DEIM Forum 2018 P5-2

Context-Sensitive Query Auto-Completion with Knowledge Base

Yaobin HU†, Chuan XIAO††, and Yoshiharu ISHIKAWA†††

† Graduate School of Information Science, Nagoya University
Furo-cho, Chikusa-ku, 464-8601 Japan
†† Institute for Advanced Research, Nagoya University
Furo-cho, Chikusa-ku, Nagoya, 464-8601 Japan
††† Graduate School of Informatics, Nagoya University
Furo-cho, Chikusa-ku, Nagoya, 464-8601 Japan
E-mail: †yaobin.hu@db.ss.is.nagoya-u.ac.jp, ††chuanx@nagoya-u.jp, †††ishikawa@i.nagoya-u.ac.jp

Abstract Query auto completion (QAC) is one of the most prominent features of a modern search engine. Current QAC algorithms mostly rank QAC candidates based on their past popularity, for example, frequency. Due to the long-tail distribution of query frequency, the frequency-based QAC algorithms deliver poor prediction results when the frequencies of users' queries are low. In our work, we observe that the high semantic similarity between queries in the same session can be used to improve the prediction quality when the query frequency is very low. We propose a semantic QAC algorithm, which ranks the candidates according to the similarity between context and query. The semantic similarity is computed based on the knowledge base. Experiments on real datasets demonstrate the effectiveness of the proposed algorithm.

Key words query auto-completion, knowledge base, context-awareness

1. Introduction

Query auto-completion (QAC) is a prominent feature provided by many modern search engines. The goal of QAC is to help users formulate his query, in other word, predict the user's intended query. After each character entered by the user in the search box, the search engine get the candidates, which match the prefix, and suggest top-ranked queries to the user. In the ranking step, QAC algorithm can be looked at as an approximate Maximum Likelihood Estimator [1]. The query candidates are ordered according to their expect likelihood. In the absence of any information of the user or other knowledge, the standard QAC approach which estimates the likelihood value approximately by their past popularity (i.e., frequency) is called MostPopularCompletion(MPC) [1]. Although other approaches which rank candidates by their predicted future popularity have been also explored [25].

Popularity based approach is very effective for popular queries. But due to the fact that the query popularity is a long-tail distribution, it does not work well when the queries are not popular. In our experience, the Mean Reciprocal Rank (MRR) of MostPopularCompletion(MPC) is 0.00715 only for the queries of which the frequency less than 25, but occupy 75% of query data. To improve accuracy of QAC, some other aspects was explored in previous research, such as search time [5], [6], [25], [26], context [1], [18], [23], user-specific feature [5], [6], [24] , etc. One of them that was proved can be used to improve QAC result obviously is context [24],a general context include all user's behavior before entering his query, it can be recent queries, recently visited web pages, and recent tweets etc. The context can give more information about user's search intent and can be used to predict the query if available. This is called context-sensitive query auto completion [1]. In many different contexts, we focus on user's recent query which is easily acquired from search log in this paper.

But how can we use the context to predict the user's query? One idea is using the past popularity of query sequences as the likelihood value of query candidates, and clustering similar query together to try to solve sparsity problem of query sequences [8], [19]. But this approach can not work when the popularity of query sequences is very low. Another view is to consider the essential relationship between query and context. Bar-Yossef and Kraus propose a similarity assumption: user's query is likely to be similar to the context queries, as the query and context have the same search intend. They use query recommendation approach (google suggestion) to expand query to be a rich representation and calculate the similarity between query and context, which can be a criterion for ranking [1]. The similarity assumption is a very nice idea, but query recommendation tends to return a high-quality result which usually means high-popularity for the query, and because of this reason, low-popular queries usually have no recommendation. So the query recommendation based similarity also do not work well when the popularity of intend query is very low.

The objective of this study is to improve the prediction quality when the query frequency is very low. Based on the context-sensitive frame and similarity assumption, we propose a new QAC algorithm based on query semantic similarity which work as follow:

Given a user's input x and context C, the algorithm return a top-k competitions of user's input x that are ranked by the semantic similarity with context y. To estimate the semantic similarity between query and context, we handle the query as a set of words, transform words to entities in the knowledge base for estimating similarity between words by their distance in knowledge base, and then generate a feature vector representation based on the word-similarity for query and measure the cosine similarity between feature vectors as the semantic similarity of queries.

The rest of the paper is organized as follows. We first review related work in section 2. Our approach will be described in section 3. In section 4, we report a detail about our empirical study. The paper is concluded in section 5.

