MEET: A Generalized Framework for Reciprocal Recommender Systems

Lei Li
School of Computing and Information Sciences
Florida International University
Miami, FL 33199
lli003@cs.fiu.edu

Tao Li
School of Computing and Information Sciences
Florida International University
Miami, FL 33199
taoli@cs.fiu.edu

ABSTRACT

Reciprocal recommender systems refer to systems from which users can obtain recommendations of other individuals by satisfying preferences of both parties being involved. Different from the traditional user×item recommendation, reciprocal recommenders focus on the preferences of both parties simultaneously, as well as some special properties in terms of "reciprocal". In this paper, we propose MEET – a generalized fra*M*ework for r*E*ciprocal r*E*commenda*T*ion, in which we model the correlations of users as a bipartite graph that maintains both *local* and *global* "reciprocal" utilities. The *local* utility captures users' mutual preferences, whereas the *global* utility manages the overall quality of the entire reciprocal network. Extensive empirical evaluation on two real-world data sets (online dating and online recruiting) demonstrates the effectiveness of our proposed framework compared with existing recommendation algorithms. Our analysis also provides deep insights into the special aspects of reciprocal recommenders that differentiate them from user×item recommender systems.

Categories and Subject Descriptors: H.3.3[Information Search and Retrieval]: Information filtering

General Terms: Algorithms, Design, Experimentation

Keywords: Reciprocal Recommender, Global and Local Regularization, Bipartite Graph

1. INTRODUCTION

Recommender systems have been intensively studied in both academia and industry, following a paradigm of providing a target user with a list of items that the user might prefer, i.e., the user×item recommendation. In recent years, a special class of recommender systems – reciprocal recommenders – have emerged, and are tailored for applications that focus on recommending people to people, in which the preferences of both parties involved in the recommendation need to be satisfied. For instance, in an online recruiting system, a job seeker would search jobs that match his/her preference, e.g., the special skills and the salary; and a recruiter might seek suitable candidates to fulfil the job requirement. Other illustrative examples of reciprocal recommenders include online dating services, online mentoring systems, customer-to-customer marketplaces, etc.

Obviously, the major challenge of reciprocal recommender systems is how to satisfy the needs of both users in a recommended match. This requires modeling *the bilateral relations between users* by considering the double-sided preferences. However, simply considering the bilateral relations is *insufficient* in the reciprocal community. In practice, reciprocal recommenders, such as online dating and online recruiting, possess special characteristics differentiating them from traditional user×item recommender systems. Figure 1 illustrates some challenges in these systems. We summarize the challenges as follows:

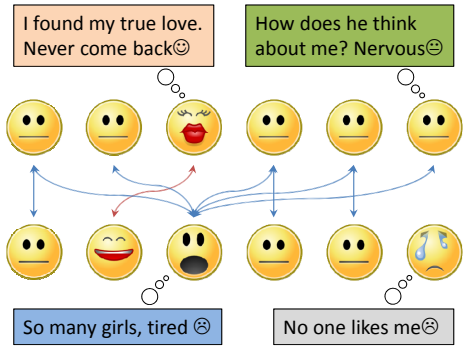


Figure 1: A toy example in an online dating system.

- *Reciprocity*: The success of a match *depends on the double-sided preference*, not solely on the user who receives the recommendation. This is the key feature of a reciprocal recommender.
- *Limitedness*: In traditional recommenders, an item can be preferred by a great amount of users. However, in reciprocal recommenders, people have *limited availability* towards other people, e.g., a boy cannot date with ten girls simultaneously.
- *Passiveness*: In reciprocal recommenders, a lot of users with limited engagement activities passively receive messages from other users. To maintain a vibrant community and further attract more users, it is imperative to consider the *passiveness* of users.

CIKM'12, October 29–November 2, 2012, Maui, HI, USA.

- *Sparsity*: Users in reciprocal communities would probably not return to the system if they find their preference. Therefore, different from traditional recommenders where users often have a long consumption history, the *data sparsity* issue in a reciprocal recommender needs more careful consideration.

The aforementioned challenges are essential to a successful reciprocal recommender system. Previous studies either focus on handling the main reciprocity of the recommender, or delve into a specific issue that exists in reciprocal recommenders. Little research work has been proposed to address the challenges in a principled and unified manner. In our work, we model the bilateral relations of users as a bipartite graph that maintains both *local* and *global* utilities in a reciprocal community. The *local* utility captures users' mutual preferences by considering *reciprocity*, *limitedness* and *passiveness*, whereas the *global* utility manages the overall quality of the entire reciprocal network in order to resolve the *sparsity* problem. The bipartite graph is constructed based on the users' self-descriptive and preference features, and then is refined by users' interactive activities.

In summary, the contribution of our work is three-fold:

1. We propose a *generalized* reciprocal recommendation framework, in which various challenges (e.g., *reciprocity*, *limitedness*, *passiveness* and *sparsity*) in reciprocal recommenders can be effectively addressed;

2. We define a novel quality metric (Set Vitality) to *intentionally evaluate the vitality of the recommended list* by considering the interaction histories of users;

3. We perform empirical evaluation on two real-world reciprocal data sets to demonstrate the effectiveness of our proposed framework.

The rest of the paper is organized as follows. Section 2 presents a brief summary of prior work relevant to reciprocal recommenders. In Section 3, we formalize the problem. Section 4 presents an overview of the generalized framework. In Section 5, we describe the feature space generally used in reciprocal recommenders. We then present the algorithmic details of our framework in Section 6, and discuss the generalization in Section 7. Empirical evaluation of our method compared with different baselines is reported in Section 8. Finally Section 9 concludes the paper.

2. RELATED WORK

Reciprocal recommenders provide platforms for people to achieve a win-win situation. A couple of interesting methods have been proposed to address issues of reciprocal recommendation from both mathematical and practical perspectives. In the following, we highlight the previous research that are most related to our work.

In mathematics, reciprocal recommendation can be modeled as the well-known stable marriage problem [10], which aims at finding an optimal matching between two sets of elements given a set of preferences for each element. Similar problems include stable roommates and college admissions problem. In our problem setting, we do not require to provide an optimal matching for any pair of users, as we need to consider other properties of a reciprocal recommender, e.g., to enhance the vitality of the entire reciprocal network.

In practice, reciprocity in a recommender system refers to the bilateral relations between people. This reciprocity is often considered in social matching in a broader sense [19]. Several existing methods [2, 13] focus on people-to-people recommendation in social networks, in which people have a dual role as both "users" and "items". Although the reciprocity is emphasized in these methods, they failed to consider other important properties of a reciprocal recommender, as we discussed in Section 1.

Pizzato *et al.* analyzed the characteristics of reciprocal recommenders in detail [15], by providing a comparison of the reciprocal recommender against traditional recommender systems. However, they did not address all the concerns of the reciprocal recommender in a unified way. In their follow-up works [1, 16, 17], they focused on how to capture the bilateral preference of users and presented different strategies to model the reciprocity between users in an online dating environment. Beside the online dating, there are a couple of works focusing on other reciprocal domains, e.g., online recruiting [21] and expertise management [18, 12]. Unfortunately, the special issues of reciprocal recommendation are not properly handled in previous works. These issues render the recommender not effective (as demonstrated in the experiment). Different from these previous works, in our study we shed light on not only the bilateral correlations between users, but also the *passiveness* and *limitedness* of users, and the *sparsity* of the reciprocal network, and provide a unified solution for reciprocal recommenders.

