Who Should Close the Questions: Recommending Voters for Closing Questions Based on Tags

Zhang Zhang[1,2], Xinjun Mao* [1,2], Yao Lu [1,2], and Jinyu Lu[1]

[1]National University of Defense Technology, Changsha 410073, China
[2]Key Laboratory of Software Engineering for Complex Systems, Changsha 410073, China

***Abstract*—Stack Overflow is the most popular community with a great number of questions daily post by users. The questions that are unfit for the community should be closed to maintain the quality of questions. Current practices of closing questions in Stack Overflow mainly rely on the votes of experienced users and community moderators, and face several challenges: (1)an increase in both the number of questions that should be closed and the proportion of these questions to all questions. (2) a decrease in participation willingness of non-moderator users to close questions. One way to tackle the problem is to extensively utilize the forces of experienced users in the community by recommending them appropriate questions against their development experiences and skills in order to increase their willingness and decrease their voting efforts. In this paper, we propose a voter recommendation method based on the tags of both users and questions, design user recommending algorithm based on the user willingness model that incorporates the quantitative and time information of user activity history in Stack Overflow. We select 1,397 questions randomly in Stack Overflow to validate the effectiveness of our method. The results show the successful recommendation probability in the top5, top10, top100, top300 users are 35.0%, 45.8%, 80.8%, and 89.8% respectively, which helps recommending users to 'closed' questions.**

***Index Terms*—Stack Overflow, question quality control, closing question, voter recommendation.**

I. Introduction

Stack Overflow(SO) is the most popular community driven by questioning and answering [1] [2] and with more than 10M registered users, 19M questions and 28M answers. It maintains a strong emphasis on question-answer based format, and encourages to post the questions related to programming problems, software algorithms, coding techniques and software development tools and discussions or *chit-chat* are discouraged [3]. However, a large number of questions that are not of the Stack Overflow concern are posted every day. Therefore maintaining question quality on such a large scale social collaborative platform is a great challenge [4]. The questions that are unfit for the community should be closed. Stack Overflow guidelines clearly outline categories of questions that are deemed unfit for its Q&A format, and questions that fall into the pre-defined sets of guidelines are marked 'closed' [5]. A question can be marked as 'closed' for five reasons: *duplicate, off-topic, unclear what you're asking, too broad and primarily opinion-based.*

According to the current rules of SO, the decision to 'close' a question lies completely on the shoulders of experienced users with a reputation over 3,000 and community moderators via a systematic voting process. Due to the rapid growth of Stack Overflow's questions and answers, there has been a steady increase in the workload on the experienced users and moderators [3] [6]. In contrast to thousands of questions created every day, the whole community has only 20 moderators and currently they undertake most of the voting tasks. According to the analysis on data of Stack Overflow between 2008 and 2018, the current practices of 'closed' questions faces several critical challenges: (1)the number of 'closed' questions and the ratio of 'closed' questions to total questions show an increasing trend; (2)the participation willingness of non-moderator users to close questions is declining, which has probably led to increase the workload of moderators.

One way to tackle the problem is to extensively utilize the forces of experienced users in the community, encourage them to participate in closing questions. Currently, the experienced users browse the questions in SO and vote for the questions based on their willingness, interest and correlation with their expertise. Obviously, such a method to vote by browsing thousands of questions and judging whether they need to be closed is inefficient. It is necessary to study the automatic method to recommendate users appropriate questions against their experiences and skills to increase their willingness and decrease their voting efforts. The current researches on 'closed' questions are mainly focused on predicting whether the question should be closed [3] [5] [6], and little attention is paid to the treatment of the 'closed' question, such as the study of voters.

In this paper, we propose a voter recommendation method for experienced users in Stack Overflow to improve their participation in the closing question activity. Different from the existing method, we build user willingness model against users' development experiences and skills by analyzing their tag-based activity history in SO. We design and implement the recommendation algorithm that outputs and ranks potential voting users against the tags of the question. The paper also conducts experiments to validate the effectiveness of our

DOI reference number: 10.18293/SEKE2020-073
Corresponding author: xjmao@nudt.edu.cn

method.

