Foundations and Trends® in Information Retrieval
Vol. 10, No. 4 (2016) 1–92

DOI: 10.1561/XXXXXXXXXX

now
the essence of knowledge

A Survey of Query Auto Completion in Information Retrieval

Fei Cai[1,2] f.cai@uva.nl
Maarten de Rijke[2] derijke@uva.nl

[1] Science and Technology on Information Systems Engineering Laboratory, National University of Defense Technology, Changsha, China
[2] Informatics Institute, University of Amsterdam, Amsterdam, The Netherlands

Contents

1 Introduction **2**
- 1.1 Motivation 2
- 1.2 Historical notes 4
- 1.3 Aims of this survey 6
- 1.4 Structure of this survey 6

2 Query auto completion **8**
- 2.1 Problem formulation 8
- 2.2 Aims of query auto completion 10
- 2.3 A taxonomy of solutions to query auto completion 11

3 Heuristic approaches to query auto completion **14**
- 3.1 Time-sensitive query auto completion models 14
- 3.2 User-centered query auto completion models 21
- 3.3 Summary 25

4 Learning-based approaches to query auto completion **26**
- 4.1 Learning from time-related characteristics 27
- 4.2 Learning from user interactions 30
- 4.3 Summary 34

5 Evaluation **35**

5.1 Datasets 35
5.2 Evaluation metrics 38
5.3 Main experimental findings 40
5.4 Summary 42

6 Efficiency and robustness 43
6.1 Computational efficiency in query auto completion . . . 43
6.2 Error-tolerance in query auto completion 47
6.3 Summary 48

7 Presentation and interaction 50
7.1 Presenting query auto completions results 51
7.2 Interactions with query auto completion results 53
7.3 Summary 56

8 Related tasks 57
8.1 Query suggestion 57
8.2 Query expansion 60
8.3 Query correction 62

9 Conclusions 66
9.1 Summary of the survey 66
9.2 Open research directions 67
9.3 Concluding remarks 71

Glossary 72

Acknowledgements 73

References 74

Abstract

In information retrieval, query auto completion (QAC), also known as type-ahead [Xiao et al., 2013, Cai et al., 2014b] and auto-complete suggestion [Jain and Mishne, 2010], refers to the following functionality: given a prefix consisting of a number of characters entered into a search box, the user interface proposes alternative ways of extending the prefix to a full query. Ranking query completions is a challenging task due to the limited length of prefixes entered by users, the large volume of possible query completions matching a prefix, and the broad range of possible search intents. In recent years, a large number of query auto completion approaches have been proposed that produce ranked lists of alternative query completions by mining query logs.

In this survey, we review work on query auto completion that has been published before 2016. We focus mainly on web search and provide a formal definition of the query auto completion problem. We describe two dominant families of approaches to the query auto completion problem, one based on heuristic models and the other based on learning to rank. We also identify dominant trends in published work on query auto completion, viz. the use of time-sensitive signals and the use of user-specific signals. We describe the datasets and metrics that are used to evaluate algorithms for query auto completion. We also devote a chapter to efficiency and a chapter to presentation and interaction aspects of query auto completion. We end by discussing related tasks as well as potential research directions to further the area.

F. Cai and M. de Rijke. *A Survey of Query Auto Completion in Information Retrieval*. Foundations and Trends® in Information Retrieval, vol. 10, no. 4, pp. 1–92, 2016.

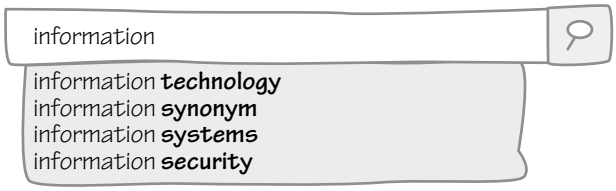
DOI: 10.1561/XXXXXXXXXX.

1

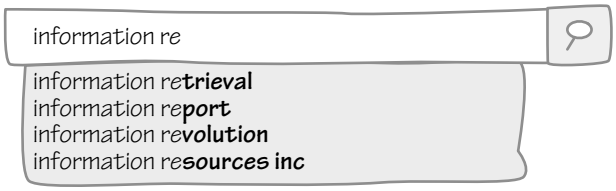
Introduction

1.1 Motivation

Word completion is a user interface feature that offers the user a list of words after one or more letters have been typed. It has been around as a feature of word editors and command shells for nearly half a century. In information retrieval, word completion is best known in the form of query auto completion, which provides users with suggested queries as they begin to enter their query in the search box. The user's incomplete input is often called a *query prefix* and the suggested queries are often called *query completions*. As shown in Figure 1.1, these query completions often start with the characters that the user has entered into the search box (see Figure 1.1a) and are adjusted as the user continues typing (see Figure 1.1b). Query auto completion (QAC) helps users to formulate their query when they have an intent in mind but not a clear way of expressing in a query. It helps to avoid possible spelling mistakes, especially on devices with small screens. It has been reported that, when submitting English queries, using QAC by selecting suggested query completions has saved more than 50% keystrokes of global Yahoo! searchers in 2014 [Zhang et al., 2015]. In addition, cutting down the search duration of users implies a lower load on the search engine, which results in savings in machine resources and maintenance [Bar-Yossef and Kraus, 2011].



(a) A list of four query completions for the prefix "information."



(b) An updated list of four query completions for the prefix "information re."

Figure 1.1: Examples of query auto completion.

Importantly, the use of QAC is not limited to web search engines such as Baidu, Bing, Google, or Yandex. QAC has also become part of other common services for search tasks. Facebook[1] provides QAC for finding friends, typically providing four suggestions; Twitter[2] uses QAC for searching tweets, also offering up to four suggestions; QAC is employed for product search in online shopping stores such as Amazon,[3] which often produce up to ten suggestions; and it assists users to formulate requests for specific videos in online video-sharing websites such as Youtube,[4] where users also get up to ten suggestions. As with web search, users of QAC functionality in other settings are helped to find an ideal query to address a particular search task. Research indicates that using the suggested queries can greatly improve user satisfaction, especially for informational queries [Song et al., 2011b].

[1] https://www.facebook.com
[2] https://twitter.com
[3] http://www.amazon.com
[4] https://www.youtube.com

1.2 Historical notes

Query auto completion is a particular form of word prediction: when a writer writes the first letter or letters of a word, a word predictor lists possible words as choices and if the intended word is listed amongst the choices, it can be selected. In a different form of word prediction, the most likely next words (instead of completions) are listed. The core idea goes back at least half a century: Longuet-Higgins and Ortony [1968] describe a method to help decrease the number of keystrokes needed in order to complete a word, in their case commands and identifiers entered by developers. Another important motivation was to help individuals with physical disabilities increase their typing speed. Early work on word prediction focused both on assessing the cognitive benefits of word prediction and on algorithmic developments. For instance, early studies found that reduction of keystrokes through word prediction is often offset by having to scan the list of predicted words and selecting the desired word [Vanderheiden and Kelso, 1987]. Displaying five suggested words in a vertical manner was found to provide a reasonable balance between keystrokes saving and cognitive load [Swiffin et al., 1987].

Early algorithmic work on word prediction, that is, work from the 1970s and 1980s, can be grouped into three classes: character predictors, word completers, and combinations of the two [Darragh, 1988]. Character predictors accelerate input by making likely letters faster and easier to select. Word completors support text input by offering words based on an initial prefix of one or more letters entered by the user. Combined systems do both. The *Reactive Keyboard* introduced by Darragh et al. [1990] is a combined approach that also incorporates many facilities of today's query auto completion methods, including the use of popular words and of task appropriate words, both as identified from past query behavior.

Witten and Darragh [1992] provide an extensive treatment of work on word completion as an assistive technology up to the early 1990s. The *Reactive Keyboard* mentioned above is a prototypical example for the period as it incorporates the functionality of word completion and accelerates typewritten communication with a computer. Its word completion is mostly based on a standard dictionary; later approaches extended it to use adaptive modeling, which produces completions based on previously entered text. Similarly, a user's previously entered text, i.e., the search context, is taken into account

for personalized query auto completion [Bar-Yossef and Kraus, 2011]. Furthermore, a user's long-term search history has been explored for query auto completion [Cai et al., 2016a], but it has not been incorporated into a word completion system. Basically, word completion relies on a special tree structure to match the input and its completions [Witten and Darragh, 1992, Darragh et al., 1990], which is similar to the data structure used in query auto completion. In addition, both word completion and query auto completion often build on simple lexical models based on n-grams, which can be directly extracted from a text corpus or query log [Cai and de Rijke, 2016b].

Sentence completion is a slightly different task: given an initial fragment of a sentence, identify the remaining part of the sentence that the user intends to write. This task setting is motivated by processes such as answering questions sent by email in a call center, or writing letters in an administrative environment [Grabski and Scheffer, 2004]. Grabski and Scheffer [2004], Bickel et al. [2005], Nandi and Jagadish [2007] study this task based on lexical statistics of text collections.

The use of word prediction for search, sometimes called *incremental search* or *real-time suggestions* before the phrase *query auto completion* caught on, goes back to the Emacs text editor [Ciccarelli, 1978], which offered a single line of feedback, not a list of suggestions. In the early 2000s, Raskin [2000] advocated the use of incremental search in which a user gets instant feedback as they enter a query based on what may exist in the content search over what he called *delimited search*, which basically is a traditional search interface allowing search in three steps: a user submits a query, the system computes a result page, the user receives the result page. Since then, query auto completion has been used by internet browsers, development environments, web sites, desktop search, operating systems, databases, email clients, and search engines, which are the focus of this survey. Around the middle of the 2000s, QAC was being offered by a growing number of search engines. For instance, Google Suggest, which launched in 2004,[5] was an early showcase of query auto completion in the search engine setting. And around the same time Kayak,[6] a travel metasearch engine, completed location related

[5] http://looksgoodworkswell.blogspot.nl/2005/12/distracting-or-narrowing-looking.html

[6] https://www.kayak.com

queries with travel related suggestions such as airport codes.

1.3 Aims of this survey

Because of the clear benefits of QAC, a considerable number of algorithmic approaches to QAC have been proposed in the past few years. Query logs have proved to be a key asset underlying most of the recent research. In this paper we survey this research. We focus on summarizing the literature on QAC and provide a general understanding of the wealth of QAC approaches that is currently available. Our contributions in this survey can be summarized as follows:

1. We provide researchers who are working on query auto completion or related problems in the field of information retrieval with a good overview and analysis of state-of-the-art QAC approaches. In particular, for new researchers to the field, the survey can serve as an introduction to the state-of-the-art.
2. We offer a comprehensive perspective on QAC approaches by presenting a taxonomy of existing solutions. In addition, we present solutions for QAC under different conditions such as available high-resolution query logs, in-depth user interactions with QAC using eye-tracking, and elaborate user engagements in a QAC process. We also discuss practical issues related to QAC.
3. We present a detailed discussion of core challenges and promising open directions in QAC.

Our focus in this survey is on effectiveness of QAC. We do, however, include chapters on efficiency (Chapter 6) and on presentation and interaction aspects of query auto completion (Chapter 7).

1.4 Structure of this survey

The rest of this survey is organized as follows. Chapter 2 describes the problem of QAC as well as some closely related tasks; Chapter 3 and Chapter 4 detail published QAC approaches based on heuristic models and on learning algorithms, respectively; Chapter 5 presents publicly available benchmarks as well as metrics for QAC evaluation. In Chapter 6 we discuss efficiency and robustness aspects of query auto completion. In Chapter 7 we include a

survey of studies into interface and interaction aspects of query auto completion. We include a short overview of three areas of research that are closely related to QAC, i.e., query suggestion, query expansion and query correction in Chapter 8. We conclude the survey in Chapter 9 and point out follow-up research directions.

2

Query auto completion

In this chapter, we provide a formal definition of the query auto completion (QAC) problem in §2.1 and present the major aims of QAC in §2.2. This is followed by a taxonomy for QAC approaches in §2.3 that forms the basis for later chapters.

2.1 Problem formulation

As pointed out in §1.2, the general public was exposed to QAC around 2004, when Google began to offer a service named "Google Suggest," which provided users with query completions shown below the search box as a drop-down menu while they typed, in real time. The completions were based on exploring what others are searching for.[1] In 2011, Bar-Yossef and Kraus [2011] referred to this functionality as *query auto completion* (QAC) and proposed a straightforward approach to deal with the task of generating query completions, i.e., the well-known *most popular completion* (MPC) approach, which is based on the search popularity of queries matching the prefix entered by the user.

[1] `http://googleblog.blogspot.com/2004/12/ive-got-suggestion.html`

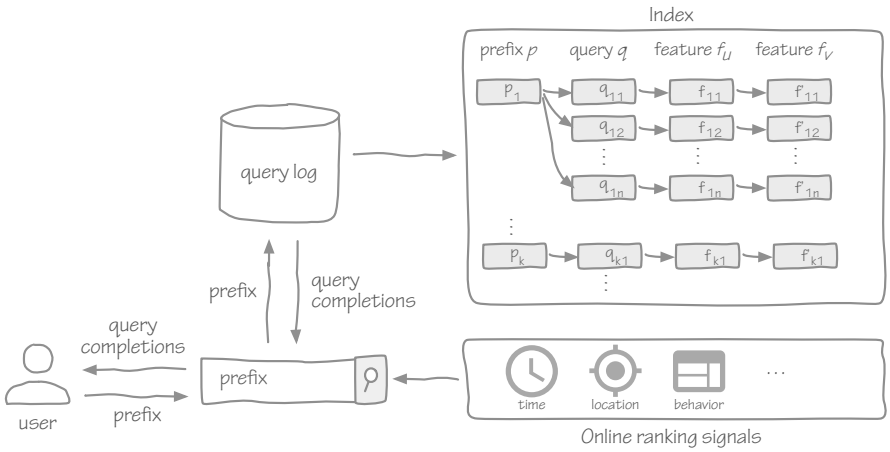


Figure 2.1: A basic QAC framework.

In essence, the QAC problem can be viewed as a ranking problem. Given a prefix, possible query completions are ranked according to a predefined criterion, and then some of them are returned to the user. Typically, a pre-computed auto-completion system is required for generating query completions corresponding to each specific prefix in advance; it stores the associations between prefix and query completions in an efficient data structure, such as prefix-trees, that allows efficient lookups by prefix matching. This index is similar to an inverted table storing a mapping from query to documents in an information retrieval system. Figure 2.1 sketches a basic QAC framework. As users' queries and interactions can be recorded by search engines, this kind of data is used for generating an index table offline; it captures the relationships between prefixes and queries. When a user enters a prefix in the search box, based on a pre-built trie-based index [Kastrinakis and Tzitzikas, 2010, Hsu and Ottaviano, 2013], a list of query completions is retrieved. Based on further re-ranking using signals determined at query time, e.g., time, location and user behavior, the user will receive a final list of query completions. In deployed systems, the list of completions returned typically has a limited length, e.g., at the time of writing, at most 4 for Baidu, 8 for Bing, 4 for Google, 10 for Yahoo and Yandex.

Next, we introduce the problem of query auto completion more formally.

Let p denote a prefix entered by a user u, i.e., a string of characters. Let $Q_I(p)$ denote a set of complete queries that extend p; it is helpful to think of $Q_I(p)$ as the set of initially retrieved completions (similar to top-k retrieval in standard document retrieval). The query auto completion problem is to find a ranking $Q_S(p)$ of the queries in $Q_I(p)$, with $|Q_S(p)| = N$, where $N > 0$ is a given cutoff denoting the number of top ranked elements of $Q_I(p)$ to be included in $Q_S(p)$, such that the cost function $\mathcal{C}(Q_S(p))$ for generating the ranking $Q_S(p)$ is optimized:

$$\mathcal{C}(Q_S(p)) = \sum_{q \in Q_S(p) \subset Q_I(p)} c(q), \tag{2.1}$$

where $c(q)$ indicates the cost of query q. Typically, for the QAC task we aim to rank the user's intended query at the top position of the ranked list of query completions. Accordingly, we model the cost function to be maximized as follows:

$$\mathcal{C}(Q_S(p)) = \sum_{q \in Q_S(p) \subset Q_I(p)} \phi(q), \tag{2.2}$$

where

$$\phi(q) = \begin{cases} \frac{1}{\text{rank}(q) \text{ in } Q_S(p)}, & \text{if } q = q' \\ 0, & \text{otherwise,} \end{cases}$$

where q' is a query that is finally submitted by user u. Algorithmic solutions to the QAC problem aim to predict the user's intended query and then return it early in the candidate list $Q_S(p)$. We refer to the items in the list $Q_S(p)$ as *query completions* or simply *completions*.

2.2 Aims of query auto completion

The aims of query auto completion are to improve the user's search experience by saving time to enter a query, avoiding spelling mistakes, discovering relevant search terms, and, importantly, formulating a better query. In a practical, experimental evaluation of QAC, the aims of QAC models are to return the user's intended query at the top position of a list of query completions.

It has been reported that most of the clicks on query completions are generated at a prefix length ranging from 3 to 12 characters while the average length of clicked query completions is close to 20 in characters for desktop search [Li et al., 2014], which is consistent with the finding that global

users of Yahoo! Search have saved more than 50% keystrokes when submitting English queries by selecting query completions for desktop search in 2014 [Zhang et al., 2015]. In addition, due to the difficulty of entering queries on mobile devices with small screens, on average, it takes a mobile user approximately 60 seconds to enter a query from a 9-key keypad although the average length of mobile queries is 2.3 words and 15.5 characters [Kamvar and Baluja, 2006]. Beyond standard desktop search, Kamvar and Baluja [2009] report that query auto completion services on mobile devices could help users to save entering up to 46% of their query characters. These numbers indicate that a query auto completion service does indeed help users save time while entering a query by reducing the number of keystrokes.

In addition, a potential aim of query auto completion is to present a user with a list of retrieved documents before they finish entering the complete query. For further reducing user effort in the search process, as a user begins to enter their query, incremental search predicts what the user is looking for and starts to show retrieved results even before they have hit the "Search" button. Based on what other people are searching for and on the content of pages indexed by the search engine, the retrieved documents are produced by scoring their relevance to the top-ranked query completion of a prefix entered by the user [Raskin, 2000]. Such a service can significantly speed up the search interaction [Li et al., 2014, Zhang et al., 2015].

2.3 A taxonomy of solutions to query auto completion

In the literature on query auto completion, several approaches have been proposed for the task. A brief comparison of recent approaches to ranking query completions is provided by Di Santo et al. [2015]. We classify existing approaches to query auto completion into two broad categories, i.e., *heuristic models* and *learning-based models*. Heuristic approaches seek to compute a score directly by considering different sources for each possible query completion that indicates how likely it is that this query would be issued. No abundant features are involved. In contrast, learning-based approaches, based on a learning algorithm, aim to extract dozens of reasonable features to capture the characteristics of each query completion. These two categories of QAC approaches can be split into three groups: time-sensitive, contextual-

Table 2.1: Representative query auto completion approaches in the literature.

		User-centered	
	Time-sensitive	Contextual-based	Demographic-based
Heuristic	[Shokouhi and Radinsky, 2012, Cai et al., 2014b, Whiting and Jose, 2014]	[Bar-Yossef and Kraus, 2011, Mitra et al., 2014, Li et al., 2015]	–
Learning-based	[Cai and de Rijke, 2016b]	[Jiang et al., 2014, Li et al., 2014, Mitra, 2015]	[Shokouhi, 2013]

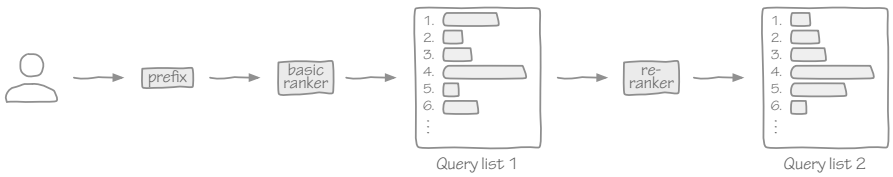


Figure 2.2: General flow of a QAC approach.

based and demographic-based. The contextual-based and demographic-based QAC approaches are mainly focused on a user's previous search history and a user's profile information, such as age and gender, respectively. As the demographic features are often hard to access, limited published work can be found on the topic. In this manner we arrive at a two-by-three grid, which we adopt to organize the publications that we survey; see Table 2.1.