Reciprocity has also been investigated in Information Retrieval (IR) community. In [6] the problem of match-making is modeled as the retrieval and ranking of candidate matches for a given user. To this end, a series of features are initially extracted from user profiles, and then the correlation between the queries and the candidate is formalized in a feature level. The ranking function is learned via gradient boosted decision tree [9] over the extracted features. Our problem setting is more general than theirs since the search can be regarded as an explicit recommendation process.

3. PROBLEM STATEMENT

In this section we formally state our problem. We consider the reciprocal recommendation scenario consisting of two sets of users $\mathcal{U}$ and $\mathcal{V}$, in which each user maintains a self-description, $\mathcal{F}^s$, about himself/herself and a preference, $\mathcal{F}^p$, towards users in the other set. For example, in an online dating system, the two sets of users correspond to men and women; in an online recruiting system, they correspond to recruiters and job seekers. $\mathcal{F}^s$ and $\mathcal{F}^p$ are composed of a set of descriptive attributes, which might be scalar, categorical or free texts. Given a user $u \in \mathcal{U}$, our goal is to recommend u a list of users $\mathcal{Q}_u \subset \mathcal{V}$ such that $\mathcal{F}^p_u$ of u and $\mathcal{F}^p_v$ of any $v \in \mathcal{Q}_u$ are both satisfied, i.e., the *reciprocity*.

Definition 1. An Optimal Match: *Given two users $u \in \mathcal{U}$ and $v \in \mathcal{V}$, the pair (u,v) is said to be an optimal match* iff *$\mathcal{F}^s_u$ maximally satisfies $\mathcal{F}^p_v$ and $\mathcal{F}^s_v$ maximally satisfies $\mathcal{F}^p_u$.*

Ideally, we expect that all the recommended pairs (u, v) are optimal matches. However in practice, the optimality cannot be fully achieved. In general, the capacity of a user receiving and responding recommendations in the scenario of reciprocal recommenders is quite dissimilar to the one of traditional user×item recommenders. For example, a boy

cannot date with ten girls simultaneously. Therefore, in reciprocal recommenders, it is imperative to consider the limited capacity of users, i.e., the *limitedness*.

DEFINITION 2. AVAILABILITY: *Given a budget b, a user $u \in \mathcal{U}$ is said to be available if the number of recommendations that u receives is less than b within a time range t. We use $\tau(u)$ to indicate the number of times that u has been recommended to other users within t, e.g., one week.*

DEFINITION 3. A SUCCESSFUL MATCH: *Given two users $u \in \mathcal{U}$ and $v \in \mathcal{V}$, the pair (u,v) is said to be a successful match* iff *$\mathcal{F}_u^s$ suboptimally satisfies $\mathcal{F}_v^p$ and $\mathcal{F}_v^s$ suboptimally satisfies $\mathcal{F}_u^p$,* w.r.t. *the availability of u and v. We denote a successful match as $u \cong_b v$ or $v \cong_b u$.*

A user-oriented recommender should be capable of getting all users involved in an interactive community. In practice, a user might often interact with other users through different channels. For example, in an online recruiting system, a job seeker often sends resumes to recruiters and a recruiter often makes interviews with job seekers. Similarly, in an online dating system, people often send messages to whom they prefer. However, there are still a great number of users that passively receive such interactions from other users, i.e., the *passiveness*, which to some extent violates the reciprocal property of the entire community. Therefore, we need to consider the *passiveness* of a user in order to enrich the vitality of the reciprocal community.

DEFINITION 4. VITALITY: *Given a user u and his/her interactive activities with other users, u is said to be vital if the average number of u's interactions, $avg(|u_{\leftrightarrow}|)$, reaches the average number of interactions within the entire reciprocal community.*

In a reciprocal recommender system, a vital user is possible to improve the engagement of passive users, e.g., by sending messages to a shy boy to ask him for a date. By recommending passive users to vital users and vice versa, the overall vitality of the recommendation community would increase to some extent. It is possible that considering the vitality of users might deviate the definition of a successful match. However, the success of a recommendation is not solely determined by the match. The engagement of users after receiving the recommendation is also an important indicator to a successful recommendation.

We are expected that all the recommendations are successful matches in a reciprocal recommender, by considering both the bilateral preferences of individuals and the vitality of users. We now define our technical problem of reciprocal recommendation as follows.

PROBLEM (Reciprocal Recommendation): *Given a user $u \in \mathcal{U}$ (or $v \in \mathcal{V}$) and a budget b (the availability), recommend for u (or v) a list of users $\mathcal{Q}_u \subset \mathcal{V}$ (or $\mathcal{Q}_v \subset \mathcal{U}$), such that $\forall\ v \in \mathcal{Q}_u$ (or $u \in \mathcal{Q}_u$), $u \cong_b v$ (or $v \cong_b u$), and also the vitality of u (or v) should have potential to increase.*

The essence of reciprocal recommendation, including *reciprocity*, *limitedness* and *passiveness*, has been well captured in the above problem formation. We will discuss our solution to the problem of *reciprocal recommendation* in Section 6.

4. FRAMEWORK OVERVIEW

Figure 2 shows an overview of our proposed reciprocal recommendation framework, *MEET*. It consists of three interrelated components:

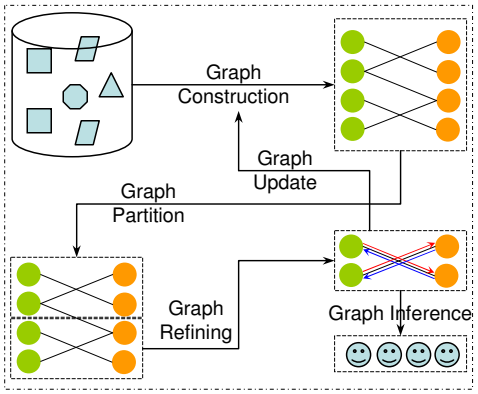


Figure 2: An overview of *MEET*.

Graph Construction and Updating (Section 5 and 6.3): The reciprocal community is represented as a bipartite graph based on the bilateral preference of users. *MEET* automatically analyzes the feature space of users and calculates the relevance between users from different sets. In addition, *MEET* treats the profiles of users who have successfully obtained their preferred information (e.g., jobs or friends) as the basis to learn the weights of features, and consequently improves the quality of the generated bipartite graph.

Bipartite Graph Partition (Section 6.1): To provide reasonable results and expedite the recommendation process, *MEET* first partitions the bipartite graph into different pieces, and then performs the recommendation on subgraphs. The subgraphs obtained by graph partition contain more specialized structures of the community, and therefore provide an elegant base for further inference.

Recommendation Inference (Section 6.2): Besides the relevance between users, *MEET* also considers the interactive activities within the community, and refines the subgraph based on such information. In each refined bipartite subgraph, the recommendation problem is modeled as an optimization problem, and the result is obtained by performing the inference over different sets of users.