The rest of the paper is organized as follows. Related works are introduced in section 2. section 3 analyzes the current practices and discusses the challenging issues of closing questions in Stack Overflow. Section 4 presents and details our user recommendation method based on tags and section 5 introduces the experiments and analyzes the results. We discuss the conclusion and future work in section 6.

II. Related work

This section introduces the related works on user recommendation in Community Question Answering(CQA) and question quality control in Stack Overflow.

A. User recommendation in CQA

There have been increasing studies on user recommendation in CQA in recent years [7] [8]. Many studies model users profiles by learning their history of behaviors, and the tag-based information from the previous questions, answers and comments play an important role in this field. Yang *et al.* [9] studied the user expertise under tags and recommended a set of possible expert users for questions to help askers to get their preferable answers. And their method performs better than the up-to-date method. Wang *et al.* [10] propose a novel personalized recommendation method that considers both the topic modeling and the link structure for routing new questions to a group of experts, and the proposed method improves the recommendation performance over other methods in expert recommendation. Wang *et al.* [11] propose the Topic Professional Level Model (TPLM) to find the right experts for questions that combines both the topic model and the professional level model to calculate the user's authority under a specific topic. Their results showed that their method is superior to the traditional expert finding method in the Chinese CQA platform-Zhihu dataset. Liu *et al.* [12] propose a gating mechanism to dynamically combine structural and textual representations based on past question-answering behaviors, and their experiments on Stackexchange and Quora show that our approach can improve the performance on expert finding tasks.

B. Question quality control in Stack Overflow

The question quality control of Stack Overflow is still a great research challenge [4]. Current researches focus on prediction of question quality and suggestions for improving question quality. Correa *et al.* [3] used a machine learning framework and build a predictive model to identify a 'closed' question at the time of question creation and achieve an overall accuracy of 73%. Goyal *et al.* [5] studied the closed questions and then built a classifier that predicted whether or not a question would be closed given the question as submitted, along with the reason that the question was closed. Tóth *et al.* [13] present a novel approach for classifying questions based exclusively on their linguistic and semantic features using deep learning method and they conclude that by combining deep learning and natural language processing methods, the maintenance of quality at Q&A forums could be supported using only the raw text of posts. Calefato *et al.* [14] investigate how information seekers can increase the chance of eliciting a successful and develop a conceptual framework of factors potentially influencing the success of questions in Stack Overflow.

III. Current practices and issues analysis of closing questions

In this section, we discuss the current practices of 'closed' questions and analyze the potential issues and challenges based on data of StackOverflow[1].

A. Current practices of closing questions

Currently, the decision to 'close' a question in Stack Overflow lies completely on the systematic voting process(seeing Fig.1). The experienced users with a reputation over 3,000 and moderators undertake the voting task. The former can cast a vote to close a question once and 5 votes can close any question, and the latter can close any question with a single vote. In addition, users with a reputation of over 250 can vote to close their questions and users who hold a gold badge for one of the question's tags can close the question as *duplicate* with a single vote. As Fig.1 shows, a question can be marked as 'closed' for five reasons: *duplicate, off-topic, unclear what you're asking, too broad and primarily opinion-based.* According to our analysis on data of SO, up to December 2019, Stack Overflow has closed more than 0.8M questions, and more than 30,000 users participated in the voting tasks.

