Expertise Ranking in Question-Answer Social Network Groups

Chokchai Puttan and Twittie Senivongse
Department of Computer Engineering, Faculty of Engineering
Chulalongkorn University
Bangkok, Thailand
chokchai.p@student.chula.ac.th, twittie.s@chula.ac.th

***Abstract*—Online social networks have become the major channel for people to maintain relationships, collaborate, and contribute shared information. A social network group can be created as a specific community for people who share an interest in a particular topic. One form of interaction within the group is the question-answer interaction by which the users in the group can ask questions or provide answers to others. A network analysis method, e.g., PageRank, can be used to analyze the interaction patterns between the users in order to identify and rank experts in the group. In this paper, we are interested in experimenting on how the quality of the users' comments can take part in the identification and ranking of experts by a PageRank-like algorithm. The quality factors are community rating, that is given to the answers, and the content-based features of the comments, i.e., length, complexity, and informativeness. We conduct an experiment on a Java Facebook group and evaluate the accuracy of the ranking with and without comment quality consideration against expertise ranking by Java experts.**

Keywords—social network analysis; expertise ranking; expert finding; community rating; content-based features

I. Introduction

Online social networks have become the major channel for people to maintain relationships, collaborate, and contribute shared information. A social network group can be created as a specific community for people who share an interest in a particular topic. One form of interaction within the group is the question-answer interaction by which the users in the group can ask questions or provide answers to others. Usually the users do not know each other but they are willing to participate in the group to share their knowledge and skills to help other users, to learn from others, and to gain respect in the community.

Alice is a Java developer and is having a problem concerning database connection. She first uses a search engine to search for the answer but cannot find the right one that can solve her problem. She is a keen Facebook user, so she decides to post the question to a Facebook group about Java issues. After a while, several users respond to her with comments. However, the comments are of different nature. Some explain what the problem is. Some explain how to solve it. Some ask for more information. Some mention having similar experience. Some even conflict with other comments. Alice is confused and does not know which comment to believe. She ends up trying several things as suggested. In this scenario, Alice finds it difficult to get to the right solution since there are many comments of different quality. If only she knew her helpers' expertise she could find out what to do with her problem more easily.

Expert finding and ranking has long been an issue for online communities. A number of approaches are proposed in the question-answer context and most of them identify experts from their activities in, for example, email correspondence [1], discussion forums [2], and question-answer portals [3]. This paper presents a method to identify experts in a question-answer social network group and rank their expertise. In this paper, we are interested in the conversations in the Java Facebook group [4]. The method is influenced by a PageRank-like algorithm in [2] which is based on the well-known PageRank algorithm for ranking Web pages [5]. It considers the activity level of the users in answering or commenting on any posted questions. In particular, we are interested in experimenting on how the quality of the comments can take part in the expert identification and ranking. The quality factors are 1) Facebook's like count as the community rating that is given to the comments, and 2) content-based features of the comments, i.e., length, complexity, and informativeness [6]. The like count reflects the quality of a comment as perceived by other users, and the features of a comment reflect the willingness to help, knowledge, and experience of the answerer, and hence the quality of the content. We evaluate the accuracy of the method with and without comment quality consideration against expertise ranking by Java experts.

Section II of this paper discusses related work and Section III describes our expertise ranking method. Section IV presents an evaluation of the method and the paper concludes in Section V with topics for future work.

II. Related Work

Expert finding and ranking has long been an issue in online communities and in different contexts. Recently social networks have become the places for user-contributed information, and hence the platforms for expertise networks and expert identification for particular topics. For example, Fu and Dong [7] use link and semantic structures of tags to identify experts in a social tagging system. Li et al. [8] discover a user with expertise in a topic by analyzing the user's topic-specific activity and other users' feedback behavior on the user's contributed information. Wu et al. [9] automatically recommend developers who are considered experts and should be able to fix bugs reported to a bug repository of an open source project. This is done by analyzing similarity between the reported bugs and the information in the historical bug reports as well as the comments associated with the reports. Minami [10] presents an information network analysis in the university library domain to estimate the expertise level of books based on the expertise of the borrowers and how long and how frequent the books are borrowed.

