People-to-People Reciprocal Recommenders

Irena Koprinska and Kalina Yacef

1 Introduction

Recommending people to people is the core task of many social websites and platforms. Examples include finding friends, professional contacts and communities to follow on social networks; matching people in online dating websites, job applicants with employers, mentors with mentees, and students in online learning courses. While social networks such as Facebook and LinkedIn aim at connecting people by creating *n*-to-*n* relationships, other applications such as online dating websites aim at matching people to create 1-to-1 relationships.

Most people-to-people recommendations, and especially the 1-to-1 recommendations, involve creating relationships that are reciprocal, i.e. where both parties can express their likes and dislikes and a good match requires satisfying the preferences of both parties. For instance, in the process of hiring someone for a job, both the candidate and the company offering the job need to assess each other, deciding whether the candidate is fit for the position and vice-versa. In online dating, reciprocity is fundamental—users will build a successful relationship only if both parties are interested in each other. Matching students in education may also require reciprocity in order to maximise learning benefits.

The key role of reciprocity for recommending people to people has only recently being recognised. In this paper we discuss the distinctive nature of reciprocal recommenders, review the previous work and present a case study in online dating.

I. Koprinska (✉) · K. Yacef
School of Computer Science, The University of Sydney, Sydney, NSW, Australia
e-mail: irena.koprinska@sydney.edu.au; kalina.yacef@sydney.edu.au

F. Ricci et al. (eds.), *Recommender Systems Handbook*,
https://doi.org/10.1007/978-1-0716-2197-4_11

2 Reciprocal vs. Traditional Recommenders

Reciprocal recommenders must satisfy the preferences and needs of the two parties involved in the recommendation. In contrast, the traditional items-to-people recommenders are one sided and must satisfy only the preference of the person for whom the recommendation is generated. Table 1 summarizes the differences between the two types of recommenders; a comprehensive comparison can be found in [1].

The user behaviour is highly dependent on whether the domain is reciprocal or not. The success of a traditional book recommender is dependent only on the person receiving the recommendation. On the other hand, in a reciprocal domain such as online dating, the user receiving the recommendation knows that the success depends on both parties and this influences his/her behaviour. In addition, users in reciprocal domains may choose to act proactively by taking the initiative to connect with other users or to remain reactive and wait for contact.

Another difference is that for traditional recommenders, users generally have no reason to provide detailed information about themselves (user profile) and their preferences. In contrast, for reciprocal recommenders, there is a clear need and benefit for providing rich user profiles. These profiles might be inaccurate (e.g. due to a lack of self-awareness or desire to have a more attractive profile) and reciprocal recommenders need to account for that.

In traditional recommenders, satisfied and loyal users are likely to repeatedly use the site, allowing it to build rich user model by exploiting the explicitly and implicitly stated user preferences. In contrast, in reciprocal domains people may leave the site permanently after a successful recommendation. For example, a person who successfully finds a lifelong spouse on a dating website or who finds a long term job on a job website may not need to use these sites after that. This creates a paradox for this service provides who want their service to be the best for their users and therefore achieve what they are set to do. But at the same time, if they do provide the best recommendations, users may not use their services for long, possibly affecting revenue. On the other hand, happy users will refer the services

Table 1 Main differences between reciprocal and traditional recommenders

Traditional recommenders	Reciprocal recommenders
Success is determined solely by the user seeking the recommendation	Success is determined by both users—the subject and object of the recommendation
Users have no reason to provide detailed explicit user profiles.	Users are expected to provide detailed self-profiles and preferences
Satisfied users are likely to return for more recommendations. Better recommendations mean more engagement with the system in the future	Users may leave the system after a successful recommendation. Better recommendations might mean less engagement with the system in the future
The same item can be recommended to all users	Popular users should not be recommended to too many users

to new users, and are likely to use the service again if there is a future need for it. This is a clear multi-objective optimization problem. However, it is important to highlight that both objectives, i.e. (1) good successful recommendations for users and (2) short term revenue goals, should not be optimized in equal weights, since an optimization for short-term revenue is likely to hurt the service in the long term, while optimizing for the goodness of users may actually benefit the whole service. The key to the multi-objective optimization here is to keep short-term revenue high without decreasing user satisfaction.

Finally, in reciprocal domains it is important that users are not recommended to others in a way that may cause them to be overloaded with recommendations. The popularity bias, which may be an issue for traditional recommender systems, becomes an exacerbated problem for reciprocal recommender systems. For instance, if a highly qualified person is recommended to every single job position that he/she fits, this person is likely to be burdened by the amount of contacts and leave the website. A similar situation can occur for popular users in a dating website. These users are important as they represent the best for each service, therefore they should only be recommended to other users when the recommender is absolutely sure that these users will reciprocate the contact. We note that this is a distinguishing feature of reciprocal recommenders compared to non-reciprocal people-to-people recommenders such as social networks where users can manage n-to-n connections.

3 Previous Work

In this section we review previous work, focusing on reciprocal recommenders but also mention some key non-reciprocal people-to-people recommenders which are relevant and illustrate important characteristics of reciprocal domains. A survey paper on reciprocal recommenders has recently become available [2]. We note, however, that it covers both reciprocal and non-reciprocal people-to-people recommenders.

3.1 Social Networks

In the broad area of social matching [3, 4], see also Chapter "Social Recommender Systems", recommending people to other people has a clear link with reciprocal recommenders because the quality of a match is determined by both parties involved in the match. However, most of the existing work on social matching tailors recommendations only to the needs of one of the parties. Just a few papers mention the need for reciprocity and even fewer attempt to act on it.

Chen at al. [5] presented algorithms for recommending people on the enterprise social network Beehive, which allows users to connect to co-workers, post new information or comment on shared information. Two recommendation approaches

were studied: content-based and collaborative filtering, and four algorithms were compared. The content-based approach assumes that if two people post content on similar topics, they are likely to be pleased to get connected. It is based on similarity of textual content posted by the user on Beehive and additional information such as job description and location. The collaborative filtering is a typical friend-of-friend approach and uses only linking information from the social network. It is based on the intuition that if many of *A*'s connections are connected to *B*, then *A* may like to connect to *B* too. The results show that all approaches increased the number of connections, compared to a control group that received no recommendations. The content-based approach was more successful in recommending contacts that were unknown to each other, while the collaborative filtering approach was more successful in finding known contacts. It is important to note that the befriending in Beehive is non-reciprocal, i.e. any user can connect with another user without the consent of the second user. However, there are still important reciprocal social considerations as noted by the authors, e.g. before adding a contact, one has to consider how the other person would perceive this action and whether they will reciprocate the connection, and also how the new contact will be perceived by the other people using the social network service.

Guy at al. [6] studied people recommendations on another enterprise social network site. They proposed an algorithm for recommending "strangers"—people who the user does not know but may be interested in, based on mutual interests. The similarity is computed based on shared activities such as bookmarking the same page, commenting on the same blog, reading the same file, using the same tag, being tagged with the same tag and being member of the same community. The results showed that the recommender was able to achieve its goal; the users commented on its usefulness for connecting them with experienced people to learn from, becoming aware of others with similar expertise, projects and roles in other places or departments. Similarly to [5], this is not a reciprocal recommender since adding connections did not require the consent of both party, however considering reciprocity when making and accepting recommendations is important due to the social dynamics in the enterprise.

Fazel-Zarandi et al. [7] studied different social drivers to predict collaborators for scientific research. These drivers were grounded in social science theories and included: level of expertise (based on topic and publications), friend-of-friend (based on previous collaborations, homophily (similar gender, affiliation, tenure status and co-citation), social exchange (dyalic relationship based on demand and supply of resources), and follow the crowd (popularity). A prototype recommender system was developed and evaluated using data from grant applications and publication histories. Homophily was found to be the most important factor for predicting collaboration, followed by expertise. This work is relevant as many of the social drivers may be considered reciprocal as aspects such as homophily, friend-of-friend and even level of expertise can be reciprocal, e.g. the mutually beneficial relationship between students and mentors in scientific collaborations.