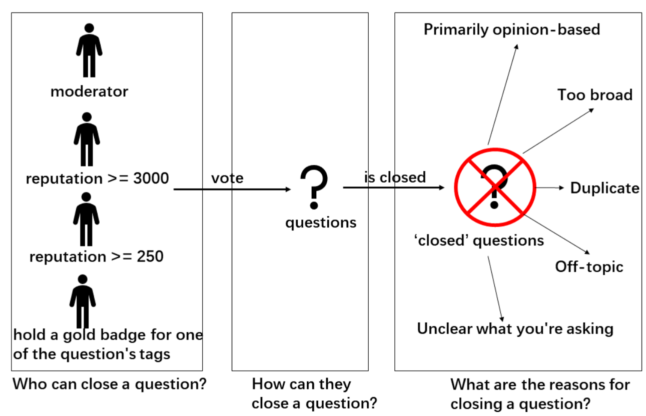

Fig. 1. **The current practices of closing questions**

B. Issues analysis of closing questions

We collect and analyze the data related to the presence of 'closed' questions in Stack Overflow between 2008 and 2018, and find the following potential issues and challenges of closing questions.

The number of 'closed' questions. Fig.2 shows the number of questions posted each year and the 'closed' questions each

[1]https://archive.org/details/stackexchange

year. First, we find a rapidly increasing trend in the number of 'closed' questions: from 205 in 2008 to 113,292 in 2018. Then, we observe the ratio of 'closed' questions to total questions over this period in Fig.2. We can find a sharp increase in the ratio of 'closed' questions after 2011 and a sharp decrease from 2014 to 2015, finally it is at around 0.056. In other words, at least one out of every 20 questions needs to be marked as 'closed' questions in 2018, which puts tremendous pressure on questions quality control of the community.

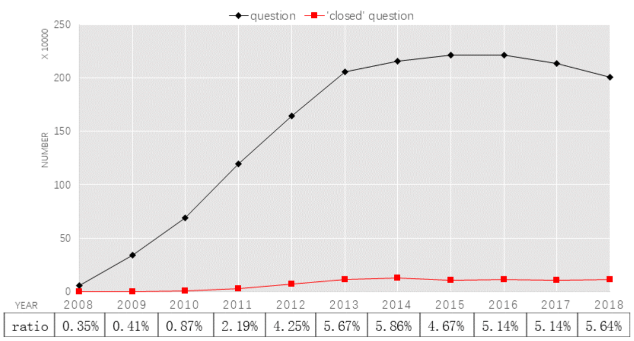

Fig. 2. The number of questions and 'closed' questions posted each year

Community Participation We analyze the voting history of experienced users and moderators to understand community participation. A question is marked as 'closed' if it reaches 5 votes but a vote from a moderator can immediately close a question. Therefore, a question can be closed with any number of 'close' votes between 1 to 5. Table I shows the distribution of the number of 'close votes' on closed questions. More than 71% of questions require moderator intervention to close. We also observe a rise in the percentage of questions being closed only by moderators over time, and a decrease in the percentage of questions being closed by experienced users.

TABLE I
THE DISTRIBUTION OF NUMBER OF 'CLOSE VOTES'

Year	1-vote	2-votes	3-votes	4-votes	5-votes
2008	100.00%	0	0	0	0
2009	8.50%	2.05%	1.49%	7.08%	80.88%
2010	4.87%	2.72%	1.94%	1.69%	88.78%
2011	22.15%	8.18%	5.71%	4.44%	59.53%
2012	23.06%	5.56%	2.56%	1.84%	66.98%
2013	8.90%	2.97%	1.79%	1.14%	85.19%
2014	18.09%	5.19%	2.53%	1.53%	72.66%
2015	36.02%	11.66%	4.38%	1.97%	45.97%
2016	41.49%	14.15%	4.93%	2.12%	37.31%
2017	45.92%	14.62%	4.95%	2.04%	32.47%
2018	48.94%	15.42%	4.96%	2.06%	28.62%

Then we further study the difference of the number of votes between the moderators and non-moderators. Table II shows descriptive statistics on the voter distribution from 2008 to 2018. We can find that the number of moderators is much less than that of non-moderators, but they have undertaken most of the voting tasks. The maximum vote number is 15,024, showing that one moderator in SO has made his own contribution to close at least 15,024 questions. The average vote number of moderators is 2,601, which is about 30 times the average number of non-moderators votes. However, the number of non-moderators is about 550 times the number of moderators. In conclusion, the status of community participation in voting to close the questions has led to a huge workload on moderators according to our analysis.

