Inf Retrieval J (2017) 20:4–24
DOI 10.1007/s10791-016-9287-1

CONSTRUCTING CLICK MODELS FOR SEARCH USERS

Modeling users' search sessions for high utility query recommendation

Jiafeng Guo[1] · **Xiaofei Zhu**[2] · **Yanyan Lan**[1] · **Xueqi Cheng**[1]

Received: 11 February 2016 / Accepted: 20 August 2016 / Published online: 20 September 2016

Abstract Query recommendation has long been considered a key feature of search engines, which can improve users' search experience by providing useful query suggestions for their search tasks. Most existing approaches on query recommendation aim to recommend relevant queries, i.e., alternative queries similar to a user's initial query. However, the ultimate goal of query recommendation is to assist users to reformulate queries so that they can accomplish their search task successfully and quickly. Only considering relevance in query recommendation is apparently not directly toward this goal. In this paper, we argue that it is more important to directly recommend queries with high utility, i.e., queries that can better satisfy users' information needs. For this purpose, we attempt to infer query utility from users' sequential search behaviors recorded in their search sessions. Specifically, we propose a dynamic Bayesian network, referred as *Query Utility Model* (QUM), to capture query utility by simultaneously modeling users' reformulation and click behaviors. We then recommend queries with high utility to help users better accomplish their search tasks. We empirically evaluated the performance of our approach on a publicly released query log by comparing with the state-of-the-art methods. The experimental results show that, by recommending high utility queries, our approach is far more effective in helping users find relevant search results and thus satisfying their information needs.

✉ Jiafeng Guo
guojiafeng@ict.ac.cn

Xiaofei Zhu
zxf@cqut.edu.cn

Yanyan Lan
lanyanyan@ict.ac.cn

Xueqi Cheng
cxq@ict.ac.cn

[1] Institute of Computing Technology, Chinese Academy of Sciences, No. 6 Kexueyuan South Road, Haidian District, Beijing, People's Republic of China

[2] Chongqing University of Technology, No. 69 Hongguang Road, Banan District, Chongqing, People's Republic of China

Keywords Search behavior · Query utility · Dynamic Bayesian network · Query recommendation

1 Introduction

Nowadays, search engines have become an indispensable tool for users to accomplish various information seeking tasks, e.g., finding particular Web pages, locating target resources, or accessing information of certain topics. However, it is often difficult for users to formulate a query that can describe their information needs properly, and find the right results quickly. Most users may reformulate their queries several times before satisfaction. According to our analysis, more than 60 % of search sessions contain query reformulation.[1] Therefore, how to assist users to better formulate queries becomes a valuable and challenging problem for modern search engines.

Query recommendation has been widely recognized as a key technique to alleviate users' reformulation burden and improve the usability of search engines. A major approach on query recommendation is relevant query recommendation, which focuses on providing alternative queries similar to a user's initial query. Collective user behaviors are usually employed to help find similar queries. For example, queries that share common terms (Wen et al. 2001), click the same URLs (Beeferman and Berger 2000; Li et al. 2008; Mei et al. 2008) or occur in the same search sessions (Zhang and Nasraoui 2006; Boldi et al. 2008) are often considered as similar.

However, the ultimate goal of query recommendation is to assist users to reformulate queries so that they can acquire their desired information successfully and quickly. Only recommending similar queries, as in relevant query recommendation approaches, is apparently not directly toward this goal. For example, given a user's initial query "company name original seven" which aims to find "which company made the original seven movie", the possible recommendations may include "company name original 7", "original seven company" and "seven movie". Obviously, the three recommended queries are all similar to the user's initial query, but only the last one can find search results satisfying the user's need. If the user clicks on the first two recommendations, he/she may feel frustrated since no satisfied results can be found through these recommendations.

Therefore, in this paper, we argue that it is more important and beneficial to directly recommend queries with high utility, i.e., queries that can better satisfy users' information needs. Intuitively, a query should satisfy two conditions to be useful for users: (1) The query's search results should be attractive to the user, so that she would like to click the results for further inspecting; (2) The clicked search results can actually satisfy the user as they are relevant to her original information needs. Therefore, the query utility can be further specified by two components. One is related to the *attractiveness* of the query's search results, referred to as *perceived utility*. The other is related to the *satisfaction* from the clicked search results, referred to as *posterior utility*. The query utility is then defined as the product of the two components. We will show that under this definition, the query utility stands for the expected satisfaction users obtain from the search results of the query according to their original information needs.

By recommending high utility query, we actually emphasize users' *post-click satisfaction*, i.e., whether users' information needs will be satisfied after clicking the

[1] The analysis is conducted on the query log shared by Yandex (http://imat-relpred.yandex.ru/en/), which contains nearly 44 million search sessions.

recommendation. We argue that users' post-click satisfaction reflects the true effectiveness of query recommendation. It is worth noting that although the concept "utility" has been used in literatures of query recommendation (Jain et al. 2011; Anagnostopoulos et al. 2010), they are significantly different from our work. Both studies (Jain et al. 2011; Anagnostopoulos et al. 2010) define a global utility function over the recommendation set, which emphasize either the diversity (Jain et al. 2011) or the expected click-through rate (Anagnostopoulos et al. 2010) of the recommendations. They do not define and learn the query utility towards users' post-click satisfaction as ours.

The central challenge in our recommendation problem is how to learn the query utility according to users' original information needs. The basic idea is that users' collective search behaviors, especially their sequential reformulation and click behaviors in search sessions, embed rich information on the usefulness of queries. Therefore, we propose a novel dynamic Bayesian network, referred as *Query Utility Model* (QUM), to inference query utility by simultaneously modeling users' reformulation and click behaviors. By learning query utility in hand, we can provide query recommendations with high utility to help users better accomplish their search tasks.

To evaluate the performance of our approach, we compare it with the state-of-the-art query recommendation approaches based on a publicly released query log. We propose two evaluation metrics, namely Query Relevant Ratio (QRR) and Mean Relevant Document (MRD), to evaluate the effectiveness of recommendations with respect to users' post-click satisfaction. The experimental results show that, by recommending high utility queries, our approach is far more effective in helping users find relevant search results and thus satisfy their information needs.

The main contributions of this paper are summarized as follows:

(1) We propose to recommend queries with respect to utility rather than relevance, and take into account both attractiveness and post-click satisfaction for the utility definition;
(2) We introduce a dynamic Bayesian network (i.e., QUM) which can mine query utility from users' reformulation and click behaviors;
(3) We propose two novel metrics (i.e., QRR and MRD) to evaluate the performance of different methods in recommending high utility query, and conduct empirical studies to demonstrate the effectiveness of our approach.

1.1 Comparison with previous publication

We extend in this article a preliminary study published in Zhu et al. (2012), introducing the following novel contributions, which allow as to give a complete picture on our algorithmic solutions:

(1) We provide more precise definitions on query utility, and correct the potential flaws in the formulation of the query utility model.
(2) We provide details on the maximum likelihood learning of the query utility model for the ease of the reproduction of our approach.
(3) We add extensive experiments to evaluate the recommendation performance according to query difficulty, and to study the impact of the hyper-parameters of the model.
(4) We add a new section to discussion the application of our model in real-world scenarios as well as how to address the new/rare queries in practice.

In the remainder of this paper, Sect. 2 reviews the related work. Section 3 presents our proposed approach and parameter estimation in detail. In Sect. 4, we conducted empirical experiments based on a publicly available data set to evaluate the performance of the proposed approach. Finally, in Sect. 4.7 we conclude and discuss some ideas for the future work.

