Diversifying Query Auto-Completion

FEI CAI, National University of Defense Technology and University of Amsterdam
RIDHO REINANDA and MAARTEN DE RIJKE, University of Amsterdam

Query auto-completion assists web search users in formulating queries with a few keystrokes, helping them to avoid spelling mistakes and to produce clear query expressions, and so on. Previous work on query auto-completion mainly centers around returning a list of completions to users, aiming to push queries that are most likely intended by the user to the top positions but ignoring the redundancy among the query candidates in the list. Thus, semantically related queries matching the input prefix are often returned together. This may push valuable suggestions out of the list, given that only a limited number of candidates can be shown to the user, which may result in a less than optimal search experience.

In this article, we consider the task of diversifying query auto-completion, which aims to return the correct query completions early in a ranked list of candidate completions and at the same time reduce the redundancy among query auto-completion candidates. We develop a greedy query selection approach that predicts query completions based on the current search popularity of candidate completions and on the aspects of previous queries in the same search session. The popularity of completion candidates at query time can be directly aggregated from query logs. However, query aspects are implicitly expressed by previous clicked documents in the search context. To determine the query aspect, we categorize clicked documents of a query using a hierarchy based on the open directory project. Bayesian probabilistic matrix factorization is applied to derive the distribution of queries over all aspects. We quantify the improvement of our greedy query selection model against a state-of-the-art baseline using two large-scale, real-world query logs and show that it beats the baseline in terms of well-known metrics used in query auto-completion and diversification. In addition, we conduct a side-by-side experiment to verify the effectiveness of our proposal.

Categories and Subject Descriptors: H.3 [**Information Storage and Retrieval**]: Information Search and Retrieval

General Terms: Design, Algorithms, Performance

Additional Key Words and Phrases: Query auto-completion, diversification, query suggestion, web search

This research was partially supported by the Innovation Foundation of NUDT for Postgraduate under No. B130503; Amsterdam Data Science; the Bloomberg Research Grant program; the Dutch national program COMMIT; Elsevier; the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement no. 312827 (VOX-Pol); the ESF Research Network Program ELIAS; the Royal Dutch Academy of Sciences (KNAW) under the Elite Network Shifts project; the Microsoft Research Ph.D. program; the Netherlands eScience Center under project no. 027.012.105; the Netherlands Institute for Sound and Vision; the Netherlands Organisation for Scientific Research (NWO) under project nos. 727.011.005, 612.001.116, HOR-11-10, 640.006.013, 612.066.930, CI-14-25, SH-322-15, 652.002.001, and 612.001.551; the Yahoo Faculty Research and Engagement Program; and Yandex. All content represents the opinion of the authors, which is not necessarily shared or endorsed by their respective employers and/or sponsors.
Authors' addresses: F. Cai (corresponding author), Science and Technology on Information Systems Engineering Laboratory, National University of Defense Technology, Changsha, China, and Informatics Institute, University of Amsterdam, Amsterdam, The Netherlands; R. Reinanda and M. de Rijke, Informatics Institute, University of Amsterdam, Amsterdam, The Netherlands; emails: {f.cai, r.reinanda, derijke}@uva.nl.

DOI: http://dx.doi.org/10.1145/2910579

ACM Reference Format:
Fei Cai, Ridho Reinanda, and Maarten de Rijke. 2016. Diversifying query auto-completion. ACM Trans. Inf. Syst. 34, 4, Article 25 (June 2016), 33 pages.
DOI: http://dx.doi.org/10.1145/2910579

1. INTRODUCTION

Query auto-completion (QAC) is a prominent feature provided by many modern search engines. The goal of QAC is to accurately predict the user's intended query with only a few keystrokes (i.e., typed prefix), helping the user formulate a satisfactory query with less effort and avoid misspelling it as well. In the absence of any knowledge of the user's context or preferences, the standard QAC approach is to return a ranked list of query candidates that strictly match the prefix by their popularity (i.e., frequency) [Bar-Yossef and Kraus 2011; Shokouhi and Radinsky 2012; Shokouhi 2013; Whiting and Jose 2014; Cai et al. 2014b], ignoring similarity between the candidates in this list and thus resulting in possibly *redundant* lists of completions. The length of the list of returned QAC candidates is often limited; if too many redundant queries are returned, user satisfaction may be negatively impacted, especially for prefixes with a large number of (possibly redundant) QAC candidates. Therefore, without further information to disambiguate the user's query aspect, the search engine needs to focus on how best to produce a list of *relevant* and *diversified* QAC candidates that can cover different interpretations.

Intuitively, a sensible QAC approach should maximize the satisfaction of the population of users typing the prefix, which involves a trade-off between presenting more possible QAC candidates of the "correct" query aspect and having diverse QAC candidates in the top positions of the list of QAC candidates for a given prefix, that is, returning the most probable queries early and removing redundant query candidates as well. By doing so, the chance that any user typing the same prefix will find at least one satisfactory query candidate for one's particular information need is maximized. Thus, it is important to capture the user's query aspect and reduce the redundancy of query completions.

Let us illustrate the meaning of query completion redundancy, which refers to the situation in which some auto-completed queries are of equivalent meaning to preceding queries in the list of QAC candidates, describing almost the same query aspect. Table I contains a search session from the well-known AOL query log [Pass et al. 2006]. For the sake of the example, let us assume that we have not yet seen the last query ("*sony*" in Row 6) and that it is, in fact, the initial segment (prefix) "*so*" of this query for which we want to recommend query completions. A regular baseline [Bar-Yossef and Kraus 2011] based on the most popular query is likely to rank the QAC candidates as a list shown in Row 8. But if we look in the list (e.g., QAC candidates at rank 1, 6, and 9), we see that the queries "*southwest airlines*," "*southwestairlines*," and "*southwest airline*" are all returned. Clearly, these three candidates are semantically closely related to each other and probably express an identical query aspect, which can be easily confirmed by considering the overlap of the search engine result pages (SERPs) produced for these three queries. We hypothesize that, in this example, the latter two candidates—"*southwestairlines*" and "*southwest airline*"—in this list of QAC candidates (Row 8) are redundant completions for the prefix "*so*." Thus, to improve the user's search satisfaction, they should be removed from the list, especially when only a few QAC candidates can be returned. By doing so, the QAC performance in this case, typically measured by Mean Reciprocal Rank (MRR), can be improved given that the final submitted query is "*sony*." Moreover, in this case, if the final query submission is "*social security administration*," the MRR score obviously can be further increased as

Table I. A Session Example from the AOL Dataset Consisting of Five Queries (Rows 2–6), and a Ranked List of Top Ten QAC Candidates (Row 8), Separated by ";" and Returned by MPC [Bar-Yossef and Kraus 2011] after Typing the Prefix "*so*" of the Last Query "*Sony*"

1	SessionID	UserID	Query	Time
2	419	1020	compaq	20060315, 14:18:42
3	419	1020	hewlit packard	20060315, 14:26:58
4	419	1020	toshiba	20060315, 14:32:31
5	419	1020	averatec	20060315, 14:35:39
6	419	1020	sony	20060315, 14:38:15
7	A ranked list of query completions for the prefix "so"			
8	southwest airlines; southwest; song lyrics; social security; sopranos; southwestairlines; sony; sofia laiti; southwest airline; social security administration;			

more redundant candidates before "*social security administration*" in the list of QAC candidates are cleared.

After removing redundant queries from the list of QAC candidates, more QAC candidates can be included, which increases the probability of matching the intended query of the user. Thus, in this article, we consider the task of *diversifying query auto-completion* (D-QAC), which aims to return the correct query early in the QAC ranking list and to reduce the redundancy among the QAC candidates. After formally describing the D-QAC problem, we develop a greedy query selection (GQS) approach for D-QAC. Rather than using an algorithm that aims to find a global optimum given a prefix, we implement a greedy algorithm to address the D-QAC task because it follows the problem-solving heuristic of making the locally optimal choice at each stage, which is consistent with the purpose of D-QAC, that is, to maximize the diversity of the reranked list of query completions. In addition, compared to greedy algorithms, a globally optimal algorithm is usually more time-consuming [Cormen et al. 2009]. GQS considers the query aspects that are implicit in query candidates that are popular at query time and in a user's previous search behavior. We capture the former by exploiting previous query frequency before query time and the latter by using a user's previous queries and the documents that they clicked in the current session. We identify a query's aspects by categorizing its clicked URLs using the Open Directory Project (ODP) taxonomy,[1] a topical hierarchy structure for URL categorization.

In practice, our GQS model faces two main challenges. One relates to a query cold-start problem. When ranking the query candidates in the testing phase, we may not know any aspect information about a query candidate from the logs in the training period. The other problem is a sparseness problem: we have only limited aspect information about every query. For the query cold-start problem, we propose a solution by which an unlabelled query can be assigned the same aspects as its most semantically related query that has been labelled by the ODP in the training period. For the sparseness problem, we apply a Bayesian probabilistic matrix factorization approach to derive a distribution of a query over all aspects.

We evaluate our GQS model using two large-scale, real-world query logs and show that it outperforms a competitive state-of-the-art baseline in terms of well-known metrics used in QAC and diversification, for example, MRR and α-nDCG, respectively. We also conduct a side-by-side comparison to assess the diversity of QAC suggestions. Our contributions in this article can be summarized as follows:

[1]http://www.dmoz.org.

(1) We propose the task of diversifying query auto-completion (D-QAC), which aims to return the user's intended query early in a list of QAC candidates and, simultaneously, to reduce the redundancy of the QAC list. To the best of our knowledge, there is no published work on D-QAC.
(2) We propose a greedy query selection (GQS) approach for D-QAC that captures the query aspect not only from the current search popularity, but also from the search context in the session.
(3) We study a query cold-start problem, for which we do not have any aspect information about a query from the logs in the training period. For such unknown queries, we assign aspect labels from its semantically most closely related query for which aspect information has been found during the training period.
(4) We analyze the effectiveness of our GQS model and find that it significantly outperforms state-of-the-art baselines for D-QAC in terms of MRR and α-nDCG. Generally, against the best baseline, GQS achieves an improvement of around 2.3% and 5.6% in terms of MRR and α-nDCG, respectively.

We describe related work in Section 2. The D-QAC problem and our proposed solution are described in Section 3. Section 4 presents our experimental setup. In Section 5, we report our results. We present our conclusions in Section 6, in which we also suggest future research directions.

2. RELATED WORK

In this section, we briefly discuss recent work that is related to our D-QAC task. To the best of our knowledge, there is no published work that focuses on D-QAC. Thus, we first give a brief background on QAC in Section 2.1, which is followed by summarizing diversifying web search results in Section 2.2, and a discussion of diversifying query suggestions in Section 2.3.

2.1. Query Auto-completion

Since the first QAC application in web search was launched by Google in 2004,[2] the feature has also been known as "type-ahead" or "auto-complete." QAC has been well studied and has been successfully embedded as a prominent feature into common search engines.

The *Most Popular Completion* (MPC) model [Bar-Yossef and Kraus 2011] is one of the most effective QAC approaches. It ranks QAC candidates according to their popularity:

$$MPC(p) = \underset{q \in C(p)}{\operatorname{argmax}} \frac{f(q)}{\sum_{q_i \in Q_{Log}} f(q_i)}, \tag{1}$$

where the function $f(\cdot)$ returns the frequency of an input query, for example, q and q_i, in query log Q_{Log}; $C(p)$ is a pool of QAC candidates matching the prefix p. Essentially, the MPC model assumes that the current query popularity distribution is the same as in the past and, consequently, sorts QAC candidates by their past popularity in order to maximize the QAC effectiveness for all users, on average.

However, temporal information, for example, recency and seasonality, is not taken into account in the original MPC model even if the query popularity is changing over time. To tackle this problem, Shokouhi and Radinsky [2012] propose ranking QAC candidates by predicted query frequency based on time series analysis, for which queries recurring within specific temporal intervals, for example, day/night and workday/

[2]http://googleblog.blogspot.nl/2004/12/ive-got-suggestion.html.

weekend, are modeled differently. In an attempt to exploit both recent trends concerning query popularity and past query popularity, Whiting and Jose [2014] propose several QAC ranking approaches by collecting query popularity evidence from a period of recent 2 to 28 days or from a recent query chunk with a fixed size for a given prefix. The cyclic behavior of query popularity can be determined with high accuracy according to its historical frequency distribution [Shokouhi 2011]. Accordingly, cyclic behavior together with recent trends in query popularity have been introduced to predict a query's future frequency. Based on this, better ranked lists of QAC candidates can be produced [Cai et al. 2014b, 2016a]. Forecasts obtained from time-series modeling are reliable, but selecting the optimal time period remains a challenge. Furthermore, Cai and de Rijke [2016a] propose a learning-based QAC approach in which features derived from similar queries and semantically related terms are taken into account.

Aside from temporal information, personalization, in the form of search context or personal preference, is another aspect that has been studied in the setting of QAC. For instance, Bar-Yossef and Kraus [2011] developed a hybrid QAC model in which the QAC candidates are ranked by a final score that is produced by linearly combining the MPC score and the similarity score between candidates and the search context extracted from a user's recent queries. Guo et al. [2011] propose a two-step approach to rank QAC candidates by learning the user's context as well, that is, the user's search history, to match against pregenerated topic models. Similarly, Liao et al. [2011] capture a user's recent queries for matching against the query clusters learned from the clickthrough graph for ranking QAC candidates. Shokouhi [2013] exploits a user's personal profile, in which a user's age, gender, location, and previous queries are utilized to generate user-based features for learning to personalize QAC; this model tries to find similar users who share common search activities with the current user, to rerank the completion candidates. In addition, Cai and de Rijke [2016c] investigate when to personalize QAC by capturing the signals from the typed query prefix, clicked documents, and preceding queries in the same session.

More recently, another factor, the user's interaction behavior, has been studied for QAC. For instance, Mitra et al. [2014] investigate the relationship between user interaction patterns and QAC engagement, and find that users are most likely to engage in QAC at word boundaries. Li et al. [2014] propose a two-dimensional click model for modeling the QAC process after observing the existence of horizontal skipping bias and vertical position bias in the QAC process. Jiang et al. [2014] exploit the search context to learn user reformulation behavior and apply a supervised learning approach for QAC, where term-, query- and session-level features of user reformulation behavior are developed. Hofmann et al. [2014] conduct an in-depth study of user interactions with QAC in web search using eye-tracking and client-side logging, through which they identify a strong position bias towards examining and using top-ranked query completions. Mitra [2015] learns distributed representations of query reformations using deep neural network models, through which session context is modeled for QAC tasks. Li et al. [2015] pay attention to users' sequential interactions with a QAC system in and across sessions, and consequently present a probabilistic model to address the QAC task by capturing the relationship between users' behaviors at different keystrokes in high-resolution QAC logs. Zhang et al. [2015] study implicit negative feedback during user-QAC interactions and propose a novel adaptive model that adapts query auto-completion to users' implicit negative feedback towards unselected query candidates. We refer the reader to Cai and de Rijke [2016b] for a recent survey of the state-of-the-art in QAC.

