Discovering Authoritative users in Question Answer Communities: A Case Study of Stack Overflow

Yue Qi[1, a], Jiuming Huang[2, b]

[1]School of Computer, National University of Defense Technology, Changsha, 410000, China

[2]School of Computer, National University of Defense Technology, Changsha, 410000, China

[a]email: qiyue7810097@sina.com, [b]email: jiuming.huang@qq.com

Keywords: Question Answer Communities; Stack Overflow; Authoritative users

Abstract. Question answer communities such as Stack Overflow and Yahoo! Answers are becoming more and more popular knowledge sharing communities, which have attracted great interest from both industry and academia. On such communities, people can ask questions of any kind and then wait for someone else to answer the questions. Nevertheless, in many cases, the questioners cannot get satisfactory answers from the answerers. In order to make the questioners obtain satisfactory answers, it is very necessary for question answer communities to know how to find the authoritative users and recommend some authoritative answerers to the questioners. We consider the problem of identifying authoritative users in Stack Overflow. So far, many methods have been proposed to automatically discover authoritative users in question answer communities. A common method is to use link analysis technique such as PageRank and HITS. The main problem with this method is that it does not consider an important factor, the time factor. In this paper, we investigate the time aspect of ranking with application in the Stack Overflow. The result shows that the proposed method is highly effective.

Introduction

Question answer communities are becoming more and more popular knowledge sharing platforms, such as Stack Overflow, Yahoo! Answers, Quora and so on. One of the reasons of their success is that all the users can propose natural language questions and answer other users' questions and share their knowledge. Stack Overflow that we're going to study is a IT technology Q&A website which is related to program. Users can submit questions freely in the website, browse questions, index related content and use simple HTML when they create a home page. Every sovled question has a "accepted answer". To get a accepted answer, the questioner can choose an answer as the accepted answer. What is more, because every answer can be voted , that is, if you think the answer is useful for you, you can give a positive evaluation, on the contrary, you can give a negative evaluation. The answer which receives the highest vote can be chosen as the accepted answer.

However, such question answering mechanisms may lead to some problems. First, for a particular problem, many users give low quality answers which have very low correlation degree with the proposed questions. To a certain extent that the answers may contain advertisings and spam messages. Second, many of the questions raised by the users are similar. Before asking questions, questioners may be able to solve their doubts by browsing the similar questions and their answsers. Third, when many users ask questions in the communities at the same time, the persons who can answer them may not be able to timely see the questions. It may make these questions can not be solved very well. Based on these reasons, it is necessary to automatically identify authoritative users to provide more accurate and complete answers and make more timely responses in Stack Overflow.

So far, some existing methods to identify authoritative users have focused on some link analysis methods including PageRank[1] or HITS[2] or one of its variants AuthorRank[3]. These methods attempt to identify authoritative users by building relationships between users where nodes represent users and arcs represent the interactions between them. After that, applying these algorithms to calculate each user’s authority score and then rank users according to their authority scores. A key limitation of these methods is that they have largely ignored the time factor. As a result, authoritative users recommended by these methods usually may not be the good candidates for answering the given questions.

To address this limitation, in this paper, we propose a novel method to find appropriate authoritative users to answer a given question in Stack Overflow. Different from traditional methods, the method that we put forward considers the time factor of ranking algorithms. Considering the time factor of ranking algorithms is very important to the development of future research techniques. In the first place, the question answer community is a dynamic environment. The users who can provide authoritative answers in the past may not provide authoritative answers now or in the future. In the next place, some users may be very active in the past and often answer the questions raised by others. Howbeit, as time goes by, those users' active degree has decreased and may not answer other people's questions any more and even no longer appear in the question answer community. Ultimately, taking users’ active degree into account over time in the question answer community can recommend more authoritative answerers to questioners. The experiment result indicates that our method performs better than other methods that only use link analysis.

Related Work

To find authoritative users in the question answer communities, the most widely used method is link analysis. The most famous link analysis methods are PageRank[4] and HITS[5]. For example, Jurczyk and Agichtein adopt the HITS algorithm[6] to compute each user’s authority score by building a social network graph based on the relation between users. Schall, Daniel, and Florian Skopik propose a personalization-based PageRank[7] to find authoritative users. These works still use the original ranking algorithm and do not consider the time aspect. Also, there are many variants of the ranking algorithms,such as [8][9][10][11][12].