As explained in §2.1, QAC approaches deal with a *re-ranking* problem [Bar-Yossef and Kraus, 2011, Shokouhi, 2013, Cai et al., 2014b, Jiang et al., 2014, Cai and de Rijke, 2016b]. That is, as shown in Figure 2.2, QAC models focus on re-ranking a ranked list of query completions that is initially returned by a basic ranker when a user enters a prefix. In this flow, the initial query list, i.e., *query list 1* in Figure 2.2, is at least as long as the final re-ranked query list, i.e., *query list 2* in Figure 2.2. Thus, we can think of QAC models as *re-rankers*. The dominant types of signals used for re-ranking are either time-related or user-centered.

Query popularity is an obvious criterion for ranking query completions. But time may affect a query's popularity, hence *time-related* aspects have been studied extensively for QAC. For instance, the popularity of queries as observed in recent trends [Whiting and Jose, 2014] or in periodic phenomena [Cai et al., 2014b] are frequently considered in QAC approaches based on query popularity. The predicted future query popularity based on time series analysis [Shokouhi and Radinsky, 2012] is another important ingredient in time-related QAC approaches. As to *personal information* of a particular user, e.g., the search context consisting of previous queries in their current session [Bar-Yossef and Kraus, 2011], their query reformulation behavior, such as adding terms [Jiang et al., 2014], and their profile context such as age and gender [Shokouhi, 2013], can all be used to infer their specific interest and search intent.

In the following chapters, i.e., Chapter 3 and Chapter 4, we describe the most representative approaches in Table 2.1. In addition, we contrast these approaches in terms of query ranking quality as well as ranking efficiency.

3

Heuristic approaches to query auto completion

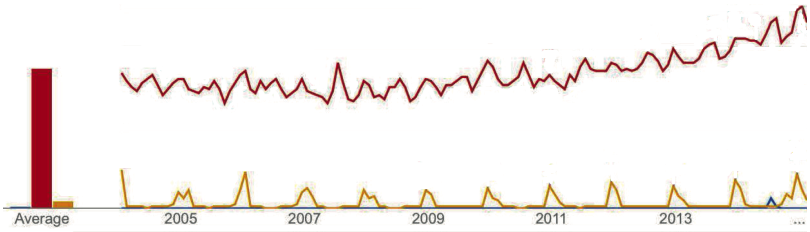
In this chapter, we describe representative heuristic query auto completion approaches in the literature that consider time-related aspects (see §3.1) and personalization (see §3.2) to compute a score for ranking query completions. Basically, these approaches aim to compute a probability $P(q_c \mid p, t, u)$ of submitting a query completion q_c after typing the prefix p, given, for instance, the time t and the user u.

3.1 Time-sensitive query auto completion models

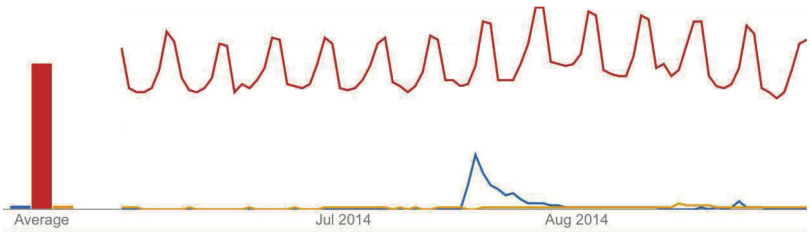
A straightforward approach to ranking query completions is to use Maximum Likelihood Estimation (MLE) based on the past popularity (i.e., frequency) of queries [Bar-Yossef and Kraus, 2011]. Bar-Yossef and Kraus [2011] refer to this type of ranking as the *most popular completion* (MPC) model:

$$MPC(p) = \arg\max_{q \in C(p)} w(q), \text{ where } w(q) = \frac{f(q)}{\sum_{q_i \in Q} f(q_i)}, \tag{3.1}$$

where $f(q)$ denotes the number of occurrences of query q in search log Q, i.e., the frequency of q, and $C(p)$ is a set of query completions that start with the prefix p. In essence, the MPC model assumes that the current query popularity distribution will remain the same as that previously observed, and hence



(a) Relative query popularity in the past few years (2004–present).



(b) Relative query popularity within 3 months (June, 2014–August, 2014).

Figure 3.1: Relative query popularity for different queries over time. Queries: "*MH17*" in blue, "*Movie*" in red and "*New year*" in yellow. The snapshot was taken on Thursday, February 12, 2015.

completions are ranked by their past popularity in order to maximize QAC effectiveness for all users on average. However, query popularity may change over time. As illustrated in Figure 3.1a and 3.1b according to Google Trends,[1] we see that query popularity strongly depends on time and that it is subject to cyclic phenomena: we see a clear burst in the frequency of the query "*MH17*" around July 18, 2014 because of the crash of flight MH17 on that date.[2] We see a yearly cycle in the frequency for the query "*New year*" and a weekly cycle for the query "*Movie*." These observations can be explored to help boost the performance of QAC models based on query popularity. Accordingly, the ranking of query completions must be adjusted to account for time-sensitive changes, which can be addressed by limiting the aggregation period of past query log evidence to increase the temporal sensitivity of QAC.

As the web increasingly becomes a platform for real-time news search

[1] `http://www.google.com/trends`
[2] `https://en.wikipedia.org/wiki/Malaysia_Airlines_Flight_17#Crash`

and media retrieval, time plays a central role in information retrieval activities; more people are turning to web search engines to find out about unpredictable emerging and ongoing events and phenomena. For instance, a substantial volume of daily top queries concern up-to-date information of recent or ongoing events [Whiting and Jose, 2014]. In addition, 15% of the daily queries to an industrial web search engine have never been seen before.[3] A substantial proportion of these queries are attributed to fresh events and real-time phenomena, rather than the long-tail of very uncommon queries [Whiting et al., 2013]. QAC approaches that are based on aggregating long-term historic query-logs are not sensitive to very fresh real-time events, which may result in poor performance for ranking query completions as newly popular queries will be outweighed by long-term popular queries, especially for short prefixes (e.g., consisting of 1 or 2 characters) [Lefortier et al., 2014].

Inspired by interesting findings from Google Trends, Shokouhi [2011] focuses on detecting seasonal queries using time-series analysis. The author depicts how a query's monthly popularity cycle can be treated as time-series and decomposed by Holt-Winters additive exponential smoothing [Goodwin, 2010] into three main components, i.e., level ($\mathcal{L}$), trend ($\mathcal{T}$) and season ($\mathcal{S}$):

$$\begin{cases} \mathcal{L}_t = \alpha \cdot (\hat{X}_t - \mathcal{S}_{t-s}) + (1-\alpha) \cdot (\mathcal{L}_{t-1} + \mathcal{T}_{t-1}) \\ \mathcal{T}_t = \beta \cdot (\mathcal{L}_t - \mathcal{L}_{t-1}) + (1-\beta) \cdot \mathcal{T}_{t-1} \\ \mathcal{S}_t = \gamma \cdot (\hat{X}_t - \mathcal{L}_t) + (1-\gamma) \cdot \mathcal{S}_{t-s}, \end{cases} \tag{3.2}$$

where α, β and γ are trade-offs in $[0, 1]$. The parameter t denotes the current time-span, and s specifies the length of the seasonal periodicity (e.g., 12 months). Furthermore, $\hat{X}_t$ represents the value of the data point at time t (e.g., in one month). Then, if the decomposed *season* component $\mathcal{S}$ and raw data $\hat{X}$ have similar distributions, the query is deemed to be a *seasonal* query. For instance, cyclic repeated events such as *Halloween* and *Christmas* happen every year, resulting in periodic spikes in the frequency of related queries (e.g., "*halloween costume*" and "*Christmas gift*"). Other queries, like "*sigir 2016*," are requests that target events that are close in time and thus will present a relatively obvious search burst than older alternatives (e.g., "*sigir 2010*"). Thus, for effective QAC, it is important to correctly identify recent popular queries

[3] `http://jungle.marketing/google-and-search-engines/search-queries/`

and make sure that users who submit such a query do indeed have a temporal information need.

Building on the work of Shokouhi [2011], Shokouhi and Radinsky [2012] propose a time-sensitive approach for query auto completion. Instead of ranking query completions by their past popularity, they apply time-series analysis and rank candidates according to the forecasted frequencies by modeling the temporal trends of queries. Unlike the ranking criterion specified in (3.1), this time-sensitive QAC approach can be formalized as follows:

$$TS(p,t) = \arg\max_{q \in C(p)} w(q \mid t), \text{ where } w(q \mid t) = \frac{\hat{f}_t(q)}{\sum_{q_i \in Q} \hat{f}_t(q_i)}. \tag{3.3}$$

Here p is the input prefix, $C(p)$ represents the list of query completions matching the prefix p, and $\hat{f}_t(q)$ denotes the estimated frequency of query q at time t in the query log Q.

In practice, the future frequency of queries $\hat{y}_{t+1}$ can simply be forecast using the single exponential smoothing method [Holt, 2004] based on a previous observation y_t and a smoothed output $\bar{y}_{t-1}$ as follows:

$$\hat{y}_{t+1} = \bar{y}_t = \lambda \cdot y_t + (1-\lambda) \cdot \bar{y}_{t-1}, \tag{3.4}$$

where y_t and $\bar{y}_t$ denote the actually observed and smoothed values for the query frequency at time t, λ is a trade-off parameter ranging between 0 and 1, and $\hat{y}_{t+1}$ is the estimated future frequency of the query at time $t+1$. As the single exponential smoothing method cannot capture the *trend* and *periodicity* of query popularity, the double and triple exponential smoothing methods [Holt, 2004] have been applied to forecast a query's future frequency [Shokouhi and Radinsky, 2012]. For instance, the double exponential smoothing method extends the model in (3.4) as follows:

$$\begin{cases} \bar{y}_t = \lambda_1 \cdot y_t + (1-\lambda_1) \cdot (\bar{y}_{t-1} + F_{t-1}) \\ F_t = \lambda_2 \cdot (\bar{y}_t - \bar{y}_{t-1}) + (1-\lambda_2) \cdot F_{t-1} \\ \hat{y}_{t+1} = \bar{y}_t + F_t, \end{cases} \tag{3.5}$$

where λ_1 and λ_2 are smoothing parameters, and the forecast $\hat{y}_{t+1}$ at time $t+1$ depends on the variable F_t, which models the linear trend of the time-series at time t, and on the smoothed frequency $\bar{y}_t$ at time t. The triple exponential

smoothing method adds a periodicity variable S_t to (3.5) as follows:

$$\begin{cases} \bar{y}_t = \lambda_1 \cdot (y_t - S_{t-T}) + (1-\lambda_1) \cdot (\bar{y}_{t-1} + F_{t-1}) \\ F_t = \lambda_2 \cdot (\bar{y}_t - \bar{y}_{t-1}) + (1-\lambda_2) \cdot F_{t-1} \\ S_t = \lambda_3 \cdot (y_t - \bar{y}_t) + (1-\lambda_3) \cdot S_{t-T} \\ \lambda_1 + \lambda_2 + \lambda_3 = 1 \\ \hat{y}_{t+1} = (\bar{y}_t + F_t) \cdot S_{t+1-T}, \end{cases} \tag{3.6}$$

where T indicates the periodicity, while λ_1, λ_2 and λ_3 are free smoothing parameters in $[0, 1]$. QAC methods based on time series analysis techniques consistently outperform baselines that use aggregated data; in other words, using more accurate query popularity predictions produced by time-series modeling leads to higher quality query suggestions [Shokouhi and Radinsky, 2012]. Strizhevskaya et al. [2012] study actualization techniques for measuring prediction accuracy of various daily query popularity prediction models using query logs, which can be helpful to query auto completion.

Still based on time series analysis, Cai et al. [2014b] propose a time-sensitive QAC method that ranks query completions by predicted popularity based on their periodicity and recent trends to detect both cyclically and instantly frequent queries. Their time-sensitive QAC method not only inherits the merits of time-series analysis for long-term observations of query popularity, but also considers recent variations in query counts. Specifically, a query q's next-day popularity $\tilde{y}_{t_0+1}(q, \lambda)$ at day $t_0 + 1$ can be predicted by both its recent trend and periodicity with a free parameter λ ($0 \leq \lambda \leq 1$) controlling the contribution of each as follows:

$$\tilde{y}_{t_0+1}(q, \lambda) = \lambda \cdot \hat{y}_{t_0+1}(q)_{trend} + (1-\lambda) \cdot \bar{y}_{t_0+1}(q)_{peri}, \tag{3.7}$$

where $\lambda = 1$ for aperiodic queries and $0 \leq \lambda < 1$ for periodic queries. In particular, the prediction $\hat{y}_{t_0+1}(q)_{trend}$ is derived from the first order derivative of q's daily count $C(q, t)$ as:

$$\hat{y}_{t_0+1}(q)_{trend} = y_{t_0-TD}(q) + \int_{t_0-TD}^{t_0+1} \frac{\partial C(q,t)}{\partial t} \mathrm{d}t, \tag{3.8}$$

where $y_{t_0-TD}(q)$ is the observed count of query q. The periodicity term $\bar{y}_{t_0+1}(q)_{peri}$ in (3.7) is smoothed by simply averaging the recent M observations y_{t_p} at preceding time points $t_p = t_0 + 1 - 1 \cdot T_q, \ldots, t_0 + 1 - M \cdot T_q$ in the query log:

$$\hat{y}_{t_0+1}(q)_{peri} = \tfrac{1}{M} \textstyle\sum_{m=1}^{M} y_{t_0+1-m \cdot T_q}(q), \tag{3.9}$$

where T_q denotes query q's periodicity. By considering these clues of query search popularity from recent trends and cyclic phenomena, a query's future popularity can be accurately forecast, which boosts the performance of time-sensitive QAC ranking models [Cai et al., 2014b]. Cai et al. [2016a] extend on this work by focusing on long-tail prefixes, which have few completions to rerank. This is achieved by learning optimal weights for balancing the contributions from time-sensitivity and personalization.

Compared to query popularity prediction based on long-term query logs, short-range query popularity estimation has received more attention. Golbandi et al. [2013] develop a regression model to detect bursting queries for enhancing trend detection. By analyzing query logs, they seek to accurately predict what the most trending query items on the Web are. Various attempts have been made to make search trend predictions more accurate with low latency relative to the actual event that sparked the trend. For instance, Kulkarni et al. [2011] classify queries into different categories based on the changes in popularity over time and show that monitoring query popularity can reveal strong signals for detecting trends in query intent. In addition, Michail et al. [2004] develop a compressed representation for time-series and propose a model for detecting bursts in query frequencies. Instead of applying time-series analysis techniques to predict a query's future popularity for query auto completion [Shokouhi and Radinsky, 2012, Cai et al., 2014b], Whiting and Jose [2014] propose several practical query completion ranking approaches, including: (1) using a sliding window to collect query popularity evidence from the past few days, (2) exploiting the query popularity distribution in the stream of the last N queries observed with a given prefix, and (3) predicting a query's short-range popularity based on recently observed trends. The predictions produced by their last N query distribution approach help achieve the best performance for query auto completion after learning the corresponding parameters online.

In addition to the aforementioned time-based clues for query auto completion, the relationship between query terms, such as semantic similarity, has also been studied in the setting of QAC. For instance, Chien and Immorlica [2005] demonstrate that queries with similar temporal patterns can be semantically related for query completion even in the absence of lexical overlap. Liu et al. [2008] introduce a unified model for forecasting query frequency,

in which the forecast for each query is influenced by the frequencies predicted for semantically similar queries. These approaches are able to produce accurate popularity predictions of queries and thus boost the effectiveness of popularity-based query completion approaches.

Table 3.1: Comparison of time-sensitive heuristic query auto completion approaches.

Angle	Short-range	Long-range	Both
Strength	Easy to implement. Sensitive to bursting events.	Clear search intent of popular search topics.	Correct search intents can be found.
Weakness	Limited information results in unclear search intent.	Difficult to deal with new search interests.	Hard to find a tradeoff between short-range and long-range search history.
Requirement	Recent search logs, i.e., search history in a recent period before querying time.	A large volume of search logs, i.e., search history in a long-term period.	Short- and long-term search logs.
Application[a]	Online shops ($\star\star\star$) News search ($\star\star\star$) Web search ($\star\star$)	Online shops ($\star$) News search ($\star$) Web search ($\star$)	Online shops ($\star\star$) News search ($\star\star$) Web search ($\star\star\star$)
Performance[a]	$\star\star$	$\star$	$\star\star\star$
Representative work	[Shokouhi and Radinsky, 2012, Golbandi et al., 2013, Whiting et al., 2013, Whiting and Jose, 2014]	[Shokouhi, 2011]	[Shokouhi and Radinsky, 2012, Cai et al., 2014b]

[a] More stars indicate relatively higher performance.

In Table 3.1, we compare the main approaches of time-sensitive heuristic query auto completion in the field of information retrieval. Generally speaking, more work has been published on the use of recent evidence than on the long-term search history for query auto completion. QAC models that

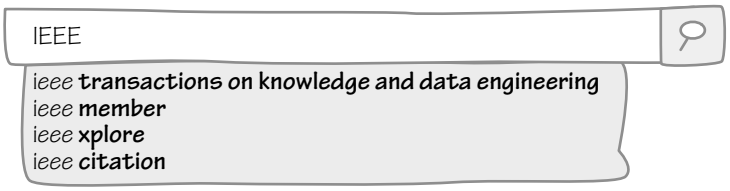
combine both short-term observations (ongoing trend) and long-term observations (periodic phenomena) achieve the best performance in terms of query completion ranking, and QAC models only based on a long-term aggregated frequency are worse than those that only exploit more recent data [Cai et al., 2016a]. As we pointed out before, QAC models based on observed query popularity, e.g., [Bar-Yossef and Kraus, 2011, Whiting and Jose, 2014], basically assume that the current or future query popularity is the same as past query popularity. In contrast, models based on predicted query popularity, such as, e.g., [Shokouhi and Radinsky, 2012, Cai et al., 2014b], take strong temporal clues into consideration as it is assumed that such information often influences the queries to be entered.

3.2 User-centered query auto completion models

User activities recorded in browsers have been used extensively to create a precise picture of the information needs of users; this concerns both long-term browsing logs [Tan et al., 2006, Liu et al., 2010b, Matthijs and Radlinski, 2011, Bennett et al., 2012] and short-term search behavior [Shen et al., 2005, Collins-Thompson et al., 2011, Jiang et al., 2011, Ustinovskiy and Serdyukov, 2013]. Such data has become an important resource for search personalization [Dou et al., 2007, Sontag et al., 2012]. In this section, we give an overview of user-centered QAC models, i.e., personalized query auto completion approaches where information from a user's search context and information about their interactions with a search engine are exploited for query auto completion.

In most work mentioned so far, query completions are computed globally. That is, for a given prefix, all users are presented with the same list of candidates. But exploiting the user's personal context has led to increases in QAC effectiveness [Bar-Yossef and Kraus, 2011, Liao et al., 2011, Santos et al., 2013, Shokouhi, 2013]. As illustrated in Figure 3.2, personalized QAC may interleave the most popular query completions with completions that a user has submitted in the past as query candidates for that particular user; see Figure 3.2a (not personalized) and 3.2b (personalized). As the two queries "*ieee transactions on knowledge and data engineering*" and "*ieee member*" have been issued before, they are ranked at the top positions of the list of suggested

(a) Query auto completion for *"IEEE"* without personalization.



(b) Query auto completion for *"IEEE"* with personalization.

Figure 3.2: Query auto completions for the prefix "*IEEE*" under different settings.

query completions in the personalized case.

The search context to be used for personalization can be collected from the current search session, e.g., the preceding queries. Bar-Yossef and Kraus [2011] treat the user's preceding queries in the current session as context and propose a context-sensitive query auto completion algorithm that produces completions of the user's input that are most similar to the context queries. In particular, they propose to output a set of completions q_c of prefix p whose vector representation v_q has the highest cosine similarity to the search context C as represented by a vector v_C:

$$q_c \leftarrow \arg\max_{q \in Q_I(p)} \frac{v_q \cdot v_C}{||v_q|| \cdot ||v_C||}, \tag{3.10}$$

where $Q_I(p)$ is the set of complete queries that extend p (see (2.1)) and $v_q \cdot v_C$ denotes the dot product of v_q and v_C. By doing so, a ranked list L_{NC} of query completions of prefix p consisting of the top k completions can be returned. At the same time, another list L_{MPC} consisting of the top k query completions can be produced based on query popularity. The final ranked list of query completions is then constructed by aggregating the two lists L_{NC}

and L_{MPC} according to a hybrid score:

$$hybscore(q_c) \leftarrow \alpha \cdot NormSim(q_c) + (1 - \alpha) \cdot NormMPC(q_c), \quad (3.11)$$

where $0 \leq \alpha \leq 1$ is a free parameter determining the weight of the normalized similarity score $NormSim(q_c)$ relative to the weight of the normalized popularity score $NormMPC(q_c)$. This model works well in cases where the immediate search context is relevant to the user's intended query and, hence, query similarity can contribute. When the search context is irrelevant, this model has to rely on query popularity.