Despite many advances, previous QAC approaches focus only on ranking the query candidates, pushing the most promising queries to the top positions in the list of candidates but ignoring the redundancy of these candidates, which is likely to hurt

the user's search satisfaction as users may fail to find an acceptable query completion, especially when the number of QAC candidates that can be returned to users is very limited. In contrast, D-QAC considers both relevance and diversity.

2.2. Diversifying Web Search Results

There has been a significant amount of research on search result diversification that is aimed at tackling query ambiguity. In one of the early influential publications on web search result diversification, Maximal Marginal Relevance (MMR) [Carbonell and Goldstein 1998] is introduced. Here, a trade-off between relevance and diversity of search results to a query is determined by reducing redundancy while maintaining relevance when selecting a document (or URL).

A popular strategy for search result diversification is to maximally cover topics, concepts, or aspects of returned documents. Zhai et al. [2003] combine novelty and relevance for subtopic retrieval, in which they are concerned with finding documents that cover many different subtopics of a query topic. Santos et al. [2011] introduce an intent-aware approach for search result diversification and propose diversification of the retrieved results for a given query by learning the appropriateness of retrieval models targeted to each of these intents. Santos et al. [2010a] propose the eXplicit Query Aspect Diversification (xQuAD) framework, which explicitly models the aspects underlying an initial query in the form of subqueries. The authors directly estimate the relevance and novelty of the retrieved documents at the level of multiple subqueries. Vargas et al. [2012] improve upon intent-oriented diversification schemes through the introduction of an explicit relevance model. They suggest that a relevance-based foundation may be better suited for the description of diversification processes and their underlying principles.

Personalized search result diversification enhances the search experience by considering both personal preference and plain diversification. For instance, Radlinski and Dumais [2006] rerank the top results for a given query by introducing diversity into those results, using past query reformulations learned from a user's query logs. They conclude that using diversification is a promising method to improve personalized reranking of search results. Vallet and Castells [2012] combine personalization and diversification components effectively and develop a generalized framework for personalized diversification, enhancing each of them by introducing the user as an explicit random variable in state-of-the-art diversification methods. Liang et al. [2014b] set up a structured learning framework for supervised personalized diversification, in which they extract features directly from documents and from a user-interest latent Dirichlet topic model, resulting in better performance than unsupervised diversification algorithms.

Other well-known diversification approaches, including probabilistic models [Chen and Karger 2006], fusion-based methods [Liang et al. 2014a], query-dependent methods [Santos et al. 2010b], text-level methods [Bache et al. 2013; Dang and Croft 2013], and PM-2 [Dang and Croft 2012], will not be described in detail here. However, so far, all listed work is centered around diversifying web-search results rather than diversifying QACs. To the best of our knowledge, we are the first to address the task of diversifying QACs.

2.3. Diversifying Query Suggestions

Compared to QAC and diversifying web-search results, there is limited work on diversifying query suggestions (DQS). The task, however, is close to our task in this article, that is, diversifying QAC.

First, Ma et al. [2010] propose a unified approach to suggest both semantically relevant and diverse queries to Web users, which leverages a Markov random walk model

on the query-URL bipartite graph. Their method can effectively prevent semantically redundant queries from receiving a high rank. However, the sparseness problem of the query-URL bipartite graph is challenging. To alleviate this problem, Song et al. [2011] rerank the query suggestions by maximizing the diversity of the original search results from suggested queries. They find that their postranking framework significantly improves the relevance of suggested queries.

Instead of using query graphs, Li et al. [2012] pick up recommended queries to avoid redundancy at the level of query concept. They propose a probabilistic model that allows query recommendations to be determined much more efficiently to satisfy real-time requirements. To avoid query redundancy, Santos et al. [2013] try to locate related queries from query logs, which relate to common clicks or common sessions for diversifying suggestions. Kharitonov et al. [2013] introduce a contextualization framework that utilizes a short-term context—for example, the previous queries, the examined documents, and the candidate query suggestions that the user has discarded—to contextualize and diversify the ranking of query suggestions by modeling the user's information need as a mixture of intent-specific user models.

Kim and Croft [2014] propose a framework for diversifying query suggestions to assist domain-specific searchers; this framework identifies diverse query aspects, then suggests both effective and diverse queries based on the identified aspects. Jiang et al. [2009] propose a new query-suggestion paradigm with both personalization and diversity awareness. They first identify the most relevant query suggestion by a multibipartite representation mined from the query log, then explore the latent structure of the multibipartite representation, and identify the remaining suggestion candidates with diversity awareness.

DQS and D-QAC differ in their user input and in the query candidates to be ranked. The combination of search intent expressed by current search popularity and previously submitted queries in a session has not been considered for DQS. In this article, this is one of the approaches that study for D-QAC tasks.

3. DIVERSIFYING QUERY AUTO-COMPLETION

We formally introduce the problem of D-QAC in Section 3.1, describe our greedy query selection model to deal with D-QAC in Section 3.2, and derive query distributions over aspects in Section 3.3.

3.1. The D-QAC Problem

Before introducing our method for D-QAC, we first recall the main notation used in this article in Table II, then state the problem of D-QAC. As the search engine returns only the top N QAC candidates to its users, the objective of D-QAC is to maximize the probability that the average user finds at least one useful QAC candidate within the top N candidates.

Assume that the following are given:

—a prefix p of the last query q_T in a search session consisting of T queries $\{q_1, q_2, \ldots, q_T\}$;
—an initial list of QAC candidates R_I produced for this prefix p with length $|R_I| = k_I$;
—the probability of relevance $P(Rel \mid a, p, C_S)$ of query aspect a for prefix p given the search context C_S consisting of a sequence of preceding queries before q_T, that is, $\{q_1, q_2, \ldots, q_{T-1}\}$;
—and a satisfaction value $P_s(Rel \mid q_c, p, a, C_S)$, that is, a probability of QAC candidate q_c matching the query aspect a given the search context C_S.

First, we start with a simplified objective of the D-QAC problem, which aims to satisfy the average user who enters the prefix p by finding at least one acceptable QAC

Table II. Main Notation used in the Article

Notation	Description
T	number of queries in a session
q_t	the tth ($t = 1, 2, \ldots, T$) query in a session
p	prefix of the last query q_T in a session
R_R	a list of QAC candidates for prefix p returned to the user
R_I	an initial QAC ranking list matching prefix p
k_I	number of QAC candidates in R_I, that is, $\|R_I\|$
a	aspect
$q(i)$	the probability of relevance of query q to the ith aspect
C_S	search context, that is, sequence of queries preceding q_T: $\{q_1, q_2, \ldots, q_{T-1}\}$
q_c	query candidate in a QAC list
λ	trade-off between search popularity and previous search context
$f(q)$	frequency of query q
θ	decay factor
q_s	selected query in R_R
ω_t	normalized decay brought by temporal interval
$TD(q_t)$	time interval between q_t and q_T
N	number of QAC candidates finally returned to a user, that is, a cutoff
$\mathbb{Q}$	set of unique queries
$\mathbb{Q}_L$	set of labelled queries $\subseteq \mathbb{Q}$
$\mathbb{A}$	set of unique aspects
N_q	number of unique queries, that is, $\|\mathbb{Q}\|$
M_a	number of unique aspects, that is, $\|\mathbb{A}\|$
k_f	number of latent features used in BPMF

candidate among the top N QAC candidates returned, given the user's search context C_S, where $R_R \subseteq R_I$ with $|R_R| = N$, such that $N \leq k_I$. This is achieved by maximizing

$$P(R_R \mid p, C_S) = P(Rel \mid p, C_S) \left(1 - \prod_{q_c \in R_R} (1 - P_s(Rel \mid q_c, p, C_S))\right). \tag{2}$$

Let us illustrate this objective and see how it formalizes our intuition. Note that $P_s(Rel \mid q_c, p, C_S)$ can be interpreted as the probability that a QAC candidate q_c satisfies a user who enters the prefix p given the search context C_S. Then, $(1 - P_s(Rel \mid q_c, p, C_S))$ indicates the probability that q_c fails to satisfy the user. Therefore, the probability that all selected QAC candidates fail to satisfy the user equals its product under the query independence assumption. One minus that product is the probability that at least one QAC candidate satisfies the user. Finally, the score, weighted by $P(Rel \mid p, C_S)$, denotes the probability that the set of query candidates R_R satisfies the average user. Then, we break this objective in Equation (2) down to the aspect level as follows:

$$P(R_R \mid p, C_S) = \sum_a P(Rel \mid a, p, C_S) \left(1 - \prod_{q_c \in R_R} (1 - P_v(Rel \mid q_c, p, a, C_S))\right), \tag{3}$$

where a is a given aspect and summing over all aspects weighted by $P(Rel \mid a, p, C_S)$ denotes the probability that the set of query candidates R_R satisfies the average user who enters prefix p at the aspect level.

The objective of the D-QAC framework indicated in Equation (3) is to promote diverse rankings of query completions by penalizing redundancy at every rank in the list of QAC candidates. To achieve this objective, it selects a candidate that is maximally different from those previously selected in R_R (thus minimizing redundancy), while

still relevant to the query aspect. This is operationalized by iteratively filling R_R with one query $q^\star \in R_I \backslash R_R$ each time until $|R_R| = N$. Thus, we produce this optimal query $q^\star$ at each rank by the objective in Equation (4):

$$q^\star \leftarrow \underset{q_c \in R_I \backslash R_R}{\operatorname{argmax}} \sum_a P(Rel \mid q_c, p, a, C_S) \prod_{q_s \in R_R} (1 - P(Rel \mid a, p, q_s, C_S)), \tag{4}$$

where $P(Rel \mid q_c, p, a, C_S)$ denotes the probability that, for prefix p, candidate q_c meets the query aspect a given the search context C_S, and $P(Rel \mid a, p, q_s, C_S)$ indicates the conditional probability that the selected query q_s for prefix p matches the aspect a given the search context C_S. Thus,

$$\prod_{q_s \in R_R} (1 - P(Rel \mid a, p, q_s, C_S))$$

denotes the probability that all previously selected queries $q_s \in R_R$ fail to satisfy the user, and the product

$$P(Rel \mid q_c, p, a, C_S) \prod_{q_s \in R_R} (1 - P(Rel \mid a, p, q_s, C_S))$$

indicates the probability that none of the selected queries in R_R satisfy the query aspect a but, finally, q_c does. Finally, the QAC candidate $q_c \in R_I \backslash R_R$ with the maximal probability satisfying the query aspect is chosen for inclusion in the list R_R.

3.2. Greedy Query Selection For D-QAC

In this section, we propose our GQS model to deal with D-QAC. In this model, we assume that the probability in Equation (4) that a query candidate q_c meets the query aspect $P(Rel \mid q_c, p, a, C_S)$ can be expressed either by the current search popularity or implicitly by the closeness to previous queries in the session context C_S, with a trade-off λ ($0 \leq \lambda \leq 1$) controlling the contributions from these two parts. Thus,

$$\begin{aligned} P(Rel \mid q_c, p, a, C_S) &= \lambda P(q_c \mid p) + (1-\lambda) P(Rel \mid q_c, a, C_S) \\ &= \lambda P(q_c \mid p) + (1-\lambda) P(Rel \mid q_c, a, q_1, q_2, \ldots, q_{T-1}) \\ &= \lambda P(q_c \mid p) + (1-\lambda) \prod_{q_t \in C_S} P(Rel \mid q_c, a, q_t), \end{aligned} \tag{5}$$

where the query aspect expressed by search popularity $P(q_c \mid p)$ can be directly estimated by

$$P(q_c \mid p) = \frac{f(q_c)}{\sum_{q \in R_I} f(q)}, \tag{6}$$

where $f(q)$ indicates the frequency of query $q \in R_I$. Before combining these two probability scores, they should be normalized. Obviously, the query aspect is dominated solely by the search popularity if $\lambda = 1$ or by the search context if $\lambda = 0$. The probability $P(Rel \mid q_c, a, q_t)$ in Equation (5) can be simply estimated by the normalized distance between q_c and q_p given a specific aspect a, weighted by the temporal intervals:

$$P(Rel \mid q_c, a, q_t) = \omega_t \times \left(1 - \frac{|q_c(a) - q_t(a)|}{\sqrt{\sum_{i=1}^{M_a} (q_c(i) - q_t(i))^2}}\right), \tag{7}$$

where q_t and q_c are represented by feature vectors over M_a aspects to be discussed in Section 3.3. ω_t is a normalized decay factor brought by the temporal interval between

q_t and q_c (namely, q_T) to make $\sum \omega_t = 1$, as we argue that temporally close queries in a search session are apt to share common query aspects. Actually, any other similarity function can be used, including, for example, cosine similarity. However, for computing the distance at a specific aspect, we do need to normalize the score at this specific aspect dimension. Different similarity functions will return similar results. Note that, in our model, all queries are represented by a vector; each entry in this vector indicates the relevance of the query to a particular aspect a. Specifically, $q_c(i)$ denotes the probability of relevance of query q_c represented by a vector to its ith aspect. We compute ω_t as $\omega_t \leftarrow norm(\theta^{TD(q_t)-1})$, where θ is a decay factor and $TD(q_t)$ refers to the time interval, for example, $TD(q_t) = 1$ for the last query q_{T-1} in the context C_S. The query, for example, q_c or q_t, can be represented by a probability distribution over aspects returned by Bayesian probabilistic matrix factorization, which is to be discussed in Section 3.3. Thus, these probabilities can be computed offline before ranking.

Next, the probability $P(Rel \mid a, p, q_s, C_S)$ in Equation (4) indicates to what degree the selected QAC candidate $q_s \in R_R$ meets the query aspect, which can be learned from the search logs. We simplify $P(Rel \mid a, p, q_s, C_S)$ in Equation (4) as

$$P(Rel \mid a, p, q_s, C_S) = P(Rel \mid a, q_s, C_S), \tag{8}$$

indicating the probability that a query candidate matching the query aspect is dominated by the closeness to the aspect of preceding queries in the session. Finally, based on the aforementioned query independency assumption, we further have

$$P(Rel \mid a, p, q_s, C_S) = P(Rel \mid a, q_s, C_S) \propto \prod_{q_t \in C_S} P(Rel \mid q_s, a, q_t), \tag{9}$$

where $P(Rel \mid q_s, a, q_t)$ can be derived in the same way as $P(Rel \mid q_c, a, q_t)$ in Equation (5). By doing so, we can gradually assign one query candidate into the list R_R at a time until reaching the list length $|R_R|$. The details of our query-selection method can be found in Algorithm 1. We first calculate the semantical similarity of query candidates to the search context (Row 3), and inject the most semantically similar query into R_R, shown in Row 6, Algorithm 1, then score the remaining candidates in R_I by measuring how close they are to the query aspects and how diverse they are to the selected query in R_I (see Row 10), and finally select the optimal candidate to R_I by Row 12 and 13. We iteratively select one candidate at a time from the remaining list until $R_R = N$.