2 Related work

In this section, we review two research topics which are relevant to our work: query recommendation and click model.

2.1 Query recommendation

Query recommendation has been employed as a key technique by modern industrial search engines. While most early query recommendation methods explore document information, query log data has been widely used in recent work. Query click-through (Beeferman and Berger 2000; Li et al. 2008; Mei et al. 2008), and search sessions (Zhang and Nasraoui 2006; Boldi et al. 2008) are among the most used types of information in query logs.

A major approach on query recommendation is relevant query recommendation, which focuses on providing alternative queries similar to the user's initial query. Collective users' behaviors are usually employed to help find similar queries. For example, Wen et al. (2001) attempted to find similar queries by clustering queries in query logs based on both query content information and user click-through data. Beeferman and Berger (2000) applied agglomerative clustering algorithm over the click-through bipartite graph to identify related queries for recommendation. Li et al. (2008) recommended related queries by computing the similarity between queries based on query-URL vector model and leveraging a hierarchical agglomerative clustering method to rank similar queries. Ma et al. (2008) built two bipartite graphs (i.e., user-query and query-URL bipartite graphs) based on the click-through data and then developed a two-level query recommendation method. Zhang and Nasraoui (2006) first proposed to model users' sequential querying behaviors as a query graph and calculate query similarity based on search sessions for recommendations. Boldi et al. (2008) further introduced the concept the query-flow graph by aggregating the session information of users, and then performed a random walk on this graph to find relevant queries.

A number of studies have been conducted to take into account diversity in query recommendation. Mei et al. (2008) boosted long tail queries to enhance recommendation diversity by using a hitting time approach based on the query-URL bipartite graph. Guo et al. (2010) introduced a structured recommendation approach to recommend diverse queries that satisfy both users' exploratory interests and search interests. Zhu et al. (2011) explicitly addressed the diversity in query recommendation based on an extended manifold ranking method.

Recently, some studies consider the utility[2] in query recommendation (Jain et al. 2011; Anagnostopoulos et al. 2010; Ozertem et al. 2012), and formalize query recommendation as a problem of optimizing a global utility function. However, the utility defined in these works is largely different from ours. For example, In Jain et al. (2011), the utility actually

[2] The word "utility" has usually been used in recommender systems (Wang and Zhang 2011), which attempts to recommend users high marginal utility products.

refers to the diversity of the recommendations. While in Anagnostopoulos et al. (2010), the query utility is defined as the possibility that users will be satisfied by its search results, and simply calculated by its click-through rate. Obviously, this utility definition is independent of users' initial information needs, while in our case it is dependent. Moreover, we will show later that their utility is actually part of our model, which cannot reveal the true utility of the query alone. In Ozertem et al. (2012), the query utility is defined as the difference in discounts of clicked documents between a reformulated query and the original query. Obviously, this utility definition again only captures the click-through information without considering the post-click behaviors. In Zhu et al. (2013), the query utility is defined over the click-through and reformulation behaviors without a specific form as in this paper, and an absorbing random walk based method is employed over the session-flow graph to estimate the query utility for recommendation.

2.2 Click model

Our model is inspired by the click models on modeling sequential user behaviors in Web search, based on the analogy between search result examination/click behaviors and query reformulation/click behaviors. Therefore, we also briefly review the related work on click models.

Click model Chuklin et al. (2015) is proposed to tackle the task of estimating unbiased relevance of documents from query logs in Web search ranking. Granka et al. (2004) first noticed the position bias in their eye-tracking experiment, i.e., a document appearing in higher position is more likely to be clicked than those documents in lower positions. Richardson et al. (2007) introduced a multiplicative factor to increase the relevance of documents in lower positions. After that, Craswell et al. (2008) first formalized this idea as the *examination hypothesis*, which assumes that the user will click a document if and only if it is both examined and relevant. There are a variety of variants Dupret and Piwowarski (2008), Hu et al. (2011), Liu et al. (2014) based on the examination hypothesis but with different derivation of the examination probability. For example, Dupret and Piwowarski (2008) extended the examination hypothesis and proposed the user browsing model (UBM), which assumes that the examination event depends not only on the current position but also on the previous clicked position. Hu et al. (2011) assumed that users' clicks are also affected by intent bias, and proposed to characterize the diversity of search intents. Liu et al. (2014) proposed a two-stage examination model based on they eye-tracking/click-through data, and found that the two stage examination behaviors acan be predicted with mouse movement behavior.

Craswell et al. (2008) further introduced the *cascade hypothesis*, which assumes that users examine search results in strictly sequential order without skips, and the first document is always examined. The cascade model Craswell et al. (2008) combined the examination hypothesis and the cascade hypothesis, and it also constrained that users stop as soon as a relevant document is clicked and abandon the search session. However, this model is too restrictive and cannot deal with the search sessions with multiple clicks. To alleviate this limitation, some variants (Guo et al. 2009a, b; Chapelle and Zhang 2009) were further proposed. Dependent click model (DCM) Guo et al. (2009b) used position-dependent parameters as conditional probabilities of examining the next document after a click and generalized the cascade model to sessions with multiple clicks. Click chain model (CCM) Guo et al. (2009a) assumed that the probability of examining the current position also depends on the relevance at the previous position. Dynamic Bayesian network (DBN) Chapelle and Zhang (2009) distinguished the perceived relevance and actual

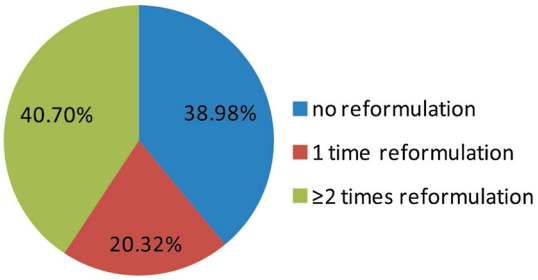

Fig. 1 Distribution of query reformulation on Yandex query logs

relevance, and assumed that a user continues to examine the next document if she is unsatisfied with the clicked document. There are also some other models that do not employ the cascade hypothesis, such as the session utility model (SUM) Dupret and Liao (2010), the pure relevance model (PRM) Srikant et al. (2010), the federated click model (FCM) (Chen et al. 2012; Chuklin et al. 2014), The vertical click model (VCM) Wang et al. (2013), the task-centric click model (TCM) Zhang et al. (2011), and Partially Sequential Click Model (PSCM) Wang et al. (2015).

The most related work to our QUM is the DBN model proposed in Chapelle and Zhang (2009). However, our approach differs from DBN in two significant ways: (1) The problem studied and the data set are completely different. Their goal is to estimate the relevance of URLs for Web search ranking based on the search results and clicks given a query, while we consider the problem of estimating the utility of queries for query recommendation based on the reformulations and clicks given an initial query. (2) The model is also different. In DBN, the satisfaction event only depends on the current state, while in our case, we assume that it is dependent on all previous states.

3 Our approach

In our work, we propose to recommend high utility queries, i.e., queries that can better satisfy users' information needs, to users. The major problem is how to learn the query utility according to users' original information needs. In this section, we first take a look at a typical users' search session to show what reveals query utility, and give the definition of the query utility. We then introduce a novel dynamic Bayesian network, referred to as Query Utility Model (QUM), to inference query utility by simultaneously modeling users' sequential reformulation and click behaviors. Finally, we show how to estimate the parameters in our model and infer the query utility.