3.2 Mentor-Mentee Matching

The i-Help system [8] helped students find people who could assist them with university courses, e.g. with first year computer science problems. It matched helpers with helpees in a non-reciprocal manner by considering the attributes of the helpers and the preferences of the helpees. For the helpers, it stored or inferred attributes such as knowledge of the topic, interests, cognitive style, eagerness to help, helpfulness, availability, and current load. The information was collected from several sources including self-evaluation and peer feedback in previous help sessions. An initial ranked list of potential helpers was produced. It was then refined by considering the preferences of the helpee, e.g. the importance of criteria such as helpfulness and urgency, the preferred and banned helpers. A final list of five potential helpers was compiled and the first of them to reply became the helper.

The PHelpS system [9] is an earlier prototype of i-Help. It was used in a workplace to train staff in how to use a new data management system. The candidate helpers were filtered based on their knowledge of the task, availability and load using a constraint solver. The list was presented to the helpee who chose the helper. Both i-Help and PHelpS relied on rich user models encoding the expertise and preferences of helpers and helpees.

Li [10] considered the task of matching mentors with mentees on the online platform Codementor which provides a 1-to-1 help for software developers. Users (mentees) post requests for help with programming issues and are matched with a sutable mentor who can help. Reciprocity is taken into account by considering the requirements of both sides. The recommendation problem is divided into two tasks—mentor willingness prediction and mentee acceptance prediction, which are solved simultaneously, generating a ranked list of mentors. Four groups of predictive feature are proposed: (1) availability of mentors in terms of time and availability of mentees in terms of budget, (2) capability of mentors to deal with the request based on the mentor's expertise, ratings by past mentees and past mentoring sessions, (3) activity of mentors—how frequently the mentor has expressed willingness to tackle request and how frequently the mentor was accepted by mentees and (4) proximity of the mentor to the current request—high values are assigned to mentors who have tackled similar requests in the past and are similar to other mentors who have tackled similar requests. The features were extensively evaluated using various supervised machine learning methods and shown to be effective for both tasks, with activity and proximity found to be most effective. The proposed recommendation approach was also evaluated online on Codementor for two weeks, showing accurate mentor prediction, faster selection and higher satisfaction by mentees.

3.3 Online Learning Courses

Labarthe et al. [11] studied the effect of a reciprocal peer recommender to foster students' persistence and success in a Massive Online Open Course (MOOCs). This work is based on the finding that social isolation is a contributing factor for poor learning experience and attrition in MOOCs, and that this is further exacerbated by the difficulty of finding the right people to interact with in a newly-formed MOOC community. The authors hypothesized that helping students to connect with other students in the MOOC would alleviate this problem. Their peer recommender provided each student with an individual list of potentially interesting contacts, created in real-time based on their own profile and activities. They evaluated the effect of the reciprocal recommender in four dimensions of learners' persistence: attendance, completion, success and participation. Results of their controlled study ($n = 8376$) showed improvement across all these four factors: students who received peer recommendations were much more likely to persist and engage in the MOOC than if those who did not [12]. The same team then investigated further what recommendation strategy had the most impact [13]. They conducted another controlled study ($n = 2025$) comparing three recommendation strategies: one using a socio-demographic-background similarity (gender, geo-location, education level and prior MOOC experience), another one based on the current stage of progress in the MOOC (to facilitate students contacting each other for specific questions about the content of the course), and a third one recommending random people to each other. Their findings showed that the socio-demographic-based peer recommender had a higher success than the other two.

Prabhakar et al. [14] proposed a reciprocal recommender algorithm for a MOOC using a similarity matrix based on the learners' preferences. They compute a reciprocal score for each pair of users that is the harmonic mean of the distance scores between them, where the distance score represents how the preferences of the first user matches the attributes of the other user. They found that the reciprocal algorithm tends to recommend to a given user people who also have that user in their recommendation list, which is as expected given that the reciprocity was promoted. Further work is needed to evaluate the recommender in real settings, as it is yet to be deployed with real users.

3.4 Job Recommendation

Malinowski et al. [15] investigated the problem of matching people and jobs and argued that the matching should be reciprocal, considering the preferences of both the job seeker and the recruiter. They built two recommender systems. The first one recommended job seekers (i.e. their resumes/profiles) to job descriptions of a particular recruiter. To create training data, a recruiter manually labelled a set of resumes as either fit or not fit for a list of jobs, based on demographic, educational,

job experience, language, technology skills and other attributes. The second system recommended jobs to job seekers. To create training data, the job candidates were asked to rank a set of job descriptions indicating how well the jobs fitted their preferences. In both cases the authors used the expectation maximisation algorithm to build the prediction model. The two recommender systems were evaluated separately, showing promising prediction accuracy results. Several methods for combining the two recommendations were proposed but were not implemented and evaluated.

Hong at al. [16] developed and deployed iHR—an online recruitment system, linking job seekers with recruiters, and providing reciprocal recommendations, in addition to content-based and colaborative filtering. For job seeks, iHR collects information from three sources: (1) user profile and job preferences provided by the job seeker, (2) information automatically extracted from the uploaded resume and (3) behavioural information from the user's activity on the site, e.g. keywords used for searching, preferred profiles viewed and recommendations clicked. The reciprocal recommender takes into consideration three properties: *reciprocity, availability* and *diversity*. *Reciprocity* uses a relevance score which matches the preferences and user profiles of both parties; *availability* limits the number of recommendations during a given time period for both parties so that they are not overwhelmed; *diversity* is calculated based on the similarity of the job seekers—the goal is to recommend candidates with different personal strengths. The reciprocal recommendation task is formulated as an optimisation problem by considering the three properties.

The reciprocal recommender is evaluated on a large dataset for 3 years (almost 200,000 job seekers and 47,000 recruiters) and compared with other state-of-the-art methods, showing that it was the most accurate method and, hence, demonstrating the benefits of reciprocal recommendations. To assess the user experience, a user study based on surveys was also conducted comparing the content-based, collaborative filtering and reciprocal approaches, in terms of relevance, interpretability, diversity and ordering of the results. The reciprocal recommender received the most positive feedback. Hence, it outperformed the other methods in both accuracy and user experience. iHR is one of the first systems that incorporates reciprocal recommenders for recruitment. Importantly, it was also deployed in practice for the Xiamen Talent Service Center in China.

3.5 Online Dating

The reported research on building recommender systems for online dating is growing, with most of the papers published in the last 10 years. Since reciprocity is fundamental for online dating, the majority of work on reciprocal recommenders is in this area.

One of the first studies of recommender systems for online dating is the work of Brozovsky and Petricek [17] who evaluated the performance of two collaborative

filtering approaches (item-to-item and user-to-user). They used data from a Czech online dating site containing about 12 million ratings and 195,00 users. The results showed that both algorithms outperformed the baselines based on random and mean predictions. A user study was also conducted confirming that the users preferred the collaborative filtering approaches. The authors mentioned the need for reciprocity, but did not explore it.

Another early work is [18], which formulated the matchmaking task as an information retrieval problem, where user profiles were ranked with respect to a given ideal partner profile (i.e. explicit user preferences). Using historical data, a training set of matches (pairs of users represented with their profile attributes) was created and labelled as relevant and non-relevant. A match was considered relevant if users exchanged contact information, and irrelevant if one of the users inspected the other user's profile but did not send a message or if he/she sent a message but the other user did not reply. A machine learning classifier (ensemble of boosted regression trees) was built and used to predict the relevance of new matches; given a new user, the potential candidates were ranked based on their predicted score. The approach was evaluated using data from an online dating website. The authors described the reciprocal aspect of their work as two-sided relevance and stressed its usefulness for ranking candidates in matchmaking problems. Also relevant is the work of McFee and Lanckriet [19] who proposed a method for learning optimal distance metrics and evaluated it an online dating data. However, reciprocity was not discussed in the paper; its main focus is the new general algorithm for learning distance metrics rather than the online dating application.