TABLE II
THE STATISTICS OF MODERATORS AND NON-MODERATORS

Types	Number	Mean of voting number	Median of voting number	Min of voting number	Max of voting number
Moderators	56	2601	2076	4	15024
Non-moderators	30873	86	6	1	20695

Summary. We now summarize the potential issues and challenges of closing question in SO:

- From the perspective of 'closed' questions, there is an increasing number of 'closed' questions and the percentage of 'closed' questions, which requires more votes to participate in closing the questions.
- From the perspective of users who vote to close questions, there is a decrease in community participation of non-moderator users to close the questions, which has probably led to an increase in the workload for moderators, and thereby requires an effective method to encourage non-moderator users to participate in voting.

According to the above analysis, one way to tackle the issues is to seek an effective way to help and encourage experienced users to participate in the voting. The feasible solution is to actively recommendate users appropriate questions for voting against their development experiences and skills. Such a method can increase their willingness and decrease their voting efforts, and therefore improve the efficiency of closing questions and enhance community question quality control.

IV. User Recommendation Method Based on Tags

This section details our method, including user willingness model incorporating the quantitative information and time information of tag-based information, and the user recommendation algorithm.

A. User willingness model based on tags

One potential voting candidate that is willing to participate in voting questions depends on several factors that are related to the questions and the user itself. These factors include his/her expertise, activeness, etc. We can build the user willingness model based on these factors to help user recommendation. User models can be established by learning their history of behaviors [15], we can analyze the tag information in the user history data to establish an accurate user willingness model.

Quantitative information of tag-based activity history. In a user's quantitative information of tag-based activity history, we extract the frequency of activity about a tag(the number of questioning, answering and commenting related to a specific tag). And it represents the user's willingness on this tag [16]. The more frequently a user participates in a post related to a particular tag, the more interested he is in that tag. We use the $Fre_{tag_i(u)}$ to measure the frequency of tag *i* for user *u*, which is defined as:

$$\text{Fre}_{tag_i(u)} = number\ of\ activities\ in\ tag\ i\ for\ user\ u. \tag{1}$$

For example, if user *u* has commented 2 posts related to *java*, posted 1 question related to *java*, and answered 5 questions related to *java*, then $Fre_{tag_{java}(u)}$ = 5+2+1=8.

Time information of tag-based activity history. The user's willingness is dynamically changing [17], thus, the time information of tags is valuable. We extract the recency of user activity about a tag from the time information, that is, the chronological order of activities related to the tag. The activity data close to the current temporal period is usually more important than that temporally far from the current period [17]. This study defines the recency to which user *u* participate in tags *i*(abbreviated as $\text{RecT}_{tag_i(u)}$) as the following:

$$\text{RecT}_{\text{tag}_i(u)} = 1 - \frac{Current - Last_{tag_i(u)}}{Current - First_{tag_i(u)}} \tag{2}$$

where *Current* is the time point at which the user tag recency is currently measured. $Last_{tag_i(u)}$ is the last time user *u* participated in the posts or comments related to tag *i*. And $First_{tag_i(u)}$ is the first time user *u* participated in the posts or comments related to tag *i*.

We also extract the duration of user activity from the time information, that is, the length of time a user has participated in activities related to a specific tag. It is another important factor based on the time information to represent a user's willingness [18]. The long-duration activity history about a tag usually reflects a user's willingness more than the short-duration ones. We use the $Duration_{tag_i(u)}$ to measure the duration of user *u*'s participation in posts or comments related to tag *i*, which is defined as follow:

$$Duration_{tag_i(u)} = Last_{tag_i(u)} - First_{tag_i(u)} \tag{3}$$

Where $Last_{tag_i(u)}$ and $First_{tag_i(u)}$ have been mentioned above.