Here we focus on the question-answer context in online communities. The former work by Dom et al. [1] identifies experts in an organization by mining email correspondents. To enhance mining text messages, they analyze communication patterns between email correspondents. They build a weighted directed graph representing the flow of information among email correspondents and study several graph-based ranking algorithms, including PageRank and HITS for ranking Web pages, when using them to rank the expertise level of people on subjects of interest. Chen and Nayak [3] argue that the widely used link analysis algorithm like HITS is vulnerable to a spam problem when being used to analyze a question-answer portal. To overcome this, they propose a structural analysis on Yahoo! Answer and a method to determine expertise scores for the answerers based on their reputation that reflects their usage patterns on the portal. The method can determine a local score for a certain category of posting or a global score across all categories.

Our work is motivated greatly by the research of Zhang et al. [2]. They analyze the posting-replying threads in Java programming forum, using a number of network analysis methods, in order to identify users with high expertise. In particular, they map the posting-replying relationships to a directed graph called a community expertise network and propose a PageRank-like algorithm, called ExpertiseRank, to classify the users into five levels of expertise. We apply their approach to the Java Facebook group, but since their approach considers only the communication patterns between the users, we add to it by also considering quality of the comments in our experiment. Hsu et al. [6] propose quality criteria for user-contributed contents in online communities, some of which we can adopt. Their criteria consist of comment visibility, user reputation and influence, and content-based features. Using these criteria, they apply Support Vector Regression to rank comments in the social news aggregator Digg.

III. Expertise Ranking Method

The expertise ranking method comprises preparation of conversation data, calculation of preliminary expertise scores, calculation of expertise scores, and expert identification and expertise ranking.

A. Preparation of Conversation Data

The first step is to import all post-comment threads, or conversation data, from the Java Facebook group [4]. The import program is developed using nodeJS [11] and Facebook API [12]. The conversations are acquired as feed data in JSON format and stored in a MySQL database. The stored data consist of post ID, post message, questioner ID of a post, comment ID, comment message, answerer ID of a comment, post ID associated with a comment, and like count of a comment. The Java Facebook group is an open group and has 41,717 member users, 12,269 posts, and 56,013 comments as of 8 January 2013.

B. Calculation of Preliminary Expertise Score

We determine the preliminary expertise score of a user on the basis of the quality of the comments that the user has responded to the questions. If the comments are of good quality, the user is likely to have expertise.

1) Quality of a comment

We consider the like count and three content-based features in [6].

a) Like count: The like count of a comment is the number of positive votes the user gets from other users. This rating says that the comment is useful and well-received by a number of community members.

b) Comment length: The length of a comment is measured by the number of words or terms in the comment. A long comment can show the willingness of the answerer to help and a sign that the answerer has experience and knowledge about the topic that is asked.

c) Comment complexity: A complex comment can be a sign of the answerer's ability to understand and explain difficult and complicated ideas. The complexity of a comment c_j can be measured by the *entropy* of the comment as in (1):

$$entropy(c_j) = \frac{1}{\lambda}\sum_{i=1}^{n} p_i[log_{10}(\lambda) - log_{10}(p_i)] \tag{1}$$

where λ = number of terms in comment c_j

n = number of distinct terms in comment c_j

p_i = frequency of each term i.

d) Comment informativeness: Informativeness captures the uniqueness of the content of the comment relative to that of other comments in the same conversation. The uniqueness can be a sign of different ideas or novel solutions to the posted question. The informativeness *inform* of a comment c_j can be measured by a variation of the term frequency and inverse document frequency (TF-IDF). TF values terms that occur frequently in a comment whereas IDF values terms that occur infrequently across comments. The formulas are in (2)-(4):

$$inform(c_j) = \sum_{t_i \in c_j} tf_{ij} * idf_i \tag{2}$$

$$tf_{ij} = \frac{n_{ij}}{\sum_k n_{kj}} \tag{3}$$

$$idf_i = log\frac{|C|}{|\{c : t_i \in c\}| + 1} \tag{4}$$

where tf_{ij} = term frequency of term i in comment j

n_{ij} = frequency of term i in comment j

n_{kj} = frequency of each term k in comment j

idf_i = inverse document frequency of term i

$|C|$ = number of comments in a conversation

$|\{c : t_i \in c\}|$ = number of comments in a conversation in which t_i appears.