Clearly, as shown in Algorithm 1, first, we should initialize R_R. We consider two options. The first option starts with $R_R \leftarrow q^*$, where q^* is the most popular completion in R_I at the time of querying because popular queries normally receive high attention. We write GQS_{MPC} to denote this variant of the GQS model. Another option initializes R_R with the most semantically related query to the previous queries in the search context, where q^* can be directly returned by using word2vec [Mikolov et al. 2013b] because queries in the same session normally express closely similar aspects. We write GQS_{MSR} to denote the variant of our GQS model that starts with the semantically most related query.

Finally, to get these probabilities used in our GQS model, for example, $P(Rel \mid q_c, a, q_t)$ in Equation (5) and $P(Rel \mid q_s, a, q_t)$ in Equation (9), we should know the query distribution over all aspects, which is returned by a Bayesian probabilistic matrix factorization (BPMF)–based algorithm, overcoming the sparseness problem of not knowing the direct relationship between a query and an aspect (see Section 3.3). However, BPMF needs to know at least one aspect label of each query while it happens that we may not know any aspect information about a query. To address this, we use the scenario

ALGORITHM 1: GQS for D-QAC

Input: Prefix p, an initial QAC list R_I, size of returned QAC list: N, search context C_S
Output: A reranked QAC list R_R;
1: $R_R = \emptyset$
2: **for** each candidate $q_c \in R_I$ **do**
3: $FirstQuery(q_c) \leftarrow Similarity(q_c, C_S)$; %%% Alternatively, $MPC(q_c)$
4: **end for**
5: $q^\star \leftarrow \arg\max_{q_c \in R_I} FirstQuery(q_c)$
6: $R_R \leftarrow R_R \cup \{q^\star\}$
7: $R_I \leftarrow R_I \backslash \{q^\star\}$
8: **for** $|R_R| \leq N$ **do**
9: **for** $q_c \in R_I$ **do**
10: $s(q_c) \leftarrow \sum_i P(Rel \mid q_c, p, a, C_S) \prod_{q_s \in R_R} (1 - P(Rel \mid a, p, q_s, C_S))$
11: **end for**
12: $q^\star \leftarrow \arg\max_{q_c} s(q_c)$
13: $R_R \leftarrow R_R \cup \{q^\star\}$
14: $R_I \leftarrow R_I \backslash \{q^\star\}$
15: **end for**
16: **Return** R_R

proposed in Algorithm 2 by finding a labelled target query that is the semantically most related query to the unlabelled query.

3.3. Generating a Query Distribution over Aspects

In this section, we discuss how to generate a query distribution over aspects. We use BPMF to overcome the sparseness problem of not knowing direct relationships between a query and an aspect using the ODP. Before detailing our BPMF-based approach to this, we address the query cold-start problem, that is, not knowing any aspect information about a query using ODP categorization from the training period. We assign the aspect labels from its semantically most related query in the labelled query set, because semantically related queries (or words), which often express similar search aspects, have been either directly suggested to the user or internally used by the search engine to improve the search quality [Bollegala et al. 2007; Chien and Immorlica 2005].

More precisely, given an unlabelled query q and a set of labelled queries Q_L, we return a labelled query $q_o \in Q_L$ for q as

$$q_o \leftarrow \operatorname*{argmax}_{q_l \in Q_L} \cos(q, q_l) = \operatorname*{argmax}_{q_l \in Q_L} \frac{1}{W} \sum_{w_k \in q} \sum_{w_j \in q_l} \cos(w_k, w_j). \tag{10}$$

We take the cosine similarity between two queries, represented by the average of the cosine similarity between two sets of normalized word vectors, excluding the stop words. The word vector representation can be directly returned by word2vec [Mikolov et al. 2013a, 2013b] learned from the query logs. The details are shown in Algorithm 2, in which we first score each labelled query in Q_L by its cosine similarity to the unlabelled input query in row 2, then select the most similar query (Row 4), from which we obtain the aspect labels that are finally assigned to the input query as aspect labels (Row 5). Using Algorithm 2, all queries in our datasets can be categorized by the ODP. After that, BPMF can be applied to derive the query distribution over all aspects in order to calculate the probabilities used in Algorithm 1 to rerank QAC candidates. However, queries are usually short. For example, the majority in the AOL and MSN logs that we use in this article consists of fewer than three words (see Figure 2(b)). Thus, it

ALGORITHM 2: Dealing with the Query Cold-Start Problem

Input: An unlabelled query q, a set of labelled queries Q_L with their labels L
Output: Labels of q: $l(q)$;
1: **for** each query $q_l \in Q_L$ **do**
2: $score(q_l) = \cos(q, q_l)$
3: **end for**
4: $q_o \leftarrow \text{argmax}_{q_l \in Q_L} score(q_l)$
5: $l(q) \leftarrow l(q_o) \in L$
6: **Return** $l(q)$ to q

makes sense to use the clicked documents rather than the query itself to identify query aspects, which is commonly used in search-result diversification. We thus build a large query-aspect matrix $QC_{N_q \times M_a}$ using the ODP, with N_q unique queries and M_a unique aspects, as we will explain.

In the diversity task of TREC,[3] the ground truth is generated by humans both for relevance and for aspects. In our setting of diversified QAC, we first train our model by proposing a new approach for inferring multiaspect relevance for a query from clickthrough data in the log using aspect information from the ODP.[4] The clickthrough data is produced from the search behaviors of real searchers and has been proved effective for labelling the relevance between a document and a query [Joachims 2002]. More specifically, our methodology consists of two major steps. The first step involves extracting the clickthrough data from the search log. By doing so, we obtain a list of all clicked URLs for each unique query. The second step involves categorizing these URLs using the ODP. After that, we infer the aspects of a query by aggregating all aspects from its clicked URLs. Let us make this more precise.

Definition 3.1 (*Multiaspect Relevance*). Let a query be given. Given an aspect set that has m pertinent aspects, that is, an aspect relevance label is independent of other aspect relevance labels, the multiaspect relevance of a query is an m-dimensional vector with each entry corresponding to a relevance label for an aspect given the query.

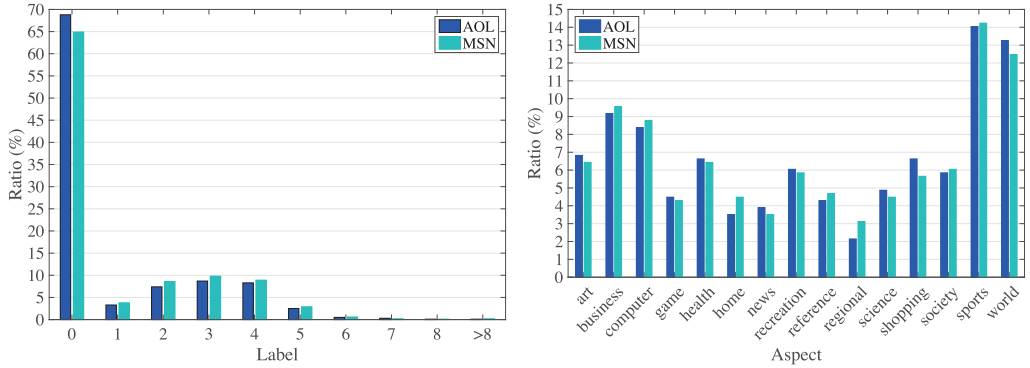
Thus, each entry in a multiaspect relevance vector corresponds to an aspect-relevance label. This relevance label can be mapped to a numerical value $n_e(q, url, a)$ as

$$n_e(q, url, a) = \sum_{url \in U(q)} J(url, a) \times f(q, url), \tag{11}$$

where $U(q)$ contains all clicked URLs of a query q, the indicator $J(url, a) = 1$ if the clicked url is categorized by aspect a and $J(url, a) = 0$ if not, and $f(q, url)$ indicates the number of clicks on URL url after submitting query q. We use the ODP to categorize the clicked URLs. In practice, following Chapelle et al. [2009], we split the aspect-relevance judgments into a 5-grade scale set, such as {*perfect*, *excellent*, *good*, *fair*, *bad*} by $n_e(q, d, a) \leftarrow \min(n_e(q, d, a), 4)$. After checking the distribution of labels (see Figure 1(a)) and aspects[5] (see Figure 1(b)), we find that the relevance labels, for exampel, 2, 3, and 4, account for the majority of nonzero labels. In this manner, we generate our ground truth for the relevance of queries to aspects.

To calculate the probabilities mentioned in Section 3.2, that is, $P(q_c|p, a, C_S)$ in Equation (5) and $P(a|p, q_s, C_S)$ in Equation (8), we should replace the zeros in the

[3]http://trec.nist.gov/tracks.html.
[4]DMOZ – the open directory project, http://www.dmoz.org.
[5]To save space, we plot only the distribution of level-one aspects: in total, 15 aspects.



(a) The ratio of relevance labels of queries at two-level aspects in the AOL and MSN datasets.

(b) The ratio of queries at one-level aspects in the AOL and MSN datasets.

Fig. 1. Distributions of labels (left) and aspects (right) in the AOL and MSN datasets, respectively.

original query-aspect matrix $QC_{N_q \times M_a}$ for the cases in which no direct relationships between query q and aspect a are inferred using the ODP in the training period. We use BPMF [Salakhutdinov and Mnih 2008a] to derive the distribution of queries over aspects. BPMF can be directly applied to the original query-aspect matrix $QC_{N_q \times M_a}$, returning an approximation matrix that assigns a nonzero value to each entry in the original matrix to overcome the problem of sparseness and zero probabilities. By doing so, the original query-aspect matrix $QC_{N_q \times M_a}$ is approximated by:

$$QC^*_{approx} = Q^*_{N_q \times k_f} \times {C^*_{M_a \times k_f}}^\top, \tag{12}$$

where $Q^*_{N_q \times k_f}$ and $C^*_{M_a \times k_f}$ represent the query- and aspect-specific latent feature matrix, and N_q, M_a, and k_f indicate the number of queries, aspects, and latent features, respectively. Like Cai et al. [2014a, 2016b], we compute the value $QC^*_{approx}(i, j)$ of query i at aspect j in matrix QC^*_{approx} by marginalizing over the model parameters and the hyperparameters:

$$p(QC^*_{approx}(i, j)|QC_{N_q \times M_a}, \Theta_0) = \tag{13}$$
$$\int\int p(QC^*_{approx}(i, j)|Q_i, C_j)p(Q, C|QC_{N_q \times M_a}, \Theta_Q, \Theta_C)p(\Theta_Q, \Theta_C|\Theta_0)d\{Q, C\}d\{\Theta_Q, \Theta_C\},$$

where $\Theta_Q = \{\mu_Q, \Sigma_Q\}$ and $\Theta_C = \{\mu_C, \Sigma_C\}$ are hyperparameters of query set Q consisting of all unique queries and of aspect set $\mathbb{A}$ consisting of all unique aspects, respectively. The prior distributions over the query and aspect feature vectors are assumed to be Gaussian, and $\Theta_0 = \{\mu_0, \Sigma_0, W_0\}$ is a Wishart distribution hyperparameter with $\Sigma_0 \times \Sigma_0$ scale matrix W_0. The intuition behind this approximation is that the relevance of a query to aspects is determined by a small number of unobserved hyperparameters. This means that taking a Bayesian approach to the prediction problem involves integrating the model hyperparameters. In addition, the use of Markov chain Monte Carlo (MCMC) methods [Neal 1993] for approximating relevance comes from finding only point estimates of model hyperparameters instead of inferring the full posterior distribution over them, which results in a significant increase in predictive accuracy [Salakhutdinov and Mnih 2008a].

BPMF introduces priors for the hyperparameters, which allows model complexity to be controlled based on the training data [Salakhutdinov and Mnih 2008b]. When the prior is Gaussian, the hyperparameters can be updated by performing a single EM

Table III. An Overview of Models Discussed in the Article

Model	Description	Source
QD-MPC	A typical QAC ranking approach, which ranks QAC candidates according to their current popularity collected from the query logs.	[Bar-Yossef and Kraus 2011]
QD-CON	A context-based query-ranking approach, which reranks QAC candidates (returned by MPC) by a hybrid score considering the query popularity and the similarity to search context in current session.	[Bar-Yossef and Kraus 2011]
QD-QCR	A diversification-oriented query-ranking approach, which reranks QAC candidates (returned by MPC) by selecting queries from distinct query clusters.	[He et al. 2011]
QD-MMR	A diversification-oriented query-ranking approach, which ranks QAC candidates according to both their popularity and the dissimilarity between a query candidate to be selected and those previously selected.	[Carbonell and Goldstein 1998]
GQS_{MPC+AQ}	Greedy query selection approach starting with the most popular query and taking all preceding queries in session as search context.	This article
GQS_{MPC+LQ}	Greedy query selection approach starting with the most popular query and taking only the last preceding query in session as search context.	This article
GQS_{MSR+AQ}	Greedy query selection approach starting with the most semantically related query to the preceding queries in session and taking all preceding queries as search context.	This article
GQS_{MSR+LQ}	Greedy query selection approach starting with the most semantically related query to the preceding queries in session and taking the last preceding query in session as search context.	This article

step [Dempster et al. 1977], which scales linearly with the number of observations without significantly affecting the time to train the model. The details of BPMF can be found in Salakhutdinov and Mnih [2008a]. As some of the values in the matrix QC^*_{approx} generated by BPMF are negative, we normalize QC^*_{approx} to guarantee $QC^*_{approx}(i, j) \in (0, 1)$. After normalizing, distributions of queries over aspects can be produced.

4. EXPERIMENTAL SETUP

Section 4.1 provides an overview of the D-QAC models studied in this article and lists the research questions that guide our experiments, Section 4.2 describes the datasets, Section 4.3 gives details about our evaluation metrics and baselines, Section 4.4 details the design of a side-by-side experiment, and we specify our settings and parameters in Section 4.5.

4.1. Research Questions

We list all models to be discussed in Table III. There are three state-of-the-art baselines and four types of approaches that we introduce in this article: GQS (Algorithm 1) with two notions of context (AQ, all preceding queries vs. LQ, only the last query) and two notions of starting query (MPC, most popular query, vs. MSR, semantically most closely related query).

The research questions guiding the remainder of the article are the following:

RQ1 Does our GQS model beat the baselines for the D-QAC task in terms of metrics for QAC ranking (e.g., MRR) and for diversification (e.g., α-nDCG)? (See GQS models vs. baselines in Section 5.1.)

RQ2 How does the choice of selecting the first query to be included in the QAC result list impact the performance in D-QAC of our GQS model? (See GQS_{MPC+AQ} vs. GQS_{MSR+AQ} and GQS_{MPC+LQ} vs. GQS_{MSR+LQ} in Section 5.2.)

RQ3 What is the impact on D-QAC performance of our GQS model of the choice of search context, that is, choosing all previous queries in a session or only the last preceding query? (See GQS_{MPC+AQ} vs. GQS_{MPC+L} and GQS_{MSR+AQ} vs. GQS_{MSR+LQ} in Section 5.3.)