Alternatively, the search context can also be extracted from a user's long-term search history, e.g., the most frequently submitted queries of a particular user. Cai et al. [2014b] propose to personalize QAC by scoring the query completions q_c using a combination of similarity scores $Score(Q_s, q_c)$ and $Score(Q_u, q_c)$, where Q_s refers to recent queries in the current search session and Q_u refers to queries previously issued by the same user, if available, as:

$$Pscore(q_c) = \omega \cdot Score(Q_s, q_c) + (1 - \omega) \cdot Score(Q_u, q_c), \quad (3.12)$$

where ω controls the weight of the corresponding components. This personalized QAC scheme works at the session-based and user-dependent level. The similarity scores, i.e., $Score(Q_s, q_c)$ and $Score(Q_u, q_c)$, are computed at the word level.

A third option is for the search context to be determined by a user's interactions. Li et al. [2015] pay attention to a user's sequential interactions with a QAC engine in and across QAC sessions, rather than their interactions at each keystroke of each QAC session. Through an in-depth analysis of a high-resolution query log dataset, they propose a probabilistic model that addresses the QAC task by capturing the relationship between a user's sequential behaviors at different keystrokes. Zhang et al. [2015] study implicit negative feedback from a user's interactions with a search engine, and propose a novel adaptive model *adaQAC* that adapts query auto completion to a user's implicit negative feedback about skipped query completions. This model is based on the assumption that top ranked but skipped query completions are not likely to be submitted. They propose a probabilistic model to encode the strength of the implicit negative feedback to a query completion q_c from a user u with personalization.

Table 3.2: Comparison of heuristic query auto completion approaches.

Angle	Time-sensitive	User-centered
Strength	Easy to implement.	Personalization works to improve user's search satisfaction.
Weakness	Limited information results in unclear search intent.	Difficult to detect user's new search interests. Relatively difficult to implement.
Requirement	Search logs.	Special equipment to record user's search logs and activities.
Assumption	Generalized search follows a hidden tradition.	User's interactions release strong clues of his search intents.
Issue	Relatively low robustness.	Search intent is irrelevant to user's context and may be unpredictably changed.
Performance[a]	⋆⋆	⋆⋆⋆
Representative work	[Whiting and Jose, 2014]	[Zhang et al., 2015]

[a] More stars indicate relatively higher performance.

The user-centered QAC models that we have just discussed mainly explore information from the user's search context, e.g., previous queries, or the user's interactions while engaging with QAC, e.g., typing and skipping at each keystroke. Search context-based QAC models do not need a high-resolution query log dataset to record the user's sequential keystrokes for specific query completions; however, such data is mandatory in interaction-based QAC models. In addition, it is more difficult to reproduce interaction-based QAC models, e.g., [Li et al., 2015, Zhang et al., 2015] than QAC models based on search context, e.g., [Bar-Yossef and Kraus, 2011, Cai et al., 2014b], which are relatively easy to implement. Table 3.2 provides a general comparison of heuristic time-sensitive QAC approaches with user-centered QAC methods.

3.3 Summary

In this chapter, we have surveyed work on query auto completion that ranks query completions based on the likelihood that a completion is submitted by the user. We have first discussed the major time-sensitive QAC approaches, where some time-series analysis technologies are borrowed to predict query's future popularity for generating a ranked list of query completions. The user-related query auto completion models that we have surveyed so far are mainly based on either the search context [Bar-Yossef and Kraus, 2011, Cai et al., 2014b, Cai and de Rijke, 2016a] or on user interactions [Li et al., 2015, Zhang et al., 2015] to estimate the probability of submitting a query completion. Despite their intuitiveness, these heuristic QAC approaches are always outperformed by relatively simple probabilistic methods such as the MPC model [Bar-Yossef and Kraus, 2011], regardless of their choice of how to compute the ranking scores for query completions. Experimental results from [Li et al., 2015, Zhang et al., 2015] show that incorporating detailed user interaction data, e.g., implicit negative feedback about query completions, can significantly and consistently boost the ranking quality of query completion. See Chapter 5 for results on experimental comparisons.

4

Learning-based approaches to query auto completion

The rise of learning to rank for QAC came on the heels of the introduction of a broad range of time series-based ranking principles in query popularity prediction [Shokouhi and Radinsky, 2012, Cai et al., 2014b] and an increased understanding of ranking principles based on user interactions with search engines [Li et al., 2015, Zhang et al., 2015]. In a typical (supervised) feature-based learning to rank framework for document retrieval, the training data for a specific learning algorithm (whether pointwise, pairwise or listwise [Liu, 2009]) consists of a set of query-document pairs represented by feature vectors associated with relevance labels, and the goal is to learn a ranking model by optimizing a loss function [Liu, 2009]. In contrast, in a learning to rank-based framework for QAC, the input is a prefix $p_i = \{char_1, char_2, \ldots, char_{n_c}\}$, i.e., a string of n_c characters. Generally, there is a pool $Q_I(p_i)$ consisting of query completions that start with the prefix p_i and that could be issued after entering p_i. These completions can simply be sorted by popularity. Each completion in $Q_I(p_i)$ can be represented as a prefix-query pair feature vector v_q. In a learning to rank framework for QAC, the model is trained on prefix-query pairs, which can be associated with binary labels, i.e., "submitted" or "non-submitted," similar to the "relevance" label of query-document pairs in learning to rank for document retrieval. Table 4.1 provides a high

Table 4.1: Comparison of learning to rank for document retrieval (DR) vs. query auto completion (QAC). In a document retrieval task, for a given query q_i, each of its associated documents d can be represented by a feature vector $v_d = \Phi(d, q)$, where Φ is a feature extractor; $m^{(i)}$ is the number of documents associated with query q_i, i.e., D. In a QAC task, for a given prefix p_i, each candidate query completion q can be represented by a feature vector $v_q = \phi(p, q)$, where ϕ is a feature extractor; $n^{(i)}$ is the number of candidate query completions associated with prefix p_i), i.e., $Q_I(p_i)$.

DR	Input	Query q_i
	Ranking candidates	Documents $d \in D$
	Candidate features	TF, IDF, BM25, etc.
	Output	A ranked list of documents
	Ground truth	Multi-level relevance labels, i.e., 0, 1, 2
	Major metrics	Average Precision (AP), Precision@K (P@K), etc. (See [Manning et al., 2008])
QAC	Input	Prefix p_i
	Ranking candidates	Queries $q \in Q_I(p_i)$
	Candidate features	Popularity, length, position in session, etc.
	Output	A ranked list of queries
	Ground truth	Binary labels: 1 for submitted query and 0 for the others.
	Major metrics	Reciprocal Rank (RR), Success Rate@K (SR@K), etc. (See §5.2)

level comparison of learning to rank for document retrieval with learning to rank for QAC.

As illustrated in Figure 4.1, learning-based QAC approaches mainly focus on extracting useful features to seek a correct prediction of the user's intended query. Such QAC models eventually learn a ranking function from the extracted features. We split learning-based QAC approaches into two groups according to where the features used come from, i.e., time-related characteristics in §4.1 and user interactions in §4.2.

4.1 Learning from time-related characteristics for query auto completion

Time-related characteristics used for learning to rank query completions mainly relate to popularity-based features from two sources: previous obser-

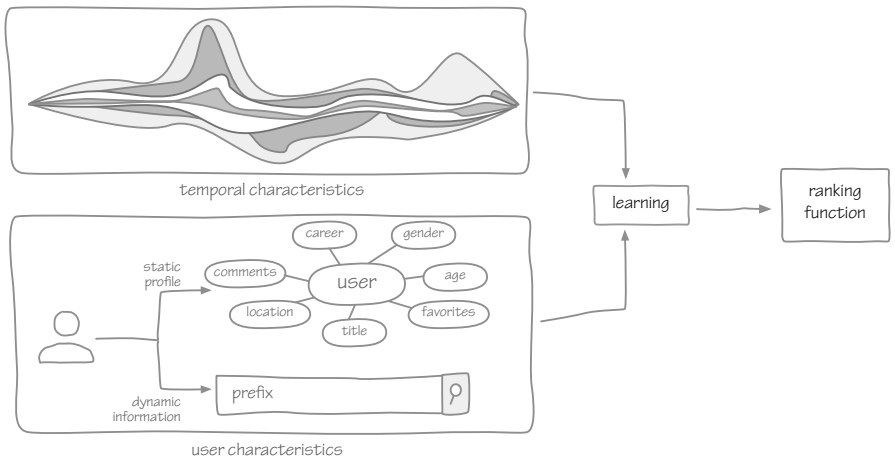


Figure 4.1: A general framework for learning-based query auto completion approaches.

vations and future predictions. Such signals have been used in heuristic query auto completions [Cai et al., 2014b, Whiting and Jose, 2014]. After that, Cai and de Rijke [2016b] propose to generate a ranking function by learning both observed and predicted query popularity. Similar to [Whiting and Jose, 2014], query popularity is determined from recent days, for instance, from the recent 1, 2, 4, or 7-day interval preceding the prefix submission time as well as from the entire query log. In addition, the predicted query popularity for learning is generated by considering the recent trend as well as the cyclic behavior of query popularity. Cai and de Rijke [2016b] also extract such features for so-called homologous queries of a query completion.[1] When extracting features from homologous queries for a candidate query completion, a discount parameter that weighs the contribution of homologous queries to score of the candidate query completion, is estimated based on the similarity between the homologous queries and the candidate query completion.

In addition, semantic features have been developed, including the semantic relatedness of terms in queries and of queries in sessions. For instance, the likelihood ratio (LLR) between two query terms has been introduced to

[1]Given a query q, a *homologous* query either extends q or is a permutation of the terms that make up q.

capture the semantic relatedness of term pairs inside a query. Specifically, assuming that there are two terms $term_1$ and $term_2$ in a query, the LLR score for this term-pair is calculated as:

$$LLR(term_1, term_2) = -2 \log \frac{L(term_2 \mid \neg term_1)}{L(term_2 \mid term_1)}, \tag{4.1}$$

where $L(term_2 \mid \neg term_1)$ denotes the number of queries containing $term_2$ but without $term_1$, and $L(term_2 \mid term_1)$ indicates the volume of queries containing both $term_1$ and $term_2$.

Besides, following [Chien and Immorlica, 2005], query relatedness using temporal correlations can be captured. In other words, two queries are assumed to be semantically related in this sense if their popularity behaves similarly over time. Thus, Pearson's correlation coefficient can be employed, to capture this notion of similarity between candidate q_c and previous query q_x in session as follows:

$$r(q_c, q_x) = \frac{1}{n_u} \sum_{i=1}^{n_u} \left(\frac{q_{c,i} - \mu(q_c)}{\delta(q_c)} \right) \left(\frac{q_{x,i} - \mu(q_x)}{\delta(q_x)} \right), \tag{4.2}$$

where the frequency of query q_c (or q_x) over n_u days is an n_u-dimensional vector $q_c = (q_{c,1}, q_{c,2}, \ldots, q_{c,n_u})$ with $\mu(q_c)$ and $\delta(q_c)$ indicating the mean frequency and the standard deviation of its frequency, respectively. The correlation $r(q_c, q_x)$ of two queries q_c and q_x is a standard measure of how strongly two queries are linearly related. It always lies between $+1$ and -1, with $+1$ implying a positive linear relationship between them and -1 a negative linear relationship.

Finally, recently introduced ranking models that incorporate features extracted from user behavior in sessions [Jiang et al., 2014], i.e., the term-, query- and session-level features, generally achieve better performance than the baseline approaches that represent the state-of-the-art. In particular, they significantly outperform the state-of-the-art for long prefixes consisting of 4 or 5 characters but only present a modest improvement over the state-of-the-art for short prefixes. In general, Jiang et al.'s experimental results confirm that observed and predicted time-sensitive popularity features of query completions and of their homologous queries help to generate better QAC rankings in a learning to rank approach.

4.2 Learning from user interactions for query auto completion

There is more published work on learning-based QAC approaches that focus on features based on user related characteristics than work that focuses on features based on time-related characteristics. User centered features typically originate from one of two sources: (1) from user behavior in search sessions [Shokouhi, 2013, Jiang et al., 2014, Mitra, 2015], which can capture a user's search interests and how they change queries during search sessions; or (2) from high-resolution query logs [Li et al., 2014] that capture user feedback like query typing, skipping, viewing and clicking as feedback in an entire QAC process.

It has been observed that the popularity of certain queries may vary drastically across different demographics [Weber and Castillo, 2010] and users [Shokouhi, 2013]. To better understand the role played by user profiles and search contexts, Shokouhi [2013] presents a supervised framework for personalized ranking of query auto completions, where several user-specific and demographic-based features are extracted for learning. In particular, for user-specific context features, Shokouhi [2013] investigates the effectiveness for personalizing query auto completion of features developed from the search context consisting of both the short-term and long-term search history of a particular user. To define short-term history features, the queries previously submitted in the current search session are used. To define long-term history features, the entire search history of a user is considered. Then, the similarity features of query completions can be measured by n-gram similarity to those queries in the context. Similar features have been used in previous work on personalized re-ranking of web search results [Xiang et al., 2010, Teevan et al., 2011, Bennett et al., 2012] and query suggestion [Cao et al., 2008, Mei et al., 2008]. For demographic-based features, the information of user's age, gender and location (e.g., zip-code) is considered. Based on a learning algorithm, e.g., [Burges et al., 2011], the experimental results show that demographic features such as location are more effective than others for personalizing query auto completion. In addition, adding more features based on a user's demographic properties and search history leads to further boosts in performance of personalized query auto completion.

Similarly, focusing on search context in the current search session, Jiang

et al. [2014] investigate the feasibility of exploiting the search context consisting of previously submitted queries to learn user reformulation behavior features for boosting the performance of query auto completion. An in-depth analysis is conducted to see how users reformulate their queries. Three kinds of reformulation-related features are considered, i.e., term-level, query-level and session-level features, which can capture clues as to how users change their preceding queries during query sessions. For instance, for term-level features, the ratio of used terms is introduced and formulated as

$$f_{termratio} = \frac{|S_{used}(q_T)|}{|S(q_T)|}, \tag{4.3}$$

where q_T is a query completion following a sequence of $T-1$ queries in a session, $S(q_T)$ is a set of terms in query q_T and $S_{used}(q_T)$ is formed by removing the terms not appearing in the previous T queries from $S(q_T)$. For query-level features, the average cosine similarity is defined as

$$f_{cossim} = \frac{1}{T-1}\sum_{i=1}^{T-1} sim_{cos}(q_i, q_T). \tag{4.4}$$

Finally, for session-level features, the time duration is taken into account and their trends are formulated as a feature:

$$f_{timetrend} = (t_{T-1} - t_{T-2}) / \frac{1}{T-2}\sum_{i=1}^{T-2}(t_{i+1} - t_i), \tag{4.5}$$

where t_{T-1} indicates the duration of time that users stay on the search results for query q_{T-1}, which might affect how the next query is reformulated. Compared to traditional context-aware QAC methods that directly model term similarities or query dependencies, this learning-based QAC approach tries to learn how users change preceding queries. This use of user reformulation behavior during query sessions results in significant improvements over the performance of start-of-the-art context-aware query completion or suggestion approaches [He et al., 2009, Bar-Yossef and Kraus, 2011, Liao et al., 2011].

Mitra [2015] builds on the insight that search logs contain lots of examples of frequently occurring patterns of user reformulations of queries. They represent query reformulations as vectors, which is achieved by studying the distributed representation of queries learnt by deep neural network models, like the convolutional latent semantic model (CLSM) [Shen et al., 2014]. By

doing so, queries with similar intents are projected to the same neighborhood in the embedding space. Based on distributed representation of queries learnt by the CLSM model, topical similarity features measured by the cosine similarity in (4.6) between the CLSM vectors corresponding to a query completion q_1 and the previous queries in the same search session, e.g., q_2 in (4.6), are computed and used as features for learning in a QAC ranking model:

$$Sim(q_1, q_2) = cosine(v_1, v_2) = \frac{v_1^T v_2}{||v_1|| \cdot ||v_2||}, \tag{4.6}$$

where v_1 and v_2 are the CLSM vectors corresponding to q_1 and q_2, respectively. In addition to topical similarity features determined by the CLSM model, CLSM reformulation features, represented by low-dimensional vectors that map syntactically and semantically similar query changes close together in the embedding space, are used in the supervised learning model. Interestingly, Mitra and Craswell [2015] focus on recommending query completions for the prefixes that are sufficiently rare as previous QAC approaches mostly focus on recommending queries for prefixes that have frequently been seen by the search engine. In particular, a candidate generation approach using frequently observed query suffixes mined from historical search logs is proposed by adopting the CLSM [Shen et al., 2014] trained on a prefix-suffix pairs dataset. The training data for the CLSM model is generated by sampling queries in the search logs and segmenting each query at every possible word boundary. This approach provides a solution to recommend queries that have not been seen during the training period, an issue that cannot be addressed by previously proposed heuristic algorithms, e.g., [Bar-Yossef and Kraus, 2011, Shokouhi and Radinsky, 2012, Cai et al., 2014b, Whiting and Jose, 2014, Cai et al., 2016a], nor by learning-based QAC approaches, e.g., [Shokouhi, 2013, Jiang et al., 2014, Cai and de Rijke, 2016b], because these proposed models ultimately depend on query popularity as a first step: only previously seen, sufficiently frequent queries can be returned.

High-resolution query logs capture more detailed information than traditional logs, which typically contain the timestamp of a query, the anonymous user ID and the final submitted query. High-resolution logs may capture user interactions at every keystroke and associated system response. Based on observations from such rich data, Li et al. [2014] report that, in QAC, users often skip query completion lists even though such lists contain the final sub-

Table 4.2: Comparison of learning-based query auto completion approaches.

Angle	Time-sensitive	User-centered
Strength	Easy to implement; effective for fresh events.	Dynamic intent is released by interaction; static interest can be clearly captured by user profile.
Weakness	Limited number of features; hard to predict unpopular completions; sensitive to the unpredictable events	Poor performance if intent shifts; hard to record a detailed QAC engagement; sensitive to user search habit; relatively difficult to reproduce.
Requirement	Query logs with timestamp.	Personalized user information, e.g., profile and search context.
Performance[a]	⋆	⋆⋆
Representative reference	[Cai and de Rijke, 2016b]	[Jiang et al., 2014, Mitra, 2015]

[a] More stars indicate relatively higher performance.

mitted query and most of the clicked queries are concentrated at top positions of the query completion list. Based on these observations, they propose two types of bias related to user behavior, i.e., a *horizontal skipping bias* and a *vertical position bias*, that are crucial for relevance prediction in QAC. First, horizontal skipping bias is introduced if a query completion does not receive a click because the user does not stop and examine the suggested list of query completions, regardless of the relevance of the query completion. Vertical position bias relates to the exact position of a query completion in the list of completions and the fact that queries that are ranked higher rank tend to attract more clicks regardless of their relevance to the search intent.

Based on these observations, the authors propose a two-dimensional click model to interpret the QAC process with particular emphasis on these two types of behaviors. The phenomenon of position bias has previously been observed for traditional document retrieval [Granka et al., 2004, Craswell

et al., 2008], and several solutions have been proposed in the setting of click models [Richardson et al., 2007, Dupret and Piwowarski, 2008, Chapelle and Zhang, 2009, Liu et al., 2010a, Zhu et al., 2010, Zhang et al., 2011, Chuklin et al., 2015]. Newly extracted features from high-resolution query logs, which accurately explain user behavior during QAC engagement, reveal that people tend to stop and look for query completions when they are typing at word boundaries, which is consistent with the findings in [Mitra et al., 2014].

4.3 Summary

In this chapter, we have summarized learning-based query auto completion approaches that explore features from time-related characteristics and from user-specific characteristics. Such approaches build upon advances in time series analysis by predicting the future popularity of queries and on user behavior analysis by understanding the interaction data in a QAC engagement. We summarize the main strengths and weaknesses of the learning-based QAC approaches that we have discussed in Table 4.2. So far, user-centered QAC approaches have received more attention than time-related methods. We believe that this is due to the rich data recorded, based on which a user's search intent can often be correctly predicted, resulting in good QAC performance.