2) Preliminary expertise score

Each feature discussed above refers to the quality level in each aspect of a single comment, and hence it is considered a rough estimation of the expertise of a user who gives that comment. To calculate the preliminary expertise score of a user, we propose the following equations to determine the quality of all comments that the user replies in a number of conversations within the Java Facebook group.

a) Quality level of a user with regard to a conversation: We first have to determine the quality of all comments the user replies in a conversation.

- *Like count of a user for a conversation:* In a conversation, a user may respond with one or more comments. Some comment may have a like count and some may not. The like count l_{ij} of a user j with regard to a conversation i in which the user participates is computed by (5):

$$l_{ij} = \frac{(\sum_{j \in i} lc_j) + 1}{pl_i + m_i} \tag{5}$$

where lc_j = like count of a comment of user j in conversation i

pl_i = total like count of all users in conversation i

m_i = number of users giving comments in conversation i.

Note that (5) considers the case when the user j does not have the like count and when none of the comments in the conversation have the like count. We assume that when users respond to a question, they should have some knowledge in the topic and should get some credit.

Given a conversation as an example below:

"Question: Can someone explain the difference between Interfaces and Abstract Class?

User A: (like count = 2) Abstract Class can contain another methods not Abstract but the Interface all its methods are abstract.

User B: (like count = 1) take a look at this http://docs.oracle.com/javase/tutorial/java/IandI/abstract.html."

The like count of the user A for this conversation (l_{iA}) is (2+1)/((2+1)+2) = 0.60.

- *Comment length of a user for a conversation:* The comment length ln_{ij} of a user j with regard to a conversation i in which the user participates is computed by (6):

$$ln_{ij} = \sum_{j \in i} cl_j \tag{6}$$

where cl_j = number of terms in a comment of user j in conversation i.

- *Comment complexity of a user for a conversation:* The comment complexity e_{ij} of a user j with regard to a conversation i in which the user participates is computed by (7):

$$e_{ij} = \sum_{j \in i} entropy(c_j) \tag{7}$$

where $entropy(c_j)$ = complexity of a comment of user j in conversation i as computed by (1).

- *Comment informativeness of a user for a conversation:* The comment informativeness t_{ij} of a user j with regard to a conversation i in which the user participates is computed by (8):

$$t_{ij} = \sum_{j \in i} inform(c_j) \tag{8}$$

where $inform(c_j)$ = informativeness of a comment of user j in conversation i as computed by (2).

b) Preliminary expertise score of a user with regard to a conversation: The preliminary expertise score $pesc_{ij}$ of a user j with regard to a conversation i is an average of the normalized values of all quality aspects with regard to that conversation. It is computed by (9):

$$pesc_{ij} = \frac{sl_{ij} + sln_{ij} + se_{ij} + st_{ij}}{4} \tag{9}$$

where sl_{ij} = normalized like count of user j with regard to conversation i, and

$$sl_{ij} = \frac{l_{ij}}{L_i}, l_{ij} \text{ is computed by (5) and } L_i \text{ is } max(l_{im}) \text{ where user } m \in i \tag{10}$$

sln_{ij} = normalized comment length of user j with regard to conversation i, and

$$sln_{ij} = \frac{ln_{ij}}{LN_i}, ln_{ij} \text{ is computed by (6) and } LN_i \text{ is } max(ln_{im}) \text{ where user } m \in i \tag{11}$$

se_{ij} = normalized comment complexity of user j with regard to conversation i, and

