Knowledge-Based Systems 182 (2019) 104794

Contents lists available at ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys

Multi-feature based Question–Answerer Model Matching for predicting response time in CQA ☆

YueLiu [a,b,*], Aihua Tang [a], FeiCai [a], Pengfei Ren [a], Zhibin Sun [c]

[a] School of Computer Engineering and Science, Shanghai University, Shanghai 200444, China
[b] Shanghai Institute for Advanced Communication and Data Science, Shanghai University, Shanghai 200444, China
[c] Natural Resource Ecology Laboratory, Colorado State University, Fort Collins, CO 80523, USA

ARTICLE INFO

Article history:
Received 6 October 2018
Received in revised form 18 March 2019
Accepted 3 June 2019
Available online 12 June 2019

Keywords:
Community Question Answering
Topic-sensitive model
Stack overflow
The prediction of question response time

ABSTRACT

Users of Community Question Answering (CQA) could not manage their time conveniently because their questions are often not answered quickly enough. To address this problem, we try to provide a function for CQA sites to inform users when their questions will be answered. In this paper, we propose a Question–Answerer Model Matching based answerer's response time prediction named (QAM^2), which consists of two parts: the construction of the Multi-feature based Question–Answerer Model (MQAM, including the answerer model and the question model) and the prediction of question response time based on MQAM Matching Strategy (QAMMS). Firstly, the MQAM is built according to some extracted deep features (e.g., answerer's interest, professional level, activity, question category and difficulty), which are neglected in most existing methods on the prediction of question response time. Herein, the Label Cluster Latent Dirichlet Allocation (LC-LDA) model was proposed to overcome the compulsive allocation behaviors caused by traditional topic models (e.g. LDA), which treats the words that are irrelevant or weakly related to the subject as the topic of short texts when extracting the feature of answerer's interest and question category. Meanwhile, an improved PageRank algorithm-topic sensitive weighted PageRank (TSWPR) is used to eliminate the impact of "indiscriminate" users who have answered many questions with low quality of answers. Secondly, we use the model matching strategy based on multiple classifier for matching MQAM and calculating the question response time of each answerer. Experiments conducted on two real data sets of Stack Overflow show that the proposed method can improve significantly the accuracy of question response time prediction in CQA.

1. Introduction

Community Question Answering (CQA), such as Yahoo! Answers,[1] Stack Exchange[2] and Baidu Knows,[3] is a website for users to pose and answer questions. Although there are many services for users to handle their confusion conveniently (e.g., question recommendation [1], question retrieval [2], expert identification [3]), most CQA sites are not able to guarantee that users can receive satisfying answers to their questions on time, resulting in users' being disappointed and frustrated. Based on the research of Bhat et al. [4], we analyzed the response time of the questions which have been answered in Stack Overflow (one of the most popular CQA sites on computer programming), finding that about 37.7% of the question response time is over one hour, and even 11.81% of the question response time is longer than one day, as shown in Fig. 1. It indicates that the question response time is with a larger range of fluctuation.

The above issues make it difficult for questioners to decide whether to switch focus to other parts of software development or to keep waiting for answers. This problem has brought great inconvenience to questioners' time management. The mechanism of providing users with an accurate time of answering their questions in CQA can not only help them manage their time reasonably, but also prompt them to rephrase their questions for obtaining answers faster.

Some CQA sites have been aware of the importance of the prediction of response time and launched related services. For example, Baidu Knows ever released the predictive service of

☆ No author associated with this paper has disclosed any potential or pertinent conflicts which may be perceived to have impending conflict with this work. For full disclosure statements refer to https://doi.org/10.1016/j.knosys.2019.06.002.

* Corresponding author at: School of Computer Engineering and Science, Shanghai University, Shanghai 200444, China.
E-mail addresses: yueliu@shu.edu.cn (YueLiu), ah1995@shu.edu.cn (A. Tang), cflovelei@shu.edu.cn (FeiCai), rpf5002@shu.edu.cn (P. Ren), zhibin.sun@colostate.edu (Z. Sun).

1 https://answers.yahoo.com
2 http://stackexchange.com
3 https://zhidao.baidu.com

https://doi.org/10.1016/j.knosys.2019.06.002
0950-7051/

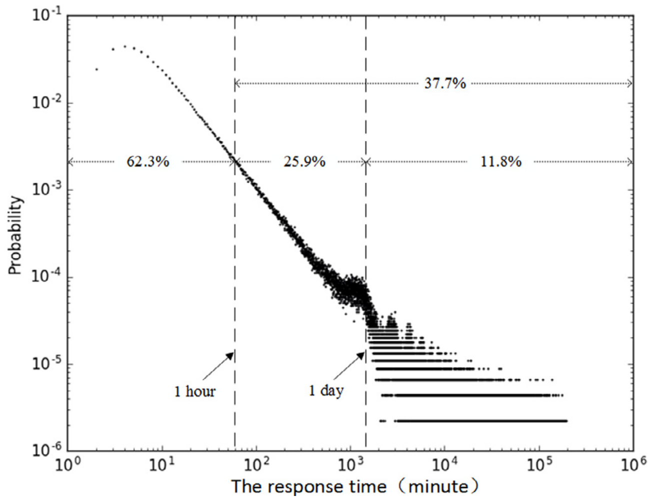

Fig. 1. The distribution of answering response time in Stack Overflow.

the question response time through the countdown mode, in order to offer a psychological expectation of the time length to a questioner when he/she is waiting for the answer. This service predicts the time only based on the type of question and does not consider other effects which would also affect the question response time in CQA. Due to its low accuracy, this service had been shelved.

During last few years, many attentions were paid to the prediction of question response time. Then a large body of work has been produced in CQA. Asaduzzaman et al. [5] laid the foundation for the prediction of question response time, inspiring many researchers to do further research. They investigated unanswered questions of Stack Overflow, finding that some features related to questions (such as question's tag similarity, title length, post length, etc.) have a high correlation to the response time of questions. Then they found that the label of question has a significant impact on the response time and proposed a prediction method of the question response time based on the question's label [6] Goderie et al. [7] extended the question label to multiple label-related features (such as the activity of label, the responsibility of label, the popularity of label, etc.), and then employed K-Nearest Neighbor method to predict the question response time. Bhat et al. [4] fused the label-related features and the statistical characteristics of question's content (such as question's title length, post length, code length, number of code snippets, etc.), and then used the logic regression, support vector machine, decision tree and other machine learning methods to construct a model to predict whether the question can be answered before a certain time. Its experimental results showed that this method obtains a slight improvement (by only another 1%–2%) when comparing with other prediction methods, which only consider the label-related features.

All the above methods can predict the question response time in CQA by building models to simulate the relationship between the response time and the characteristics of the questions. In fact, the question response time not only depends on the characteristics of the questions but also is affected by the answerer's behavior in CQA such as total questions asked and total answers posted [8]. Therefore, some researchers turned their attention to combine the relevant features of answerers and questions to predict question response time. Burlutski et al. [9] analyzed the features influencing the question response time in view of questions and answerers, and established the answerer model based on the answerer's characteristics (such as the number of questions the answerers responded, the average response time of answerers, the reputation of answerers, the location of answerers, etc.) and the question's model based on the question's characteristics (such as the length of the title, the length of text, number of labels, tag popularity, etc.). Then they formulated the prediction as a binary classification task to predict whether a question would be answered before a given response time threshold. The limitation of the binary classification method is that the time range of the question response cannot be accurately given. Furthermore, in CQA sites, the question response time would be longer with the difficulty of the questions and the lack of answerer's professionalism [10]. Berger et al. [11] found the quality of CQA's reward system would affect the answerer's willingness to answer questions. Another important factor affecting the question response time is the user's activity in CQA [12] because the interaction among users of CQA is essential during its development. This method did not take these factors (such as the difficulty of the question, the user's level of expertise, activity, etc.) into account, leading to low accuracies.

To improve the accuracy of predictions, it is necessary to take more features into account. However, these features are not provided in CQA sites, and they can only be approximately estimated through the user's history records in CQA. In the process of calculating these factors, one primary challenge is how to evaluate the professional level of answerers. Zhou et al. [13] proposed a topic-sensitive probabilistic model to solve this problem. The main shortcoming of this method is that the authors used Latent Dirichlet Allocation (LDA) [14] to extract the topics of answer and question's text, which leads to the topic compulsory distribution because most of the questions in CQA are short texts. Lately, the Label Latent Dirichlet Allocation (L-LDA) [15] defines a one-to-one correspondence between the LDA's latent topics and the existing labels. The mechanism of L-LDA compensates for the deficiency of LDA on topic distillation at a certain extent. However, the labels of CQA show a variety of multi-level features, which makes it difficult for users to specify the label at the level of the subordinate label. In such cases, the label level could be specified to be too small or too large, resulting in over-fitting or under-fitting of the L-LDA classification results. Besides, they ignored the influence of the answers' quality. It results that the professional level of some "indiscriminate" users, who answered many questions with low quality of answers, are equal to or even higher than the real expert, reducing the accuracy of the prediction.

