"Make New Friends, but Keep the Old" – Recommending People on Social Networking Sites

Jilin Chen*, Werner Geyer, Casey Dugan**, Michael Muller**, Ido Guy*****

*University of Minnesota
200 Union Street SE
Minneapolis, MN 55455
jilin@cs.umn.edu

**IBM T.J Watson Research
One Rogers Street
Cambridge, MA 02116
{werner.geyer, cadugan, michael_muller}@us.ibm.com

***IBM Haifa Research Lab
Mt. Carmel
Haifa 31905, Israel
ido@il.ibm.com

ABSTRACT

This paper studies people recommendations designed to help users find known, offline contacts and discover new friends on social networking sites. We evaluated four recommender algorithms in an enterprise social networking site using a personalized survey of 500 users and a field study of 3,000 users. We found all algorithms effective in expanding users' friend lists. Algorithms based on social network information were able to produce better-received recommendations and find more known contacts for users, while algorithms using similarity of user-created content were stronger in discovering new friends. We also collected qualitative feedback from our survey users and draw several meaningful design implications.

Author Keywords
Social networking, friend, recommender system

ACM Classification Keywords
H.5.3 Information Interfaces and Presentation (e.g., HCI): Group and Organization Interfaces.

INTRODUCTION

Social networking sites allow users to articulate their social networks by adding other users to their "friend lists". Research shows that users connect to both friends they already know offline and new friends they discover on the site. For example, many users of popular social networking sites such as Facebook and MySpace primarily communicate with people they already know offline [1]. On the other hand, research on enterprise social networking [3] shows that users in a corporate context are interested in finding valuable contacts not yet known to them, or connecting to weak ties, in addition to staying in touch with their close colleagues. Given the size of social networking sites, finding known contacts and interesting new friends to connect with on the site can both be a challenge.

One approach to address this problem is to proactively make personalized people recommendations on the site. Facebook has recently launched a feature, called "People You May Know", which recommends people to connect with based on a "friend-of-a-friend" approach [14]. However, data on the effectiveness of this approach is not available. As a recommendation problem, recommending people on social networking sites is worth studying because it is different from traditional recommendations of books, movies, restaurants, etc. due to the social implications of "friending". For example, before adding a friend, one often has to consider how the other person would perceive this action and whether he or she would acknowledge the friendship. Furthermore, because the friend list appears on one's profile, one also has to consider how the new friend will be perceived by others on the site. These social dynamics can be obstacles in accepting recommendations, even when they are relevant and otherwise desirable. This problem could be more prominent if unknown or barely known people are recommended, because in those cases users often lack enough motivation to contact or reach out to the other person. However, despite the difficulty, connecting with weak ties or unknown but similar people can be more valuable to users than merely re-finding existing strong ties [6]. In contrast, recommendations of books, movies, restaurants etc. do not have such reciprocal social or impression management issues.

In this paper we describe the results of an empirical study of a people recommender system for an enterprise social networking site. Our goal was to get a basic understanding of this area with a particular focus on the following two research questions:

1) How effective are different algorithms in recommending people as potential friends, and what are their characteristics in terms of recommending known versus unknown people?

2) Can a people recommender system effectively increase the number of friends a user has, and what would be the overall impact of such a recommender system on the site?

To answer these questions, we designed and implemented a people recommender system for Beehive, an enterprise social networking site within IBM, using four different algorithms. We conducted two separate experiments, a personalized survey and a controlled field study. The survey was targeted at a group of 500 users who were asked to answer questions related to their friending behavior, and to rate personalized recommendations created from each algorithm. In the controlled field study we deployed our people recommender as a feature on the site to 3,000 users.

In the next section, we discuss how existing work relates to our research. We then provide a brief overview of Beehive, followed by a detailed description of the four algorithms chosen for our study. The next two major sections describe the results from the personalized survey and the controlled field study. We conclude with a discussion of our findings and possible future work.

RELATED WORK

There is a wealth of research on recommender systems. Many approaches employ the technique of collaborative filtering [10], which utilizes similarities of preferences among users to recommend items such as movies for a user to consume. These approaches do not rely on the actual content of the items, but instead require users to indicate preferences on them, usually in the form of ratings.

Another body of research utilizes the content of items to make recommendations. Pazzani et al. [15] characterized websites by words contained on individual pages and built user profiles using websites the user considered “hot”. They applied a naïve Bayes classifier to recommend interesting websites based on the profile. Mooney et al. [13] followed a similar paradigm with a content-based recommender for books. Sen et al. [18] built a hybrid alert filtering system combining collaborative filtering techniques and the content features extracted from the alert message itself.

Research has also been done using articulated social network structures for recommendations. For instance, Spertus et al. [19] made recommendations of online communities to users based on their current community membership, and compared several different similarity measures in a large study on the social networking site Orkut. Geyer et al. [5] built a system to recommend topics for self-descriptions using social network information, and showed that a social network-based recommender yielded better performance than simple content matching. Groh et al. [7] generated user neighborhood information from articulated social network structures and demonstrated that collaborative filtering based on such neighborhoods outperforms classic collaborative filtering methods.

Many of the techniques mentioned above could potentially be used for recommending people on social networking sites. For example, methods utilizing content similarity can be directly adapted by computing similarity from profiles, photos, comments, etc. of users. Similarly, methods exploiting articulated social network structure are also readily applicable. In contrast, classic collaborative filtering does not directly apply because it recommends items in a heterogeneous network of users and items using ratings, while in our case we want to recommend users to users without requiring a concept of items or ratings.