$$se_{ij} = \frac{e_{ij}}{E_i}, e_{ij} \text{ is computed by (7) and } E_i \text{ is } max(e_{im}) \text{ where user } m \in i \tag{12}$$

st_{ij} = normalized comment informativeness of user j with regard to conversation i, and

$$st_{ij} = \frac{t_{ij}}{T_i}, t_{ij} \text{ is computed by (8) and } T_i \text{ is } max(t_{im}) \text{ where user } m \in i \cdot \tag{13}$$

c) Preliminary expertise score of a user: Once we have determined the preliminary expertise score of a user with regard to a conversion, we then compute the preliminary expertise score pes_j of a user j with regard to all conversations in the Java Facebook group in which the user participates. It is computed by (14):

$$pes_j = \sum_{i \in P_j} pesc_{ij} \tag{14}$$

where $pesc_{ij}$ = preliminary expertise score of user j with regard to conversation i as computed by (9)

P_j = set of conversations in which user j participates.

Note that a user who posts questions but never replies to other users' posts will have $pes = 0$.

C. Calculation of Expertise Score

To calculate the expertise score of a user, we first build a community expertise network [2] to represent communication patterns between the users in the Java Facebook group. Then a PageRank-like algorithm is applied to the community expertise network.

a) Community expertise network: Using the patterns of conversations within the group, we can build a question-answer network by representing each user in any conversation as a node and an interaction from a questioner to an answerer as an edge. Fig. 1 shows, on the left, the question-answer interaction between users A, B, C, D, and E, and the posts W, X, Y, and Z on which these users converse. A dotted arrow represents a relationship between a questioner and a posted question and a solid arrow represents a relationship between an answerer and a posted question. Given this interaction, a community expertise network is created as on the right of Fig. 1. The user A's question, for example, is responded by the comments from the users B, C, and D.

b) Expertise Ranking: The direction of the edges in the community expertise network can indicate that a user answering a question posted by another has more expertise in the topic than the questioner. The expertise is distributed along the network of responses. For example, a user who replies to a question posted by another user who in turn responds to a question posted by a different user should be considered as having the greatest expertise among the three. This behavior is analogous to the well-known PageRank algorithm [5]. The PageRank algorithm ranks a Web page by considering not only the number of Web pages pointing to it, but also the number of Web pages linking to those pages, and so on. This indicates that a link from a popular Web page (i.e., it is pointed to by many pages) is given a higher weight than one from an unpopular page. In a similar manner, the expertise score of a user is dependent on the expertise score of others whom the user has helped. Expertise ranking in [2] therefore follows PageRank.

Based on the original PageRank, the expertise score ER of a user j can be computed by (15):

$$ER(j) = (1-d) + d\left(\frac{ER(U_1)}{C(U_1)} + \ldots + \frac{ER(U_n)}{C(U_n)}\right) \tag{15}$$

where $ER(U_i)$ = expertise score of user U_i whom user j has helped

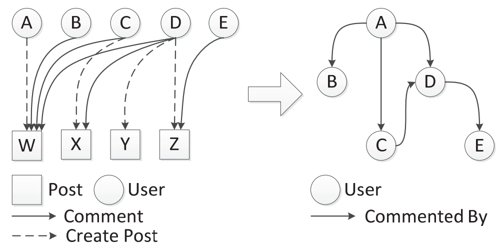

Fig. 1. Bipartite graph of users and posts that is mapped to directed graph representing community expertise network [2].

$C(U_i)$ = number of users who answer the questions posted by U_i

d = damping factor which is the probability for a random walker in the network iteratively following links to this user j's node (i.e., 1-d is the probability for a random walker jumping to this user j's node after stopping following links); d is set to 0.85 [2].

The expertise score computation is iterative, requiring several passes or iterations to adjust the approximate scores so that they get closer to the true scores. In the first iteration, the expertise score ER of a user Ui whom a user j has helped is computed by (16):

$$ER(U_i)_{initial} = \frac{1}{N} \tag{16}$$

where N = number of users in the community expertise network.