RQ4 What is the relative D-QAC performance of our QAC models when evaluated using a side-by-side comparison? (See Section 5.4.)

RQ5 What is the sensitivity of our GQS model? In particular, how is the performance of our GQS model influenced by, for example, the number of returned QAC candidates, namely, the cutoff N, the number of latent features used in BPMF k_f, and the trade-off λ in Equation (5)? (See Section 5.5.)

4.2. Datasets

We use two publicly available query log datasets in our experiments: AOL [Pass et al. 2006] and MSN [Craswell et al. 2009]. The queries in the AOL log were sampled between March 1, 2006 and May 31, 2006; those in the MSN log were recorded for one month in May 2006. For consistency, we partition each log into two parts: a training set consisting of the first 75% of the query log (ordered by time) and a test set consisting of the remaining 25%. Additionally, for every log, only queries appearing in both of the two partitions are kept. Traditional k-fold cross-validation is not applicable to a streaming sequence since it would disorder the temporal data [Gama et al. 2014]. Thus, queries in the training set are the ones submitted before May 8, 2006 in the AOL dataset and before May 24, 2006 in the MSN dataset.

We filter out a large volume of navigational queries with URL substrings (.com, .net, .org, http, .edu, www, and so on) and removed queries starting with the special characters (&, $, %, #, and so on) from both datasets. We divide the queries into sessions by 30min inactivity[6] and keep only English queries appearing in both partitions. We focus on sessions consisting of at least two queries because we would like to leverage the search aspect of the search context, that is, the previous queries in the current session.

For each session, the prefix is set to be the first 1 to 5 character(s) of the last query in the session. To get the training/test set, we remove the input prefixes whose ground truth is not included in the top 20 QAC candidates returned by MPC to guarantee that the candidate set contains the final query submission, following standard practice in the experimental design for QAC tasks (e.g., see Jiang et al. [2014], Shokouhi [2013], and Cai et al. [2014b]). We remove all test cases in which the final submitted query cannot be categorized by ODP in the training phase, that is, we cannot infer the query aspect of such queries, making it impossible to generate the ground truth.

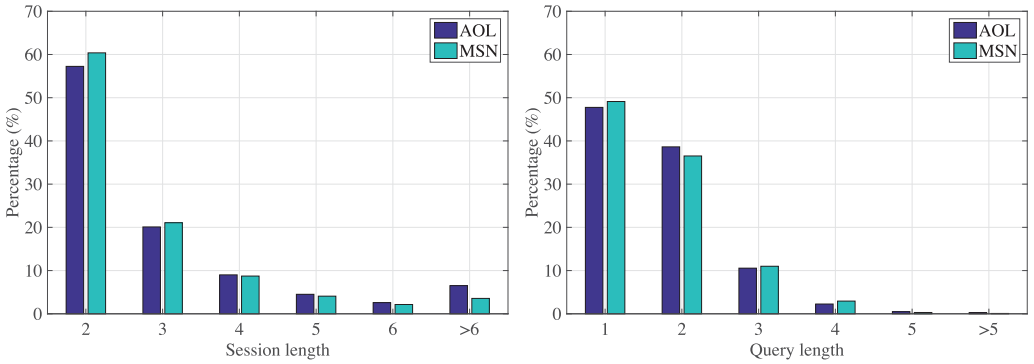
Table IV details the statistics of the datasets used. More than half (64.9%) of all unique queries in AOL can be categorized by our scenario using the ODP. For the MSN log, we reach a higher ratio (66.2%). In the AOL log, if the user clicked on a search result of a query, only the domain portion of the URL in the clicked result is listed; however, in the MSN log, the full URL is recorded. Toi make it consistent with our processing of the AOL logs, we keep only the domains of clicked URLs in the MSN log when inferring the query aspects via clickthrough data. Notice also that, compared to the average number of queries in a session of AOL ($\sim$3.3), the MSN users seem to submit more queries in a session ($\sim$5.5).

[6]This heuristic is only applied to the AOL dataset as the MSN dataset provides session IDs.

Table IV. Statistics of the AOL and MSN Datasets Used

	AOL		MSN	
Variables	Training	Test	Training	Test
# Queries	3,808,083	1,571,346	3,784,925	1,402,308
# Unique queries	452,980	452,980	304,943	304,943
# Labelled Unique Qs	294,363	294,363	201,872	201,872
# Unlabelled Unique Qs	158,617	158,617	103,071	103,071
# Sessions	1,149,528	465,302	674,717	256,512
# Queries / session	3.31	3.38	5.60	5.46
# All prefixes	3,109,247	1,146,768	2,013,671	697,870
# Prefix-1	262,924	90,688	196,831	65,179
# Prefix-2	458,999	162,007	319,362	105,082
# Prefix-3	698,716	251,623	455,109	154,020
# Prefix-4	826,984	309,522	517,570	182,502
# Prefix-5	861,624	332,928	524,799	191,087

Note: The number of prefixes at various lengths are generated when top-20 QAC candidates are returned by MPC, that is, $k_I = 20$. The n in "# prefix-n" indicates the length of the prefix in characters.



(a) The ratio of sessions at various lengths in queries in the AOL and MSN dataset.

(b) The ratio of queries at various lengths in terms in the AOL and MSN dataset.

Fig. 2. Distribution of session length in queries (left) and query length in terms (right) in the processed AOL and MSN datasets, respectively.

In Figure 2, we take a closer look at the processed datasets to be able to report the ratio of queries at various lengths in words and of sessions at various lengths in queries. As shown in Figure 2(a), generally, for both datasets, nearly half of the sessions contain only two queries. The majority of sessions are short (<4 queries), accounting for 77.3% in the AOL log and 81.2% in the MSN log. Long sessions (>6 queries) are really rare, accounting for only 6.5% in the AOL log and only 3.5% in the MSN log. For the length of queries in words, as shown in Figure 2(b), more than 90% of the queries consist of at most three words, making it challenging to infer the query aspects directly from queries.

4.3. Evaluation Metrics and Baselines

To evaluate the effectiveness of QAC rankings, MRR has been commonly used in Shokouhi and Radinsky [2012], Shokouhi [2013], Cai et al. [2014b], Jiang et al. [2014], and Whiting and Jose [2014] as a measure. For a query q with prefix p in the query set associated with a list of QAC candidates $S(p)$ and the user's finally completed query

q', Reciprocal Rank (RR) is computed as

$$RR = \begin{cases} \frac{1}{\text{rank of } q' \text{ in } S(p)}, & \text{if } q' \in S(p) \\ 0, & \text{else.} \end{cases} \tag{14}$$

Then, MRR is computed as the mean of RR for all prefixes.

Regarding the diversity of QAC rankings, we report our results based on the official evaluation metrics in the diversity task of the TREC 2013 Web track [Collins-Thompson et al. 2013]: ERR-IA [Chapelle et al. 2009], α-nDCG [Clarke et al. 2008], NRBP [Clarke et al. 2009], and MAP-IA [Agrawal et al. 2009]. We take α-nDCG at cutoff N as an example for evaluating the diversity of a QAC ranking, which extends the traditional nDCG metric [Järvelin and Kekäläinen 2002] for aspect-specific rankings according to

$$\alpha\text{-}nDCG@N = Z_N \sum_{i=1}^{N} \frac{\sum_{a \in A_p} g_{i|a} (1-\alpha)^{\sum_{j=1}^{i-1} g_{j|a}}}{\log_2(i+1)}, \tag{15}$$

where a is an aspect in the set of query aspects A_p, $g_{i|a}$ denotes the aspect-specific gain of the ith query given aspect a, and the normalization constant Z_N is chosen so that the perfect QAC list gets an α-nDCG@N score of 1. The α-nDCG@N in our experiments is computed at $\alpha = 0.5$ in order to give equal weights to both relevance and diversity, and all rewards are discounted by a log-harmonic discount function of rank. Generally, these diversity metrics reward newly-added aspects and penalize redundant aspects of QAC candidates.

We consider four QAC baselines:

QD-MPC. A popularity-based QAC approach, which ranks the query candidates by their frequency, as computed from the counts in the preceding log [Bar-Yossef and Kraus 2011];

QD-CON. A context-based QAC approach,[7] in which we rerank the query candidates by a hybrid score considering the query popularity and the similarity to the search context in the current session [Bar-Yossef and Kraus 2011];

QD-QCR. A diversification-oriented query-ranking approach based on query clusters [He et al. 2011], which reranks QAC candidates returned by MPC via selecting a query from distinct clusters; the K-means clustering algorithm is applied to cluster the queries with a fixed number of clusters, that is, 5, as we evaluate the results produced by returning at least 5 QAC candidates in our experiments;

QD-MMR. An MMR-based [Carbonell and Goldstein 1998] query ranking, which considers the candidate's popularity as well as its dissimilarity to those previously selected queries with setting a trade-off to 0.5; in particular, the dissimilarity is computed as

$$dissim(q_c, R_R) \leftarrow \frac{1}{|R_R|} \sum_{q \in R_R} (1 - \cos(q, q_c)), \tag{16}$$

where q_c is a query candidate to be selected and q is a query that has been selected in the query set R_R. Both queries are represented by a vector returned by BPMF, as detailed in Section 3.3.

[7]We implement the BPMF process to generate a rich query representation over aspects to overcome the sparsity problem.

Table V. Statistics of the Side-by-Side Experiment

Number of participants	50
Total number of prefixes assessed	2500
Number of prefixes assessed per prefix length (1, 2, 3, 4, 5)	500
Number QAC-candidates shown per prefix per model	10
Number of prefixes assessed per individual participant	50
Number of models compared	5
Number of pairs of QAC models judged	4

Statistical significance of observed differences between the performance of two approaches is tested using a two-tailed paired t-test and is denoted using $^{\blacktriangle}/^{\blacktriangledown}$ for significant differences at level $\alpha = .01$, or $^{\triangle}/^{\triangledown}$ at level $\alpha = .05$.

4.4. Side-by-Side Experiments

In addition to standard contrastive "bake-off" experiments, we use a second method for evaluating the diversity of QAC lists. Following Vallet and Castells [2011], Thomas and Hawking [2006], Vallet [2011], and Chapelle et al. [2012], we show human judges in a lab setting two ranked lists of QAC candidates for a given prefix and ask them which of the two is more diverse. This side-by-side evaluation experiment is performed using 50 master students in computer science. Each participant is given 50 different test prefix samples. For each test prefix, five lists of query candidates returned by corresponding QAC models—baseline, GQS_{MPC+AQ}, GQS_{MPC+LQ}, GQS_{MSR+AQ}, and GQS_{MSR+LQ}—are presented in pairs to the individual participant, who is asked to indicate which query list is more diverse: GQS_{MPC+LQ} versus baseline, GQS_{MPC+LQ} versus GQS_{MPC+AQ}, GQS_{MSR+LQ} versus baseline, and GQS_{MSR+LQ} versus GQS_{MSR+AQ}, respectively. Participants were asked to indicate which of the two lists was more diverse or whether there was a tie. During their assessments, participants were allowed to use a web search engine to help them decide. The selection of pairs of approaches for the side-by-side comparison is aligned with the comparisons of those for the standard contrastive "bake-off" experiments in Section 5.3. By doing so, we can examine the *agreement* on diversity preference between human judgments and the preferences inferred from the contrastive experiments. Table V summarizes the statistics of the side-by-side experiment.

Following Chapelle et al. [2012], we use *agreement* to quantify to which extent the relative order of rankings obtained by computing the α-nDCG@10 scores of system-produced lists of QAC candidates coincides with human preferences.

4.5. Parameters and Settings

Following Bennett et al. [2012], we set the factor $\theta = 0.95$ in a decay function mentioned in Section 3.2. We first use one-third of the original test data for validation to optimize the free parameter λ controlling the contribution of query aspects signaled by the search popularity and the search context in Equation (5), then use the remaining two-thirds for the final test. We further discuss the influence of λ in Section 5.5.1. Similarly, the number of latent features used by BPMF in Section 3.3 is set to $k_f = 10$, which is followed by additional experiments and detailed discussions on the results generated when k_f is changing in Section 5.5.2.

For labeling query aspects, we use multiple relevance labels and proceed as follows. We first get a list of all clicked URLs of remaining queries in the training period. Then, we try to project each clicked URL for a query into a two-level ODP aspect based on the links within each aspect[8] in which the URLs have been categorized. Finally, given

[8]http://www.dmoz.org/docs/en/rdf.html.

a query, we aggregate all aspects of its clicked URLs and use the aggregated aspect information to label the relevance of aspects for the query. Let us consider an example of the labelling process.

Example 4.1. Given a query q, we find that two URLs, for example, d_1 and d_2, are clicked c_1 and c_2 times, respectively, according to the query log after submitting q. At the same time, based on the links within each ODP aspect, d_1 is labelled with aspects a_1 and a_2, and d_2 is labelled with aspects a_2 and a_3. Consequently, to query q, we assign aspects a_1, a_2, and a_3, with counts c_1, $c_1 + c_2$, and c_2 as relevance, respectively.

In total, we find 513 level-two topical aspects originated from 15 level-one topical aspects in our dataset based on this process.[9] These aspects are explicit. For instance, we can find aspects such as "/arts/movies," "/shopping/crafts," and "/business/financial_services," for which "/arts," "/shopping," and "/business" are level-one aspects; "/movies," "/crafts," and "/financial_services" are their corresponding level-two aspects, respectively.

In previous research on QAC, it is often assumed that users are often given a list of top $N = 10$ (at most) query completion candidates. This is a common setting used by many web-search engines and works [Cai et al. 2014b; Jiang et al. 2014; Shokouhi 2013]. In our experiments, we first retrieve the top 20 QAC candidates (at most) as determined by MPC, then return a final list of the top 10 (at most) candidates for evaluation after reranking the original QAC list by each specific model; that is, $N = 10$ is initially used as a cutoff to test the D-QAC performance. In addition, we examine the performance of the models that we discuss when less or more QAC candidates are finally returned, for example, cutoff $N = 5$ and 20, respectively.

In practice, QAC methods should consider efficiency. Thus, the algorithm needs an efficient data structure, like a hash table, to support fast lookups for input prefix keys. Before testing, we generate an initial QAC ranking for each prefix offline by the MPC approach, and represent queries using vectors returned by the BPMF process. Then, for the diversification task, the main cost is from computing the similarities for reranking these QAC candidates. In general, our model can work online given that all these preprocessing steps are done offline.

5. RESULTS AND DISCUSSIONS

In Section 5.1, we examine the performance of models for D-QAC in terms of MRR and α-nDCG@10, and so on. We follow with a section discussing the scenario used in our GQS model for selecting the first query into the QAC list in Section 5.2. We examine the performance of our GQS model under different choices of the search context and zoom in on the performance at each prefix length in Section 5.3. We report on the outcomes of a side-by-side comparison on D-QAC performance in Section 5.4. Section 5.5 details the impact of parameters used in our GQS model and provides an analysis of our GQS model under various settings.