5. BILATERAL FEATURES

In our problem setting, the profile of a user is composed of two types of features: a self-description, $\mathcal{F}^s$, and a preference, $\mathcal{F}^p$. Given a user $u \in \mathcal{U}$, we are interested in finding a relevant user $v \in \mathcal{V}$, such that (u,v) is a successful match. In Definition 3, we relax the condition of the perfect reciprocal match and allow partial matches of the users' preferences. By *relevance*, we mean that $\mathcal{F}_v^s$ matches $\mathcal{F}_u^p$, and meanwhile, $\mathcal{F}_u^s$ matches $\mathcal{F}_v^p$. Therefore, the *relevance* includes two components, $rel(u \sim v)$ and $rel(v \sim u)$. Here, the *relevance* cannot be regarded as the affinity between users, but a quantitative measure that describes how matchable the two users from different user sets are. Both $rel(u \sim v)$ and $rel(v \sim u)$ are calculated using the same set of attribute-value pairs for any user pair, and hence the *relevance* is comparable.

To calculate the relevance, we consider three groups of features that we believe have the predictive power of rele-

Table 1: Bilateral features in an online dating system. Features with the prefix "match" are all the preference features, whereas the others are the self-description features.

scalar		categorical					free text	
birth_year	match_min_age	sex	dating_status	marriage	avatar	match_avatar	self_character	
height	match_max_age	location	valid_mobile	children	match_certified	match_location	match_character	
income	match_min_height	sublocation	house_status	nation	match_marriage	match_sublocation		
	match_max_height	education	auto_status	belief	match_education			
		industry	honest_level	access_photo	match_edu_more			

vance, including scalar features, categorical features and free texts. The features used in online dating data set are listed in Table 1. For scalar features, we first categorize them into different ranges, e.g., the length of working experience includes 0-3 years, 3-5 years, 5-10 years and 10- years. We then encode these ranges as binary features, e.g., if a user's working experience falls into the range of 0-3 years, then this feature (0-3 years) would be set to true (1), and all other ranges would be set to false (0). With categorical attributes, we use the same strategy of dealing with working experiences ranges. For free text features, we transform the text into an l_2-normalized TF.IDF-based term vector, and then combine this vector with the vector obtained based on other types of attributes. We can also specify the importance of each attribute. Finally, $rel(u \sim v)$ or $rel(v \sim u)$ is computed using the cosine similarity of the two corresponding vectors.

Once we obtain the relevance scores of both $rel(u \sim v)$ and $rel(v \sim u)$, we integrate these two scores as the final relevance between u and v, described in Eq.(1):

$$rel(u, v) = rel(u \sim v) \cdot rel(v \sim u). \quad (1)$$

We use the product of these two scores instead of the linear combination to eliminate the condition of unilateral preference, e.g., a job position is preferred by an applicant but the applicant is not suitable for this job. In addition, the relevance model is simplified since no parameter is involved.

6. RECOMMENDATION METHODOLOGY

In *MEET*, a reciprocal network is represented by a bipartite graph based on the mutual relevance of users. The recommendation is achieved by first partitioning the graph into multiple subgraphs, and then performing label inference on a specific subgraph with respect to the target user.

6.1 Community Specialization

A reciprocal community involves two different sets of users interacting with each other. It is intuitive to represent the community as a bipartite graph, where each set of users can be denoted as a set of nodes in the graph, and the relevances between users can be described by the edges in the bipartite graph. Note that the relevance quantifies the degree of match between two users, and hence the edges in the graph are not directional. The relevance score is filtered by a threshold θ to reduce the probability of a dense graph. A toy example can be found in Figure 1.

The recommendation can be achieved by analyzing the properties of the bipartite graph and the special characteristics of the target user. However in practice, the number of users involved in such a bilateral community might be large, which renders direct analysis on the entire bipartite graph inefficient. Taking into account the scalability issue, we propose to simultaneously separate the two sets of users in the graph into several groups, such that within each subgraph, the focus of users are more specialized, e.g., users that are all in the community of engineering in an online recruiting system. In this way, the generated subgraph contains less users with more similar preferences.

A simple way of partitioning the data into groups is to categorize the data by the attributes. However, the data considered in our problem setting is not fully structured, with text descriptions involved. Even if the text description can be transformed to quantitative values, the values of other attributes might be missing, which results in the difficulty of attribute selection for partitioning. The way we represent the data as a bipartite graph and use graph-partitioning methods to regroup users helps resolve the issues mentioned above.

Formally, given a bipartite graph $\mathbb{G} = (\mathcal{U}, \mathcal{V}, E, \mathbf{w})$ where $\mathbf{w} : E \to \mathbb{R}$, our goal is to partition the vertex sets $\mathcal{U}$ and $\mathcal{V}$ into k disjoint clusters, i.e., $\mathcal{U} = \{\mathcal{U}_1, \mathcal{U}_2, \cdots, \mathcal{U}_k\}$ and $\mathcal{V} = \{\mathcal{V}_1, \mathcal{V}_2, \cdots, \mathcal{V}_k\}$. Clearly the "best" clustering would correspond to a partitioning of the graph such that the crossing edges between clusters have minimum weight, denoted as *cut*,

$$cut(\mathcal{U}_1 \cup \mathcal{V}_1, \cdots, \mathcal{U}_k \cup \mathcal{V}_k) = \min_{G_1, \cdots, G_k} cut(G_1, \cdots, G_k), \quad (2)$$

where $G_1, \cdots, G_k$ is any k-partitioning of the graph $\mathbb{G}$.

The problem defined in Eq.(2) is essentially a co-clustering problem, which can be solved by many existing methods [5, 20]. In this paper, we will not focus on how to resolve the co-clustering problem. We employ a graph partitioning method [4] as the solution. The method proposes a spectral co-clustering algorithm that uses the second left and right singular vectors of an appropriately scaled user-user matrix to yield good bi-partitions, which is essentially suitable to our partitioning task.

By partitioning, we can obtain a list of subgraphs that embrace specialized information about each small reciprocal community, which can be well explained under different real-world cases. Let us take the online recruiting system as an example. For each subgraph, it may represent different areas of work, e.g., *computer engineering*, *chemical engineering* and *finance*. Such a subgraph characterizes the special properties of the area of work, and therefore provides us a reasonable and effective base to perform reciprocal recommendation.

6.2 Local Recommendation

Given a specialized community represented by a bipartite subgraph, our goal is to recommend for each user an attractive list of users from the other user set. By considering the characteristics of the reciprocal recommendation, we employ the bipartite graph inference to obtain the recommendation result. Besides the relevance between users, we

have the interactive activities, e.g., messages and chatting in online dating, and adding favorite jobs and sending interviews to applicants in online recruiting. We also take into account the availability of the users. We refine the bipartite subgraph based on these information. After refinement, each vertex has its vertex attribute, i.e., the availability, and also has two sets of edges, including the relevance edges and the activity edges. The relevance edge is undirected, and the activity edge is directed, indicating which vertex is the initiator of the activity.

A natural question is why we do not consider the activity information when partitioning the bipartite graph. Within the reciprocal community, not all the users have sufficient activities; if we incorporate the activities into the partitioning process, the generated results might isolate users with few activities from more active users, which may render the recommendation for these inactive users not reasonable.