Research on expert-finding using social information is very relevant as well. McDonald [12] discussed leveraging social network information when finding knowledgeable colleagues for collaborations in the workplace. Ehrlich et al. [4] introduced a system which supports users who are searching for experts in their social network, using email and chat messages. However, compared to prior research, our focus is not on the user-directed task of finding an expert, but on recommending people on social networking sites for the purpose of establishing connections (“friends”) and communication, similar to Facebook’s “People You May Know” application [14]. In these cases, users are not necessarily actively looking for new friends even if they are open to meeting or connecting with them.

Our people recommender could also be viewed as a concrete example of a "social matching system" as described in Terveen et al. [21]. Their work discusses issues closely related to our paper, and our comparison of different people recommendation algorithms in a large scale experiment could answer several questions laid out in their work. Because of the unique challenges of recommending people as illustrated in our introduction and in [21], we believe designing, implementing, deploying, and studying different recommenders can greatly help us understand and improve people recommendation systems.

THE SOCIAL NETWORKING SITE

Beehive is an enterprise social networking site within IBM. It was officially launched in September 2007 and had more than 38,000 users with an average of 8.2 friends per user at the time when we started our study in July 2008. Similar to other social networking sites, Beehive has an individual profile page for each user, and supports features like friending other people, setting status messages, sharing photos, lists, events, and commenting on users as well as on shared content. Beehive has experienced viral adoption since its launch and users share a wealth of personal and professional information on the site. For more details about activity on the site and various motivations for employees to participate in social networking in the enterprise see [3].

The concept of a “friend” in Beehive is more similar to Flickr than Facebook in that Beehive friends are directional, i.e. there could be a non-reciprocal friendship. Accordingly, instead of requiring prior consent by both people before a friend relation is established, one can connect to any user on the site right away. The target user will be notified by email and given the option to connect back. Note that throughout this paper, we will use the term “connected” to describe the state of a user having added another user as a friend, and “connect to” / “friending” for the action of adding another user as a friend.

In order to facilitate friending and in particular friending of unknown users, Beehive also has an introduction feature, where a user can request to be introduced to another user through Beehive. The user can specify the recipient of the introduction, add shared content of common interest, and add a message to be sent with the request. Beehive will then send out the introduction and let the recipient decide how to respond and whether to make the requester a friend.

PEOPLE RECOMMENDATION ALGORITHMS

For our study, we evaluated four different algorithms. Our choice was driven by a number of considerations. First, following our discussion in the Related Work Section, we focused on algorithms that utilize social network structure and those based on content similarity, because they have been successfully used in related fields [5, 7, 13, 14, 15, 18] and would likely yield recommendations of great variety and coverage due to their different underlying mechanisms. Second, when choosing between different alternatives within each type, we preferred well-established algorithms of that type, so that whatever differences we observe in the experiment can be attributed to the type of the algorithm and not to a very particular technique used in that specific algorithm of choice. Finally, considering the potential obstacles to adding friends as discussed in the introduction, we required that all our algorithms be able to provide additional information as explanations to users, to explain why a person was recommended and thus increase the users' motivation to friend the recommended person.

Algorithm 1: Content Matching

Our content matching algorithm is based on the intuition that "if we both post content on similar topics, we might be interested in getting to know each other". In other words, the algorithm strives to find users associated with similar content on Beehive. This approach is closely related to finding documents of similar content in the information retrieval field [17].

Following a paradigm commonly used for information retrieval, we first create a bag-of-words representation of each user, using textual content both from within Beehive and from our corporate directory. From Beehive, we extract words from profile entries and status messages of users, as well as the title, description, tags, and any textual content associated with their photos and shared lists. From the corporate directory, we extract the job title and the city of the user's work location. All words are stemmed using a Porter stemmer [16], and then filtered using a customized stop word list containing about 550 common English words. All remaining word stems associated with a user *u* are used to create a word vector $V_u = (v_u(w_1), \ldots, v_u(w_m))$ to describe *u*, where *m* is the total number of distinct words used in all included texts and each $v_u(w_i)$ describes the strength of *u*'s interest in word w_i. The value of $v_u(w_i)$ is calculated using a term-frequency inverse-user-frequency weighting, a direct adaptation of TF-IDF [17]:

$TF_u(w_i)$= (#uses of w_i by u)/(#all words used by u)

$IDF_u(w_i)$=log[(#all users)/(#users using w_i at least once)]

$v_u(w_i) = TF_u(w_i) \cdot IDF_u(w_i)$

The similarity of two users *a* and *b* is then measured by the cosine similarity of their word vectors V_a and V_b. Intuitively this means *a* and *b* would be considered similar if they share many common keywords in their associated content, and even more so if only a few users share those keywords. Users similar to the recipient user *u* are recommended in decreasing order of similarity. As an explanation for a recommendation *c*, we show up to 10 top words *w* whose dot product $v_u(w) \cdot v_c(w)$ is among the highest in all words shared. Intuitively they are the strongest common words shared by *u* and *c*. On Beehive we were able to compute at least one content based recommendation for 99.1% of all users.

We also analyzed newer and more sophisticated content similarity algorithms, including Latent Semantic Analysis [2] and Probabilistic Latent Semantic Analysis [11]. However, in a preliminary test they did not yield significantly better results. Moreover, since they cannot easily provide an intuitive explanation for recommendations like common keywords, we decided against using them.

