}{|ae^+ \cap ie|_{e\in E}} \tag{2}$$

Note that the conditional probability-based similarity of [8] occurs as a special case of this framework when there is only a single type of action. For example, in item-based CF the similarity between two items is the conditional probability that one item will be purchased given that the other has been purchased. This is equivalent to $P(ic|ac)$, which is obtained by dropping the sign of the interactions and simplifying in either of Definitions 1 or 2, when they become identical.

To compute the acceptance and rejection probability, as shown in Fig. 2, we first find the number (C) of common entities each of which has a positive action with the actor, and either a positive or negative action with the imitator. Here the common entities with negative actions with respect to the actor ($-+$ or $--$), shown greyed-out in Fig. 2, are not taken into account. Secondly, we count how many of those have positive actions in common with the imitator (A), and how many of them have negative actions with the imitator (B). Then, the acceptance probability is (A) divided by (C) and the rejection probability is (B) divided by (C). For the example in Fig. 2, the number of such common entities (in the dashed circle) is 4. Out of these the number of entities having positive actions with the imitator ($++$) is 2 and those having negative actions ($+-$) is 2. Therefore, the acceptance probability is $2/4 = 0.5$ and the rejection probability is $2/4 = 0.5$. In this case the actor and imitator as similar entities have equivalent acceptance and rejection probability with respect to the coactors.

4.2 Probability Residue

We consider both acceptance probability and rejection probability in estimating the impact of the preferences of similar entities. If we have a similar pair for which the acceptance probability (P^+, for short) is greater than the rejection probability (P^-), we would naturally be inclined to decide that the similar pair will contribute to the acceptance of the recommendation of a similar entity. Conversely, if P^- is greater than P^+, we would be inclined to consider rejection. To justify this decision procedure, we can calculate the probability of error. Whenever we observe a particular similar pair i, the probability of an error is:

$$P(error|i) = \begin{cases} P^- \text{ if we decide acceptance} \\ P^+ \text{ if we decide rejection} \end{cases} \tag{3}$$

Given i we can minimise the probability of error by deciding acceptance if $P^+ > P^-$ and rejection otherwise. Thus, the *probability residue* is the contribution of each similar entity to the ranking by minimising the decision error:

Definition 8. *The probability residue is the difference between the acceptance probability and rejection probability:*

$$\omega_{ai} = P(ic^+|ac^+) - P(ic^-|ac^+) = P^+ - P^- \tag{4}$$

The probability residue reflects the balance of the acceptance and rejection probability of an interaction between a pair of entities. It measures the degree to which the interaction departs from random, i.e., increases the possibility of either success or failure. Thus, a similar entity will contribute towards a positive rating and thus success if the probability residue $\omega_{ai} > 0$, contribute towards a negative rating and thus failure if the probability residue $\omega_{ai} < 0$, or not contribute to recommendation at all if the probability residue $\omega_{ai} = 0$.

For the example in Fig. 2 the probability residue is 0 and thus will be ignored in recommendation, since the actor as a similar entity to the imitator, as an active entity or candidate, has equivalent acceptance and rejection probability.

4.3 Rating

Rating of candidates as active entities is then based on probability residues. It is the sum of probability residues of all the entities similar to the current active entity corresponding to a candidate:

$$r = \sum_{k \in S} \omega_k \tag{5}$$

where S is the set of similar entities and ω_k the probability residue of Definition 8.

Theorem 1. *If ω is a probability residue and $|S|$ is the number of similar entities, then the candidate rating function r of Equation 5 is a non-monotonically increasing function on the number of similar entities $|S|$.*

Proof. Since ω_i could be negative, for all $|S_1|$ and $|S_2|$ such that $|S_1| \leq |S_2|$, one could have $r(|S_1|) \geq r(|S_2|)$ according to Equation 5. Thus function r does not preserve the ordering and is non-monotonic.

Notice that this non-monotonic characteristic of the rating function is desired in recommender systems. In conventional recommender systems [11,13,8], the rating functions are usually monotonically increasing on the number of similar entities. This is a problem since it will cause popular (i.e., preferred by a large number of entities) or active (i.e., preferring a large number of entities) entities to have higher rating than others since popular or active entities have usually more similar entities than other entities.

More specifically, popular objects are preferred by a large number of subjects and thus have more chance to be co-preferred with other objects, which makes popular objects have more similar objects than non-popular ones. Similarly, active subjects prefer a large number of objects and thus have more chance to co-prefer with other subjects, which makes active subjects have more similar subjects than non-active ones. However, since the rating function defined in Equation 5 is non-monotonic it is *not* necessarily increasing for popular or active entities. Therefore, by using the probability residue defined above, we are able to avoid the common problem of favouring popular entities in recommendation, and therefore increase the diversity as shown in Section 5. Promoting novel recommendation generates global diversity and improves user experience [6].