6.2.1 Bipartite Graph Preliminaries

Formally in our problem setting, a bipartite graph $\mathbb{G}^* = (\mathcal{U}, \mathcal{V}, E_r, E_a, \mathbf{w}_r, \mathbf{w}_a)$ consists of two sets of vertices, $\mathcal{U}$ and $\mathcal{V}$, and two sets of edges, $E_r, E_a \subseteq \mathcal{U} \times \mathcal{V}$. Each edge in E_r is an undirected pair of nodes weighted by $rel(u, v)$, i.e., $\mathbf{w}_r : \mathcal{U} \times \mathcal{V} \to \mathbb{R}_r$. Each edge in E_a is an ordered pair of nodes $[u, v]$ representing the activity connection from u to v, weighted by the ratio of the activities toward the end node and all the activities of the initial node, i.e., $\mathbf{w}_a : \mathcal{U} \times \mathcal{V} \to \mathbb{R}_a$. Given a vertex v in $\mathbb{G}^*$ (either $v \in \mathcal{U}$ or $v \in \mathcal{V}$), the in-degree $p(v)$ and out-degree $q(v)$ are defined as

$$p(v) = \sum_{\{u|[u,v]\in E_a\}} w_a(u, v), \quad q(v) = \sum_{\{u|[v,u]\in E_a\}} w_a(v, u).$$

Let $\mathcal{H}(\mathbb{G}^*)$ denote the space of functions $f : \mathcal{U}, \mathcal{V} \to \mathbb{R}$, which assigns a real value $f(v)$ to each vertex v.

6.2.2 Inference on Bipartite Graph

Given a bipartite graph representing the relevance structure of the reciprocal network, a simple solution to the recommendation is to select top relevant users that directly link to a given user as the recommended result. However, the specific properties (availability and vitality) of users in a reciprocal community would be ignored if we follow such a simple paradigm. For example, a job seeker u has been recommended to multiple job recruiters; if we recommend u to a new job recruiter v, then u will have little chance to respond v, even if they are relevant in some sense. In such a situation, it would be more reasonable to recommend for v other job seekers who are similar to u, which is essentially collaborative filtering. Yet, both directional and non-directional information on the bipartite graph cannot be easily incorporated into traditional collaborative filtering algorithms. A much more natural solution to this problem is to perform graph inference on the bipartite graph to obtain the recommendation list.

Our inference paradigm is motivated by [23], in which the graph inference is performed on a directed bipartite graph to solve the problem of classification. The problem setting in their method is similar to ours. However, they only consider the directional information within the bipartite graph, i.e., the in-degree and out-degree of nodes, but fail to consider the rationality of the connectivity between nodes, i.e., why the two nodes are connected with each other, which is essential in the problem of reciprocal recommendation. In our work, we explicitly model the rationality of the connectivity of nodes as the relevance between users, by which the connectivity can be naturally explained, and the final recommendation result is more reasonable and explainable.

Formally, if two distinct vertices u_1 and u_2 in $\mathcal{U}$ are co-linked by vertex v in $\mathcal{V}$, it indicates that the properties of both u_1 and u_2 are likely to be similar, e.g., both job seekers are similar in profile-wise since they are all preferred by the same job recruiter. The co-linkage strength induced by v between u_1 and u_2 can be measured by

$$c_v(u_1, u_2) = w_r(u_1, v) \cdot w_r(u_2, v) \cdot \frac{w_a(v, u_1) w_a(v, u_2)}{q(v)}. \quad (3)$$

With such a similarity measure, we not only consider the interactive activities of users ($w_a(v, u_1)$ and $w_a(v, u_2)$), but also emphasize the relevance between users ($w_r(u_1, v)$ and $w_r(u_2, v)$). It can be naturally understood in the context of online dating. If two boys are simultaneously contacted by a girl, then it indicates that both boys have similar characteristics that are preferred by the girl. Moreover, the more girls contact both boys, the more significant the similarity. A natural question arising in this context is why the similarity measure is further normalized by out-degree of v. It can be easily interpreted if we use the previous example. A girl who sends messages to a lot of boys may not have clear preference on what boy characteristics, and therefore the induced similarity of two boys by this girl is not significant.

In Eq.(3), we penalize the influence of active users by normalizing the similarity score using $q(v)$. It should be clarified that the vitality of the community cannot be reflected by flooded messages without definite purposes. The way we formalize the similarity considers the importance of dedicated users, e.g., if a girl is interested in a boy, she will not send too many messages to other boys but focus on building the relationship with this boy. Such an observation is beneficial to construct a vibrant reciprocal community.

Let f denote a function defined on one vertex set $\mathcal{U}$. Then the inference cost of function f can be measured by the following functional:

$$\Omega_{\mathcal{U}}(f) = \frac{1}{2} \sum_{u_1, u_2 \in \mathcal{U}} \sum_{v \in \mathcal{V}} \frac{1}{\tau(v)} c_v(u_1, u_2) \left(\frac{f(u_1)}{\sqrt{p(u_1)}} - \frac{f(u_2)}{\sqrt{(p(u_2))}} \right)^2. \quad (4)$$

In Eq.(4), we penalize large differences in function values for vertices in $\mathcal{U}$. Notice that the function values are normalized by in-degrees of the corresponding vertices. In the context of online dating, the explanation is similar to the one given before. Many girls will prefer a handsome and successful man, which does not mean that these girls have similar preferences over the characteristics of the man. However, if two girls are sending messages to a boy without handsome appearance and strong background, it is likely to express a common interest of both girls. We also consider the availability of users using the reverse of $\tau(u)$. If the user has been recommended to other users many times within a time range, e.g., one week, then the possibility of this user being recommended to the target user should be small.

Similarly, the inference cost of function f on the vertex set $\mathcal{V}$ can be measured by:

$$\Omega_{\mathcal{V}}(f) = \frac{1}{2} \sum_{v_1, v_2 \in \mathcal{V}} \sum_{u \in \mathcal{U}} \frac{1}{\tau(u)} c_u(v_1, v_2) \left(\frac{f(v_1)}{\sqrt{p(v_1)}} - \frac{f(v_2)}{\sqrt{(p(v_2))}} \right)^2. \quad (5)$$

Convexly combining together the two cost functionals Eq.(4) and Eq.(5), we can obtain an inference cost measure of function f over the bipartite graph $\mathbb{G}^*$:

$$\Omega_\gamma(f) = \gamma \cdot \Omega_\mathcal{U}(f) + (1-\gamma) \cdot \Omega_\mathcal{V}(f), \quad \text{s.t. } 0 \leq \gamma \leq 1, \quad (6)$$

where the parameter γ indicates the relative importance between $\Omega_\mathcal{U}(f)$ and $\Omega_\mathcal{V}(f)$.