Algorithm 2: Content-plus-Link (CplusL)

Our content-plus-link algorithm enhances the content matching algorithm with social link information derived from social network structure. The motivation behind this algorithm is that by disclosing a network path to a weak tie or unknown person, the recipient of the recommendation will be more likely to accept the recommendation. The content-plus-link algorithm computes similarity in the same way as the content matching algorithm described in the previous section. However, instead of recommending users with top similarity scores, we boost the similarity of a candidate user *c* and *u* by 50% if a valid social link from *u* to *c* exists, i.e. content matches with less strength in keyword overlap but with a social link between *c* and *u* can be ranked higher than content matches with strong keyword overlap but no link in the social network.

A valid social link is defined as a sequence of three or four users, the first being the recipient of the recommendation and the last being the recommended user. Every two consecutive users *a* and *b* in the sequence must satisfy at least one of the following conditions:

1. *a* connects to *b*
2. *a* has commented on *b*
3. *b* connects to *a*

This definition guarantees that a social link exists between two users if and only if there is at least a minimum level of acquaintance and interactions between them or their friends. An example of such a link between user Alice and Charles would be "Alice has commented on Bob, who is considered a friend by Charles." [1]

By increasing the similarity scores of recommendation candidates with valid links, this algorithm favors people in close social network proximity to the user over people more disconnected from the user in the social network. For recommendations with a valid link, besides the common words generated from the content matching technique, we also show the social link as an explanation, including the type of interactions of all users in the link between user *u* and candidate *c*. On average 77.8% of the top 10 recommendations computed with this algorithm in our experiments contain valid social link information.

Algorithm 3: Friend-of-Friend (FoF)
In contrast to the previous algorithm, the friend-of-friend algorithm leverages only social network information of friending based on the intuition that "if many of my friends consider Alice a friend, perhaps Alice could be my friend too". Many social network analysis approaches have adopted similar ideas to find neighborhoods and paths within the network [5, 6, 7]. This particular variant that recommends friends of a friend is interesting not only because of the clear intuition behind it, but also because, as implied in the official Facebook blog [14], it is the primary algorithmic foundation of the "People You May Know" feature on Facebook, which is one of the few known people recommenders deployed on a social networking site.

Formally speaking, if we define predicate $F(a,b)$ to be true if and only if *b* is a friend of *a* for users *a* and *b* on Beehive, the algorithm can be described as follows: for a user *u* being the recipient of the recommendation, its recommendation candidate set is defined as

$$RC(u) = \{\text{user } c \mid \exists \text{ user } a \text{ s.t. } F(u,a) \text{ and } F(a,c)\}.$$

For each candidate $c \in RC(u)$, its mutual friends[2] set is

$$MF(u,c) = \{\text{user } a \mid F(u,a) \text{ and } F(a,c)\},$$

which represents the friends of *u* that connect to *c* and thus serve as a bridge between *u* and *c*. We then define the score of each candidate *c* for recipient *u* as the size of $MF(u,c)$. The candidates are recommended to *u* in decreasing order of their score. For a single recommended candidate *c*, we supply the mutual friends in $MF(u,c)$ as the explanation for recommending *c*. Note that, because the algorithm requires existing friends, it cannot generate recommendations for people with no or a limited number of friends. We were able to compute at least one recommendation for 57.2% of all Beehive users.

Algorithm 4: SONAR
This algorithm is based on the SONAR system, which aggregates social relationship information from different public data sources within IBM [8, 9]. In this paper we use SONAR to aggregate relationship information from the following seven data sources within our Intranet: (1) organizational chart, (2) publication database, (3) patent database, (4) friending system, (5) people tagging system, (6) project wiki, and (7) blogging system. A relationship is indicated if within that data source two people have somehow interacted with each other, such as co-authoring a paper or leaving comments on each others' blog.

For each of these data sources SONAR computes a normalized relationship score in the range of *[0,1]* between two people, where *0* indicates no relationship and *1* indicates the strongest relationship. These scores are then aggregated to a unified single score by equally weighting each data source [8]. Given a user *u*, SONAR returns a list of users related to *u* and their aggregated relationship score with *u*, ordered by this score. The number of interactions in each data source is used to provide explanations. For example, from the publication database an explanation could be "You two have co-authored 2 papers".

In essence, SONAR incorporates all information available within IBM that implies an explicit acquaintance between pairs of people, and ranks them based on the strength and frequency of their interactions on record. While SONAR runs as a service in IBM, the above algorithm can be easily replicated in other applications. As a minimum basic data source, enterprises typically have a corporate directory with an organizational chart. But SONAR can be extended with additional data sources through a plug-in model. With the data sources configured within IBM, SONAR was able to provide relationship information for almost all users. However, after eliminating existing friends, we were able to create at least one recommendation for 87.7% of all Beehive users.

EXPERIMENT I: PERSONALIZED SURVEY
We conducted a personalized online survey on Beehive in order to get a detailed assessment and comparison of our four algorithms. We also hoped to understand our users' needs, and in particular, whether recommending people is a desired feature and how many users hope to discover new friends on Beehive.

Methodology
We invited 500 active users to participate in a within-subject study, i.e. every user was exposed to all four

[1] We did not include the case of *b* commenting on *a* because we previously discovered spamming through commenting. Including the case would lead to a small group of "spammers" being the top recommended people. Similarly, we did not include links that resulted from connecting behavior by a small number of disproportionately active users because they have friended a majority of users on the site and would otherwise link many unrelated people.

[2] While strictly speaking they are not necessarily mutual friends because of the non-reciprocal friendship on Beehive, we call it this for simplicity reasons.

algorithms. Subjects were randomly selected from all users satisfying the following criteria: First, they must have logged into Beehive during the week preceding the start of the survey. Second, they must have enough data in Beehive so that we can generate at least 10 recommendations using every algorithm. Third, users must have at least 5 words in their associated content that can be used by the content based algorithms, and 3 friends each for the FoF algorithm, so that there is a reasonable amount of data for all algorithms to work with. As shown in Table 1, different algorithms have small overlap in their top 10 recommendations for the 500 selected users except for the two content-based algorithms, which use the same content matching technique.