2. Related Work

Query auto-completion(QAC) [2], [3] is widely applied in most modern search engines and other information retrieval systems. In the first step, auto-completion system generates candidates, which match the user's input by using information retrieval and NLP techniques [9], [12], [22]. In the second step, matched candidates can be ranked by different criteria. The most common and simple criteria is to use past popularity. Bar-Yossef and Kraus [1] refer to this approach as the MostPopularCompletion (MPC) model:

$$MPC(p) = \arg\max_{q \in C} w(q), w(q) = \frac{f(q)}{\sum_{i \in Q} f(i)} \qquad (1)$$

Where $f(q)$ is the number of occurrences of query q in the query log Q, and C is a set of candidates generated by first step. The original MPC treats every past query at the same. Some research [11], [21], [25], [26] take time information of past query into consideration for improving ranking results, and further more, try to forecast the future query frequencies and use it to replace $f(q)$ in equation 1 [4], [25]. Shokouhi [24] explores the user's personal profile, including user's age, gender, location, etc., try to find user whose search activities is similar to the current user, to adjust the candidates rank.

The context-sensitive query auto completion [1], [8], [16], [18], [23] is an another aspect of this work. The context, in the narrow sense, usually means recent queries in the same session with user's intend query. The NearestCompletion method [1] uses the similarity between context and candidate query rather than past popularity as likelihood value, and expand query based query recommendation for similarity measure. The framework we use in this paper is similar to them. But the NearestCompletion does not work well for the queries of which frequency is very low. Andreas Schmidt [23] uses the distance of entities' occurrences in the document as a relatedness score between context and candidates when searching in a document collection and the query is the name of the entity. Guo et al. [13] propose a two-step approach to rank QAC candidates by learning the user's context. Cao et al. [8] and Liao et al. [19] cluster queries into a smaller set of concepts based on a click graph for query suggestion. They use the past popularity of query which occurrence following same context concept for ranking query candidates. Mitra [20] learns distributed representations of query reformations using deep neural network models, through which context is modeled for QAC tasks.

More general notion of context include all user's behavior before entering the intend query. Li et al. [18] propose a two-dimensional click model for modeling the QAC process after observing the existence of horizontal skipping bias and vertical position bias in the QAC process. Hofmann et al. [14] conduct an in-depth study of user interactions with QAC in web search using eye-tracking and client-side logging, through which they identify a strong position bias towards examining and using top-ranked query completions.

In other aspects, Zhang et al. [27] study implicit negative feedback during user QAC interactions and propose a novel adaptive model that adapts query auto-completion to users' implicit negative feedback towards unselected query candidates. Cai et al [7] focus on reducing the redundancy among query auto-completion candidates.

3. Our Approach

3.1 context-sensitive framework

A search session is a sequence of queries $q_1, q_2 ... q_n$ ($n \geqq 1$, also include the information about query like search time and so on) which own the same intention generated by user. For a user input x which is a prefix of intent query q_i in the session, the context y is the sequence of queries $q_1, q_2 ...... q_i - 1$ before q_i. Since a search session has the same intention, all queries in the session should have same obvious relativity. Of course, in fact, it is impossible to detect every search session completely accurately, since the user may change intention suddenly. Some previous research is about how to detect sessions accurately [10], [15]. In our work, we assume the detector is prefect so that the context is always relevant to intent query.

Our objective is to predict the query which user intends to type after a context by generating a candidate query set which contains k queries by ranking. We call it top-k completions. The context-sensitive query auto-completion (QAC) algorithm accepts a user's input x which is a prefix of a complete query q that the user wants to search, and consists of a few characters, and a context y before q. The output of the context-sensitive QAC algorithm is top-k completions set $C(x)$. If the top-k completions set $C(x)$ contains the query q, the position of query q is the main measure to evaluate how successful the QAC algorithm is.

Most query auto-completion system own a query database, extracted from the query log which records past query information. Each QAC system has its criteria for judging whether a query q is an appropriate completion for user's input x. The most common criteria is prefix-completion that x is a prefix of q, for example, *donald trump* own prefix *don*, so it is a completion for input don. The modern QAC system tends to support other match criteria like mid-completion (for example, *tr*→*donald trump*), and spell-completion (for example, *dt*→*donald trump*). An appropriate completions set of user's input x is denoted by $completion(x)$.

After a $completion(x)$ is filtered, the context-sensitive QAC resorts candidates in the $completion(x)$ according to their relatedness (or similarity) score with context y, and returns the top-k completions set $C(x)$.