5

Evaluation

A *good* list of query completions is one that returns the user's intended query at the top of the list, even when the user input is short, consisting of very few characters only. In this chapter, we present experimental designs that have been used in the setting of QAC. In particular, we describe publicly available query log datasets in §5.1, where the exact query content is provided to guarantee the prefix can be simulated. In addition, the most prominent metrics used for query auto completion evaluation are presented in §5.2. We highlight some important lessons learned in §5.3.

5.1 Datasets

A lot of research on QAC takes a query log as its starting point. Each record in such a log contains at least three main components: a submitted query, a user ID (or session ID) and a timestamp. The timestamp and the user ID (or session ID) are used for extracting time-sensitive and user-specific characteristics, respectively, while the query can be used to generate query prefixes and to count the frequency in the query logs. Some additional information, e.g., clicked URLs and their ranks, may be available too. Such data can be used to

Table 5.1: An example of a search session in the AOL query log dataset.

User ID	Query	Timestamp	Rank	URL
2722	goodyear	2006-05-21 09:10:29	1	http://www.goodyear.com
2722	hyundaiusa	2006-05-21 09:28:54	1	http://www.hyundaiusa.com
2722	order hyundai parts	2006-05-21 09:51:02	2	http://www.hypartswholesale.com
2722	order hyundai parts	2006-05-21 09:51:02	9	http://www.the-best-source.com
2722	order hyundai parts	2006-05-21 09:51:02	6	http://www.racepages.com

infer users' search intents. Table 5.1 shows an example of a search session[1] in the well-known AOL query log dataset [Pass et al., 2006]. In this particular example, the user (with ID 2722) submitted three queries in total. The first two queries, "*goodyear*" and "*hyundaiusa*", both generate one clicked URL each, which is ranked at position 1 in the returned lists of URLs. The last query, "*order hyundai parts*," has three clicked URLs, ranked at position 2, 9 and 6, respectively.

As far as we know, there are three suitable publicly available query logs to support experimentation with QAC: the AOL dataset [Pass et al., 2006], the MSN dataset [Craswell et al., 2009], and the SogouQ dataset.[2] These logs have been widely used for QAC training and evaluation. For instance, the AOL dataset is commonly used in time-sensitive QAC approaches [Cai et al., 2014b, Whiting and Jose, 2014, Cai and de Rijke, 2016b] as a relatively long period of time is covered, and in personalized QAC methods [Bar-Yossef and Kraus, 2011, Shokouhi, 2013]. The MSN dataset is preferred in [Cai and de Rijke, 2016b, Cai et al., 2016b] for personalizing query auto completion. The SogouQ dataset has been used by Whiting and Jose [2014] and its use is encouraged in other tasks, e.g., the NTCIR INTENT[3] and iMine tasks, for search intent and subtopic mining [Sakai et al., 2013, Yamamoto et al., 2016]. Other query log datasets from modern commercial search engines, that record more search details but are not publicly available for privacy or competitiveness reasons, have also been applied for research [Mishne and de Rijke, 2006, Huurnink et al., 2010, Shokouhi and Radinsky, 2012, Jiang

[1] Segmented by a fixed time window of 30 minutes of inactivity [Jones and Klinkner, 2008, Lucchese et al., 2011].

[2] `http://www.sogou.com/labs/dl/q.html`

[3] `http://research.nii.ac.jp/ntcir/index-en.html`

Table 5.2: Statistics of publicly available query log collections, i.e., the AOL dataset (AOL), the MSN dataset (MSN) and the SogouQ dataset (SogouQ), for query auto completion evaluation.

Statistic	AOL	MSN	SogouQ
Language	English	English	Chinese
Start date	2006-03-01	2006-05-01	2008-06-01
End date	2006-05-31	2006-05-31	2008-06-30
# Days	92	31	30
# Records	36,389,567	14,921,285	43,545,444
# Queries	10,154,742	8,831,281	8,939,569
# Users	657,426	–	9,739,704
# Sessions[a]	–	7,470,916	25,530,711
# URLs	1,632,788	4,975,897	15,095,269

[a] Split by 30 minutes of inactivity.

et al., 2014, Li et al., 2014, 2015, Zhang et al., 2015].

The AOL dataset is one of the few datasets that contain actual queries and are sufficiently large for researchers to be able to establish statistically significant differences between competing QAC approaches. The queries in the AOL log were sampled between March 1, 2006 and May 31, 2006. The MSN logs were recorded for one month in May 2006 from the MSN Search engine. An early version of the SogouQ dataset was released in 2008 but it was replaced by a bigger dataset in 2012. Private information has been removed as well as illegal query sessions. Basic descriptive statistics of these three publicly available query logs are provided in Table 5.2. Although these logs facilitate research on query auto completion in information retrieval, they are limited in the sense that they do not contain information about detailed user interactions with a QAC engine.

Recently, a high-resolution QAC query log that records every keystroke in a QAC session has been collected from the Yahoo! search engine [Li et al., 2014]. In this private dataset, the records contain information of each QAC session, including the anonymous user ID, the final clicked query, the keystroke a user has entered, the corresponding timestamp of such a keystroke as well as the top 10 suggested query completions for the prefix entered by the

user. In addition, the user's previous queries in the session are also recorded, which can be used for personalization. Based on such a rich query log dataset, a user's sequential behavior [Li et al., 2015] and feedback on query completions suggested in the QAC process [Zhang et al., 2015] can be exploited for boosting the performance of today's QAC approaches.

5.2 Evaluation metrics

To evaluate the effectiveness of QAC rankings, Mean Reciprocal Rank (MRR) has become a standard measure. For a prefix p of a query q in the query set Q, a list of matching QAC candidates $Q_I(p)$ and final query q' submitted by the user are given. Then, the Reciprocal Rank (RR) for this prefix is computed as follows:

$$RR = \begin{cases} \frac{1}{\text{rank of } q' \text{ in } Q_I(p)}, & \text{if } q' \in Q_I(p) \\ 0, & \text{otherwise.} \end{cases} \tag{5.1}$$

Finally, the MRR score is computed as the average of all RR scores for the tested prefixes of queries in Q. From this formula, it is clear that MRR is a precision-oriented metric.

The choice of MRR as a performance metric is common in settings that are concerned with finding a single known solution. However, because of the issues reported in [Hofmann et al., 2014] on the formulation of MRR (averaging over a non ratio-scale), a less problematic metric, i.e., *success rate at top K* (SR@K), is also being used for QAC evaluation, which denotes the average ratio of the actual query that can be found in the top K query completion candidates over the test data. This metric is widely used for tasks whose ground truth consists of only one instance, such as query completion [Jiang et al., 2014].

MRR treats all test prefixes equally, i.e., it is calculated by averaging all RR scores. Hence, it does not consider the difficulty of completing a particular prefix. For instance, completing the prefix "c" is more difficult than completing the prefix "z" as a larger volume of query completions start with the prefix "c" than with the prefix "z." Bar-Yossef and Kraus [2011] introduce a weighted MRR to evaluate the performance of QAC approaches, which is achieved by weighing the RR scores according to the number of completions available for the prefix.

Furthermore, if we suppose that the user would submit a query by clicking a completion once it is ranked at the top position of the QAC list, we can use the metric Minimal Keystrokes (MKS) as well to indicate the minimal number of keypresses that the user has to type before a query is submitted by clicking a query completion [Duan and Hsu, 2011].

So far, the QAC metrics used are not specifically designed to reflect user satisfaction. Following work in web search evaluation [Chapelle et al., 2009], which involves user modeling to measure the quality of search results, Kharitonov et al. [2013] introduce two user model-based evaluation metrics to evaluate query suggestions, *pSaved* and *eSaved*, which could be used for QAC evaluation. *pSaved* is defined as the probability of using a query suggestion while submitting a query. *eSaved* equates to the normalized amount of keypresses a user can avoid due to the deployed query suggestion mechanism. In more detail, *pSaved* is formally defined as follows:

$$pSaved(q) = \sum_{i=1}^{|q|} I(S_{ij}) P(S_{ij} = 1) = \sum_{i=1}^{|q|} P(S_{ij} = 1), \tag{5.2}$$

where the query to be submitted is q, S_{ij} is a binary variable representing if the user is satisfied with the j-th suggestion shown for the prefix $q[1, \ldots, i]$, $I(S_{ij})$ equals 1 if the user clicks the query suggestion and 0 otherwise, and $P(S_{ij})$ defines the probability of user satisfaction, which is parameterized by the user model. Formally, *eSaved* can be calculated using the following expression:

$$eSaved(q) = \sum_{i=1}^{|q|} \left(1 - \frac{i}{|q|}\right) \sum_{j} P(S_{ij} = 1). \tag{5.3}$$

The user model can be estimated from logged user behavior.

For evaluating the performance of methods for diversified QAC, the official evaluation metrics in the diversity task of the TREC 2013 Web track [Collins-Thompson et al., 2013], e.g., ERR-IA [Chapelle et al., 2009], α-nDCG [Clarke et al., 2008], NRBP [Clarke et al., 2009] and MAP-IA [Agrawal et al., 2009] can be borrowed. We take the metric α-nDCG at cutoff N as an example for evaluating the diversity of a ranking of query completions, which extends the traditional nDCG metric [Järvelin and Kekäläi-

nen, 2002] for aspect-specific rankings according to:

$$\alpha\text{-}nDCG@N = Z_N \sum_{i=1}^{N} \frac{\sum_{a \in A_p} g_{i|a} (1-\alpha)^{\sum_{j=1}^{i-1} g_{j|a}}}{\log_2(i+1)}, \tag{5.4}$$

where a is an aspect in the set of query aspects A_p, $g_{i|a}$ denotes the aspect-specific gain of the i-th 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. Determining the optimal QAC list (based on ground truth) is an NP-complete problem and thus a greedy approximation is used in practice. The α-nDCG@N score is commonly computed at a trade-off $\alpha = 0.5$, which reflects the possibility of assessor error in an unbiased manner, and all rewards are discounted by a log-harmonic discount function of the rank. Generally, these diversity metrics reward newly-added aspects and penalize redundant aspects of QAC candidates. We refer the reader to the references provided for details on how these metrics are computed.

5.3 Main experimental findings

In this section, we provide a high-level overview of experimental outcomes related to the query auto completion problem. The aim is to provide insights on which QAC strategies to use in which circumstances.

In the experiments of most published work, where real inputs entered by users into a search box are unavailable, user inputs are simulated by considering all possible prefixes of the query. Based on studies where a high-resolution query log dataset is available, such as [Li et al., 2014], where users' reactions and the QAC system's response to each particular user input are provided, the following important lessons have been learned:

- QAC models that exploit user behavioral information, e.g., implicit negative feedback towards unselected query completions [Zhang et al., 2015] and skipping behavior towards query completions [Li et al., 2014], can outperform popularity-based QAC approaches, e.g., [Bar-Yossef and Kraus, 2011, Whiting and Jose, 2014] as user's real-time search intent can be correctly predicted by analyzing their interactions.
- Insights from click models in the field of document retrieval can be further explored in a QAC framework. An exploratory study has been

conducted in [Li et al., 2014, 2015], where behavioral information is incorporated into a learning-based method to further improve the QAC performance.

For privacy and competitiveness reasons, rich information about user interactions with a QAC system is not publicly available. Most researchers, whether academic or industrial, working on QAC implement their experiments on publicly available query logs, i.e., the AOL, MSN and SogouQ logs. Using these datasets, the following are important lessons learned:

- Personalized QAC [Cai et al., 2014b], which is based on a user's short- and long-term search history, can boost the performance of a generic QAC approach as search context can be explored to infer user's particular interests.
- Query popularity-based QAC approaches generate better rankings of query completions by predicted query popularity than by observed query popularity [Shokouhi and Radinsky, 2012, Cai et al., 2014b], as search popularity changes over time and the predictions can capture the recent trends of search popularity.

In general, user-centered QAC strategies have been proven to be effective as evidenced by a series of personalized query auto completion approaches [Bar-Yossef and Kraus, 2011, Shokouhi, 2013, Cai et al., 2014b, Jiang et al., 2014, Li et al., 2015, Zhang et al., 2015]. For instance, for personalized query auto completion, the short-term search context of a user, such as the previous queries in the session, is more important than their long-term search history [Cai et al., 2014b]. For learning user reformulation behavior in query auto completion, query-level features are the most significant ones, e.g., query frequency and query co-occurrence [Jiang et al., 2014]. When a user's feedback concerning a list of suggested query completions is available, user behavior, like skipping the whole query completion list and skipping unselected query completions, can be injected into a static QAC model to significantly and consistently boost the accuracy of static QAC models [Li et al., 2015, Zhang et al., 2015].

5.4 Summary

In this chapter, we have detailed currently established methodologies for evaluating ranked lists of query suggestions. In particular, we have described publicly available query logs that are often used for query auto completion in information retrieval. In addition, we introduced the most widely used metrics for QAC evaluation. In the next two chapters, we will discuss some practical issues related to query auto completion for information retrieval, focusing on efficiency issues in Chapter 6 and on presentation issues in Chapter 7.

6

Efficiency and robustness

Query auto completion (QAC) makes for a nice user experience [Zhang et al., 2015], but it can be a challenge to implement efficiently. In many cases, a practical implementation of QAC requires the use of efficient algorithms and data structures for fast lookups. Unlike studies related to efficiency issues in traditional keyword search [Ji et al., 2009, Li et al., 2011] and document retrieval [Bast et al., 2007, Francès et al., 2014], relatively few investigations have targeted efficiency in query auto completion. After summarizing previous work related to efficiency issues in QAC, we find two main types of practical issue in QAC, i.e., computational efficiency and error-tolerance.

6.1 Computational efficiency in query auto completion

Before ranking query completions, indexing hundreds of millions of distinct queries is required to guarantee a reasonable coverage corresponding to an input prefix. A trie, also known as a prefix-tree, is a simple data structure that supports fast query completion lookups. A trie is a tree data structure that allows strings with similar character prefixes to use the same prefix and store only the tails as separate data [Cormen et al., 2009]. Figure 6.1 presents an example trie with the queries “sit,” “see,” “sea,” “seafox,” “seafood,” “tree”

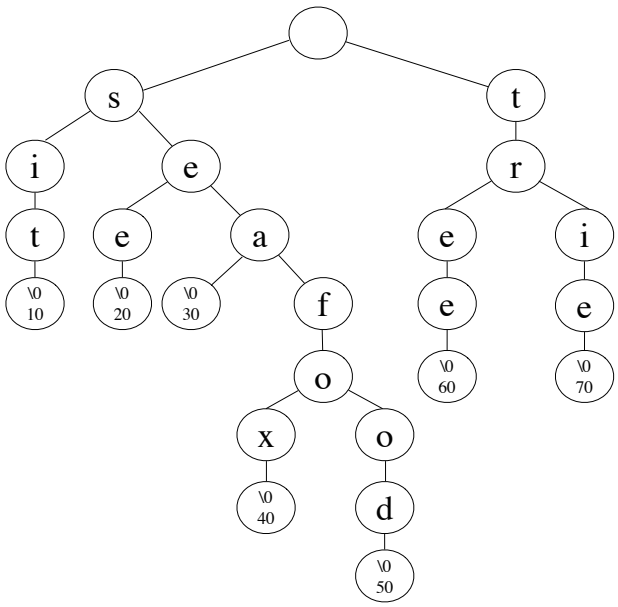


Figure 6.1: An example trie.

and "trie" as well as their corresponding values, e.g., the frequency. For simplicity, this example only indexes single-word queries. As we can see from Figure 6.1, the root is associated with an empty character and values are normally not associated with every node but with leaves only. Each completed query is associated with a sequence of nodes, including the regular nodes with characters and a final node, characterized with "\0", to store the values, e.g., the query frequency. For generating top-k completions after inputting a prefix, we do not need to traverse the whole trie for the particular prefix, because each query completion is associated with a value, e.g., popularity, and we can sort the queries by their popularity offline before ranking. Thus, a complexity of $O(k)$ is needed for lookups, which is highly important to improve the efficiency, especially for short prefixes like c and s, etc.

Hsu and Ottaviano [2013] focus on the case where the completion set is so large that compression is needed to fit the data structure in memory. The authors present three trie-based data structures to handle such cases, i.e., Completion Trie, RMQ Trie and Score-Decomposed Trie. Each has different space vs. time vs. complexity trade-offs. Each has the potential to sig-

nificantly reduce the storage of large volumes of query data but still enable efficient operations over it. Trie-based indexes of queries need to be maintained continuously. They support fast query completion lookups but need a considerable amount of memory. To address the memory problem of tries, a ternary tree is a good solution where each trie node is represented as a tree-within-a-tree. Typically, the ternary tree is stored on a server before testing. Then, in a real-time QAC process, the client will send a prefix to the server to get query completions based on the constructed ternary tree. Matani [2011] presents an algorithm that takes time proportional to $O(k \log n)$ for providing the user with the top-k ranked query completions out of a corpus of n possible suggestions based on a ternary tree for completing the prefix that the user has entered.

Trie-based data index schemes suffer from an inherent problem. The number of prefixes is huge for the first few characters of the query and it is exponential in the alphabet size. This will result in a slow query response even if the entire query approximately matches only few prefixes. In addition, efficiency critically depends on the number of active nodes in the trie. Xiao et al. [2013] study a version of the QAC problem that tolerates errors in user's input using edit distance constraints; they propose a novel neighborhood generation-based algorithm, i.e., *IncNGTrie*, that can achieve up to two orders of magnitude speedup over existing methods for the query auto completion problem. Their proposed algorithm only needs to maintain a small set of active nodes in a trie, thus saving both space and time to process the query. In addition, they propose optimization techniques to remove duplicates and reduce the index size by merging common data strings and common subtrees.

Instead of keeping query completions in one trie, Kastrinakis and Tzitzikas [2010] propose to partition a trie into two or more subtries to make query completion services more scalable and faster. Thus, they use a forest of tries for query auto completion. In particular, each subtrie contains queries whose starting characters belong to a specific set of characters assigned to this trie. The key challenge is to partition the trie, which involves collecting all possible prefixes of a fixed number of characters from the query log and counting their frequency. A subtrie is created as a partition and grows until its size becomes larger than a desired partition capacity. After that, a new partition is created and so on. Experimental results show that the proposed

partitioning scheme allows one to increase the number of query completions and speed up their computation at query time.

Similar to the work on learning-to-rank for document retrieval [Liu, 2009], where features of query-document pairs are generated offline, learning-based QAC approaches [Shokouhi, 2013, Jiang et al., 2014, Cai and de Rijke, 2016b] extract the features of prefix-query completions during training, i.e., before online testing for effective QAC performance. Table 6.1 shows examples of a typical data format for learning-to-rank in document retrieval and for learning-to-rank in query auto completion. As shown in Figure 6.1a, the features are meant to capture the relationships between the query and its associated documents, such as query frequency in the document title, word frequency in the document abstract and query frequency in the whole document. Such features are used for measuring the similarity between a query and a document. The labels in Table 6.1a indicate the relevance of a document to a query. In a learning-based QAC framework as shown in Table 6.1b, additional features are used for measuring the possibility of submitting a query completion after entering the prefix and the label denotes whether the provided query completion is submitted or not after entering the prefix. Examples include features related to the popularity of a query completion in the query log, the similarity of a query completion to the search context as determined by the session, and the similarity of a query completion to a user's interest contained in their profile. Such features are generated offline for training a ranking model. In an online test setting, the main time cost for ranking query completions are lookups for matching such query completions and calculating a final ranking score based on a trained model, which generally consumes little time [Jiang et al., 2014, Cai and de Rijke, 2016b].

A number of authors have worked on compact data structures for indexing so as to facilitate instant display of completions of the last query word entered by the user, with every single keystroke, resulting in very fast response times [Bast and Weber, 2006, Bast et al., 2008, Ji et al., 2009].

To the best of our knowledge, the QAC services provided by commercial web search engines only present completions of queries that have been submitted before. In other words, the query completions that they offer all come from what real people actually searched before.[1] Accordingly, all the features

[1] `http://searchengineland.com/how-google-instant-`

Table 6.1: Examples of data formats for learning-to-rank in document retrieval and in query auto completion.

(a) An example of data format for learning-to-rank in document retrieval.

						⋮			
QueryID_i	DocID	Feature_1	Value_1	Feature_2	Value_2	…	Feature_m	Value_m	Label
QueryID_i	DocID	Feature_1	Value_1	Feature_2	Value_2	…	Feature_m	Value_m	Label
						⋮			
QueryID_i	DocID	Feature_1	Value_1	Feature_2	Value_2	…	Feature_m	Value_m	Label
						⋮			

(b) An example of data format for learning-to-rank in query auto completion.

