Information Processing and Management 53 (2017) 666–683

Contents lists available at ScienceDirect

Information Processing and Management

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

Investigating queries and search failures in academic search

Xinyi Li[a,*], Bob J.A. Schijvenaars[b], Maarten de Rijke[a]

[a] Informatics Institute, University of Amsterdam, Science Park 904, 1098 XH Amsterdam, The Netherlands
[b] Elsevier B.V., Radarweg 29, 1043 NX Amsterdam, The Netherlands

ARTICLE INFO

Article history:
Received 12 September 2016
Revised 25 November 2016
Accepted 11 January 2017

ABSTRACT

Academic search concerns the retrieval and profiling of information objects in the domain of academic research. In this paper we reveal important observations of academic search queries, and provide an algorithmic solution to address a type of failure during search sessions: null queries. We start by providing a general characterization of academic search queries, by analyzing a large-scale transaction log of a leading academic search engine. Unlike previous small-scale analyses of academic search queries, we find important differences with query characteristics known from web search. E.g., in academic search there is a substantially bigger proportion of entity queries, and a heavier tail in query length distribution. We then focus on search failures and, in particular, on null queries that lead to an empty search engine result page, on null sessions that contain such null queries, and on users who are prone to issue null queries. In academic search approximately 1 in 10 queries is a null query, and 25% of the sessions contain a null query. They appear in different types of search sessions, and prevent users from achieving their search goal. To address the high rate of null queries in academic search, we consider the task of providing query suggestions. Specifically we focus on a highly frequent query type: non-boolean informational queries. To this end we need to overcome query sparsity and make effective use of session information.

We find that using entities helps to surface more relevant query suggestions in the face of query sparsity. We also find that query suggestions should be conditioned on the type of session in which they are offered to be more effective. After casting the session classification problem as a multi-label classification problem, we generate session-conditional query suggestions based on predicted session type. We find that this session-conditional method leads to significant improvements over a generic query suggestion method. Personalization yields very little further improvements over session-conditional query suggestions.

1. Introduction

Academic search systems existed long before the advent of the World Wide Web (Lindberg, 2000). Early systems such as MEDLINE were made for librarian use and supported only preprogrammed queries and a limited number of simultaneous users in a non-interactive way. Modern academic search engines have taken completely different forms, with support of ad hoc queries, which users themselves enter in a search box, made for public use, and with support for many online users in an interactive search environment. In this paper we look at academic search in this modern form. In recent years there

* Corresponding author.
E-mail addresses: x.li@uva.nl (X. Li), b.schijvenaars@elsevier.com (B.J.A. Schijvenaars), derijke@uva.nl (M. de Rijke).

http://dx.doi.org/10.1016/j.ipm.2017.01.005
0306-4573/

have been numerous studies on a range of aspects of academic search (Hemminger, Lu, Vaughan, & Adams, 2007; Niu et al., 2010; Pontis & Blandford, 2015; Pontis, Blandford, Greifeneder, Attalla, & Neal, 2015). These studies reveal the information-seeking behavior of researchers by conducting surveys or user studies on a relatively small sample of researchers. They point out that academic search engines have become the primary portal for researchers to gain information. Surprisingly, despite the widespread usage of academic search environments, very little is known about the actual search behavior of users of academic search engines that is based on a large-scale transaction log analysis. Query log analysis is known to be a valuable source for improving search systems (Silvestri, 2010), whether it concerns query suggestions, learning to rank or personalization. Numerous analyses have been conducted on commercial web search engines, such as Microsoft Bing, Yahoo and AOL (Beitzel, Jensen, Chowdhury, Frieder, & Grossman, 2007; Bendersky & Croft, 2009; Wedig & Madani, 2006) as well as on verticals such as blog search (Mishne & de Rijke, 2006) or people search (Weerkamp et al., 2011). However, little work has been done on academic search engines. A small-scale analysis of a transaction log of ScienceDirect, a leading academic search engine for multiple disciplines, reveals basic statistics such as page views, article downloads, and journal access frequencies (Ke, Kwakkelaar, Tai, & Chen, 2002); the study was conducted 14 years ago on a district scale, based on only 0.42 million queries, and it does not differentiate between individual users due to limitations of the logging system at the time, and hence it does not inform us about users' behavior during search sessions. In this paper, we analyze a large, global-scale academic search transaction log containing over 39 million queries in which we are able to track and analyze the behavior of individual users.

Our analysis of user behavior comes in two stages. First, we set the scene by providing a general descriptive analysis of academic queries and highlighting the differences between academic search and web search. Then we zoom in on search failures, and in particular on so-called null queries for which the search engine produces an empty result page as defined in Gupta and Bendersky (2015). We study null queries from three angles: queries, sessions and users. Compared to web search, where they make up less than 2% of all queries (Altingovde et al., 2012), null queries appear more frequently in academic search. A recent study shows that the null query rate is 15% in PubMed, a popular biomedical academic search engine (Dogan, Murray, Névéol, & Lu, 2009). As we will see below, in ScienceDirect the null query rate is over 10%. When checking whether planned research is novel, an empty SERP (search engine result page) may be a desirable outcome, but in general it is an outcome that a search engine wants to avoid, thus motivating the development of effective strategies for dealing with null queries and to guide users, either automatically or interactively, to consider alternative queries.

To address the failure phenomena uncovered by our log analysis, we consider a query suggestion task. Query suggestion is a feature in modern search that improves the search experience by providing query recommendations. Previous work on query suggestion comes in several flavors, from suggestions based on syntactic variations to suggestions based on semantic relatedness or behavioral similarity. Techniques based on behavioral similarity perform well for so-called head queries, that occur frequently. The long tail of queries is more challenging for query suggestions, since most of these queries are rare. We contribute an approach to query suggestion for null queries in academic search. We discover that there are different types of null queries and specifically target a highly frequent type: non-boolean informational queries.

In particular, we make use of our analysis of academic search queries by considering entities. This helps us overcome the sparsity issue in academic search queries. Moreover, using automatically predicted types of sessions, we condition our query suggestions on the type of null session by reranking different types of query suggestion candidates. Query suggestion candidates are first generated using graph-based models that incorporate different kinds of relations: links between queries and entities, transitions between queries and transitions between entities. Then they are reranked based on the predicted session types. We also build a personalized model by considering users' preferences of entities.

In this paper, we address the following research objectives:

Research Objective 1: Characterize academic search queries and identify their differences from web search queries.

For Research Objective 1, we perform a transaction log analysis on an academic search engine. We examine the query-level characteristics of academic search, namely query length and query types. The observational results demonstrate clear differences between academic search queries and web search queries.

Research Objective 2: Characterize the characteristics of null queries in academic search.

To investigate Research Objective 2, we first look at null queries, then at null sessions and users to understand how null queries happen. The insights enable us to devise a method to cope with null queries.

Research Objective 3: Examine whether query, session and user information helps improve query suggestions for null queries.

For Research Objective 3, we first define the task of using query suggestions to address null queries. Then we use the obtained observational insights (entity-richness, different session types) to improve the ranking of query suggestions.

Our novel contribution of this paper is three-fold. First is the analysis of academic search queries and their differences from web search. Our findings based on a large-scale transaction log analysis give more insights into query contents and

Table 1
Query length statistics in word count. The AOL query log covers over 650,000 users in a 3-month period. The AOL log statistics come from (Arampatzis & Kamps, 2008).

Category	#N	Min	Max	Mean	Median
AOL	21M	1	245	2.34	2
Sciencedirect	39M	1	419	3.77	3

user behavior in sessions than previous small-scale analyses (Han, Jeong, & Wolfram, 2014; Jones, Cunningham, McNab, & Boddie, 2000; Ke et al., 2002). Second, we drill down on search failures in academic search and thoroughly analyze problems around null queries. Third, we propose a query suggestion method that addresses null queries when they occur. Our query-conditional method uses entities to overcome the severe sparsity in academic search queries. Moreover, our session-conditional query suggestion method results in significant improvements over state-of-the-art query suggestion baselines. We find that a personalized model that infers user preferences of entities further improves query suggestion performance.

Our main findings can be summarized as follows.

1. Academic search differs from web search in query properties, namely in length distribution, query types and noticeable entity richness.
2. Frequent null queries are a unique phenomenon in academic search. They happen more frequently compared to web search, and they happen in various types of sessions, such as refining and exploratory sessions.
3. Query-conditional and session-conditional query suggestion methods improve over methods that do not consider entities in queries and session types, respectively. However, when to apply personalization effectively remains an open problem.

In Section 2 we describe the dataset and query properties that form the basis for the remainder of the paper. In Section 3 we focus on failures in academic search: null queries, null sessions and users who frequently issue them. In Section 4 we describe our model of query suggestions to address null queries. In Section 5 we discuss personalization. In Section 6 we discuss related work. We present our conclusions in Section 7.

2. Dataset

As we will see in the related work section (Section 6), previous analyses of academic search logs are limited in scale. We are particularly interested in studying user behavior from large-scale log analysis, which is why we conduct a new transaction log analysis.