Then we use the $ActDuration_u$ to measure the duration of user *u*'s participation in posts or comments, which is defined as follow:

$$ActDuration_u = LastTime_u - FirstTime_u \tag{4}$$

Where $LastTime_u$ is the last time user *u* participated in the posts or comments and $FirstTime_u$ is the first time user *u* participated in the posts or comments.

After getting the above two indicators, we use their ratio to measure the duration of the user *u*'s preference for tag *i*(abbreviated as $\text{DurT}_{tag_i(u)}$) as the following:

$$DurT\text{tag}_i\text{(u)} = \frac{Duration_{tag_i(u)}}{ActDuration_u} \tag{5}$$

User willingness model. The user willingness model is composed of several model elements for each tag. To construct user *u*'s model element for tag *i*, this paper uses $\text{Pre}_{tag_i(u)}$ to combine the frequency, recency and duration of user *u*'s activity history about tag *i*, which is defined as follow:

$$Pre_{tag_i(u)} = Fre_{tag_i(u)} * (\alpha * \text{RecT}_{tag_i(u)} + \beta * DurT_{tag_i(u)}) \tag{6}$$

Where α and β are used to control the relative impact of $\text{RecT}_{tag_i(u)}$ and $ActDuration_u$, and α + β =1(0≤α,β≤1).

Fig.3 shows user *u*'s activity history, and user *u*'s willingness model elements for tag *java* are based on it. Firstly, we extract quantitative information and time information of activities related to *java* as follows:

- User *u* posted a question related to *java* on *2019-02-14*, answered a post related to *java* on *2019-03-15*, commented a post related to *java* on *2019-04-01*, so $Fre_{tag_{java}(u)}$ = 1 +1 +1 = 3 according to Eq.1.
- $RecT_{tag_{java}(u)} = 1 - \frac{2019\text{-}05\text{-}01 - 2019\text{-}04\text{-}01}{2019\text{-}05\text{-}01 - 2019\text{-}02\text{-}14} \approx 0.605$ with $First_{tag_{java}(u)}$ = *2019-02-14*, $Last_{tag_{java}(u)}$ = *2019-04-01*, $Current$ = *2019-05-01* according to Eq.2.
- $DurT_{tag_{java}(u)} = \frac{2019\text{-}04\text{-}01 - 2019\text{-}02\text{-}14}{2019\text{-}04\text{-}01 - 2019\text{-}02\text{-}01} \approx 0.779$ with $FirstTime_u$ = *2019-02-01*, $LastTime_u$ = *2019-04-01*, $First_{tag_{java}(u)}$ = *2019-02-14*, and $Last_{tag_{java}(u)}$ = *2019-04-01* according to Eq.3, Eq.4 and Eq.5.

Then, we get $Pre_{tag_{java}(u)}$ = $3*(0.5*0.605+0.5*0.779) \approx 2.08$ with $\alpha = 0.5$ and $\beta = 0.5$ according to Eq.6.

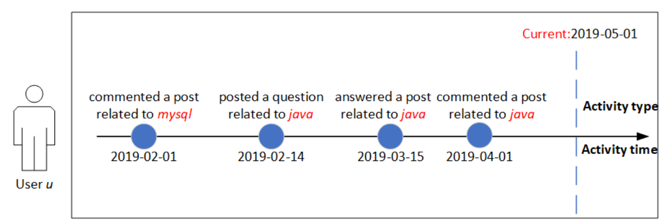

Fig. 3. An illustration of the user's activity history

B. User recommend algorithm

This section proposes a user recommendation algorithm based on the above user willingness model. The algorithm inputs the question with tags and the candidates with their activity history and outputs ranked candidates based on the user willingness model on the tags of the question to be voted, and the top *k* users will be recommended to vote for the question.