To make up the deficiency of these above methods, in this paper, we propose Question–Answerer Model Matching based Answerer's Response Time Prediction method named QAM^2 in this paper. It firstly constructs a Multi-feature based Question–Answerer Model (MQAM, consisting of the answerer model and the question model) according to the characteristics of answerers (e.g., the answerer's interest, professional level, activity degree) and questions (e.g., the question category, difficulty). Then Question–Answerer Model Matching Strategy (QAMMS) based on multi-classifier is used to match MQAM, thus calculating the question response time of each answerer. Specifically, the main contributions of this paper are the followings:

(1) To overcome over-fitting or over-generalization of the topic extraction caused by L-LDA, the Label Cluster Latent Dirichlet Allocation (LC-LDA) model is proposed to distill the categories of questions and uses' preference. It uses the Markov Cluster Algorithm to cluster low frequency tags of CQA and then these newly generated tag clusters are used as new tags for L-LDA topic discovery. (Section 2.1.1)
(2) To eliminate the impact of "indiscriminate" users on measuring the quality of answers, the topic sensitive weighted PageRank (TSWPR) algorithm is proposed to calculate the

difficulty of questions and the professional level of askers in different areas, in which the voting data is used as a factor to improve the calculation of the probability that user will answer questions. (Section 2.1.2)
(3) To effectively evaluate the user's activity, a calculation method based on the Local Weighted Linear Regression (LWLR) is proposed based on ideas that the activity of user is affected by the activity of the adjacent time. MQAM is constructed on these above-mentioned extracted features. (Section 2.1.3)
(4) To solve the problem that the asker cannot obtain accurate response time, from the mechanism of binary classification, we transform the prediction of question response time as a multi-class problem by the multi-classifier (e.g. SoftMax algorithm), which is used to match the answerer model and the question model (QAMMS). The question response time for each user can be obtained. (Section 2.2)
(5) Finally, we use Stack Overflow as our research object and carry out experiments on the Stack Overflow data set. The results show that our proposed method and these feature extraction methods can significantly improve the accuracy. (Section 3)

The remainder of this paper is as follows. In Section 2 describes our proposed method. Section 3 analyzes the experiment and its result. Finally, conclusions and future research directions are presented in Section 4.

2. The mechanism of the proposed prediction method

Our proposed Question–Answerer Model Matching based Answerer's Response Time Prediction method (QAM^2) can be divided into following two modules:

(1) Multi-Feature based Question–Answerer Model (MQAM). It includes the answerer model and the question model. The answerer model describes answerers from the aspects, including answerer interest, answerer professional level, answerer activity and the statistical features of answers. Similarly, we construct the question model based on question category, question difficulty, question time and the statistical features of questions.
(2) Question–Answerer Model Matching Strategy (QAMMS). It aims to match the answerer model and the question model based on the multi-classifier model matching strategy, and then to calculate the response time. The structure of the proposed prediction method is depicted in Fig. 2. More details are presented in the following subsections.

2.1. The mechanism of the proposed prediction method

In the process of constructing MQAM, we firstly extract the relevant characteristics of answerer and question. Then we give a formal representation of the answerer model and the question model according to extracted features. The process of feature extraction can be divided into four parts according to different methods. These four methods are introduced in detail in the following.

2.1.1. Feature extraction of answerer interest and question category based on LC-LDA

Question classification is essentially important to correctly find answerers who are interested in the question. However, the artificial classification of the question always results in classification error in CQA. L-LDA is often used for extracting the interested distribution of users in CQA. It is a supervised topic model, defining a one-to-one correspondence between the LDA's latent topics and the existing labels [15]. It is suitable for the topic extraction of the CQA documents because each record in CQA has its own labels. Thus, L-LDA can map the label of the text to the subject level and find the topic distribution of the text in CQA. However, this method does not acquire the answerer interest distribution accurately because of many low frequency tags in CQA. To overcome this difficulty, we use the Markov Cluster Algorithm (MCL) to cluster labels and then proposes an LC-LDA model (based on L-LDA), which is used to find the feature of answerer interest and question category. Next, we describe the LC-LDA model in detail.

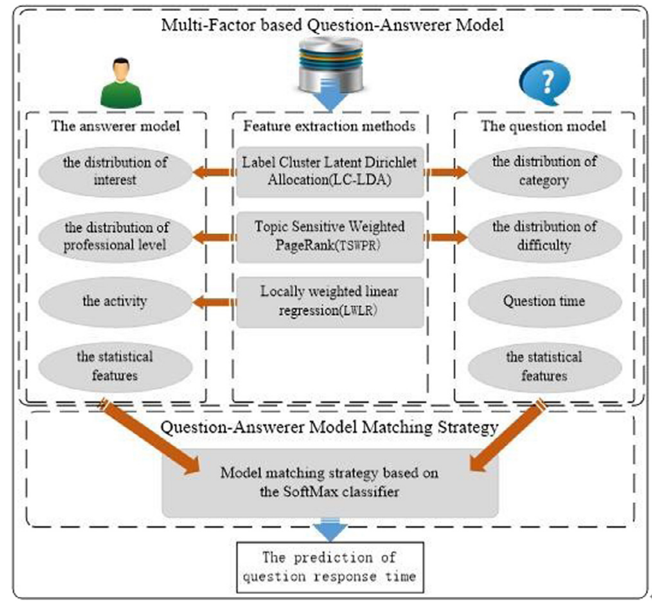

Fig. 2. Question–Answerer Model based on Multi-Feature.

MCL, proposed by Van Dongen et al. in 2000, is a graph clustering algorithm based on random walks [16]. According to the basic idea of MCL algorithm, this study uses the coherence between labels to cluster the tags, and the tags with high co-occurrence are clustered into label clusters. Therefore, it is necessary to establish a label co-occurrence matrix and a label co-occurrence probability matrix as defined below.

Definition 1. Label co-occurrence Matrix: Given a collection of tags $T^{(n)}$ in CQA, $U_{n\times n}$ is a label co-occurrence matrix which represents the number of simultaneous occurrences of different labels in the question of CQA, where each element u_{ij} represents the times that the tag t_i and t_j are commonly labeled with the same question, and n represents the number of tags in CQA:

$$u_{ij} = \begin{cases} |S_i \cap S_j|, & i \neq j \\ 1, & i = j \end{cases} \tag{1}$$

where S_i represents the question set labeled by t_i.

Definition 2. Label co-occurrence Probability Matrix: Given a collection of tags $T^{(n)}$ in CQA, $C_{n\times n}$ is a label co-occurrence probability matrix. Each element c_{ij} of the matrix represents the probability that the label t_i and t_j label the same question in CQA:

$$c_{ij} = \frac{u_{ij}}{\sum_{k=1}^{n} u_{kj}} \tag{2}$$

The matrix can be considered as a probability transition matrix in random walk. Next, in the tag clustering algorithm based on

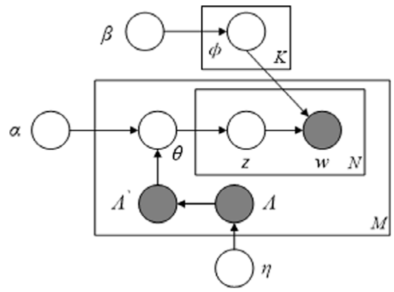

Fig. 3. Graphical model of LC-LDA.

MCL, "expand" and "inflation" are performed alternately until the matrix convergence. The definitions of these two operations are described as follows:

Expand refers to the product of label co-occurrence probability matrix. In the view of the random walk, it is the process of the Markov chain iteration [17], which means the probability of the walker in the various nodes at the next moment:

$$C_{exp} = Expand(C) = C \cdot C \tag{3}$$

Inflation refers to stopping the process of the Markov chain iteration. The specific method is first to expand each of the elements in the label co-occurrence probability matrix by the power of r, and then normalize each column:

$$C_{ij}^{(inf)} = \frac{c_{ij}^r}{\sum_{k=1}^{n} c_{kj}^r} \tag{4}$$

where $r > 1$ represents the expansion coefficient. r can strengthen the connection of nodes within the cluster, reducing the connection of inter-cluster. This is followed by the pruning operation, which makes the matrix converge faster by setting the value of the label co-occurrence probability matrix below the threshold f to zero.

After C converges, there is no transition path between the labels within different clusters, and the clustering results can be explained in the following way:

In C, the nonzero value c_{ij} in the column means that the label v_i is the cluster center and the labels v_i and v_j belong to the same class. It maps original labels of the question to the tag cluster, which can be used as the new tag for L-LDA topic discovery.

Fig. 3 illustrates the generative process with a graphical representation of LC-LDA. In this figure, the hollow circles represent observed variables, and the dark circles represent latent variables. An arrow indicates a conditional dependency between variables. Plates (i.e., boxes in the figure) indicate repeated sampling with the number of repetitions given by the variable M, K, N at the bottom. α and β are the Dirichlet hyperparameters of topic and word respectively, η is the Bernoulli hyperparameters of labels, ϕ and θ are the multinomial distributions of word and topic. z represents the topic, and w represents observed word variables. $\Lambda^{(d)} = (l_1, \cdots, l_K)$ is a list of binary topic label presence/absence indicators of questions, and $\Lambda^{'(d)} = (l'_1, \cdots, l'_K)$ is the vector of label clustering obtained by clustering the labels from $\Lambda^{(m)}$, where each $l_k, l'_c \in \{0, 1\}$.