We study a 5-month query log from the ScienceDirect search engine collected from September 28, 2014 to March 5, 2015. Due to institution-authorization from ScienceDirect, users in a certain IP range are able to access the search engine from shared devices (e.g., library computers), and they may share the same session ID and user ID in the transaction log. Moreover, many institutions have proxies or firewalls whose IP is recorded instead of that of the terminal device. Therefore, it is difficult to differentiate these IP-users in the log. It is only safe to assume a one-to-one mapping between IDs and users when users log in, or when they access the search engine from outside the institution; we refer to such users as *non-IP users*. They contribute about one third of the total query traffic.

2.1. Queries

In this section we describe the query characteristics as identified from the transaction log. Table 1 contrasts the statistics of academic queries with those of web queries. A marked difference between academic search and web search is the query length, where academic queries are on average 1.4 words longer than web queries. Fig. 1 shows that the distribution of academic queries follows a power law, which is similar to web search queries (Bailey, White, Liu, & Kumaran, 2010) but featured with a "bigger tail." The query length statistics in our study differ from the much smaller scale log analysis in Ke et al. (2002), e.g., the average length of our queries is 1.5 words longer than their findings.

The verbosity of academic search queries yields multiple challenges for the search engine. One of them is sparsity, which makes it difficult to generate query suggestions for rare queries (Bonchi, Perego, Silvestri, Vahabi, & Venturini, 2012).

Apart from query length, academic queries also differ from web queries in their content and intent. Below we exemplify three typical query intents known from web search (Jansen, Booth, & Spink, 2008) in the academic search setting:

Navigational queries A "navigational query" in academic search guides the user to a certain publication (identified by special operators such as DOI, ISBN or a "title" operator). E.g., the query "DOI(10.1111/ jcmm.12096)" seeks to locate a specific publication. These queries are sometimes referred to as known-item queries (Ogilvie & Callan, 2003). Using automatic identification with the special operators, these queries are found to make up 7.6% of all queries.

Transactional queries These are queries that directly aim to retrieve academic information resources, e.g., a PDF file. For instance, "oil paper filter design pdf" and "download journal pressure sensor." The proportion of this type of query is only 0.5%.

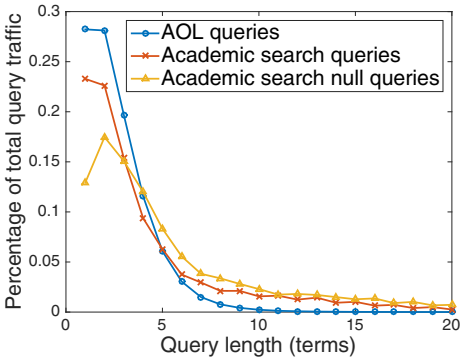


Fig. 1. Query length distribution of web search queries, academic search queries and academic search null queries.

Informational queries Queries that seek, refine or explore research topics act as "informational queries." For example, "vitamin C and cosmetic" aims to retrieve information resources on the topic "vitamin C and cosmetic." We find that the majority of queries belong to this type, making up 92.3% of all the queries. However, we notice that the boundary between query types is not strict. For instance, an "informational query" may first lead to information resources, then a user might not be content with the acquired information and further downloads a resource (e.g., a PDF file), thereby transforming the query into a "transactional query." And sometimes, the user might type in a few key words with the purpose of locating a specific paper. These queries may appear to be informational queries but act as navigational queries in effect. Thus it can be difficult to distinguish these navigational queries accurately, as is also shown by (Khabsa, Wu, & Giles, 2016).

Besides the three familiar categories just given, which correspond to categories commonly used to classify web queries, we also identify the following types of query:

Entity queries Academic search differs from web search in the proportion of queries that contain named entities. Entities in academic queries are identified using a controlled vocabulary derived from key concepts in papers, author names etc. They are not exactly the same as the entities in web search (people, places or things), but they both represent something that is existent. In web search the percentage of entity queries varies from 43% up to 70% (Guo, Cheng, Xu, & Zhu, 2011; Lin, Pantel, Gamon, Kannan, & Fuxman, 2012). In academic search, the proportion of queries that contain entities is 92.37%, far more than in web search queries.[1]

Boolean operator queries These queries are often issued by advanced searchers who like to use boolean operators to perform precise match. Specifically, AND, OR and NOT operators (in upper case) are applied to address complex search intents. These queries constitute 8.2% of the total traffic.

Now that we have provided the basic descriptive statistics of our dataset, we can zoom in on the phenomenon that is at the center of our interest in this paper: search failure.

3. Search failure

In this section we zoom in on search failures, in particular on null queries, whose high frequency is peculiar for our academic search setting. We analyze null queries and the sessions that contain them (null sessions). We also analyze users who frequently issue null queries.

3.1. Null queries

Null queries are queries that lead to an empty search engine result page; they present a severe challenge to the search engine. In academic search, null queries appear four times more often than in web search. Specifically, in our academic search setting the null query rate is 10.3%, much higher than the 2% reported for web search (Altingovde et al., 2012). Altingovde et al. (2012) show that null queries in web search contain un-indexed URIs (unique resource identifier), or they are merely meaningless queries. In academic search the situation of null queries is different. For instance, Fig. 1 shows that null queries tend to be longer than normal queries, which brings in more sparsity in the queries. And at 19.6%, the proportion of boolean operator queries amongst null queries is higher than amongst all queries (8.2%). The vast majority of null queries (80.1%) are non-boolean informational queries.

[1] In web search it is common to apply entity disambiguation to queries, and link entities to, e.g., Wikipedia, DBpedia and Freebase (Blanco, Ottaviano, & Meij, 2015; Meij, Bron, Hollink, Huurnink, & de Rijke, 2011). In academic search, queries are mostly disambiguated (e.g., in most cases technical terms and names are unique), therefore we simplify the entity linking to matching query terms to a thesaurus. To identify entities, we use a human-calibrated thesaurus derived from over 13 million academic papers. The thesaurus is built and maintained by domain experts at Elsevier who review the papers manually. We apply left-most longest matching to extract entities in queries, which provides a lower bound on the number of queries containing entities. There may still be a small portion of them that need disambiguation (Sun, Kaur, Possamai, & Menczer, 2013), but elaborating on this task is beyond the scope of this paper.

3.2. Null sessions

Null sessions constitute 25.0% of all sessions, indicating the frequent appearances of null queries. Given the high frequency of null queries, we expect that they feature in diverse search contexts. Put differently, if we define a null session to be a session in which a null query occurs, then what types of null sessions occur in the logs? We hypothesize that understanding null sessions should help us to identify a solution to address null queries. To address the question, we annotate null sessions, using 2 professionals with years of experience in academic search engines and a researcher in information retrieval; in total, 300 sessions are annotated.

To begin, null session types were identified through an initial pass through the data by our annotators. This gave rise to a total of 7 null session types; a "stereotypical example" was provided for every null session type. Subsequently, our annotators annotated all null sessions with one or more session types; the annotation is non-trivial as each session may have several types. Below, we provide definitions and examples for the types of session that occur; the null query is marked in ***bold italics***.

(1) Refining sessions. Failure in refining sessions happens when users are searching around one goal. After a failure happens, the user reformulates the query by rephrasing a synonym or correcting spelling mistakes etc, as shown in the following example:

04Mar2015:15:42:03.203	Query	pid acel
04Mar2015:15:42:17.041	***Query***	***pid acelerometer***
04Mar2015:15:42:38.643	Query	pid accelerometer
04Mar2015:15:42:54.911	Click	shorturl=/scie...pii/S0141029614005793

(2) Generalizing sessions. Some queries contain too many entities and cause the search engine to return zero results. Users in this setting may drop terms to obtain some results. We call this a generalizing session as shown in the following example:

03Mar2015:23:36:45.022	***Query***	***Leaf blast (Magnaporthe oryzae)***
03Mar2015:23:37:35.456	Query	Leaf blast
03Mar2015:23:37:43.372	Click	shorturl=/scie...pii/S1049964411001009

(3) Exploring sessions. Users often formulate queries around a pivot entity. They may encounter a failure in their exploration but continue to explore. In the example below, the user encounters a failure when searching around "composite beams," and keeps reformulating the query until finally obtaining a click:

01Mar2015:08:55:07.291	Query	Determination of rotation capacity...and composite beams
01Mar2015:08:55:46.910	Click	shorturl=/scie...pii/S0143974X95939000
01Mar2015:09:03:18.931	***Query***	***a stady on elastic plastic...of composite beams***
01Mar2015:09:03:37.627	Query	a study on elastic plastic...of composite beams
01Mar2015:09:04:19.937	Query	behavior of composite beam...dynamic testing
01Mar2015:09:05:29.587	Query	Inelastic analysis of steel frames with composite beams
01Mar2015:09:06:48.446	Click	shorturl=/scie...pii/S0141029613004379

(4) Item search session. Users sometimes search with a very specific goal, e.g., looking for an article or a book with a unique ID (e.g., DOI). They issue a navigational or transactional query, and the search engine returns no results if the desired item is not indexed, which may be due to a mal-formed query, e.g. one that occurs with an error in DOI. Or when they query is correct, the failure may be due to the fact that the item is not in the database. For such failures, the search engine is unable to satisfy the user's intent. However, if the item is not indexed online, this may not be a "failure" as the user has confirmation that the item is not online. In the following example the user searches a book using a title identifier and finds no results for the first query.