3.2 Query similarity

A knowledge base can be seen as a graph consists of entities and relations between entities. One word usually contains some different senses, but in certain situation, the user uses one word with only one sense. That is, one word can correspond kinds of entities in the knowledge base, but only one is right. Considering the fact that there are too many entities a simple word corresponding to, and it is difficult to find which one is right. So we use wordnet instead. WordNet is a lexical database for the language. It groups words (or phrases) into sets of synonyms called synsets, and records a number of relations among these synonym sets. We also can see wordnet as a graph consists of synsets and relations between synsets. Since the inherent corresponding relationship between word and synset, it is easier to find the right sense of the word. We play a Word Sense Disambiguation by using lesk algorithm [17] to find the appropriate sense s_i of the word $word_i$ in the query. Other word-similarity measure technologies also can be used in our approach. The word-similarity definition used in this paper as follow:

$$Sim(word_1, word_2) = \frac{1}{dis(s_1, s_2) + 1} \tag{2}$$

Where $dis(s_1, s_2)$ is the shortest path length from s_1 to s_2. If there is no path from s_1 to s_2, $Sim(word_1, word_2) = 0$.

A query q is processed into a set of terms, dropping the function words and transform content words or phrases in its original form as terms. Every term can be mapped to a *synset* by carrying a Word Sense Disambiguation algorithm which uses the query q as a context. So the original vector presentation of query is $v_q = s_1 ... s_n$ (n is the number of *synsets* in the set). Since the context is the sequence of queries, we can produce context vector from query vector. Formally, if a context is $y = q_1 ... q_t$ and v_{q_i} is the vector form of query q_i, the vector form of y is:

$$v_y = \sum_{i=1}^{i} w_i v_{q_i} \tag{3}$$

Where the weight $w_i \in [0, 1]$ denotes the contribution of q_i to context.

To acquire more semantic information of a term, we use the word-similarity for expanding the original vector form as follow:

Algorithm 1 Produce Expanding Vector

Input: $v_y = s_{y_1} ... s_{y_m}, v_q = s_{q_1} ... s_{q_n}$
Output: V_y, V_q
  $V_y = \emptyset, V_q = \emptyset$
  for each s_{y_i} and s_{q_j} **do**
    $Sim_{ij} = Sim(s_{y_i}, s_{q_j})$
    $C \Leftarrow (s_{y_i}, s_{q_j}, Sim_{ij})$
  end for
  while $C \neq \emptyset$ **do**
    find max Sim_{IJ}
    $V_y \Leftarrow s_{IJ} = Sim_{IJ}, s_{I0} = s_{y_I} - Sim_{IJ}$
    $V_q \Leftarrow s_{IJ} = Sim_{IJ}, s_{0J} = s_{q_J} - Sim_{IJ}$
    delete all (s_{y_I}, s_q, Sim) and (s_y, s_{q_J}, Sim) from C
  end while
  return V_y, V_q

3.3 Hybrid QAC

Our approach is designed for queries of which the frequency is low, and it needs a context. So it can not work alone for non-context queries and don not work well for popular queries compared to MPC. On the other hand, due to the problem that the segmentation of search session may be incorrect, some unrelated context will lead to poor quality result. For expanding our approach to the more general QAC occasion, we propose a Hybrid QAC approach which is combining our approach with MPC.

Given a user input x and a context y, Hybrid QAC accepts two completions set produced by our approach and MPC: SET_{sim} is the result of our approach with the similarity score denoted $simscore()$ and SET_{mpc} is the MPC result with the popularity score denoted $mpcscore()$. By combining the two scores into a hybrid score denoted as $hybscore()$, Hybrid QAC return a top-k completions set SET_hyb which is reranked according to the hybrid score. Because of the different criteria of two approach, we first standardize them as follow:

$$Zscore(q) = \frac{score(q) - \mu}{\sigma} \quad (4)$$

Where μ and σ are the estimated mean and standard deviation. The hybrid score is:

$$hybscore(q) = \alpha Zmpcscore(q) + (1-\alpha) Zsimscore(q) \quad (5)$$

Where $\alpha \in [0, 1]$ is the weight determined by the past popularity of q, that is, the more popular the candidate query is, the more applicable the MPC maybe. When the intend query has no context, $\alpha = 1$.