Firstly, we input the question that needs votes and get the tags of this question. We also need to get the candidates and their activity history. Then, we get the $PerScore_u$(the sum of the candidate u's user willingness model for tags of this question) for each user, and rank them based on the $PerScore_u$. Finally, we select top k users to vote for this question. The details of the algorithm are as follows:

Algorithm: User Recommend

Input:

tag_list: the list of tags of the question that needs voters.
user_list: users who belong to the candidate list.
history_list: activity history of users who belong to user_list.

Output:

RecommendedUser-list: ranked users

Procedure:

```
let PerScore be a list
for u in user_list:
    let PerScore_u = 0
    for i in tag_list:
        calculate the Pre_{tag_i(u)} [1]
        PerScore_u += Pre_{tag_i(u)}
    PerScore.append(PerScore_u)
sort PerScore based on the PerScore_u
return PerScore
```

[1]The user u's willingness model for tag i

V. Experiments and Results

In this section, we will describe the design of our experiment, and analyze the results of the experiment.

A. Experiment design

In order to validate the effectiveness of our recommendation method, firstly, we construct the candidates' user willingness model based on their activity history in SO, then select the 'closed 'questions in SO and use our recommendation method to recommend experienced users to vote these questions. We then analyze the overlap between the users who actually vote to close these questions and our recommended users.

Data selection. We collect user activity history from January 2018 to September 2019 in SO, and select randomly 1,397 questions that were closed between January 2019 and September 2019 by the experienced users and use them to perform the experiment.

Evaluation metrics. We evaluate the overlap between the actual voters and the recommendation result by *r_top@k*. We define *r_top@k* as follows: if any actual voter of a 'closed' question ranks among the top *k* in our recommendation results, this question is called `hit question`. And *r_top@k* = (the number of `hit question`)/(the number of all tested questions). Our candidate list is consist of 30,000+ experienced users who have participated in closing the questions, so we choose the 1% of the candidates as the maximum value of *k*: 300. On the other hand, any question can be closed if it reaches 5 votes, so the minimum value of *k* is set to 5. Thus, *k* of *r_top@k* is varied as 5, 10, 100, and 300.

Parameter settings. We use 5 different sets of α and β in our experiment: (1)α = 0, β = 1; (2)α = 0.3, β = 0.7; (3)α = 0.5, β = 0.5; (4)α = 0.7, β = 0.3; (5)α = 1, β = 0. For the construction of the user willingness model, we have taken the activity history of the questions, answers and comments of the users in the 30 days and 365 days before the tested question is posted respectively.

B. Results and analysis

The experiment results are shown in Table III. From the time period of the data used to build the user willingness model, the effectiveness of the user recommendation method based on the 30-day data is better than that using the 365-day data when the values of α and β are the same. We speculate that the latter contains more user activity history, but the old data may be misleading to reflect the user's willingness with the user's willingness changing over time. From the values of α and β, with the increase of α and the decrease of β, the effectiveness of the user recommendation method is gradually improved, and reaches the best when α = 1 and β = 0. This may be because the recency of the activity is more expressive of the user's willingness than the duration of the activity in the time information or the indicator $DurT_{tag_{java}(u)}$ we use to measure the duration of the user's willingness is not accurate.

TABLE III
THE EFFECTIVENESS OF USER RECOMMENDATION METHOD WITH DIFFERENT PARAMETER SETTINGS

Time period	α	β	r_top@5	r_top@10	r_top@100	r_top@300
30 days	0	1	0.346	0.450	0.797	0.88
30 days	0.3	0.7	0.348	0.452	0.803	0.89
30 days	0.5	0.5	0.349	0.455	0.803	0.89
30 days	0.7	0.3	0.349	0.455	0.805	0.894
30 days	1	0	0.350	0.458	0.808	0.898
365 days	0	1	0.334	0.422	0.772	0.867
365 days	0.3	0.7	0.335	0.424	0.777	0.879
365 days	0.5	0.5	0.335	0.424	0.779	0.88
365 days	0.7	0.3	0.336	0.424	0.779	0.882
365 days	1	0	0.337	0.424	0.778	0.881