3.1 Search session and query utility

Here we investigate search sessions to see what makes a query useful for users according to their original information needs. A search session, or session, is defined as the sequence of user search actions (such as submitting a query, clicking on a search result URL and reformulating a query) within a time limit for a particular search task. A typical search process in a session is illustrated in Fig. 2. Given some information needs, a user submits an initial query to the search engine, and obtains some returned results. Here the documents

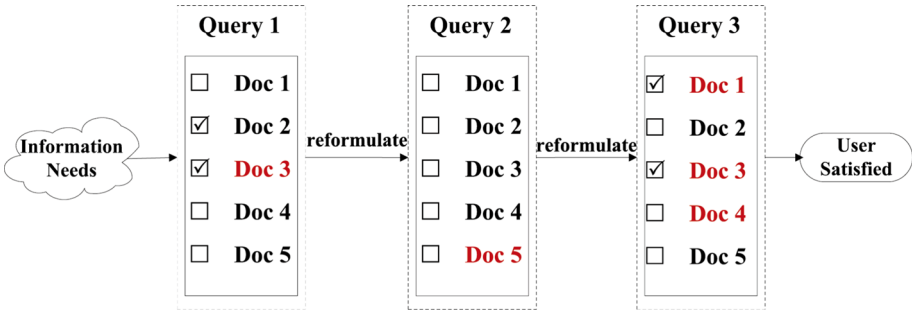

Fig. 2 A typical user search session. Tick '$\surd$' denotes a user click on a document, and *red color* denotes the relevance of a document

which are relevant to the user's information needs are shown in red, otherwise in black. The user then goes through the search results, and clicks the second and third results for further inspecting (i.e., denoted by ticks before the results) as they seem to be relevant. The user obtains some useful information from the third result but is not satisfied yet. Therefore, the user reformulates a second query and submits it to the search engine. However, this time the user finds no interesting results and thus no clicks are performed on the results. Therefore, a third query is generated by the user. From the returned results, the user finds two possibly relevant results, i.e., the first and third results, and clicks them for further investigation. The user then accumulates enough information he/she needs and accomplishes the search task.

From the above search session, we can see that the first and third queries are useful for the user according to the user's original information needs. It shows that the usefulness of a query should satisfy two conditions: (1) The query's search results should be *attractive* to the user, so that he/she would like to click the results for further inspecting, such as the first and third queries. If the search results do not attract the user to click, as in the second query case, the query will not be useful for the user even if it contains relevant search results. (2) The clicked search results can actually *satisfy* the user to some extent by providing relevant information to the user's original needs, e.g., the first and third search results under the third query. Otherwise, the user cannot obtain any useful information after clicking, e.g., the second search result of the first query.

Based on the above analysis, we further divide the utility of a query into the following two components.

Perceived utility, which refers to the attractiveness of the query's search results perceived by the user before investigating the content of these results. The higher the perceived utility a query has, the more likely the user would click the returned results.
Posterior utility, which refers to the satisfaction of the user from the clicked search results given his/her original needs. The higher the posterior utility a query has, the more likely the user would obtain useful information from the clicked results.

The query utility is then defined as the product of the two components, since the first one models the click likelihood while the latter models the post-click satisfaction.

The remaining problem is how to learn the query utility automatically. From the above search session, we can see that the problem would be quite simple if we have observed all the search behaviors from users and the exact relevance labels of the search results.

However, the true relevance labels are not observed in real search logs, which makes such a learning problem non-trivial. Without loss of generality, here we assume that a user will be more likely to click the search results of a query if she deems the results relevant to their information needs. We also assume that the user will acquire some useful information if the clicked results are relevant to their information needs, and will choose to reformulate the next query if she has not been satisfied. Based on these assumptions, we are able to infer queries' utility from the collective search sessions automatically.

3.2 Query utility model

Based on the above analysis, we now introduce a novel dynamic Bayesian network, referred to as Query Utility Model (QUM), to learn query utility based on users' sequential search behaviors. Suppose there are N search sessions starting from the same information needs,[3] and there are a total of T distinct reformulated queries occurring in these N search sessions, denoted by $q_t (1 \leq t \leq T)$.

For a particular search session, we define four binary random variables, R_i, C_i, A_i and S_i to model reformulation, click, attractiveness and satisfaction events at the i-th reformulation position:

- R_i: whether there is a reformulation at position i;
- C_i: whether the user clicks on some of the search results of the reformulation at position i;
- A_i: whether the user is attracted by the search results of the reformulation at position i;
- S_i: whether the user's information needs have been satisfied at position i;

where the first two events are observable from search sessions and the last two events are hidden.

Figure 3 plots the graphical model of QUM for a particular search session. The full model specification that accompanies Fig. 3 is as follows:

$$P(C_i = 1 | R_i = 1, A_i = 1) = 1, \tag{1}$$

$$P(A_i = 1) = \alpha_{\phi(i)}, \tag{2}$$

$$P(S_i = 1 | C_{1:i}) = 2\sigma\left(\sum_{k=1}^{i} \beta_{\phi(k)} \cdot I(C_k = 1)\right) - 1, \tag{3}$$

$$P(R_i = 1 | R_{i-1} = 1, S_{i-1} = 1) = 0. \tag{4}$$

where $\phi(i)$ denotes the index of the query at the position i in a search session.

The above equations describe our model in the following way: The user will reformulate queries sequentially, and stop reformulation if her information needs have been satisfied. There are some clicks on the search results of a reformulated query if and only if the user reformulates the query and is attracted by the search results (1). The probability of the attractiveness only depends on the search results of the reformulated query (2), which is controlled by the variable $\alpha_{\phi(i)} \geq 0$, i.e., the perceived utility of the query at position i. After the user clicks and inspects the search results, there is certain probability she will

[3] In practice, we treat the initial query as a proxy of users' search intent and collect all the search sessions starting from the same initial query to approximate the search behaviors from the same intent.

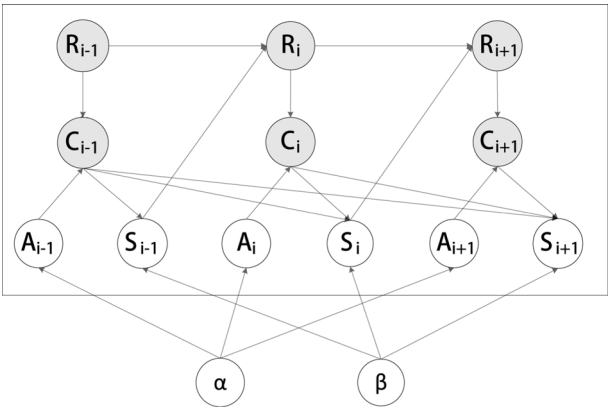

Fig. 3 Graphical model representation of query utility model. Observed click and reformulation variables are *shaded*

be satisfied at the current position. The probability of users' satisfaction at position i depends on how much utilities she has accumulated from the previously clicked search results (3).[4] Note here $C_{1:i}$ denote the vector $(C_1, C_2, \ldots, C_i)$, $\beta_{\phi(i)} \geq 0$ denotes the posterior utility of the query at position i, $I(C_k = 1)$ is an indicator function, and $\sigma(x)$ is the logistic function:

$$\sigma(x) = \frac{1}{1 + e^{-x}}. \tag{5}$$

Once the user is satisfied, she will not reformulate the next query (4).