In each subsequent iteration, the expertise score ER of a user Ui whom a user j has helped is the $ER(U_i)$ from the previous iteration.

Since we are interested in experimenting on how the quality of the comments can take part in expertise ranking, we aim to influence the ranking of certain users by assigning them the special initial ER values, instead of using (16), which in this case are their preliminary expertise scores. However, according to [13], assigning such special initial values has no effect on the results of the computation when the original PageRank is used. To allow the preliminary expertise scores to influence the ranking, we have to use the modified PageRank [13]; the initial ER values are left intact but the modification is made to 1-d instead. (The rationale behind the modified PageRank is that a random walker will not jump to any Web page but will rather jump to certain Web pages with a higher probability than to others.) Hence the computation of the expertise score ER of a user j becomes (17):

$$ER(j) = S(j)(1-d) + d\left(\frac{ER(U_1)}{C(U_1)} + \ldots + \frac{ER(U_n)}{C(U_n)}\right) \tag{17}$$

where $$S(j) = N\frac{pes_j}{\sum_{m=1}^{N} pes_m} \tag{18}$$

= normalized preliminary expertise score of user j weighted by number of users in the community expertise network.

Fig. 2 shows an example of a community expertise network of five users with the given initial *ER* computed by (16) and *S* by (18). The expertise score of a user D in the first iteration of the computation is dependent on the initial *ER* scores of the users A and C whom D has helped. Using (17), we calculate *ER*(*D*) as:

$$ER(D) = 1.4(1-0.85) + 0.85\left(\frac{ER(U_A)}{C(U_A)} + \frac{ER(U_C)}{C(U_C)}\right)$$
$$= 1.4(1-0.85) + 0.85\left(\frac{0.2}{3} + \frac{0.2}{1}\right)$$
$$= 0.44$$

At the end of the computation, we obtain the expertise scores of all users in the community expertise network.

D. Expert Identification and Expertise Ranking

To identify experts and rank their expertise, we first filter out any users who only post questions but never help other users because there is no evidence that they are experts. In the resulting expert list, there are 2,564 users who have commented on the posts. We sort the expertise scores of these users in descending order and then equally divide this expert list into five levels of expertise as in Table I. Each of these users belongs to one of the expertise levels.

IV. Evaluation

To evaluate the accuracy of the method, we compare expert identification and ranking with human rating. We create a questionnaire that consists of 20 sample conversations. Each conversation contains a question and 2-5 answers. We ask ten experienced Java developers to identify and rank the experts. For each conversation, each developer specifies a rank (1 to 5) for each answerer and identifies which ones are not experts.

A. Accuracy of Expertise Ranking

Two measures are used to evaluate ranking accuracy [14] (Table II). For each conversation *i*, Spearman's rank correlation coefficient (ρ_i) is used to determine correlation between ranking by the method and ranking by a developer. ρ_i is between [-1, 1]; 1 indicates a perfect correlation of ranks, 0 indicates no correlation, and -1 indicates a perfect negative correlation. Also for each conversation *i*, Kendall tau rank distance (k_i) is used to measure distance (or disagreement) between the two ranking lists. k_i is between [0, 1]; 0 indicates identical ranking lists and 1 indicates maximum disagreement.

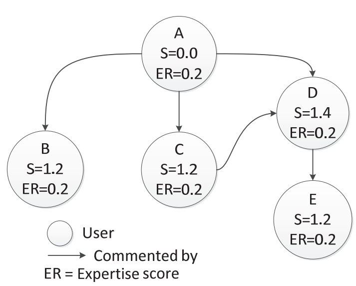

Fig. 2. Example of community expertise network with *S* and initial *ER*.