	Content	CplusL	FoF	SONAR
Content		52.8%	1.8%	8.3%
CplusL			3.3%	9.6%
FoF				13.1%

Table 1. Overlap ratios between recommendations generated by different algorithms.

The survey for each selected user, presented on a single web page, contained 12 recommendations in total, 3 from each algorithm.[3] To control for ordering effects, individual recommendations were presented in a regular and mirrored Latin square sequence, each sequence started randomly with a different algorithm. The 12 recommendations we presented were selected from top ranked recommendation candidates generated by each algorithm. To avoid duplicate recommendations due to overlap between algorithms, if a candidate had already been recommended by another algorithm before, the next highest ranked candidate from the same algorithm would be shown instead.

For each recommendation, we showed a photo, the job title and the work location of that person, as well as the explanation generated by the algorithm. The user could also click a link to view the profile of the recommended person in a separate window. For each recommendation, we asked the following questions:

- Do you already know this person? [yes/no]
- Is this a good recommendation? [yes/no]
- Did the reason we chose this person help you make your decision? [yes/no]
- What action would you like to take? [single choice]
  - Connect to this person
  - Be introduced to this person
  - Nothing
- Additional feedback? [open ended]

[3] Note that all of the algorithms filtered away the people a user is already connected to.

We also asked users more general questions at the beginning and the end of the survey, regarding whether finding people to connect to is difficult, their interest in meeting new people on the site, the kind of information that would make them more likely connect to someone they do not know yet, and whether they consider people recommendations a desired feature for the site.

Results

Of the 500 users, 415 logged in and 258 submitted their survey form. The recommendation response data was analyzed on a per user basis, i.e. we first average each user's responses for each algorithm then summarize the responses over all users who have at least one valid response for every algorithm. Because of missing responses in the survey, the actual sample size of users for some questions dropped to 230.

Understanding users' need

We argued that people recommendations on social networking sites can help users find the right people to communicate with or connect to. In our survey 95% of the users considered people recommendations to be useful and would like to see them as a feature on the site. Our survey also quantitatively confirms DiMicco et al [3] in that users on Beehive are interested in connecting to weak ties and meeting new people: 61.6% said they are interested in meeting new people, 31% said maybe and 7.4% said no.

When asked what kind of information would make them more likely to connect to an unknown person, 75.2% of the users chose common friends, 74.4% said common content (e.g. photos, lists, interests, etc.), 39.2% indicated geographical location of the person, 27% said the division within IBM, and 14.5% chose "other"[4]. Information typically listed as "other" included work/business information (e.g. "customers in common" and "business effort that is similar or relevant to my team's") and skills/expertise (e.g. "reputation in their subject matter" and "they have expressed a skill in an area I could use help discovering/learning"). According to the data, friends and content in common play an important role in decision making and thus, support our design of the content-plus-link algorithm.

Known vs. unknown, Good vs. not good

For every recommendation, users were able to indicate whether or not they already knew that person and they could rate the recommendation as good or not good. Figure 1 shows a breakdown of the results by algorithm. The percentages of unknown people recommended by each algorithm are shown above the horizontal center line and the percentages of known people below. The chart also shows the percentages of good versus not good in two

[4] Users were able to select more than one and up to five types of information including "other". Users chose 2.3 items on average.

different colors, broken down by known and unknown recommendations.

As we originally expected, the pure content matching algorithm recommends mostly unknown people. SONAR, which relies heavily on explicit relationship data, recommends mostly known people. On average each user already knows 85.9% of the people recommended by SONAR, followed by the friend-of-friend algorithm with 60.6%. In contrast, users only know 36.2% of the recommendations from the content-plus-link algorithm, and 22.5% of those from the content matching algorithm (F[3,711] = 213.5, p < .001). Post-hoc comparison (LSD) showed that the percentages for each algorithm were significantly different from each other (p < .001). These results confirm the intuition that the more explicit relationship information an algorithm leverages, the more known people it would recommend.

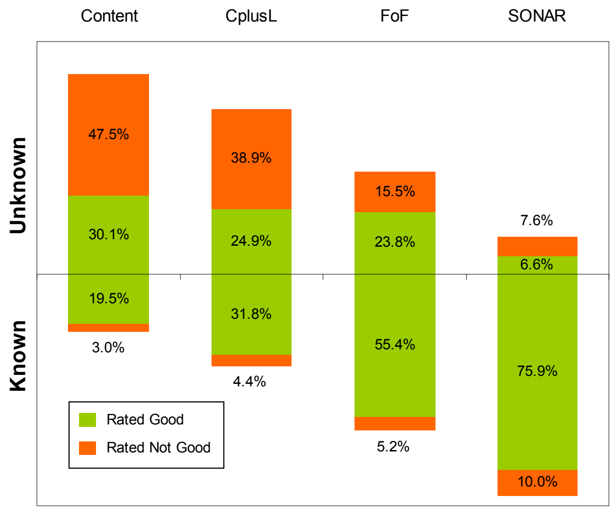


Figure 1. Known vs. unknown, Good vs. not good.