01Mar2015:12:34:14.148	***Query***	***(ttl(Identification of micro-RNA...))***
01Mar2015:12:35:01.556	Query	(ttl(end-stage heart failure...))
01Mar2015:12:39:39.638	Query	(DOI(10.1111/jcmm.12096))

(5) Expanding sessions. It may happen that users add terms when the current query returns no results. This looks like query specialization but it is not. In query specialization the first query is usually a successful one with results, despite being too general to the user. Here it is a null query. Adding terms might increase the possibility for the search engine to have a matching term in the query. In the example below, the user initially enters a query that causes a failure, and then adds a term to obtain some results.

Table 2
Annotation agreement; annotators are denoted as 0, 1, and 2.

	0+1	0+2	1+2	0+1+2
All sessions in the sample				
Percentage Agreement	0.52	0.35	0.32	0.40
Kappa Agreement	0.43	0.23	0.20	0.29
Observed Disagreement	0.48	0.65	0.68	0.60
Expected Disagreement	0.85	0.86	0.85	0.86
Leaving out sessions of type dropout or anomaly				
Percentage Agreement	0.57	0.62	0.65	0.61
Kappa Agreement	0.39	0.45	0.53	0.46
Observed Disagreement	0.43	0.38	0.35	0.39
Expected Disagreement	0.71	0.69	0.76	0.72

05Mar2015:13:51:45.227	***Query***	***Ulocladium***
05Mar2015:13:52:25.827	Query	ulocladium helioanthus

(6) Dropout. Dropout sessions are sessions where users abandon search when failure happens, as shown in the example below.

04Mar2015:08:52:02.891	***Query***	***ROS1 Rearrangements ...Lung Cancers***

(7) Anomaly sessions. Some users enter non-ASCII characters that are not supported by the search engine, or other meaningless queries that cause a failure; 20% of the sessions that we annotate belong to this type. It is useless to provide query suggestions for these sessions so we remove them from further consideration in our query suggestion work.

Table 2(top) lists the agreement scores for the null session annotation task. We consider Percentage Agreement (Artstein & Poesio, 2008), Kappa Agreement (Fleiss, 1971), Observed Disagreement and Expected Disagreement (Krippendorff, 2004). The Percentage Agreement divides the number of agreements by the item count, while the rest assume the same probability distribution for all raters. Note that this is a multi-label annotation task (one session may have several labels), therefore it is not easy to achieve high agreement scores among the annotators. The annotation agreement is fair by Kappa values (Landis & Koch, 1977) for all types of null sessions. We also compute agreement for a subset of the null sessions where the users have subsequent actions after the null query happens, namely those that are not of type dropout (type 6) or anomaly (type 7). For this set of null sessions, the annotation agreement is moderate (Landis & Koch, 1977); see Table 2 (bottom).

Certain session labels are prevalent, e.g., refining (39%) and generalizing (26%), as it is common for users to modify or drop a query term upon a null query. Also, 26% of the sessions are item-search oriented whose failures are due to non-existent resources. Exploring (15%) and expanding session labels (13%) are found to be less frequent. And 32% of the sessions ended up as dropout sessions, as the users give up the search goal.

3.3. Users who fail frequently

Next we consider users who frequently fail, that is, who frequently submit null queries: “at least 20% of their queries are null queries.” These users account for 20.1% of the users who have submitted at least one null query. We analyze these users and compare them with “normal users,” who encounter null queries less frequently.

First, we try to find out if a user’s consistency of interests affects their null query rate. To determine a user’s consistency of interests, we look at the “self-similarity score” and “cross-session similarity score” of users.

Our notion of self-similarity is defined as follows. We model users as a bag of queries, i.e., profiling every user using all the queries they issued. We measure the similarity score of each user in a pairwise manner by treating each query as a bag of words and compute the mean similarity score between all the query pairs (Hassan, White, Dumais, & Wang, 2014). The similarity of queries q_i and q_j is computed as:

$$\mathrm{sim}(q_i, q_j) = |q_i \cap q_j| / (|q_i| + |q_j| - |q_i \cap q_j|),$$

where $|q_i|$ is the number of terms in query q_i, and $|q_i \cap q_j|$ is the number of common terms between q_i and q_j.

The self-similarity score reflects how consistent a user’s interests are, i.e., a higher score means many similar queries are issued by the user, and a lower score means queries are diverse. We find out that frequently failed users tend to have higher similarity scores than our normal users; see Table 3.

Similarly, we computed similarity scores between query pairs from different sessions, i.e., cross-session similarity score, and have found exactly the same tendency; see Table 4.

Next, we relate the appearance of null queries to the user’s characteristics in Fig. 2, in particular, to the user’s self-similarity score, cross-session similarity score, the number of queries, and the number of clicks. We find a medium correlation ($r = 0.46$) between the frequency of null queries and a user’s self-similarity score, and a weak correlation ($r = 0.27$) for cross-session similarity score. This suggests that null queries tend to happen to users who have more consistent search interests.

Table 3
Self-similarity score of users.

	Min	Max	Mean	Median
Frequently failed users	0	1	0.290	0.218
Normal users	0	1	0.280	0.212

Table 4
Cross-session similarity score of users with at least 3 sessions.

	Min	Max	Mean	Median
Frequently failed users	0	1	0.140	0.079
Normal users	0	1	0.132	0.077

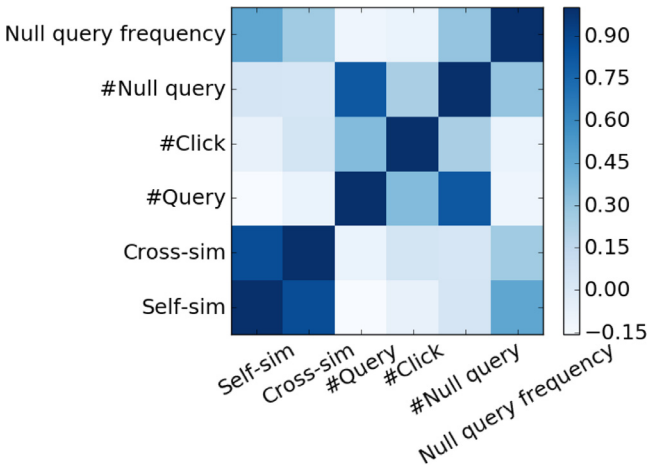


Fig. 2. Pearson correlation matrix between users' self-similarity score, cross-session similarity score, the number of queries, clicks and null queries, and null query frequency. Correlations with cross-session similarity score are calculated for users with at least 3 sessions.

There is a very weak correlation ($r = -0.10$) between the number of queries issued and the null query frequency, meaning that frequent users do not necessarily become prone to failures. As to clicks, the correlation with the number of null queries is weak ($r = 0.23$).

3.4. Upshot

Our analysis so far yields some suggestions for how to address null queries. For instance, the entity-richness of academic search queries (i.e., the high percentage of queries that contain an entity) might help circumvent the sparsity problem in null queries. And the different null session types might help to tune query suggestions based on session type. Users' entity preferences also provide a hint for personalization.

4. Query suggestions for null sessions

In this section we address the task of providing query suggestions for null queries. We combine this task with findings from the log analysis in previous sections and consider two variations: a query suggestion method that uses information about entities found in queries and one that uses information in sessions.

4.1. Evaluation and data

In the context of query suggestions, evaluation is aimed at measuring how well the provided suggestion is able to help users continue the session upon entering a failed query. Therefore, it is preferable to use an evaluation scheme that considers the actual user behavior in search sessions, over schemes that do not model user behavior (Kim, Seo, Croft, & Smith, 2014). We adopt a method that is similar to one used in Vahabi, Ackerman, Loker, Baeza-Yates, and Lopez-Ortiz (2013), but with some variations to suit our setup. We consider sessions that contain a cascade of queries q_1, q_2, ..., q_n, where q_1 is the first null query. If there is no click after q_1 to q_{n-1}, and there is at least one click after q_n, then we treat q_n as a successful query suggestion. We evaluate whether our query suggestion algorithm can predict q_n, given the null query q_1. In this way, we are simulating the real use case that a query suggestion helps a user upon entering a null query.

Since there is only one relevant query in the evaluation (the one that leads to a click in the log), it is proper to use Success Rate (SR) as a metric (Vahabi et al., 2013). Specifically, given a query q, a ranked list of query suggestions $S(q)$, and the successful query q_n, the success rate is 1 if $q_n \in S(q)$, and 0 otherwise. We use cutoffs at 1, 3, 5, 10 because usually only a limited number of query suggestions can be displayed in a search engine's user interface. We report the mean SR scores averaged over all test cases.

We select test cases from the last 5 days in the log: sessions are chosen that contain a null query and a successful query with a click, both of which have appeared earlier in the log. We do not recommend queries that we have never seen before, instead we look at existent queries that are in the log. Moreover, we focus on sessions that contain non-boolean informational queries, for two reasons:

(1) Item search queries' failures are often due to un-indexed items in the database; (2) Boolean queries (with logic operators in uppercase) have been discussed earlier (Kim, Seo, & Croft, 2011) (a simple method to address boolean queries is to relax the logic relations, e.g., changing "AND" to "OR" may surface more search results). In total we have 310 sessions for evaluation.

We test for statistical significance of observed differences in SR using a two-tailed paired t-test and denote significant differences using $^{\blacktriangle}$ for significance at $\alpha = .01$, or $^{\triangle}$ for $\alpha = 0.05$.