4. Experiments

4.1 Dataset

In our experiments, we use AOL query log datasets which were sampled between March 1, 2006 and May 31, 2006. We filtered out a large number of queries containing URL substrings (*www., .com, .org, .net, .edu, http*), and Non-English words (including special characters such as &, \$ and #). Moreover, we remove all queries which are too long (longer than 100 characters) or incomprehensible (like *"0 1 2 3 ecg 1"* or *"p; .; p;' p; ' ;' ;';"*). Formally, the incomprehensible query is defined as: The query does not contain English words (longer than 1 character) which can be recognized by WordNet. The query database was constructed from the queries that appear in the AOL log after data cleaning, there are in total 6191999 unique queries in query database. We use a simple standard segmentation (the boundaries are identified by 30 minutes of inactivity) to segment query log into sessions, and merge duplicate queries in the same session. Figure 1 and Figure 2 display the distribution of queries and sessions. As we can see, above 70% queries only occur one time in the query log, and above 49% sessions own more than 1 query. As our approaches rely on a non-empty context and WordNet, we choose a sample randomly consists of 10000 sessions which own more than 1 query and all words in the sample can be recognized by WordNet.

4.2 Evaluation metric and Baseline

We choose the standard MostPopularCompletion as our baseline, the past popularity which we use is the frequency of query in query database. As in past QAC work [1], [5], [25], we use the Mean Reciprocal Rank (MRR) as the standard measure for our approach and baseline. For a intend query q of which prefix is user's input x, if q in the result completions set $C(x)$, the rank of the q is denoted by $rank(q)$. The RR is computed as:

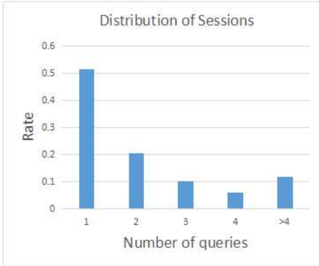

Figure 1 query distribution

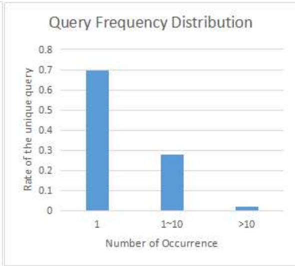

Figure 2 session distribution

$$RR(x) = \begin{cases} \frac{1}{rank(q)}, & if q \in C(x), \\ 0, & else. \end{cases} \quad (6)$$

For a QAC algorithm A,and a sample S, the MRR is computed as:

$$MRR(A) = \frac{1}{|S|} \sum_{q \in S} RR(q) \quad (7)$$

4.3 Result

We compare our approaches to MPC. The details as follow: a) The past popularity of queries is computed by query database after data cleaning. b) The user's input x is the prefix 3 characters of intend query. c) The number of the query used in context is 1. d) The weight parameter $\alpha = 0.5$. Figure 3 provides a comparison of MRR of the three algorithms on the 10000 queries with context. It is very obvious that MPC does not work when intend query owning the low popularity. It is also clear our sim-approach has a good performance for low-popular queries and inferior to MPC for the high-popular queries. The hyb-approach have a good stability.

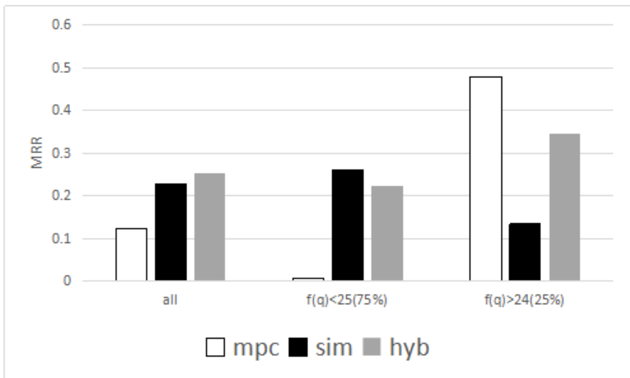

Figure 3 MRR of the 3 algorithms on 10,000 queries owning context. Results are for all queries, for queries (75%) of which the frequency lower than 25, and for queries (25%) of which the frequency higher than 24.

5. Discussion and Future Work

In this paper, we focus on the most challenging query auto completion situation: the long-tail query completion. We propose a new approach to tackle it by estimating the semantic similarity between queries based knowledge base. We show that our approaches work better than the standard MostPopularCompletion approach when the intend query has low popularity. Moreover, we combine our approach with MPC to a hybrid-QAC approach which can work well for all queries. There are a number of possible interesting directions for further development of our techniques: a) We estimate the semantic similarity between queries by mapping query words to entity of knowledge base, the word sense disambiguation problem for knowledge base is key to improve our approach more accurately. b) The session segmentation problem is a crucial point for all context-sensitive auto-completion approach. c) The weight parameter α should be adjustable dynamically with the frequency of query, how to adjust it is our next work.