Overall, our users rated 82.5% of the SONAR recommendations as good, followed by 79.2% for the friend-of-friend, 56.7% for the content-plus-link and 49.6% for the content matching algorithm (F[3,705] = 69.1, p < .001). While there was no significant difference between SONAR and friend-of-friend, post-hoc comparison (LSD) showed that they have a significantly higher percentage of good recommendations than the two content-based algorithms (p < .001). Also, the percentage of "good" recommendations from the content-plus-link algorithm is significantly higher than basic content matching (p < .005). Overall, this suggests that the more known recommendations an algorithm produces, the more likely users are to consider those recommendations good.

When looking only at recommendations of known people (Figure 1, below the center line), we can see that most of those recommendations were considered good for all algorithms (around 90% for each algorithm). In other words, users considered recommendations of known people to be good, no matter how they were computed and what kinds of explanations were provided. Indeed, user feedback for recommendations rated as good and known, across algorithms, cited not the specific explanation provided but how good the recommendation was based on how well they knew the person: "*Very nice catch here. I know [name] from when he first interviewed at Watson (and went to Almaden),*" "*[name] is a good friend and previous colleague I would like to stay connected with.*"

In contrast, the situation for unknown recommendations is very different in that more recommendations are considered to be not good. The number of "not good" recommendations increases from right to left, i.e. the content-based algorithm produces the highest number of recommendations not considered good. One could argue that the more strangers an algorithm recommends, the more likely users will reject or not like the recommendations. We did sometimes find recommendations being rated as not good for this reason alone: "*I'd prefer to know them before being introduced to another stranger in the same city,*" and "*I generally want to know someone at least by reputation or interaction before making a connection*."

However, not knowing a person was not always an obstacle to rating a recommendation as good. The content matching algorithm also produced the highest number of good unknown recommendations, i.e. an average user found 30.1% recommendations to be both good and unknown at the same time. Content-plus-link and friend-of-friend algorithms followed with 24.9% and 23.8%, respectively, followed by SONAR with only 6.6% good unknown recommendations (F[3,705] = 37.1, p < .001). Post-hoc comparison (LSD) showed that the content matching had a significantly higher percentage than the other three algorithms (p < .05), and SONAR had a significantly lower percentage than the other three (p < .001).

While unknown recommendations were not consistently rated good, users did provide positive feedback about some unknown recommendations ("*good find, I'll comment on his favorite music hive5*" and "*Connected to lots of the same folks; I should know her!*"), leaving us unable to draw a simple conclusion as with known recommendations. Therefore, we analyzed the user feedback looking specifically for themes related to "good" recommendations. Users found all kinds of recommendations and explanations valuable: "*I find the recommendations based on tags (or keywords?) or non-direct shared connections most interesting*" and "*Useful – especially the mutual connections links*." Users also confirmed our intuition that explanations were not only helpful but necessary: "*I connect to people for a wide variety of contexts but not just because…,*" "*Always state why you are recommending someone,*" and "*I have to have a legitimate reason to connect to someone.*"

In particular, there seemed to be a minimum threshold of information necessary to rate a recommendation as "good.", and we heard this when that threshold was not met: "*Her* *profile did not have enough interesting items for me to do a 'cold call' at this time*" and "*The keywords in common caused me to at least look at his profile. We share some interests, but not enough to get connected at this time. Good try*." All algorithms suffered from this at times, whether by not providing enough information ("*the matching of only one keyword is a bit low for making recommendations, isn't it?*") or not enough information of value ("*I am not close with those 5 mutual connections…*"). The keywords in particular seemed to suffer from the latter and were often considered "*random*", "*irrelevant*," "*WWWAAAAYYYY too much noise*," or "*too generic to be helpful*." Nevertheless, the users spoke more highly of keywords when they were coupled with network relationships, as in the Content-Plus-Link algorithm: "*Similar to [..] Facebook [..] This is richer, since it ties into common interests/tags*," and "*At least two keyword matches (with keywords that really interest me) and only one degree of separation -- that to me is a good connection.*" And more generally of their need to be coupled with more information: "*recommendations must go beyond tagging and be multi-dimensional*." Obviously this threshold is different for different people, as one user said, "*Do NOT use obvious connections, i.e. People Management Relationships*" while another thinks "*Org[anizational] structure recommendations are great.*" But, in general, users seemed to want the recommender to provide as much potentially useful information as possible, of all types, to help them decide whether a recommendation was good.

Immediate actions resulted from recommendations

For every recommendation, users were also able to take an immediate action as described in our survey above. Figure 2 shows the percentages of connection and introduction actions for each algorithm per user compared to the overall percentage of recommendations rated good.

As expected, the number of actions taken on recommendations for different algorithms follows the same trend as the number of good recommendations. The majority of good recommendations resulted in either direct connection or introduction requests. On average 66.0% of the recommendations from the SONAR, 57.1% from the friend-of-friend, 42.4% from the content-plus-link and 32.8% from the content matching algorithm resulted in actions ($F[3,705] = 63.8$, $p < .001$). Post-hoc comparison (LSD) showed that every algorithm is different from all others ($p < .001$).

Note that for all algorithms, the percentage of recommendations resulting in actions is consistently lower than the percentage of good recommendations, i.e. a good recommendation does not necessarily result in an action. There are a number of possible explanations. For example, users might consider known people as good recommendations simply because the algorithm found a known person. However, that does not necessarily mean that a user would consider the person a friend. Or, as for unknown people recommendations, users might find them good but nonetheless not be interested in contacting to those people at that moment. Qualitative feedback from our survey is supportive of both explanations: "*Lots of people I know through being on the same program I don't interact with enough to want them in my contacts list*," or "*Because of the strength of mutual connections, I feel this is a good recommendation. I simply do not choose to connect at this time*." And that the threshold for an action is even higher than that for rating a recommendation as good: "*I'd be interested in checking out his profile, but probably not in connecting*," and "*I have a large network already. It is difficult to keep up with existing critical contacts. Need to be very judicious in discerning value proposition of new contacts*."