6.2.3 Recommendation by Regularization

$\Omega_\gamma(f)$ captures the inference cost of labeling nodes in a bipartite graph. For recommendation, the intuitive idea is to minimize the inference cost, since we want to find the set of users closely relevant yet not recommended to the target user, by making use of the co-linkage of nodes. Besides the inference, we have additional information about users in reciprocal communities, i.e., the interactive activities, which can be regarded as a user's engagement profile for recommendation. Formally, given a user in $u \in \mathcal{U}$, we can define a function y in $\mathcal{H}(\mathbb{G}^*)$ in which $y(v \in \mathcal{V}) = 1$ if vertex v has interaction with u, or 0 if v has never interacted with u. Then the recommendation problem can be regarded as the problem of finding a function f, which infers new vertices for u while reproducing the target function y to a sufficient degree of accuracy [22, 7]. A formalization of this idea leads to the following optimization problem:

$$f^* = \arg\min_{f \in \mathcal{H}(\mathbb{G}^*)} \{\Omega_\gamma(f) + \mu \|f - y\|^2\}, \quad (7)$$

where $\mu > 0$ is the regularization parameter. The first component measures the inference cost of function f, and the second component indicates the closeness of f with respect to the given function y. The trade-off between these two competitive terms is captured by μ. The solution of Eq.(7) can be found in [22].

After obtaining the result of $f^*(u)$ for user u, we can take $\text{sign} f^*(u)$ to select the vertices in $\mathcal{V}$ whose labels are 1, and then rank the selected users based on the mutual relevance of users, i.e., $rel(u, v)$. The final recommendation result is obtained by selecting the top ranked ones without considering the users who have already interacted with u and the users whose availabilities exceed the availability budget b. Further, if the target user is a vital user, then the recommended list will be ranked via the vitality of users in an ascend order; otherwise, the list will be ranked in a descend order of the vitality. In this way, the engagement of passive users is possible to be improved to some extent.

6.3 Feature Weighting

In the context of reciprocal recommendation, e.g., online dating and online recruiting, users will not return to the system during a long time period if they obtain their preferred friends or jobs. In this case, the system will not recommend them any further information. However, such users' profiles are good indicators showing that the recommendation of the system is successful, and therefore they have great potential to improve the quality of the reciprocal network. In *MEET*, we filter out the profiles of these "successful" users, and then analyze their properties in order to facilitate the construction of the bipartite graph.

Specifically, we use feature weighting algorithms to capture the relative importance among different user features. Feature weighting has already been extensively explored [3, 8]. In our framework, we choose the Passive-Aggressive (PA) algorithm [3] to learn the weights for different features. The goal of PA is to change the model as little as possible to correct for any mistakes and low-confidence predictions it encounters, with the following optimization:

$$\mathbf{w}_{t+1} \leftarrow \arg\min_{\mathbf{w}} \frac{1}{2}\|\mathbf{w}_t - \mathbf{w}\|^2, \text{s.t. } y_i(\mathbf{w} \cdot \mathbf{x}_t) \geq 1. \quad (8)$$

Updates occur when the inner product does not exceed a fixed confidence margin, i.e., $y_t(\mathbf{w}_t \cdot \mathbf{x}_t) < 1$. The closed-form update for all examples is as follows:

$$\mathbf{w}_{t+1} \leftarrow \mathbf{w}_t + \alpha_t y_t \mathbf{x}_t, \quad (9)$$

where $\alpha_t = \max\{\frac{1-y_t(\mathbf{w}_t \cdot \mathbf{x}_t)}{\|\mathbf{x}_t\|^2}, 0\}$. (Please refer to [3] for more details.) The PA algorithm has been successfully applied in practice because the updates explicitly incorporate the notion of classification confidence. Once we obtain the weights for each feature in the feature repository, we feed them into the relevance model described in Section 5, and then use the refined relevance to construct the bipartite graph.

7. DISCUSSION

The reciprocal recommendation framework we propose is quite general. In this section, we show the connections and differences between our framework and various existing methods for reciprocal recommendation. The methods discussed in this section include gradient boosted decision trees (GBDTs) [6], reciprocal recommender for online dating (RECON) [15] and content-collaborative reciprocal recommender (CCR) [1].

GBDTs and *MEET*: GBDTs considers the relevance between the query and the candidate by integrating the matching attributes and post-presentation (activities) features into a unified feature vector. Based on this vector, GBDTs calculates the relevance score between two single users. *MEET* considers a more general problem – recommendation, in which a user might have no definite preference on the information and therefore no explicit query is specified. Also the post-presentation features might not be available to new users. Moreover, *MEET* formalizes the reciprocal community as a dynamic network, whereas in GBDTs, the users are treated individually.

RECON and *MEET*: RECON calculates the compatibility scores between users from different sets by considering the self-description attributes and the activities. However, it fails to consider the other special characteristics of reciprocal recommendation, e.g., the *sparsity*, the *limitedness* and the *passiveness*. In such sense, RECON can be regarded as a special case of our proposed generalized framework.

CCR and *MEET*: CCR computes the users' similarity based on the content of user profiles, and then performs recommendation from collaborative-wise. It considers the a single-step diffusion of "like" and "dislike" of users towards other users. Such a diffusion is also incorporated into *MEET* by the inference on the refined bipartite graph. Therefore, CCR can also be treated as a special case of *MEET*.

8. EXPERIMENTAL EVALUATION

8.1 Real-world Data

Online Dating Data: This data set is collected from a dating web site from Oct, 2008 to Mar, 2011, with anonymized user information. The original feature space is described in

Table 1. We transform the features using the strategy described in Section 5, and then calculate the user relevance based on the new feature space. We set $\mathcal{U}$ as the male set and $\mathcal{V}$ as the female set. The statistics of this data set is depicted in Table 2(a).

Table 2: Statistics of two data sets.

(a) Online Dating

Male (u)	344,552
Female (v)	203,843
# of $\mathcal{F}_u^s$	528
# of $\mathcal{F}_u^p$	506
# of $\mathcal{F}_v^s$	506
# of $\mathcal{F}_v^p$	528
Activities	8,599,013

(b) Online Recruiting

Job Seekers (u)	199,999
Recruiters (v)	46,629
# of $\mathcal{F}_u^s$	860
# of $\mathcal{F}_u^p$	928
# of $\mathcal{F}_v^s$	928
# of $\mathcal{F}_v^p$	860
Activities	664,943

Online Recruiting Data: This data set is provided by Xiamen Talent Service Center[1], a popular online recruiting platform in China. We collected the profiles and activities for anonymized users from Jan, 2008 to Oct, 2011. The feature set is not listed due to the space limit. We set $\mathcal{U}$ as the job seeker set and $\mathcal{V}$ as the recruiter set. We use the same strategy to process this data set. The statistics of the data after processing is described in Table 2(b).

8.1.1 Experimental Setup

For experiments, we split the two data sets into training and testing sets. Each set includes two sets of users, associated with their interactive activities. Since a reciprocal recommender system may suffer from the *sparsity* problem, we treat users who do not have activities within 3 continued months as "successful" users and feed their profiles into the learning module for feature weighting. These profiles will not be included in the bipartite graph. Table 3 summarizes the details of training and testing datasets for both online dating and online recruiting data.

Table 3: Statistics of training and testing datasets.

(a) Online Dating

	training	testing
Male	293,470	51,082
Female	168,329	35,514
Activities	6,735,210	1,863,803
Successful	26,380	-

(b) Online Recruiting

	training	testing
Job Seekers	176,423	23,576
Recruiters	39,850	6,779
Activities	493,128	171,815
Successful	10,458	-

8.1.2 Evaluation Metrics

For each user in the testing set, we recommend top ranked users (top@10, top@20 and top@30) to the user at each week of the testing range. Within the testing set, each user has a series of activities, e.g., sending messages to other users. Based on such activities, we use different evaluation metrics to verify the quality of the recommended user list.