						⋮			
PrefixID_i	QueryCompletionID	Feature_1	Value_1	Feature_2	Value_2	…	Feature_m	Value_m	Label
PrefixID_i	QueryCompletionID	Feature_1	Value_1	Feature_2	Value_2	…	Feature_m	Value_m	Label
						⋮			
PrefixID_i	QueryCompletionID	Feature_1	Value_1	Feature_2	Value_2	…	Feature_m	Value_m	Label
						⋮			

of available query completions can be extracted offline and stored using an efficient data structure for fast lookups. The features used for QAC, both offline and online, are similar to those used for web search result ranking. For instance, the features related to a signed-in user who typically has an account, can be extracted offline and served for personalized QAC online. Such features, mainly based on their search activities, are stored and associated with their corresponding accounts. For a signed-out user, the personalized features are extracted from his search activity linked to an anonymous cookie in the browser and can be updated when a new search activity is available. It is completely separate from the search engine account and web history, which are only available to signed-in users.

6.2 Error-tolerance in query auto completion

Another aspect related to making QAC practical has to do with resilience to typing errors. Just as there is a basic need for making the lookup opera-

`autocomplete-suggestions-work-62592`

tion tolerant to such typing errors, the QAC systems need to be error-tolerant when responding to an input query prefix. Chaudhuri and Kaushik [2009] take a first step towards addressing this problem by capturing input typing errors using edit distance and propose an approach of invoking an offline edit distance matching algorithm. Their proposal, a trie-based QAC strategy, generates query completions per-character at a minimal cost; the authors report an average per-character response time at the millisecond level and their approach significantly outperforms an n-gram based algorithm.

As misspellings are commonly found in web search and more than 10% of search engines queries are misspelled [Cucerzan and Brill, 2004], Duan and Hsu [2011] study the problem of online spelling correction for query completions, which involves not only completing a typed prefix but also correcting parts of an incorrectly typed prefix when providing query completions as the query is being entered. Specifically, the search engine responds to each keystroke with a list of query completions that best correct and complete the partial query. To deal with the task of online spelling correction for query completion, the authors model search queries with a generative model, where the intended query is transformed into a potentially misspelled query through a noisy channel that describes the distribution of spelling errors. They finally propose to find the top spelling-corrected query completions in real-time by adapting an A* search algorithm with various pruning heuristics to dynamically expand the search space in an efficient manner. Their experimental results demonstrate a substantial increase in effectiveness of online spelling correction over existing techniques.

6.3 Summary

In this chapter, we have discussed two practical issues related to QAC efficiency, i.e., computational efficiency and error-tolerance. Addressing these two issues is required to make a QAC service usable in practice.

In general, in the literature less attention is paid to issues such as efficiency and robustness in QAC than to ranking models for query completions, although QAC services should operate in a real-time and error-rich setting. Most processing steps in a QAC system can be performed offline and the QAC ranking models can be trained regardless of the time consumption.

Based on a trained QAC ranking model, the test phase comes with very limited time costs.

7

Presentation and interaction

In search engines, the most common way that search results are displayed is as a vertical list of items summarizing the retrieved documents [Hearst, 2009]. The search result listings returned by search engines are referred to as *search engine result pages* (SERPs). In query auto completion (QAC), a similar way to present query completions in response to a prefix entered into the search box can be implemented: matching query completions appear below the search box as a drop-down menu with the typed characters highlighted in each query completion; see Figure 1.1.

In addition, users interact with SERPs by clicking, reading and skipping, abandoning, etc., which provides valuable implicit feedback about the relevance of documents to a query as well as a user's search intent. As part of the QAC process, user's interaction behavior with a QAC system, e.g., typing, skipping and clicking, can also be recorded so as to generate a personal search pattern for a particular user, which could improve the prediction accuracy of their intended query. Most research efforts are aimed at improving the accuracy of query completions, e.g., [Bar-Yossef and Kraus, 2011, Shokouhi and Radinsky, 2012, Shokouhi, 2013, Cai et al., 2014b], and generally pay less attention to the presentation issues.

In this chapter, we specifically discuss how to present QAC results and

how people interact with QAC results during the information retrieval process. These two points directly affect QAC usage in practice.

7.1 Presenting query auto completions results

The representation for a retrieved document in a SERP is called a search *hit* [Morris et al., 2008, Kazai et al., 2011] or *document surrogate* [Beitzel et al., 2005, Marchionini and White, 2007]. Example result presentations in document retrieval and in query auto completion are shown in Figure 7.1 and Figure 7.2, respectively. As shown in Figure 7.1, in SERPs, a brief summary of a relevant portion of the document is included to help the user understand the primary object [Marchionini and White, 2007] and is highlighted. The quality of the document surrogate has a strong effect on the ability of the searcher to judge the relevance of the document [Clarke et al., 2007]. Even the most relevant document is unlikely to be selected if the title is uninformative or misleading. In addition, users are known to be biased towards clicking documents higher up in the rankings [Joachims et al., 2005]. Such presentation issues in document retrieval have been studied extensively; see, e.g., [Joho and Jose, 2006, 2008].

In a QAC system, as shown in Figure 7.2, query completions are listed below the search box and only a limited number of query completions are displayed to the user. In addition, query completions that have been issued before are sometimes highlighted with a different color. Other information about the query completions is usually not provided, although query completions for product search are sometimes displayed with an explicit type (such as “in Books” or “in English Literature”). Similarly, bias towards clicking query completions at top positions of the QAC list does exist [Li et al., 2014, Zhang et al., 2015]. In fact, more than three quarters of clicks on query completions for desktop search are located within the top 2 positions of the query completions lists and an even higher percentage is observed on mobile devices [Li et al., 2014]. On the other hand, if the intended query is ranked below the top 2 positions, only 13.4% of them will be clicked by users [Li et al., 2015]. Hence, in a QAC system, issues related to QAC result presentation should be taken into consideration for designing a QAC system as well as improving the user experience.

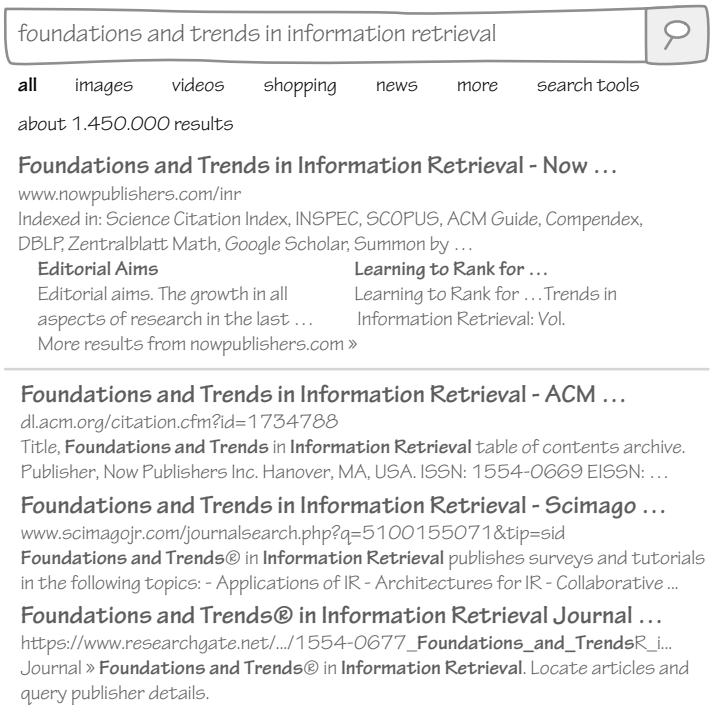


Figure 7.1: SERP for the query "*foundation and trends in information retrieval*."

Amin et al. [2009] focus on the presentation of query completions to see how it influences a user's search performance. They implement user studies that investigate organization strategies, e.g., alphabetical order, group and composite, of query completions in a known-item search task, i.e., searching for terms taken from a thesaurus. They find that for a geographical thesaurus, a sensible grouping strategy, e.g., by country or by place type, that people can understand relatively easily is preferred. Regarding organization strategies of query completions, both group and composite interfaces are found to help the user search for terms faster than an alphabetical ordering, and are generally easier to use and to understand than alphabetical orderings.

Based on the assumption that users would benefit from a presentation in which query completions are not only ordered by likelihood but also organized by a high-level user intent, Jain and Mishne [2010] specifically focus

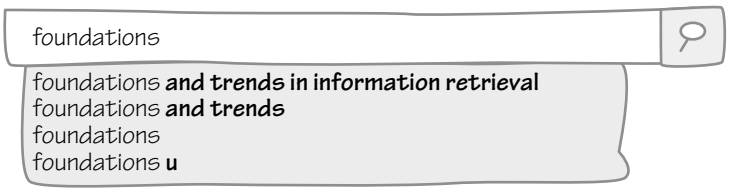


Figure 7.2: Query completions for "*foundations*."

on organizing query completions by topic for web search, which is achieved by clustering and ordering sets of query completions. In particular, they first present a variety of unsupervised techniques to cluster query completions based on the information available to a large-scale web search engine, and then label each cluster using different strategies. Finally, the clusters are ranked by minimizing an expected cost of locating query completions, e.g., time to read a cluster label and time to scan a cluster. Through a pilot user study, a number of interesting and promising observations are obtained. For instance, users prefer query completions displayed in a clustered style over an unclustered presentation. The user effort in identifying the set of relevant completions can be substantially reduced by grouping query completions and labeling them, resulting in increased user satisfaction. However, the tasks related to the clustering, e.g., selection of the queries to cluster, assignment of a label to each cluster, ordering the clusters as well as the query completions within each cluster, are still challenging. In general, this work opens promising directions towards improving the user experience by reducing the user effort in identifying the set of relevant query completions. Jain and Mishne's conclusions are consistent with later findings in the field of query suggestion [Kato et al., 2012] and have since motivated follow-up work on diversifying query auto completion [Cai et al., 2016b].

7.2 Interactions with query auto completion results

In this section, we discuss user interactions with QAC results in more detail. With special devices, a user's explicit interactions during query auto completion engagement can be recorded. Such interactions provide valuable information for capturing a user's personal characteristics, e.g., typing habits and search interests.

Mitra et al. [2014] present the first large-scale study of user interactions with query auto completion. Their investigations reveal that lower-ranked query completions receive substantially less engagement, i.e., fewer clicks, than those ranked higher. In addition, users are most likely to submit a query by clicking a completion after typing about half of the query and in particular at word boundaries. Another factor that affects QAC engagement is related to the distance between query characters on the keyboard. For instance, a monotonic increase in the probability of QAC engagement with keyboard distance up to a distance of four keys can be observed, which demonstrates a clear relation between the position of the next character on the keyboard and the likelihood of QAC usage. Overall, these results appear to confirm typical intuitions of when users would like to use QAC. Furthermore, these results could be due to other attributes like user's typing habit, e.g., skipping and viewing completions. Li et al. [2014] and Zhang et al. [2015] study these matters based on a high-resolution query log dataset.

An in-depth eye-tracking study of user interactions with query auto completion in web search is performed by Hofmann et al. [2014]. In their study, the participants' interactions are recorded using eye-tracking and client-side logging when they complete the assigned web search tasks. A strong position bias towards examining and using top-ranked query completions is identified, which is consistent with similar findings on user interactions with web search results [Guo et al., 2009a, Yue et al., 2010, Chuklin et al., 2015]. Due to this strong position bias, ranking quality does affect QAC usage, as evidenced by different behavioral patterns, which includes activities such as monitoring, skipping and searching for query completions. The work by Hofmann et al. suggests that injecting user feedback into a QAC framework may be beneficial. The issue is partially addressed by Zhang et al. [2015], who take user behavior such as skipping and viewing query completions into account for query auto completion.

The QAC process typically starts when a user enters the first character into the search box and ends when they click a completed query; at the intermediate stages, user interactions with the QAC system may involve examining the list of query completions, continuing to type, and clicking on a query in the list. Complete interaction traces with a QAC service have not been well studied. Li et al. [2014] adopt a click model to shed light on the QAC

user interactions, which consists of a horizontal component that captures the skipping behavior, a vertical part that depicts the vertical examination bias, and a relevance model that reflects the intrinsic relevance to a prefix of a query completion. 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.

Following [Li et al., 2014], Zhang et al. [2015] exploit implicit negative feedback in the QAC process, e.g., dwelling on query completion lists for a long time without selecting query completions ranked at the top. This type of user feedback indicates that the user may not favor unselected query completions even though they are ranked at the top positions in the list. To model this negative feedback, dwell time and the position of unselected query completions are taken into account. By incorporating a (static) relevance score with an implicit negative feedback strength, the proposed model re-ranks the top-k query completions initially returned by popularity. Finally, a case study verifies that the new model outperforms start-of-the-art QAC models for the cases that users submit new queries when reformulating older queries and that users have a clear query intent to rephrase unambiguous queries.

In addition, Li et al. [2015] pay considerable attention to a user's sequential interactions with a QAC system in and across QAC sessions, rather than at each particular keystroke of QAC sessions. Three types of relationship between user's behaviors at different keystrokes are considered, i.e., state transitions between skipping and viewing, user's real preference of query completions and user-specific cost between position clicking and typing. These relationships capture the signals corresponding to whether the user has viewed the query completion. A hidden Markov model is involved to model the transitions and estimate whether the query completion satisfies the user's intent. A logistic regression model is applied to weigh a set of features and predict whether the user is willing to click the query completion; a Dirichlet model is used to estimate the ratio between position-biased clicking and typing costs, respectively. By integrating these three parts, Li et al.'s proposal explains how the three parts together determine a user's choice of clicking a query completion.

7.3 Summary

In this chapter, we have briefly summarized previous work on the presentation of query completions as well as on users' interactions with query completions. A reasonable presentation style of query completions can further improve the user experience. In addition, user interactions with query completion results have not been used extensively for boosting the ranking performance of query completions, which opens new directions for future work in the field of query auto completions.

8

Related tasks

In this chapter, we focus on summarizing work on three tasks that are closely related to query completion, viz., query suggestion in §8.1, query expansion in §8.2, and query correction in §8.3. Together these four tasks are meant to help search engine users to formulate their queries. In addition, we briefly touch on other tasks that we believe can be nicely incorporated into query auto completion for personalization and diversification purposes.

There are subtle but important differences between the task of query auto completion and the related tasks of query suggestion, query expansion and query correction. Below, we briefly summarize related work on these three topics to clarify the difference with query auto completion.

8.1 Query suggestion

Query suggestion, sometimes called query recommendation, has received considerable attention from researchers. Typically, for query suggestion, given a user's input, one recommends a list of *relevant* queries to the original user input, e.g., a real query [Kato et al., 2013] or just a string [Cai et al., 2014b]. In Figure 8.1 we see a number of suggestions of more specific universities in Amsterdam in response to the original query "amsterdam university."

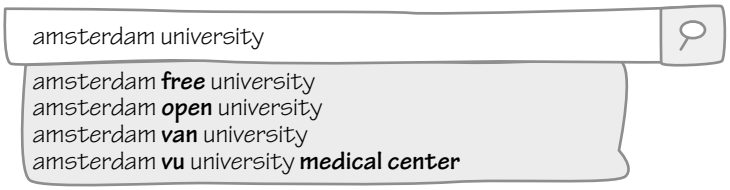


Figure 8.1: An example of query suggestions for the query "amsterdam university."

The main purpose of the query suggestion functionality is to enable the user to formulate a good query and thus submit a related query when they find it difficult to express their information need. We differentiate query auto completion from query suggestion in Table 8.1: for QAC we tend to follow a

Table 8.1: Query auto completion vs. query suggestion

Task	Input	Output
Query auto completion	Prefix p	Query set $Q = \{q_o : q_o \text{ starts with } p\}$
Query suggestion	Query q	Query set $Q = \{q_o : q_o \text{ is relevant to } p\}$

strict matching policy while for query suggestions we do not. Accordingly, in QAC we rank a limited number of query completions and allow the user to complete their query before it has been fully entered without modifying their previously entered input. In contrast, query suggestions are semantically related to the user's input query; the problem of ranking a large number of query suggestions is challenging [Cai et al., 2016a]. In addition, query completions are often shown while the user is entering a query in a special purpose dropdown menu; in contrast, query suggestions are often presented after a query has been submitted and are then displayed on the search engine result page.

Typical baseline approaches to the query suggestion problem include search result-based query suggestion [Yang et al., 2008] and query log-based query suggestions [Baeza-Yates et al., 2004, Anagnostopoulos et al., 2010, Strizhevskaya et al., 2012], where user behavior such as querying and clicking is taken into account. For instance, Cao et al. [2008] propose a context-aware query suggestion approach by mining clickthrough and session data,

where the current query as well as recent queries in the same session are taken into account as context. Before testing, queries are summarized into concepts offline by clustering a click graph, and then query suggestions are provided online based on the concepts. Their model outperforms the baselines in both coverage and quality of suggestions. Ma et al. [2008] aim to provide semantically relevant queries for a user and develop a two-level query suggestion model by mining two bipartite graphs (a user-query and a query-URL bipartite graph) extracted from clickthrough data. Their algorithm is based on a joint matrix factorization method to represent queries and can capture the semantic similarity of queries from a constructed query similarity graph. The results show that both lexically similar and semantically relevant queries can be recommended to users effectively. Mei et al. [2008] propose a query suggestion method based on ranking queries with the so-called hitting time on a large-scale bipartite graph, which is the expected time for a random walk on a finite graph to reach a specific vertex in a set of vertexes. The authors show that this notion is able to capture semantic relations between the suggested query and the original query. In addition, the proposed algorithm can successfully boost long tail queries. Following the query suggestion approach using the hitting time, Ma et al. [2010] focus on suggesting both semantically relevant and diverse queries to Web users based on Markov random walks and hitting time analysis on the query-URL bipartite graph. Their model effectively prevents semantically redundant queries from receiving a high rank, hence encouraging diversity in the results. For diversifying query suggestions, Zhu et al. [2011] leverage a ranking process over a query manifold, which can make full use of relationships among queries encoded in the manifold structure to find relevant and salient queries. Kim and Croft [2014] also focus on diversifying query suggestions based on query documents to help domain-specific searchers. Diverse query aspects of an original query are first identified by representing a query aspect as a set of related terms from the query document, and then these query aspects are used to generate query suggestions that are both effective for retrieving relevant documents and related to more query aspects.

A broad variety of sources has been used for query suggestions: information from social media data [Guo et al., 2010], returned results of queries [Song et al., 2011b], user behavior [Zhu et al., 2012] as well as user

context [Feild and Allan, 2013, Shokouhi et al., 2015] for personalization in general IR [Cai et al., 2014a, 2016c]. Liu et al. [2012] argue that query suggestion is much more useful for difficult queries than easy ones, and hence, they propose a learning-to-rank based query suggestion approach for difficult queries according to the estimated query difficulty. In the absence of query logs, Bhatia et al. [2011] propose a probabilistic approach for generating query suggestions from a corpus; the latter is utilized to extract a set of candidate phrases that are highly correlated with the user query. Their proposed approach is tested on a variety of datasets and the experimental results clearly demonstrate the effectiveness of their proposal in suggesting queries with higher quality. Moreover, Ma et al. [2012] introduce new assessment criteria for query suggestion to address the case where the original query fails to return satisfactory search results.

However, in the work on query suggestion listed above, the recommended query candidates do not need to match the user input strictly but they should be highly relevant, which is different form the task of query auto completion where the completion candidates must start with the user input [Cai et al., 2014b] and need not be semantically related to the original user input.

8.2 Query expansion

Query expansion is a widely used successful technique to extend the original query with other terms that best capture the correct user intent, or that simply generate a better query to retrieve relevant documents [Carpineto and Romano, 2012]. Query expansion is often hidden in the search systems, i.e., not directly available for users to interact with, and works mainly to resolve the "*vocabulary problem*" arising from the use of short queries, such as synonymy and polysemy [Furnas et al., 1987].