In [20], our research group proposed the content-based system RECON which is one of the first reciprocal recommenders for online dating. RECON uses both user profiles and user interactions for matching people; to produce recommendations for a given user, it extracts his/her implicit preferences (i.e. the preferences that are inferred from the interactions with the other users) in terms of attribute values and then matches them with the profiles of the other users. Two one-sided compatibility scores are calculated and then combined using the harmonic mean to produce a *reciprocal compatibility score* of the two users. Using data from an Australian dating website, we showed that reciprocity improved both the success rate and recall of the recommender (see further in the article how the success rate was computed).

In [21], we proposed CCR—a recommender system for online dating that combines content-based and collaborative filtering approaches and utilises both user profiles and user interactions (see the Case Study section for more details).

In another early work, Kim et al. [22] created a people recommender system for social networking websites where users can reply positively or negatively to messages from other users. The system was evaluated on data from an online dating website. The authors distinguish between recommender systems for one-way and two-way interaction. They propose an approach for a two-way interaction that considers both the interest of the sender and the interest of the recipient of message, and makes recommendations by combining them with a weighted harmonic mean. The method uses both user profiles and information about previous user interactions. For a given user, it finds the best matching values for every attribute and then

combines them in a rule that can be used to generate recommendations. The proposed method was more accurate than a baseline, where users simply browse the site to search for people to connect to.

Cai et al. [23] developed a neighbour-based collaborative filtering approach, called SocialCollab, which considers the preferences of both sides. It is based on similarity of users in terms of attractiveness and taste. Two users are similar in *attractiveness* if they are liked by a common group of users, and similar in *taste* if they like a common group of users. To generate a recommendation for a user A, the SocialCollab algorithm considers all potential candidates R. For each candidate in R it first finds two groups of similar users, in terms of attractiveness and taste; the candidate is added to the recommendation list for A if there is at least one similar user in both groups that reciprocally liked A. The recommendations are ranked according to the number of similar users. SocialCollab was evaluated using data from a comercial online dating websiye for 2 weeks and shown to outperform standard collaborative filtering, confirming the importance of reciprocity in people-to-people recommenders. Cai et al. [23] improved on these results by using gradient descent to learn the relative contribution of similar users in the ranking of the recommendations given by SocialCollab. In the same domain, the work of [24] have reported improvements over Cai et al. by using a recommendation method based on tensor decomposition.

Building upon ideas from RECON and SocialCollab, Xia et al. [25] proposed content-based and collaborative filtering algorithms. Their content-based algorithm is very similar to RECON and the four collaborative filtering algorithms are based on the notion of attractiveness and taste similarity to SocialCollab with some extensions. An evaluation using a large dataset of 200,000 users from an online dating website in China showed that the collaborative filtering methods were more accurate than the content-based method and RECON. The reciprocal collaborative filtering methods proposed by Zhao at al. [26] are also based on attractiveness and taste as in SocialCollab and share similar ideas. The evaluation was done on a dataset of 47,00 users from a popular online dating website, containing user interactions for a period of 6 months.

Alsaleh et al. [27] used clustering to group the male users based on their attributes and the female users based on their preferences. The recomendations were generated by matching the male clusters with the female clusters based on user interactions, and recommending cluster members using compatibility scores. In their subsequent paper [28], a tensor space model was developed for finding latent relationships between users based on user attributes and interactions. The results showed that the proposed model was more accurate than SocialCollab [23] and other recommendation methods and baselines.

Kutty at al. [29] developed a novel reciprocal recommender for online dating based on graph theory and machine learning. They studied in depth the properties of online dating networks and proposed a novel representation, called attributed bipartite graph, to model the user profiles and interactions. In this representation, the nodes correspond to the users, the node attributes include both user profile attributes (e.g. age, gender, education, etc.) and user preferences, and the links correspond to

the user interactions. To generate the recommendations, three algorithms are combined: (1) node-based similarity algorithm to predict similarity between bipartite nodes, (2) k-means similar node checking to help with the cold-star problem and (3) a reciprocal node compatibility algorithm that matches the users using a support vector machine prediction algorithm. An extensive evaluation using a large online dating website shows the advantage of the proposed reciprocal approach compared to collaborative filtering and other state-of-the-art recommendation methods.

Alanazi and Bain [30, 31] investigated reciprocal recommenders that consider temporal aspects of user behavior. The motivation is that user behaviour and preferences are not static but change over time. Using Hidden Markov Models (HMM), in their early work [30] the recommendation problem was represented is a graph, where the nodes are the users and the edges are the links between them; given the current state, the goal is to predict the next state, and in particular the new links (matches). In their subsequent work [31], Alanazi and Bain proposed to combine the content-based HMM part with a collaborative filtering part to improve speed and scalability to large datasets. In the proposed hybrid recommender CFHMM-HRT, the collaborative filtering part generates an initial list of recommendations, which is then validated by the trained HMM recommender to output a smaller list, that is ranked to produce the final list of recommendations. CFHMM-HRT was evaluated using a large dataset from an online dating website, showing that it was time efficient and generated considerably more accurate recommendations compared to its content-based and collaborative filtering counterparts.

Vitale at al. [32] formulated the reciprocal recommendation problem as a sequential learning problem. The learning consists of a sequence of rounds; at each round, a user from one of the parties becomes active and based on past feedback, the learning algorithm recommends a user from the other party. The goal is to uncover as many reciprocal matches as possible, and to do this quickly. The paper introduces assumptions for effective learning and an algorithm called SMILE, designed under these assumptions. SMILE is analysed theoretically in terms of computational complexity and limitations on the number of matches. An experimental evaluation is also conducted using synthetic datasets and the real dataset used in [17].

Kleinerman et al. [33] proposed a novel reciprocal recommender method called Reciprocal Weighted Score (RWS), which finds the optimal balance between the interests of both users, in order to create a successful interaction. It calculates two scores: (1) CF—the interest of user A in the recommended user B using collaborative filtering and (2) PR—the likelihood that B will reply positively to A, using a machine learning method. These two scores are combined in a weighed sum, where the weights are calculated separately for each user by formulating the task as an optimization problem. The results show the importance of individual weighting for successful interactions—the weights of the two components vary considerably between users, with most of the women having a higher weight for the CF score and most of the men having a higher weight for the PR score. The machine learning task is formulated as a binary classification problem: (1) positive reply and (2) negative or no reply, with AdaBoost employed as a classifier. The feature vector consists of 54 features describing the sender, receiver and their

activities on the platform, selected using domain knowledge and machine learning feature selection methods. The evaluation was done online on a operational online dating platform (Doovdevan), and the results were compared with the reciprocal collaborative filtering method of Xia et al. [25] which captures the mutual interest of the two users (referred to as RCF). The results showed that RWS was more effective than RCF—it generated a higher number of successful recommendations. An extension of this work was presented in [34]; it includes an offline evaluation of RWS, and a reciprocal explanation module which was combined with RWS and extensively evaluated (see also Sect. 6).

Reciprocal recommenders based on latent factors have also been recently proposed. Neve and Palomares [35] developed LFRR—a collaborative filtering algorithm which learns two latent factor models—one for the male preferences and another one for the female preferences towards the opposite group. Each latent model uses matrix factorization and gradient descent to minimize the error on the known ratings, and then makes predictions about the likelihood of a successful interaction between two users by calculating the dot product of the user feature vectors. Similarly to RWS [33], RCF [25] and RECON [20], LFRR computes two unidirectional preference scores from the latent models, which are then combined in a single reciprocal preference score. Four aggregation functions were investigated: arithmetic, geometric and harmonic mean, and cross-ratio uninorm; an additional study exploring these aggregation functions is [36]. The evaluation was conducted offline using data from the Japanese online dating platform Pairs, comparing LFRR with RCF. The results showed that LFRR performed similarly to RCF in terms of accuracy but was faster at generating the recommendations, and also that the type of aggregation function significantly affects the results. In [37] Ramanathan et al. report on the development and deployment of a latent factor reciprocal system similar to LFRR, but which learns from both positive and negative preferences, not only positive as LFRR. It was compared with LFRR in an offline evaluation showing an improvement, and then deployed on a popular online dating platform in Japan with more than 5 million users.