In our model, there are two parameters α_t and β_t $(1 \leq t \leq T)$ related to the utility of a query. The first one models the perceived utility, as it reflects the attractiveness of a query's search results. The second one models the posterior utility, since it captures the satisfaction of a user from the clicked search results. As aforementioned, we define the utility of a query as the product of the two components

$$u_t := \alpha_t \times \beta_t. \tag{6}$$

Note that in QUM, α_t also denotes the probability that users click the search results of the query q_t (i.e., click likelyhood), and β_t models the post-click satisfaction. Based on the above definition, we can see that u_t actually stands for the expected satisfaction users obtained from the search results of the query according to their original information needs.

So far as we know, this is the first dynamic Bayesian network which attempts to model the query utility from users' sequential search behaviors. To simplify our query utility model, in this work we only consider whether the search results of a reformulated query have some clicks or not, but do not specify which URL has been clicked. We may further extend our utility model to capture the specific clicked URLs for finer modeling. Moreover, we take a search session terminated with a user clicking some results as a successful session, otherwise a failure. It is clear that this assumption may not be realistic since users may also stop searching after clicking some results because of impatience. However, even

[4] We assume the utilities of a set of queries can be addictive. Meanwhile, we define the probability in the form of $2\sigma() - 1$ to ensure the probability range due to the non-negative variables in the logistic function.

under this simple assumption, we find our model can capture the utility of queries very well as shown in the experimental section. In the future work, we may further improve our model by introducing a persistent parameter to capture the impatience.

3.3 Parameter estimation

We use the maximum likelihood estimation to estimate the model parameters $\alpha = \{\alpha_t | 1 \le t \le T\}$ and $\beta = \{\beta_t | 1 \le t \le T\}$. For a specific search session with M reformulations (Note that the initial query is not modeled), the joint probability of all the variables is given by:

$$\begin{aligned} &P(C_{1:M}, R_{1:M}, A_{1:M}, S_{1:M}) \\ &= \prod_{i=1}^{M} P(C_i|A_i, R_i) \cdot P(R_i|R_{i-1}, S_{i-1}) \\ &\cdot P(A_i) \cdot P(S_i|C_{1:i}), \end{aligned} \tag{7}$$

where

$$P(C_i|R_i, A_i) = (R_i \cdot A_i)^{C_i} \cdot (1 - R_i \cdot A_i)^{1-C_i}, \tag{8}$$

$$\begin{aligned} P(R_i|R_{i-1}, S_{i-1}) = &(R_{i-1} \cdot (1 - S_{i-1}))^{R_i} \\ &\cdot (1 - R_{i-1} \cdot (1 - S_{i-1}))^{1-R_i}, \end{aligned} \tag{9}$$

$$P(A_i) = \alpha_{\phi(i)}{}^{A_i} \cdot (1 - \alpha_{\phi(i)})^{1-A_i}, \tag{10}$$

$$\begin{aligned} P(S_i|C_{1:i}) = &\left(2\sigma\left(\sum_{k=1}^{i} \beta_{\phi(k)} \cdot I(C_k = 1)\right) - 1\right)^{S_i} \\ &\cdot \left(2 - 2\sigma\left(\sum_{k=1}^{i} \beta_{\phi(k)} \cdot I(C_k = 1)\right)\right)^{1-S_i}. \end{aligned} \tag{11}$$

Note here R_i is deterministic in the search session that $R_i = 1$ for all $(1 \le i \le M)$. By substituting Eqs. (8)–(11) into the Eq. (7), we obtain the joint probability as follows:

$$\begin{aligned} &P(C_{1:M}, R_{1:M}, A_{1:M}, S_{1:M}) \\ &= \prod_{i=1}^{M} \alpha_{\phi(i)}{}^{A_i} \cdot (1 - \alpha_{\phi(i)})^{1-A_i} \\ &\cdot \left(2\sigma\left(\sum_{k=1}^{i} \beta_{\phi(k)} \cdot I(C_k = 1)\right) - 1\right)^{S_i} \\ &\cdot \left(2 - 2\sigma\left(\sum_{k=1}^{i} \beta_{\phi(k)} \cdot I(C_k = 1)\right)\right)^{1-S_i}. \end{aligned} \tag{12}$$

Therefore, the overall log-likelihood of the N search sessions can be written as:

$$\begin{aligned}\mathcal{L} = &\sum_{j=1}^{N}\sum_{i=1}^{M} A_i^j \cdot \log\left(\alpha_{\phi_j(i)}\right) + \left(1 - A_i^j\right) \cdot \log\left(1 - \alpha_{\phi_j(i)}\right) \\ &+ s_i^j \cdot \log\left(2\sigma\left(\sum_{k=1}^{i} \beta_{\phi_j(k)} \cdot I(C_k^j = 1)\right) - 1\right) \\ &+ \left(1 - S_i^j\right) \cdot \log\left(2 - 2\sigma\left(\sum_{k=1}^{i} \beta_{\phi_j(k)} \cdot I(C_k^j = 1)\right)\right),\end{aligned} \tag{13}$$

where the click events C_i are observed in the search session. If $C_i = 1$, we can infer that the results of the reformulation at position i is attractive (i.e., $A_i = 1$); Otherwise, $A_i = 0$. Besides, based on our assumption, we have that $S_{1:M-1} = 0$ and $S_M = 1$ if the search session ends with a user clicking some results (i.e., a successful search session); Otherwise, we have that $S_{1:M} = 0$.

By maximizing the log-likelihood function in Eq. (13), we can easily obtain the estimation of the perceived utility

$$\begin{aligned}\alpha_t &= \frac{\sum_{j=1}^{N}\sum_{i=1}^{M} A_i^j \cdot I(\phi_j(i) = t)}{\sum_{j=1}^{N}\sum_{i=1}^{M} I(\phi_j(i) = t)} \\ &= \frac{\sum_{j=1}^{N}\sum_{i=1}^{M} I(C_i^j = 1) \cdot I(\phi_j(i) = t)}{\sum_{j=1}^{N}\sum_{i=1}^{M} I(\phi_j(i) = t)}.\end{aligned} \tag{14}$$

Note here $\phi_j(i)$ denotes the index of the query at the position i in the j-th search session. From the result we can see that, the perceived utility $\{\alpha_t\}_{t=1}^{T}$ captures the probability that users click the search results of the query q_t given their original information needs.

The remaining problem is to estimate the posterior utility $\{\beta_t\}_{t=1}^{T}$. We now consider the problem of maximizing $\mathcal{L}$ subject to the inequality constraints of the $\beta_t \geq 0$ $(1 \leq t \leq T)$. The new objective function is:

$$\Lambda = \mathcal{L} + \sum_{t=1}^{m} \lambda_t \beta_t - \mu ||\beta||_2, \tag{15}$$

where λ_t is the Lagrangian coefficient and μ is the regularization parameter.[5] Note here we impose a L2-norm constraint on the objective function to avoid over-fitting.

An optimal solution to this problem should satisfy the Karush–Kuhn–Tucker (KKT) Kuhn and Tucker (1951) optimality conditions:

$$\begin{cases} \beta_t \geq 0, \\ \lambda_t \geq 0, \\ \lambda_t \cdot \beta_t = 0. \end{cases} \tag{16}$$

As in Tognola and Bacher (1999), we convert the inequality constraints $\beta_t \geq 0$ $(1 \leq t \leq T)$ to equality constraints by introducing slack variables z_t $(1 \leq t \leq T)$ which satisfy

[5] In our experiments, we set $\mu = 1$, and we will explain the reason in Sect. 4.

$$\beta_t - z_t = 0, \tag{17}$$

with $z_t \geq 0$. Due to the fact that λ_t and z_t must be positive or zero, we put all the λ_t and z_t in a quadratic form, i.e., λ_t^2 and z_t^2.