A ranked candidate list is then generated by descending sort of all candidates on rating. For tied ratings, we have two steps: (i) favour the candidate with a greater total number of interactions used in calculating similarity; and if this is also tied (ii) favour the candidate with more contributed similar pairs. In these rare cases the increased support means favouring more reliable ratings.

4.4 Summary of the ProCF Algorithm

The proposed framework is realised in the ProCF algorithm. It constructs a similarity table and then generates a recommended candidate list for each subject. Specifically, to construct the similarity table, ProCF collects all similar pairs, each of which has at least one common coactor, and then assigns each pair a probability residue value according to Equation 4. To generate recommendations for an active entity, ProCF finds each imitator in the similarity table for which all pairs of their corresponding actors were interacted with by the active entity. It then computes a rating for the imitator according to Equation 5 and adds the imitator to the recommendation list.

The complexity of ProCF is approximately $O(N)$, and $O(N^2)$ in the worst case, where N is the number of entities in the training set, since it examines N entities and up to $N-1$ other entities for each entity. However, because the average entity interaction vector is extremely sparse, the performance of the algorithm tends to be closer to $O(N)$ in practice. Scanning every entity is approximately $O(N)$ rather than $O(N^2)$ because almost all entity interaction vectors contain only a small number of interactions with other entities. Although there are a few entities who interact with a significant percentage of all other entities, each still only requires $O(N)$ processing time.

5 Experiments

In these experiments, we evaluate the proposed approach on people to people recommendation (Top-N recommendation) in a demanding real-world social network data set. We compare the proposed probabilistic CF method to several conventional recommendation strategies based on a set of common evaluation metrics. Owing to space restrictions, we can only summarize our results, which will be detailed in an extended version of the paper.

Datasets. Data was collected from a commercial online dating site. In online dating, people are looking for potential partners. A user contacts people they like by sending messages. Receivers of messages then have options to reply, positively if they like the sender, negatively if they do not like the sender or are not sure, or they may just not reply. Specifically, the data contains interaction records, each of which represents a contact by a tuple containing the identities of the sender and receiver and whether the contact was accepted (positive response from receiver to sender) or not. The former case is denoted a successful or positive interaction, otherwise it is unsuccessful or negative.

The training set covered a four week period in February, 2010 and the test set a one week period from the first of March, 2010 (test results from a three week period from the same date were essentially identical and are omitted due to lack of space). Training and test sets contained all users with at least one contact in the respective periods. The training (resp. test) sets contained 166699 (95814) users with a total of 1710332 (436128) interaction tuples. Of these 264142 (66482) were positive interactions and 1446190 (369646) were negative (including non-replies). The *default success rate* (DSR), the proportion of interactions that were positive, was 15.4% (15.2%).

Methodologies. We compare the proposed algorithm with the P2P *Best 2CF+* method [12] on their evaluation metric and a new metric defined below. As far as we are aware this is the best-performing published CF method for recommendation in online dating. Model-based methods such as matrix factorisation methods were also tested but were not able to handle such a large dataset.

We trained ProCF and Best 2CF+ using the above training set. The learned model for each approach was then tested by generating the Top-N recommendations for each user in the test set. Finally, recommendations from ProCF and Best 2CF+ were evaluated as described below and the results are compared.

Evaluation Metrics. Evaluation in this domain is more complex than standard CF applications. Metrics used to capture key aspects of system performance are defined as follows. *Precision/Success Rate(SR)*: proportion of interactions predicted to be successful that were actually successful to all predicted successful interactions (PSI). *Default Success Rate(DSR)*: proportion of actual successful interactions (SI) to all interactions in the dataset. *SRI*: ratio of SR to DSR. *Recall and F Value*: recall is proportion of true PSI to all true SI. F Value is defined by $F = \frac{2*Precision*Recall}{Precision+Recall}$. *Accept Rate and ARI*: accept rate (AR) is proportion of true PSI to all PSI with either positive or negative reply. ARI is the ratio of AR to default accept rate without recommendation. *Reject Rate and RRI*: reject rate (RR) is proportion of false PSI with negative reply to all PSI with any reply. RRI is the ratio of RR to default reject rate without recommendation.

We use SRI to test how likely recommendation is to help the active user to have a positive interaction. Recall and F value tests how different user behaviour based on recommendation is to default. Finally, ARI and RRI test responses of the recommended user when the active user follows the recommendation.