In the recent years, some algorithms have tried to consider time dimension of the original ranking algorithms. [13] uses the variants of PageRank to predict future PageRank scores for new pages. [14] apply time sensitive ranking algorithm(PageRank) to the publication search. It integrates the publication time of each paper into the ranking algorithm.

The Proposed Techniques

A. PageRank considering time factor

Before introducing the proposed PageRank algorithm which considers the time factor, we first describe the original PageRank algorithm. The original PageRank vector iteration formula is:

$$\vec{V} = \beta * M * \vec{V} + \vec{E} * \frac{1-\beta}{N} \qquad (1)$$

where

vector $\vec{V}$ =(V1,V2......,VN) is all the N nodes’ pagerank scores.

β is a damping factor, ranging between 0 and 1.

matrix M is a probability transfer matrix.

vector $\vec{E}$ is a unit vector. The reason for multiplying $\vec{E}$ is that the first half of the formula is a vector, so it is necessary to turn the (1-β)/N into a vector to add together.

N is the number of nodes in the directed graph.

Initially, the PageRank score for each node is set to 1/N. The calculation is done in an iterative fashion until the result finally converges. Moreover, parameter β is often set to a relatively big value (0.8 or more).

We now describe the PageRank algorithm which considers the time factor. According to the relationships among users in Stack Overflow, we construct a directed graph G = (V, E), where V is a set of vertex denoting all users and E is a set of directed edges. In the E set, for any directed edge $e \in E$, where $e=(u_i, u_j)$, $u_i \in V$ and $u_j \in V$,which indicates that user u_j answers user u_i's question. The following picture is used as an example.

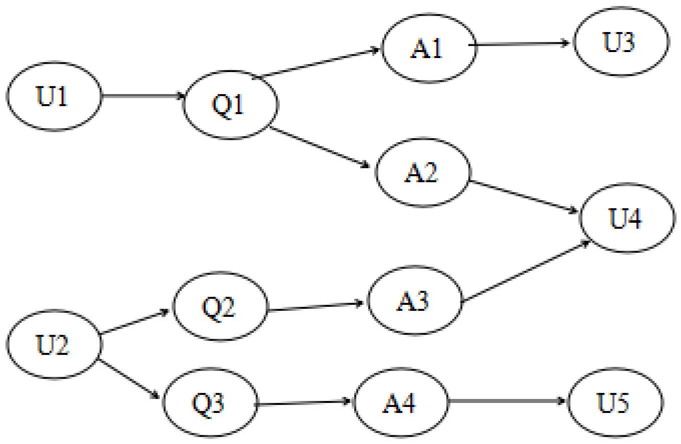

Fig.1. the relationship between users,questions and answers

As is shown in the above picture, user U2 puts forward 2 questions(Q2 and Q3). Accordingly, user U4 provides an answer(A3) to the question(Q2) and user U5 provides an answer(A4) to the question(Q3).

We can summarize the relationships between users in a multigraph shown in the following picture.

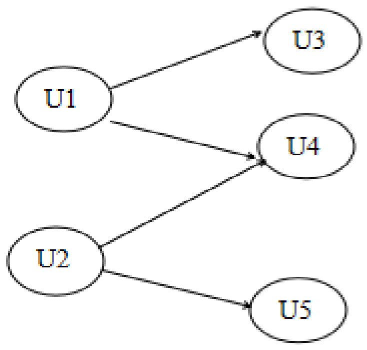

Fig.2. Summary mutigraph between users

In this way, we can derive the link structure of the question answer community.

Because we want to add the time factor to the original PageRank formula, we can transform the original vector iteration formula as folllows.

$$\vec{V} = \vec{P} .* (M * \vec{V}) + \frac{\vec{E} - \vec{P}}{N} \quad (2)$$

where

vector $\vec{V}$ =(V1,V2......,VN) is all the N users' pagerank scores.

vector $\vec{P}$ =(P1,P2......PN) is a prediction probability vector. Usually, its value varies with the time.

matrix M is a probability transfer matrix.

vector $\vec{E}$ is a unit vector.

N is the number of users in the directed graph.

Clearly, we can find that we replace constant β with vector $\vec{P}$. Generally speaking, each component value of vector $\vec{P}$ is different. We can calculate the value of the vector $\vec{P}$ in the following manner.