Applying these steps to the optimality conditions, we get the following transformed optimality conditions:

$$\begin{cases} \frac{\partial}{\partial \beta_t}(\mathcal{L} + \sum_{t=1}^{m} \lambda_t^2 \beta_t - \mu||\beta||_2) = 0, \\ -\beta_t + z_t^2 = 0, \\ \lambda_t^2 \beta_t = 0. \end{cases} \tag{18}$$

The solution of the above optimization problem is achieved by a Newton–Raphson algorithm. The partial derivatives with respect to all the variables β_t, λ_t and z_t are as follows:

$$\begin{bmatrix} \frac{\partial^2 \mathcal{L}}{\partial \beta_t^2} - 2\mu & 0 & 2\lambda_t \\ -1 & 2z_t & 0 \\ \lambda_t^2 & 0 & 2\lambda_t \beta_t \end{bmatrix} \begin{bmatrix} \Delta\beta_t \\ \Delta z_t \\ \Delta\lambda_t \end{bmatrix} = - \begin{bmatrix} \frac{\partial \mathcal{L}}{\partial \beta_t} - 2\mu\beta_t + \lambda_t^2 \\ -\beta_t + z_t^2 \\ \lambda_t^2 \beta_t \end{bmatrix}. \tag{19}$$

Solving the above linear system of equations (19), we obtain the updating functions:

$$\begin{cases} \Delta\beta_t = \dfrac{-\beta_t \frac{\partial \mathcal{L}}{\partial \beta_t} + 2\mu\beta_t^2}{\beta_t \frac{\partial^2 \mathcal{L}}{\partial \beta_t^2} - 2\mu\beta_t - \lambda_t^2}, \\ \Delta z_t = \dfrac{\beta_t - z_t^2 + \Delta\beta_t}{2z_t}, \\ \Delta\lambda_t = \dfrac{-\lambda_t^2 \beta_t - \lambda_t^2 \Delta\beta_t}{2\lambda_t \beta_t}. \end{cases} \tag{20}$$

3.4 High utility recommendation

Based on the above query utility model, we finally obtain our high utility query recommendation approach. Given the original information needs, typically represented by an input query, we collect all the search sessions starting from the same given query to approximate the search behaviors from that information needs. All the queries in these search sessions other than the input query become the recommendation candidates. We then apply our query utility model and infer the utilities of the recommendation candidates according to the original information needs. Finally, we rank all the candidate recommendations according to their utilities and provide the top ranked ones to users.

"how big are us petroleum reserves"

"where and who exactly did buy louisiana from napoleon"

"what is the largest amount of rain to fall in one hour in the us and where and when did it happen"

"find a story about a person who became bankrupt because they had to pay too much on healthcare answer is url"

"does truth serum really exist"

"what animal is smaller than a bear but it eats a plant called bearberry"

Fig. 4 Some examples of test queries from UFindIt collection

4 Experiments

In this section, we empirically evaluate the effectiveness of our proposed high utility recommendation method. We first introduce the dataset, baseline methods used in comparison, and evaluation metrics. Then we conduct extensive experiments to compare the performance of our approach with state-of-the-art query recommendation methods. Finally, we also study the effect of the parameters on our proposed model.

4.1 Dataset

Our experiments are based on a publicly available dataset, namely UFindIt log data,[6] which was collected over a period of 6 months. In this dataset, there are totally 40 initial queries representing 40 original information needs, which are real queries selected from sites such as *wiki.answers.com* and *Yahoo! Answers*. The selection criteria follow that, the queries can clearly express users' information needs while such needs can not be simply satisfied by directly issuing these test queries to search engines. Some of the examples are shown in Fig. 4. Users are then required to search and find right answers (i.e., relevant results) according to these information needs. All the users' search sessions are logged in this process. For each session, the following details are recorded: the queries, clicked URLs, click positions, browsing trails, and the time-stamps of all the events. The ground-truth (relevance) labels are then annotated on users submitted search results indicating whether this result is a right answer of the original information need.

We process the data by ignoring some interleaved sessions, where the participants search for multiple information needs in one search process. We also remove sessions which have no reformulations, and sessions started without queries. After processing, we obtain 1298 search sessions, 1086 distinct queries and 1555 distinct clicked URLs. For each initial query, the average number of search sessions is 32 and the average number of distinct reformulated queries is 26.

For experiments, we take all the 40 initial queries for testing, and evaluated the quality of top 10 recommendations for each query based on ground-truth labels from the UFindIt

[6] http://ir-ub.mathcs.emory.edu/uFindIt/.

dataset. Note that for each test query, we collect all the search sessions in the log data to infer the parameters in our model for recommendation.

4.2 Baseline methods

To evaluate the performance of our QUM method, we compare it with four baseline query recommendation methods. These baseline methods can be grouped into two categories, namely frequency-based methods and graph-based methods:

Frequency-based methods:

- Adjacency (ADJ): Given a test query q, the top frequent queries in the same session adjacent to q are recommended to users Jones et al. (2006).
- Co-occurrence (CO): Given a test query q, the top frequent queries co-occurred in the same session with q are selected as recommendations Huang et al. (2003).

Graph-based methods:

- Query-Flow Graph (QF): This is a state-of-the-art query recommendation method proposed by Boldi et al. (2008). In QF, a query-flow graph is first constructed based on collective search sessions, and a random walk is then performed on this graph for query recommendation.
- Click-through Graph (CT): This is another state-of-the-art query recommendation method proposed by Mei et al. (2008). In CT, a query-URL bipartite graph is first constructed by mining query logs, a random walk is then performed on this graph and the hitting time is leveraged as a measure to select queries for recommendation.

Besides, to further investigate the influence of the two component utilities (i.e., perceived utility and posterior utility) in our QUM method, we also use them separately to recommend queries as two baselines, namely Perceived Utility method (PCU) and Posterior Utility method (PTU). Note that the PCU method is actually similar to the model proposed in Anagnostopoulos et al. (2010), which defines the query utility by its click-through rate.

4.3 Evaluation metrics

In this paper, we propose to recommend high utility queries to directly toward the goal of query recommendation, i.e. assisting users to reformulate queries so that they can acquire their desired information successfully and quickly. Correspondingly, the evaluation should also reflect this goal, i.e. whether users will be satisfied by the recommendation with respect to their original information needs. Traditional evaluation methods in relevant query recommendation (e.g., precision and recall He et al. (2009)) no longer fit our scenario since they are not directly toward our goal of high utility query recommendation. Traditional evaluation in relevant query recommendation is usually based on relevance labels manually assigned to recommended queries (Cao et al. 2008). However, a query assigned as relevant may not necessarily help users find target results, thus such evaluation cannot evaluate the true effectiveness of query recommendation. This is the particular reason why some larger datasets (e.g., Yandex relevance prediction challenge dataset)[7]

[7] https://academy.yandex.ru/events/data_analysis/relpred2011/.