Table 1. Average Positive Reply Rate per Receiver

	baseline	all	100	90	80	70	60	50	40	30	20	10
B2CF	0.372	0.370	0.370	0.370	0.370	0.371	0.371	0.371	0.371	0.372	0.374	0.378
ProCF	0.372	0.369	0.370	0.370	0.370	0.370	0.370	0.370	0.370	0.370	0.372	0.379

Results. A comparison of ProCF and Best2CF+ on the test set in terms of the evaluation metrics is shown in Fig. 3. Clearly ProCF outperforms Best 2CF+ on precision for all Top-N recommendations; the most significant comparative improvement is on Top 100 where ProCF outperforms Best 2CF+ by 34%. SRI shows that although Best 2CF+ improves the baseline performance of the system for all Top-N, ProCF achieves greater improvement. Since [12] show that Best 2CF+ outperforms traditional CFs on P2P recommendation, ProCF has a clear advantage. This suggests that considering both positive and negative collaborative information in creating similar pairs using probability residue leads to recommending users with higher probabilities of successful interaction with the active user. Recall and F Value improvements for ProCF indicates greater reliability in recommendation. Also, ProCF shows increased accept rate and reduced reject rate for the most highly ranked users. This supports the hypothesis that by looking at the difference between positive and negative information, ProCF can down-rank candidates with higher reject rate while up-ranking those with higher accept rate.

In Table 1, we show the average positive reply rate (APRR) per receiver over all recommended users compared to the those over all users in datasets. We also compared ProCF to Best 2CF+ on APRR with ProCF shown in bold in the figure. We can see from the results that both Best 2CF+ and ProCF have a similar APRR to the baseline over all Top-N. Best 2CF+ achieved smaller APRR than the baseline except from Top 20 and 10 while ProCF achieved even smaller APRR than Best 2CF+ except only Top 10. This indicates that ProCF does not recommend users who have higher positive reply rate. In contrast, it recommends users with lower positive reply rate in general. A recommender system that prefers recommending users with higher positive reply could improve the success rate. However, ProCF does not have a concentration bias on those users and thus does not commit the problem of recommending very frequent items as in [8]. Diversity in P2P recommendation is important; for example, recommending people who always say "yes" to any contact is poorly personalized and leads to poor recommendation performance. ProCF's achievement of high success rate while maintaining diversity is a significant result.

Discussion. The experiments have shown that ProCF implementing our probabilistic similarity function significantly outperforms Best 2CF+ on all evaluation metrics used. This proves that ProCF also outperforms all standard CF methods and combined CF methods evaluated in [12]. Note that ProCF achieved its performance by a single improved CF method not by any combination of CFs or profile-based methods. The characteristics of ProCF and its good performance suggest it could be used as an improved standard CF method to be integrated

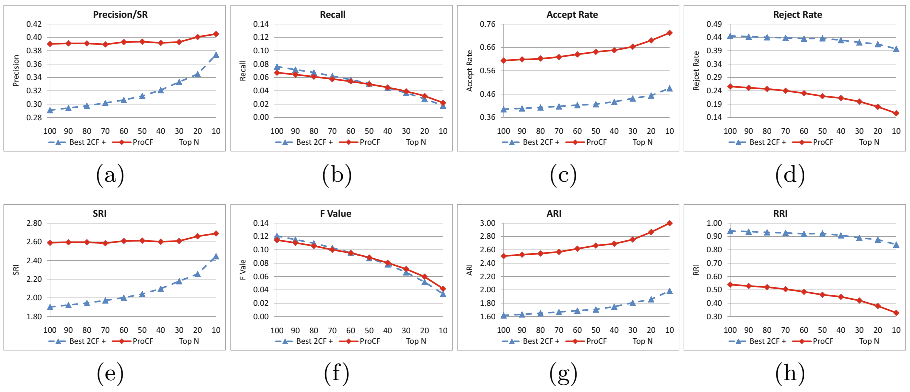


Fig. 3. Results on Test Set (1-7 March, 2010). ProCF significantly outperformed Baseline and *Best 2CF+* in precision, accept rate and reject rate on all Top-N evaluations and improved the recall/F value from Top 50 to Top 10). (For reject rate/RRI, the lower the better. For others, the higher the better.)

into existing common recommendation frameworks for increased performance. Beside good performance, ProCF has the advantage of being simple, easy to implement and fast. All presented algorithms were implemented using SQL in Oracle 11. ProCF required about 1 hour for training and several minutes for testing on a workstation with 64-bit Windows 7 Professional, 2 processors of Intel(R) Xeon(R) CPU x5660@2.80GHz and 32GB RAM.

6 Conclusion

We presented a general and straightforward framework, ProCF, for recommender systems. Although ProCF is in general applicable to both I2P and P2P recommendation, this paper focuses on P2P recommendation only. We demonstrated the usefulness of ProCF in a set of extensive experiments. The experiments were conducted on demanding real world datasets collected from a commercial social network site. The experimental evaluation of ProCF shows that it is suitable for P2P recommendation. The comparative evaluation to two of the best CF methods on this task shows that ProCF outperforms the best CF-based method for P2P recommendation. We also showed that ProCF retains diversity of recommendation while providing higher accuracy recommendation. It does not only recommend a small group of users with high positive reply rate.