Set Evaluation: For comparison, we compute the averaged precision and recall based on users' activities over weeks. Specifically, the ground truth of a user u's activities, including who have been clicked or contacted by u, is denoted by M, and the recommended user list by algorithms is denoted by N. Then the precision (P) and recall (R) can be computed as

$$P = \frac{M \cap N}{N}, \quad R = \frac{M \cap N}{M}. \tag{10}$$

We then compute the F_1-score of the recommendation results, i.e., $F_1 = \frac{2PR}{P+R}$.

Ranking Evaluation: We employ Normalized Discount Cumulative Gain (NDCG) to evaluate the ranking quality of the recommended list based on a user's actual activity sequence. NDCG at position n is defined as

$$NDCG@n = N(n) \times \sum_{i=1}^{n} \frac{2^{r_i} - 1}{\log_2(i+1)}, \tag{11}$$

where $N(n)$ is the NDCG at n of the ideal ranking list, and r_i is the relevance rating of item at rank i. In our scenario, $r_i = 1$ if the user has clicked on or contacted with the recommended users and 0 otherwise.

Vitality Evaluation: The vitality of a user is an important feature within the reciprocal community. It defines how active the user is, e.g., how often the user sends messages to other users. By explicitly considering the vitality for recommendation, a vital user improve the engagement of other passive users, which renders the reciprocal network more healthy and energetic. To measure how active that users within the recommended list are, we define the *set vitality* measurement as the *average activeness* of all the users in the list. Specifically, given a recommended user set $\mathcal{S}$, associated with each user's interactive activities $u_{i\leftrightarrow}$, the *set vitality* of $\mathcal{S}$ is calculated as

$$f_a(\mathcal{S}) = \frac{1}{|\mathcal{S}|} \sum_{u_i \in \mathcal{S}} \frac{1}{|V_{u_i}|} \sum_{v_j} \frac{|u_{i\to j}|}{|u_{i\leftrightarrow j}|}, \tag{12}$$

where V_{u_i} is the set of users that have been clicked or contacted by u_i, $|u_{i\to j}|$ denotes the number of clicks on v_j by u_i and messages that u_i has sent to v_j and $|u_{i\leftrightarrow j}|$ represents the total number of interactions between u_i and v_j.

8.2 Experiment Results

In this subsection, we first evaluate the effects of local recommendation (recommending users within a bipartite subgraph) and global recommendation (recommending users within the entire bipartite graph); We then investigate the impact of different reciprocal properties to the recommendation result; Further, we demonstrate the effectiveness of our framework on improving the vitality of the reciprocal network. We also provide comprehensive comparisons with recently published approaches discussed in Section 7.

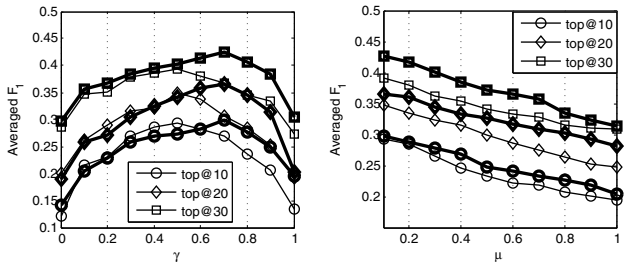
8.2.1 Parameter Setting

In our framework, there are several parameters for tuning, including the availability threshold b, the relevance threshold θ to construct the bipartite graph, the number of clusters k in Section 6.1, the importance factor γ in Section 6.2.2 and the regularization parameter μ in Section 6.2.3. For b, we empirically set it as 30 for online dating and 50 for online recruiting. For θ, we empirically set it as 0.01 for both online dating and online recruiting data. For k, we empirically set it as 10 for online dating and 20 for online recruiting. We will evaluate the performance of the algorithm with different k in the next section. To explore the effect of the parameters γ and μ towards the recommendation result, we use F_1-score as the evaluation metric to compare the quality of

[1]http://www.xmrc.com.cn.

the recommended list (top@10, top@20 and top@30). Note that the optimal parameter setting is obtained by *performing cross-validation on the training data*.

Figure 3 shows the performance curve with different parameter settings. We first empirically fix μ as 0.1 and evaluate γ. Figure 3(a) shows the F_1-score measured as a function of γ. The optimal value of γ for online dating is obtained when $\gamma = 0.5$, meaning that the two sets of users have approximately equal importance towards the development of the online dating environment. However, for the online recruiting dataset, the optimal γ is 0.7, indicating that within the online recruiting system, the inference cost of recommendation for job seekers is dominating. Such an observation is consistent with the actual situation since in practice the number of job seekers is superior to recruiters. We then fix γ as 0.5 for online dating and 0.7 for online recruiting, and evaluate the changing of μ. Figure 3(b) shows the F_1-score measured as a function of μ. The best result is achieved when $\mu = 0.1$. In the following experiments, we set μ as 0.1, γ as 0.5 for online dating and γ as 0.7 for online recruiting.



(a) $\mu = 0.1$ (b) $\gamma = 0.5(0.7)$

Figure 3: Parameter Tuning. Remark: thinner – online dating; thicker – online recruiting.

8.2.2 Local V.S. Global Recommendation

For reciprocal recommendation, our proposed framework first partitions the bipartite graph into multiple subgraphs (with specialized information), and then recommends users to the target user within the specific subgraph that embraces this target user. Such a paradigm can not only expedite the recommendation (since we only consider partial bipartite graph), but also improve the quality of the recommendation list. In the experiment, we choose different values of k (the number of clusters defined in Section 6.1) and compare the results with the one using the entire bipartite graph (Global, for short). The comparison is reported in Figure 4.

With different k, the performance of our proposed framework varies. *MEET* achieves the optimal performance when $k = 20$ for online recruiting dataset. *By partitioning the entire bipartite graph into multiple subgraphs, the information within each subgraph is more specialized, and therefore the reciprocity within each subgraph becomes considerably dense.* However, the performance on online dating is not quite sensitive to the number of clusters k. When $k = 10$, the results are slightly better than the other settings. The reason behind this might be that the populations within the online dating community are more general, whereas the ones within the online recruiting community are mostly related to some specific areas. Another interesting finding is that the overall performance of local recommendation is better than the one of global recommendation, especially for online recruiting. This could benefit from the fact that *the local presentation of the reciprocal network could help specialize the local community, and therefore facilitate the candidate matching within particular domains*.