4.2. Query-conditional suggestions

In Section 2.1 we mentioned the frequent occurrence of entities in academic queries. Entities contain important information on query intent, which can be utilized for query suggestion. To this end we take inspiration from the *query-entity graph* (QEG) for query suggestions in search sessions. We refer to our proposed method as the *query-conditional* suggestion method, as it bases its suggestions on the query itself and the entities in it.

The original QEG method was used for recommending queries for web pages (Bordino, De Francisci Morales, Weber, & Bonchi, 2013), in which a set of query suggestions are produced given the current web page. There is a very important difference between the original setup and the modification that we consider for academic search: the input for recommendation. In the original setup, the input consists of a web page, but in our case it consists of a query. Importantly, contrary to the web page recommendation setting where a web page contains many seed entities, a query in a search session contains far fewer. To fit the characteristics of search sessions, we tailor the graph structure and the random walk on the QEG, and introduce the *mQEG*: the modified query entity graph model.

The modified query entity graph model is a triple $G_{qe} = (V, E, w)$ that satisfies the following conditions:

1. V, the set of nodes, consists of all unique queries in the query log plus all entities identified in the queries;
2. $E = V \times V$, the set of directed edges, is the union of E_{QQ} (query to query edges), E_{QE} (query to entity edges), and E_{EE} (entity to entity edges).
   (a) Here, E_{QQ} follows the definition in the QFG (Query Flow Graph) (Boldi, Bonchi, Castillo, Donato, & Vigna, 2009). Each edge E_{ij} in E_{QQ} corresponds to a query transition from q_i to q_j in the query log.
   (b) Concerning E_{QE} we deviate from the definitions in Bordino et al. (2013). E_{QE} consists of edges connecting queries and the entities extracted from them, which are bidirectional, unlike (Bordino et al., 2013). Making edges in E_{QE} bidirectional enables the random walker to visit query nodes and entity nodes alternatively, which helps to expand the limited number of entities in a query by using behavioral information (query transitions).
   (c) E_{EE} is defined by connecting entities in q_i to those in q_j if there is a query transition from q_i to q_j in E_{QQ}, the same as in Bordino et al. (2013).
3. w is the weighting function that assigns a weight to each edge in E_{ij} representing the likelihood of transitions.
   (a) For transitions in E_{QQ} we use the same weighting function as in the QFG (Boldi et al., 2009):

$$w(q_i \to q_j) = |q_i \to q_j| / |q_i|,$$

   where $|q_i \to q_j|$ is the number of occurrences of a query transition from q_i to q_j, and $|q_i|$ the total number of query transitions that start from q_i.
   (b) For entity to query transitions in E_{QE} the weight is:

$$w(e \to q) = |q| / \sum |e \to q_i|,$$

   where $|q|$ denotes the frequency of query q that contains e and $\Sigma|e \to q_i|$ the number of occurrences of all queries that contain entity e. Since edges in E_{QE} are made bidirectional, unlike (Bordino et al., 2013), we define the weight of query to entity transitions in E_{QE} as:

$$w(q \to e) = |q \to e| / \sum |q \to e_i|,$$

   where $|q \to e|$ is the number of mentions of entity e in query q, and $\Sigma|q \to e_i|$ is the total number of entity mentions in q. We set the probability of walking away from the query subgraph to the entity subgraph and vice versa as 0.5 by weight normalization.

Table 5
Automatic evaluation results for the query-conditional method (mQEG) vs. QFG and QEG. (Success rate as a percentage.)

Model	SR@1	SR@3	SR@5	SR@10
QFG (Boldi et al., 2009)	0	0	0.32	0.65
QEG (Bordino et al., 2013)	2.90	2.90	2.90	3.23
mQEG	▲3.22	▲3.55	▲4.19	▲6.45

(c) For edges in E_{EE}, we define the weight to be proportional to the frequency of the entity transitions:

$$w(e_i \rightarrow e_j) = |e_i \rightarrow e_j| / |e_i|,$$

where $|e_i \rightarrow e_j|$ denotes the number of occurrences of an entity transition from e_i to e_j, and $|e_i|$ the number of occurrences of all entity transitions starting from e_i.

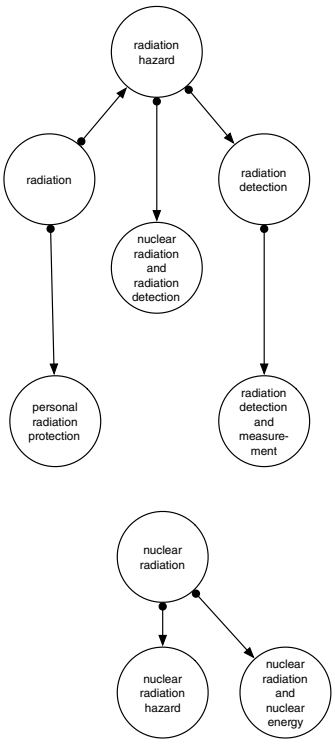
We provide an illustration of the mQEG with the QFG (Boldi et al., 2009) and QEG (Bordino et al., 2013) in Fig. 3, for the query "radiation hazard" that needs query suggestions. The difference between Fig. 3(a) and Fig. 3(c) shows that the entities in the mQEG have enriched the semantic connections among queries compared to QFG (Boldi et al., 2009). Furthermore, the mQEG offers stronger connectivity via the bidirectional edges between entities and queries compared to the QEG (Bordino et al., 2013). For instance, given the query "radiation hazard," the mQEG is able to provide the query suggestion "nuclear radiation hazard" while the QFG (Boldi et al., 2009) and QEG (Bordino et al., 2013) cannot as the required edges that link these queries are simply missing.

To construct the graph model, we use the entire query log except the last 5 days, which are used as our test set. In total there are 14,774,893 nodes and 100,679,495 edges in the mQEG. We use the Graphchi framework (Kyrola, 2013) to implement Personalized PageRank. We run 10 iterations for each round of personalized PageRank to achieve an approximation of the power iteration method. The number of walks is set to 100 to achieve a reasonable trade-off between speed and scale, while preventing the inclusion of queries that are only remotely relevant into the candidate sets. Given the input query, we run the algorithm and rank the query suggestion candidates by visiting frequencies in descending order.

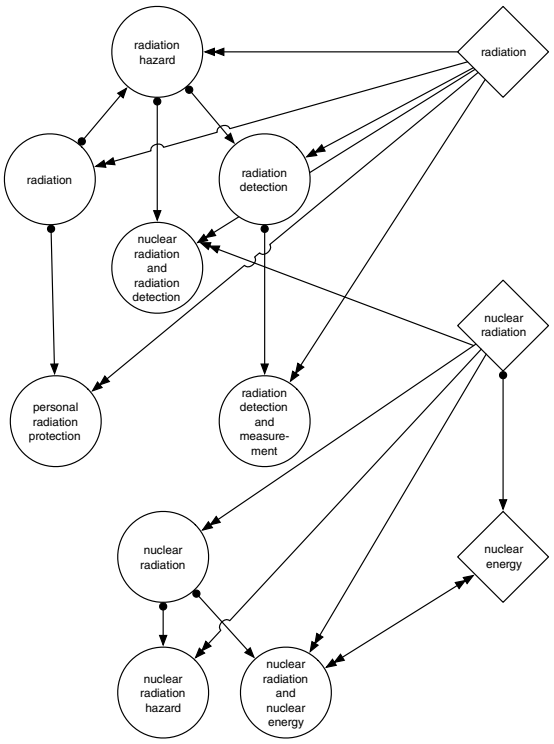
To examine the utility of this query-conditional query suggestion method, we compare the performance of the mQEG against the query flow graph (QFG) (Boldi et al., 2009), which is based on a similar graph model but does not consider the queries' entities. We also compare the mQEG against QEG (Bordino et al., 2013), although the QEG (Bordino et al., 2013) is designed for a different task: recommending queries for web pages. We follow the original steps in Bordino et al. (2013) by first performing entity set expansion on the entity subgraph in the QEG, which only consists of entity nodes and their links, to expand the entities in a query. And then we perform Personalized PageRank on the full QEG by using a uniform distribution over the expanded entities. We rank the query nodes by their visiting frequencies in descending order. The results (as a percentage) are presented in Table 5.

In absolute terms, the SR scores achieved by the mQEG are low because it is difficult to predict the exact same query with a click. Nevertheless, for SR@10 the scores achieved by the mQEG are comparable to the state-of-the-art in a different but related task (recommending orthogonal queries (Vahabi et al., 2013)). In Table 5, a significant improvement ($\alpha = .01$) in SR is observed in the query-conditional method (mQEG) over the QFG and the QEG. The low scores of the QFG demonstrate its weakness when dealing with rare queries. The performance increase from both models that use entities (mQEG and QEG) shows that utilizing the queries' entities helps surface more relevant query suggestions. Additionally, it helps tackle the query sparsity problem in academic search (Section 2.1), by using entities to relate queries that are not connected in the QFG. The performance increase of the mQEG over the QEG (Bordino et al., 2013) shows that the mQEG is more suitable for this task setting. This may be explained from two perspectives: (1) The bidirectional query-entity edges in the mQEG enhance connectivity and more effectively deal with the sparsity of queries. (2) The QEG (Bordino et al., 2013) is designed to make recommendations for web pages that usually have more entities than a query; after performing the entity expansion step, the QEG (Bordino et al., 2013) uses a uniform distribution over the expanded entities to produce query suggestions, which might bring undesirable topic drift to the original query and entities. Since the mQEG and QEG are both entity-based models and the former performs better in this task setting, we will only show the results of the mQEG in the remaining sections. The increase in SR@10 demonstrates the utility of the models as only a limited number of query suggestions can be shown to users.