User feedback is an important source for suggesting a set of related queries for a user-issued keyword query, and is typically mined based on the analysis of search logs. For instance, Buscher et al. [2008] propose to expand queries using gaze-based feedback from the user on the subdocument level by employing eye tracking data as implicit feedback for personalizing the search process. Cui et al. [2002] propose an approach that extracts correlations between query terms and document terms by analyzing query logs. These ex-

tracted correlations are then used to select high-quality expansion terms for new queries. The feedback information can also be collected from returned results of queries. For instance, Liu et al. [2011] propose a new framework for keyword query expansion by clustering the results according to user specified granularity, so that for each cluster one expanded query is generated. Similarly, Riezler et al. [2008] propose to translate queries into snippets for improved query expansion, where pairs of user queries and snippets of clicked results are applied to train a machine translation model that helps achieve improved contextual query expansion when compared to an approach based on term correlations. Broder et al. [2009] mainly focus on online query expansion of rare queries in sponsored search systems that are tasked with matching queries to relevant advertisements; the returned results are used to enrich the representation of the original tail query.

Web queries are normally short, e.g., around 2 words on average in the AOL dataset [Pass et al., 2006]. Thus, it is often assumed that user's information needs are not fully specified. Chirita et al. [2007] aim to improve the retrieval performance of such short queries by expanding them with terms collected from each user's personal information repository in order to automatically extract additional keywords related both to the query itself and to user's interests. Fonseca et al. [2005] propose a concept-based query expansion technique that allows one to disambiguate queries submitted to search engines; the concepts are extracted by analyzing a query-relation graph, and concepts related to a given query are used to expand the original query for improving the search quality.

Other signals corresponding to active feedback [He and Ounis, 2009] or linguistic features [Bai et al., 2007] have also been examined for query expansion. Generally speaking, query expansion has become a widely used strategy to improve the retrieval effectiveness of document ranking [Carpineto and Romano, 2012], which is usually evaluated by taking into account both recall and precision. Recent and not so recent experimental studies show that query expansion results in better retrieval effectiveness [Liu et al., 2004, Lee et al., 2008]. It has become one of the standard workhorses of information retrieval, as witnessed by its inclusion in commercial applications like Google

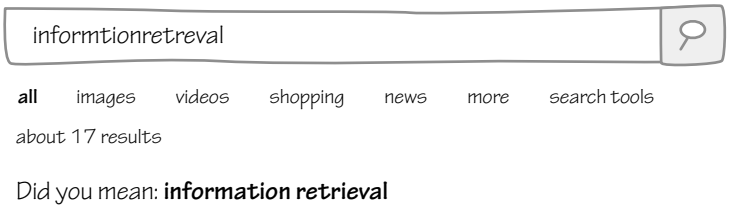


Figure 8.2: An example of a query correction for "informtionretreval".

Enterprise[1] and MySQL.[2]

8.3 Query correction

Query correction, or query spelling correction, is another topic that is closely related to query auto completion. This is functionality that is meant to automatically correct potentially misspelled queries. Figure 8.2 shows an example of a search engine providing a corrected query corresponding to a user's original input which may be spelled incorrectly. The incorrectly spelled query is "informtionretreval"; the search engine's response is "Did you mean: information retrieval" but the results that it returns are matches for the original incorrectly spelled query rather than for "information retrieval."

Automatic query spelling correction may help the search engine's understanding of the user's correct search intents; it can therefore improve retrieval effectiveness [Ruch, 2002] as well as user's satisfaction of search experience [Li et al., 2012]. So far, a remarkable number of methods have been proposed to query spelling correction, mainly collecting correct queries from pre-trained word or query pairs based on large text collections [Gao et al., 2010, Suzuki and Gao, 2012] or query logs [Gao et al., 2010, Sun et al., 2010]. For instance, Li et al. [2012] propose a generalized query spelling correction approach based on a Hidden Markov Model with discriminative training, which cannot only handle spelling errors, e.g., splitting and concatenation errors, but can also evaluate candidate corrections so as to guarantee an optimal correction. Similarly, Duan et al. [2012] introduce a discriminative model for query spelling correction that directly optimizes the search stage

[1] `https://enterprise.google.com/`
[2] `https://www.mysql.com`

with a discriminative model based on the latent structural SVM. In particular, query spelling correction is treated as a multi-class classification problem with structured input and output, where the latent structure is used to model the alignment of words in the spelling correction process. Sun et al. [2012] explore the use of an online multi-task learning framework for query spelling correction, where correction candidates are initially generated by a ranking-based system and then re-ranked by a multi-task learning algorithm. Their method works well on highly biased training datasets. Suzuki and Gao [2012] propose a unified approach to the problem of spelling correction using an approach inspired by a phrase-based statistical machine translation framework; it works at two levels, i.e., at the phrase level and at the sentence level, corresponding to a monotone character conversion decoder and a transliteration decoder, respectively.

Gao et al. [2010] develop a system for query spelling correction that uses massive Web corpora and search logs for improving a ranker-based search query speller. In addition, clickthrough data captured in query logs has been explored for query spelling correction. Sun et al. [2010] derive large numbers of query-correction pairs by analyzing users' query reformulation behavior from clickthrough data. In addition, Chen et al. [2007] use web search results to improve existing query spelling correction models solely based on query logs by leveraging the rich information on the web related to the query and its top-ranked candidates. Bhole and Udupa [2015] focus on correcting misspelled queries for email search using a user's own email data and Udupa and Kumar [2010] concentrate on spelling correction of personal names.

Query auto completion shares some features with the closely related tasks discussed in this chapter, i.e., query suggestion, query expansion and query correction:

1. For the majority of these approaches, dealing with query auto completion, query suggestion and query expansion, etc, relies on the collected query logs, where the search intent of a user and the query topics are typically identified based on clickthrough data;
2. The main purpose of these types of functionality is the same, i.e., to help users formulate the correct query. Hence, the evaluation metrics, basically measuring how the algorithms can provide the ranked list of suitable query candidates, are often the same;

3. The main overall challenges of these three types of functionality are sometimes similar, e.g., how to exploit personalized information to boost the performance and how to identify the right kind of semantic information for these tasks? In practice, they all suffer from the problem of sparseness when inferring query topics based on click data.

In contrast, the main differences between query auto completion and the three other types of functionality are related to their input and output. For instance, a query expansion approach should add new terms to the original query while query auto completion method should complete the user's input prefix by providing real queries. However, query suggestion and query correction approaches may replace some or all of the user's original input query with alternative terms.

Regarding the algorithms used, the most important differences between query auto completion, query suggestion, query expansion, and query correction are listed below:

1. The popularity-based Maximum Likelihood Estimation (MLE) algorithm has been shown to perform well in query auto completion; however, it is not commonly used in the fields of query suggestion, query expansion, and query correction;
2. Synonym expansion algorithms based on an indexed vocabulary of term pairs can be applied to the field of query expansion, but it is of limited use in other areas such as query auto completion and query suggestion;
3. Misspelling detection algorithms based on term-level matching can be borrowed to deal with query correction, however, it they do not constitute obvious solutions for query auto completion or query expansion.

So far, we have briefly summarized work on query suggestion, query expansion as well as query correction. A concise comparison is made to distinguish these tasks from query auto completion. In practice, other types of query reformulation are also commonly adopted by search engine users, e.g., query generalization and specialization, which may be derived from a semantic categorization of query reformulation in previous work [Rieh and Xie, 2006]. According to [Kato et al., 2012, 2013], query specializations are often used when the original query is a single-term query or the search intent is ambiguous; query generalizations are often adopted when the original query is a rare

query. In general, query specializations and generalizations serve a similar purpose as query completions, i,e., they aim to formulate useful queries that users can submit by simply clicking. In essence, query specializations and generalizations are alternatives to a user's original input query while query completions actually are extensions of user's original input string. In addition, the key criterion for assessing the quality of query specializations and generalizations, i.e., whether the suggested queries really satisfy the user's information need, coincides with that of query completions and can be determined by tracking a user's interactions with the search system.

Finally, other tasks, like subtopic mining (i.e., given a query, generate a ranked list of possible subtopics [Song et al., 2011a, Sakai et al., 2013]) and subtask mining (i.e., given a task, automatically find subtasks [Liu et al., 2014, Yamamoto et al., 2016]) can also be related to query auto completion. For instance, for the task of subtopic mining, a list of query completions can be provided as possible subtopic candidates [Liu et al., 2014]. Insights from tasks such as subtopic mining and subtask mining can be borrowed to improve the performance of query auto completion for a particular task, such as diversifying query auto completion [Cai et al., 2016b].

9

Conclusions

In this chapter, we briefly summarize the main topics presented in this survey in §9.1 and point out potential future research directions in §9.2.

9.1 Summary of the survey

In this survey, we have described the major advances in the field of query auto completion in information retrieval. In particular, in Chapter 1, we have provided a general description of the topic. In Chapter 2, a formal definition and a general framework of query auto completion in IR are presented. In Chapters 3 and 4, we have introduced the most prominent QAC approaches in the literature, i.e., heuristic approaches and learning-based approaches, both of which exploit time-sensitive and user-centered characteristics. In Chapter 5, the publicly available query logs that are often used for evaluation purposes are described. In addition, the metrics used for quantitatively evaluating precision-oriented QAC rankings as well as diversity-oriented QAC rankings are discussed. Some practical issues, like efficiency and robustness in QAC and presentation of query completions, are discussed in Chapters 6 and 7, respectively. After that, in Chapter 8, we have summarized recent work on closely related research topics to query auto completion, i.e., query sugges-

tion, query expansion and query correction.

9.2 Open research directions

Query auto completion has received a considerable amount of attention from the information retrieval community in recent years, resulting in substantial achievements by incorporating knowledge from available sources, e.g., time and user interactions. Despite the progress, several research challenges related to QAC remain. In this section, we sketch eight such directions: mining entities for QAC in §9.2.1, diversifying QAC in §9.2.2, mobile QAC in §9.2.3, QAC for voice search in §9.2.4, online learning for QA in §9.2.5, click models for QAC in §9.2.6, QAC evaluation in §9.2.7, and finally, research opportunities related to multi-objective optimization in §9.2.8.

9.2.1 Entities help query auto completion

One possible direction of development is to consider the entities in a query. It has been reported that a remarkable proportion of queries submitted to web search engines involve entities [Guo et al., 2009b, 2011, Lin et al., 2012], varying from 43% up to 70%, which calls for an automatic approach to identifying entity-related queries. One of the key challenges in considering entity information for query auto completion is the correct detection, recognition and disambiguation of entities from query prefixes. This involves the segmentation of queries, the classification of entities as well as matching entities against a knowledge graph [Meij et al., 2011]. A semantic approach to suggesting query completions that considers entity information has been presented in [Meij et al., 2009]. It has been shown that, compared to a frequency-based approach, such entity information mostly helps rare queries. From the algorithmic side, this could yield methods for promoting entity-related queries that are close to a user's search context. Similar methods have been developed in the web search community [Reinanda et al., 2015], but need to be adapted to be applicable to query auto completion.

More importantly, compared to the percentage of entity queries for web search, a substantially larger proportion of entity queries are submitted in academic search, accounting for more than 93% of all queries [Li et al., 2016]. Such observations suggest that entity characteristics in a query can bene-

fit today's query auto completion approaches, especially in the area of academic search and potentially also in other entity-rich search scenarios such as health-related search.

9.2.2 Diversifying query auto completion

The task of diversifying query auto completion is relatively new. It has been studied in [Cai et al., 2016b], where a greedy query selection approach based on the Open Directory Project (ODP) is proposed to remove semantically close query candidates from the originally returned QAC list and to include query completions covering more aspects. This solution achieves good performance in terms of MRR for query ranking and in terms of α-nDCG score for diversification. But it is far from optimal. We believe that the task of web search result diversification [Dang and Croft, 2012, Bache et al., 2013, Dang and Croft, 2013] provides inspiration for new scenarios to be considered for diversifying query auto completion. For instance, following Vallet and Castells [2012], one could combine personalization and diversification and develop a generalized framework for personalized query auto completion diversification, enhancing each of them by introducing the user as an explicit random factor in state-of-the-art query auto completion methods. In addition, like Vargas et al. [2012], one could introduce an explicit relevance model to improve intent-oriented diversification for query auto completion.

9.2.3 Query auto completion for mobile search

The usage of search engines on mobile devices has experienced a rapid growth in recent years [Church and Oliver, 2011]. At the time of writing, mobile usage of search has surpassed desktop usage. However, text input on small screen devices is relatively slow and generally clumsier than on traditional desktop machines. Thus, assisting users to formulate their queries will contribute to a more satisfactory search experience. QAC for mobile search has not been well studied so far. One possible solution to QAC for mobile search is to fully exploit the spatio-temporal information of query candidates, such as the query time, the moving direction of user and the distance from this particular user to a candidate query completion. This type of spatio-temporal information could be incorporated into QAC models for mobile search. For

instance, in many countries users will favor submitting a query about a restaurant at lunch time than at breakfast time; and users probably are not inclined to select a query candidate (a location) that is either very far away or extremely close to their present location.

9.2.4 Query auto completion for voice search

User interactions with mobile devices increasingly depend on voice as a primary input modality [Lei et al., 2013]. Due to the disadvantages of sending audio across potentially costly and scarce network connections for speech recognition, there has been growing attention to performing recognition on mobile devices, where the recognizer must transcribe an utterance if it is in a target set or hand it off for off-device handling [Van Gysel et al., 2015]. How should query auto completion for voice search on a mobile device work? Can query completions for prefixes in the target set for the voice recognizer be dealt with on-device in an efficient and effective manner?

9.2.5 Online learning for query auto completion

Online learning-to-rank for information retrieval is an active area [Hofmann et al., 2013a,b, Lefortier et al., 2014, Schuth et al., 2016, Oosterhuis et al., 2016]. Here, system performance is optimized directly from implicit feedback inferred from user interactions, e.g., clicks. How can we reliably infer user preferences from interactions with query completions [Hofmann et al., 2011]? Limited by the availability of rich interactions from users with the QAC system, the task of online learning to rank for query auto completion has so far received limited attention.

An alternative is to explore the interactions from users with the search results. In an online setting, an important challenge that learning for query auto completion approaches have to address is to learn as quickly as possible from the limited quality and quantity of feedback that can be inferred from user behaviors. Consequently, on the assumption that suitable user feedback for learning can be observed, approaches to online learning to rank for query auto completion should take into account both the quality of the current list of query completions, and the potential to improve the quality in the future based on inferred user preference.

9.2.6 Click models for query auto completion

In the area of web search, the main purpose of modeling user clicks [Chuklin et al., 2015] is to infer the intrinsic relevance for a query of the results returned in response to it [Dupret and Piwowarski, 2008, Liu et al., 2010a, Zhu et al., 2010]. A core assumption here is that a click is a positive signal expressing attractiveness of a result to a user. In click models, position bias is a factor that is explicitly taken into account when estimating relevance [Granka et al., 2004, Chapelle and Zhang, 2009]. Li et al. [2014] introduce a two-dimensional click model for query auto completion, which is the first click model for modeling the QAC process and opens a new research direction.

Factors known from the literature on click models for web search can be introduced to predict the likelihood of a user clicking a query completion given a prefix they entered. Given the additional contextual factors that play a role in the setting of query auto completion, explicitly designing graphical models that capture all factors may not be the most interesting way forward from an algorithmic point of view; we believe that approaches that automatically infer the statistical structure of interactions with query completions, in the spirit of [Borisov et al., 2016], are a more promising road ahead.

9.2.7 Query auto completion performance and evaluation

Very little work has been done on performance prediction for query auto completion [He et al., 2008]. E.g., in a learning-based approach to the task, can we reliably estimate the quality of the completions before displaying them?

So far, the performance of various QAC approaches has mostly been evaluated by using metrics such as Mean Reciprocal Rank (MRR) and Success Rate [Jiang et al., 2014, Whiting and Jose, 2014, Cai et al., 2016a, Cai and de Rijke, 2016a,b], which have been introduced in §5.2. Such metrics measure the quality of rankings of query completions. If a query completion is finally submitted by the user, the higher its rank in the ranked list of query completions, the better the corresponding approach that produced the list is. However, users care much more about the search results than the query completions. Hence, in addition to evaluating rankings of QAC, it would be interesting to systematically evaluate the performance of QAC approaches by considering the search results after making use of completions. For instance, we

can examine user satisfaction by analyzing interactions on different SERPs resulting from different completions.

9.2.8 Multi-objective query auto completion

So far, we have only considered a single objective for the generation and evaluation of QAC results: satisfaction of the user entering the prefix to be completed. We may want to consider additional objectives. For instance, in a commercial setting, especially when completing queries, and hence recommending products, from a large pool of candidate items, commercial objectives could be injected into a basic QAC system. In e-commerce settings, as users are often overwhelmed by the sheer volume of products available, an effective query should directly lead to a purchase decision [Dias et al., 2008, Fleder and Hosanagar, 2007]. Hence, the benefit of an electronic retailer is a salient objective of queries and of query completions. While there are recent attempts at multi-objective optimization in the context of information retrieval [van Doorn et al., 2016], we are not aware of studies that specifically target multiple objectives for QAC.

9.3 Concluding remarks

In this chapter, we have briefly summarized the materials discussed throughout this survey. In addition, we have provided a landscape of open research directions related to QAC, concerning entities, diversification, mobile search, voice search, online learning, click models, evaluation and multi-objective query auto completion. Information access is increasingly a *mixed initiative scenario*, where human and artificial agents take turns. Query auto completion is a very specific area where initiatives taken by artificial agents prove to be particularly effective and broadly accepted. We believe that this is only the beginning. The quantity of the work described in this survey and the steady pace of publications in the field together with the open research follow-ups identified in this chapter are an indication of a promising future ahead.

Glossary

α-nDCG – α-Normalized discounted cumulative gain, p. 39
AOL – America online dataset, p. 36
AP – Average precision, p. 27
CLSM – Convolutional latent semantic model, p. 31
DR – Document retrieval, p. 27
ERR-IA – Intent-aware expected reciprocal rank, p. 39
LLR – Likelihood ratio, p. 28
MAP-IA – Intent-aware mean average precision, p. 39
MLE – Maximum likelihood estimation, p. 14
MPC – Most popular completion, p. 14
MRR – Mean reciprocal rank, p. 38
MSN – MSN search dataset, p. 36
nDCG – Normalized discounted cumulative gain, p. 39
NRBP – Novelty- and rank-biased precision, p. 39
ODP – Open directory project, p. 68
QAC – Query auto completion, p. 2
RR – Reciprocal rank, p. 38
RMQ – Range minimum query, p. 44
SERP – Search engine result page, p. 50
SVM – Support vector machine, p. 63
TREC – Text retrieval conference, p. 39

Acknowledgements

We would like to thank our colleagues Xinyi Li, Daan Odijk, Milad Shokouhi, and our anonymous reviewers for valuable feedback and suggestions. We are also grateful to the editors of the journal, Doug Oard and Mark Sanderson, for their advice and support throughout the writing process.

The writing of this survey was supported by the Innovation Foundation of NUDT for Postgraduate under No. B130503, the China Scholarship Council (CSC), Ahold Delhaize, Amsterdam Data Science, Blendle, the Bloomberg Research Grant program, the Dutch national program COMMIT, Elsevier, the European Community's Seventh Framework Program (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 Organization 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, 314-98-071, 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

Rakesh Agrawal, Sreenivas Gollapudi, Alan Halverson, and Samuel Ieong. Diversifying search results. In *Proceedings of the 2nd ACM International Conference on Web Search and Data Mining*, WSDM '09, pages 5–14, New York, NY, USA, 2009. ACM.

Alia Amin, Michiel Hildebrand, Jacco van Ossenbruggen, Vanessa Evers, and Lynda Hardman. Organizing suggestions in autocompletion interfaces. In *Proceedings of the 31st European Conference on Information Retrieval*, ECIR '09, pages 521–529, Berlin, Heidelberg, 2009. Springer-Verlag.

Aris Anagnostopoulos, Luca Becchetti, Carlos Castillo, and Aristides Gionis. An optimization framework for query recommendation. In *Proceedings of the Third ACM International Conference on Web Search and Data Mining*, WSDM '10, pages 161–170, New York, NY, USA, 2010. ACM.

Kevin Bache, David Newman, and Padhraic Smyth. Text-based measures of document diversity. In *Proceedings of the 19th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '13, pages 23–31, New York, NY, USA, 2013. ACM.

Ricardo Baeza-Yates, Carlos Hurtado, and Marcelo Mendoza. Query recommendation using query logs in search engines. In *Proceedings of the 2004 International Conference on Current Trends in Database Technology*, EDBT '04, pages 588–596, Berlin, Heidelberg, 2004. Springer-Verlag.

Jing Bai, Jian-Yun Nie, Guihong Cao, and Hugues Bouchard. Using query contexts in information retrieval. In *Proceedings of the 30th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '07, pages 15–22, New York, NY, USA, 2007. ACM.