8.2.3 Effects of Reciprocal Properties

In our generalized framework, we comprehensively consider the special properties of reciprocal recommendation, i.e., *reciprocity*, *limitedness*, *passiveness* and *sparsity*. To examine the influence of different properties on the recommendation results, we evaluate several alternatives of *MEET* as follows:

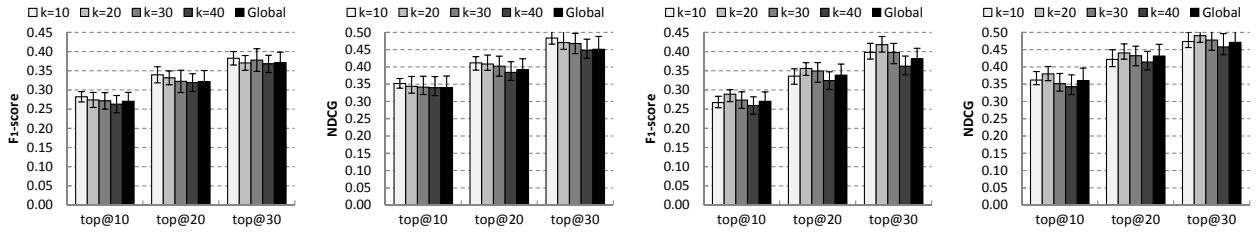
- *MEET*^{1}: Do not consider *reciprocity*, *passiveness*, *limitedness* and *sparsity*, i.e., to drop the term $\frac{1}{\tau(v)}c_v(u_1, u_2)$ in Eq.(4) and Eq.(5) and do not perform feature weight learning process;
- *MEET*^{2}: In Eq.(4) and Eq.(5), only consider the relevance between users, i.e., to drop the term $\frac{1}{\tau(v)}$ and the term $\frac{w_a(v,u_1)w_a(v,u_2)}{q(v)}$ in Eq.(3);
- *MEET*^{3}: In Eq.(3), do not consider the passiveness of users, i.e., to drop the term $\frac{w_a(v,u_1)w_a(v,u_2)}{q(v)}$;
- *MEET*^{4}: In Eq.(4) and Eq.(5), do not consider the availability of users, i.e., to drop the term $\frac{1}{\tau(v)}$;
- *MEET*^{5}: Do not perform the feature weight learning process.

We compare these alternatives with the comprehensive *MEET* in terms of F_1-score and NDCG. Figure 5 shows the comparison results on two datasets.

It is evident that the generalized model *MEET* significantly outperforms the alternatives from both accuracy and ranking perspective. The reason behind this is quite straightforward: in the generalized model, *the special characteristics of reciprocal community are well captured*, rendering the recommendation results derived from such unified model more reasonable. Besides this, we observe that: (1) The *reciprocity* is the dominant aspect in the reciprocal network, since only considering the relevance between users can significantly improve the quality of the recommendation results; and (2) The *limitedness*, *passiveness* and *sparsity* are also important properties of the reciprocal community, by which the performance of *MEET* can achieve slight improvement.

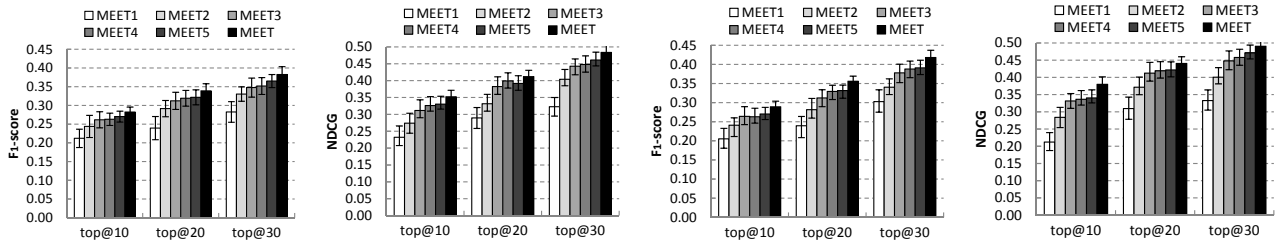
8.2.4 Comparison with Existing Methods

Our proposed framework is designated to reciprocal recommendation, which cannot be easily tackled by traditional collaborative filtering approaches. To verify this claim, we choose two recently published collaborative filtering methods [11, 14] as our baselines. [11] (CFIF for short) proposed treating the data as indication of positive and negative preference associated with vastly varying confidence levels, which is a pure collaborative filtering approach. [14] (OCCF for short) exploited the rich user information available in community-based interactive information systems, and incorporated user information into modeling the recommendation. For this method, we use the neighborhood model as the baseline. We also implement GBDTs [6], RECON [15] and CCR [1] (in Section 7) for comparison. We



(a) F1 on online dating. (b) NDCG on online dating. (c) F1 on recruiting. (d) NDCG on recruiting.

Figure 4: Performance comparison of local recommendation with different k and global recommendation.



(a) F_1 on online dating. (b) NDCG on online dating. (c) F_1 on recruiting. (d) NDCG on recruiting.

Figure 5: Performance comparison of different alternatives of *MEET*.

use F_1-score and NDCG to compare these algorithms with *MEET* for both online dating and online recruiting datasets. The feature set used in the baselines are identical to the one in our proposed method, and also the parameters in the baselines are optimally tuned.

The results are shown in Table 4. It is evident that *MEET* significantly outperforms the baselines on both F_1-score and NDCG. The two collaborative filtering based methods cannot effectively handle the reciprocal task. We investigated the recommendation results of both methods and found that users in most recommended matches are relevant. However, there are two reasons that both users in a match have *few or even no interactions*: (1) *The recommended user has been recommended to multiple users, and therefore he/she has limited availability*; and (2) *Both users are not vital, and hence they do not contact with each other*. The three reciprocal methods being compared can slightly improve the recommendation performance; however, they only focus on different aspects of the reciprocal community. Instead, *MEET* provides a comprehensive overview of the reciprocal network, and therefore achieves the best.

8.2.5 Vitality Evaluation

The recommended user list provided by *MEET* exhibits a great vitality, which is originated from the sparkle that we intentionally consider the *passiveness* of users. We assume that passive users can be spurred by active users, and formalize the activeness of users in Eq.(3). Such activeness is beneficial to construct an energetic reciprocal network, in which users are willing to proactively contact with each other, and therefore improve the vitality of the network.

To evaluate the vitality of the recommended results, we use the *set vitality* measurement defined in Eq.(12), and compare *MEET* with GBDTs, RECON and CCR. These three methods consider the interactive activities of users from different perspectives. We also compare *MEET* with an alternative $MEET^3$ that does not consider the passiveness of users. Note that the recommended list are ranked based on the activeness of users. The ranking quality of the recommended list has been verified in Section 8.2.4, and therefore we put our concern on the overall vitality of the list. We report the comparison results in Table 5.

Table 5: Evaluation on the set vitality of the recommended results. (Bold indicates the best performance. * indicates the statistical significance at $p < 0.01$.)

	top@10	top@20	top@30
GBDTs	0.3513	0.3306	0.3027
RECON	0.3629	0.3430	0.3085
CCR	0.3731	0.3325	0.2964
$MEET^3$	0.3515	0.3276	0.2909
MEET	**0.4639***	**0.4517***	**0.4425***

From the comparison, we observe that the set vitality of results from different methods varies a lot. Since our framework explicitly formalizes the interaction between users into Eq.(3), it achieves the best performance. An interesting phenomenon is that when the number of recommended results increases, most methods show a decreasing trend in terms of the *set vitality*. Take CCR as an example for further analysis. CCR generates the results by considering the users who have interactions with users that are similar to the target user (based on the profile). The user ranking is based on the reciprocal interests. When recommending more users to the target user, the reciprocal interests of users with lower rankings will decrease significantly, and therefore the set vitality of the recommended list deceases. Comparatively, in our framework, we prefer dedicated users, i.e., *the users who have a lot of interactions with several other users but do not send flooded messages*. Based on this intuition, *MEET* outperforms other candidates.