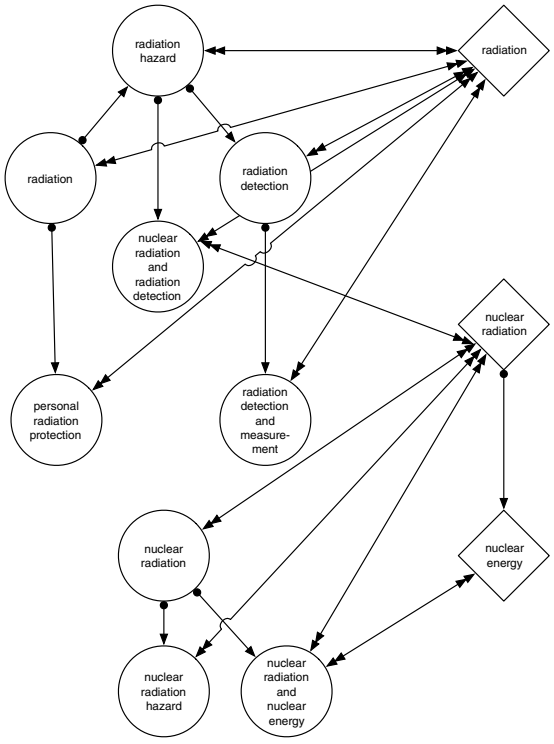
While the mQEG achieves a significant improvement over the QFG and QEG, there is still considerable room for improvement, especially at SR@1 and SR@3. Since the mQEG only uses query information, we hypothesize that using additional session information may improve the suggestions. Consider the example in Section 3.2, where a user is interested in the topic "composite beams" and is exploring. In this case, given that the session is an exploring session, it would be useful to directly provide exploratory suggestions around the entity "composite beams" upon observing a null query, such as the spelling-corrected query "a study on elastic plastic... of composite beams," or "Inelastic analysis of steel frames with composite beams" which is an even better query suggestion because it leads to a click. We hypothesize that predicting session types and biasing suggestions based on those predictions helps surface better query suggestions. We address the issue in the following section, with so-called session-conditional query suggestions.



(a) Query Flow Graph (Boldi et al., 2009).



(b) Query Entity Graph (Bordino et al., 2013).



(c) Modified Query Entity Graph.

Fig. 3. Illustration of a Query Flow Graph (Boldi et al., 2009), a Query Entity Graph (Bordino et al., 2013), and a Modified Query Entity Graph, all for the query "radiation hazard." Query nodes are represented by circles and entity nodes by squares.

Table 6
Features for prediction of null session type.

Name	Description
Query features	
wLength	Average query length in number of words
cLength	Average query length in number of chars.
numPlainSearch	Number of non-boolean informational queries
perPlainSearch	Percentage of non-boolean informational queries
numFail	Number of null search queries
numQuery	Number of queries
dwellTimeQuery	Average dwell time for each query
Query transition features	
numTrans	Number of transitions
avgQuerySim	Average query similarity
addTerms	Number of adding transitions
delTerms	Number of deleting transitions
subTerms	Number of substituting transitions
perAdd	Percentage of adding transitions
perDel	Percentage of deleting transitions
perSub	Percentage of substituting transitions
Click features	
numClick	Number of clicks
dwellTimeClick	Average dwell time for each click

4.3. Session-conditional query suggestions

The assumption underlying our session-conditional method is that we should give certain types of suggestion candidates a higher ranking in the suggestion list, depending on the session type. E.g., for a session that is likely to be exploring, exploratory query suggestions should appear higher up in the ranking. Therefore, we first need to predict the null session type; based on the predicted null session type, we then rerank the suggestion candidates generated by the graph models (QFG, mQEG).

4.3.1. Predicting null session type

We use the annotated sessions described in Section 3.2 and assign labels to sessions by considering the majority votes by the annotators as the training set. Recall that each session can be assigned multiple labels; hence, this prediction problem is cast as a multi-label classification problem. As we saw in Section 3.2, there is an imbalance of session types in the training set; therefore, the trained classifier is also likely to be biased towards prevalent session types. However, given that the annotated sessions come from random sampling, the distribution of session types is expected to be similar between the training set and the test set.

Our model for predicting null session types uses three feature sets: query features, query transition features, and click features; see Table 6.

Query features. These features describe basic characteristics of a query, such as length and dwell time.

Query transition features. These features describe how users reformulate queries during the session. Typical reformulations are adding terms, deleting terms and substituting terms. These transition patterns are likely to help to distinguish types of null sessions. E.g., in exploring sessions there are many substituting transitions, in which the user searches for related entities around a pivot entity. In refining sessions, the user may attempt adding and substituting terms to refine a search goal.

It is important to inspect query similarity within a session as this tells us how diverse the queries are, and also reflects the user's interest changes. Therefore, we inspect the query similarity in a session using the same pairwise similarity metric as we used in Section 3.3.

Click features. Click features are important feedback from users. They show the satisfaction of a user for a query, as well as the search intents. We look at the number of clicks and the mean dwell time for each click.

Predicting null session type is a multi-label classification problem where the input features and the labels may have very high degrees of dependency. To capture the dependencies, we use a 3-layer deep belief network with Restricted Boltzmann Machines (RBM) (Read, Perez-Cruz, & Bifet, 2015). We apply RBMs to learn a compact representation of the underlying patterns of the input features as well as the labels. The final hidden layer contains the output units for each label.

We use 10-fold cross validation and test this method against other methods such as classifier chains (Read, Pfahringer, Holmes, & Frank, 2011), binary relevance with SVM and random forests (Tsoumakas, Katakis, & Vlahavas, 2010). We only report on the RBM-based method as its performance is better. See Table 7.

We choose several metrics to measure the prediction performance:

Table 7
Performance of predicting null session type using a Restricted Boltzman Machine.

Performance over all types of null session	
Hamming score	0.891
Exact match score	0.591
F1 micro avg	0.744
F1 macro avg	0.698
Accuracy per null session type	
Refining sessions	0.768
Generalizing sessions	0.892
Exploring sessions	0.974
Expanding sessions	0.923

Table 8
Performance of predicting null session type at different stages.

Time of prediction	Hamming score	F1 micro avg.	F1 macro avg.
After initial null query	0.816	0.578	0.494
... 1st subsequent query	0.686	0.427	0.418
... 2nd subsequent query	0.885	0.731	0.679
... 3rd subsequent query	0.887	0.736	0.684
At session end	0.891	0.744	0.698

Hamming score: For each prediction of a session, let T be the true set of labels and S be the predicted set of labels, the hamming score is defined as:

$$Hamming\,score = |T \cap S|/|T \cup S|,$$

The Hamming score is then averaged over all predictions.

Exact match: is the percentage of sessions that have all their labels predicted correctly.

Macro-F1 is the harmonic mean between precision and recall, first averaged per label and then across all labels.

Micro-F1 is the harmonic mean between micro-precision (precision averaged over all the predictions) and micro-recall (recall averaged over all the predictions).

The Hamming score, i.e., the accuracy over all labels in this multi-label setting, reaches a value of 0.891, which indicates a good overall prediction result. If we split out the accuracy per label (in the bottom half of Table 7), we see that exploring sessions have the highest accuracy score of 0.974, while refining sessions have the lowest score of 0.768. These differences among the session types shows that certain types are easier to identify automatically.

How well does the prediction of null session type work at different stages in sessions, without knowing later session information? Table 8 shows that the performance of predicting the null session type following the first null query achieves a fair performance. Prediction after the next query after initial failure performs worse. This may indicate that the user's initial response upon a failure may be vague at first, which makes it difficult to assign a session type. Then, using two queries after the first failure achieves better performance. Our model for predicting null session type is capable of achieving good performance even with partial information.

4.3.2. Query suggestions

We use our null session type classifier to predict the types of null sessions. Then, the probability of a null session type will be used for reranking suggestion candidates. Here, we proceed as follows.

Given the null query, we generate suggestion candidates from one of the graph models (QFG, mQEG). Then the candidates are classified by an unsupervised multi-label classifier as refining, generalizing, exploring and expanding suggestions. The classification is a simple rule-based approach, defined by syntactic variance and term changes: specifically, generalizing suggestions contain a subset of the terms in the original query. Expanding suggestions contain new terms that are added to the original query. For exploring suggestions, there is term substitution in the original query while at least one entity term remains. For refining queries, we use a character-level edit-distance metric and classify all queries below a distance threshold θ as refining suggestions.

Algorithm 1 details how the ingredients are combined to produce session-conditional query suggestions. At line 1 we obtain query suggestion candidates produced by a baseline method (QFG or mQEG). At line 2 the candidates are divided into different types. In lines 3–9 the candidates are reranked based on the predicted session type probabilities, to form the final suggestion list.