5.1. D-QAC Performance of GQS

To answer our first research question, **RQ1**, we examine the D-QAC performance of all mentioned models in Table III and report the results in Table VI for the AOL and MSN logs, respectively.

As shown in Table VI, QD-QCR achieves the best performance among the four baselines in terms of MRR and diversity metrics, for example, α-nDCG@10. Thus, we use only QD-QCR as the baseline for comparisons with our proposed models in latter

[9]As an aside, in total, there are 16 level-one topical aspects in ODP; the aspect "*Kids and Teens*" was not found as an aspect in our dataset.

Table VI. Performance on the AOL and MSN Logs Based on the Top 20 QAC Candidates Initially Returned by MPC, Which Are Then Reranked by the Methods Listed in the Table

Dataset	Method	MRR	ERR-IA@10	α-nDCG@10	NRBP	MAP-IA
AOL	QD-MPC	.5372	.3765	.6513	.3487	.2768
	QD-CON	.5391	.3782	.6526	.3488	.2783
	QD-QCR	.5393	.3791	.6538	.3491	.2794
	QD-MMR	.5377	.3783	.6530	.3490	.2785
	GQS_{MPC+AQ}	.5465	.3872$^{\triangle}$	.6681$^{\triangle}$	.3598$^{\triangle}$	.2864$^{\triangle}$
	GQS_{MSR+AQ}	.5509$^{\triangle}$	.3958$^{\blacktriangle}$	.6799$^{\triangle}$	.3632$^{\triangle}$	.2885$^{\triangle}$
	GQS_{MPC+LQ}	.5516$^{\triangle}$	.3965$^{\blacktriangle}$	.6852$^{\blacktriangle}$	.3645$^{\blacktriangle}$	.2898$^{\triangle}$
	GQS_{MSR+LQ}	**.5520**$^{\triangle}$	**.4007**$^{\blacktriangle}$	**.6901**$^{\blacktriangle}$	**.3679**$^{\blacktriangle}$	**.2907**$^{\blacktriangle}$
MSN	QD-MPC	.6158	.4184	.6562	.3891	.2546
	QD-CON	.6173	.4211	.6674	.4002	.2613
	QD-QCR	.6191	.4315	.6810	.4064	.2698
	QD-MMR	.6134	.4205	.6658	.3914	.2602
	GQS_{MPC+AQ}	.6285	.4417$^{\triangle}$	.6933	.4138	.2757$^{\triangle}$
	GQS_{MSR+AQ}	.6301	.4438$^{\triangle}$	.6994$^{\triangle}$	.4152$^{\triangle}$	.2771$^{\triangle}$
	GQS_{MPC+LQ}	.6307	.4452$^{\triangle}$	.7003$^{\triangle}$	.4174$^{\triangle}$	.2797$^{\triangle}$
	GQS_{MSR+LQ}	**.6324**$^{\triangle}$	**.4458**$^{\triangle}$	**.7025**$^{\triangle}$	**.4191**$^{\triangle}$	**.2794**$^{\triangle}$

Note: The results are reported at a cutoff of $N = 10$ for all prefixes. The values produced by the best baseline and the best performer in each column are underlined and boldfaced, respectively. Statistical significance of pairwise differences (GQS model vs. the best baseline) are determined by the t-test ($^{\blacktriangle/\blacktriangledown}$ for $\alpha = .01$, or $^{\triangle/\triangledown}$ for $\alpha = .05$).

experiments both on the AOL and MSN logs. In particular, for most test cases in these two datasets, all these baselines can return the final submitted queries early at the top two positions, which is evidenced by the fact that the MRR scores are larger than 0.5. In addition, the MRR scores of the baselines are close to each other. In particular, on the AOL log, QD-QCR achieves a competitive MRR score with QD-CON, but reports a minor MRR improvement (<0.5%) against QD-MMR and QD-MPC. QD-MMR displays a marginally better performance than QD-MPC in terms of MRR, indicating that, for some cases, QD-MMR is able to remove some redundant candidates in the original QAC list generated by the basic MPC approach. However, the MRR improvement is limited because QD-MMR partially relies on the query popularity by which QD-MPC solidly ranks the QAC candidates and QD-MMR does not consider the in-session context when calculating the dissimilarity between queries. In contrast, on the MSN log, the baselines achieve a somewhat higher MRR and diversity scores compared to those on the AOL log. This can be attributed to the difference in session length. Compared to the sessions in the AOL log, the average length of sessions in the MSN log is much longer and some repeated queries are found, which helps identify the query aspects of the last query in session and result in higher MRR and diversity scores.

In order to assess the performance of our GQS models, we compare their results against those of the baseline, QD-QCR. For the AOL log, as shown in Table VI, compared to the baseline, all of our four QGS models outperform it in terms of MRR and the diversity metrics. However, the improvements of our models are limited, starting from a relatively low level of 1.37% improvement achieved by GQS_{MPC+AQ} to a peak improvement of 2.25% reported by GQS_{MSR+LQ} in terms of MRR. In the middle, GQS_{MPC+LQ} outperforms GQS_{MSR+AQ}, reaching an MRR improvement of 2.28% against the baseline, while 2.15% made by GQS_{MSR+AQ} against the baseline. All improvements of our GQS models, except GQS_{MPC+AQ}, against the baseline are statistically significant at level $\alpha = 0.05$ using the t-test. For the diversity results, our GQS models report notable improvements. As an example, let us take the diversity metric α-nDCG@10 to analyze the models' performance. Clearly, GQS_{MSR+LQ} performs best, resulting in a 5.55%

improvement over the baseline on the AOL log. Competitive results are generated by GQS_{MPC+LQ}, with a 0.7% drop against GQS_{MSR+LQ} but still a 4.80% improvement over the baseline. Importantly, the improvements achieved by GQS_{MPC+LQ} and GQS_{MSR+LQ} over the baseline are significant at $\alpha = 0.01$. In contrast, more modest improvements over the baseline in terms of α-nDCG@10 are achieved by GQS_{MPC+AQ} and GQS_{MSR+AQ}, both of which are significant at $\alpha = 0.05$.

In contrast, on the MSN log, we can see from Table VI that the best results are again generated by GQS_{MSR+LQ} but with a more modest MRR improvement against the baseline compared to that on the AOL log. In terms of MRR, significant improvements are achieved only by the GQS_{MSR+LQ} model over the baseline at level $\alpha = 0.05$. This can be explained by the fact that most QAC rankings generated both by the baseline and by our models are high performing, leaving limited space for significant MRR improvements of our models over the baseline. Regarding the diversity results on the MSN log, generally, our GQS models beat the baseline in terms of four metrics in Table VI, Columns 3 to 6. Significant improvements over the baseline in terms of these four diversity metrics are achieved at level $\alpha = 0.05$ for most cases. In particular, using α-nDCG@10, GQS_{MSR+LQ} achieves the largest improvement, around 3.15% over the baseline. Compared to the MRR improvements achieved by our GQS models, the improvements in terms of α-nDCG@10 are more pronounced. For some cases, redundant queries can be removed from the QAC list by GQS models, resulting in improved diversity scores. However, these redundant candidates could be ranked lower than the final submitted query in the original QAC list. Consequently, they do not affect the reciprocal rank score; thus, this has a limited impact on the MRR scores.

Summing up, based on the results achieved on the AOL and MSN logs, we conclude that our greedy query selection approach can remove redundant queries in the original QAC list, returning the final submitted query early and making the final returned list cover more aspects of queries. We will compare our GQS models in detail in Sections 5.2 and 5.3.

5.2. Effect of the First Candidate Selected by GQS

Next, we move to research question **RQ2**, for which we test our proposed GQS models on both the AOL and MSN log under different choices of the search context used, that is, *all* preceding queries before the last query in the session or *only the last* preceding query. We begin by analyzing the results generated on the AOL log and reported in Table VI.

First, we use all previous queries before the last query in session as search context: $C_S \leftarrow \{q_1, q_2, \ldots, q_{T-1}\}$. Then, we can compare the results produced by GQS_{MPC+AQ} and GQS_{MSR+AQ} in Table VI to examine the effect of the first query candidate chosen in our GQS model. Clearly, GQS_{MSR+AQ} outperforms GQS_{MPC+AQ} in terms of MRR and the diversity metrics. Thus, to some extent, our GQS model, starting with the query candidate that is semantically most similar to the current search context for D-QAC tasks, outperforms that choosing the most popular candidate first into the final list R_R. In particular, as shown in Table VI, for QAC ranking on AOL, GQS_{MSR+AQ} achieves a marginal MRR improvement near 1% over GQS_{MPC+AQ}. For some cases, GQS_{MPC+AQ} and GQS_{MSR+AQ} start with the same candidate, that is, the most popular candidate is also the most semantically related one. Consequently, these two models generate the absolutely same QAC ranking lists. As to QAC diversification, GQS_{MSR+AQ} achieves very high α-nDCG@10 scores of over 0.6, and still achieves near 2% improvement against GQS_{MPC+AQ}, indicating that, on the aspect level, GQS_{MSR+AQ} returns more queries with multiple aspects as well as pushes the potential query to be submitted higher than GQS_{MPC+AQ}. Similar findings can be obtained by setting the last preceding query in a session as the search context, that is, $C_S \leftarrow q_{T-1}$ by comparing

Table VII. Performance, in Terms of MRR and α-nDCG@10, of GQS Models Under Various Choices of the First Query Candidate Selected and of the Search Context Used, at a Prefix Length #p Ranging from 1 to 5 Characters on the AOL Log

Metric	#p	Baseline	GQS$_{MPC+AQ}$	GQS$_{MSR+AQ}$	GQS$_{MPC+LQ}$	GQS$_{MSR+LQ}$
MRR	1	.4673	.4716	.4738	.4739	**.4745**
	2	.4861	.4927	.4946	.4954	**.4960**$^{\triangle}$
	3	.5140	.5221	.5253$^{\triangle}$	.5258$^{\triangle}$	**.5263**$^{\triangle}$
	4	.5556	.5620	.5681$^{\triangle}$	.5686$^{\triangle}$	**.5689**$^{\triangle}$
	5	.5889	.5975	.6030$^{\triangle}$	.6039$^{\triangle}$	**.6045**$^{\triangle}$
α-nDCG@10	1	.6012	.6117	.6235$^{\triangle}$	.6272$^{\blacktriangle}$	**.6315**$^{\blacktriangle}$
	2	.6270	.6393	.6496$^{\triangle}$	.6551$^{\blacktriangle}$	**.6592**$^{\blacktriangle}$
	3	.6357	.6490$^{\triangle}$	.6605$^{\triangle}$	.6658$^{\blacktriangle}$	**.6713**$^{\blacktriangle}$
	4	.6611	.6758$^{\triangle}$	.6883$^{\blacktriangle}$	.6944$^{\blacktriangle}$	**.6984**$^{\blacktriangle}$
	5	.6882	.7051$^{\triangle}$	.7171$^{\blacktriangle}$	.7220$^{\blacktriangle}$	**.7276**$^{\blacktriangle}$

Note: The best performer per row is in boldface. Statistical significance of pairwise difference (GQS model vs. baseline) is determined using the t-test ($^{\blacktriangle/\blacktriangledown}$ for $\alpha = .01$, or $^{\triangle/\triangledown}$ for $\alpha = .05$).

GQS$_{MPC+LQ}$ versus GQS$_{MSR+LQ}$. As shown in Table VI, compared with the MRR difference between GQS$_{MPC+AQ}$ and GQS$_{MSR+AQ}$, the MRR margin between GQS$_{MPC+LQ}$ and GQS$_{MSR+LQ}$ is smaller. The same phenomena can be found on the metric α-nDCG@10. However, both GQS$_{MPC+LQ}$ and GQS$_{MSR+LQ}$ show better performance than GQS$_{MPC+AQ}$ and GQS$_{MSR+AQ}$, which motivates us to consider research question **RQ3** in Section 5.3. Thus, so far on the AOL log, we can conclude that the first query selected in our GQS model impacts the QAC ranking performance in the D-QAC tasks and our GQS model can achieve better D-QAC performance when starting with the most semantically similar query rather than the most popular.

The results achieved on the AOL log discussed in Table VI are produced by averaging the scores of all prefixes at different lengths of prefixes, ranging from 1 to 5. Next, we compare our models at specific prefix lengths. For comparison, we report the results in terms of MRR and α-nDCG@10 in Table VII. Generally, as shown in Table VII, our GQS models, starting with the most semantically related query, that is, GQS$_{MSR+AQ}$ and GQS$_{MSR+LQ}$, perform better in terms of MRR than corresponding GQS models starting with the most popular query, that is, GQS$_{MPC+AQ}$ and GQS$_{MPC+LQ}$. Interestingly, significant improvements of GQS models over the baseline are more easily observed at long prefixes, for example, #$p = 4$ and 5, than short ones, for example, #$p = 1$ and 2. Intuitively, longer prefixes can cut down the space of candidates sharply, thus can include more similar candidates in the original QAC list, which would be pushed down in the list by our approaches, resulting in significant improvements over the baseline.

However, the MRR improvements of GQS$_{MSR+AQ}$ over GQS$_{MPC+AQ}$ and of GQS$_{MSR+LQ}$ over GQS$_{MPC+LQ}$ are not significant. In terms of α-nDCG@10, as reported in Table VII, one particularly different finding is that, for some cases, our GQS model can report significant improvements over the baseline at level $\alpha = .01$, which is not the case for MRR, as reported in Table VII. In addition, a near 2% improvement of GQS$_{MSR+AQ}$ over GQS$_{MPC+AQ}$ and around 1% improvement of GQS$_{MSR+LQ}$ over GQS$_{MPC+LQ}$ are observed in terms of α-nDCG@10. In contrast, the analogous MRR improvements in Table VII are lower; this means that our models can help diversify queries.

Next, we turn our attention to results obtained using the MSN log, as reported in Table VI. Some findings consistent with those achieved on the AOL log can be observed: (1) the first query to be selected by the GQS approaches has a slight impact on D-QAC performance; (2) compared to starting with the most popular query, selecting the semantically most closely related query first in the GQS models is more effective. We report on performance at the prefix level in terms of MRR and α-nDCG@10 in

Table VIII. Performance, in Terms of MRR and α-nDCG@10, of GQS Models Under Different Choices of the First Query Candidate Selected and of the Search Context Used, at a Prefix Length #p Ranging from 1 to 5 Characters on the MSN Log

Metric	#p	Baseline	GQS$_{MPC+AQ}$	GQS$_{MSR+AQ}$	GQS$_{MPC+LQ}$	GQS$_{MSR+LQ}$
MRR	1	.4881	.4963	.4991$^{\triangle}$	.4995$^{\triangle}$	**.5014**$^{\triangle}$
	2	.5456	.5538	.5572$^{\triangle}$	.5578$^{\triangle}$	**.5605**$^{\triangle}$
	3	.6041	.6147	.6134	.6154	**.6152**
	4	.6523	.6616	.6647	.6646	**.6673**$^{\triangle}$
	5	.6846	.6945	.6948	.6960	**.6975**
α-nDCG@10	1	.6313	.6395	.6427	.6423	**.6431**
	2	.6564	.6637	.6691	.6708$^{\triangle}$	**.6721**$^{\triangle}$
	3	.6679	.6839$^{\triangle}$	.6893$^{\triangle}$	.6927$^{\triangle}$	**.6942**$^{\blacktriangle}$
	4	.6916	.7051	.7113$^{\triangle}$	.7132$^{\triangle}$	**.7146**$^{\triangle}$
	5	.7118	.7245	.7302$^{\triangle}$	.7332$^{\triangle}$	**.7351**$^{\triangle}$

Note: The best performer per row is boldfaced. Statistical significance of pairwise difference (GQS model vs. baseline) is determined using the t-test ($^{\blacktriangle/\blacktriangledown}$ for $\alpha = .01$, or $^{\triangle/\triangledown}$ for $\alpha = .05$).