The generative process for LC-LDA is as follows:

(1) For each topic $k \in \{1, \cdots, K\}$, we obtain the topic-word distribution $\phi_k = (\phi_{k,1}, \phi_{k,2}, \cdots, \phi_{k,v}) \sim Dir(\bullet \mid \beta)$ sampled from the Dirichlet distribution of the parameter β.
(2) For all topics of a document m, we obtain the vector of document's label sampled from the Bernoulli distribution of the parameter η. Each element in the vector is $\Lambda_i^{(m)} \in \{0, 1\} \sim Bernoulli(\bullet \mid \eta_i)$.
(3) The vector of document's label $\Lambda^{(m)}$ is mapped to the vector of label clustering $\Lambda^{'(m)}$ obtained by clustering the labels from $\Lambda^{(m)}$.
(4) Generate the Dirichlet prior parameter $\overrightarrow{\alpha}^{(m)} = L^{(m)} \times \overrightarrow{\alpha}$ of the document m's topic.
(5) Generate the topic polynomial distribution $\overrightarrow{\theta}^{(m)} = (\theta_{l_1}, \cdots, \theta_{l_{L_m}}) \sim Dir(\bullet \mid \overrightarrow{\alpha}^{(m)})$ of the document m sampled from the Dirichlet distribution of the parameter $\overrightarrow{\alpha}^{(m)}$.
(6) Generate the topic $z_{m,n}$ of the document m's word n sampled from the topic polynomial distribution of the parameter $\overrightarrow{\theta}^{(m)}$.
(7) Generate the word $w_{m,n}$ sampled from the topic polynomial distribution $\phi_{z_{m,n}}$.
(8) Repeat (5) to (7) until all words of the document m are generated.

The above $L^{(m)}$ is a document label projection matrix with a size of $L_m \times K$, and $\overrightarrow{\lambda}^{(m)} = \{k \mid \Lambda_k^{(m)} = 1\}$ is a vector of the document tag index. Then the elements in $L^{(m)}$ are as follows:

$$L_{ij}^{(d)} = \begin{cases} 1, & if\, \lambda_i^{(m)} = j \\ 0, & otherwise \end{cases} \tag{5}$$

Next, we use the Gibbs sampling [18] method to estimate the potential parameters θ and ϕ in LC-LDA. Specific steps are as follows:

(1) Assigning a topic $z^{(0)}$ to each word in the document randomly.
(2) Counting the number of feature words appearing under each topic z and the number of words in the topic z appearing in each document m. At each round the posterior probability $p(z_i = k \mid z_{\neg i,w})$ of the subject z_i is computed to exclude the distribution of the subject of the current word. According to the distribution of other words, the current word distribution of the topic is calculated. Then according to the probability distribution for the word, a new theme $z^{(j)}$ is randomly selected.
(3) Repeating step (2) to update the subject distribution of the next word until the subject distribution $\overrightarrow{\theta}$ for each document and the word distribution ϕ under each topic converge.

The posterior probability of the topic is calculated as follows:

$$\begin{aligned} p(z_i = k \mid z_{\neg i,w}) &= \frac{p(w,z)}{p(w,z_{\neg i})} \\ &= \frac{p(w \mid z)}{p(w_{\neg i} \mid z_{\neg i}) p(w_i)} \cdot \frac{p(z)}{p(z_{\neg i})}, k \in \overrightarrow{\lambda}^{(m)} \\ &\propto \frac{n_{k,\neg i}^{(t)} + \beta_t}{\sum_{t=1}^{V} n_{k,\neg i}^{(t)} + \beta_t} \cdot \frac{(n_{m,\neg i}^{(k)} + \alpha_k)}{\sum_{t=1}^{V} n_{m,\neg i}^{(t)} + \alpha_k} \end{aligned} \tag{6}$$

where z_i represents the topic variable of word i, $\neg i$ means other words that not include in the current word i, n_k^t is the count of word t in topic k, β_t is the Dirichlet prior parameter of word t, n_m^t is the count of topic k in document m, α_k is the Dirichlet prior parameter of topic k, and $k \in \overrightarrow{\lambda}^{(m)}$ means the topic distribution of document restricted by the set of labels.

When the Gibbs sample is converged, the parameters can be estimated from the following equation:

$$\begin{aligned} \phi_{k,t} &= \frac{n_k^{(t)} + \beta_t}{\sum_{t=1}^{V} n_k^{(t)} + \beta_t} \\ \theta_{m,k} &= \frac{n_m^{(k)} + \alpha_k}{\sum_{k=1}^{K} n_m^{(k)} + \alpha_k} \end{aligned} \tag{7}$$

where $\phi_{k,t}$ is the probability of using word k in topic t, and $\theta_{m,k}$ is the probability of using topic k in document m. Thus the data of answer and the question are mapped onto the topic space. We can obtain the distribution of answerer interest and question category respectively. Its specific process is as follows: Given a set of questions Q and answers U in CQA, we first construct the answerer's interest text sets for each answerer, which include the information of questions and the corresponding answer. Then the answerer's interest text sets and the set of questions Q are merged into a corpus D. Finally, we use LC-LDA to find the distribution of the answerer interest $Z^{(u)} = (z_1^{(u)}, z_2^{(u)}, \ldots, z_K^{(u)})$ and that of the question's category $Z^{(q)} = (z_1^{(q)}, z_2^{(q)}, \ldots, z_K^{(q)})$.

2.1.2. Feature extraction of the professional level of the answerer and the difficulty of the question based on TSWPR

In general, PageRank is often used as an estimate of the professional level of the answerer in CQA [13]. Thus, based on the topic distilled by LDA, Zhou et al. proposed Topic-sensitive PageRank (TPR) model [13] to find some users with highly professional ability in CQA. TPR constructs multiple user graphs according to the question–answer behaviors among the users in different topics. PageRank algorithm is then employed to obtain the rank of the users, which can be regarded as the professional level of users under different topics. However, it would be affected by a number of "indiscriminate" users. We use voting data and asker's satisfaction to measure the quality of the answer, and combine the result of Section 2.1.1 to improve the PageRank algorithm. The improved algorithm, i.e., the topic sensitive weighted PageRank (TSWPR) algorithm, can be used to calculate the professional level of askers and the difficulty of questions in different areas.

Based on the result distilled in Section 2.1.1, this study builds a topic sensitive weighted graph model to represent the relationship between questioners and the answerers:

$$G = (U, E) \tag{8}$$

where U is a set of nodes representing users (questioners and answerers), E is an edge set representing the interactive relationship between users. A directed edge $e_{ij} = (u_i, u_j) \in E$ indicates that user u_j answers the question posted by user u_i, and e_{ij} is associated with an affinity weight $w_z(i \to j)$ between u_i and u_j. The weight represents the probability that user j will answer questions of user i on topic z:

$$w_z(i \to j) = \begin{cases} \frac{\sum_{q \in Q(i) \cap A(j)} sim_z(q, u_j)}{\sum_{t: u_i \to u_t} \sum_{q \in Q(i) \cap A(t)} sim_z(q, u_t)}, & if (Q(i) \cap A(j)) \neq \phi \\ 0, & otherwise \end{cases} \tag{9}$$

where $\sum_{k=1}^{K} w_k(i \to j) = 1$, $w_z(i \to j)$ is usually not equal to $w_z(j \to i)$, $Q(i)$ is the set of questions posted by user i, $A(j)$ is the set of questions answered by user j, and $sim_z(q, u_j)$ represents the similarity of topic z between user j and question q. Therefore, this study improves the cosine similarity and proposes a method to calculate the similarity between users and questions on the topic:

$$sim_z(q, u_j) = \frac{p_z(q) \cdot \sum_{k=1}^{K} p_k(q) \cdot p_k(u_j)}{\sqrt{\sum_{k=1}^{K} p_k(q)^2} \sqrt{\sum_{k=1}^{K} p_k(u_j)^2}} \tag{10}$$

where $p_z(q)$ represents the probability distribution of the question q on the subject z, and $p_k(u_j)$ represents the probability distribution of the user j on the subject k.

However, Eq. (10) calculates the probability only from the perspective of subject similarity and number of answers. It does not take the quality of the answers into account by the user, so it will cause the "indiscriminate" of the user's ranking to rise.

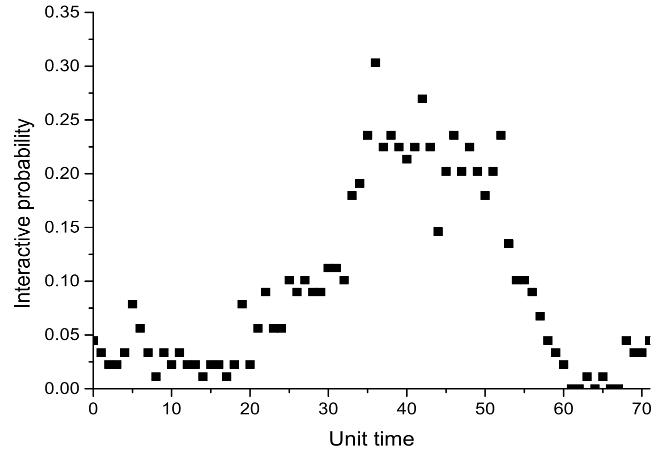

Fig. 4. The graph of user interaction probability.

Therefore, according to the score data of questions, Eq. (10) is improved as follows:

$$w_z(i \to j) = \frac{\sum q \in Q(i) \cap A(j) (sim(q, u_j) \cdot s + 1)}{\sum_{t: u_i \to u_t} \sum_{q \in Q(i) \cap A(t)} (sim(q, u_t) \cdot s + 1)}$$
$$s = \begin{cases} score, & score > 0 \\ 1, & score = 0 \\ 0, & score < 0 \\ 2 \cdot score, & Accepted \end{cases} \tag{11}$$

where *score* represents the score of answers, *Accepted* means that the answer is satisfied by the questioner. And factors are added to avoid the situation that the number of s is zero.