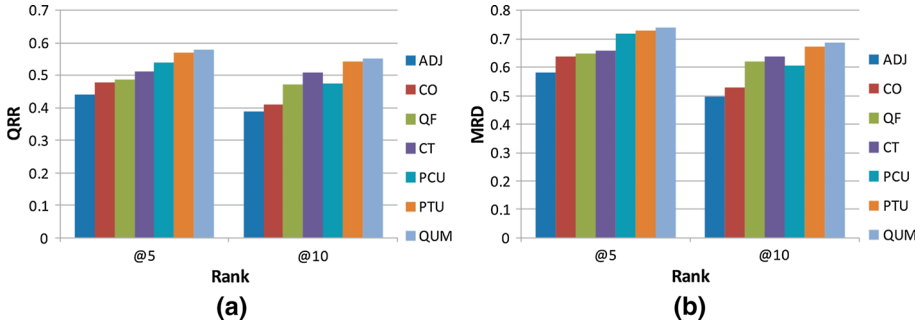

Fig. 5 Comparison of the performance of all approaches (ADJ,CO,QF,CT,PCU,PTU,QUM) in terms of QRR and MRD

cannot be used for the evaluation of utility based recommendation although they also contain reformulations and relevance labels.

Fortunately, the UFindIt log data provides ground-truth (relevance) labels on users' submitted search results indicating whether this result is a right answer of the original information need. We can rely on these labels to evaluate the utility of the recommended queries. Note that none of these labels are used in our training process. Specifically, we define two evaluation metrics, namely the Query Relevant Ratio (QRR) and the Mean Relevant Document (MRD), to measure the quality of the recommendations.

For a specific information need, the metric QRR is defined as:

$$QRR_{q_0}(q) = \frac{RQ_{q_0}(q)}{N_{q_0}(q)}, \tag{21}$$

where $RQ_{q_0}(q)$ denotes the total frequency of query q with relevant results submitted by users with respect to the original search need q_0, and $N_{q_0}(q)$ denotes the total frequency of query q issued by users with respect to the original search need q_0. This metric measures the probability that a user finds relevant results when she uses query q for her search task given the original query q_0. A higher QRR means that a user will be more likely to find useful results with respect to the original information needs.

Moreover, for a specific information need, the metric MRD is defined as:

$$MRD_{q_0}(q) = \frac{RD_{q_0}(q)}{N_{q_0}(q)}, \tag{22}$$

where $RD_{q_0}(q)$ denotes the total frequency of relevant results submitted by users when they use query q for their search tasks with respect to the original search need q_0, and $N_{q_0}(q)$ denotes the total frequency of query q issued by users with respect to the original search need q_0. This metric measures the average number of relevant results a user finds when she uses query q for her search task given the original query q_0. A higher MRD means that a user will find more relevant results according to the original information needs.

4.4 Overall evaluation results

Figure 5a, b show the performance of top recommendations from different methods under the metric QRR and MRD, respectively. From Fig. 5, we can see that the two frequency-based methods ADJ and CO perform poorly under the two metrics. It shows that by simply

considering the most frequently adjacent or co-occurring queries in the same session with the given query (which are usually highly relevant), we cannot guarantee to recommend useful queries to satisfy users' information needs. The two graph-based methods, i.e., QF and CT, show better performance than the frequency-based methods. It indicates that by leveraging the local relationships (i.e., either the co-click or the reformulation relationship) between query pairs to collectively reveal the global relationships between queries, we are able to find better query recommendations.

The PCU method, which only relies on queries' perceived utility for recommendation, presents a comparable performance with the two graph-based methods. As we know, the PCU method actually recommends queries according to the expected click-through rate over their search results (i.e., perceived utility). Since users are more likely to click the search results that they deem relevant, the perceived utility implicitly convey the utility information of the queries. However, only after inspecting the content of the clicked results, users can decide whether the results are truly relevant. Therefore, a useful query may have high perceived utility, but the query with high perceived utility is not necessary to be highly useful. This explains the reason why the PCU method may work well at top 5 positions but perform poorly at top 10 positions as compared with the graph-based methods.

Moreover, the PTU method, which takes queries' posterior utility for recommendation, shows better performance as compared to the above baseline methods under both metrics. It indicates that by simultaneously consider users' reformulation and click behaviors, the learned posterior utility can better reflect the usefulness of the queries. Moreover, when compared with the PCU method, we can find that the PTU method shows constantly better performance. It further demonstrates the importance to take into account users' reformulation behaviors (i.e., satisfaction event) to capture the utility of queries.

Finally, as we can see from Fig. 5, our QUM method performs better than all the baseline recommendation methods. We conduct t-test ($p\text{-}value \leq 0.05$) over the results and find that the performance improvements are significant as compared with both the frequency-based and graph-based baselines. It shows that by leveraging our query utility model, we are able to infer and recommend high utility queries for users. Meanwhile, as compared with the two component utility methods (i.e., PCU and PTU methods), the QUM methods can also constantly outperform the two baselines. It demonstrates that to recommend high utility queries, we need to find those queries that can not only attract users to click their search results, but also provide useful information with the clicked search results. The two aspects are complementary and neither can be ignored for high utility recommendation.

4.5 Evaluation according to query difficulty

To further investigate the performance of our QUM method in high utility query recommendation, we break down the comparison among different approaches according to the query difficulty. According to the UFindIt log data, we separate the test queries into three groups (easy, medium, hard) based on their difficulty.[8] The evaluation results under the metrics QRR and MRD are shown in Table 1, where the percentages in the parentheses are the relative improvements of our QUM method over the corresponding methods.

[8] The difficulty of the queries depends on users' answers in the QA tasks. A query is easy if more than 75 % participants issued correct answers for it, and hard if less than 25 % participants submitted correct answers. The remaining queries are medium.

Table 1 Performance comparison of different methods under different query difficulty levels

Query Difficulty	Method	QRR		MRD	
		@5	@10	@5	@10
Easy	ADJ	0.588 (18.64 %)	0.526 (26.30 %)	0.771 (20.32 %)	0.674 (25.22 %)
	CO	0.609 (14.55 %)	0.529 (25.63 %)	0.830 (11.80 %)	0.687 (22.89 %)
	QF	0.618 (12.94 %)	0.604 (9.89 %)	0.846 (9.67 %)	0.806 (4.69 %)
	CT	0.654 (6.62 %)	0.635 (4.65 %)	0.836 (11.02 %)	0.805 (4.79 %)
	PCU	0.656 (6.37 %)	0.611 (8.74 %)	0.889 (4.35 %)	0.798 (5.79 %)
	PTU	0.689 (1.22 %)	0.663 (0.17 %)	0.908 (2.18 %)	0.837 (0.86 %)
	QUM	0.698	0.664	0.928	0.844
Medium	ADJ	0.460 (30.00 %)	0.429 (33.19 %)	0.596 (24.14 %)	0.527 (33.76 %)
	CO	0.495 (20.81 %)	0.441 (29.65 %)	0.640 (15.72 %)	0.550 (28.10 %)
	QF	0.511 (17.07 %)	0.500 (14.39 %)	0.615 (20.43 %)	0.630 (11.79 %)
	CT	0.534 (12.07 %)	0.549 (4.02 %)	0.689 (7.54 %)	0.692 (1.81 %)
	PCU	0.544 (9.91 %)	0.485 (17.74 %)	0.703 (5.31 %)	0.588 (19.76 %)
	PTU	0.581 (2.87 %)	0.557 (2.70 %)	0.722 (2.53 %)	0.689 (2.18 %)
	QUM	0.598	0.572	0.740	0.704
Hard	ADJ	0.259 (65.27 %)	0.216 (91.19 %)	0.351 (54.37 %)	0.284 (77.27 %)
	CO	0.314 (36.29 %)	0.261 (58.17 %)	0.412 (31.63 %)	0.340 (48.00 %)
	QF	0.324 (32.08 %)	0.312 (32.20 %)	0.441 (22.94 %)	0.414 (21.78 %)
	CT	0.334 (28.08 %)	0.343 (20.17 %)	0.437 (24.15 %)	0.424 (18.85 %)
	PCU	0.404 (5.90 %)	0.324 (27.07 %)	0.534 (1.54 %)	0.413 (22.02 %)
	PTU	0.426 (0.28 %)	0.402 (2.51 %)	0.526 (3.18 %)	0.485 (3.92 %)
	QUM	0.427	0.412	0.542	0.504