The only variable for tuning is the distance threshold θ; we iterated over possible values and obtained the optimal performance of query suggestion results at $\theta = 0.2$.

Algorithm 1 Session conditional query suggestions.

Input:

Session s; Null query q; θ

The probability of session type $P(i \mid s)$ for the i-th type of session, where $i = 1, 2, 3, 4$ which correspond to refining, generalizing, exploring and expanding sessions.

Output:

Fused list of suggestions R;

1: Generate query suggestion candidate list L by one of the graphical models (QFG, mQEG)
2: Classify suggestions L into sublists L_i of different types, with θ being the distance threshold
3: **while** R is not full **do**
4: select L_i probabilistically according to $P(i \mid s)$
5: select the top-most unchosen query q on L_i
6: **if** $q \notin R$ **then**
7: append q to R
8: **end if**
9: **end while**

Table 9
Baseline query suggestion methods (QFG, mQEG) vs. session conditional versions of the baseline methods. Prediction of null session type is after the initial null query.

Model	SR@1	SR@3	SR@5	SR@10
QFG	0	0	0.32	0.65
SC-QFG	0.65	0.97	0.97	1.61
mQEG	3.23	3.55	4.19	6.45
SC-mQEG	4.52	$^{\triangle}$7.10	$^{\triangle}$7.42	8.39

Before discussing the experimental results, let us consider an example of Algorithm 1 in action. Consider the following session:

04Mar2015:06:49:58.489	Query	supply chain risk management
04Mar2015:06:58:13.402	Query	risk management
04Mar2015:06:58:48.198	Click	shorturl=/scie...pii/S026323730900005X
04Mar2015:06:59:27.762	Click	shorturl=/scie...pii/026323739190056V
04Mar2015:07:00:34.362	***Query***	***AHP TOPSIS***
04Mar2015:07:00:43.393	Query	AHP
04Mar2015:07:00:59.431	Click	shorturl=/scie...pii/S0263237312001107

First, given the null query “AHP TOPSIS”, the mQEG generates a list of query suggestion candidates at line 1. Then the candidates are classified into refining, generalizing, exploring and expanding suggestions at line 2. Using the session information from the input, the session type predictor determines that the current session has a very high probability of being a generalizing session, as the user has dropped terms “supply chain” in the first query reformulation. Therefore, the algorithm pushes the suggestion candidates that belong to the generalizing type higher up in the ranking among all the candidates at line 3 to line 9, of which the query “AHP” is benefited, and that is the successful query that leads to a click.

Next, we report on the query suggestion results for our session conditional methods in Table 9. An increase in SR is observed for both models (QFG and mQEG) after applying our session conditional approach. On top of the mQEG, the session conditional extension leads to significant improvements in SR@3 and SR@5 ($\alpha = .05$).

When we look at the successful recommendations made by the SC-mQEG and mQEG, we find that the SC-mQEG gives a better ranking than the mQEG for the successful query suggestions for 48% of the cases, a draw for 28%, and a lower ranking for 24%.

For the cases where the SC-mQEG and mQEG draw, the successful suggestions rank at the first place for 87.5% of the cases, and rank at the second place for 12.5% of the cases, making it difficult for the SC-mQEG to make further improvements over the mQEG.

For the cases where the SC-mQEG is outperformed by the mQEG, we find that for 57% of the cases the successful suggestions are exploratory ones, but the session prediction’s output sees the session as refining or generalizing. For 29% of the cases the session prediction is expanding, which is correct, but query suggestions that belong to other types have been

pushed up in the reranking process due to the randomness, thus lowering the rank of the desired query. In the remaining 14% of the cases it should be a refining suggestion but the session prediction is expanding.

For the cases where the SC-mQEG outperforms the mQEG, the improvements come from the session type predictions that are most likely to be refining and generalizing, constituting 29% and 71% of the cases, respectively. Nevertheless, the "less confident" session predictions for exploring and expanding types may still contribute: in the test cases for which the SC-mQEG outperforms the mQEG, the exploratory query suggestions account for 7% of them and expanding suggestions account for 14%, although the most likely session type prediction for these sessions is neither exploring or expanding but refining. This shows that it is not always the most likely prediction that works, instead it can also be a less-likely session type prediction that contributes to the successful query suggestion.

From the analyses we infer that the session type prediction's most confident prediction may not always be correct; however, even a sub-optimal prediction may help push the desired query higher up in reranking. Overall, our findings confirm that in most cases, predicting session type helps make equal or better query suggestions.

5. Discussion: user-conditional query suggestions

We have used both queries (or rather: entities in queries) and sessions to improve query suggestions for null queries. It seems natural to consider using user-specific information, i.e., personalization, for query suggestion. However, when we examined the cross-session similarity for each user that has at least 3 sessions, as shown in Table 4. The average cross-session similarity score is very low, which indicates shifting interests across sessions, and this, in turn, suggests that the benefit of personalization may be limited. Below, we report on experiments aimed at determining the benefit of personalized query suggestions.

We focus on users' preferences over entities for two reasons: (1) The majority of queries contain at least an entity; (2) Entities reflect the users' topic interests. We aim to see if they help to improve the quality of query suggestions. To this end, we create a personalized query-entity graph (PmQEG) by integrating user preferences into the edge weights (transition probabilities) of the graph.

5.1. Inferring user preference

We derive each user's preference for entities by looking at query reformulations. These reformulations fall into two broad categories: query reformulations that have at least a common entity in the queries and those that do not, from which we can infer conditional and unconditional entity preferences, respectively:

1. For the conditional type, we look at three common cases of query reformulation:
   (a) deletion;
   (b) addition;
   (c) substitution.

   We cannot infer a clear user preference based on deletion or addition, as in these cases users try to generalize or refine a query. But in the case of a substitution, we are able to infer a conditional preference. Specifically, assume that two consecutive queries q_i and q_j contain entities E_i and E_j, respectively, and shared entities $E_c = E_i \cap E_j$. Then, the user prefers the new entity set $E_j \backslash E_c$ over the previous entity $E_i \backslash E_c$ given the shared entity set E_c. Put more formally: $P(E_j \setminus E_c \mid E_c) \succ P(E_i \setminus E_c \mid E_c)$.
2. For the second type, where there is no common entity in two consecutive queries, it is possible to infer an unconditional preference. If a user issues consecutive queries q_i and q_j, which contain entity set E_i and E_j respectively, we infer that they have a preference for E_j over E_i ($P(E_j) \succ P(E_i)$).

All inferences of entity preference inherently consider a user's interest shifts over time: since queries arrive sequentially, this pattern favors new entity preferences over old ones.

5.2. Preference model

For each user u, we derive preference information from the query log. Let s denote the shared set of entities, which could be $\emptyset$ (in such cases the preference is not conditional), and p and c are two entity sets between which the user prefers p. Then the training data D is made of all preference pairs: $D = \{(u, s, p, c) \mid (p \mid s) \succ (c \mid s)\}$. We use MAP (maximum a posteriori probability) estimation to derive the preferences over entities. For unobserved preferences of entities, we use Laplace smoothing to assign a non-zero probability: $P = (x + \alpha)/(N + \alpha \times \epsilon_{ij})$, where ϵ_{ij} is the number entity transitions given a source entity e_i, N is the number of observations for e_i and x is the number of the observed entity preference; $x = 0$ for unobserved preferences.

When personalization is applied to all users, an increase of SR at different cutoffs is observed; see Table 10. Specifically, the PmQEG has improvements of SR at all cutoffs, and SC-PmQEG improves the SR@3 and SR@5 scores. The improvements show that personalization is better able to put high quality query suggestions into higher rankings, but the differences are not statistically significant.

Table 10
Personalized models (PmQEG, SC-PmQEG) vs. Non-personalized models (mQEG, SC-mQEG).

Model	SR@1	SR@3	SR@5	SR@10
mQEG	3.23	3.55	4.19	6.45
PmQEG	3.55	4.84	4.84	6.77
SC-mQEG	4.52	7.10	7.42	8.39
SC-PmQEG	4.52	7.42	7.74	8.39

The consistency of a user's interests does not necessarily impact the performance of personalization. For some cases personalization brings performance gains, because users issue queries with exactly the same interest in the history (e.g., a user has a historical preference for "management," which leads to the query suggestion "customer relationship management" that is desired). But in other cases personalization may not produce much utility for users who have consistent interests in a certain topic, but are constantly exploring around subtopics within it. When a new subtopic comes up, the historical preferences of previous subtopics might lead query suggestions astray and hurt personalization.

In this section we have discussed personalization for users' entity preferences, as entities are prevalent in academic queries. Looking further, personalization can be achieved by considering topical interests, user behavior, session information, and also specific search patterns such as refinding (Teevan, 2007). It is yet to be seen whether general personalization methods in web search would work in academic search. After all, personalization should be a discreet decision to make for the search engine in order not to hurt user experience. We put the problem of an in-depth analysis of personalization as future work.

6. Related work

We consider four areas of related work: academic search, search failure, query suggestions and query auto-completion.