Ziv Bar-Yossef and Naama Kraus. Context-sensitive query auto-completion. In *Proceedings of the 20th International World Wide Web Conference*, WWW '11, pages 107–116, New York, NY, USA, 2011. ACM.

Holger Bast and Ingmar Weber. Type less, find more: Fast autocompletion search with a succinct index. In *Proceedings of the 29th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '06, pages 364–371, New York, NY, USA, 2006. ACM.

Holger Bast, Alexandru Chitea, Fabian Suchanek, and Ingmar Weber. Ester: Efficient search on text, entities, and relations. In *Proceedings of the 30th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '07, pages 671–678, New York, NY, USA, 2007. ACM.

Holger Bast, ChristianW. Mortensen, and Ingmar Weber. Output-sensitive autocompletion search. *Information Retrieval*, 11(4):269–286, 2008.

Steven M. Beitzel, Eric C. Jensen, Ophir Frieder, Abdur Chowdhury, and Greg Pass. Surrogate scoring for improved metasearch precision. In *Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '05, pages 583–584, New York, NY, USA, 2005. ACM.

Paul N. Bennett, Ryen W. White, Wei Chu, Susan T. Dumais, Peter Bailey, Fedor Borisyuk, and Xiaoyuan Cui. Modeling the impact of short- and long-term behavior on search personalization. In *Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '12, pages 185–194, New York, NY, USA, 2012. ACM.

Sumit Bhatia, Debapriyo Majumdar, and Prasenjit Mitra. Query suggestions in the absence of query logs. In *Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '11, pages 795–804, New York, NY, USA, 2011. ACM.

Abhijit Bhole and Raghavendra Udupa. On correcting misspelled queries in email search. In *Proceedings of the 29th AAAI Conference on Artificial Intelligence*, AAAI '15, pages 4266–4267, New York, NY, USA, 2015. AAAI Press.

Steffen Bickel, Peter Haider, and Tobias Scheffer. Learning to complete sentences. In *Proceedings of the 16th European Conference on Machine Learning*, ECML '05, pages 497–504, Berlin, Heidelberg, 2005. Springer Berlin Heidelberg.

Alexey Borisov, Ilya Markov, Maarten de Rijke, and Pavel Serdyukov. A neural click model for web search. In *Proceedings of the 25th International World Wide Web Conference*, WWW '16, pages 531–541, New York, NY, USA, 2016. ACM.

Andrei Broder, Peter Ciccolo, Evgeniy Gabrilovich, Vanja Josifovski, Donald Metzler, Lance Riedel, and Jeffrey Yuan. Online expansion of rare queries for sponsored search. In *Proceedings of the 18th International World Wide Web Conference*, WWW '09, pages 511–520, New York, NY, USA, 2009. ACM.

Christopher J.C. Burges, Krysta M. Svore, Paul N. Bennett, Andrzej Pastusiak, and Qiang Wu. Learning to rank using an ensemble of lambda-gradient models. *Journal of Machine Learning Research*, 14:25–35, 2011.

Georg Buscher, Andreas Dengel, and Ludger van Elst. Query expansion using gaze-based feedback on the subdocument level. In *Proceedings of the 31st International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '08, pages 387–394, New York, NY, USA, 2008. ACM.

Fei Cai and Maarten de Rijke. Selectively personalizing query auto completion. In *Proceedings of the 39th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '16, pages 993–996, New York, NY, USA, 2016a. ACM.

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

Fei Cai, Shangsong Liang, and Maarten de Rijke. Personalized document re-ranking based on bayesian probabilistic matrix factorization. In *Proceedings of the 37th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '14, pages 835–838, New York, NY, USA, 2014a. ACM.

Fei Cai, Shangsong Liang, and Maarten de Rijke. Time-sensitive personalized query auto-completion. In *Proceedings of the 23rd ACM Conference on Information and Knowledge Management*, CIKM '14, pages 1599–1608, New York, NY, USA, 2014b. ACM.

Fei Cai, Shangsong Liang, and Maarten de Rijke. Prefix-adaptive and time-sensitive personalized query auto completion. *IEEE Transactions on Knowledge and Data Engineering*, 28(9):2452–2466, September 2016a.

Fei Cai, Ridho Reinanda, and Maarten de Rijke. Diversifying query auto-completion. *ACM Transactions on Information Systems*, 34(4):Article 25, 2016b.

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

Huanhuan Cao, Daxin Jiang, Jian Pei, Qi He, Zhen Liao, Enhong Chen, and Hang Li. Context-aware query suggestion by mining click-through and session data. In *Proceedings of the 14th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '08, pages 875–883, New York, NY, USA, 2008. ACM.

Claudio Carpineto and Giovanni Romano. A survey of automatic query expansion in information retrieval. *ACM Computing Surveys*, 44(1):1–50, 2012.

Olivier Chapelle and Ya Zhang. A dynamic bayesian network click model for web search ranking. In *Proceedings of the 18th International World Wide Web Conference*, WWW '09, pages 1–10, New York, NY, USA, 2009. ACM.

Olivier Chapelle, Donald Metlzer, Ya Zhang, and Pierre Grinspan. Expected reciprocal rank for graded relevance. In *Proceedings of the 18th ACM Conference on Information and Knowledge Management*, CIKM '09, pages 621–630, New York, NY, USA, 2009. ACM.

Surajit Chaudhuri and Raghav Kaushik. Extending autocompletion to tolerate errors. In *Proceedings of the 2009 ACM SIGMOD International Conference on Management of Data*, SIGMOD '09, pages 707–718, New York, NY, USA, 2009. ACM.

Qing Chen, Mu Li, and Ming Zhou. Improving query spelling correction using web search results. In *Proceedings of the 2007 Conference on Empirical Methods in Natural Language Processing*, EMNLP '07, pages 181–189, Stroudsburg, PA, USA, 2007. Association for Computational Linguistics.

Steve Chien and Nicole Immorlica. Semantic similarity between search engine queries using temporal correlation. In *Proceedings of the 14th International World Wide Web Conference*, WWW '05, pages 2–11, New York, NY, USA, 2005. ACM.

Paul Alexandru Chirita, Claudiu S. Firan, and Wolfgang Nejdl. Personalized query expansion for the web. In *Proceedings of the 30th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '07, pages 7–14, New York, NY, USA, 2007. ACM.

Aleksandr Chuklin, Ilya Markov, and Maarten de Rijke. *Click Models for Web Search*. Synthesis Lectures on Information Concepts, Retrieval, and Services. Morgan & Claypool Publishers, San Rafael, CA, USA, August 2015.

Karen Church and Nuria Oliver. Understanding mobile web and mobile search use in today's dynamic mobile landscape. In *Proceedings of the 13th International Conference on Human Computer Interaction with Mobile Devices and Services*, MobileHCI '11, pages 67–76, New York, NY, USA, 2011. ACM.

Eugene Ciccarelli. An introduction to the Emacs editor. Technical Report AIM-447, MIT Artificial Intelligence Laboratory, Cambridge, Massachusetts, 1978.

Charles L. A. Clarke, Eugene Agichtein, Susan Dumais, and Ryen W. White. The influence of caption features on clickthrough patterns in web search. In *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '07, pages 135–142, New York, NY, USA, 2007. ACM.

Charles L.A. Clarke, Maheedhar Kolla, Gordon V. Cormack, Olga Vechtomova, Azin Ashkan, Stefan Büttcher, and Ian MacKinnon. Novelty and diversity in information retrieval evaluation. In *Proceedings of the 31st International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '08, pages 659–666, New York, NY, USA, 2008. ACM.

Charles L.A. Clarke, Maheedhar Kolla, and Olga Vechtomova. An effectiveness measure for ambiguous and underspecified queries. In *Proceedings of the 2009 International Conference on the Theory of Information Retrieval*, ICTIR '09, pages 188–199, Berlin, Heidelberg, 2009. Springer-Verlag.

Kevyn Collins-Thompson, Paul N. Bennett, Ryen W. White, Sebastian de la Chica, and David Sontag. Personalizing web search results by reading level. In *Proceedings of the 20th ACM Conference on Information and Knowledge Management*, CIKM '11, pages 403–412, New York, NY, USA, 2011. ACM.

Kevyn Collins-Thompson, Paul Bennett, Charles L.A. Clarke, and Ellen M. Voorhees. TREC 2013 web track overview. In *Proceedings of the 21st Text REtrieval Conference*, TREC '13, pages 1–15, Berlin, Heidelberg, 2013. Springer-Verlag.

Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms*. The MIT Press, 3rd edition, 2009.

Nick Craswell, Onno Zoeter, Michael Taylor, and Bill Ramsey. An experimental comparison of click position-bias models. In *Proceedings of the 1st ACM International Conference on Web Search and Data Mining*, WSDM '08, pages 87–94, New York, NY, USA, 2008. ACM.

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

Silviu Cucerzan and Eric Brill. Spelling correction as an iterative process that exploits the collective knowledge of web users. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, EMNLP '04, pages 293–300, Stroudsburg, PA, USA, 2004. Association for Computational Linguistics.

Hang Cui, Ji-Rong Wen, Jian-Yun Nie, and Wei-Ying Ma. Probabilistic query expansion using query logs. In *Proceedings of the 11th International World Wide Web Conference*, WWW '02, pages 325–332, New York, NY, USA, 2002. ACM.

Van Dang and Bruce W. Croft. Term level search result diversification. In *Proceedings of the 36th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '13, pages 603–612, New York, NY, USA, 2013. ACM.

Van Dang and W. Bruce Croft. Diversity by proportionality: An election-based approach to search result diversification. In *Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '12, pages 65–74, New York, NY, USA, 2012. ACM.

John J. Darragh. Adaptive predictive text generation and the reavtive keyboard. Master's thesis, Computer Science Department, University of Calgary, 1988.

John J. Darragh, Ian H. Witten, and Mark L. James. The reactive keyboard: A predictive typing aid. *Computer*, 23(11):41–49, 1990.

Giovanni Di Santo, Richard McCreadie, Craig Macdonald, and Iadh Ounis. Comparing approaches for query autocompletion. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '15, pages 775–778, New York, NY, USA, 2015. ACM.

M. Benjamin Dias, Dominique Locher, Ming Li, Wael El-Deredy, and Paulo J.G. Lisboa. The value of personalised recommender systems to e-business: A case study. In *Proceedings of the 2008 ACM Conference on Recommender Systems*, RecSys '08, pages 291–294, New York, NY, USA, 2008. ACM.

Zhicheng Dou, Ruihua Song, and Ji-Rong Wen. A large-scale evaluation and analysis of personalized search strategies. In *Proceedings of the 16th International World Wide Web Conference*, WWW '07, pages 581–590, New York, NY, USA, 2007. ACM.

Huizhong Duan and Bo-June (Paul) Hsu. Online spelling correction for query completion. In *Proceedings of the 20th International World Wide Web Conference*, WWW '11, pages 117–126, New York, NY, USA, 2011. ACM.

Huizhong Duan, Yanen Li, ChengXiang Zhai, and Dan Roth. A discriminative model for query spelling correction with latent structural SVM. In *Proceedings of the 2012 Conference on Empirical Methods in Natural Language Processing*, EMNLP '12, pages 1511–1521, Stroudsburg, PA, USA, 2012. Association for Computational Linguistics.

Georges E. Dupret and Benjamin Piwowarski. A user browsing model to predict search engine click data from past observations. In *Proceedings of the 31st International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '08, pages 331–338, New York, NY, USA, 2008. ACM.

Henry Feild and James Allan. Task-aware query recommendation. In *Proceedings of the 36th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '13, pages 83–92, New York, NY, USA, 2013. ACM.

Daniel M. Fleder and Kartik Hosanagar. Recommender systems and their impact on sales diversity. In *Proceedings of the 8th ACM Conference on Electronic Commerce*, EC '07, pages 192–199, New York, NY, USA, 2007. ACM.

Bruno M. Fonseca, Paulo Golgher, Bruno Pôssas, Berthier Ribeiro-Neto, and Nivio Ziviani. Concept-based interactive query expansion. In *Proceedings of the 14th ACM Conference on Information and Knowledge Management*, CIKM '05, pages 696–703, New York, NY, USA, 2005. ACM.

Guillem Francès, Xiao Bai, B. Barla Cambazoglu, and Ricardo Baeza-Yates. Improving the efficiency of multi-site web search engines. In *Proceedings of the 7th ACM International Conference on Web Search and Data Mining*, WSDM '14, pages 3–12, New York, NY, USA, 2014. ACM.

George W. Furnas, Thomas K. Landauer, Louis M. Gomez, and Susan T. Dumais. The vocabulary problem in human-system communication. *Communications of the ACM*, 30(11):964–971, 1987.

Jianfeng Gao, Xiaolong Li, Daniel Micol, Chris Quirk, and Xu Sun. A large scale ranker-based system for search query spelling correction. In *Proceedings of the 23rd International Conference on Computational Linguistics*, COLING '10, pages 358–366, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics.

Nadav Golbandi, Liran Katzir, Yehuda Koren, and Ronny Lempel. Expediting search trend detection via prediction of query counts. In *Proceedings of the 6th ACM International Conference on Web Search and Data Mining*, WSDM '13, pages 295–304, New York, NY, USA, 2013. ACM.

Paul Goodwin. The Holt-Winters approach to exponential smoothing: 50 years old and going strong. *Foresight: The International Journal of Applied Forecasting*, 1 (19):30–33, 2010.

Korinna Grabski and Tobias Scheffer. Sentence completion. In *Proceedings of the 27th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '04, pages 433–439, New York, NY, USA, 2004. ACM.

Laura A. Granka, Thorsten Joachims, and Geri Gay. Eye-tracking analysis of user behavior in www search. In *Proceedings of the 27th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '04, pages 478–479, New York, NY, USA, 2004. ACM.

Fan Guo, Chao Liu, and Yi Min Wang. Efficient multiple-click models in web search. In *Proceedings of the 2nd ACM International Conference on Web Search and Data Mining*, WSDM '09, pages 124–131, New York, NY, USA, 2009a. ACM.

Jiafeng Guo, Gu Xu, Xueqi Cheng, and Hang Li. Named entity recognition in query. In *Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '09, pages 267–274, New York, NY, USA, 2009b. ACM.

Jiafeng Guo, Xueqi Cheng, Gu Xu, and Huawei Shen. A structured approach to query recommendation with social annotation data. In *Proceedings of the 19th ACM Conference on Information and Knowledge Management*, CIKM '10, pages 619–628, New York, NY, USA, 2010. ACM.

Jiafeng Guo, Xueqi Cheng, Gu Xu, and Xiaofei Zhu. Intent-aware query similarity. In *Proceedings of the 20th ACM Conference on Information and Knowledge Management*, CIKM '11, pages 259–268, New York, NY, USA, 2011. ACM.

Ben He and Iadh Ounis. Finding good feedback documents. In *Proceedings of the 18th ACM Conference on Information and Knowledge Management*, CIKM '09, pages 2011–2014, New York, NY, USA, 2009. ACM.

Jiyin He, Martha Larson, and Maarten de Rijke. Using coherence-based measures to predict query difficulty. In *30th European Conference on Information Retrieval (ECIR 2008)*, number 4956 in LNCS, pages 689–694. Springer, 2008.

Qi He, Daxin Jiang, Zhen Liao, Steven C. H. Hoi, Kuiyu Chang, Ee-Peng Lim, and Hang Li. Web query recommendation via sequential query prediction. In *Proceedings of the 25th International Conference on Data Engineering*, ICDE '09, pages 1443–1454, Washington, DC, USA, 2009. IEEE Computer Society.

Marti A. Hearst. *Search User Interfaces*. Cambridge University Press, 2009.

Katja Hofmann, Shimon Whiteson, and Maarten de Rijke. A probabilistic method for inferring preferences from clicks. In *Proceedings of the 20th ACM Conference on Information and Knowledge Management*, CIKM âĂŹ11. ACM, October 2011.

Katja Hofmann, Anne Schuth, Shimon Whiteson, and Maarten de Rijke. Reusing historical interaction data for faster online learning to rank for IR. In *Proceedings of the Sixth ACM International Conference on Web Search and Data Mining*, WSDM '13, pages 183–192, New York, NY, USA, 2013a. ACM.

Katja Hofmann, Shimon Whiteson, and Maarten Rijke. Balancing exploration and exploitation in listwise and pairwise online learning to rank for information retrieval. *Information Retrieval*, 16(1):63–90, 2013b.

Katja Hofmann, Bhaskar Mitra, Filip Radlinski, and Milad Shokouhi. An eye-tracking study of user interactions with query auto completion. In *Proceedings of the 23rd ACM Conference on Information and Knowledge Management*, CIKM '14, pages 549–558, New York, NY, USA, 2014. ACM.

Charles C. Holt. Forecasting seasonals and trends by exponentially weighted moving averages. *International Journal of Forecasting*, 20(1):5–10, 2004.

Bo-June (Paul) Hsu and Giuseppe Ottaviano. Space-efficient data structures for top-k completion. In *Proceedings of the 22nd International World Wide Web Conference*, WWW '13, pages 583–594, New York, NY, USA, 2013. ACM.

Bouke Huurnink, Laura Hollink, Wietske van den Heuvel, and Maarten de Rijke. Search behavior of media professionals at an audiovisual archive: A transaction log analysis. *Journal of the Association for Information Science and Technology*, 61(6):1180–1197, 2010.

Alpa Jain and Gilad Mishne. Organizing query completions for web search. In *Proceedings of the 19th ACM Conference on Information and Knowledge Management*, CIKM '10, pages 1169–1178, New York, NY, USA, 2010. ACM.

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

Shengyue Ji, Guoliang Li, Chen Li, and Jianhua Feng. Efficient interactive fuzzy keyword search. In *Proceedings of the 18th International World Wide Web Conference*, WWW '09, pages 371–380, New York, NY, USA, 2009. ACM.

Di Jiang, Kenneth Wai-Ting Leung, and Wilfred Ng. Context-aware search personalization with concept preference. In *Proceedings of the 20th ACM Conference on Information and Knowledge Management*, CIKM '11, pages 563–572, New York, NY, USA, 2011. ACM.

Jyun-Yu Jiang, Yen-Yu Ke, Pao-Yu Chien, and Pu-Jen Cheng. Learning user reformulation behavior for query auto-completion. In *Proceedings of the 37th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '14, pages 445–454, New York, NY, USA, 2014. ACM.

Thorsten Joachims, Laura Granka, Bing Pan, Helene Hembrooke, and Geri Gay. Accurately interpreting clickthrough data as implicit feedback. In *Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '05, pages 154–161, New York, NY, USA, 2005. ACM.

Hideo Joho and Joemon M. Jose. A comparative study of the effectiveness of search result presentation on the web. In *Proceedings of the 28th European Conference on Information Retrieval*, ECIR '06, pages 302–313, Berlin, Heidelberg, 2006. Springer-Verlag.

Hideo Joho and Joemon M. Jose. Effectiveness of additional representations for the search result presentation on the web. *Information Processing and Management*, 44(1):226–241, 2008.

Rosie Jones and Kristina Lisa Klinkner. Beyond the session timeout: Automatic hierarchical segmentation of search topics in query logs. In *Proceedings of the 17th ACM Conference on Information and Knowledge Management*, CIKM '08, pages 699–708, New York, NY, USA, 2008. ACM.

Maryam Kamvar and Shumeet Baluja. A large scale study of wireless search patterns: Google mobile search. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '06, pages 701–709, New York, NY, USA, 2006. ACM.

Maryam Kamvar and Shumeet Baluja. The role of context in query input: Using contextual signals to complete queries on mobile devices. In *Proceedings of the 11th International Conference on Human Computer Interaction with Mobile Devices and Services*, MobileHCI '09, pages 405–412, New York, NY, USA, 2009. ACM.

Dimitrios Kastrinakis and Yannis Tzitzikas. Advancing search query autocompletion services with more and better suggestions. In *Proceedings of the 10th International Conference on Web Engineering*, ICWE'10, pages 35–49, Berlin, Heidelberg, 2010. Springer-Verlag.

Makoto P. Kato, Tetsuya Sakai, and Katsumi Tanaka. Structured query suggestion for specialization and parallel movement: Effect on search behaviors. In *Proceedings of the 21st International Conference on World Wide Web*, WWW '12, pages 389–398, New York, NY, USA, 2012. ACM.

Makoto P. Kato, Tetsuya Sakai, and Katsumi Tanaka. When do people use query suggestion? A query suggestion log analysis. *Information Retrieval*, 16(6):725–746, 2013.