Acknowledgments

This research was partly supported by KAKENHI (16H01722).

References

[1] Ziv Bar-Yossef and Naama Kraus. Context-sensitive query auto-completion. In *WWW 2011*, pages 107–116. ACM, 2011.

[2] Holger Bast, Debapriyo Majumdar, and Ingmar Weber. Efficient interactive query expansion with complete search. In *CIKM 2007*, pages 857–860. ACM, 2007.

[3] Holger Bast and Ingmar Weber. Type less, find more: fast autocompletion search with a succinct index. In *SIGIR 2006*, pages 364–371. ACM, 2006.

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

[5] Fei Cai, Shangsong Liang, and Maarten De Rijke. Time-sensitive personalized query auto-completion. In *IEEE TKDE*, pages 1599–1608. ACM, 2014.

[6] Fei Cai, Shangsong Liang, and Maarten de Rijke. Prefix-adaptive and time-sensitive personalized query auto completion. *IEEE TKDE*, 28(9):2452–2466, 2016.

[7] Fei Cai, Ridho Reinanda, and Maarten De Rijke. Diversifying query auto-completion. *ACM TOIS*, 34(4):25, 2016.

[8] 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 *KDD 2008*, pages 875–883. ACM, 2008.

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

[10] Carsten Eickhoff, Kevyn Collins-Thompson, Paul N Bennett, and Susan Dumais. Personalizing atypical web search sessions. In *WSDM 2013*, pages 285–294. ACM, 2013.

[11] Nadav Golbandi Golbandi, Liran Katzir Katzir, Yehuda Koren Koren, and Ronny Lempel Lempel. Expediting search trend detection via prediction of query counts. In *WSDM 2013*, pages 295–304. ACM, 2013.

[12] Korinna Grabski and Tobias Scheffer. Sentence completion. In *SIGIR 2004*, pages 433–439. ACM, 2004.

[13] Jiafeng Guo, Xueqi Cheng, Gu Xu, and Xiaofei Zhu. Intent-aware query similarity. In *CIKM 2011*, pages 259–268. ACM, 2011.

[14] Kajta Hofmann, Bhaskar Mitra, Filip Radlinski, and Milad Shokouhi. An eye-tracking study of user interactions with query auto completion. In *CIKM 2014*, pages 549–558. ACM, 2014.

[15] Bernard J Jansen, Amanda Spink, and Vinish Kathuria. How to define searching sessions on web search engines. In *International Workshop on Knowledge Discovery on the Web(WebKDD 2006)*, pages 92–109. Springer, 2006.

[16] Weize Kong, Rui Li, Jie Luo, Aston Zhang, Yi Chang, and James Allan. Predicting search intent based on pre-search context. In *SIGIR 2015*, pages 503–512. ACM, 2015.

[17] Michael Lesk. Automatic sense disambiguation using machine readable dictionaries: how to tell a pine cone from an ice cream cone. In *5th Annual International Conference on Systems Documentation (SIGDOC'86)*, pages 24–26. ACM, 1986.

[18] Yanen Li, Anlei Dong, Hongning Wang, Hongbo Deng, Yi Chang, and ChengXiang Zhai. A two-dimensional click model for query auto-completion. In *SIGIR 2014*, pages 455–464. ACM, 2014.

[19] 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 TIST*, 3(1):17, 2011.

[20] Bhaskar Mitra. Exploring session context using distributed representations of queries and reformulations. In *SIGIR 2015*, pages 3–12. ACM, 2015.

[21] Taiki Miyanishi and Tetsuya Sakai. Time-aware structured query suggestion. In *ACM TIST*, pages 809–812. ACM, 2013.

[22] Arnab Nandi and HV Jagadish. Effective phrase prediction. In *VLDB 2007*, pages 219–230. VLDB Endowment, 2007.

[23] Andreas Schmidt, Johannes Hoffart, Dragan Milchevski, and Gerhard Weikum. Context-sensitive auto-completion for searching with entities and categories. In *SIGIR 2016*, pages 1097–1100. ACM, 2016.

[24] Milad Shokouhi. Learning to personalize query auto-completion. In *SIGIR 2013*, pages 103–112. ACM, 2013.

[25] Milad Shokouhi and Kira Radinsky. Time-sensitive query auto-completion. In *SIGIR 2012*, pages 601–610. ACM, 2012.

[26] Stewart Whiting and Joemon M Jose. Recent and robust query auto-completion. In *WWW 2014*, pages 971–982. ACM, 2014.

[27] 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 *SIGIR 2015*, pages 143–152. ACM, 2015.