Table VIII. As shown in Table VIII, few MRR improvements over the baseline achieved by our models at various prefix lengths is significant. However, our models do produce more diverse QAC rankings as they receive higher α-nDCG@10 scores compared to the baseline, especially when using the last query in the session as the search context. We can find from Table VIII that whatever search context is used, again, selecting the most semantically related query by GQS models is more effective than injecting the most popular query into the QAC ranking list first. In addition, long prefixes (e.g., #$p = 4$ and 5) seem to gain more in terms of diversity than short ones (e.g., #$p = 1$ and 2), which is also confirmed by the significance tests; for example, GQS$_{MPC+LQ}$ and GQS$_{MSR+LQ}$ achieve significant improvements at $\alpha = .05$ in terms of α-nDCG@10 over the baseline at #$p = 4$ and 5, but not at #$p = 1$. These results are consistent with those on the AOL log. Thus, apart from the conclusions established based on the AOL log, we come to another conclusion: that our models can work better for D-QAC tasks in the cases in which users continue to type more in the search box—bigger diversity gains over the baseline are achieved with long prefixes rather than with short inputs.

5.3. Effect of the Search Context used by GQS

In this section, we address research question **RQ3** by changing the search context, that is, using either the most recent query q_{T-1} as C_S in Equation (4) or all preceding queries $C_S \leftarrow \{q_1, q_2, \ldots, q_{T-1}\}$. As shown in Figure 2(a), nearly half of the sessions consist of more than two queries. In addition, we argue, in a long session with multiple queries, the query aspects of later queries may be changed as the searcher has read some results returned for previous queries; thus, they may be different from the original one. The last preceding query in the search context could be a good signal of the user's updated query aspects. We first compare the results reported in Table VI on the AOL and MSN logs, then move to a prefix level analysis.

We first compare the overall results of GQS$_{MPC+LQ}$ against GQS$_{MPC+AQ}$ generated on the AOL log and reported in Table VI. The improvements of GQS$_{MPC+LQ}$ over GQS$_{MPC+AQ}$ in terms of diversity, for example, α-nDCG@10, are obvious, but not in terms of MRR. For instance, GQS$_{MPC+LQ}$ shows a near 3% improvement against GQS$_{MPC+AQ}$ in terms of α-nDCG@10, but a less than 1% MRR improvement. In addition, a statistically significant improvement at level $\alpha = .05$ is observed in terms of α-nDCG@10, but not in terms of MRR. Apparently, using only the last query as search context in our GQS models can generate notably diverse QAC ranking list on AOL. Similar findings are obtained by comparing GQS$_{MSR+LQ}$ against GQS$_{MSR+AQ}$, although the

Table IX. Per Prefix Bake-off on the AOL Log, in Terms of MRR and α-nDCG@10: GQS_{MPC+LQ} versus Other Models

	MRR						α-nDCG@10					
$\#p$	Baseline			GQS_{MPC+AQ}			Baseline			GQS_{MPC+AQ}		
1	26.83	37.91	35.26	17.58	60.39	22.03	22.37	24.68	52.95	15.14	58.02	26.84
2	20.72	49.17	30.11	18.06	62.15	19.79	20.36	26.83	52.81	14.65	58.97	26.38
3	14.68	60.47	24.85	16.23	63.06	20.71	18.94	27.02	54.04	12.21	60.62	27.17
4	11.38	61.24	27.38	16.08	62.97	20.95	18.51	27.63	53.86	11.45	62.78	25.77
5	10.62	62.54	26.84	15.41	61.53	23.06	18.43	28.37	53.20	9.84	64.15	26.01

Note: The ratios (%) of test prefixes at various lengths for which GQS_{MPC+LQ} loses against the corresponding model listed in Row 2 have a red background, ratios with equal performance have a yellow background, and those of prefixes for which GQS_{MPC+LQ} wins have a green background.

Table X. Per Prefix Bake-off on the AOL Log, in Terms of MRR and α-nDCG@10: GQS_{MSR+LQ} versus Other Models

	MRR						α-nDCG@10					
$\#p$	Baseline			GQS_{MSR+AQ}			Baseline			GQS_{MSR+AQ}		
1	24.37	36.53	39.10	17.91	63.31	18.78	20.12	21.83	58.05	17.60	62.61	19.79
2	19.65	47.86	32.49	16.47	64.39	19.14	20.21	23.07	56.72	17.08	63.75	19.17
3	13.78	57.85	28.37	15.93	66.87	17.20	19.55	24.38	56.07	16.35	65.05	18.60
4	12.84	59.17	27.99	15.55	67.62	16.83	18.42	25.82	55.76	15.21	66.36	18.43
5	11.53	61.02	27.45	14.41	68.15	17.44	18.59	27.03	54.38	15.57	67.32	17.11

Note: The ratios (%) of test prefixes at various lengths for which GQS_{MSR+LQ} loses against the corresponding model listed in Row 2 have a red background, ratios with equal performance have a yellow background, and those of prefixes for which GQS_{MSR+LQ} wins have a green background.

improvements are smaller. We compare the results for different search contexts at various prefix lengths in Table VII in terms of MRR and α-nDCG@10, respectively. The MRR improvements of GQS_{MPC+LQ} over GQS_{MPC+AQ} and of GQS_{MSR+LQ} over GQS_{MSR+AQ} are limited, but stable, at different prefix lengths. However, the α-nDCG@10 results set GQS_{MPC+LQ} apart from GQS_{MPC+AQ} with significant improvements at $\alpha = .05$, except for $\#p = 5$. In contrast, GQS_{MSR+LQ} and GQS_{MSR+AQ} yield very similar α-nDCG@10 scores. We attribute these findings to the fact that (1) diverse queries can be returned by our GQS models usually at positions lower than the final submitted query, resulting in undistinguishable MRR scores but discriminable diversity scores; and (2) half of the sessions consist of only two queries (see Figure 2(a)), which means that the search contexts used are the same, resulting in many ties. To verify it, we compare GQS_{MPC+LQ} versus GQS_{MPC+AQ} and GQS_{MSR+LQ} versus GQS_{MSR+AQ} as well as the baseline in a per-prefix bake-off. We report the results in Tables IX and X, respectively.

From Table IX, we see that, against the baseline, GQS_{MPC+LQ} wins many comparisons, especially in terms of α-nDCG@10 (>50%). We also find many ties between the baseline and GQS_{MPC+LQ} in terms of MRR. In contrast, GQS_{MPC+LQ} yields a majority of draws against GQS_{MPC+AQ} in terms of both MRR and α-nDCG@10. Some of the draws occur when GQS_{MPC+LQ} and GQS_{MPC+AQ} return the same ranked list of QAC candidates; others happen on prefixes for which the two models return the final submitted query at top positions in the QAC list, for example, 1 or 2. As to a comparison of GQS_{MSR+LQ} versus GQS_{MSR+AQ}, similar results can be found except that there are more ties in terms of MRR and α-nDCG@10. One particularly interesting point shown in Tables IX and X is that, for most cases, more ties occur when the prefix becomes longer. This is because the QAC models can return the final submitted query early on long prefixes, thus can generate more similar QAC rankings.

Table XI. Per Prefix Bake-off on the MSN Log, in Terms of MRR and α-nDCG@10: GQS_{MPC+LQ} versus Other Models

	MRR						α-nDCG@10					
$\#p$	Baseline			GQS_{MPC+AQ}			Baseline			GQS_{MPC+AQ}		
1	25.14	39.43	35.43	16.17	63.51	20.32	27.65	21.75	50.60	14.07	62.34	23.59
2	18.62	50.83	30.55	15.57	65.01	19.42	25.18	22.82	52.00	13.26	64.13	22.61
3	13.69	62.45	23.86	13.69	67.87	18.44	26.74	23.95	49.31	11.33	66.22	22.45
4	12.79	63.53	23.68	13.16	68.69	18.15	22.14	26.07	51.79	10.76	67.66	21.58
5	13.02	64.68	22.30	12.88	69.79	17.33	25.28	26.73	47.99	10.01	67.79	22.20

Note: The ratios (%) of test prefixes at various lengths for which GQS_{MPC+LQ} loses against the corresponding model listed in Row 2 have a red background, ratios with equal performance have a yellow background, and those of prefixes for which GQS_{MPC+LQ} wins have a green background.

Table XII. Per Prefix Bake-off on the MSN log, in Terms of MRR and α-nDCG@10: GQS_{MSR+LQ} versus Other Models

	MRR						α-nDCG@10					
$\#p$	Baseline			GQS_{MSR+AQ}			Baseline			GQS_{MSR+AQ}		
1	22.63	40.77	36.60	14.33	64.32	21.35	28.39	23.43	48.18	15.37	63.07	21.56
2	15.32	51.75	32.93	13.57	65.33	21.10	23.73	23.24	53.03	15.10	64.12	20.78
3	12.27	62.68	25.05	12.23	68.11	19.66	23.85	25.07	51.08	14.22	65.91	19.87
4	11.48	63.93	24.59	12.54	68.79	18.67	23.77	25.87	50.36	14.51	66.74	18.75
5	10.65	65.11	24.24	11.53	69.70	18.77	22.26	26.31	51.43	13.24	67.35	19.41

Note: The ratios (%) of test prefixes at various lengths for which GQS_{MSR+LQ} loses against the corresponding model listed in Row 2 have a red background, ratios with equal performance have a yellow background, and those of prefixes for which GQS_{MSR+LQ} wins have a green background.

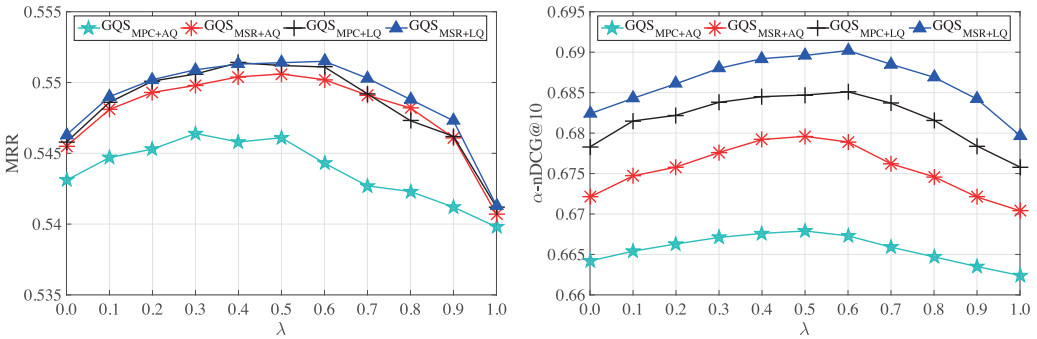
The outcomes of the main comparisons on the MSN log—that is, GQS_{MPC+LQ} versus GQS_{MPC+AQ} and GQS_{MSR+LQ} versus GQS_{MSR+AQ}—are consistent with those on the AOL log. Regarding the prefix-level analysis (see Table VIII), although the GQS models using the last query as search context still beat the corresponding models that use all preceding queries in terms of MRR and α-nDCG@10, the improvements are not statistically significant. Some significant improvements of GQS_{MPC+LQ} and GQS_{MSR+LQ} against the baseline are observed, especially on long prefixes. Similarly, we report on a per-prefix bake-off in terms of MRR and α-nDCG@10 in Tables XI and XII. As the MSN log contains more sessions with only two queries than the AOL log (see Figure 2(a)), more draws are found between GQS_{MPC+LQ} versus GQS_{MPC+AQ} as well as GQS_{MSR+LQ} versus GQS_{MSR+AQ}. This simply happens because, for two-query sessions, the search context used in GQS models consisting either of all preceding queries or only of the last query are always the same.

5.4. Side-by-Side Experiments

To answer **RQ4**, we follow the set-up in Chapelle et al. [2012] and investigate the agreement between the side-by-side comparison produced by human judges and the relative ranking of QAC approaches that results from the bake-offs discussed in Section 5.3 (see Table XIII). We find that the two evaluation methodologies point in the same direction in the vast majority of pairwise comparisons: the agreement ranges between 83% and 96%. For instance, for the comparison between GQS_{MPC+LQ} and the baseline, we find that human preferences agree with the preferences obtained from the bake-offs in Section 5.3 in more than 90% of the cases. The agreement between the two types of preference for GQS_{MPC+LQ} and GQS_{MPC+AQ} are somewhat lower. We explain this as follows. The difference in performance in terms of diversity between GQS_{MPC+LQ} and

Table XIII. Agreements (%) Between Side-by-Side Comparisons by Humans and Per-Prefix Bake-offs by Algorithms

	GQS_{MPC+LQ} vs.		GQS_{MSR+LQ} vs.	
#p	Baseline	GQS_{MPC+AQ}	Baseline	GQS_{MSR+AQ}
1	89.20	85.20	92.60	83.40
2	90.20	86.20	93.40	84.60
3	91.40	87.40	94.40	85.40
4	92.40	88.40	95.00	87.00
5	94.00	90.60	96.00	88.20



(a) Performance in terms of MRR. **(b) Performance in terms of α-nDCG@10.**

Fig. 3. Effect on D-QAC performance of GQS models in terms of MRR (left) and α-nDCG@10 (right) by changing the trade-off λ in Equation (5), tested on the AOL log.

GQS_{MPC+AQ} is smaller than between GQS_{MPC+LQ} and the baseline, making it harder for human judges to identify differences or to identify the direction of the difference. We also observe that agreement tends to be higher for longer prefixes. We explain this as follows. With longer prefixes, the number of possible completions is smaller than for shorter prefixes, reducing the possibilities for disagreement and making it easier for both systems and humans find it easier to determine which aspects and completions are relevant.