The percentages in the parentheses are the improvements of our QUM method over the corresponding methods

From the results we can see that, not surprisingly, the performances of all the recommendation methods decline with the increase of query difficulty. However, our QUM method consistently shows better performance than all the other methods under all the query difficulty levels. Interestingly, we can find that with the increase of the query difficulty, the improvements of our QUM method over other recommendation methods also increase. For example, when the queries are easy, the performances of the frequency-based methods and the graph-based methods under the metric MRD@10 are above 0.6 and 0.8, respectively, while our QUM methods reaches 0.844. It seems that for easy queries, most recommendation methods may work well since popular or relevant recommendations can already help to find the right results.

However, when the queries become hard, the performances of the two frequency-based methods (i.e., ADJ and CO) drops to 0.284 and 0.34 in terms of MRD@10, respectively, and that of the two graph-based methods (i.e., QF and CT) drops to 0.414 and 0.424, respectively. While our QUM method reaches 0.504 under MRD@10. The relative improvements of QUM over the frequency-based methods and the graph-based methods are above 40 % and around 20 %, respectively. It shows that when users' search task is difficult, it would be more beneficial to directly recommend high utility queries beyond relevant queries. Similar results can also be observed under the metrics MRD@5, QRR@5, and QRR@10.

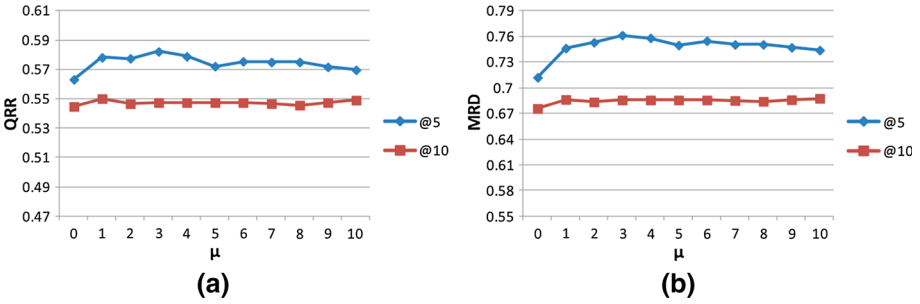

Fig. 6 The impact of parameter μ to the performance of QUM method in terms of QRR and MRD

4.6 Impact of parameter μ

There is one parameter μ in our QUM model, which denotes the weight of the regularization over the posterior utility $\{\beta_t\}_{t=1}^{T}$. In this section, we discuss the impact of the parameter μ on the performance of our QUM method. We varied μ from 0 to 10 with step length 1 in our experiment, and evaluate the performance of our QUM method in terms of QRR and MRD, respectively. The results are depicted in Fig. 6a, b.

From Fig. 6, we observe that our QUM method achieves the lowest performance when $\mu = 0$, while obtains better performance when $\mu > 0$. Moreover, our approach reaches the best performance when μ is set between 1 and 4, while the performance slightly decreases with further increase of the value of μ. This phenomenon illustrates that imposing the L2-norm constraint on the objective function plays a positive role. Generally speaking, the impact of μ to the performance of QUM is relatively stable when μ is small, e.g., between 1 and 4.

4.7 Discussion

In this paper, we conducted experiments and evaluation based on the UFindIt log data due to the following reasons. On one hand, it is a publicly available data set for performing controlled, yet realistic and reproducible studies. On the other hand, the data set conveys the necessary information for evaluating the true effectiveness of query recommendation, i.e. explicit original information needs and the relevance labels on users' clicked search results with respect to their original needs. While other public query logs, such as AOL,[9] cannot directly support such kind of evaluation.

In fact, our approach can be directly applied to real scenarios, e.g. query recommendation in search engines based on real search logs. With richer session information from large scale log data, we may expect to infer the query utility more accurately. However, there may also be some challenging problems when applying the recommendation approach based on real search logs. For example, it is not a trivial task to detect sessions in real query logs. Recently the problem of query session detection has been studied in many literatures (Boldi et al. (2008); He and Göker (2000)), where the using of timeout threshold (e.g. 30 min) for splitting sessions are commonly employed and empirically proved to be effective. However, the detection of sessions is out of the scope of this paper. Meanwhile,

[9] http://www.gregsadetsky.com/aol-data/.

real log data may convey some noise which may affect the performance of recommendation. This can be alleviated by employing some pre-process of data cleaning.

A potential concern on applying the proposed recommendation approach in real-world settings is the availability of the reformulations for a particular query. For example, if a query has never appeared in a search session (e.g., rare/new query), we will not be able to know its reformulations and estimate their utilities in advance. Such rare/new query case is the most difficult problem in practice for most query recommendation methods. A possible solution to this problem is to first find some candidate queries (which have session information) with similar intent to such rare/new query, based on certain similarity metrics (e.g., query string similarity or search result similarity). Then we can rely on the search sessions of these candidate queries to estimate the utility of the possible reformulations, and produce the recommendation for the original rare/new query with respect to both the utility and intent similarity. We will leave this as our future work.

5 Conclusions

In this paper, we argue that we shall recommend high utility queries rather than only relevant queries, to directly achieve the ultimate goal of query recommendation, i.e., to assist users to reformulate queries so that they can acquire their desired information successfully and quickly. We formally define the query utility and identify its two components (i.e., perceived utility and posterior utility). We then propose a novel dynamic Bayesian network QUM to infer query utility from users' search behaviors for recommendation. We evaluate the performance on a real query log, and the experimental results show that the proposed approach can outperform the state-of-the-art baselines in providing useful recommendations.

For the future work, one major aspect is to improve our generative model to better capture query utility. We may further extend our utility model to capture the specific clicked URLs for finer modeling. Besides, we may introduce a persistent parameter into our model to capture the case that the user stops searching because of impatience. As mentioned above, we may also make use of search sessions of similar queries in our QUM to help address the rare/new query recommendation problem. Moreover, it would be interesting to take into account diversity in our model to reduce the redundancy in a set of recommended high utility queries.

Acknowledgments This research work was funded by National Basic Research Program of China (973 Program) of China under Grant Nos. 2014CB340401, National Key Research and Development Program of China under Grant No. 2016YFB1000902, and National Natural Science Foundation of China under Grant Nos. 61472401, 61425016, 61203298.

References

Anagnostopoulos, A., Becchetti, L., Castillo, C., & Gionis, A. (2010). An optimization framework for query recommendation. In *Proceedings of the Third ACM International Conference on Web search and Data Mining* (pp. 161–170).

Beeferman, D., & Berger, A. (2000). Agglomerative clustering of a search engine query log. In *Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 407–416).

Boldi, P., Bonchi, F., Castillo, C., Donato, D., Gionis, A., & Vigna, S. (2008). The query-flow graph: model and applications. In *Proceedings of the 17th ACM Conference on Information and Knowledge Management* (pp. 609–618).