Gabriella Kazai, Jaap Kamps, Marijn Koolen, and Natasa Milic-Frayling. Crowdsourcing for book search evaluation: Impact of hit design on comparative system ranking. In *Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '11, pages 205–214, New York, NY, USA, 2011. ACM.

Eugene Kharitonov, Craig Macdonald, Pavel Serdyukov, and Iadh Ounis. User model-based metrics for offline query suggestion evaluation. In *Proceedings of the 36th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '13, pages 633–642, New York, NY, USA, 2013. ACM.

Youngho Kim and W. Bruce Croft. Diversifying query suggestions based on query documents. In *Proceedings of the 37th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '14, pages 891–894, New York, NY, USA, 2014. ACM.

Anagha Kulkarni, Jaime Teevan, Krysta M. Svore, and Susan T. Dumais. Understanding temporal query dynamics. In *Proceedings of the Fourth ACM International Conference on Web Search and Data Mining*, WSDM '11, pages 167–176, New York, NY, USA, 2011. ACM.

Kyung Soon Lee, W. Bruce Croft, and James Allan. A cluster-based resampling method for pseudo-relevance feedback. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '08, pages 235–242, New York, NY, USA, 2008. ACM.

Damien Lefortier, Pavel Serdyukov, and Maarten de Rijke. Online exploration for detecting shifts in fresh intent. In *Proceedings of the 23rd ACM Conference on Information and Knowledge Management*, CIKM '14. ACM, November 2014.

Xin Lei, Andrew Senior, Alexander Gruenstein, and Jeffrey Sorensen. Accurate and compact large vocabulary speech recognition on mobile devices. In *Interspeech 2013*, pages 662–665, 2013.

Guoliang Li, Shengyue Ji, Chen Li, and Jianhua Feng. Efficient fuzzy full-text type-ahead search. *The VLDB Journal*, 20(4):617–640, 2011.

Liangda Li, Hongbo Deng, Anlei Dong, Yi Chang, Hongyuan Zha, and Ricardo Baeza-Yates. Analyzing user's sequential behavior in query auto-completion via Markov processes. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '15, pages 123–132, New York, NY, USA, 2015. ACM.

Xinyi Li, Bob Schijvenaars, and Maarten de Rijke. Investigating search behavior and search failures in academic search. In *Submitted*, 2016.

Yanen Li, Huizhong Duan, and ChengXiang Zhai. A generalized hidden Markov model with discriminative training for query spelling correction. In *Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '12, pages 611–620, New York, NY, USA, 2012. ACM.

Yanen Li, Anlei Dong, Hongning Wang, Hongbo Deng, Yi Chang, and ChengXiang Zhai. A two-dimensional click model for query auto-completion. In *Proceedings of the 37th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '14, pages 455–464, New York, NY, USA, 2014. ACM.

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

Thomas Lin, Patrick Pantel, Michael Gamon, Anitha Kannan, and Ariel Fuxman. Active objects: Actions for entity-centric search. In *Proceedings of the 21st International World Wide Web Conference*, WWW '12, pages 589–598, New York, NY, USA, 2012. ACM.

Chao Liu, Fan Guo, and Christos Faloutsos. Bayesian browsing model: Exact inference of document relevance from petabyte-scale data. *ACM Transactions on Knowledge Discovery from Data*, 4(4):19:1–19:26, 2010a.

Chao Liu, Ryen W. White, and Susan Dumais. Understanding web browsing behaviors through weibull analysis of dwell time. In *Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '10, pages 379–386, New York, NY, USA, 2010b. ACM.

Ning Liu, Jun Yan, Shuicheng Yan, Weiguo Fan, and Zheng Chen. Web query prediction by unifying model. In *Proceedings of the 2008 IEEE International Conference on Data Mining Workshops*, ICDMW '08, pages 437–441, Washington, DC, USA, 2008. IEEE Computer Society.

Shuang Liu, Fang Liu, Clement Yu, and Weiyi Meng. An effective approach to document retrieval via utilizing wordnet and recognizing phrases. In *Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '04, pages 266–272, New York, NY, USA, 2004. ACM.

Tie-Yan Liu. Learning to rank for information retrieval. *Foundations and Trends in Information Retrieval*, 3(3):225–331, 2009.

Yang Liu, Ruihua Song, Yu Chen, Jian-Yun Nie, and Ji-Rong Wen. Adaptive query suggestion for difficult queries. In *Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '12, pages 15–24, New York, NY, USA, 2012. ACM.

Yiqun Liu, Ruihua Song, Min Zhang, Zhicheng Dou, Takehiro Yamamoto, Makoto Kato, Hiroaki Ohshima, and Ke Zhou. Overview of the NTCIR-11 IMine task. In *Proceedings of the 11th NTCIR Conference*, NTCIR '11, Tokyo, Japan, 2014.

Ziyang Liu, Sivaramakrishnan Natarajan, and Yi Chen. Query expansion based on clustered results. *Proceedings of the VLDB Endowment*, 4(6):350–361, 2011.

H. Christopher Longuet-Higgins and Andrew Ortony. The adaptive memorization of sequences. In *Proceedings of the 3rd Annual Machine Intelligence Workshop*, Machine Intelligence 3, pages 311–322. Edinburgh University Press, 1968.

Claudio Lucchese, Salvatore Orlando, Raffaele Perego, Fabrizio Silvestri, and Gabriele Tolomei. Identifying task-based sessions in search engine query logs. In *Proceedings of the 4th ACM International Conference on Web Search and Data Mining*, WSDM '11, pages 277–286, New York, NY, USA, 2011. ACM.

Hao Ma, Haixuan Yang, Irwin King, and Michael R. Lyu. Learning latent semantic relations from clickthrough data for query suggestion. In *Proceedings of the 17th ACM Conference on Information and Knowledge Management*, CIKM '08, pages 709–718, New York, NY, USA, 2008. ACM.

Hao Ma, Michael R. Lyu, and Irwin King. Diversifying query suggestion results. In *Proceedings of the 24th AAAI Conference on Artificial Intelligence*, AAAI '10, pages 1399–1404, New York, NY, USA, 2010. AAAI Press.

Zhongrui Ma, Yu Chen, Ruihua Song, Tetsuya Sakai, Jiaheng Lu, and Ji-Rong Wen. New assessment criteria for query suggestion. In *Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '12, pages 1109–1110, New York, NY, USA, 2012. ACM.

Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. *Introduction to Information Retrieval*. Cambridge University Press, New York, NY, USA, 2008.

Gary Marchionini and Ryen White. Find what you need, understand what you find. *International Journal of Human-Computer Interaction*, 23(3):205–237, 2007.

Dhruv Matani. An O(k log n) algorithm for prefix based ranked autocomplete. `http://www.dhruvbird.com/autocomplete.pdf`, 2011.

Nicolaas Matthijs and Filip Radlinski. Personalizing web search using long term browsing history. In *Proceedings of the Fourth ACM International Conference on Web Search and Data Mining*, WSDM '11, pages 25–34, New York, NY, USA, 2011. ACM.

Qiaozhu Mei, Dengyong Zhou, and Kenneth Church. Query suggestion using hitting time. In *Proceedings of the 17th ACM Conference on Information and Knowledge Management*, CIKM '08, pages 469–478, New York, NY, USA, 2008. ACM.

Edgar Meij, Peter Mika, and Hugo Zaragoza. An evaluation of entity and frequency based query completion methods. In *Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '09, pages 678–679, New York, NY, USA, 2009. ACM.

Edgar Meij, Marc Bron, Laura Hollink, Bouke Huurnink, and Maarten de Rijke. Mapping queries to the linking open data cloud: A case study using DBpedia. *Journal of Web Semantics*, 9(4):418–433, 2011.

Vlachos Michail, Meek Christopher, Vagena Zografoula, and Gunopulos Dimitrios. Identifying similarities periodicities and bursts for online search queries. In *Proceedings of the 2004 ACM SIGMOD International Conference on Management of Data*, SIGMOD '04, pages 131–142, New York, NY, USA, 2004. ACM.

Gilad Mishne and Maarten de Rijke. A study of blog search. In *Proceedings 28th European Conference on Information Retrieval (ECIR 2006)*, number 3936 in LNCS, pages 289–301. Springer, April 2006.

Bhaskar Mitra. Exploring session context using distributed representations of queries and reformulations. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '15, pages 3–12, New York, NY, USA, 2015. ACM.

Bhaskar Mitra and Nick Craswell. Query auto-completion for rare prefixes. In *Proceedings of the 24th ACM Conference on Information and Knowledge Management*, CIKM '15, pages 1755–1758, New York, NY, USA, 2015. ACM.

Bhaskar Mitra, Milad Shokouhi, Filip Radlinski, and Katja Hofmann. On user interactions with query auto-completion. In *Proceedings of the 37th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '14, pages 1055–1058, New York, NY, USA, 2014. ACM.

Meredith Ringel Morris, Jaime Teevan, and Steve Bush. Enhancing collaborative web search with personalization: Groupization, smart splitting, and group hit-highlighting. In *Proceedings of the 2008 ACM Conference on Computer Supported Cooperative Work*, CSCW '08, pages 481–484, New York, NY, USA, 2008. ACM.

Arnab Nandi and H. V. Jagadish. Effective phrase prediction. In *Proceedings of the 33rd International Conference on Very Large Data Bases*, VLDB '07, pages 219–230, Berlin, Heidelberg, 2007. VLDB Endowment.

Harrie Oosterhuis, Anne Schuth, and Maarten de Rijke. Probabilistic multileave gradient descent. In *ECIR 2016: 38th European Conference on Information Retrieval*, LNCS, pages 661–668. Springer, 2016.

Greg Pass, Abdur Chowdhury, and Cayley Torgeson. A picture of search. In *Proceedings of the 1st International Conference on Scalable Information Systems*, InfoScale '06, pages 1–7, New York, NY, USA, 2006. ACM.

Jef Raskin. *The Humane Interface*. Addison-Wesley Professional, 2000.

Ridho Reinanda, Edgar Meij, and Maarten de Rijke. Mining, ranking and recommending entity aspects. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '15, pages 263–272, New York, NY, USA, 2015. ACM.

Matthew Richardson, Ewa Dominowska, and Robert Ragno. Predicting clicks: Estimating the click-through rate for new ads. In *Proceedings of the 16th International World Wide Web Conference*, WWW '07, pages 521–530, New York, NY, USA, 2007. ACM.

Soo Young Rieh and Hong Xie. Analysis of multiple query reformulations on the web: The interactive information retrieval context. *Information Processing and Management*, 42(3):751 – 768, 2006.

Stefan Riezler, Yi Liu, and Alexander Vasserman. Translating queries into snippets for improved query expansion. In *Proceedings of the 22nd International Conference on Computational Linguistics*, COLING '08, pages 737–744, Stroudsburg, PA, USA, 2008. Association for Computational Linguistics.

Patrick Ruch. Using contextual spelling correction to improve retrieval effectiveness in degraded text collections. In *Proceedings of the 19th International Conference on Computational Linguistics*, COLING '02, pages 1–7, Stroudsburg, PA, USA, 2002. Association for Computational Linguistics.

Tetsuya Sakai, Zhicheng Dou, Takehiro Yamamoto, Yiqun Liu, Min Zhang, and Ruihua Song. Overview of the NTCIR-10 INTENT-2 task. In *Proceedings of the 10th NTCIR Conference*, NTCIR '10, pages 94–123, Tokyo, Japan, 2013.

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

Anne Schuth, Harrie Oosterhuis, Shimon Whiteson, and Maarten de Rijke. Multileave gradient descent for fast online learning to rank. In *Proceedings of the Ninth ACM International Conference on Web Search and Data Mining*, WSDM '16, pages 457–466, New York, NY, USA, 2016. ACM.

Xuehua Shen, Bin Tan, and ChengXiang Zhai. Context-sensitive information retrieval using implicit feedback. In *Proceedings of the 28th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '05, pages 43–50, New York, NY, USA, 2005. ACM.

Yelong Shen, Xiaodong He, Jianfeng Gao, Li Deng, and Gregoire Mesnil. Learning semantic representations using convolutional neural networks for web search. In *Proceedings of the 23rd International World Wide Web Conference*, WWW '14, pages 373–374, New York, NY, USA, 2014. ACM.

Milad Shokouhi. Detecting seasonal queries by time-series analysis. In *Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '11, pages 1171–1172, New York, NY, USA, 2011. ACM.

Milad Shokouhi. Learning to personalize query auto-completion. In *Proceedings of the 36th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '13, pages 103–112, New York, NY, USA, 2013. ACM.

Milad Shokouhi and Kira Radinsky. Time-sensitive query auto-completion. In *Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '12, pages 601–610, New York, NY, USA, 2012. ACM.

Milad Shokouhi, Marc Sloan, Paul N. Bennett, Kevyn Collins-Thompson, and Siranush Sarkizova. Query suggestion and data fusion in contextual disambiguation. In *Proceedings of the 24th International World Wide Web Conference*, WWW '15, pages 971–980, New York, NY, USA, 2015. ACM.

Ruihua Song, Min Zhang, Tetsuya Sakai, Makoto P. Kato, Yiqun Liu, Miho Sugimoto, Qinglei Wang, and Naoki Orii. Overview of the NTCIR-9 IMine task. In *Proceedings of the 9th NTCIR Conference*, NTCIR '09, Tokyo, Japan, 2011a.

Yang Song, Dengyong Zhou, and Li-wei He. Post-ranking query suggestion by diversifying search results. In *Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '11, pages 815–824, New York, NY, USA, 2011b. ACM.

David Sontag, Kevyn Collins-Thompson, Paul N. Bennett, Ryen W. White, Susan Dumais, and Bodo Billerbeck. Probabilistic models for personalizing web search. In *Proceedings of the 5th ACM International Conference on Web Search and Data Mining*, WSDM '12, pages 433–442, New York, NY, USA, 2012. ACM.

Alisa Strizhevskaya, Alexey Baytin, Irina Galinskaya, and Pavel Serdyukov. Actualization of query suggestions using query logs. In *Proceedings of the 21st International World Wide Web Conference*, WWW '12, pages 611–612, New York, NY, USA, 2012. ACM.

Xu Sun, Jianfeng Gao, Daniel Micol, and Chris Quirk. Learning phrase-based spelling error models from clickthrough data. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, ACL '10, pages 266–274, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics.

Xu Sun, Anshumali Shrivastava, and Ping Li. Fast multi-task learning for query spelling correction. In *Proceedings of the 21st ACM Conference on Information and Knowledge Management*, CIKM '12, pages 285–294, New York, NY, USA, 2012. ACM.

Hisami Suzuki and Jianfeng Gao. A unified approach to transliteration-based text input with online spelling correction. In *Proceedings of the 2012 Conference on Empirical Methods in Natural Language Processing*, EMNLP '12, pages 609–618, Stroudsburg, PA, USA, 2012. Association for Computational Linguistics.

Andrew Swiffin, John Arnott, J. Adrian Pickering, and Alan Newell. Adaptive and predictive techniques in a communication prosthesis. *Augmentative and Alternative Communication*, 3:181–191, 1987.

Bin Tan, Xuehua Shen, and ChengXiang Zhai. Mining long-term search history to improve search accuracy. In *Proceedings of the 12th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '06, pages 718–723, New York, NY, USA, 2006. ACM.

Jaime Teevan, Daniel J. Liebling, and Gayathri Ravichandran Geetha. Understanding and predicting personal navigation. In *Proceedings of the Fourth ACM International Conference on Web Search and Data Mining*, WSDM '11, pages 85–94, New York, NY, USA, 2011. ACM.

Raghavendra Udupa and Shaishav Kumar. Hashing-based approaches to spelling correction of personal names. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, EMNLP '10, pages 1256–1265, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics.

Yury Ustinovskiy and Pavel Serdyukov. Personalization of web-search using short-term browsing context. In *Proceedings of the 22nd ACM Conference on Information and Knowledge Management*, CIKM '13, pages 1979–1988, New York, NY, USA, 2013. ACM.

David Vallet and Pablo Castells. Personalized diversification of search results. In *Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '12, pages 841–850, New York, NY, USA, 2012. ACM.

Joost van Doorn, Daan Odijk, Diederik Roijers, and Maarten de Rijke. Balancing relevance criteria through multi-objective optimization. In *SIGIR 2016: 39th international ACM SIGIR conference on Research and development in information retrieval*, pages 769–772. ACM, 2016.

Christophe Van Gysel, Leonid Velikovich, Ian McGraw, and Françoise Beaufays. Garbage modeling for on-device speech recognition. In *Interspeech 2015*, September 2015.

Gregg Vanderheiden and David Kelso. Comparative analysis of fixed-vocabulary communication acceleration techniques. *Augmentative and Alternative Communication*, 3:196–206, 1987.

Saúl Vargas, Pablo Castells, and David Vallet. Explicit relevance models in intent-oriented information retrieval diversification. In *Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '12, pages 75–84, New York, NY, USA, 2012. ACM.

Ingmar Weber and Carlos Castillo. The demographics of web search. In *Proceedings of the 35rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '10, pages 523–530, New York, NY, USA, 2010. ACM.

Stewart Whiting and Joemon M. Jose. Recent and robust query auto-completion. In *Proceedings of the 23rd International World Wide Web Conference*, WWW '14, pages 971–982, New York, NY, USA, 2014. ACM.

Stewart Whiting, James McMinn, and Joemon M. Jose. Exploring real-time temporal query auto-completion. In *Proceedings of the 12th Dutch-Belgian Information Retrieval workshop*, DIR '13, pages 1–4, Delft, The Netherlands, 2013. Technical University Delft.

Ian H. Witten and John J. Darragh, editors. *The Reactive Keyboard*. Cambridge University Press, Cambridge, UK, 1992.

Biao Xiang, Daxin Jiang, Jian Pei, Xiaohui Sun, Enhong Chen, and Hang Li. Context-aware ranking in web search. In *Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '10, pages 451–458, New York, NY, USA, 2010. ACM.

Chuan Xiao, Jianbin Qin, Wei Wang, Yoshiharu Ishikawa, Koji Tsuda, and Kunihiko Sadakane. Efficient error-tolerant query autocompletion. *Proceedings of the VLDB Endowment*, 6(6):373–384, 2013.

Takehiro Yamamoto, Yiqun Liu, Min Zhang, Zhicheng Dou, Ke Zhou, Ilya Markov, Makoto Kato, Hiroaki Ohshima, and Sumio Fujita. Overview of the NTCIR-12 IMine-2 task. In *Proceedings of the 12th NTCIR Conference*, NTCIR '12, pages 8–26, Tokyo, Japan, 2016.

Jiang-Ming Yang, Rui Cai, Feng Jing, Shuo Wang, Lei Zhang, and Wei-Ying Ma. Search-based query suggestion. In *Proceedings of the 17th ACM Conference on Information and Knowledge Management*, CIKM '08, pages 1439–1440, New York, NY, USA, 2008. ACM.

Yisong Yue, Rajan Patel, and Hein Roehrig. Beyond position bias: Examining result attractiveness as a source of presentation bias in clickthrough data. In *Proceedings of the 19th International World Wide Web Conference*, WWW '10, pages 1011–1018, New York, NY, USA, 2010. ACM.

Aston Zhang, Amit Goyal, Weize Kong, Hongbo Deng, Anlei Dong, Yi Chang, Carl A. Gunter, and Jiawei Han. adaQAC: Adaptive query auto-completion via implicit negative feedback. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '15, pages 143–152, New York, NY, USA, 2015. ACM.

Yuchen Zhang, Weizhu Chen, Dong Wang, and Qiang Yang. User-click modeling for understanding and predicting search-behavior. In *Proceedings of the 17th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '11, pages 1388–1396, New York, NY, USA, 2011. ACM.

Xiaofei Zhu, Jiafeng Guo, Xueqi Cheng, Pan Du, and Hua-Wei Shen. A unified framework for recommending diverse and relevant queries. In *Proceedings of the 20th International World Wide Web Conference*, WWW '11, pages 37–46, New York, NY, USA, 2011. ACM.

Xiaofei Zhu, Jiafeng Guo, Xueqi Cheng, and Yanyan Lan. More than relevance: High utility query recommendation by mining users' search behaviors. In *Proceedings of the 21st ACM Conference on Information and Knowledge Management*, CIKM '12, pages 1814–1818, New York, NY, USA, 2012. ACM.

Zeyuan Allen Zhu, Weizhu Chen, Tom Minka, Chenguang Zhu, and Zheng Chen. A novel click model and its applications to online advertising. In *Proceedings of the 3rd ACM International Conference on Web Search and Data Mining*, WSDM '10, pages 321–330, New York, NY, USA, 2010. ACM.