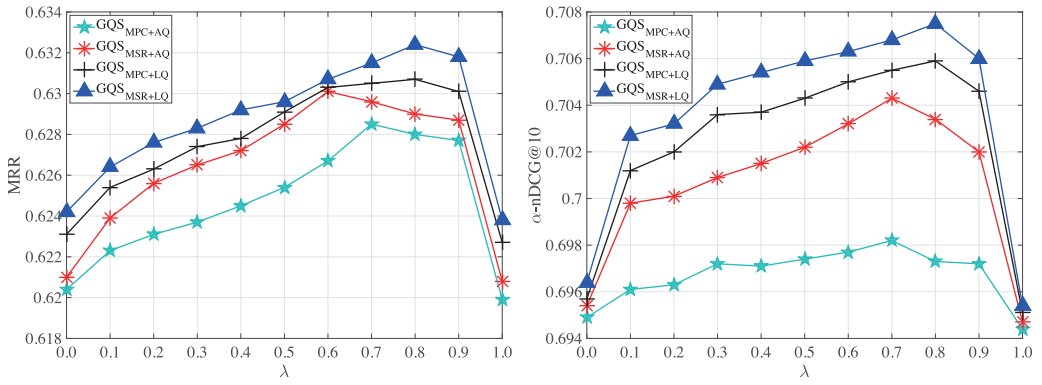
Given the high levels of agreement between human preferences and preferences induced from contrastive experiments, we conclude that the (significant) differences between QAC approaches that we found in Section 5.3 are confirmed by the side-by-side experiments.

5.5. Impact of Parameter Tuning

In this section, we conduct a parameter sensitivity analysis of our GQS models. We examine the performance of our GQS models in Section 5.5.1 by changing the trade-off parameter λ in Equation (5) and by varying the number of latent features k_f used for BPMF in Section 5.5.2. We then see how the models perform when more (or fewer) QAC candidates are returned by varying the cutoff N in Section 5.5.3.

5.5.1. Zooming in on the Trade-off Parameter λ in Equation (5). We first examine the overall performance of our GQS models in terms of MRR and α-nDCG@10 by gradually changing the trade-off parameter λ from 0 to 1 with interval 0.1, then plot the results in Figures 3 and 4 tested on the AOL and MSN logs, respectively.

On the AOL log, we can see from Figure 3(a) that, when λ varies from 0 to 0.3, the MRR scores of all GQS models are invariably increased; they continue to go up until $\lambda = 0.5$ except for one case in which GQS_{MPC+AQ} displays an MRR decrease after



(a) Performance in terms of MRR. **(b) Performance in terms of α-nDCG@10.**

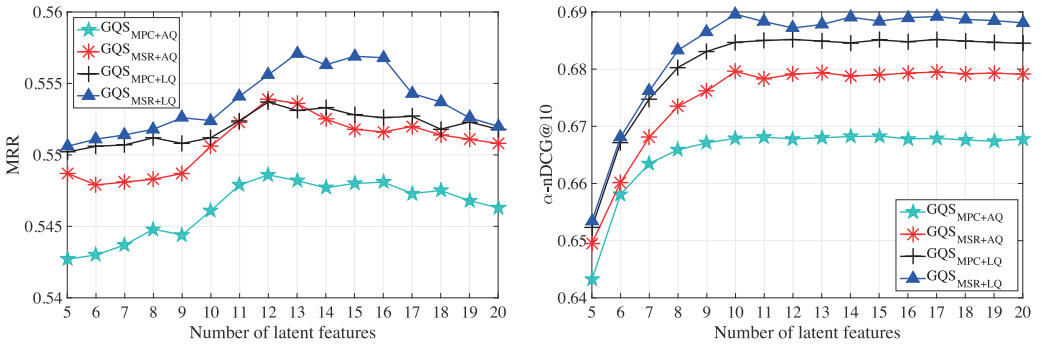
Fig. 4. Effect on D-QAC performance of GQS models in terms of MRR (left) and α-nDCG@10 (right) by changing the trade-off λ in Equation (5), tested on the MSN log.

$\lambda = 0.3$. For most GQS models, the performance in terms of MRR goes down when λ changes from 0.5 to 1. For any GQS model, if it only focuses on search popularity, that is, $\lambda = 1$ in Equation (5), the performance is worse than when it focuses only on the search context, that is, $\lambda = 0$ in Equation (5). In terms of α-nDCG@10, the peak performance appears near $\lambda = 0.5$ for GQS_{MPC+AQ} and GQS_{MSR+AQ} or near $\lambda = 0.6$ for GQS_{MPC+LQ} and GQS_{MSR+LQ}. For any λ, GQS_{MSR+LQ} always performs best among the four models in terms of both MRR and α-nDCG@10.

In contrast, for MRR on the MSN log, GQS_{MPC+LQ} and GQS_{MSR+LQ} favor a large λ. For instance, they achieve peak MRR scores near $\lambda = 0.8$. However, GQS_{MPC+AQ} and GQS_{MSR+AQ} prefer a somewhat smaller λ than GQS_{MPC+LQ} and GQS_{MSR+LQ}. As shown in Figure 4(a), a maximal MRR score is returned for GQS_{MSR+AQ} near $\lambda = 0.6$ and near $\lambda = 0.7$ for GQS_{MPC+AQ}. For the performance in terms of α-nDCG@10, a sharp increase is observed when λ changes from 0 to 0.1 on all four models, as shown in Figure 4(b). This means that the search context does help diversify the query candidates. In addition, the α-nDCG@10 scores go up until $\lambda = 0.8$ for GQS_{MPC+LQ} and GQS_{MSR+LQ} and $\lambda = 0.7$ for GQS_{MPC+AQ} and GQS_{MSR+AQ}. All four GQS models present a relatively low score when $\lambda = 1.0$. In addition, compared to GQS_{MPC+AQ} and GQS_{MSR+AQ}, the other two GQS models show bigger fluctuations in terms of both MRR and α-nDCG@10 when λ changes.

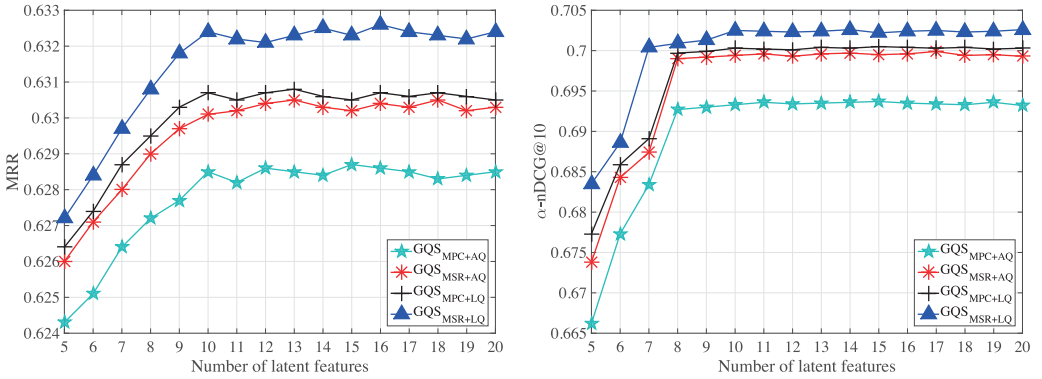
From the observations in Figures 3 and 4, we can conclude that (1) in our GQS models for the D-QAC task, search popularity and search context are both important for query diversification. Compared to search popularity, search context may contribute much more in GQS models for D-QAC as a larger λ ($0.5 < \lambda < 0.9$) results in better performance than that of $0.1 < \lambda < 0.4$, especially on the MSN log (see Figure 4); (2) the search context used in GQS models, that is, either only the last query or all preceding queries in session, has a small impact on the performance as these four GQS models show their peak performance at various λ; and (3) λ may exert a bigger influence on α-nDCG@10 than MRR as relatively noticeable margins can be seen when λ changes shown in Figures 3(b) and 4(b).

5.5.2. Effect of the Number of Latent Features k_f Uses in Bayesian Probabilistic Matrix Factorization. Next, we zoom in on the number of latent features k_f used in Bayesian probabilistic matrix factorization for generating the query distributions over aspects.



(a) Performance in terms of MRR. **(b) Performance in terms of α-nDCG@10.**

Fig. 5. Effect on D-QAC performance of GQS models in terms of MRR (left) and α-nDCG@10 (right), tested on the AOL log, by changing the number of latent features used in BPMF.

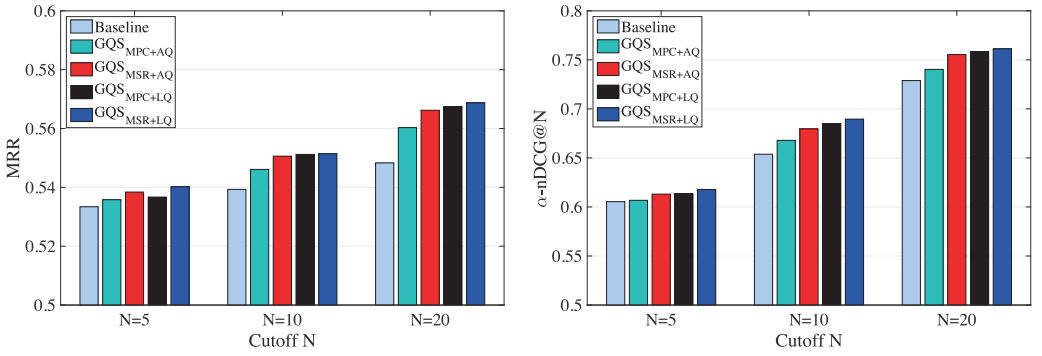


(a) Performance in terms of MRR. **(b) Performance in terms of α-nDCG@10.**

Fig. 6. Effect on D-QAC performance of GQS models in terms of MRR (left) and α-nDCG@10 (right), tested on the MSN log, by changing the number of latent features used in BPMF.

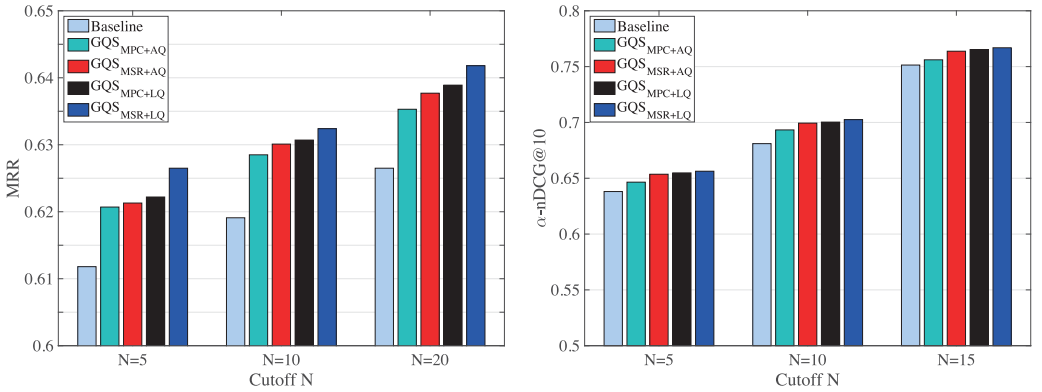
We manually vary the value of k_f in GQS models from 5 to 20. See Figure 5 on the AOL log and Figure 6 on the MSN log, respectively.

Generally, for the AOL log, when the number of latent features k_f used in BPMF increases from 5 to 12, the performance of our GQS models increases dramatically in terms of MRR, with a little fluctuation. However, the α-nDCG@10 scores stop increasing after $k_f = 10$. In addition, when the number of latent features k_f varies from 10 to 20, the performance of our GQS models seems to level off, especially in terms of α-nDCG@10. Another important finding is that the performance in terms of MRR sometimes goes down when k_f increases. For instance, when k_f varies from 18 to 20, the MRR scores of the GQS models, except GQS_{MPC+LQ}, drops. For the MSN log, the MRR scores of the GQS models invariably increase from $k_f = 5$ to 10 for all GQS models and remain stable for $k_f = 10, \ldots, 20$. Compared to the MRR results on the AOL log (in Figure 5(a)), the GQS models seem to be more sensitive to the number of latent features on the MSN log. When k_f is small, for example, $5 < k_f < 10$, the MRR jumps (see Figure 6(a)) are easily observed, especially for GQS_{MSR+LQ} and GQS_{MPC+LQ}. Regarding α-nDCG@10 on the MSN log, similar findings can be observed except that all GQS models arrive at a stable level of performance much earlier (when $k_f = 8$) than on the AOL log (when $k_f = 10$). From the results shown in Figures 5 and 6, we conclude



(a) Performance in terms of MRR. **(b) Performance in terms of α-nDCG@10.**

Fig. 7. D-QAC performance of all discussed models, tested on the AOL log, in terms of MRR (left) and α-nDCG@10 (right) when more (or less) QAC candidates are returned. Note: the scales are different.



(a) Performance in terms of MRR. **(b) Performance in terms of α-nDCG@10.**

Fig. 8. D-QAC performance of all discussed models, tested on the MSN log, in terms of MRR (left) and α-nDCG@10 (right) when more (or less) QAC candidates are returned. Note: the scales are different.

that our GQS models are robust and not sensitive to the number of latent features k_f when it is "large enough," for example, $k_f > 10$.

5.5.3. Zooming in on the Cutoff N. Finally, we examine the performance of our GQS models and the baseline, QD-QCR, when less (or more) QAC candidates are finally returned by setting the cutoff $N = 5$ (or $N = 20$). We plot the results in terms of MRR and α-nDCG@N scores ($N = 5, 10, 20$) in Figures 7 and 8, tested on the AOL log and the MSN log, respectively.[10]

As shown in Figures 7 and 8, for all five models on both logs, the overall performance in terms of MRR increases when more QAC candidates are initially returned for reranking, that is, when N becomes larger. A larger value of N simply increases the probability of including the ground truth in the QAC list. In particular, on both the AOL and MSN log, these models report competitive MRR scores when $N = 5$. Moreover, the MRR improvements realized by our GQS models over the baseline are further magnified as N goes up. For instance, on the AOL log, GQS_{MSR+LQ} results in a 1.27%

[10]The results for $N = 10$ were already partially reported in Table VI.

MRR improvement over the baseline at $N = 5$, a 2.24% improvement at $N = 10$, and a 3.72% improvement at $N = 20$. With respect to query diversification, the improvements of the GQS models are more obvious in terms of α-nDCG@N ($N = 5$, 10, and 20) than MRR, as indicated by the relative improvements over the baseline. For instance, at cutoff $N = 20$, GQS_{MSR+LQ} shows a 4.43% improvement over the baseline in terms of α-nDCG@20. This may be because not too much redundant queries can be found among the top 5 candidates in the list of QAC candidates; thus, it is difficult to make significant improvements over the baseline at cutoff $N = 5$. However, as more candidates are returned, more query redundancy is introduced into the list of QAC candidates, and it becomes easier for the GQS models to improve over the baseline. Therefore, based on our findings from Figures 7 and 8, we conclude that, compared to the baseline, the advantages of our GQS models over the baseline become more prominent when more QAC candidates are returned.