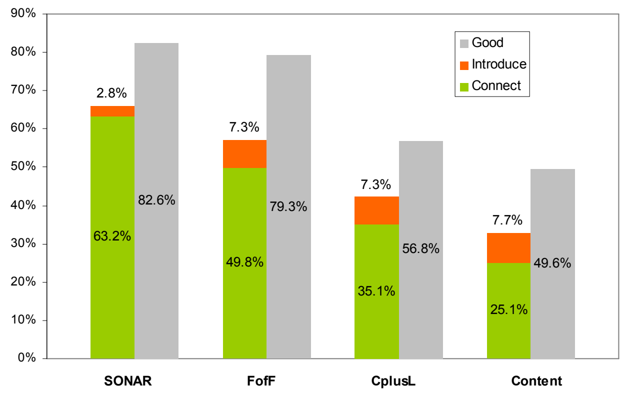


Figure 2. Good recommendations that resulted in actions.

We looked at individual feedback for recommendations marked as "good" or "known" that did not result in a connection action, as well as other "anomalous" responses such as marking a recommendation as known but not good. We found a variety of common reasons across these cases, including many where users said the recommended person had resigned, retired, or otherwise left IBM. There were other cases where those higher in the organizational structure were obviously known, but connection was not an option: "*I wouldn't connect to an executive without a personal relationship or a compelling reason,*" "*I feel awkward connecting to my 2^{nd} line*," and "*This person is [an] influential senior executive – using this criteria for connection would result in 1000's of people having access to this person.*" Some users described how they knew the recommended people, and more specifically knew those users have a reputation for "collecting colleagues" on Beehive, and thus decided not to connect.

We had originally sought to offer the introduction option to facilitate friending, especially for recommendations of unknown people. Our data show that most introduction requests were indeed made to unknown people. It is interesting to note that the ratio of introduction versus connection actions is higher for algorithms that produce a high number of unknown people recommendations. From right to left in Figure 2, 23.5% of all actions taken on the content matching algorithm were introductions, followed by 17.2% for content-plus-link, 12.8% for friend-of-friend, and 4.2% for SONAR. And we did receive positive feedback in cases in which introductions were used: "*Neale has one of those job roles that mean he might come in handy to know someday…,*" "*This is an interesting recommendation, I would like to see how this progresses*," and even "*I'm coming around to the idea that Beehive could be a decent mentor connection system.*"

However, the low usage of introductions and additional user feedback suggests that even the two actions we offered, introduction and connection, were not granular enough. Users suggested a number of alternate actions for a good person recommendation: "*I wouldn't mind being able to save people of interest that I could work on meeting through more conventional means*," "*maybe have a way to separate my 'top shelf' connections from more casual ones,*" or "*Need some method to specify interesting candidates – kind of like virtual speed dating*." Many suggested the use of a third person: "*I would prefer if some one I know introduces a new person to me (e.g. Linked In),*" or "*Potentially a good recommendation, but I would not connect to her [..] without a person in common that recommended her to me*."

One final theme that emerged was that the recommender's quantitative assessment might be overly strict in terms of success. For cases where a recommendation was rated as not good, unknown, and produced no action it would be judged as a failure. However they were not necessarily seen as such by some users who were merely undecided at rating time ("*I do not know whether I will connect to this person or not*,"). And particularly that the recommendation might be useful at a later time: "*Might be a good connection in the future for job networking*," or "*It would be a good recommendation if I had a need for such a person*."

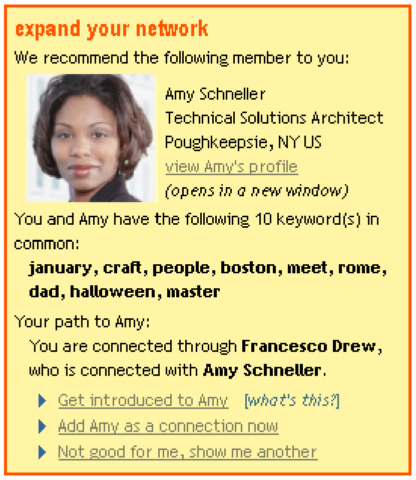
EXPERIMENT II: CONTROLLED FIELD STUDY

The first experiment invited users to participate in a survey. In our second experiment, we wanted to test the algorithms in a more natural setting. We deployed the four different recommender algorithms to a larger group of users on the site. The deployment of different recommenders allowed us to investigate how they can actually help people find friends during daily usage. And by comparing these users to a control group that did not receive recommendations we can test their effectiveness in increasing the number of friends and their impact on overall user activity on the site.

Methodology

For this experiment we randomly selected 3,000 users using similar criteria as in Experiment I. This time we required users to have logged into Beehive during the preceding 60 days instead of 1 week. The experiment was carried out as a between-subjects study during a 3-week period. We divided the 3,000 users randomly into 5 groups, each with 600 users. Four of the five groups were experimental groups, each one getting recommendations from a single algorithm only, while the remaining 600 subjects were a control group that did not get any recommendations. As in Experiment I, we guaranteed at least 10 recommendations of each type for all users, though those in the experimental groups only saw recommendations of one type. Also, whenever possible we computed more than 10 recommendations, up to a total of 30 for each user.

During the experiment, users in the experimental groups saw a new recommender widget on their Beehive home page, as shown in Figure 3.



expand your network

We recommend the following member to you:

Amy Schneller
Technical Solutions Architect
Poughkeepsie, NY US
view Amy's profile
(opens in a new window)

You and Amy have the following 10 keyword(s) in common:

january, craft, people, boston, meet, rome, dad, halloween, master

Your path to Amy:

You are connected through **Francesco Drew**, who is connected with **Amy Schneller**.