Cao, H., Jiang, D., Pei, J., He, Q., Liao, Z., Chen, E., & Li, H. (2008). Context-aware query suggestion by mining click-through and session data. In *Proceeding of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 875–883).

Chapelle, O., & Zhang, Y. (2009). A dynamic bayesian network click model for web search ranking. In *Proceedings of the 18th International Conference on World Wide Web* (pp. 1–10).

Chen, D., Chen, W., Wang, H., Chen, Z., & Yang, Q. (2012). Beyond ten blue links: enabling user click modeling in federated web search. In *Proceedings of the Fifth ACM International Conference on Web Search and Data Mining, ACM* (pp. 463–472).

Chuklin, A., Zhou, K., Schuth, A., Sietsma, F., & De Rijke, M. (2014). Evaluating intuitiveness of vertical-aware click models. In *Proceedings of the 37th International ACM SIGIR Conference on Research & Development in Information Retrieval, ACM* (pp. 1075–1078).

Chuklin, A., Markov, I., & Rijke, Md. (2015). Click models for web search. *Synthesis Lectures on Information Concepts, Retrieval, and Services*, *7*(3), 1–115.

Craswell, N., Zoeter, O., Taylor, M., & Ramsey, B. (2008). An experimental comparison of click position-bias models. In *Proceedings of the International Conference on Web Search and Web Data Mining* (pp. 87–94).

Dupret, G., & Liao, C. (2010). A model to estimate intrinsic document relevance from the clickthrough logs of a web search engine. In *Proceedings of the Third ACM International Conference on Web Search and Data Mining* (pp. 181–190).

Dupret, GE., & Piwowarski, B. (2008). A user browsing model to predict search engine click data from past observations. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (pp. 331–338).

Granka, LA., Joachims, T., & Gay, G. (2004). Eye-tracking analysis of user behavior in www search. In *Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (pp. 478–479).

Guo, F., Liu, C., Kannan, A., Minka, T., Taylor, M., Wang, YM., & Faloutsos, C. (2009a). Click chain model in web search. In *Proceedings of the 18th International Conference on World Wide Web* (pp. 11–20).

Guo, F., Liu, C., & Wang, YM. (2009b). Efficient multiple-click models in web search. In *Proceedings of the Second ACM International Conference on Web Search and Data Mining* (pp. 124–131).

Guo, J., Cheng, X., Xu, G., & Shen, H. (2010). A structured approach to query recommendation with social annotation data. In *Proceedings of the 19th ACM International Conference on Information and Knowledge Management* (pp. 619–628).

He, D., & Göker, A. (2000). Detecting session boundaries from web user logs. In *Proceedings of the BCS-IRSG 22nd Annual Colloquium on Information Retrieval Research* (pp. 57–66).

He, Q., Jiang, D., Liao, Z., Hoi, SC., Chang, K., Lim, EP., & Li, H. (2009). Web query recommendation via sequential query prediction. In *2009 IEEE 25th International Conference on Data Engineering* (pp. 1443–1454).

Hu, B., Zhang, Y., Chen, W., Wang, G., & Yang, Q. (2011). Characterizing search intent diversity into click models. In *Proceedings of the 20th International Conference on World Wide Web* (pp. 17–26).

Huang, C. K., Chien, L. F., & Oyang, Y. J. (2003). Relevant term suggestion in interactive web search based on contextual information in query session logs. *Journal of the American Society for Information Science and Technology*, *54*(7), 638–649.

Jain, A., Ozertem, U., & Velipasaoglu, E. (2011). Synthesizing high utility suggestions for rare web search queries. In *Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information* (pp. 805–814).

Jones, R., Rey, B., Madani, O., & Greiner, W. (2006). Generating query substitutions. In *Proceedings of the 15th International Conference on World Wide Web* (pp. 387–396).

Kuhn, HW., & Tucker, AW. (1951). Nonlinear programming. In *Proceedings of the Second Berkeley Symposium on Mathematical Statistics and Probability* (pp. 481–492).

Li, L., Yang, Z., Liu, L., & Kitsuregawa, M. (2008). Query-url bipartite based approach to personalized query recommendation. In *Proceedings of the 23rd National Conference on Artificial Intelligence* (pp. 1189–1194).

Liu, Y., Wang, C., Zhou, K., Nie, J., Zhang, M., & Ma, S. (2014). From skimming to reading: A two-stage examination model for web search. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management, ACM* (pp. 849–858).

Ma, H., Yang, H., King, I., & Lyu, MR. (2008). Learning latent semantic relations from clickthrough data for query suggestion. In *Proceeding of the 17th ACM Conference on Information and Knowledge Management* (pp. 709–718).

Mei, Q., Zhou, D., & Church, K. (2008). Query suggestion using hitting time. In *Proceeding of the 17th ACM Conference on Information and Knowledge Management* (pp. 469–477).

Ozertem, U., Chapelle, O., Donmez, P., & Velipasaoglu, E. (2012). Learning to suggest: a machine learning framework for ranking query suggestions. In *Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval, ACM* (pp. 25–34).

Richardson, M., Dominowska, E., & Ragno, R. (2007). Predicting clicks: estimating the click-through rate for new ads. In *Proceedings of the 16th International Conference on World Wide Web* (pp. 521–530).

Srikant, R., Basu, S., Wang, N., & Pregibon, D. (2010). User browsing models: relevance versus examination. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 223–232)

Tognola, G., & Bacher, R. (1999). Unlimited point algorithm for opf problems. *IEEE Transactions on Power Systems*, *14*(3), 1046–1052.

Wang, C., Liu, Y., Zhang, M., Ma, S., Zheng, M., Qian, J., & Zhang, K. (2013). Incorporating vertical results into search click models. In *Proceedings of the 36th International ACM SIGIR Conference on Research and Development in Information Retrieval, ACM*, (pp. 503–512).

Wang, C., Liu, Y., Wang, M., Zhou, K., Nie, Jy., & Ma, S. (2015). Incorporating non-sequential behavior into click models. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval, ACM* (pp. 283–292)

Wang, J., & Zhang, Y. (2011). Utilizing marginal net utility for recommendation in e-commerce. In *Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information* (pp. 1003–1012).

Wen, JR., Nie, JY., & Zhang, HJ. (2001). Clustering user queries of a search engine. In *Proceedings of the 10th International Conference on World Wide Web* (pp. 162–168).

Zhang, Y., Chen, W., Wang, D., & Yang, Q. (2011). User-click modeling for understanding and predicting search-behavior. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 1388–1396).

Zhang, Z., & Nasraoui, O. (2006). Mining search engine query logs for query recommendation. In *Proceedings of the 15th International Conference on World Wide Web* (pp. 1039–1040).

Zhu, X., Guo, J., Cheng, X., Du, P., & Shen, HW. (2011). A unified framework for recommending diverse and relevant queries. In *Proceedings of the 20th International Conference on World Wide Web* (pp. 37–46).

Zhu, X., Guo, J., Cheng, X., & Lan, Y. (2012). More than relevance: high utility query recommendation by mining users' search behaviors. In *Proceedings of the 21st ACM International Conference on Information and Knowledge Management, ACM* (pp. 1814–1818).

Zhu, X., Guo, J., Cheng, X., Lan, Y., & Nejdl, W. (2013). Recommending high utility query via session-flow graph. In *European Conference on Information Retrieval* (pp. 642–655). Springer.