An appealing property of our framework is its simplicity and modularity. Because it follows a standard CF framework with its improved similarity and ranking functions, it can be applied or integrated into existing CF recommender systems to improve system performance.

In the future, we will extend this work to test ProCF using other probabilistic functions for similarity and other distribution functions for calculating probability residue. We will also investigate the integration of profile based approach into ProCF for even better recommendation performance.

Acknowlegement. This work was funded by Smart Services Cooperative Research Centre. We would also like to thank our industry partners for providing the datasets.

References

1. Billsus, D., Pazzani, M.J.: Learning collaborative information filters. In: Proc. ICML 1998, pp. 46–54 (1998)
2. Breese, J., Heckerman, D., Kadie, C.: Empirical analysis of predictive algorithms for collaborative filtering. In: Proc. UAI 1998, pp. 43–52 (1998)
3. Bull, S., Greer, J.E., McCalla, G.I., Kettel, L., Bowes, J.: User modelling in i-help: What, why, when and how. In: Bauer, M., Gmytrasiewicz, P.J., Vassileva, J. (eds.) UM 2001. LNCS (LNAI), vol. 2109, pp. 117–126. Springer, Heidelberg (2001)
4. Cai, X., Bain, M., Krzywicki, A., Wobcke, W., Kim, Y., Compton, P., Mahidadia, A.: Collaborative Filtering for People to People Recommendation in Social Networks. In: Li, J. (ed.) AI 2010. LNCS, vol. 6464, pp. 476–485. Springer, Heidelberg (2010)
5. Cai, X., Bain, M., Krzywicki, A., Wobcke, W., Kim, Y., Compton, P., Mahidadia, A.: Learning collaborative filtering and its application to people to people recommendation in social networks. In: Proc. ICDM 2010, pp. 743–748 (2010)
6. Castells, P., Vargas, S., Wang, J.: Novelty and diversity metrics for recommender systems: Choice, discovery and relevance. In: DDR 2011 Workshop at ECIR 2011 (2011)
7. Delgado, J., Ishii, N.: Memory-based weighted-majority prediction for recommender systems. In: ACM SIGIR 1999 Workshop Recommender Systems (1999)
8. Deshpande, M., Karypis, G.: Item-based Top-N Recommendation Algorithms. ACM Transactions on Information Systems 22(1), 143–177 (2004)
9. Fleder, D.M., Hosanagar, K.: Recommender systems and their impact on sales diversity. In: Proc. ACM Electronic Commerce, pp. 192–199 (2007)
10. Hofmann, T.: Collaborative filtering via gaussian probabilistic latent semantic analysis. In: Proc. ACM SIGIR 2003, pp. 259–266 (2003)
11. Konstan, J., Miller, B., Maltz, D., Herlocker, J., Gordon, L., Riedl, J.: Grouplens: Applying collaborative filtering to usenet news. In: CACM, pp. 77–87 (1997)
12. Krzywicki, A., Wobcke, W., Cai, X., Mahidadia, A., Bain, M., Compton, P., Kim, Y.: Interaction-based collaborative filtering methods for recommendation in online dating. In: Chen, L., Triantafillou, P., Suel, T. (eds.) WISE 2010. LNCS, vol. 6488, pp. 342–356. Springer, Heidelberg (2010)
13. Linden, G., Smith, B., York, J.: Amazon.com recommendations: item-to-item collaborative filtering. IEEE Transactions on Internet Computing 7(1), 76–80 (2003)
14. Liu, T.Y.: Learning to Rank for Information Retrieval. Springer (2011)
15. Malinowski, J., Keim, T., Wendt, O., Weitzel, T.: Matching people and jobs: A bilateral recommendation approach. In: Proc. HICSS 2006, vol. 6, p. 137.3 (2006)
16. Pavlov, D., Pennock, D.: A maximum entropy approach to collaborative filtering in dynamic, sparse, high-dimensional domains. In: NIPS 2002, pp. 1441–1448 (2002)
17. Sarwar, B., Karypis, G., Konstan, J., Reidl, J.: Item-based collaborative filtering recommendation algorithms. In: Proc. WWW 2001, pp. 285–295 (2001)
18. Shardanand, U., Maes, P.: Social information filtering: Algorithms for automating "word of mouth". In: Proc. HCI 1995, pp. 210–217 (1995)
19. Terveen, L., McDonald, D.W.: Social matching: A framework and research agenda. ACM Transactions on Computer-Human Interaction 12(3), 401–434 (2005)
20. Yu, K., Schwaighofer, A., Tresp, V., Xu, X., Kriegel, H.: Probabilistic memory-based collaborative filtering. IEEE TKDE 16(1), 56–69 (2004)