Physical appearance and attractiveness are important factors for successful relationships. Although user photos are widely used in online dating platforms, they have rarely been used in recommender systems, perhaps because photos are more complex to analyse (different angles, quality, etc.) and attractiveness is subjective. Recently, Neve at al. [38] proposed ImRec—a reciprocal recommender that utilizes user photos to train a machine learning classifier (Siamese convolutional neural network), followed by a Random Forest algorithm, to predicts user A's preference for user B, based on A's previous positive and negative interactions. Two uni-directional scores are calculated and subsequently combined in in a single reciprocal score. An offline evaluation on data from the Pairs platform (also used in their previous work [35, 36]) showed that InRec outperformed RECON which doesn't image data, and also LFRR but only when very little data is available about the user (less than 5 positive and negative interactions). This shows the effectiveness if ImRec for addressing the cold-start problem for new users and suggests that ImRec and LFRR can be useful in a switching hybrid system based

on the number of interactions. The authors emphasize the importance of providing effective recommendations on the first day—it was found that the Pairs users often decide if to commit to the platform based on their experience during the first 24 hours.

4 A Case Study in Online Dating

Online dating services, e.g. Match.com, eHarmony, RSVP, Zoosk, OkCupid, PlentyOfFish, Meetic, Badoo, Tinder and Bumble, are used by millions of people and their popularity is growing. Their revenue is also steadily increasing, e.g. according to [39] the online dating revenue in US was $602 million dollars in 2020 and is projected to reach $755 billion by 2024. In 2020 44.2 million Americans used online dating and this is expected to increase to 53.3 million by 2024. A recent study [40] showed that meeting online has become the most popular way heterosexual couples meet in US, overtaking traditional ways such as meeting through friends and family.

To find dating partners using an online dating website or app, users usually provide information about themselves (*user profile*) and their preferred partner (*user preferences*); an example using predefined attributes is shown in Table 2. The *explicit* user preferences are the preferences stated by the user as shown in Table 2. The *implicit* user preferences are inferred from the interactions of the user with other users and may be quite different to the explicit user preferences (e.g. when a user contacts exclusively short people who smoke in spite of stating in their preferences that they are looking for tall people who do not smoke).

We worked with a major Australian dating site where the user interactions include four steps:

1. Creating a user profile and specifying the explicit user preferences—New user Bob creates an account on the website and provides information about himself (user profile) and his preferred dating partner (explicit user preferences) using a set of predefined attributes such as the ones shown in Table 2. He can also add some textual information to expand on his tastes and personality.
2. Browsing the user profiles of other users for interesting matches—Bob finds Alice and decides to contact her.
3. Mediated interaction—Bob chooses a message from a predefined list, e.g. *I'd like to get to know you, would you be interested?* We call these messages *Expressions of Interest (EOI)*. Alice can reply with a predefined message that is either positive (e.g. *I'd like to know more about you.*) or negative (e.g. *I don't think we are a good match.*) or may not reply at all. When an EOI receives a positive reply, we say that the interest is *reciprocated*.

   We define an interaction between users A and B as *successful* if *A* has sent an EOI to *B* and *B* has responded positively. Similarly, an interaction between A and B is *unsuccessful* if *A* has sent an EOI to *B* and *B* has responded negatively.

Table 2 User profile and explicit user preferences

Bob	My details (Who I am?)	My ideal partner details (Who I am looking for?)
Age	44 years old	35–46 years old
Location	Sydney	within 20 km
Height	175 cm	at most 175 cm
Body type	Athletic	Slim, average, athletic
Smoking	Trying to quit	Trying to quit, Don't smoke
Relationship status	Divorced	Single, divorced, widowed, separated
Have children	Have children who don't live at home	Have children who don't live at home, Have children living at home, Have no children
	How many: 2	
	Age range: 18–23 years old	
Personality	Social	Social, average
Eye colour	Blue	–
Hair color	Brown	–
Nationality	Australian	–

4. Unmediated interaction—Typically after a successful interaction, Bob or Alice buys tokens from the website to send each other unmediated messages. This is the only way to exchange contact details and develop further their relationship.

While the relationship, once taken offline, may or may not become successful for Bob and Alice, reaching the fourth step is crucial and necessary to make it possible for them to find out. It is also the extent to which the dating website can go.

A major hurdle for progressing through these steps is that users must find, fairly quickly, relevant users among the hundreds of thousands available. Failure to do so can result in a loss of interest ("there is no-one who I like"), or a feeling of rejection when they contact people who don't reciprocate ("noone wants to talk to me"). Therefore, an efficient reciprocal recommender algorithm is essential for a good customer experience.

4.1 CCR: Content-Collaborative Reciprocal Recommender for Online Dating

CCR is our Content-Collaborative Reciprocal recommender [21]. It uses information from the user profile and user interactions to recommend potential matches for a given user. The content-based part computes similarities between users based on their profiles. The collaborative filtering part uses the interactions of the set of similar users, i.e. who they like/dislike and are liked/disliked by, to produce

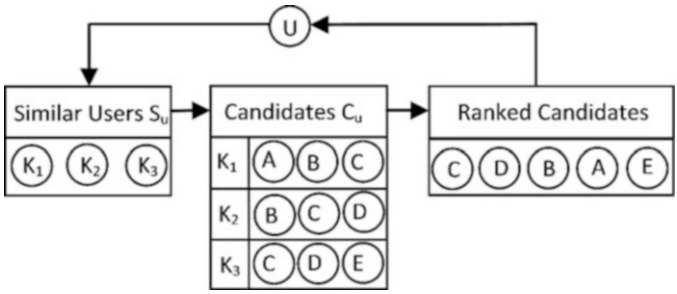

Fig. 1 The CCR recommender

the recommendation. The recommender is *reciprocal* as it considers the likes and dislikes of both sides and aims to match users so that the pairing has a high chance of success.

The main assumption of CCR, reflected in steps 1 and 2 below, is that a pair of users who have similar profiles will *reciprocally like* (meaning "like and be liked by") the same type of people (in terms of user profiles), i.e. if U has a similar profile to K_1 and K_1 reciprocally likes A, B and C, then U will reciprocally like A, B and C. We tested this hypothesis in [21] using correlation analysis and a large dataset of more than 7000 users and 167,00 EOI, and found that indeed similar people are reciprocally liked by the same type of people.

To generate a recommendation list for user U, CCR uses the following steps, also shown in Fig. 1:

1. **Generating similar users based on user profiles**
   This step produces a set of K users who have the most similar profile to U, i.e. that have the lowest possible distance to U. We use a modified version of the K-Nearest Neighbor algorithm, with seven attributes (*age, height, body type, education level, smoker, have children and marital status*) and a distance measure specifically developed for these attributes. For example, in Fig. 1 the set of similar users S_u for user U consists of K_1, K_2 and K_3.
2. **Generating recommendation candidates based on user interactions**
   This step produces a set C_u of candidate users for recommending to U. For every user K_i in S_u, we compute the list of all users with whom K_i had reciprocal interest with and add it to the set of candidates C_u. For example in Fig. 1, K_1 and A liked each other, so did K_1 and B, K_1 and C and so on, resulting in a recommendation candidate set for U of *{A, B, C, D, E}* with a frequency of 1, 2, 3, 2, 1 respectively.
3. **Ranking the candidates**
   This step uses a ranking method to order the candidates based on their desirability, and provide meaningful recommendations for U. Figure 1 shows a ranking method based on frequency—C is ranked the highest because it is the most frequent candidate in C_u.