6.1. Academic search

Academic search concerns the retrieval and profiling of information objects (papers, journals, authors, reviewers, ...) in the domain of academic research. The first academic search engine MEDLINE came operational in 1971, which supported up to 25 users simultaneously (Lindberg, 2000). However, its use was limited to libraries and only pre-programmed search tasks were supported instead of online queries. It was not until the 1990s when the World Wide Web became popular, that online academic search engines started to thrive and became accessible to a larger user base. These online academic search systems include Citeseer (Giles, Bollacker, & Lawrence, 1998) and Aminer (Tang, 2016), which focus on citation indexing and metadata extraction as well as academic social network extraction, respectively. Multiple heuristics such as term frequency and citation scores can also be applied to increase the performance of academic search engines (Amolochitis, Christou, Tan, & Prasad, 2013). This paper studies one of the world's most popular academic search engines, Sciencedirect (ScienceDirect, 2015), which is widely used in the physical sciences, engineering and life sciences.

Several transaction log analyses have been conducted on search engines of digital libraries. However they are either focusing on a single discipline (Han et al., 2014; Jones et al., 2000), or limited in scale (Ke et al., 2002), thus making them not representative of academic search. Moreover, these analyses focus on revealing basic statistics, and little insight on user behavior in search sessions is given.

6.2. Search failure

Lancaster (1968) uncovers failure phenomena in an early academic retrieval system MEDLARS in the 1960s. Dwyer, Gossen, and Martin (1991) examine failures of interlibrary loan-request forms for items in two university libraries from 1989 to 1990. They find 17 types of error such as "in circulation" and "incorrect citation." However, the "queries" in those obsolete systems are hand-crafted and static requests mostly from librarians, which are completely different from the modern form of queries that we type in the search box. The "failures" in the past are therefore different from what modern users encounter in online and interactive academic search engines.

Singh, Parikh, and Sundaresan (2012) study search trails of various eBay users and the impact of null queries on purchase rate. They observe a degradation of purchase rate for null search trails compared to the non-null search trails. They find that the purchase rate for both power users as well as novices is lowered when null recall situations are encountered on their trails. They also find the repetition factor to be as low as 1.45 for null queries versus 19.57 for non-null queries. A low repetition factor makes it difficult to use query log-based signals to improve the performance of null queries.

6.3. Query suggestions

Query suggestion is a feature in modern web search that improves the search experience by providing recommendations of queries. Most query suggestion techniques exploit a query log in order to give useful suggestions.

Zhang and Nasraoui (2006) use a similarity measure by exploiting consecutive queries during sessions combined with a content-based method using search frequency and query frequency. Boldi et al. (2009) introduce the query-flow graph by examining different reformulation patterns in search sessions, and uses random walk on the graph to obtain suggestions. Guo, Cheng, Xu, and Shen (2010) further use social annotation data to construct a query-URL-tag tripartite graph and use random walks to recommend queries in a structured way. Bordino, Castillo, Donato, and Gionis (2010) project the query-flow graph to a lower-dimension space to measure the similarity between queries for diverse query recommendations. Guo et al. (2011) use clicks and snippets to identify search intents and provide query recommendations under different intents. Song, Zhou, and He (2012) propose a term-transition graph for query suggestions, using information from queries and documents.

These techniques, however, perform well for queries that come with clicks and other user feedback information. The long tail of the queries is more challenging for query suggestion, since such queries are rare and very little user feedback is available for them. Bonchi et al. (2012) use a term-query graph and provide query suggestions at a term level by computing the center-piece subgraph of the terms in queries. Vahabi et al. (2013) propose orthogonal query recommendation, which suggests queries that are syntactically different but semantically similar, to address the situation when the original query is ill-posed.

Among the queries in the long tail, there is a specific kind of query that are difficult to deal with: the ones that return very few or no results. In a study by Altingovde et al. (2012), these queries are characterized and most of these “failed” queries are found to contain a URI. Apart from the malformed URI queries, one third of them still contain a regular URI that the search engine can not retrieve. In these cases it is difficult to provide query suggestions as the search engine can not understand user intents.

There is an increasing volume of research on providing suggestions around entities. The use of entities utilizes auxiliary resources by linking entities in queries to knowledge bases such as Freebase and Wikipedia. Bordino et al. (2013) extend the query-flow graph by introducing entities in queries, and uses personalized PageRank to give query suggestions given a visited page. Hassan Awadallah, White, Pantel, Dumais, and Wang (2014) deal with sessions with complex search tasks. They tag entities in query text and group queries into tasks for recommendation.

6.4. Query auto-completion

Query auto-completion (QAC) is a popular function in search engines to help user formulate queries given the prefixes that the user is typing (Cai & de Rijke, 2016). Contrary to the post-remedy role of query suggestions, QAC has the potential to prevent null queries’s appearances. Mitra, Shokouhi, Radlinski, and Hofmann (2014) studies how users engage in QAC and finds that they tend to use it on word boundaries, which is helpful for difficult words. Shokouhi (2013) studies personalized QAC by considering a user’s previous queries. Cai, Liang, and de Rijke (2014) augment the personalized model by considering time-sensitivity. Recently, Zhang et al. (2015) propose an adaptive model that uses implicit negative feedback during user-QAC interaction. Mitra and Craswell (2015) design a QAC system for rare query prefixes using a latent semantic model.

Our work differs from previous work on academic search, by studying user behavior at a fine-grained level through a large-scale transaction log analysis. It also differs from algorithmic work on query suggestion by taking into account the unique characteristics of academic search, namely entity queries and, especially, null session types. This work is complementary to query auto-completion, as a post-remedy measure as opposed to being a precaution method.

7. Conclusions

In this paper we have investigated search behavior and failures in academic search. First we have identified the unique features of academic search, some of which provide observational insights for algorithmic improvements, e.g., richness of entity queries, verbosity of queries and unique search session types.

Then we have pointed out the problem of null queries, and motivate the use of query suggestions to address null queries. We generate query suggestion candidates using graph models that utilize entities in queries. Given the various session types, we propose a session-conditional approach. We have subsequently trained a multi-label classifier to predict the type of session in which a null query occurs. We then base the query suggestions on the predicted types of the null sessions. We find that the improvement of the session-conditional method is significant. Furthermore, we investigated personalization and have achieved a slight improvement.

The theoretical implication of this research lies in two main aspects in the query suggestion method. First the query-conditional approach, which effectively uses entities to surface more relevant query suggestions, further proves the importance of utilizing entities in information retrieval tasks. Moreover, the session-conditional approach shows that session information will help reranking query suggestions for null queries. This approach also does not rely much on characteristics of academic search data, which makes it possible to be generalized to web search and other domains, but of course it definitely requires further investigation. The practical implication of this research is the observation of user behavior in a modern academic search engine. The basic characteristics of search queries, their differences with web search, and the failure phenomenon all help to draw the big picture of academic search, and will draw more attention to research in this domain.

As to limitations of our study, we acknowledge that there are many options of personalization techniques for query suggestions, and the one adopted by the authors is only one of them that is motivated by the characteristic of academic queries (entity richness). It is certainly possible to examine whether general personalization methods in web search will achieve comparable performance in academic search, which we leave as future work. It should also be noted that the appearances of null queries may be related to certain search engine design techniques, e.g. whether to use query auto completion, query expansion and semantic matching. Yet it is meaningful to study how users would react upon null queries, especially in our setting where it is prevalent, so that we can deepen our understanding of how to alleviate this problem. In addition, some of those null queries are actually "positive failures" for searchers doing provenance finding: no result for a query confirms the non-existence of novel research ideas. Therefore it is interesting to combine research on null queries with provenance finding so that the search engine can judge whether a null query makes sense, and only provides query suggestions when necessary.

As to future work, we recommend improved profiling of searchers on multiple dimensions, e.g., preferences at the topic level, modeling intent shifts and it would be meaningful to examine when and whether to personalize.

Acknowledgments

We would like to thank our reviewers for their helpful comments. This research was supported by Ahold Delhaize, Amsterdam Data Science, Blendle, the Bloomberg Research Grant program, the Dutch national program COMMIT, Elsevier, the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement nr 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 number 027.012.105, the Netherlands Institute for Sound and Vision, the Netherlands Organisation for Scientific Research (NWO) under project nrs 727.011.005, 612.001.116, HOR-11-10, 640.006.013, 612.066.930, CI-14-25, SH-322-15, 652.002.001, 612.001.551, 652.001.003, 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.

References

Altingovde, I. S., Blanco, R., Cambazoglu, B. B., Ozcan, R., Sarigil, E., & Ulusoy, O. (2012). Characterizing web search queries that match very few or no results. In *CIKM* (pp. 2000–2004).

Amolochitis, E., Christou, I. T., Tan, Z.-H., & Prasad, R. (2013). A heuristic hierarchical scheme for academic search and retrieval. *Information Processing & Management, 49*(6), 1326–1343.

Arampatzis, A., & Kamps, J. (2008). A study of query length. In *SIGIR* (pp. 811–812).

Artstein, R., & Poesio, M. (2008). Inter-coder agreement for computational linguistics. *Computational Linguistics, 34*(4), 555–596.

Bailey, P., White, R. W., Liu, H., & Kumaran, G. (2010). Mining historic query trails to label long and rare search engine queries. *ACM Transactions on the Web, 4*(4), 15:1–15:27.