TABLE I. FIVE LEVELS OF EXPERTISE (ADAPTED FROM [2])

Level	Class	Description
5	Top expert	Knows very advanced concepts and techniques, can answer very difficult questions, and knows specific topics deeply.
4	Highly experienced user	Knows advanced concepts and techniques, can answer all or most questions, and knows some specific topics in detail.
3	Experienced user	Knows concepts and techniques at intermediate level, can program well, and is able to answer some difficult questions.
2	Learner	Still learns Java, knows basic concepts and techniques, can program, and can answer basic questions.
1	Novice	Begins to learn Java.

TABLE II. EVALUATION MEASURES FOR EACH CONVERSATION *I*

ρ_i	k_i	EIC_i
$\rho_i = 1 - \frac{6\sum d_j^2}{n(n^2-1)}$ (19) where d_j = difference of ranks of j^{th} answerer (or j^{th} rank pair) n = number of answerers	$k_i = \frac{D}{n(n-1)/2}$ (20) where D = number of answerers (or rank pairs) with different ranks n = number of answerers	$EIC_i = \frac{ic}{ic + iic}$ (21) where *ic* = number of correct identifications *iic* = number of incorrect identifications

B. Accuracy of Expert Identification

In our method, the answerers who have the expertise level 4 or 5 are identified as experts. We evaluate the identification results of the method against the developers' opinions for each conversation *i* in the questionnaire using EIC_i in Table II.

C. Results and Discussion

Table III reports the evaluation results, with regard to all conversations in the questionnaire, when expertise ranking considers the quality of the comments, i.e., (17) is used. Table IV reports the results of the case where the quality of the comments is not considered, i.e., (15) is used.

TABLE III. RESULTS WITH COMMENT QUALITY CONSIDERATION

Developer #	Average ρ	Average k	Average *EIC*
1	0.26	0.40	0.81
2	0.14	0.44	0.67
3	0.20	0.41	0.66
4	0.02	0.50	0.73
5	0.31	0.38	0.76
6	0.15	0.43	0.85
7	0.19	0.41	0.80
8	0.04	0.51	0.72
9	0.33	0.36	0.80
10	0.07	0.47	0.76
Overall Average	0.17	0.43	0.76

TABLE IV. RESULTS WITHOUT COMMENT QUALITY CONSIDERATION

Developer #	Average ρ	Average k	Average *EIC*
1	0.18	0.43	0.81
2	0.07	0.47	0.67
3	0.12	0.45	0.67
4	0.00	0.50	0.74
5	0.28	0.38	0.78
6	0.08	0.46	0.86
7	0.14	0.45	0.80
8	0.01	0.51	0.72
9	0.31	0.36	0.81
10	-0.03	0.51	0.76
Overall Average	0.12	0.45	0.76

The overall average of expert identification accuracy *EIC* is quite satisfactory for both cases whereas, for expertise ranking, the average of Spearman's ρ is positive but quite close to zero for both cases. This indicates that ranking by the method correlates with ranking by the developers but the association is quite weak. Using Kendall's k, ranking by the method moderately agrees with ranking by the developers. Even though the performance of both cases is not much different, the results show that comment quality consideration can give better ranking accuracy. The reason might be that, ranking by the developers also considers comment quality. The developers rank the answerers by comparing the contents of their comments, i.e., they examine comment quality, and not the interaction patterns of the users.

There are several factors that can hinder the method and the evaluation. It is seen that ranking 2-5 answerers in each conversation is more difficult a task than classifying answerers into just two classes of expert and nonexpert. This can hinder the evaluation for expertise ranking accuracy. In many cases, the differences between good expert comments and average comments are clear. But at the same time, the users do not repeat what the previous answerers in the conversations have replied, and they are likely to add certain points only. Therefore there may be no clear evidence of expertise in their comments. Some user may be an expert and his/her comments do show the expertise, but if he/she gives comments infrequently, the expertise score is affected.

We consider the evaluation here as giving approximate results because the expertise scores that we use to rank and classify the users are calculated from all conversations in which the users participate, whereas the developers are asked to rank and classify experts on the basis of individual conversations.