Table 4: Comparison with existing methods. (Bold indicates the best performance. * indicates the statistical significance at $p < 0.01$.)

Methods	Online Dating						online recruiting					
	top@10		top@20		top@30		top@10		top@20		top@30	
	F_1	NDCG	F_1	NDCG	F_1	NDCG	F_1	NDCG	F_1	NDCG	F_1	NDCG
CFIF	0.2307	0.3069	0.2918	0.3534	0.3206	0.4417	0.2301	0.3174	0.3121	0.3813	0.3481	0.4036
OCCF	0.2415	0.3134	0.3059	0.3670	0.3385	0.4498	0.2485	0.3320	0.3219	0.3929	0.3569	0.4127
GBDTs	0.2607	0.3146	0.3106	0.3881	0.3475	0.4662	0.2567	0.3592	0.3304	0.4131	0.3718	0.4432
RECON	0.2523	0.3221	0.3027	0.3672	0.3503	0.4530	0.2604	0.3608	0.3247	0.4025	0.3839	0.4507
CCR	0.2309	0.3250	0.3389	0.3724	0.3428	0.4621	0.2431	0.3745	**0.3573**	0.3987	0.3912	0.4729
MEET	**0.2823***	**0.3521***	0.3390	**0.4118***	**0.3826***	**0.4836***	**0.2890***	**0.3799**	0.3560	**0.4405***	**0.4181***	**0.4904***

9. CONCLUDING REMARKS

In this paper, we study the problem of reciprocal recommendation. We comprehensively investigate the special properties of a reciprocal community, including *reciprocity*, *limitedness*, *passiveness* and *sparsity*. We propose a generalized reciprocal framework, *MEET*, in which the aforementioned properties are seamlessly integrated. Specifically, *MEET* first constructs a bipartite graph based on the mutual relevance of users, and then performs graph inference on the resulted subgraphs to obtain the recommendation list for individuals. The inference model formalizes the properties of the reciprocal network and elegantly casts the recommendation as an optimization problem.

Bilateral reciprocal recommendation discussed in our work might not cover all possible reciprocal recommendation tasks in a broader perspective. For example, friend recommendations on Facebook and colleague recommendations on LinkedIn exhibit different characteristics, since the recommendation activities on these two platforms might involve multiple parties instead of two. In the future, we plan to expand our reciprocal framework to tackle more reciprocal tasks.

ACKNOWLEDGEMENT

The work is partially supported by NSF grants DMS-0915110 and CNS-1126619 and DHS grants 2009-ST-062-000016 and 2010-ST-062-000039. We would like to thank Xiamen University and Xiamen Talent Service Center for providing us with the online recruiting data set used in the paper.

10. REFERENCES

[1] J. Akehurst, I. Koprinska, K. Yacef, L. Pizzato, J. Kay, and T. Rej. Ccr-a content-collaborative reciprocal recommender for online dating. In *Proc. of IJCAI*, pages 2199–2204, 2011.

[2] X. Cai, M. Bain, A. Krzywicki, W. Wobcke, Y. Kim, P. Compton, and A. Mahidadia. Collaborative filtering for people to people recommendation in social networks. *Advances in Artificial Intelligence*, pages 476–485, 2011.

[3] K. Crammer, O. Dekel, J. Keshet, S. Shalev-Shwartz, and Y. Singer. Online passive-aggressive algorithms. *The Journal of Machine Learning Research*, 7:551–585, 2006.

[4] I.S. Dhillon. Co-clustering documents and words using bipartite spectral graph partitioning. In *Proc. of SIGKDD*, pages 269–274. ACM, 2001.

[5] I.S. Dhillon, S. Mallela, and D.S. Modha. Information-theoretic co-clustering. In *Proc. of SIGKDD*, pages 89–98. ACM, 2003.

[6] F. Diaz, D. Metzler, and S. Amer-Yahia. Relevance and ranking in online dating systems. In *Proc. of SIGIR*, pages 66–73, 2010.

[7] C. Ding, R. Jin, T. Li, and H.D. Simon. A learning framework using greenąŕs function and kernel regularization with application to recommender system. In *Proc. of SIGKDD*, pages 260–269. ACM, 2007.

[8] M. Dredze, K. Crammer, and F. Pereira. Confidence-weighted linear classification. In *Proc. of ICML*, pages 264–271, 2008.

[9] J.H. Friedman. Greedy function approximation: a gradient boosting machine. *Annals of Statistics*, pages 1189–1232, 2001.

[10] D. Gale and L.S. Shapley. College admissions and the stability of marriage. *American Mathematical*, 69(1):9–15, 1962.

[11] Y. Hu, Y. Koren, and C. Volinsky. Collaborative filtering for implicit feedback datasets. In *Proc. of ICDM*, pages 263–272. IEEE, 2008.

[12] M. Karimzadehgan and C.X. Zhai. Constrained multi-aspect expertise matching for committee review assignment. In *Proc. of CIKM*, pages 1697–1700. ACM, 2009.

[13] Y. Kim, A. Mahidadia, P. Compton, X. Cai, M. Bain, A. Krzywicki, and W. Wobcke. People recommendation based on aggregated bidirectional intentions in social network site. *Knowledge Management and Acquisition for Smart Systems and Services*, pages 247–260, 2011.

[14] Y. Li, J. Hu, C.X. Zhai, and Y. Chen. Improving one-class collaborative filtering by incorporating rich user information. In *Proc. of CIKM*, pages 959–968. ACM, 2010.

[15] L. Pizzato, T. Rej, T. Chung, I. Koprinska, and J. Kay. Recon: a reciprocal recommender for online dating. In *Proc. of Recommender Systems*, pages 207–214. ACM, 2010.

[16] L. Pizzato, T. Rej, K. Yacef, I. Koprinska, and J. Kay. Finding someone you will like and who won't reject you. *User Modeling, Adaption and Personalization*, pages 269–280, 2011.

[17] L.A. Pizzato and C. Silvestrini. Stochastic matching and collaborative filtering to recommend people to people. In *Proc. of Recommender Systems*, pages 341–344. ACM, 2011.

[18] D. Richards, M. Taylor, and P. Busch. Expertise recommendation: A two-way knowledge communication channel. In *Proc. of ICAAS*, pages 35–40. IEEE, 2008.

[19] L. Terveen and D.W. McDonald. Social matching: A framework and research agenda. *ACM Transactions on Computer-Human Interaction*, 12(3):401–434, 2005.

[20] F. Wang, T. Li, X. Wang, S. Zhu, and C. Ding. Community discovery using nonnegative matrix factorization. *Data Mining and Knowledge Discovery*, 22(3): 493-521.

[21] X. Yi, J. Allan, and W.B. Croft. Matching resumes and jobs based on relevance models. In *Proc. of SIGIR*, pages 809–810. ACM, 2007.

[22] D. Zhou, O. Bousquet, T.N. Lal, J. Weston, and B. Schölkopf. Learning with local and global consistency. *Advances in neural information processing systems*, 16:321–328, 2004.

[23] D. Zhou, B. Schölkopf, and T. Hofmann. Semi-supervised learning on directed graphs. *Advances in neural information processing systems*, pages 1633–1640, 2005.