Beitzel, S. M., Jensen, E. C., Chowdhury, A., Frieder, O., & Grossman, D. (2007). Temporal analysis of a very large topically categorized web query log. *Journal of the Association for Information Science and Technology, 58*(2), 166–178.

Bendersky, M., & Croft, W. B. (2009). Analysis of long queries in a large scale search log. In *WSCD* (pp. 8–14).

Blanco, R., Ottaviano, G., & Meij, E. (2015). Fast and space-efficient entity linking for queries. In *WSDM* (pp. 179–188).

Boldi, P., Bonchi, F., Castillo, C., Donato, D., & Vigna, S. (2009). Query suggestions using query-flow graphs. In *WSCD* (pp. 56–63).

Bonchi, F., Perego, R., Silvestri, F., Vahabi, H., & Venturini, R. (2012). Efficient query recommendations in the long tail via center-piece subgraphs. In *SIGIR* (pp. 345–354).

Bordino, I., Castillo, C., Donato, D., & Gionis, A. (2010). Query similarity by projecting the query-flow graph. In *SIGIR* (pp. 515–522).

Bordino, I., De Francisci Morales, G., Weber, I., & Bonchi, F. (2013). From machu picchu to rafting the urubamba river anticipating information needs via the entity-query graph. In *WSDM* (pp. 275–284).

Cai, F., Liang, S., & de Rijke, M. (2014). Time-sensitive personalized query auto-completion. In *CIKM* (pp. 1599–1608).

Cai, F., & de Rijke, M. (2016). A survey of query auto completion in information retrieval. *Foundations and Trends in Information Retrieval, 10*(4), 273–363.

Dogan, R. I., Murray, G. C., Névéol, A., & Lu, Z. (2009). Understanding pubmed® user search behavior through log analysis. *Database, 2009*, bap018.

Dwyer, C. M., Gossen, E. A., & Martin, L. M. (1991). Known-item search failure in an OPAC. *RQ*, 228–236.

Fleiss, J. L. (1971). Measuring nominal scale agreement among many raters. *Psychological Bulletin, 76*(5), 378.

Giles, C. L., Bollacker, K. D., & Lawrence, S. (1998). Citeseer: An automatic citation indexing system. In *DI* (pp. 89–98).

Guo, J., Cheng, X., Xu, G., & Shen, H. (2010). A structured approach to query recommendation with social annotation data. In *CIKM* (pp. 619–628).

Guo, J., Cheng, X., Xu, G., & Zhu, X. (2011). Intent-aware query similarity. In *CIKM* (pp. 259–268).

Gupta, M., & Bendersky, M. (2015). Information retrieval with verbose queries. *Foundations and Trends in Information Retrieval, 9*(3–4), 209–354.

Han, H., Jeong, W., & Wolfram, D. (2014). Log analysis of academic digital library: User query patterns. In *iConference 2014 Proceedings* (pp. 1002–1008).

Hassan, A., White, R. W., Dumais, S. T., & Wang, Y.-M. (2014). Struggling or exploring?: Disambiguating long search sessions. In *WSDM* (pp. 53–62).

Hassan Awadallah, A., White, R. W., Pantel, P., Dumais, S. T., & Wang, Y.-M. (2014). Supporting complex search tasks. In *CIKM* (pp. 829–838).

Hemminger, B. M., Lu, D., Vaughan, K., & Adams, S. J. (2007). Information seeking behavior of academic scientists. *Journal of the American Society for Information Science and Technology, 58*(14), 2205–2225.

Jansen, B. J., Booth, D. L., & Spink, A. (2008). Determining the informational, navigational, and transactional intent of web queries. *Information Processing & Management, 44*(3), 1251–1266.

Jones, S., Cunningham, S. J., McNab, R., & Boddie, S. (2000). A transaction log analysis of a digital library. *International Journal on Digital Libraries, 3*(2), 152–169.

Ke, H.-R., Kwakkelaar, R., Tai, Y.-M., & Chen, L.-C. (2002). Exploring behavior of e-journal users in science and technology: Transaction log analysis of elsevier's sciencedirect onsite in taiwan. *Library & Information Science Research, 24*(3), 265–291.

Khabsa, M., Wu, Z., & Giles, C. L. (2016). Towards better understanding of academic search. In *JCDL* (pp. 111–114).

Kim, Y., Seo, J., & Croft, W. B. (2011). Automatic boolean query suggestion for professional search. In *SIGIR* (pp. 825–834).

Kim, Y., Seo, J., Croft, W. B., & Smith, D. A. (2014). Automatic suggestion of phrasal-concept queries for literature search. *Information Processing & Management, 50*(4), 568–583.

Krippendorff, K. (2004). *Content analysis: An introduction to its methodology*. Sage.

Kyrola, A. (2013). DrunkardMob: Billions of random walks on just a PC. In *RECSYS* (pp. 257–264).

Lancaster, F. W. (1968). *Evaluation of the MEDLARS demand search service.*. U.S. Dept. of Health, Education, and Welfare, Public Health Service.
Landis, J. R., & Koch, G. G. (1977). The measurement of observer agreement for categorical data. *Biometrics, 33*(1), 159–174.
Lin, T., Pantel, P., Gamon, M., Kannan, A., & Fuxman, A. (2012). Active objects: Actions for entity-centric search. In *WWW* (pp. 589–598).
Lindberg, D. (2000). Internet access to the National Library of Medicine. *Effective Clinical Practice, 3*(5), 256–260.
Meij, E., Bron, M., Hollink, L., Huurnink, B., & de Rijke, M. (2011). Mapping queries to the linking open data cloud: A case study using dbpedia. *Web Semantics, 9*(4), 418–433.
Mishne, G., & de Rijke, M. (2006). A study of blog search. In *ECIR* (pp. 289–301).
Mitra, B., & Craswell, N. (2015). Query auto-completion for rare prefixes. In *CIKM* (pp. 1755–1758).
Mitra, B., Shokouhi, M., Radlinski, F., & Hofmann, K. (2014). On user interactions with query auto-completion. In *SIGIR* (pp. 1055–1058).
Niu, X., Hemminger, B. M., Lown, C., Adams, S., Brown, C., Level, A., … Cataldo, T. (2010). National study of information seeking behavior of academic researchers in the United States. *Journal of the American Society for Information Science and Technology, 61*(5), 869–890.
Ogilvie, P., & Callan, J. (2003). Combining document representations for known-item search. In *SIGIR* (pp. 143–150).
Pontis, S., & Blandford, A. (2015). Understanding “influence:” an exploratory study of academics' processes of knowledge construction through iterative and interactive information seeking. *Journal of the Association for Information Science and Technology, 66*(8), 1576–1593.
Pontis, S., Blandford, A., Greifeneder, E., Attalla, H., & Neal, D. (2015). Keeping up to date: An academic researcher's information journey. *Journal of the American Society for Information Science and Technology*. Online since 22 October 2015
Read, J., Perez-Cruz, F., & Bifet, A. (2015). Deep learning in partially-labeled data streams. In *SAC* (pp. 954–959).
Read, J., Pfahringer, B., Holmes, G., & Frank, E. (2011). Classifier chains for multi-label classification. *Machine Learning, 85*(3), 333–359.
ScienceDirect (2015). http://sciencedirect.com. Last accessed September 14, 2015.
Shokouhi, M. (2013). Learning to personalize query auto-completion. In *SIGIR* (pp. 103–112).
Silvestri, F. (2010). Mining query logs: Turning search usage data into knowledge. *Foundations and Trends in Information Retrieval, 4*(1–2), 1–174.
Singh, G., Parikh, N., & Sundaresan, N. (2012). Rewriting null e-commerce queries to recommend products. In *WWW* (pp. 73–82).
Song, Y., Zhou, D., & He, L.-w. (2012). Query suggestion by constructing term-transition graphs. In *WSDM* (pp. 353–362).
Sun, X., Kaur, J., Possamai, L., & Menczer, F. (2013). Ambiguous author query detection using crowdsourced digital library annotations. *Information Processing & Management, 49*(2), 454–464.
Tang, J. (2016). Aminer: Toward understanding big scholar data. In *WSDM* (p. 467).
Teevan, J. (2007). The re:search engine: Simultaneous support for finding and re-finding. In *UIST* (pp. 23–32).
Tsoumakas, G., Katakis, I., & Vlahavas, I. (2010). Mining multi-label data. In *Data mining and knowledge discovery handbook* (pp. 667–685). Springer.
Vahabi, H., Ackerman, M., Loker, D., Baeza-Yates, R., & Lopez-Ortiz, A. (2013). Orthogonal query recommendation. In *RECSYS* (pp. 33–40).
Wedig, S., & Madani, O. (2006). A large-scale analysis of query logs for assessing personalization opportunities. In *KDD* (pp. 742–747).
Weerkamp, W., Berendsen, R., Kovachev, B., Meij, E., Balog, K., & de Rijke, M. (2011). People searching for people: Analysis of a people search engine log. In *SIGIR* (pp. 45–54).
Zhang, A., Goyal, A., Kong, W., Deng, H., Dong, A., Chang, Y., … Han, J. (2015). adaQAC: Adaptive query auto-completion via implicit negative feedback. In *SIGIR* (pp. 143–152).
Zhang, Z., & Nasraoui, O. (2006). Mining search engine query logs for query recommendation. In *WWW* (pp. 1039–1040).