The topic sensitive PageRank algorithm can be used to calculate the authority value of the user in the subject z in a recursive manner at graph G.

$$R_{u_i}^{(z)} = \lambda \sum_{j: u_j \to u_i} R_{u_j}^{(z)} \cdot w_z(j \to i) + (1 - \lambda) p_z(u_i) \tag{12}$$

where $\lambda \in (0, 1)$ is a damping factor, which indicates that the user i has a probability of $(1-\lambda)p_z(u_i)$ to perform random answer to another user's question of subject z.

As the summary, in order to extract the characteristics of the answerers' profession and the questions' difficulty, this study constructs a directed graph model $G = (U, E)$ with weights to represent these interactions, and uses the TSWPR algorithm to calculate the authoritative value of each user in each subject area. The value is each user's professional level $L^{(u)} = (l_{z_1}^{(u)}, l_{z_2}^{(u)}, \ldots, l_{z_K}^{(u)})$. The algorithm uses $dif^{(q)} = (l_{z_1}^{(u)}, l_{z_2}^{(u)}, \ldots, l_{z_K}^{(u)})$ posted by users as the difficulty of questions.

2.1.3. Feature extraction of answerer activity based on LWLR

According to the analysis by Mingrong Liu et al. [19], the user's degree of activity may be quite different from each other. Those more "activity" users are more likely to answer questions raised by others. So, it is necessary to take user's activity into consideration when predicting the response time of the question in CQA sites. Fig. 4 shows that the average probability that a user participates in community interaction within three months in CQA. It implies that the probability of the user's interacting with others presents a certain degree of continuity.

In order to measure the user's degree of activity in each time period, this study makes the following definition:

Definition 3. User Interactive Index (UII): Given a user u, UII refers to the probability that the user interacts with others at a

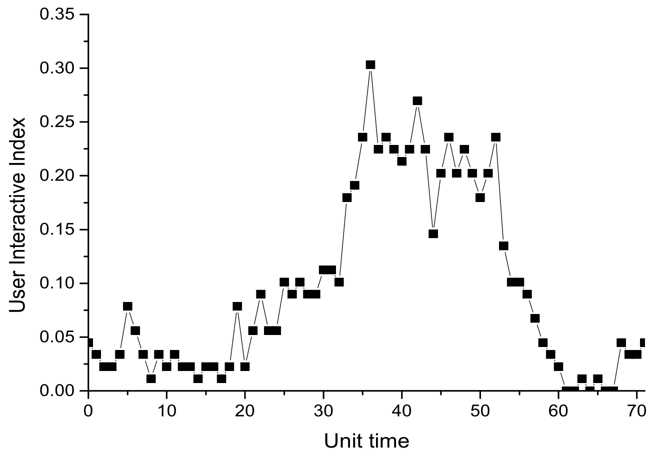

Fig. 5. The graph of user interaction index.

certain time. One day is divided into n unit time periods with length d of each period, so the interaction index $ap_{u_i,t}$ of user i in the unit time j becomes:

$$ap_{u_i,t} = \frac{| A_{u_i,j} |}{T} \tag{13}$$

where $A_{u_i,j}$ represents the set of valid interaction actions (such as question, answer, comment, and modifying answers or questions) by the user u_i at the j unit time. If a user has many interactive actions in a day of the j unit time, they are only recorded as one time. T represents the time span which is used to measure user activity and the length of one day.

In general, the probability that the user is active in the adjacent time is higher when the user is active at a certain time. It means the activity of user is affected by the activity of the adjacent time. Fig. 5 shows the user's interactive index of Fig. 4. It can be seen that UII is a series of fluctuating discrete data, so it is not appropriate to describe the user's activity with UII. Thus this study uses the Locally Weighted Linear Regression (LWLR) to smooth UII and continuous discrete data. And then we can obtain the user's activity function. The details of the algorithm are shown below:

For a $(t, ap_{u_i,t})$ (time $t = (1, 2, \cdots, \frac{day}{d})$, user interaction index $ap_{u_i,t}$), the user's activity function can be calculated by the following linear model:

$$Activity^{(\theta)}_{u_i}(t) = ap_{u_i,t} = \theta t \tag{14}$$

where θ is the coefficient of the continuous activity function. According to the above analysis, UII at target time t is highly relevant to the local influence (i.e., the activity index at the adjacent time t_j). Thus, we take the local influence into consideration when constructing the objective function $J_i(\theta)$ for estimating the parameter θ as Eq. (15):

$$\theta = \arg\min J_i(\theta) = \arg\min \frac{1}{2}\sum_{j=1}^{n} w_j (ap_{u_i,t_j} - Activity^{(\theta)}_{u_i}(t_j))^2 \tag{15}$$

where w_j denotes the weight, representing the influence of the activity index at adjacent time to target interactive index. The function for estimating w_j in Eq. (16) is to describe the effect of the model's error at these points on the empirical function based on the distance between the points to be predicted and the other points in the data set. If the time span between a time point and the predicted one is longer, its weight is smaller. Otherwise, the weight is larger.

$$w_j = exp(-\frac{(t_j - t)^2}{2\tau^2}) \tag{16}$$

where τ is the wavelength parameter to control the rate of declining in weight with distance. The regression coefficients at the time point t can be obtained by minimizing the experience function $J_i(\theta)$:

$$\theta_{it} = (T^T W T)^{-1} T^T W ap_{u_i,t} \tag{17}$$

where T represents the vector of all the time points, and W represents the vector of weights of all the time points.

2.1.4. The representation of question and answerer model

According to Ref. [9], we also extract some statistical features of answers and questions. These features are as follows: the title's average length of question answered by answerer $TitleLen_u$, the content's average length of question answered by answerer $BodyLen_u$, the average code segments of question answered by answerer $CodeNum_u$, the tag's average number of question answered by answerer $TagNum_u$, the tag's average popularity of question answered by answerer $TagPop_u$, the title's length of question $TitleLen_q$, the content's length of question $BodyLen_q$, the code segments of question $CodeNum_q$, the tag's number of question $TagNum_q$, and the tag's average popularity of question $TagPop_q$.

After extracting the characteristics of the answerers and the questions, a formal representation of the answerer model and the question model is given as the following:

Definition 4. The answerer model: Given an answerer u_i, the multivariate group $\overrightarrow{u_i} = (\overrightarrow{Z}^{(u_i)}, \overrightarrow{L}^{(u_i)}, Activity_{u_i}(t), \overrightarrow{O}_{u_i})$ represents the answerer model of $\overrightarrow{u_i}$, where $\overrightarrow{Z}$ and $\overrightarrow{L}$ represent the topic distribution and the degree of professionalism of answerers under different topics, respectively. $Activity_{u_i}(t)$ is the degree of activity of the answerer at time t, and $\overrightarrow{O}$ is the statistical features of answers.

Definition 5. The question model: Given a question q_i, the multivariate group $\overrightarrow{q_i} = (\overrightarrow{Z}^{(q_i)}, \overrightarrow{dif}^{(q_i)}, t, \overrightarrow{O}_{q_i})$ represents the question model of u_i, where $\overrightarrow{Z}$ and $\overrightarrow{dif}^{(q_i)}$ represent the topic distribution and the degree of professionalism of questions under different topics, respectively. t is the question time, and $\overrightarrow{O}_{q_i}$ is the statistical features of questions.

2.2. Question–answerer model matching strategy

The next step is to predict the response time of the question. The prediction of question response time is defined as predicting the time span r between the time when a question q is proposed and that when the corresponding answer is acquired:

$$time_{q,a} = t_{answer} - t_{question} + 1$$
$$r_q = \begin{cases} time_{q,a}, & if\ time_{q,a} < s \\ s, & if\ time_{q,a} \geq s \\ 0, & if\ no\ answer \end{cases} \tag{18}$$

where t_{answer} and $t_{question}$ represent the numbers of unit time of questions and answers submitted to the CQA respectively, s represents the number of unit time d contained in a day, thereinto, d is the unit time to describe the user's activity.

Through the above definition, we formulate the prediction as a multi-classification task and use the SoftMax classifier, a kind of multi-classifier, as the Question–Answerer model matching strategy to calculate the question response time. Next, we introduce the specific process to solve the question response time by SoftMax classifier.

Assume that the answer interval can be represented by m sets of a tuple $(\overrightarrow{x}, r)$, where $\overrightarrow{x}$ is the eigenvectors of $\langle answerers,$

questions⟩:

$$\begin{aligned}\overrightarrow{x} &= (x_1, x_2, \cdots, x_n) = (Sim_{Z_{q,u}}, Dis_{q,u}, Activity_u(t), t, \overrightarrow{O_q})\\ Sim_{q,u} &= Sim_{Z_{u,q}} \cdot Dis_{u,q}\\ Sim_{Z_{u,q}} &= \frac{\sum_{k=1}^{K} z_{u,k} \cdot z_{q,k}}{\sqrt{\sum_{k=1}^{K} z_{u,k}^2}\sqrt{\sum_{k=1}^{K} z_{q,k}^2}} \qquad (19)\\ Dis_{u,q} &= \sum_{k=1}^{K} z_{q,k} \cdot (l^d(u)_k - l(q)_k)\end{aligned}$$

$Sim_{Z_{q,u}}$ represents the topic similarity between answerers and questions, and it is calculated by the cosine similarity. $Dis_{u,q}$ represents the gap between answerer professional level and question difficulty. $z_{u,k}$ is the probability distribution of the user u on the subject k, and $z_{q,k}$ is the probability distribution of the question q on the subject k. $l_k^{(q)}$ is the professional level of the user u on the subject k, and $l_k^{(q)}$ is the difficulty level of the question q on the subject k.