Ranking Method Support We developed and compared a number of ranking methods, including two that utilise the explicit and implicit user preferences, and found that the Support ranking method was the best in spite of its simplicity [41]. The Support ranking method is based on the interactions between the group of similar users S_u and the group of candidates. Users are added to the candidates pool if they have responded positively to at least one S_u user or have received a positive reply from at least one S_u user. However, some candidates might have received an EOI from more than one S_u user, responding to some positively and to others negatively. Thus, some candidates have more successful interactions with S_u than others. The Support ranking method computes the support of S_u for each candidate. The support score for X is the number of positive interactions minus the number of negative interactions that X had with S_u. The higher the score for X, the more reciprocally liked is X by S_u. The candidates are then sorted in descending order based on their support score.

4.1.1 Evaluation

Data To evaluate the performance of CCR, we used a real dataset from our Australian website partner from active users who were Sydney residents and interacting with people of different genders. For each run of the experiment, the dataset is partitioned into two distinct sets, training and test, containing approximately 2/3 and 1/3 of the users, respectively. Each training/test partition contains an even distribution of males and females. Each set was also evenly assigned users who were more popular than their cohort in terms of EOI sent and/or received. The data characteristics are shown in Table 3.

EOIs and their responses from users in the training set to users in the test set and vice versa were removed to ensure fair evaluation. Users in either the test or training set who no longer meet the minimum number of EOI required were removed. This resulted in the removal of less than 1% of the users before the split into training and test sets. Information about the interactions of the users from the test set is never included when ranking the candidates for this user to ensure clear separation between the two sets.

Table 3 Data characteristics

Total users	216,662
Male users	119,102 (54.97%)
Female users	97,560 (45.03%)
EOIs	167,810
Successful EOIs	24,079 (25.59%)
Users sent/received at least 1 EOI	7322
Male users sent/received at least 1 EOI	3965
Female users sent/received at least 1 EOI	3357

Selected Attributes and Distance Measure The original dataset consists of 39 user profile attributes. After a preliminary analysis of the distribution of these attributes to assess their importance and suitability, we manually selected 7 attributes: 2 numeric (*age and height*) and 5 nominal: *body type* (*values: slim, average, overweight*), *education level* (*secondary, technical, university*), *smoker* (*yes, no*), *have children* (*yes, no*) and *marital status* (*single, previously married*). Some attribute values were merged together, e.g. the values *overweight* and *largish* of *body type* were merged into *overweight*.

To measure the similarity between the profiles of users A and B, we used a distance measure that considers the differences between all attributes, but weights higher the age difference. For nominal attributes, we used a binary representation and Hamming distance and for numerical attributes we used a function of absolute differences; for more details, see [21].

Performance Measures For a user U we define the following sets:

- Successful EOI sent by U, *successful_sent*: The set of users whom U sent an EOI where the response was positive.
- Successful EOI received by U, *successful_recv*: The set of users who sent an EOI to U where the response was positive.
- Unsuccessful EOI sent by U, *unsuccessful_sent*: The set of users whom U sent an EOI where the response was negative.
- Unsuccessful EOI received by U, *unsuccessful_recv*: The set of users who sent an EOI to U where the response was negative.
- All successful and unsuccessful EOI for U: *successful* = *successful_sent* + *successful_recv*, U: *unsuccessful* = *unsuccessful_sent* + *unsuccessful_recv*

For each user U from the test set, a list of N ordered recommendations *N_rec* is generated. The successful and unsuccessful EOI for U in this list are: *successful@N* = *successful* ∩ *N_rec* and *unsuccessful@N* = *unsuccessful* ∩ *N_rec*.

Then, the *success rate* at N (i.e. given the N recommendations) is defined as:

$$successRate@N[\%] = \frac{\#successful@N}{\#successful@N + \#unsuccessful@N} \quad (1)$$

Hence, given a set of N ordered recommendations, the success rate at N is the number of correct recommendations over the number of interacted recommendations (correct or incorrect).

For comparison we use the following baseline: the success rate of the recommender using a random set of K users in S_u as opposed to K nearest neighbors in step 1 of the CCR algorithm (see Fig. 1). The random set of K users is used to generate candidates that are then ranked, i.e. there is no change in steps 2 and 3.

Each experiment has been run ten times; the reported success rate is the average over the ten runs.

Results We evaluated the performance of CCR for different number of recommendations N (from 10 to 500) and different number of minimum number of EOI sent

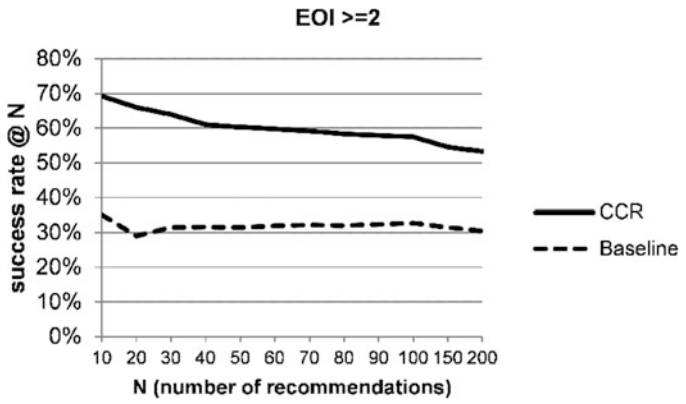

Fig. 2 CCR success rate results for *minEOI_sent* = 2

by a user *minEOI_sent* (from 1 to 20) and compared it to the baseline success rate of the recommender using a random set of K users in S_u as opposed to the K nearest neighbours. As an example, Fig. 2 shows the success rate for *minE-OI_sent* $= 2$. We found that CCR significantly outperforms the baseline for all cases. For example, for $N = 10$ and *minEOI_sent* $= 2$, the success rate of CCR is 69.26% and the baseline success rate is 35.19%.

As the number of recommendations N increases from 10 to 500, the success rate decreases by 10–20%. This means that the best recommendations are at the top of the list and adding more recommendations only dilutes the success rate. Hence, our ranking criterion is useful and effective. In practice, the success rate for a smaller N, e.g. $N = 10$–30 is very important as this is the typical N presented to the user. Unsuccessful recommendations, especially recommendations leading to rejection can be very discouraging.

Our results also show that as the number of *minEOI_sent* increases from 1 to 20, the success rate trends are very similar. However, for users who sent more EOIs, the success rate is slightly lower (e.g. 60.16% for *minEOI_sent* $= 10$ and 58.54% *minEOI_sent* $= 20$, for $N = 10$). This can be explained by the fact that the highly active users may be less selective.

In all experiments we used $K = 100$ and $C = 250$. With these parameters it took approximately 100 milliseconds to generate the recommendation list for a user which confirms the efficiency of our algorithm for generation of similar users and candidate recommendations.

4.2 Explicit and Implicit User Preferences

User preferences are essential for building a successful recommender. We investigate which type of user preferences (explicit or implicit) is more predictive of

success in user interaction. More details can be found in [41]. A related investigation is [42] where we defined a histogram-based model of implicit preferences learned from successful interactions only (as opposed to both successful and unsuccessful as in CCR), and evaluated them in the context of a content-based recommender.

Explicit User Preferences We define the explicit preferences of a user U as the vector of attribute values specified by U. The attributes and their possible values are predefined by the website.

In our study we used all attributes except *location*, i.e. 19 attributes - 2 numeric (*age* and *height*) and 17 nominal (*marital status, have children, education level, occupation industry, occupation level, body type, eye color, hair color, smoker, drink, diet, ethnic background, religion, want children, politics, personality and have pets*). For simplicity we only considered people from Sydney and interactions between people from different genders.

Implicit User Preferences We learn the implicit user preferences from the user interaction data by applying a Bayesian classification method; an overview of data mining methods for recommender systems is provided in [43].

The implicit user preferences of a user U are represented by a binary classifier which captures U's likes and dislikes. It is trained on U's previous successful and unsuccessful interactions. The training data consists of all users $U+$ with whom U had successful interactions and all users $U-$ with who U had unsuccessful interactions during a given time period. Each user from $U+$ and $U-$ is one training example; it is represented as a vector of user profile attribute values and labeled as either Success (successful interaction with U) or Failure (unsuccessful interaction with U). We used the same 19 user profile attributes as the explicit user preferences listed in the previous section. Given a new instance, user U_{new}, the classifier predicts how successful the interaction between U and U_{new} will be (class *Success* or *Failure*).