6. CONCLUSION

In this article, we have proposed the challenge of D-QAC. We believe that this can help search engine designers, especially in settings with a limited number of QAC candidates. We have proposed a GQS model to address the QAC diversification task and use the ODP taxonomy to identify aspects of URLs, based on which we then assign query aspects via clickthrough data derived from query logs. The problems of data sparsity and cold start in traditional recommendation systems are overcome by incorporating a BPMF approach and determining the semantically most closely related query using word2vec, respectively.

We have experimentally investigated the D-QAC performance of our GQS models under various settings. Our results show that the GQS model performs best when starting with the semantically most closely related QAC candidate and using only the last preceding query in a session as the search context. This finding indicates that query aspects are commonly shared by successive queries in a session and may be changed, especially in long sessions with multiple queries. In addition, we have found that our GQS model performs better when more QAC candidates are fed to it.

As future work, we plan to move our models to other datasets, for which the ground truth of query aspects is to be generated by humans rather than the automatically generated clickthrough data used in this article. In addition, it would be interesting to introduce other scenarios to deal with the query cold-start problem, as this may be helpful to obtain the query distribution over aspects.

Furthermore, since we only consider query aspects as they are expressed, either explicitly or implicitly, by current search popularity or previously submitted queries, it would be interesting to further collect users' long-term search history to enhance the performance of D-QAC, thereby personalizing D-QAC, which can help narrow the space of query topics, that is, adjust the amount of diversification, by removing nonrelevant queries, but can still promote relevant queries as well as diversify QAC completions.

Finally, Li et al. [2014] adopt a click model [Chuklin et al. 2015] to shed light on QAC user interactions. This model is motivated by the fact that users frequently skip query completion lists even though such lists contain the final submitted query because the intended query is not ranked at the top positions of the list. What is the relation between such skips and the degree to which a list of query completions is diversified? Can the model incorporate diversity, for example, similar to Chuklin et al. [2013]?

ACKNOWLEDGMENT

We would like to thank our anonymous reviewers for their helpful comments and valuable suggestions.

REFERENCES

Rakesh Agrawal, Sreenivas Gollapudi, Alan Halverson, and Samuel Ieong. 2009. Diversifying search results. In *WSDM'09*. ACM, New York, NY, 5–14.

Kevin Bache, David Newman, and Padhraic Smyth. 2013. Text-based measures of document diversity. In *KDD'13*. ACM, New York, NY, 23–31.

Ziv Bar-Yossef and Naama Kraus. 2011. Context-sensitive query auto-completion. In *WWW'11*. ACM, New York, NY, 107–116.

Paul N. Bennett, Ryen W. White, Wei Chu, Susan T. Dumais, Peter Bailey, Fedor Borisyuk, and Xiaoyuan Cui. 2012. Modeling the impact of short- and long-term behavior on search personalization. In *SIGIR'12*. ACM, New York, NY, 185–194.

Danushka Bollegala, Yutaka Matsuo, and Mitsuru Ishizuka. 2007. Measuring semantic similarity between words using web search engines. In *WWW'07*. ACM, New York, NY, 757–766.

Fei Cai and Maarten de Rijke. 2016a. Learning from homologous queries and semantically related terms for query auto completion. *Information Processing and Management* 52, 4, 628–643.

Fei Cai and Maarten de Rijke. 2016b. Query auto completion in information retrieval. *Foundations and Trends in Information Retrieval* Submitted.

Fei Cai and Maarten de Rijke. 2016c. Selectively personalizing query auto-completion. In *SIGIR'16*. ACM, New York, NY, To appear.

Fei Cai, Shangsong Liang, and Maarten de Rijke. 2014a. Personalized document re-ranking based on Bayesian probabilistic matrix factorization. In *SIGIR'14*. ACM, New York, NY, 835–838.

Fei Cai, Shangsong Liang, and Maarten de Rijke. 2014b. Time-sensitive personalized query auto-completion. In *CIKM'14*. ACM, New York, NY, 1599–1608.

Fei Cai, Shangsong Liang, and Maarten de Rijke. 2016a. Prefix-adaptive and time-sensitive personalized query auto completion. *IEEE Transactions on Knowledge and Data Engineering* To appear.

Fei Cai, Shuaiqiang Wang, and Maarten de Rijke. 2016b. Behavior-based personalization in web search. *Journal of the Association for Information Science and Technology* To appear.

Jaime Carbonell and Jade Goldstein. 1998. The use of MMR, diversity-based reranking for reordering documents and producing summaries. In *SIGIR'98*. ACM, New York, NY, 335–336.

Olivier Chapelle, Thorsten Joachims, Filip Radlinski, and Yisong Yue. 2012. Large-scale validation and analysis of interleaved search evaluation. *ACM Transactions on Information Systems* 30, 1, 6:1–6:41.

Olivier Chapelle, Donald Metzler, Ya Zhang, and Pierre Grinspan. 2009. Expected reciprocal rank for graded relevance. In *CIKM'09*. ACM, New York, NY, 621–630.

Harr Chen and David R. Karger. 2006. Less is more: Probabilistic models for retrieving fewer relevant documents. In *SIGIR'06*. ACM, New York, NY, 429–436.

Steve Chien and Nicole Immorlica. 2005. Semantic similarity between search engine queries using temporal correlation. In *WWW'05*. ACM, New York, NY, 2–11.

Aleksandr Chuklin, Ilya Markov, and Maarten de Rijke. 2015. *Click Models for Web Search*. Morgan & Claypool Publishers, San Francisco, CA.

Aleksandr Chuklin, Pavel Serdyukov, and Maarten de Rijke. 2013. Using intent information to model user behavior in diversified search. In *ECIR'13*. Springer, Berlin, 1–13.

Charles L. A. Clarke, Maheedhar Kolla, Gordon V. Cormack, Olga Vechtomova, Azin Ashkan, Stefan Büttcher, and Ian MacKinnon. 2008. Novelty and diversity in information retrieval evaluation. In *SIGIR'08*. ACM, New York, NY, 659–666.

Charles L. Clarke, Maheedhar Kolla, and Olga Vechtomova. 2009. An effectiveness measure for ambiguous and underspecified queries. In *ICTIR'09*. Springer, Berlin, 188–199.

Kevyn Collins-Thompson, Paul Bennett, Charles L. A. Clarke, and Ellen M. Voorhees. 2013. TREC 2013 Web Track overview. In *TREC'13*. Springer, Berlin, 1–15.

Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. 2009. *Introduction to Algorithms*. MIT Press, Cambridge, MA.

Nick Craswell, Rosie Jones, Georges Dupret, and Evelyne Viegas (Eds.). 2009. *WSCD'09: Proceedings 2009 Workshop on Web Search Click Data*. ACM.

Van Dang and Bruce W. Croft. 2013. Term level search result diversification. In *SIGIR'13*. ACM, New York, NY, 603–612.

Van Dang and W. Bruce Croft. 2012. Diversity by proportionality: An election-based approach to search result diversification. In *SIGIR'12*. ACM, New York, NY, 65–74.

Arthur P. Dempster, Nan M. Laird, and Donald B. Rubin. 1977. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society: Series B* 39, 1, 1–38.

João Gama, Indrė Žliobaitė, Albert Bifet, Mykola Pechenizkiy, and Abdelhamid Bouchachia. 2014. A survey on concept drift adaptation. *Computing Surveys* 46, 4, 44:1–44:37.

Jiafeng Guo, Xueqi Cheng, Gu Xu, and Xiaofei Zhu. 2011. Intent-aware query similarity. In *CIKM'11*. ACM, New York, NY, 259–268.

Jiyin He, Edgar Meij, and Maarten de Rijke. 2011. Result diversification based on query-specific cluster ranking. *Journal of the Association for Information Science and Technology* 62, 3, 550–571.

Katja Hofmann, Bhaskar Mitra, Filip Radlinski, and Milad Shokouhi. 2014. An eye-tracking study of user interactions with query auto completion. In *CIKM'14*. ACM, New York, NY, 549–558.

Kalervo Järvelin and Jaana Kekäläinen. 2002. Cumulated gain-based evaluation of IR techniques. *ACM Transactions on Information Systems* 20, 4, 422–446.

Di Jiang, Kenneth Wai-Ting Leung, Jan Vosecky, and Wilfred Ng. 2009. Web query recommendation via sequential query prediction. In *ICDE'09*. IEEE Computer Society, Washington, DC, 1443–1454.

Jyun-Yu Jiang, Yen-Yu Ke, Pao-Yu Chien, and Pu-Jen Cheng. 2014. Learning user reformulation behavior for query auto-completion. In *SIGIR'14*. ACM, New York, NY, USA, 445–454.

Thorsten Joachims. 2002. Optimizing search engines using clickthrough data. In *KDD'02*. ACM, New York, NY, 133–142.

Eugene Kharitonov, Craig Macdonald, Pavel Serdyukov, and Iadh Ounis. 2013. Intent models for contextualising and diversifying query suggestions. In *CIKM'13*. ACM, New York, NY, 2303–2308.

Youngho Kim and W. Bruce Croft. 2014. Diversifying query suggestions based on query documents. In *SIGIR'14*. ACM, New York, NY, 891–894.

Liangda Li, Hongbo Deng, Anlei Dong, Yi Chang, Hongyuan Zha, and Ricardo Baeza-Yates. 2015. Analyzing user's sequential behavior in query auto-completion via Markov processes. In *SIGIR'15*. ACM, New York, NY, 123–132.

Ruirui Li, Ben Kao, Bin Bi, Reynold Cheng, and Eric Lo. 2012. DQR: A probabilistic approach to diversified query recommendation. In *CIKM'12*. ACM, New York, NY, 16–25.

Yanen Li, Anlei Dong, Hongning Wang, Hongbo Deng, Yi Chang, and ChengXiang Zhai. 2014. A two-dimensional click model for query auto-completion. In *SIGIR'14*. ACM, New York, NY, 455–464.

Shangsong Liang, Zhaochun Ren, and Maarten de Rijke. 2014a. Fusion helps diversification. In *SIGIR'14*. ACM, New York, NY, 303–312.

Shangsong Liang, Zhaochun Ren, and Maarten de Rijke. 2014b. Personalized search result diversification via structured learning. In *KDD'14*. ACM, New York, NY, 751–760.

Zhen Liao, Daxin Jiang, Enhong Chen, Jian Pei, Huanhuan Cao, and Hang Li. 2011. Mining concept sequences from large-scale search logs for context-aware query suggestion. *ACM Transactions on Intelligent Systems and Technology* 3, 1, Article 17.

Hao Ma, Michael R. Lyu, and Irwin King. 2010. Diversifying query suggestion results. In *AAAI'10*. AAAI Press, New York, NY, 1399–1404.

Tomas Mikolov, Kai Chen, Greg S. Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space. In *Proceedings of Workshop at ICLR*. MIT Press, Cambridge, MA, 1–13.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S. Corrado, and Jeffrey Dean. 2013b. Distributed representations of words and phrases and their compositionality. In *NIPS 26*. MIT Press, Cambridge, MA, 3111–3119.

Bhaskar Mitra. 2015. Exploring session context using distributed representations of queries and reformulations. In *SIGIR'15*. ACM, New York, NY, USA, 3–12.

Bhaskar Mitra, Milad Shokouhi, Filip Radlinski, and Katja Hofmann. 2014. On user interactions with query auto-completion. In *SIGIR'14*. ACM, New York, NY, 1055–1058.

Radford M. Neal. 1993. *Probabilistic Inference Using Markov Chain Monte Carlo Methods*. Technical Report CRG-TR-93-1. Department of Computer Science, University of Toronto, Toronto, ON.

Greg Pass, Abdur Chowdhury, and Cayley Torgeson. 2006. A picture of search. In *InfoScale'06*. ACM, New York, NY, 1–7.

Filip Radlinski and Susan Dumais. 2006. Improving personalized web search using result diversification. In *SIGIR'06*. ACM, New York, NY, 691–692.

Ruslan Salakhutdinov and Andriy Mnih. 2008a. Bayesian probabilistic matrix factorization using Markov chain Monte Carlo. In *ICML'08*. ACM, New York, NY, 880–887.

Ruslan Salakhutdinov and Andriy Mnih. 2008b. Probabilistic matrix factorization. In *NIPS 20*. MIT Press, Cambridge, MA, 1–8.

Rodrygo L. T. Santos, Craig Macdonald, and Iadh Ounis. 2010a. Exploiting query reformulations for web search result diversification. In *WWW'10*. ACM, New York, NY, 881–890.

Rodrygo L. T. Santos, Craig Macdonald, and Iadh Ounis. 2010b. Selectively diversifying web search results. In *CIKM'10*. ACM, New York, NY, 1179–1188.

Rodrygo L. T. Santos, Craig Macdonald, and Iadh Ounis. 2011. Intent-aware search result diversification. In *SIGIR'11*. ACM, New York, NY, 595–604.

Rodrygo L. Santos, Craig Macdonald, and Iadh Ounis. 2013. Learning to rank query suggestions for adhoc and diversity search. *Information Retrieval* 16, 4, 429–451.

Milad Shokouhi. 2011. Detecting seasonal queries by time-series analysis. In *SIGIR'11*. ACM, New York, NY, 1171–1172.

Milad Shokouhi. 2013. Learning to personalize query auto-completion. In *SIGIR'13*. ACM, New York, NY, 103–112.

Milad Shokouhi and Kira Radinsky. 2012. Time-sensitive query auto-completion. In *SIGIR'12*. ACM, New York, NY, 601–610.

Yang Song, Dengyong Zhou, and Li-wei He. 2011. Post-ranking query suggestion by diversifying search results. In *SIGIR'11*. ACM, New York, NY, 815–824.

Paul Thomas and David Hawking. 2006. Evaluation by comparing result sets in context. In *CIKM'06*. ACM, New York, NY, 94–101.

David Vallet. 2011. Crowdsourced evaluation of personalization and diversification techniques in web search. In *CIR'11 Workshop (SIGIR'11)*. ACM, New York, NY, 1–6.

David Vallet and Pablo Castells. 2011. On diversifying and personalizing web search. In *SIGIR'11*. ACM, New York, NY, 1157–1158.

David Vallet and Pablo Castells. 2012. Personalized diversification of search results. In *SIGIR'12*. ACM, New York, NY, 841–850.

Saúl Vargas, Pablo Castells, and David Vallet. 2012. Explicit relevance models in intent-oriented information retrieval diversification. In *SIGIR'12*. ACM, New York, NY, 75–84.

Stewart Whiting and Joemon M. Jose. 2014. Recent and robust query auto-completion. In *WWW'14*. ACM, New York, NY, 971–982.

Cheng Xiang Zhai, William W. Cohen, and John Lafferty. 2003. Beyond independent relevance: Methods and evaluation metrics for subtopic retrieval. In *SIGIR'03*. ACM, New York, NY, 10–17.

Aston Zhang, Amit Goyal, Weize Kong, Hongbo Deng, Anlei Dong, Yi Chang, Carl A. Gunter, and Jiawei Han. 2015. adaQAC: Adaptive query auto-completion via implicit negative feedback. In *SIGIR'15*. ACM, New York, NY, 143–152.

Received June 2015; revised March 2016; accepted March 2016