SoftMax classifier takes $\overrightarrow{x}$ as input and the probability that the samples belong to each category as output. It can be written as follows:

$$h_{\overrightarrow{\theta}(x^{(i)})} = \begin{pmatrix} p(y^{(i)} = 1 \mid x^{(i)}; \overrightarrow{\theta}) \\ p(y^{(i)} = 2 \mid x^{(i)}; \overrightarrow{\theta}) \\ \vdots \\ p(y^{(i)} = s \mid x^{(i)}; \overrightarrow{\theta}) \end{pmatrix} = \frac{1}{Z}\begin{pmatrix} e^{\overrightarrow{\theta}_1^T \overrightarrow{x}^{(i)}} \\ e^{\overrightarrow{\theta}_2^T \overrightarrow{x}^{(i)}} \\ \vdots \\ e^{\overrightarrow{\theta}_s^T \overrightarrow{x}^{(i)}} \end{pmatrix} \qquad (20)$$

where $\overrightarrow{\theta_1}, \overrightarrow{\theta_2}, \cdots, \overrightarrow{\theta_s} \in \Re^{(n+1)}$ are the parameters of each category in the model and $Z = \sum_{j=1}^{k} e^{\overrightarrow{\theta}_j^T \overrightarrow{x}^{(i)}}$ is the normalized function. The SoftMax model uses the following empirical functions to optimize the parameters:

$$\begin{gathered} J(\theta) = -\frac{1}{m}[\sum_{i=1}^{m}\sum_{j=1}^{s} l\{y^{(i)} = j\} \log \frac{e^{\theta_j^T \overrightarrow{x}^{(i)}}}{\sum_{l=1}^{s} e^{\theta_l^T \overrightarrow{x}^{(i)}}}] + \frac{\lambda}{2}\sum_{i=1}^{s}\sum_{j=0}^{n}\theta_{ij}^2 \\ l\{ture\} = 1 \\ l\{false\} = 0 \end{gathered} \qquad (21)$$

where $\frac{\lambda}{2}\sum_{i=1}^{s}\sum_{j=0}^{n}\theta_{ij}^2$ is weight attenuation term which can be used to penalize excessive parameter values. Then the empirical function is derived as

$$\frac{\partial J(\theta)}{\partial \theta_j} = \frac{1}{m}\sum_{i=1}^{m}[\overrightarrow{x}^{(i)}(l\{y^{(i)} = j\} - p(y^{(i)} = j \mid \overrightarrow{x}^{(i)}; \overrightarrow{\theta}))] + \lambda \overrightarrow{\theta}_j \qquad (22)$$

The parameters of the model can be obtained by the gradient descent method. The maximum probability of the output of the SoftMax function is the response time for the user to answer the question. In summary, the detail steps of QAM^2 are described in Algorithm 1.

Algorithm 1 QAM^2

Input:
A new question, q_{new};
Answerer label answer set, D_U;
Answerer set, U;
Question set, Q;
User's question-answering behavior set, G;
User's interaction time set, T;
Answerer answer set, Q_U
Output:
The new question response time, t;

1. Calculate the distribution of the answerer interest θ_U and the question's category θ_Q;
   (a) θ_U,$\theta_Q = LC - LDA(D_Q, D_U)$;
2. Calculate the professional level of answerers R;
   (a) $R = TSWPR(G, \theta_U)$;
3. Construct the question model $MQAM_q^Q$;
   (a) **for** each $q \in Q$ **do**
   $Difficult_q = \overrightarrow{R_{asker}}$;
   $\overrightarrow{O_q} = StatisticalAnalysis(Q_q)$;
   $MQAM_q^Q = (\theta_q, Difficult_q, t, \overrightarrow{O_q})$;
   (b) **end for**
4. Construct the answerer model $MQAM_u^U$ and the QAM_u^2 for each user u;
   (a) **for** each $u \in U$ **do**
   $UserActive_u = LWLR(T_u)$;
   $\overrightarrow{O_u} = StatisticalAnalysis(Q_u)$;
   $MQAM_u^U = (\theta_u, \overrightarrow{R_u}, UserActive_u, \overrightarrow{O_u})$;
   $QAM_u^2 = Softmax(MQAM_u^U, MQAM_q^Q)$;
   (b) **end for**
5. Calculate the response time t of each user u to the new question q_{new};
   (a) $t = QAM^2(MQAM_u^U, MQAM_{q_{new}}^Q)$;

return t;

3. Experimental evaluation

3.1. Data set

Two Community Question Answering data sets from Stack Overflow are used in experiments. One data set is collected from an API supplied by Stack Overflow,[4] which contains the questions of Stack Overflow community from February to May of 2014. Another data set is published by Doris Hoogeveen from The University of Melbourne [20], containing records of Stack Overflow community from August 2012 to September 2014. The statistics of these two data sets are listed in Table 1. In the data sets, the questions which have not been answered are deleted, and the users refer to those who answered questions for at least 14 times. Furthermore, each data set is organized into six subsets:

- The question set. It consists of the question's title, content, questioner, label, and the posted time.
- The label set of question.
- The answer set of answerer.
- A set of questions answered by answerer.
- A set of question-answering behavior of user. It consists of answerer, questioner, the answered time, the voting score of the question, the satisfaction of the answer, and the question response time.
- A set of history interaction time of user.

[4] http://api.stackexchange.com/docs

Table 1
Statistics of Stack overflow-API data and CQADupStack data.

Data set	Number of questions	Number of answers	Number of comments	Number of users
Stack overflow-API data	3961	4145	12955	512
CQADupStack data	3380	6159	72664	443

Besides, the 5-fold cross-validation method is employed in constructing the predictions models to partition the training and testing data, avoiding contingency during the modeling.

3.2. Experimental settings

3.2.1. Evaluation metrics

In this section, several evaluation metrics in perplexity and accuracy are adopted in this paper to measure the effectiveness of LC-LDA model and QAM^2. They are described briefly as below.

Perplexity is a common criterion to evaluate the language models and is used to evaluate the performance of LDA and LC-LDA methods in this paper. The perplexity is computed by Eq. (23).

$$perplexity(D) = \exp\left(-\frac{\sum_{d=1}^{M}\sum_{m=1}^{N_d}\log p(w_{dm})}{\sum_{d=1}^{M} N_d}\right) \tag{23}$$

where M is the number of questions, N_d denotes the number of terms of question d, and $P(w_{dm})$ represents the probability of the mth term in the learned language model D.

Accuracy is used to evaluate the effectiveness of predicting question response time. As the time series are divided into multiple unit times to forecast, the accuracy is computed through Eq. (24).

$$Accuracy = \frac{\|R_a\|}{\|Test_a\|} \tag{24}$$

where R_a is the number of questions whose response time is predicted correctly, $Test_a$ represents the number of questions in testing data. The QAM^2 method proposed in this paper calculates the response time of each answerer to the question by model matching, To evaluate methods for question response time prediction comprehensively, we adopted four different evaluation metrics based on *Accuracy*:

- $Accuracy_u$ is used to evaluate the prediction accuracy of response time from the respect of the user. The R_a of $Accuracy_u$ represents the number of users whose response time is predicted correctly, and the $Test_a$ denotes the number of answers in the testing data.
- $Accuracy_{min}$ is used to evaluate the prediction accuracy of the fastest response time to the question. The R_a of $Accuracy_{min}$ represents the number of questions whose fastest response time is predicted correctly, and the $Test_a$ denotes the number of questions in the testing data.
- $Accuracy_{most}$ is used to evaluate the prediction accuracy of modal number in response time of all the users, that is value that is repeated most often in the predicted results. The R_a of represents the number of questions whose mode of predicted response time is correct, and the $Test_a$ denotes the number of questions in the testing data.
- $Accuracy_{total}$ is used to evaluate the prediction accuracy of the average response time of total users to the question. The R_a of $Accuracy_u$ represents the number of questions whose average response time of total users is predicted correctly, and the $Test_a$ denotes the number of questions in the testing data.

3.2.2. Comparisons with different methods

The following approaches are used in the experiments:

- LDA and LC-LDA are used to categorize answerers respectively.
- Topic PageRank (TPR) and TSWPR are used to sort answerers respectively.
- UII and LWLR are used to describe the user's activity respectively.
- SoftMax classifier and Backpropagation Neural Network (BPNN) are used to predict the response time of questions.

In order to verify the effectiveness of our proposed method based on QAM^2, the above methods are used to construct different models to make comparisons.