Overall, the user recommendation method using the 30-day data to build the user willingness model is the most effective with α = 1 and β = 0 in our experiment: r_top@5 = 0.350, r_top@10 = 0.458, r_top@100 = 0.808, r_top@300 = 0.898. Then we use this set of parameters to analyze the effectiveness of our method for different kinds of 'closed' questions(seeing Fig.4). For r_top@5 and r_top@10, our method is most effective for the questions closed as *Unclear what you're asking*: r_top@5 = 0.395, r_top@10 = 0.499. However, our method is not ideal for the questions closed as *Too broad* and *Primarily opinion-based*. We speculate that the user's willingness to participate in the closing questions is also affected by the ease of identifying the reasons of the closing questions.

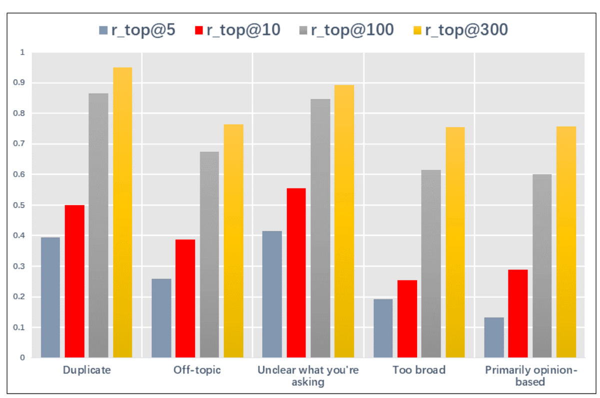

Fig. 4. The effectiveness of our method for different kinds of 'closed' questions

We pick 10 most frequently occurring tags in the test set and verify the effectiveness of our method for questions related to these tags(seeing Table IV) with above parameter settings. Because our test set is randomly selected, the situation of the tag frequency distribution in the test set is similar to that in the community. There are significant differences in the effectiveness of our method for questions related to different tags: `c#` with r_top@5 = 0.102, r_top@10 = 0.284, `python-3.x` with r_top@5 = 0.519, r_top@10 = 0.597, and so on. This indicates that the effectiveness of our method is influenced by the tags of 'closed' questions, and this may be because the popularity of a tag affects users' willingness to participate in voting activities related to the tag.

TABLE IV
THE EFFECTIVENESS OF OUR METHOD FOR 'CLOSED' QUESTIONS RELATED TO DIFFERENT TAGS

Tag	r_top@5	r_top@10	r_top@100	r_top@300	Number
python	0.431	0.546	0.838	0.927	260
javascript	0.289	0.371	0.811	0.962	159
java	0.194	0.3125	0.792	0.917	144
php	0.402	0.413	0.772	0.913	92
c#	0.102	0.284	0.83	0.92	88
python-3.x	0.519	0.597	0.909	0.961	77
html	0.365	0.486	0.824	0.905	74
c++	0.233	0.466	0.795	0.932	73
android	0.125	0.266	0.609	0.781	64
r	0.365	0.476	0.984	0.984	63

VI. CONCLUSION AND FUTURE WORK

To close the unfit questions in CQA is extremely significant in order to manage and guarantee the quality of the question and the whole community. Current practices of closing questions face several challenges, which requires encouraging experienced users to participate in closing questions and increases community efficiency. One way to solve these challenges is to actively recommend experienced users appropriate questions against their development experiences and skills, instead of relying on them to randomly browse the questions to determine whether they need to vote in the past. In this paper, we present an effective method to actively recommend users for questions in CQA. Our contribution of this paper is threefold:(1) obtaining some important findings about the potential issues and challenges of voting. (2)building a user willingness model based on the relationship of tags of both users and questions by extracting the quantitative and time information of user activity history. (3)proposing a user recommendation algorithm that outputs and ranks the potential voters for questions. We conduct experiment to validate the effectiveness of our proposed method. The experiment results are positive and impressive in successful recommendation.