- Get introduced to Amy [what's this?]
- Add Amy as a connection now
- Not good for me, show me another

Figure 3. People recommender widget on Beehive showing a recommendation generated by the CpL algorithm.

The widget shows one recommendation a time, starting from the highest ranked ones. Each recommendation shows photo, job title, and work location of the person and the explanation generated by the algorithm. Users can also open a separate window to check the profile of the person. Users can respond to the recommendation by choosing one of three actions: connect to the person, ask to be introduced, and decline by choosing "not good for me". After responding, the widget will refresh and show the next recommendation. The widget also refreshes each time a user visits their home page to increase their chances of viewing different recommendations. In the email messages sent to users as part of their normal daily/weekly updates of Beehive, we included a personalized recommendation as well as a link that would take them to their home page and show that recommendation, allowing them to respond.

To balance the extra attention that recommendations get by occupying prime real estate in the experimental groups, we advertised various friending features and actions in the control group at the same place in the user interface and through email notifications.

Results

Of the 3,000 users, 1,710 users logged in during the experiment. 620 users of those in the experimental groups participated by responding to 7,451 recommendations. Of those 620, 122 were from the content matching group, 131 from the content-plus-link group, 157 from the friend-of-friend group, and 210 from the SONAR group.

Effectiveness of recommender algorithms

We measured user responses to our recommendations in a way similar to our survey. As expected, the per-user percentage of recommendations resulting in connection actions for different algorithms (shown in Table 2) follows the same trend as in the survey (F(3,413) = 17.6, p < .001). Post-hoc comparison (LSD) showed that SONAR again has a significantly higher connection action rate than the other three algorithms (p < .005). The connection action rate of the friend-of-friend algorithm is also significantly higher than the content matching algorithm (p < .001).

SONAR	FoF	CplusL	Content
59.7%	47.7%	40.0%	30.5%

Table 2. Recommendations resulting in connect actions.

It is worth noting that, in contrast to the survey, users rarely chose the introduction option as a response - less than one percent of the 7,451 responses were introduction requests. One possible reason for the difference could be that, while we explained directly the introduction feature in the survey, here users had to mouse-over the "what is this?" link as shown in Figure 3 to see the explanation. It seems that instead of mousing-over as we had hoped, many chose not to bother and simply ignored the feature.

Separate from responding, users can also click a link in our widget to view the profile of the recommended person. How often users did this is interesting insofar as it might indicate the interestingness and possibly the novelty of a recommendation, since a user may choose not to view the profile because of either a lack of interest or familiarity with the person. Because the content matching algorithm recommends mostly unknown people, we expected a higher number of such views for that algorithm compared to other algorithms. Indeed, for each user 8% of content matching recommendations resulted in such immediate profile views compared to only 2.9% for SONAR. The difference in view-profile percentage is significant (F[3,605] = 7.0, p < .001). Post-hoc comparison (LSD) showed that the content matching and CplusL algorithms have significantly higher percentages than the other two (p < .05) [5].

Impact of people recommendations

The immediate goal of recommending people on a social networking site is to increase a user's network of friends. We compared the number of friends before and after the experiment in each group and found a significant *group x before/after* interaction effect (F[4,2995] = 15.0, p < .001). Post-hoc comparison (LSD) showed that all our algorithms significantly increased the number of friends compared to the control group. SONAR was most effective with an increase of 13% (3.64 more friends on average per user), followed by the other algorithms as shown in Figure 4. We also saw an increase of 5% (1.27 more friends on average per user) in the control group, which can possibly be attributed to the advertisement of friend-related features.

We had also expected that people recommendations would impact user activity on the site in general. Indeed, when comparing the number of page views on Beehive during the 3-week period of the experiment with the 3-week period before the experiment, we found that users in experimental groups viewed an average of 13.7% more pages during the experiment (3.13 more page views), while control group users viewed 24.4% less pages during the experiment (6.34 less page views). Note that the overall page views across the entire site dropped by 27.5% during that period. We have found the *experiment/control x before/after* interaction effect to be significant (F[1,2998] = 9.2, p < .005), i.e. people recommendations were effective in increasing browsing activity of users. We also observed an increase in content and comment creation in the experimental groups compared to the control group, although the number of items created in the 3-week period per user was too low to observe any significant difference.

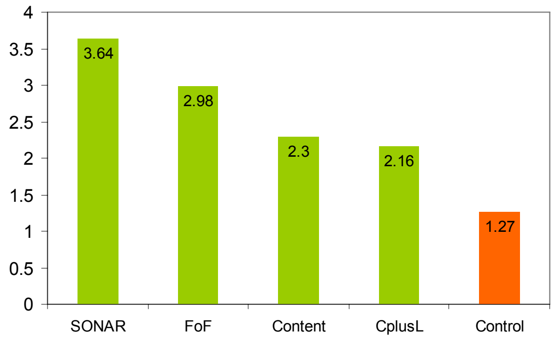


Figure 4. Increase in number of friends.

DISCUSSION AND CONCLUSION

The results from both the personalized survey and the large field study on Beehive, show that the four algorithms we compared are effective in making people recommendations and can significantly increase the number of friends of a user on the site. This not only furthers our understanding in practical social matching systems [21], but also enables us to understand the effectiveness and characteristics of different information sources and algorithms for the purpose of people recommendation.