(1) Text Statistic Based Method (TSM): This method proposed by Bhat et al. [4] predicts the question response time based on the text statistical characteristics.
(2) LDA-TPR-SoftMax (LTS): this method constructs the question model and the answerer model based on the characteristics extracted by TPR and LDA, then SoftMax is used to match these two models and predicts the question response time for each answerer.
(3) LDA-TPR-UII-SoftMax (LTUS): this method not only considers the characteristics extracted by TPR and LDA, but also adds user's activity distilled by UII to construct the eigenvectors of $\langle answerers, questions\rangle$, and then the SoftMax model is used to calculate the question response time for each answerer.
(4) LDA-TPR-LWLR-SoftMax (LTLS): this method is similar to LTUS, but the only difference is that the user's activity is extracted by LWLR.
(5) LDA-TSWPR-LWLR-SoftMax (LTSLS): the difference between LTLS and LTSLS is that LTSLS acquires answerer's authoritative value on the basis of TSWPR, other methods of distilling features are consistent.
(6) LC-LDA-TPR-LWLR-SoftMax (LCTLS): the difference between LTLS and LCTLS is that the distribution of the answerer interest and that of question's category are distilled by LC-LDA in LCTLS.
(7) LC-LDA-TSWPR-SoftMax (LCTSS): the difference between LTS and LCTSS is that LCTSS utilizes LC-LDA and TSWPR to extract the corresponding characteristics, and then construct the question model and the answerer model.
(8) LC-LDA-TSWPR-UII-SoftMax (LCTSUS): this method adds user's activity distilled by UII to construct the eigenvectors of $\langle answerers, questions\rangle$, and other methods of distilling features are consistent with LCTSS.
(9) LC-LDA-TSWPR-LWLR-SoftMax (QAM^2): this method is the proposed method for predicting the question response time in this paper.
(10) LC-LDA-TSWPR-LWLR-BPNN (QAM^2-NN): QAM^2-NN is also based on these deep-seated features (such as the answerer's interest, professional level, activity degree, the question category, difficulty, and so on), which are same as QAM^2. However, QAM^2-NN uses a BPNN with four layers architecture to predict the response time.

where (2), (3) and (4) comparing with (7), (8) and (9) is to compare the accuracy of response time under different user activity

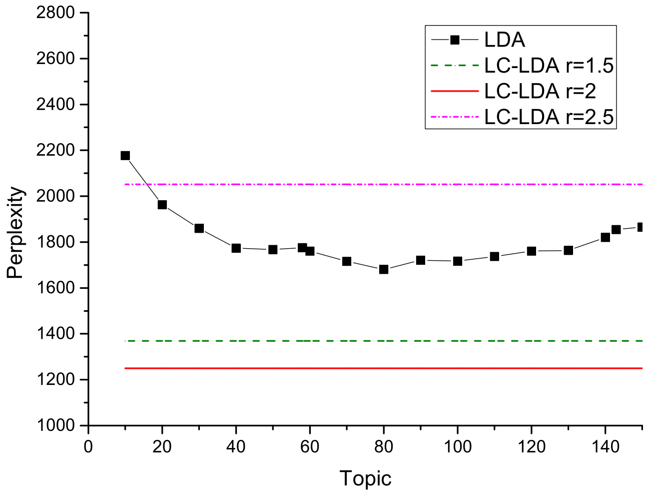

Fig. 6. The experiment results of LC-LDA and LDA.

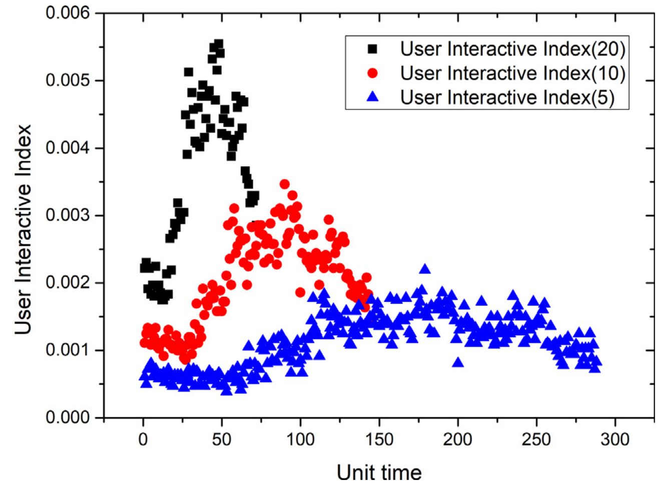

Fig. 7. The statistic results of UII.

calculation methods. The comparison among (4), (5), (6) and (9) verifies the impact of the professional level of answerer on the prediction of question response time. (1),(9) and (10) can verify the effectiveness of the method proposed by this paper.

3.2.3. The process of experiment and parameter setting

The process of the experiment is introduced as the following. First, we extract statistical features of answerers and questions (e.g., the title's length of the question) from the answerer's question set and the answer set respectively, then clean up the text of the six sets (convert the text into bags of words by removing punctuation, stop words and root). Second, we construct the answerer model and question model through the MQAM model. Finally, the answerer's characteristics are combined with question's characteristics according to the answerer's question-answering behavior set. In practice, the number of answers is very small relative to the number of people in CQA. It is likely to cause the data to be unbalanced, so we use the method [21] of resolving unbalanced data to extract question and answer examples, and build a SoftMax model for each answerer. Then we use the SoftMax model to calculate the question response time of training set for each answerer, and finally calculate the response time of the community to the question.

The prediction of question response time based on QAM^2 uses the following parameters: the expansion coefficient r in Eq. (4), the Dirichlet hyperparameters α, β of LC-LDA, the damping factor λ in Eq. (13), and the unit time d and the wavelength parameter τ. The Dirichlet hyperparameters are set as $\alpha = 0.5$, $\beta = 0.1$ [22], the damping factor is set as $\lambda = 0.2$ [23], and the wavelength parameter $\tau = 2.0$. The determination of the expansion coefficient r and the unit time d will be introduced in the next section.

3.3. Experimental results

In this section, in order to verify the effectiveness of the proposed methods, including LC-LDA and QAM^2, and determine some parameters (the expansion coefficient r and the unit time d) that can impact the performance of prediction, we will conduct several experiments on the Stack overflow-API data set and the CQADupStack data set.

3.3.1. The comparison between LDA and LC-LDA

To verify the effectiveness of LC-LDA, we conduct the comparative experiments between LDA and LC-LDA, which are used for classifying the question of experimental data through the topic distribution. The experimental results are presented in Fig. 6. LC-LDA model finds a total of 57 topics when the expansion coefficient r is 1.5. The perplexity is 1369.41 in this case, which is 22.87% lower than that of LDA model with the same number of topics. When the expansion coefficient r is 2.0, the LC-LDA model finds 142 topics and its perplexity is 1250.27, which is 32.57% lower than that of LDA model with the same number of topics. The decrease of the expansion coefficient indicates that the probability of the question belonging to the individual topic is higher. Therefore, when the expansion coefficient r is 1.5 or 2.0, the LC-LDA model outperformed LDA model in terms of finding topics. To obtain better classification results, the expansion coefficient r is set to be 2.0 in the next experiments.

3.3.2. The determine of unit time

In order to determine the unit time mentioned in Section 2.1.3, we calculated statistics about the average probability that users participate in community interaction within a day in CQA when the unit time is 5, 10 and 20 min. The results are shown in Fig. 7. When the unit time is 20 min, we can see obvious differences among the value of UII per unit time, which makes it easier to reflect the user's activity level in each unit time. Furthermore, the values of UII in most unit times are close to zero when the unit time is 5 or 10 min, and it is not conducive to use the Locally Weighted Linear Regression to smooth these continuous of discrete data. Therefore, the unit time is set to be 20 min, which is also the unit time of the question response time prediction. One day is divided into 72 units according to the unit time when predicting question response time, and then the time right after one day is regarded as the 73rd unit. Thus, the time sequences are divided into 73 units to predict the question response time.

3.3.3. The validation of QAM^2

We will validate the accuracy of the proposed method from three perspectives including expert factors, user activity and the different prediction of response time models. The experimental results are shown in Tables 2 and 3.

(1) The evaluation of methods for extracting expert factors

Fig. 8 shows that the $Accuracy_{total}$ of LC-LDA based LCTLS and QAM^2 is respectively higher than that of LTLS and LTSLS using LDA on the two data sets, indicating that the prediction of question response time is indeed affected by the classification of

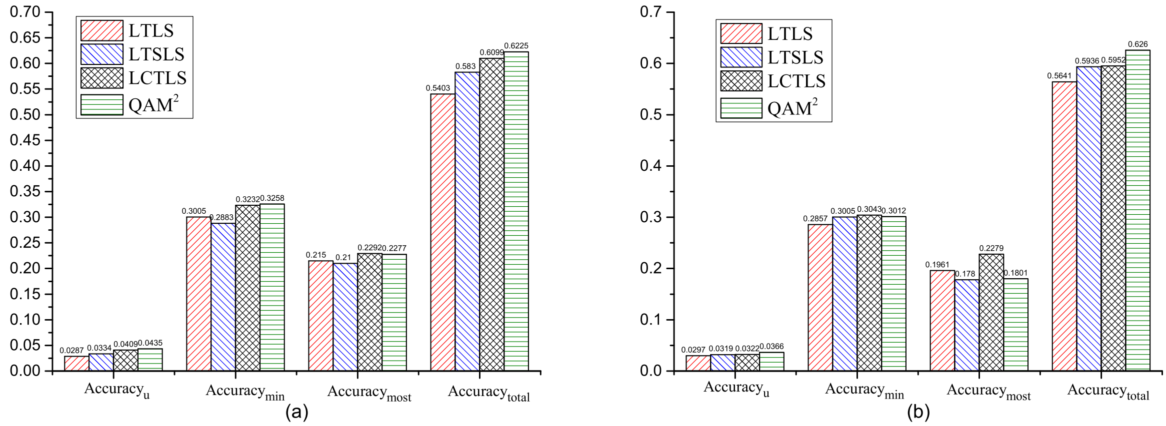

Fig. 8. The prediction accuracy of question response time under different methods of answerers classification and sorting on two data sets. (a) exhibits the experimental results on Stack overflow-API data, while (b) shows the experimental results on CQADupStack data.