In the future, we plan to use more dimensional indicators of tag-based information to model users, such as active time. In addition, the users' collaboration in the past vote history will also be included in the recommendation basis.

ACKNOWLEDGMENT

This work was supported in part by the National Key Research and Development Program of China under Grant 2018YFB1004202, and in part by the Laboratory of Software Engineering for Complex Systems.

REFERENCES

[1] Zhu J, Shen B, Cai X, et al. Building a Large-scale Software Programming Taxonomy from Stackoverflow[C]//SEKE. 2015: 391-396.
[2] Wang T, Yin G, Wang H, et al. Automatic knowledge sharing across communities: A case study on android issue tracker and stack overflow[C]//2015 IEEE Symposium on Service-Oriented System Engineering. IEEE, 2015: 107-116.
[3] Correa D, Sureka A. Fit or unfit: analysis and prediction of'closed questions' on stack overflow[C]//Proceedings of the first ACM conference on Online social networks. 2013: 201-212.
[4] Ponzanelli L, Mocci A, Bacchelli A, et al. Improving low quality stack overflow post detection[C]//2014 IEEE international conference on software maintenance and evolution. IEEE, 2014: 541-544.
[5] Mukerjee A. Predicting Closed Questions on Stack Overflow[J]. 2013.
[6] Lezina C G E, Kuznetsov A M. Predict closed questions on stackoverflow[J]. 2013.
[7] Huang W, Mo W, Shen B, et al. CPDScorer: Modeling and Evaluating Developer Programming Ability across Software Communities[C]//SEKE. 2016: 87-92.
[8] Zhao Z, Yang Q, Cai D, et al. Expert Finding for Community-Based Question Answering via Ranking Metric Network Learning[C]//Ijcai. 2016, 16: 3000-3006.
[9] Yang B, Manandhar S. Tag-based expert recommendation in community question answering[C]//2014 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM 2014). IEEE, 2014: 960-963.
[10] Wang L, Wu B, Yang J, et al. Personalized recommendation for new questions in community question answering[C]//2016 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM). IEEE, 2016: 901-908.
[11] Wang S, Jiang D, Su L, et al. Expert Finding in CQA Based on Topic Professional Level Model[C]//International Conference on Data Mining and Big Data. Springer, Cham, 2018: 459-465.
[12] Liu Z, Zhang Y. Structures or texts? a dynamic gating method for expert finding in cqa services[C]//International Conference on Database Systems for Advanced Applications. Springer, Cham, 2018: 201-208.
[13] Tóth L, Nagy B, Janthó D, et al. Towards an Accurate Prediction of the Question Quality on Stack Overflow Using a Deep-Learning-Based NLP Approach[C]//14th International Conference on Software Technologies, ICSOFT 2019. SciTePress, 2019: 631-639.
[14] Calefato F, Lanubile F, Novielli N. How to ask for technical help? Evidence-based guidelines for writing questions on Stack Overflow[J]. Information and Software Technology, 2018, 94: 186-207.
[15] Yu Y, Wang H, Yin G, et al. Reviewer recommender of pull-requests in github[C]//2014 IEEE International Conference on Software Maintenance and Evolution. IEEE, 2014: 609-612.
[16] Golder S A, Huberman B A. Usage patterns of collaborative tagging systems[J]. Journal of information science, 2006, 32(2): 198-208.
[17] Zheng N, Li Q. A recommender system based on tag and time information for social tagging systems[J]. Expert Systems with Applications, 2011, 38(4): 4575-4587.
[18] Huang C L, Yeh P H, Lin C W, et al. Utilizing user tag-based interests in recommender systems for social resource sharing websites[J]. Knowledge-Based Systems, 2014, 56: 86-96.