V. CONCLUSION

This paper presents the use of a PageRank-like algorithm to rank and identify experts in the Java Facebook group. The work experiments on how the like count and content-based features of the comments can take part in the algorithm. According to the evaluation results, there is a sign that comment quality can be of benefit to expertise ranking.

Our method can be improved in several ways. We can put a weight on each edge of the community expertise network to show how many times a user has helped another user in order to help refine his/her expertise score. We can add other quality aspects of the comments such as readability and user profile to the algorithm, and further analysis on the impact of each quality aspect on each expertise level should be conducted as well. Since comments are often not plain text but include code snippets, we should determine quality of text and quality of code separately. Question difficulty should also be considered; this may require an ontology of Java knowledge to help determine the difficulty level. Lastly, we expect to revise the evaluation of the method by using a larger set of sample data.

REFERENCES

[1] B. Dom, I. Eiron, A. Cozzi, and Y. Zhang, "Graph-based ranking algorithms for e-mail expertise analysis," Proc. 8th ACM SIGMOD Workshop Research Issues in Data Mining and Knowledge Discovery (DMKD 2003), San Diego, CA, June 2003, pp. 42-48.

[2] J. Zhang, M.S. Ackerman, and L. Adamic, "Expertise networks in online communities: structure and algorithms," Proc. 16th Int. Conf. World Wide Web (WWW 2007), Alberta, Canada, May 2007, pp. 221-230.

[3] L. Chen and R. Nayak, "Expertise analysis in a question answer portal for author ranking," Proc. 2008 IEEE/WIC/ACM Int. Conf. Web Intelligence and Intelligent Agent Technology (WI-IAT 2008), December 2008, Sydney, Australia, pp. 134-140.

[4] Facebook, Java Group, March 2013. [Online]. Available: https://www.facebook.com/groups/Javagroup123/

[5] L. Page, S. Brin, R. Motwani, and T. Winograd, The PageRank citation ranking: bringing order to the Web, Stanford Digital Library Technologies Project, 1998.

[6] C.F. Hsu, E. Khabiri, and J. Caverlee, "Ranking comments on the social Web," Proc. 2009 Int. Conf. Computational Science and Engineering (CSE 2009), Vancouver, Canada, August 2009, pp. 90-97.

[7] W.-T. Fu and W. Dong, "Facilitating knowledge exploration in folksonomies: expertise ranking by link and semantic strcutures," Proc. 2010 IEEE 2nd Int. Conf. Social Computing (SocialCom 2010), Minneapolis, Minnesota, August 2010, pp. 459-464.

[8] Y. Li, S. Ma, Y. Zhang, and R. Huang, "Expertise network discovery via topic and link analysis in online communities," Proc. 2012 12th IEEE Int. Conf. Advanced Learning Technologies (ICALT 2012), Rome, Italy, July 2012, pp. 311-315.

[9] W. Wu, W. Zhang, Y. Yang, and Q. Wang, "DREX: Developer recommendation with k-nearest-neighbor search and expertise ranking," Proc. 2011 18th Asia-Pacific Software Engineering Conf. (APSEC 2011), Ho Chi Minh, Vietnam, December 2011, pp. 389-396.

[10] T. Minami, "Expertise level estimation of library books by patron-book heterogeneous information network analysis – concept and applications to library's learning assistant service," Proc. 2012 26th Int. Conf. Advanced Information Networking and Applications Workshops (WAINA 2012), Fukuoka, Japan, March 2012, pp. 357-362.

[11] Node.js, March 2013. [Online]. Available: http://nodejs.org/

[12] Facebook API, March 2013. [Online]. Available: http://developers.facebook.com/docs/reference/apis/

[13] M. Sobek, The Yahoo Bonus and Its Impact on Search Engine Optimization, 2003. [Online]. Available: http://pr.efactory.de/e-pagerank-yahoo.shtml

[14] R. Fagin, R. Kumar, and D. Sivakumar, "Comparing top k list," SIAM J. Discrete Mathematics, vol. 17, no. 1, pp. 134–160, 2003.