When comparing algorithms, we can roughly put our four algorithms into two categories. Those based more on social relationship information (FoF and SONAR), and those based more on content similarity (Content and CplusL). In our experiment, relationship based algorithms outperform content similarity ones in terms of user response. This result could partly be attributed to the fact that both content similarity algorithms employ a simple keyword matching scheme, whereas SONAR takes advantage of the rich relationship data in IBM that might not be available to that extent elsewhere. As a result, in cases where more sophisticated content similarity can be computed and

[5] Note that in order to reduce noise in the estimate of per-user connection rate, we excluded users with less than 4 recommendation responses in its calculation. Similarly for per-user view-profile rate, users who have been shown less than 4 recommendations were also excluded.

relationship information is less available, the advantage of relationship-based algorithms might not be as large as observed in this work.

The results described in this paper also show that relationship-based algorithms are better at finding known contacts whereas content similarity algorithms were stronger at discovering new friends. As shown in Figure 1, the more relationship information an algorithm uses the more known contacts and the less new friends it discovers. In general, this suggests that on social networking sites, relationship-based algorithms would perform particularly well for newer users in finding known offline contacts that have not yet been added to their online social network. In particular, FoF can expand their contact list from a few existing contacts, while a SONAR-like aggregation can take advantage of additional data, including commenting, tagging, or organizational relationships, which are often available within organizations. However, for more established users, relationship-based algorithms would either run out of people to recommend or base themselves on social relationships that are too weak to be meaningful. In contrast, content similarity algorithms will still be able to find new interesting people. Hence, one potentially promising way to combine the strengths of both types of algorithms is to leverage relationship based algorithms initially to build up a network quickly by finding known people and, as the network grows, complement them with content similarity based algorithms. Such an approach might even have an additional benefit of increasing new users' trust in the system because, as indicated in [20], people trust recommenders more if they see familiar items recommended.

For future research, beyond developing better recommender algorithms, one could look into new applications of people recommendations on social networking sites, such as leveraging them for recommending content, based on the intuition that "if I like that person I might also be interested in his/her content." Another possibility is to investigate whether people recommendations can help bootstrap newcomers, addressing adoption issues of social networking sites.

REFERENCES

1. Boyd, d. m., & Ellison, N. B. 2007. Social network sites: Definition, history, and scholarship. Journal of Computer-Mediated Communication, 13(1), article 11.
2. Deerwester, S., Dumais, S., Furnas, G.W., Landauer, T.K., Harshman, R. 1990. Indexing by Latent Semantic Analysis. J. of Amer. Soc. Info. Sci. 41 (6): 391–407.
3. DiMicco, J., Millen, D., Geyer, W., Dugan, C., Brownholtz, B. 2008. Motivations for Social Networking at Work. ACM CSCW'08.
4. Ehrlich, K., Lin, C., and Griffiths-Fisher, V. 2007. Searching for experts in the enterprise: combining text and social network analysis. Proc. Group'07, 117-126.
5. Geyer, W., Dugan, C., Millen, D., Muller, M., Freyne, J. 2008. Recommending Topics for Self-Descriptions in Online User Profiles. ACM RecSys'08.
6. Granovetter, M. 1973. Strength of weak ties. Amer. J. Sociology 78 (1973), 1360-1380.
7. Groh, G., & Ehmig, C. 2007. Recommendations in Taste Related Domains: Collaborative Filtering vs. Social Filtering. Proc. ACM Group'07. 127-136.
8. Guy, I., Jacovi, M., Meshulam, N., Ronen, I., Shahar, E. 2008. Public vs. Private – Comparing Public Social Network Information with Email. ACM CSCW'08.
9. Guy, I., Jacovi, M., Shahar, E., Meshulam, N., Soroka, V., Farrell, S. 2008. Harvesting with SONAR: the value of aggregating social network information. Proc. ACM CHI'08. 1017-1026.
10. Herlocker , J. L., Konstan, J.A. , Riedl, J. 2000. Explaining collaborative filtering recommendations. Proc. ACM CSCW'00. 241-250.
11. Hofmann, T. 1999. Probabilistic Latent Semantic Analysis. UAI'99.
12. McDonald, D. W. 2003. Recommending collaboration with social networks: a comparative evaluation. Proc. of ACM CHI'03, 593-600.
13. Mooney, R. J., & Roy, L. 2000. Content-based book recommending using learning for text categorization. Proc ACM DL'00.195–204.
14. Official Facebook Blog: http://blog.facebook.com/blog.php?post=15610312130.
15. Pazzani, M. J., Muramatsu, J., & Billsus, D. 1996. Syskill webert: Identifying interesting web sites. AAAI/IAAI, Vol. 1, 54-61.
16. Porter, M.F. 1980. An algorithm for suffix stripping. Program, 14(3). 130-137.
17. Salton, G & Buckley, C. 1988. "Term-weighting approaches in automatic text retrieval". Information Processing & Management 24 (5): 513-523.
18. Sen, S., Geyer, W., Muller, M., Moore, M., Brownholtz, B., Wilcox, E., & Millen, D.R. 2006. FeedMe: a collaborative alert filtering system. Proc. ACM CSCW'06. 89-98.
19. Spertus, E., Sahami, M., and Buyukkokten, O. 2005. Evaluating similarity measures: a large-scale study in the Orkut social network. Proc. SIGKDD'05. 678-684.
20. Swearingen, K. and Sinha, R. 2002. Interaction Design for Recommender Systems. Proc. DIS'02.
21. Terveen, L. and McDonald, D. W. 2005. Social matching: A framework and research agenda. ACM Trans. Comput.-Hum. Interact. 12, 3. 401-434