First, we want to introduce the time series forecasting method[15]. The definition of time series forecasting can be described as followed. According to the time series data obtained by the system, a mathematical model can be established by curve fitting and parameter estimation and analysis its trend changes over time. Moreover, it is a quantitative prediction method for the prediction of the target. There are many methods in the time series forecasting, we now use one of the commonly used methods, that is, single exponential smoothing[16][17]. Single exponential smoothing is a method that is commonly used in forecasting in which a weighted average of past values of the variable are used to forecast values of the variable. The model of single exponential smoothing method is as followed.

$$F_{t+1} = \alpha * Y_t + (1-\alpha) * F_t$$

where

F_{t+1} is the exponential smoothing trend forecast value at the t+1 stage.

α is weight coefficient, also is called exponential smoothing factor. Usually, we set its value to 0.3. The effect of α on the prediction results is further studied in section “Experimental results”.

Y_t is the actual observed value at the t stage.

F_t is the exponential smoothing trend forecast value at the t stage.

As to Stack Overflow dataset, we can count each user’s number of posts in each month in the past, including the number of asking questions and answering questions. According to the statistical data of each user in the past in the Stack Overflow, we can use single exponential smoothing which is mentioned above to forecast each user’s number of posts in the next months. According to the monthly predicted data of each user, we can calculate the value of vector $\vec{P}$ as follows. If one user asks a question in month Mi, we then collect each user’s predicted number of posts in month Mi. After that, we sum up all the users’ predicted number of posts in month Mi, denoted as SUM. As to component Pi of vector $\vec{P}$, we use user Ui’s predicted number of posts in month Mi divide SUM. In accordance with this kind of calculation method, we can finally derive the vector $\vec{P}$ in the month Mi. The larger the component Pi of vector $\vec{P}$, the more number of posts the user Ui may have in month Mi. Meanwhile, it indicates the user Ui are more active among all the users in the current month. Because each user usually has different number of posts in each month, the value of vector $\vec{P}$ always varies with the month. So, if we want to find authoritative users in each month, we should respectively calculate the value of vector $\vec{P}$ in the current month.

B. The trend factor

In the time series analysis, another important issue is the trend factor[14]. We now introduce the trend factor. It can indicate a user’s active degree changes through the future months. We assume that this is reflected by each user's number of posts change at some months before a new question is asked. We define two time periods, p1 and p2. p1 is the current time period and we define p1 as the first five months before a new question is asked. p2 is the previous time period and we define p2 as the first six months before the beginning of p1. We assume that user Ui’s sum of the number of posts in p1 is n1 and the sum of the number of posts in p2 is n2. The trend factor of user Ui is defined as $0.5^{n1/n2}$ which can guarantee its value between 0 and 1. Particularly, if n1 is 0, we set the trend factor of user Ui to 0 and if n2 is 0, we set the trend factor of user Ui to 1.

Finally, we can multiply each user’s PageRank score by its trend factor score, then we can derive each user’s final rank score.

Experimental results

In this section, we evaluate the proposed technique and compare it with the orginal PageRank algorithm. We use the dataset from Stack Overflow in 2009.

First, we preprocess the data. We use the dataset from Stack Overflow in period January 2009 to November 2009. Because there is a category for each question in Stack Overflow, we select data both quesion information and answer information under the classification of JAVA. There are totally 22936 users asking questions and answering questions under the classification of JAVA. With the proposed technique and the orginal PageRank algorithm, we can separately calculate each user's final rank score under the classification of JAVA. In the testing data, we select data both quesion information and answer information under the classification of JAVA in December 2009. In order to facilitate the calculation, we select the questions that their number of answers is no less than 3. Meanwhile, we filter the questions that have no accepted answer. To evaluate the proposed technique, we use the precision evaluation metric to test result. For example, for a particular question Q, if there are 3 users(U1,U2,U3,among of which,U1 gives the accepted answer) answering the question Q, we consider U1's rank among answerers of Q in the final rank score list. We only consider U1 whether ranks in the top three. If U1 ranks the first, the precision score for Q is 3. If U2 ranks the second, the precision score for Q is 2. If U1 ranks the third, the precision score for Q is 1. In other cases, the precision score for Q is 0. We calculate all the quesions' precision scores denoted as S under the classification of JAVA. Then we use S/N(N is the number of questions under the classification of JAVA) as the ranking algorithm's precision degree score(PDS). If a ranking algorithm has more precision degree score(PDS), it indicates the ranking algorithm is more suitable to identify authoritative users. Simultaneously, the different exponential smoothing factors in the single exponential smoothing method influencing on the result have been tested by our experiment. The following table presents the experiment results.