Table 2
Comparisons of prediction performance (test errors, mean±std.) based on different prediction of response time models.

	CQADubStack			
	$Accuracy_u$	$Accuracy_{min}$	$Accuracy_{most}$	$Accuracy_{total}$
TSM	.0066±.004	.0834±.081	.0921±.087	.0907±.081
LTS	.0111±.004	.1165±.046	.1216±.025	.2395±.037
LTUS	.0234±.019	.2679±.045	.2360±.128	.4646±.024
LTLS	.0297±.017	.2857±.052	.1961±.141	.5641±.082
LTSLS	.0319±.022	.3005±.066	.1780±.143	.5936±.049
LCTLS	.0322±.007	.3043±.038	.2279±.031	.5952±.006
LCTSS	.0260±.015	.2419±.049	.1544±.063	.5409±.086
LCTSUS	.0317±.014	.2509±.096	.1985±.064	.5396±.054
QAM^2-NN	.0346±.019	.3187±.113	.0890±.071	.6015±.132
QAM^2	.0366±.006	.3012±.067	.1801±.092	.6260±.053

Table 3
Comparisons of prediction performance (test errors, mean±std.) based on different prediction of response time models.

	Stack overflow-API data			
	$Accuracy_u$	$Accuracy_{min}$	$Accuracy_{most}$	$Accuracy_{total}$
TSM	.0040±.003	.0043±.003	.0043±.003	.0043±.003
LTS	.0109±.007	.3024±.007	.2134±.032	.0109±.012
LTUS	.0252±.014	.3103±.026	.2180±.042	.5608±.233
LTLS	.0287±.012	.3005±.125	.2150±.091	.5403±.225
LTSLS	.0334±.022	.2883±.034	.2100±.082	.5830±.323
LCTLS	.0409±.021	.3232±.013	.2292±.096	.6099±.261
LCTSS	.0262±.015	.3290±.136	.2164±.092	.6200±.261
LCTSUS	.0336±.016	.3227±.133	.2223±.093	.6278±.262
QAM^2-NN	.0501±.004	.3641±.009	.1602±.081	.5865±.048
QAM^2	.0435±.019	.3258±.134	.2277±.097	.6225±.261

the answers. Similarly, it can be observed that the TSWPR based methods are superior to the TPR based methods when extracting the professional level of the answerer and the difficulty of the question. For example, on the Stack overflow-API data set, the $Accuracy_{total}$ of TSWPR-based QAM^2 is 2% higher than that of the TCR-based LCTLS. Moreover, on the CQADupStack data set, the $Accuracy_{total}$ of LTSLS based on TSWPR is also 3% higher than that of LTLS using TPR.

(2) The evaluation of methods for extracting user-activity

From Fig. 9, it can be found that the methods without considering user activity have the lower $Accuracy_u$ and $Accuracy_{total}$, which indicates that the user activity plays an important role in response time prediction. Furthermore, the LWLR based methods (LTLS and QAM^2) exhibit better performance on $Accuracy_u$ than UII based methods (LTUS and LCTSUS) on the two data sets. Especially in Fig. 9 (c), QAM^2 has increased by 29.4% in $Accuracy_u$ compared to LCTUS on Stack overflow-API data. In most cases, the $Accuracy_{total}$ of LWLR based methods is higher than that of the methods relying on UII. Through the detailed comparisons between two different user activity extraction methods, the rationality and necessity of introducing user activity into response time prediction are demonstrated.

(3) Comparison of our proposed method with traditional methods

Fig. 10(a) and (b) show the effects of three different models (i.e., TSM, QAM^2-NN and QAM^2-S) on the prediction of question response time. To enrich the comparative experiment, we consider the TSM model's prediction accuracy from two aspects: the community's perspective TSM (constructing a model based on all the data in the community), and the user's personal perspective TSM-Single (constructing a specific model for each user).

From Fig. 10 we can see that QAM^2-S is respectively 6.16% and 15.97% higher than TSM model in prediction accuracy of $Accuracy_{min}$ and $Accuracy_{most}$ on Stack overflow-API data. Similarly, on CQADupStack data, QAM^2-S is 6.16% and 15.97% higher than TSM model in prediction accuracy of $Accuracy_{min}$ and $Accuracy_{most}$, respectively. Thus, the prediction of response time method proposed in this paper is obviously better than the traditional method based on the statistical features.

By comparing the experimental results of QAM^2-NN and QAM^2-S on the two data sets, it can be found that the $Accuracy_u$ of the two methods are very closed. Similarly, through comparison of these two methods on predicting the question response time, it can be seen that the difference of $Accuracy_{total}$ between them is also not significant. It indicates the effectiveness of these characteristics of the answerers and the questions in predicting the answer response time.

4. Conclusion and future work

This paper proposes a novel approach to predict the question response time in CQA by constructing Multi-feature based Question–Answerer Model. This model first distills some deep-seated features (such as the answerer's interest, professional level, activity degree, the question category, difficulty, and so on) related to question time from user's history records in CQA. And in the process of extracting features, we propose three improved methods (LC-LDA, TSWPR and LWLR) to solve over-fitting or under-fitting of the topic extraction, the misclassification of some "indiscriminate" users, and the inaccurate calculation of user activity caused by traditional methods (L-LDA, TPR and UII),

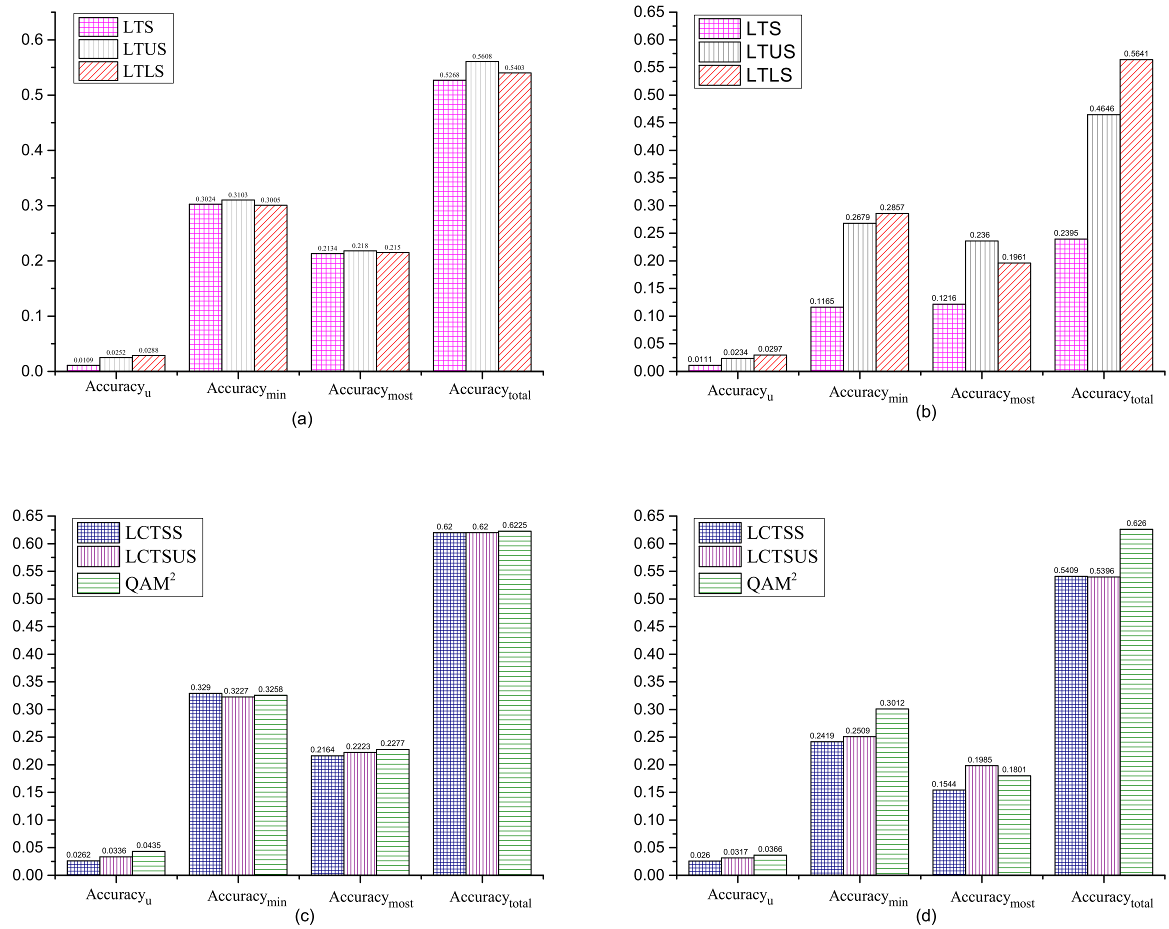

Fig. 9. The prediction accuracy of question response time under different activity measurement method on two data sets. (a) and (c) exhibit the experimental results on Stack overflow-API data, while (b) and (d) show the experimental results on CQADupStack data.