As a classifier we employed NBTree [44] which is a hybrid classifier combining decision tree and Naïve Bayes. As in decision trees, each node of a NBTree corresponds to a test for the value of a single attribute. Unlike decision trees, the leaves of a NBTree are Naïve Bayes classifiers instead of class labels. NBTree was shown to be more accurate than both decision trees and Naïve Bayes, while preserving the interpretability of the two classifiers, i.e. providing an easy to understand output which can be presented to the user [44].

4.2.1 Are Explicit Preferences Good Predictors of User Interactions?

Data To evaluate the predictive power of the explicit preferences we consider users who have sent or received at least 1 EOI during a one-month period (March 2010). We further restrict this subset to users who reside in Sydney to simplify the dataset. These two requirements are satisfied by 8012 users (called *target users*) who had 115,868 interactions, of which 46,607 (40%) were successful and 69,621 (60%)

Table 4 Matching U's explicit preferences with U_{int}'s profile for numeric attributes

U's preference for *height*	155–175	155–175	155–175
U's value in profile for *height*	160	180	Unspecified
Matching outcome (U, U_{int})	Match	Non-match	Match

Table 5 Matching U's explicit preferences with U_{int}'s profile for nominal attributes

U's preference for *body type*	Slim, average	Slim, average	Slim, average	Slim, average
U_{int}'s value in profile for *body type*	Slim	Average	Overweight	unspecified
Matching outcome (U, U_{int})	Match	Match	Non-match	Match

were unsuccessful. Each target user U has a set of interacted users U_{int}, consisting of the users U had interacted with.

Method We compare the explicit preferences of each target user U with the profile of the users in U_{int} by calculating the number of matching and non-matching attributes.

While users can specify only a single value for a given attribute in their profile, e.g. *height* = *170* or *body* = *athletic*, more than one value can be specified in their preferences—a set of values for a nominal attribute, e.g. *body* = *slim* or *athletic*, and a range of values for a numeric attribute, e.g. height = 155–175. Taking this into account, the matching between the preferences of U and the profile of U_{int} for a given attribute is done as follows. For a numeric attribute, U_{int} matches U's preferences if U_{int}'s value falls within U's range or U_{int} has not specified a value (see the examples in Table 4). For a nominal attribute, U_{int} matches U's preferences if U_{int}'s value has been included in the set of values specified by U or U_{int} has not specified a value (see the examples in Table 5). An attribute is not considered if U has not specified a value for it. The preferences of U_{int} match the profile of U if all attributes match; otherwise, they do not match.

Results As shown in Table 6, 59.40% of all interactions occur between users with non-matching preferences and profiles. A further examination of the successful and unsuccessful interactions reveals that:

- In 61.86% of all successful interactions U's explicit preferences did not match U_{int}'s profile.
- In 42.25% of all unsuccessful interactions U's explicit preferences matched the U_{int}'s profile.

Suppose that we use the matching of the user profiles and preferences to try to predict if an interaction between two users will be successful or not (if the profile and preferences match -> successful interaction; if the profile and preferences do not match -> unsuccessful interaction). The accuracy will be 49.43% (17,775+39,998 /115,868). This is lower than the baseline accuracy of always predicting the majority class (ZeroR baseline) which is 59.78%. A closer examination of the

Table 6 Explicit preferences—results

	U's preferences and U_{int}'s profile match	U's preferences and U_{int}'s profile do not match	Total
Successful interactions	17,775 (38.14%)	28,832 (61.86%) (false positives)	46,607 (all successful interactions)
Unsuccessful interactions	29,263 (42.25%) (false negatives)	39,998 (57.75%)	69,261 (all unsuccessful interactions)
Total	47,038 (40.60%)	68,830 (59.40%)	115,858 (all interactions)

misclassifications shows that the proportion of false positives is higher than the proportion of false negatives, although the absolute numbers are very similar.

In summary, the results show that the explicit preferences are not a good predictor of the success of interaction between users. This is consistent with [18].

4.2.2 Are Implicit Preferences Good Predictors of User Interactions?

Data To evaluate the predictive power of the implicit preferences we consider users who have at least 3 successful and 3 unsuccessful interactions during a one-month period (February 2010). This dataset was chosen so that we could test on the March dataset used in the study of the implicit preferences above. Here too, we restrict this subset to users who reside in Sydney. These two requirements are satisfied by 3881 users, called *target users*. The training data consists of the interactions of the target users during February; 113,170 interactions in total, 30,215 positive and 72,995 negative. The test data consists of the interactions of the target users during Match; 95,777 interactions in total, 34,958 positive (37%, slightly less than the 40% in the study above) and 60,819 negative (63%, slightly more than the 60% in the study above). Each target user U has a set of *interacted users* U_{int}, consisting of the users U had interacted with.

Method For each target user U we create a classifier by training on U's successful and unsuccessful interactions from February as described in Sect. 3.2. We then test the classifier on U's March interactions. This separation ensures that we are not training and testing on the same interactions.

Results Table 7 summarizes the classification performance of the NBTree classifier on the test data. It obtained an accuracy of 82.29%, considerably higher than the ZeroR baseline of 63.50% and the accuracy of the explicit preferences classifier. In comparison to the explicit preferences, the false positives drop from 61.86% to 30.14%, an important improvement in this domain since recommendations leading to rejection are very discouraging; the false negatives drop from 42.25 to 9.97%.

Table 7 Classification performance of NBTree on test set

<– classified as	Successful interactions	Unsuccessful interactions	Total
Successful interactions	24,060 (68.83%)	10,538 (30.14%) (false positives)	34,958 (all successful interactions)
Unsuccessful interactions	6064 (9.97%) (false negatives)	54,755 (90.03%)	60,819 (all unsuccessful interactions)

In summary, the results show that the implicit preferences are a very good predictor of the success of user interactions, and considerably more accurate than the explicit preferences.

5 Conclusions

People-to-people reciprocal recommenders are an important class of recommenders which have emerged fairly recently. In this paper we discussed their characteristics (a more comprehensive analysis is available in [1]) and present an overview of recent work in this area. To illustrate different aspects of this type of recommenders and how to take account of the reciprocity and build an effective reciprocal recommender, we presented a case study in online dating, using a large dataset from a major Australian online dating website.

In particular, we presented a case study using CCR, a reciprocal recommender system for an online dating we have developed, that combines content-based and collaborative filtering, and utilises data from both user profiles and user interactions. It is based on our finding that people with similar profiles are reciprocally liked by people with similar profiles. CCR achieved success rate of 64.24–69.26% for different number of EOI, significantly outperforming the baseline success rate of 23.44–35.19%. An important advantage of CCR is that it addresses the cold start problem of new users joining the website by being able to provide recommendations immediately, based on the profile of the new user, which is very important for engaging the new users.

We also studied the differences between the implicit and explicit user preferences. We found that the explicit user preferences, stated by the user, are not a good predictor of the success of user interactions, achieving an accuracy of 49.43%. In contrast, the implicit user preferences, that are learned from successful and unsuccessful previous user interactions, using a probabilistic classifier, were a very good predictor of the success of user interactions, achieving an accuracy of 89.29%.

6 Challenges and Future Directions

There are many research questions that arise from designing reciprocal recommenders, some of which are the same as for standard recommenders, and others are inherent to the reciprocity aspect.

Popular Users In reciprocal recommenders, some user profiles need to be handled with care, e.g. popular users should not be recommended too often, as they are likely to be overwhelmed and unresponsive. This problem does not normally occur in non-reciprocal domains or even people-to-people recommenders that are not reciprocal, e.g. Twitter.

Bait-Profiles Another issue is that some users, especially popular users, may hide bait-profiles, created by criminals to lure people into trusting them in romance scams. The detection of scamming in the online dating industry is a high priority and requires the recommender systems to ensure they do not favour bait-profiles over authentic user profiles [45]. Although this issue is very important in online dating, where people are particularly vulnerable when seeking relationships, it can also be an issue in other people-to-people recommendations.