Table 1. Comparison results of different methods using different exponential smoothing factors

algorithm	pagerank	pagerank considering time factor								
		α=0.1	α=0.2	α=0.3	α=0.4	α=0.5	α=0.6	α=0.7	α=0.8	α=0.9
PDS	1.63667	1.64347	1.64245	1.64042	1.63991	1.63941	1.63991	1.63941	1.63941	1.63890

The result indicates that the proposed pagerank considering the time factor performs better than the orginal PageRank algorithm. As to the pagerank considering the time factor, the average PDR is 1.64036 for the the different exponential smoothing factors ranging from 0.1 to 0.9, which improves 0.22% predictive accuracy than the orginal PageRank algorithm. Moreover, the result indicates that 0.1-0.3 is the optimal range for exponential smoothing factor to get a ideal prediction result in this experiment .

Conclusion

In this paper, we present an adaptation of the PageRank algorithm and address the problem of the automatic identification of authoritative users in question answer communities such as Stack Overflow. We study the time factor of the orginal PageRank algorithm and also we consider the trend factor that is an important issue in the time series analysis when we calculate each user's final score. Experimental result indicates its superior performance to the orginal PageRank algorithm.

Acknowledgement

This work is supported by the National Key Fundamental Research and Development Program (973 program) No.2013CB329604 and the National Natural Science Foundation of China under Grant No. 61502517, 61402511

References

[1] Page, L., Brin, S., Motwani, R. and Winograd., T. The Pagerank Citation Ranking: Bringing

Order to the Web,Stanford Digital Library Technologies Project, 1998.

[2] Kleinberg, J. M. Authoritative sources in a hyperlinked environment,” Journal of the ACM, Vol. 46, No.5, pp.604 – 632, 1999.

[3] X. Liu, J. Bollen, M. L. Nelson and H. V. Sompel. Coauthorship Network in the Digital Library Research Community. Information Processing and Management, 41(6): 1462-1480, 2005.

[4] T. H. Haveliwala, Topic-sensitive PageRank. WWW, 2002

[5] J. Kleinberg. “Authoritative sources in a hyperlinked environment”, SODA, San Francisco, USA, 1998, pp. 668 – 677.

[6] Jurczyk, P., and Agichtein, E. HITS on Question Answer Portals: Exploration of Link Analysis for Author Ranking.Proc. of the SIGIR’07, Amsterdam, The Netherlands, 2007.

[7] Schall, Daniel, and Florian Skopik. "An analysis of the structure and dynamics of large-scale q/a communities." Advances in Databases and Information Systems. Springer Berlin Heidelberg, 2011.

[8] Kao W C, Liu D R, Wang S W. Expert finding in question-answering websites: a novel hybrid approach[C]//Proceedings of the 2010 ACM Symposium on Applied Computing. ACM, 2010: 867-871.

[9] Bouguessa M, Dumoulin B, Wang S. Identifying authoritative actors in question-answering forums: the case of yahoo! answers[C]//Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, 2008: 866-874.

[10] Haveliwala T H. Topic-sensitive pagerank[C]//Proceedings of the 11th international conference on World Wide Web. ACM, 2002: 517-526.

[11] Yang S, Lua E K, Wang Y. Towards human-centric personalized expertise ranking in community-based question answering[C]//Proceedings of the 2013 ACM SIGCOMM workshop on Future human-centric multimedia networking. ACM, 2013: 41-46.

[12] Weng J, Lim E P, Jiang J, et al. Twitterrank: finding topic-sensitive influential twitterers[C]//Proceedings of the third ACM international conference on Web search and data mining. ACM, 2010: 261-270.

[13] J. Cho, S. Roy and R. Adams. Page Quality: In search of an unbiased Web ranking. S

[15] De Gooijer J G, Hyndman R J. 25 years of time series forecasting[J]. International journal of forecasting, 2006, 22(3): 443-473.

[16] Billah B, King M L, Snyder R D, et al. Exponential smoothing model selection for forecasting[J]. International journal of forecasting, 2006, 22(2): 239-247.

[17] Taylor J W. Smooth transition exponential smoothing[J]. Journal of Forecasting, 2004, 23(6): 385-404.