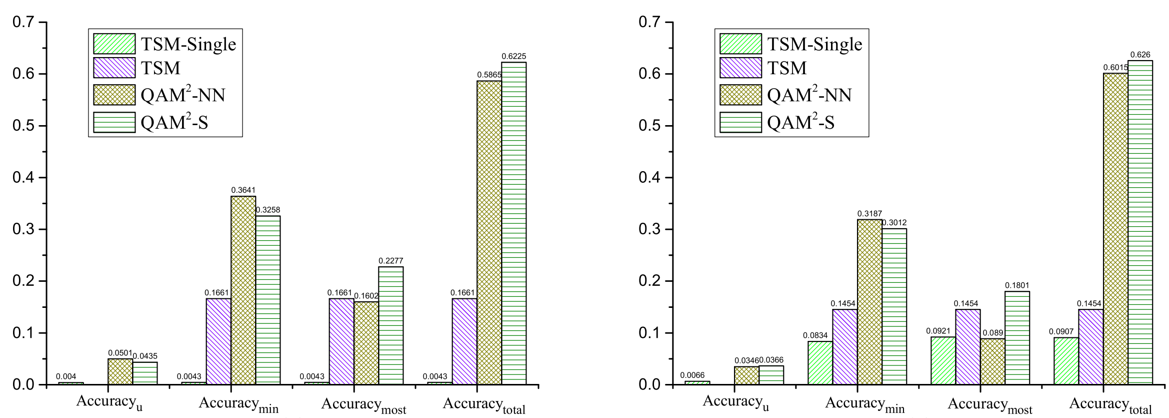

Fig. 10. The prediction accuracy of question response time under different algorithms on two data sets. (a) exhibits the experimental results of Stack overflow-API data, while (b) show the experimental result of the CQADupStack data.

respectively. Next, this model constructs the answerer model and the question model based on these deep-seated features. Finally, compared with traditional methods, we treat the prediction as a multi-classification task for obtaining more accurate results by using the model matching strategy based on multi-classifier to estimate the question response time of each answerer. We conduct extensive experiments using some real data sets from the Stack Overflow. The experimental results show that our proposed method outperforms the traditional methods and achieves the state-of-the-art performance.

The future study of this research could be continued in a few ways. First, the time dependency of the correlation between question terms and topics will lead to the deviation of the model of a question, thus affecting the accuracy of the response time. Thus, more accurately estimating the time dependency will benefit the model significantly. Second, it is also a popular solution to analyze question response time from the perspective of the content semantics. The introduce of the semantic model such as Word2Vec to our topic-based method may have significant improvement on the prediction of question response time. Additionally, the response time prediction in CQA can help users establish psychological expectations for obtaining the answer time. However, higher quality answers often require longer response times in CQA, and the first answer may not be a higher quality answer. If we can accurately predict the arrival time of a high-quality answer, then the questioner's satisfaction can be greatly improved. Therefore, not only the semantic information of texts but also the quality of answers should be considered in the future.

Acknowledgment

This research is supported by the National Key R&D Program of China (Grant Nos. 2016YFB1000600, 2016YFB1000601, and 2016YFB10006011).

References

[1] M. Qu, G. Qiu, X. He, C. Zhang, H. Wu, J. Bu, C. Chen, Probabilistic question recommendation for question answering communities, in: Proceedings of the 18th International Conference on World Wide Web, WWW '09, ACM, New York, NY, USA, 2009, pp. 1229–1230, http://dx.doi.org/10.1145/1526709.1526942.

[2] Z. Ji, F. Xu, B. Wang, B. He, Question-answer topic model for question retrieval in community question answering, in: Proceedings of the 21st ACM International Conference on Information and Knowledge Management, CIKM '12, ACM, New York, NY, USA, 2012, pp. 2471–2474, http://dx.doi.org/10.1145/2396761.2398669.

[3] A. Pal, J.A. Konstan, Expert identification in community question answering: Exploring question selection bias, in: Proceedings of the 19th ACM International Conference on Information and Knowledge Management, CIKM '10, ACM, New York, NY, USA, 2010, pp. 1505–1508, http://dx.doi.org/10.1145/1871437.1871658.

[4] V. Bhat, A. Gokhale, R. Jadhav, J. Pudipeddi, L. Akoglu, Effects of tag usage on question response time, Soc. Netw. Anal. Min. 5 (1) (2015) 24, http://dx.doi.org/10.1007/s13278-015-0263-3.

[5] M. Asaduzzaman, A.S. Mashiyat, C.K. Roy, K.A. Schneider, Answering questions about unanswered questions of stack overflow, in: Proceedings of the 10th Working Conference on Mining Software Repositories, MSR '13, IEEE Press, Piscataway, NJ, USA, 2013, pp. 97–100, http://dx.doi.org/10.1109/MSR.2013.6624015.

[6] P. Arunapuram, J.W. Bartel, P. Dewan, Distribution, correlation and prediction of response times in stack overflow, in: 10th IEEE International Conference on Collaborative Computing: Networking, Applications and Worksharing, 2014, pp. 378–387, http://dx.doi.org/10.4108/icst.collaboratecom.2014.257265.

[7] J. Goderie, B.M. Georgsson, B. van Graafeiland, A. Bacchelli, ETA: Estimated time of answer predicting response time in stack overflow, in: Proceedings of the 12th Working Conference on Mining Software Repositories, MSR '15, IEEE Press, Piscataway, NJ, USA, 2015, pp. 414–417, http://dx.doi.org/10.1109/MSR.2015.52.

[8] H. Hu, B. Liu, B. Wang, M. Liu, X. Wang, Exploring social features for answer quality prediction in CQA portals, in: 2013 International Conference on Machine Learning and Cybernetics, Vol. 04, 2013, pp. 1904–1909, http://dx.doi.org/10.1109/ICMLC.2013.6890906.

[9] N. Burlutskiy, A. Fish, N. Ali, M. Petridis, Prediction of users' response time in Q amp;amp;A communities, in: 2015 IEEE 14th International Conference on Machine Learning and Applications, ICMLA, 2015, pp. 618–623, http://dx.doi.org/10.1109/ICMLA.2015.190.

[10] T. Yan, R. Tourangeau, Fast times and easy questions: The effect of age, experience, and question complexity on web survey response times, Appl. Cogn. Psychol. 22 (1) (2010) 51–68, http://dx.doi.org/10.1002/acp.1331.

[11] P. Berger, P. Hennig, T. Bocklisch, T. Herold, C. Meinel, A journey of bounty hunters: analyzing the influence of reward systems on stackoverflow question response times, in: 2016 IEEE/WIC/ACM International Conference on Web Intelligence, WI, 2016, pp. 644–649, http://dx.doi.org/10.1109/WI.2016.0114.

[12] W. Yong, Research On Driving Factors Of Douban User Activity Based On User Experience (Master's thesis), Beijing University of Posts and Telecommunications, 2015.

[13] G. Zhou, J. Zhao, T. He, W. Wu, An empirical study of topic-sensitive probabilistic model for expert finding in question answer communities, Knowl.-Based Syst. 66 (2014) 136–145, http://dx.doi.org/10.1016/j.knosys.2014.04.032.

[14] D.M. Blei, A.Y. Ng, M.I. Jordan, Latent dirichlet allocation, J. Mach. Learn. Res. 3 (2003) 993–1022, http://dx.doi.org/10.1162/jmlr.2003.3.4-5.993.

[15] D. Ramage, D. Hall, R. Nallapati, C.D. Manning, Labeled LDA: A supervised topic model for credit attribution in multi-labeled corpora, in: Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Vol. 1 – Vol. 1, EMNLP '09, Association for Computational Linguistics, Stroudsburg, PA, USA, 2009, pp. 248–256.

[16] S.M. Van Dongen, Graph Clustering by Flow Simulation (Ph.D. thesis), Ph.D. Thesis University of Utrecht, 2000.

[17] M. Gasparini, Markov chain Monte Carlo in practice, Technometrics 39 (3) (1997) http://dx.doi.org/10.1080/00401706.1997.10485132, 338–338.

[18] A.E. Gelfand, S.E. Hills, A. Racine-Poon, A.F.M. Smith, Illustration of Bayesian inference in normal data models using gibbs sampling, J. Amer. Statist. Assoc. 85 (412) (1990) 972–985, http://dx.doi.org/10.1080/01621459.1990.10474968.

[19] M. Liu, Y. Liu, Q. Yang, Predicting best answerers for new questions in community question answering, in: Web-Age Information Management, Springer Berlin Heidelberg, Berlin, Heidelberg, 2010, pp. 127–138.

[20] D. Hoogeveen, K.M. Verspoor, T. Baldwin, CQADupStack: A benchmark data set for community question-answering research, in: Proceedings of the 20th Australasian Document Computing Symposium, ADCS '15, ACM, New York, NY, USA, 2015, pp. 3:1–3:8, http://dx.doi.org/10.1145/2838931.2838934, URL http://doi.acm.org/10.1145/2838931.2838934.

[21] H. He, E.A. Garcia, Learning from imbalanced data, IEEE Trans. Knowl. Data Eng. 21 (9) (2009) 1263–1284, http://dx.doi.org/10.1109/TKDE.2008.239, URL https://doi.org/10.1109/TKDE.2008.239.

[22] G. Zhou, S. Lai, K. Liu, J. Zhao, Topic-sensitive probabilistic model for expert finding in question answer communities, in: Proceedings of the 21st ACM International Conference on Information and Knowledge Management, CIKM '12, ACM, New York, NY, USA, 2012, pp. 1662–1666, http://dx.doi.org/10.1145/2396761.2398493.

[23] L. Page, The pagerank citation ranking : Bringing order to the web, stanford digital libraries working paper, 1998, pp. 1–14, 9 (1), URL http://ilpubs.stanford.edu:8090/422/.