Explicit and Implicit Preferences The predictive power of explicit and implicit user preferences needs further investigation. Not all explicit user preferences are equally important; if the user can specify the importance of the attributes in the explicit preferences, this information can be used to improve the prediction of successful and unsuccessful interactions. A comparison of the explicit and implicit user preferences would also be beneficial, e.g. (1) to find if there are some latent factors that are difficult to capture, and also (2) to make users aware when their stated explicit preferences are very different than their implicit preferences, and adjust the explicit preferences accordingly. It is also worth investigating if our findings about the explicit and implicit preferences carry over to other people-to-people reciprocal domains.

Data Sources and Information Fusion In order to increase the efficiency and relevance of reciprocal recommenders, a number of other data sources should also be explored—for instance, the use of temporal information (e.g. how quickly users respond to EOIs), or the use of photos and free text to refine the quality of the implicit user profiles. Analysing the importance of each data source and developing appropriate aggregation and weighting methods [46] is another avenue for future work.

Novelty and Diversity Although providing unexpected recommendations has been identified as a useful property of traditional recommender systems (Chapter "Novelty and Diversity in Recommender Systems"), it is not clear how much novelty and serendipity is needed in reciprocal recommenders. In contrast with traditional domains, in reciprocal domains, users provide more information about themselves in their user profiles and explicit user preferences. Recommending surprising matches that do not satisfy these preferences may be seen as unacceptable

by some users, and reduce their trust [47] in the system. Some other users, however, may welcome suggestions of people different to the ones they think they like. One way to safely allow novelty and serendipity is to explicitly inform the users when the recommendations deviate from their explicit preferences.

User Personality Considering user personality (Chapter "Personality and Recommender Systems") in reciprocal recommender systems is another interesting direction for future work. Some online dating website assess personality by asking users to complete long and intrusive questionnaires, and then match users based on personality type. It will be useful to acquire user personality implicitly in a non-obtrusive way [48], e.g. from free text comments in the user profile; book, movie and sport preferences; writing style, text sentiment, punctuation and grammar; user activity level and interactions.

Context-Based Recommendations User preferences depend on the context and also change over time. For example, when on holidays, travelling abroad, the user may want to connect with other travellers to explore the area together, and these other people may have different profiles than the ones the user typically connects with. Context-aware recommender approaches (Chapter "Context-Aware Recommender Systems: From Foundations to Recent Developments") used in non-reciprocal domains can be extended and applied for reciprocal tasks.

Multi-Stakeholder Recommendations In people-to-people reciprocal recommenders there are often conflicting incentives. For example, in online dating the best experience for the user is to find a lifelong spouse quickly and leave the dating platform permanently, while for the owners of the online dating platform it might be better to have longer staying or repeated users to maximize revenue. The emerging sub-area of multi-stakeholder recommenders [49] is concerned with integrating the perspectives of multiple stakeholders, not only the users of the system.

Providing Explanation Investigating methods for effective explanations in reciprocal recommenders, and how to balance privacy and transparancy (Chapter "Beyond Explaining Single Item Recommendations"), is another direction for future work. First, in reciprocal recommenders it is more challenging to provide explanations without raising privacy concerns. For examples, it is not appropriate to tell Bob that Alice was recommended to him because she liked another user Daniel who is similar to Bob. Second, methods for providing *reciprocal explanations* (why both parties are expected to benefit from the match) were recently introduced by Kleinerman et al. [34, 50], who also conducted an extensive evaluation in both simulated and operational online dating environment. The results showed that the best explanation method depends on the user—in environments where the cost (e.g. emotional cost—fear of rejection) is high, reciprocal explanations are superior to the traditional one-sided as they make the user more confident in the positive outcome. However, when the fear of rejection was removed in the simulation, the one-sided explanation was better. These results are interesting and open avenues for future research.

Acknowledgments This work was supported by the Smart Services Cooperative Research Centre. We also thank Joshua Akehurst, Luiz Pizzato and Judy Kay for their contributions to this work.

References

1. L. Pizzato, T. Rej, J. Akehurst, I. Koprinska, K. Yacef, and J. Kay, Recommending people to people: the nature of reciprocal recommenders with a case study in online dating. User Model. User-Adap. Interact. **23**, 447–488 (2013)
2. I. Palomares, C. Porcel, L. Pizzato, I. Guy, E. Herrera-Viedma, Reciprocal recommender systems: analysis of state-of-art literature, chalelnges and opportunities on social recommendation (2020). https://arxiv.org/abs/2007.16120
3. L. Terveen, D.W. McDonald, Social matching: a framework and research agenda. ACM Trans. Comput.-Human Interact. **12**,401–434 (2005)
4. I. Guy, People recommendation on social media, in ed. by P. Brusilovsky, D. He, *Social Information Access*. Lecture Notes in Computer Science, vol 10100 (Springer, Berlin, 2018)
5. J. Chen, W. Geyer, C. Dugan, M.G.I. Muller, I. Guy, Make new friends, but keep the old: recommending people on social networking sites, in *Proceedings of the International Conference on Computer-Human Interaction (CHI)* (2009)
6. I. Guy, S. Ur, I. Ronen, A. Perer, M. Jacovi, Do you want to know?: Recommending strangers in the enterprise, in *Proceedings of the ACM Conference on Computer Supported Cooperative Work (CSCW)* (2011)
7. M. Fazel-Zarandi, H.J. Devlin, Y. Huang, N. Contractor, Expert recommendation based on social drivers, social network analysis, and semantic data representation, in *Proceedings of the Second International Workshop on Information Heterogeneity and Fusion in Recommender Systems (HetRec)* (ACM, New York, 2011), pp. 41–48
8. S. Bull, J.E. Greer, G.I. McCalla, L. Kettel, J. Bowes, User modelling in I-help: what, why, when and how, in *Proceedings of the International Conference on User Modeling* (2001), pp. 117–126
9. J. Greer, G. McCalla, J. Collins, V. Kumar, P. Meagher, J. Vassileva, Supporting peer help and collaboration in distributed workplace environments. Int. J. Artif. Intell. Edu. **9**, 159–177 (1998)
10. C.-T. Li, Mentor-spotting: Recommending expert mentors to mentees for live trouble-shooting in codementor, in *Knowledge and Information Systems*, vol. 61 (Springer, Berlin, 2020), pp. 799–820
11. H. Labarthe, F. Bouchet, R. Bachelet, K. Yacef, Does a peer recommender foster students' engagement in MOOCs?, in *Proceedings of the International Conference in Educational Data Mining (EDM)* (2016), pp. 418–423
12. H. Labarthe, R. Bachelet, F. Bouchet, K. Yacef, Increasing MOOC completion rates through social interactions: A recommendation system, in *Proceedings of EMOOCS Fourth European MOOCS Stakeholders Summit, Graz, Austria*, (2016), pp. 471–480
13. F. Bouchet, H. Labarthe, K. Yacef, R. Bachelet, Comparing peer recommendation strategies in a MOOC, in *Proceedings of the 25th Conference on User Modeling, Adaptation and Personalization (UMAP)*, (2017), pp. 418–423
14. S. Prabhakar, G. Spanakis, O. Zaiane, Reciprocal recommender system for learners in massive open online courses (MOOCs), in *Proceedings of Advances in Web-Based Learning (ICWL)* (2017), pp. 157–167
15. J. Malinowski, T. Keim, O. Wendt, T. Weitzel, Matching people and jobs: A bilateral recommendation approach, in *Proceedings of the 39th Annual Hawaii International Conference on System Sciences* (2006)
16. W. Hong, L. Lei, T. Li, W. Pan, iHR: An online recruiting system for xiamen talent service center, in *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)* (2013), pp.1177–1185

17. L. Brozovsky, V. Petricek, Recommender system for online dating service, in *Proceedings of Znalosti Conference*, Ostrava (2007)
18. F. Diaz, D. Metzler, S. Amer-Yahia, Relevance and ranking in online dating systems, in *Proceedings of the 33rd international ACM Conference on Research and Development in Information Retrieval (SIGIR)* (2010)
19. B. McFee, G.R.G. Lanckriet, Metric learning to rank, in *Proceedings of the International Conference on Machine Learning (ICML)* (2010)
20. L. Pizzato, T. Rej, T. Chung, I. Koprinska, J. Kay, RECON: A reciprocal recommender for online dating, in *Proceedings of the ACM Conference on Recommender Systems (RecSys)*, Barcelona (2010)
21. J. Akehurst, I. Koprinska, K. Yacef, L. Pizzato, J. Kay, T. Rej, CCR: A content-collaborative reciprocal recommender for online dating, in *Proceedings of the 22nd International Joint Conference on Artificial Intelligence (IJCAI)*, Barcelona (2011)
22. Y.S. Kim, A. Mahidadia, P. Compton, X. Cai, M. Bain, A. Krzywicki, W. Wobcke, People recommendation based on aggregated bidirectional intentions in social network site, in *Proceedings of the Pacific Rim Knowledge Acquisition Workshop (PKAW)* (Springer, Berlin, 2010), pp. 247–260
23. X. Cai, M. Bain, A. Krzywicki, W. Wobcke, Y.S. Kim, P. Compton, A. Mahidadia, Collaborative filtering for people to people recommendation in social networks, in *Proceedings of the Australasian Joint Conference on Artificial Intelligence (AI)* (Springer, Berlin, 2010), pp. 476–485
24. S. Kutty, L. Chen, R. Nayak, A people-to-people recommendation system using tensor space models, in *Proceedings of the 27th Annual ACM Symposium on Applied Computing (SAC)* (2012)
25. P. Xia, B. Liu, Y. Sun, C. Chen, Reciprocal recommendation system for online dating, in *Proceedings of the IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)* (2015), pp. 234–241
26. P. Xia, B. Liu, Y. Sun, C. Chen, User recommendations in reciprocal and bipartite social networks-an online dating case study. IEEE Intell. Syst. **29**, 27–35 (2014)
27. S. Alsaleh, R. Nayak, Y. Xu, L. Chen, Improving matching process in social network using implicit and explicit user information, in *Proceedings of the 13th Asia-Pacific Conference on Web Technologies and Applications (APWeb)* (2011), pp. 313–320
28. S. Kutty, L. Chen, R. Nayak, A people-to-people recommendation system using tensor space model, in *Proceedings of the 27th Annual ACM Symposium on Applied Computing* (2012), pp. 187–192
29. S. Kutty, R. Nayak, L. Chen, A people-to-people matching system using graph mining techniques. World Wide Web **17**, 311–349 (2014)
30. A. Alanazi, M. Bain, A people-to-people content-based reciprocal recommender using hidden Markov models, in *Proceedings of the 7th ACM Conference on Recommender Systems, (RecSys)* (2013)
31. A. Alanazi, M. Bain, A scalable people-to-people hybrid reciprocal recommender using hidden Markov models, in *Proceedings of the Second Workshop on Machine Learning Methods for Recommender Systems, in conjunction with SIAM International Conference on Data Mining* (2016)
32. F. Vitale, N. Parotsidis, C. Gentile, Online reciprocal recommendation with theoretical performance guarantee, in *Proceedings of at the 32nd Conference on Neural Information Processing Systems (NeurIPS)* (2018)
33. A. Kleinerman, A. Rosenfeld, F. Ricci, S. Kraus, Optimally balancing receiver and recommended users' importance in reciprocal recommender systems, in *Proceedings of the 12th ACM Conference on Recommender Systems (RecSys)* (2018), pp. 131–139
34. A. Kleinerman, A. Rosenfeld, F. Ricci, S. Kraus, Supporting users in finding successful matches in reciprocal recommender systems, in *User Modeling and User-Adapted Interaction* (Springer, Berlin, 2020)

35. J. Neve, I. Palomares, Latent factor models and aggregation operations for collaborative filtering in reciprocal recommender systems, in *Proceedings of the 13th ACM Conference on Recommender Systems (RecSys)* (2019), pp. 219–227
36. J. Neve, I. Palomares, Aggregation strategies in user-to-user reciprocal recommender systems, in *Proceedings of the International Conference on Systems, Man and Cybernetics (SMC)* (2019), pp. 4031–4036
37. R. Ramanathan, N.K. Shinada, S.K. Palanianppan, Building a reciprocal recommenderation system at scale from scratch: Learning from one of Japan's prominent dating applications, in *Proceedings of the 14th ACM Conference on Recommender Systems (RecSys)* (2020), pp. 566–567
38. J. Neve, R. McConville, ImRec: Learning reciprocal preferences using images, in *Proceedings of the 14th ACM Conference on Recommender Systems (RecSys)* (2020), pp. 170–179
39. Statista (Online dating in the United States - Statistics and Facts), 2019, https://www.statista.com/topics/2158/online-dating/
40. M.C. Rosenfeld, R.J. Thomas, S. Hausen, Disintermediating your friends: How online dating in the united states displaces other ways of meeting. Proc. Nat. Acad. Sci. **116**(36), 17753–17758 (2019)
41. J. Akehurst, I. Koprinska, K. Yacef, L. Pizzato, J. Kay, T. Rej, Explicit and implicit user preferences in online dating, in ed. by L. Cao, J. Huang, J. Bailey, Y. Koh, J. Luo, *New Frontiers in Applied Data Mining*. Springer Lecture Notes in Computer Science, vol. 7104 (Springer, Berlin, 2012), pp. 15–27
42. L. Pizzato, T. Chung, T. Rej, I. Koprinska, K. Yacef, J. Kay, Learning user preferences in online dating, in *Proceedings of the Preference Learning (PL-10) Workshop, European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML PKDD)* (2010)
43. X. Amatriain, A. Jaimes, N. Oliver, J.M. Pujol, Data mining methods for recommender systems, in ed. by F. Ricci, L. Rokach, B. Shapira, *Recommender Systems Handbook*, 2nd edn. (Springer, Berlin, 2015), pp 39–72
44. R. Kohavi, Scaling up the accuracy of Naive-Bayes classifiers: A decision-tree hybrid, in *Proceedings of the International Conference on Knowledge Discovery in Databases (KDD)* (1996), pp. 202–207
45. L. Pizzato, J. Akehurst, C. Silvestrini, K. Yacef, I. Koprinska, J. Kay, The effect of suspicious profiles on people recommenders, in *Proceedings of the 20th Conference on User Modeling, Adaptation, and Personalization (UMAP)*. Lecture Notes in Computer Science, vol. 7379 (Springer, Berlin, 2012), pp. 225–236
46. G. Beliakov, T. Calvo, S. James, Aggregation of preferences in recommender systems, in ed. by F. Ricci, L. Rokach, B. Shapira, *Recommender Systems Handbook*, 2nd edn. (Springer, Berlin, 2015), pp 705–734
47. P. Victor, M. De Cock, C. Cornelis, Trust and recommendations, in ed. by F. Ricci, L. Rokach, B. Shapira, *Recommender Systems Handbook*, 2nd edn. (Springer, Berlin, 2015), pp 645–676
48. S. Berkovsky, R. Taib, I. Koprinska, E. Wang, Y. Zeng, J. Li, S. Kleitman, Detecting personality traits using eye-tracking data, in *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems (CHI)* (2019), pp. 1–12
49. H. Abdollahpouri, G. Adomavicius, R. Burke, I. Guy, D. Jannach, T. Kamishima, J. Krasnodebski, L. Pizzato, Multistakeholder recommendation: survey and research directions. User Model. User-Adap. Interac. **30**, 127–158 (2020)
50. A. Kleinerman, A. Rosenfeld, S. Kraus, Providing explanations foor recomemndations in reciprocal environments, in *Proceedings of the 12th ACM Conference on Recommender Systems (RecSys)* (2018), pp. 22–30